A Probing Method for Computing the Diagonal of the Matrix Inverse *

Jok Tang † and Yousef Saad †

April 3, 2010

Abstract

The computation of some entries of a matrix inverse arises in several important applications in practice. This paper presents a probing method for determining the diagonal of the inverse of a sparse matrix in the common situation when its inverse exhibits a decay property, i.e., when many of the entries of the inverse are small. A few simple properties of the inverse suggest a way to determine effective probing vectors based on standard graph theory results. An iterative method is then applied to solve the resulting sequence of linear systems, from which the diagonal of the matrix inverse is extracted. Results of numerical experiments are provided to demonstrate the effectiveness of the probing method.

Keywords: Matrix diagonal extraction, probing, sparse approximate inverses, graph theory, Krylov-subspace methods, Green's functions, covariance matrices.

1 Introduction

Extracting diagonal entries of a matrix inverse is important in many practical applications. Examples include examining inverse covariance matrix as in uncertainty quantification [2], finding a rational approximation for the Fermi-Dirac functions in the density functional theory [38], and evaluating Green's functions in the dynamic mean field theory [17]. The main purpose of this paper is to present an efficient method, called a probing method, for computing the diagonal entries of a matrix inverse.

We consider a large and sparse matrix, $A \in \mathbb{C}^{n\times n}$, that is assumed to be nonsingular and complex-symmetric. Its matrix inverse is denoted by $B := A^{-1}$, and is assumed to exhibit a certain decay property. In this paper, we are particularly interested in $\mathcal{D}(B)$, which is a diagonal matrix whose diagonal entries are the same as those of B.

Extracting $\mathcal{D}(B)$ is considered a challenging task, in part because the problem cannot be easily expressed in the form of a system of equations. The problem is usually harder to solve than a linear system with the same matrix A. So far, not much literature has been devoted to this topic, whereas the related problem of estimating the trace of B received much more attention, see, e.g., [20, 21, 24, 34].

The paper is organized as follows. Section 2 is devoted to present a concise overview of the available literature on the topic of computing $\mathcal{D}(B)$. The main idea of the probing method is given in Section 3. Section 4 is devoted to the analysis of the pattern of B, which is required to make the proposed method efficient. Subsequently, the components of the method are examined in Section 5 and 6. We provide details on the algorithm and other considerations in Section 7. The probing method is illustrated in the numerical experiments of Section 8. Finally, concluding remarks are presented in Section 9.

2 Literature Overview

A straightforward way to determine $\mathcal{D}(B)$ is to compute B explicitly, and extract its diagonal. Using a straightforward approach, the complexity of the method is $\mathcal{O}(n^3)$ if sparsity is not exploited. Exploiting

*Work supported in part by DOE under grant DE-FG 08ER 25841, by NSF under grant OCI-0904587, and by the Minnesota Supercomputer Institute.

†Department of Computer Science and Engineering, University of Minnesota, 200 Union Street S.E, Minneapolis, MN 55455. Email: {jtang, saad}@cs.umn.edu.

sparsity and clever reordering techniques would yield a much more favorable complexity. This approach can be quite expensive for large matrices or matrices arising from a 3-D mesh problem. While the approach can still be attractive for 2-D problems, several alternatives have been developed in the past few decades, which are briefly outlined below.

A direct method based on the LDU decomposition of A can be used to find $\mathcal{D}(B)$, see, e.g., Duff, Erisman, and Reid [15, p. 273]. The sparsity patterns of the unit lower-triangular matrix, L, the diagonal matrix, D, and the unit upper-triangular matrix, U, can be exploited to obtain $\mathcal{D}(B)$ in an economic way. For this, the following expressions due to Takahashi, Fagan, and Chin [41] are exploited:

$$B = U^{-1}D^{-1} + B(I - L); \qquad B = D^{-1}L^{-1} + (I - U)B.$$

These relations enable us to obtain entries of B belonging to the sparsity pattern of L and U without computing any other entries outside this pattern. The approach can be very economical for situations in which the LDU factorization is not costly, as is the case for example for certain matrices arising from 2-D meshes. The method can be demanding in terms of storage and computation for certain types of problems, e.g., those issued from 3-D meshes.

For tridiagonal and banded matrices, many efficient direct algorithms and explicit expressions of the entries of B have been found in the last decades, see the recent paper of Ran and Huang [32] and the references therein. In that paper, an inversion algorithm for a banded matrix is proposed by employing so-called twisted decompositions of matrices. This method is twice as fast as a standard direct method based on the LU decomposition, but both methods need extra fill-in and have a complexity of $\mathcal{O}(pn^2)$, where p denotes the bandwidth of A.

Lower and upper bounds on selected entries of B, and $\mathcal{D}(B)$ in particular, can be found by Gauss-type quadrature formulas. This approach is developed and applied by Golub and collaborators, see, e.g., [1,4,20]. The bounds can be used to construct effective preconditioners for iterative methods.

Several known methods were proposed for approximating matrix inverses, such as those based on the so-called minimum residual iterations (see, e.g., Chow and Saad [10]) and Newton-Schulz iterations method (see, e.g., Householder's book [22]). Both methods converge quadratically for an appropriate initial guess. The oldest method is the Newton-Schulz method, which is based on the iteration

$$B_{j+1} := B_j \left(2I - AB_j\right), \quad j = 0, 1, 2, \ldots.$$

This method is also referred to as the Hotelling-Bodewig algorithm, and became a popular approach in the 1980s with the rise of parallel processing, see, e.g., [31]. In the general case, it is known to converge when $B_0 := \alpha A^T$ where $0 < \alpha < 2/\rho(A^T A)$. This approach was used more recently for extracting $\mathcal{D}(B)$, see [6,26]. In addition, the paper [26] mentions that Newton-Schulz iterations can also be combined with wavelets or hierarchical matrices to compute $\mathcal{D}(B)$. Note that a numerical dropping is usually applied to B_{j+1} to keep the approximated inverse sparse, which may affect the convergence and stability of the method.

The diagonal of a matrix can also be estimated by applying a number of matrix-vector products with the matrix B, using random test vectors [3]. This is a stochastic method, which can be viewed as a Monte Carlo approach. A high accuracy of the solution can only be obtained if a sufficiently large number of vectors is taken, so the method can be expensive for the situation when a high accuracy is required.

Recently, the stochastic estimator [3] was put to work to solve an important problem in uncertainty quantification [2]. Many systems must be solved with the same matrix, A, which can be exploited by iterative solution methods. The paper [2] proposes a number of strategies, including block methods, massively parallel computing, and mixed precision arithmetic. We will not consider stochastic estimators in this paper, because the main applications of interest to us require a high accuracy of $\mathcal{D}(B)$. However, some of the underlying ideas will be exploited to construct a more effective method .

In [3], another approach was proposed to find $\mathcal{D}(B)$, which utilized Hadamard vectors. This approach can be viewed as a variant of probing for banded approximate inverses, but the probing vectors are dense and do not exploit sparsity pattern (apart from bandedness).

Lin *et al.* [27] proposed a direct method for computing $\mathcal{D}(B)$ in a problem arising from electronic structure calculations. Their method adopts a hierarchical decomposition of the computational domain. It first constructs hierarchical Schur complements of the interior points for the blocks of the domain in a bottom-up pass, and then extracts the diagonal entries of $\mathcal{D}(B)$ efficiently in a top-down pass by exploiting the hierarchical local dependence of inverse matrices. Although the practical implementation of the method is

not straightforward, the method can be efficient as it has a complexity of $\mathcal{O}(n^{3/2})$ and $\mathcal{O}(n^2)$ in typical 2-D and 3-D problems, respectively.

Another recent approach is to extract $\mathcal{D}(B)$ using the Lanczos algorithm, see [38]. The Lanczos algorithm can be recast as a simple three-term recurrence,

$$\beta_{j+1}q_{j+1} = Aq_j - \alpha_j q_j - \beta_j q_{j-1}, \quad \beta_0 q_0 = 0, \quad j = 1, \ldots, n-1,$$

where the parameters α_j and β_j are selected such that the Lanczos vector, q_{j+1}, has a unit norm and is orthogonal to its previous two vectors, q_j and q_{j-1}. Then, $\mathcal{D}(B)$ can be updated elegantly by incorporating a few extra steps in the Lanczos algorithm. The overall method is quite similar to the conjugate gradient algorithm, and can be implemented with relative ease. Two drawbacks of the method are that n iterations are usually required to obtain an accurate approximation of $\mathcal{D}(B)$, and the Lanczos algorithm suffers from instability due to a loss of orthogonality among the Lanczos vectors after a certain number of iterations. The reorthogonalization becomes expensive and dominates the overall calculation, when the number of required Lanczos vectors is relatively large.

3 Idea of the Method

In the paper by Hutchinson [24], an unbiased stochastic estimator is described for the trace of a matrix $M \in \mathbb{C}^{n\times n}$. Later, Bekas, Kokiopoulou, and Saad [3] extended this idea to develop an unbiased stochastic estimator for $\delta(M) \in \mathbb{C}^n$, which denotes the diagonal of $\mathcal{D}(M)$. Let $\{v_j\}$ be a sequence of s random vectors. Then, $\delta(M)$ can be estimated by the following vector sequence (see [3, Eq. (3)]):

$$\delta(M) \approx \left[\sum_{j=1}^{s} v_j \odot Mv_j\right] \oslash \left[\sum_{j=1}^{s} v_j \odot v_j\right], \quad s \in \mathbb{N}, \tag{1}$$

where $\odot$ and $\oslash$ represent the element-wise multiplication and division operators of vectors, respectively. The convergence of the stochastic estimator (1) is often very slow.

We now consider an alternative way of expressing (1). Using $V_s := [v_1, v_2, \ldots, v_s]$, we obtain

$$\sum_{j=1}^{s} v_j \odot v_j = \delta\left(V_s V_s^T\right); \tag{2}$$

$$\sum_{j=1}^{s} v_j \odot Mv_j = \delta\left(MV_s V_s^T\right). \tag{3}$$

Combining (1), (2), and (3) yields

$$\delta(M) \approx \delta\left(MV_s V_s^T\right) \oslash \delta\left(V_s V_s^T\right), \tag{4}$$

or, equivalently,

$$\mathcal{D}(M) \approx \mathcal{D}(MV_s V_s^T)\mathcal{D}^{-1}(V_s V_s^T), \tag{5}$$

where $\mathcal{D}^{-1}(V_s V_s^T)$ denotes the inverse of $\mathcal{D}(V_s V_s^T)$. As an example, it can be easily seen from Eq. (5) that in the special case when $s = n$ and V_n is a unitary matrix (i.e., $V_n^T V_n = I$), the approximation would be exact. In fact, general conditions for exactness of the diagonal estimator (1) were shown in [3], and are restated in the following proposition.

Proposition 3.1. *Let $M = [m_{jk}] \in \mathbb{C}^{n\times n}$ and $V_s \in \mathbb{R}^{n\times s}$ be of full rank with $s \le n$. Then,*

$$\mathcal{D}(M) = \mathcal{D}(MV_s V_s^T)\mathcal{D}^{-1}(V_s V_s^T) \tag{6}$$

holds if each j-th row of V_s is orthogonal to all those rows k of V_s for which $m_{jk} \neq 0$.

Eq. (6) can be interpreted as a nonstochastic variant of Eq. (1), in which a specific selection of the sequence $\{v_j\}$ yields a better approximation. Note that columns of V_s that satisfy Proposition 3.1 are not necessarily orthogonal. Furthermore, it is interesting to reprove Proposition 3.1 in light of Eq. (4). Let the k-th row of V_n be denoted by $w_k := V_n^T e_k$. Then,

$$\frac{e_j^T M V_n V_n^T e_j}{e_j^T V_n V_n^T e_j} = \sum_{k:m_{jk}\neq 0} \frac{m_{jk}(w_k, w_j)}{(w_k, w_j)},$$

which is equal to m_{jj} under the stated condition that $(w_k, w_j) = 0$ for all $k \neq j$ such that $m_{jk} \neq 0$. Since this is valid for all j, we obtain Eq. (6).

When we take $M := B$, Proposition 3.1 suggests that if the sparsity pattern of B is known, then we are able to select the s columns of V_s in such a way that $\mathcal{D}(B)$ has no contributions from off-diagonal entries of B. This is clearly not possible when B is dense, unless one uses a full unitary matrix V_n. However, B is often well-approximated by a sparse matrix (Section 4), and, in this case, we can seek V_s with a smallest possible s such that the approximation is exact for the so-called sparsified B (Section 5). When V_s is found, we form $BV_s \in \mathbb{C}^{n\times s}$ by solving a sequence of linear systems (Section 6). Finally, $\mathcal{D}(B)$ can be obtained via Eq. (6).

4 Pattern of the Sparsified Inverse

Although the matrix A is sparse, its inverse, B, is usually dense. Fortunately, B often has many small entries which can be neglected. If we define $E := \mathcal{D}(A) - A$ and $F := \mathcal{D}^{-1}(A) - A$ by assuming that the diagonal of A is full, then we have

$$A = \mathcal{D}(A)(I - F),$$

from which we obtain the Neumann expansion

$$B \approx \underbrace{\left(I + F + F^2 + \cdots + F^p\right)\mathcal{D}^{-1}(A)}_{=:B^{(p)}}. \tag{7}$$

If A is diagonally dominant, $\|F^k\|$ decreases rapidly for a larger k. In this case, B is well-approximated by $B^{(p)}$ for a relatively small p. If the number of nonzeros of each row of A is bounded by a small constant, A^p and F^p are sparse, and, hence, $B^{(p)}$ is sparse as well. These ideas were indeed exploited in the development of approximate inverse preconditioners, see, e.g., Chow [8, 9] and Huckle [23]. In [8, 9], approximate inverse preconditioners were proposed based on the pattern of A^p for a small p and dropping small entries of this matrix. The resulting procedure was shown to be efficient and quite attractive for some problems.

Diagonal dominance of A is a sufficient but not a necessary condition for $B^{(p)}$ to be a good approximation of B. It is known that for a Laplacian matrix A, the entries of B decay away from the diagonal, whereas Laplacian matrices are weakly diagonally dominant. Demko, Moss, and Smith [14] showed that the same result is true for any banded positive-definite matrix A. In that paper, it is proved that

$$|b_{jk}| \leq \gamma\lambda^{|j-k|}, \quad \text{with } \gamma > 0,\ 0 < \lambda < 1, \text{ and } 1 \leq j, k \leq n. \tag{8}$$

In (8), the parameters γ and λ usually depend on the bandwidth and extreme eigenvalues of A, with a decay being faster when the bandwidth is small and the matrix is well-conditioned. Such an exponential decay rate of entries of the inverse, as given in (8), can also be observed for several other types of matrices. More details and analysis on this topic can be found in, e.g., [4, 5, 16, 29, 30, 42] and the references therein.

As noted in, e.g., [42], the theoretical upper bound of $|b_{jk}|$, as given in (8), is not always sharp. In the extreme case, (8) holds where γ is relatively large, while no actual decay trend can be observed along the diagonals of B. In addition, entries of some matrix inverses decay in an oscillatory pattern. This can also be described by (8) by replacing the distance $|j - k|$ by the physical distance $d(i, j)$, where $d(i, j)$ is defined as the length of the shortest path connecting vertex i with j based on the underlying graph associated with A. The actual decay property of the entries of B is often better observed with respect to $d(i, j)$. In this case, the sparsity pattern of B can still be approximated by that of A^p for a certain p, since the underlying graph associated with A^p shows the paths of length at most p of the original graph of A, and, hence, preserve the physical distances between vertices.

The above discussion can be further formalized as follows. Consider for $\epsilon > 0$ the sparsified matrix $B_\epsilon \in \mathbb{C}^{n\times n}$, defined by

$$(B_\epsilon)_{ij} = \begin{cases} b_{ij}, & \text{if } |b_{ij}| \geq \epsilon; \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

Then, the sparsity pattern of B_ϵ, denoted by $\mathcal{S}(B_\epsilon)$, can often be approximated by the sparsity pattern of a power of A, $\mathcal{S}(A^p)$. When the entries of B show a strong decay property, then p could be small and $B \approx B_\epsilon$ holds.

Although the exact p depends on the specific A, it is generally complicated to relate p solely to the structure of A. Some additional information of B is usually required to find an appropriate approximation of p. As (8) suggests that the decay rate of entries of B is quite similar for each column, it often suffices to examine the pattern of one column of B. This column, say x_j, can be retrieved by solving the linear system

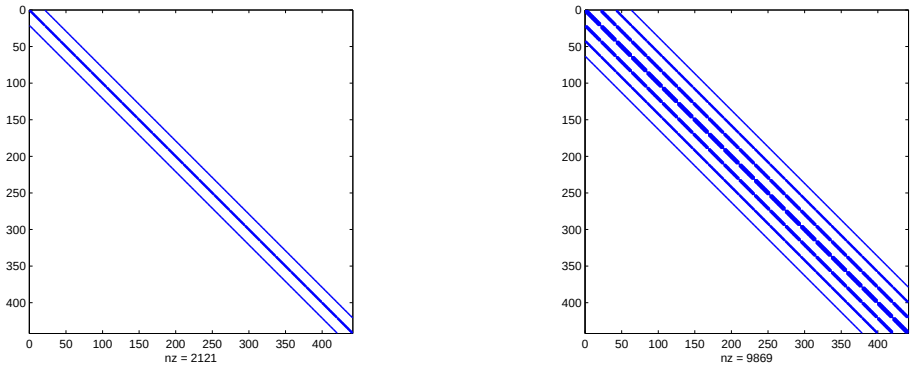
$$Ax_j = e_j, \quad 1 \leq j \leq n. \tag{10}$$

The decay property of the vector x_j can then be analyzed to determine a good value for p.

Example 4.1. *A typical sparsity pattern of a symmetric and positive-definite matrix A and its approximated inverse, B_ϵ, is presented in Figure 3. Here, the parameter p can be estimated by*

$$p = \min\{l : |x_1^{(l)}| < \epsilon\} - 2,$$

where $x_1^{(l)}$ is the l-th entry of the first column of B. For $p = 3$, we obtain that $\mathcal{S}(B_\epsilon)$ is approximated by $\mathcal{S}\left(A^3\right)$. Moreover, the decay behavior of the entries of B can be illustrated by representing the entries of some of its columns in 3-D plots, see Figure 2. From this figure, we can observe a similar decay behavior for different columns.

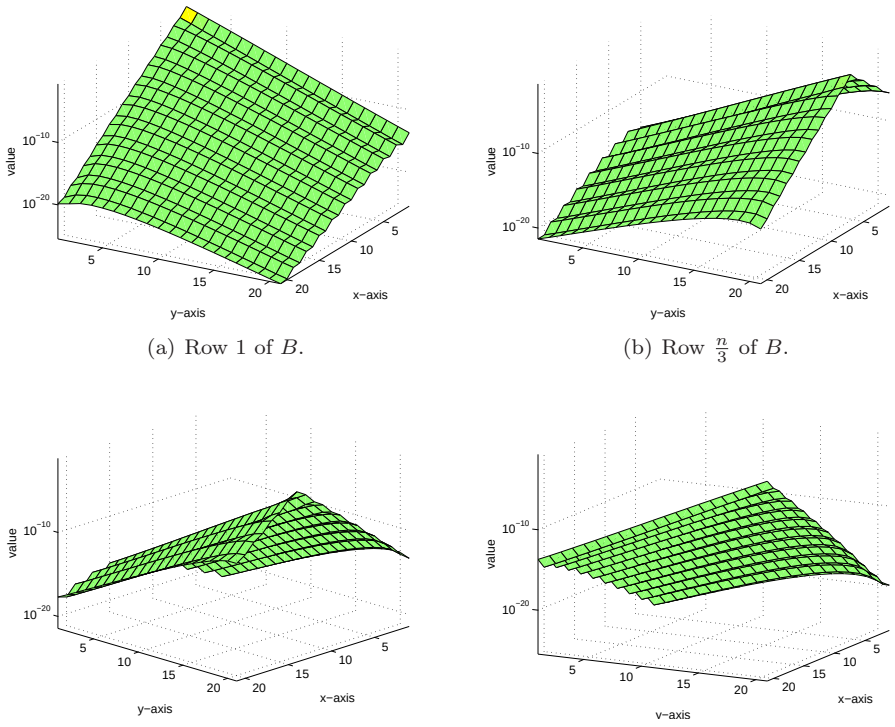


(a) $\mathcal{S}(A)$. (b) $\mathcal{S}(B_\epsilon) \approx \mathcal{S}\left(A^3\right)$.

Figure 1: A typical sparsity structure of a matrix A and its approximated inverse B_ϵ for a small ϵ.

5 Probing

Once the pattern of the sparsified inverse, B_ϵ, is determined, we need to find a matrix V_s of probing vectors which satisfy Proposition 3.1. A first approach that comes to mind is to exploit ideas related to standard probing techniques, which are based on graph coloring. A form of graph coloring has been originally exploited to evaluate sparse Jacobian and Hessian matrices, see, e.g., [11–13]. Probing techniques were also applied for constructing effective interface preconditioners in domain decomposition algorithms [7], and for approximating Schur complements for saddle-point problems [39]. These approaches seek to approximate a specific target matrix by means of relatively small set of so-called probing vectors. Multiplying this matrix by each probing vector allows to compute a large number of columns of the matrix simultaneously. In the end, the whole target matrix can be obtained by performing matrix-vector products with all probing vectors.



(a) Row 1 of B.

(b) Row $\frac{n}{3}$ of B.

(c) Row $\frac{2n}{3}$ of B.

(d) Row n of B.

Figure 2: 3-D representation of some columns of the matrix inverse, B, based on a typical diagonally dominant A.

Our situation is somewhat different from the original problems in which probing is used, because we are not interested in evaluating all entries of the target matrix, B_ϵ in our case, but only those in its diagonal. With this in mind, the probing method used in this paper is different from standard probing. As it turns out, to find a suitable set of probing vectors, we only need to color the adjacency graph associated with B_ϵ with a standard graph coloring, which requires that no two adjacent vertices have the same color. Ideally, we want to color the graph with the smallest number of colors, but this is known to be an NP-hard problem [28]. As is done in common practice, we rely on heuristics to find a coloring with a small number of colors. Algorithm 1 presents a well-known greedy algorithm for this task. The quality of the resulting coloring depends on the chosen ordering. More details and analysis on graph coloring algorithms can be found in [19, 25, 35].

Algorithm 1: Greedy Multicoloring Algorithm

input : Adjacency graph corresponding to an $n \times n$ matrix
output: Colors of the vertices of the graph

for $j \leftarrow 1$ **to** n **do**
  Set Color$(j) = 0$
for $j \leftarrow 1$ **to** n **do**
  Set Color$(j) = \min\{k > 0 \mid k \neq \text{Color}(l)\ \forall l \in \text{Adjacent}(j)\}$

After coloring the vertices of the adjacency graph associated with B_ϵ, the probing matrix, V_s can be constructed as follows:

$$(V_s)_{jk} = \begin{cases} 1, & \text{Color}(j) = k; \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

As a consequence, $s = \max\{k\}$ and each row of V_s based on (11) consists of exactly one nonzero entry, yielding $\mathcal{D}(V_s V_s^T) = \mathcal{D}^{-1}(V_s V_s^T) = I$. In addition, this choice of V_s satisfies the conditions of Proposition 3.1

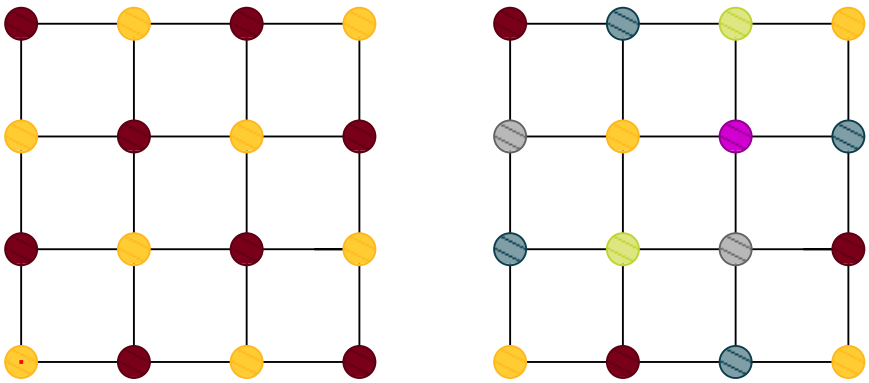
as is stated next.

Proposition 5.1. *Suppose that the vertices of the adjacency graph corresponding to $B_\epsilon = [b_{jk}]$ are colored by Algorithm 1. Let V_s be constructed by Eq. (11). Then, each j-th row of V_s is orthogonal to all those rows k of V_s for which $b_{jk} \neq 0$.*

Proof. by definition, $\text{Color}(j) \neq \text{Color}(k)$ for each $b_{jk} \neq 0$. Moreover, the j-th and k-th rows of V_s consist of zeros except for the $\text{Color}(j)$-th and $\text{Color}(k)$-th entry, respectively. Consequently, the two rows are orthogonal, and the theorem follows immediately. □

From Proposition 5.1, we conclude that V_s based on (11), where the colors are associated with a vertex coloring of the adjacency graph, can be applied to compute $\mathcal{D}(B_\epsilon)$. This is further illustrated in Example 5.1.

Example 5.1. *Let $A \in \mathbb{R}^{16\times 16}$ be the discretized matrix of a 2-D Laplacian on a uniform Cartesian grid in which the entries are ordered lexicographically. Suppose that coloring of the vertices of the adjacency graph corresponding to $\mathcal{S}(B_\epsilon) = \mathcal{S}(A)$ and $\mathcal{S}(B_\epsilon) = \mathcal{S}(A^2)$ is performed by Algorithm 1. Then, the results are presented in Figure 3. Note that we do not form the adjacency graph of A^2 explicitly, as it is equivalent to the graph whose edges represent paths of length at most 2 in the original graph.*



(a) $p = 1$ (two colors). (b) $p = 2$ (six colors).

Figure 3: Coloring the vertices of the adjacency graph of A^p for $n = 4^2$.

Using Eq. (11), the probing matrices are given by

$$V_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ \vdots & \vdots \\ 0 & 1 \end{bmatrix}, \quad V_6 = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & & \vdots \\ 1 & 0 & 0 & \cdots & 0 \end{bmatrix}.$$

Note that $\mathcal{D}(V_s V_s^T) = \mathcal{D}^{-1}(V_s V_s^T) = I$ holds for both $s = 2, 6$. For $\mathcal{S}(B_\epsilon) = \mathcal{S}(A)$ or $\mathcal{S}(B_\epsilon) = \mathcal{S}(A^2)$, we obtain

$$B_\epsilon V_2 = \begin{bmatrix} b_{11} & * \\ * & b_{22} \\ b_{33} & * \\ * & b_{44} \\ \vdots & \vdots \\ b_{nn} & * \end{bmatrix}, \quad B_\epsilon V_6 = \begin{bmatrix} b_{11} & * & * & \cdots & * \\ * & b_{22} & * & \cdots & * \\ * & * & b_{33} & \cdots & * \\ b_{44} & * & * & \cdots & * \\ \vdots & \vdots & & & \vdots \\ b_{nn} & * & * & \cdots & * \end{bmatrix},$$

*respectively, where * denotes an irrelevant entry. Consequently, for the specific s, $\mathcal{D}(B_\epsilon) = \mathcal{D}(B_\epsilon V_s V_s^T)$ follows from Proposition 3.1.*

As was already noted, we stress that the probing technique presented here differs from the standard probing technique known in the literature. In the standard probing technique, vertices j and k are allowed to have the same color if the j-th and k-th columns of B_ϵ are structurally orthogonal, i.e., they do not have nonzero entries in the same row positions. This corresponds to coloring the adjacency graph associated with B_ϵ^2 instead of B_ϵ, which may require significantly more colors. When the probing matrix is then formed by (11), it is clear that this matrix also satisfies the requirement of Proposition 3.1. In fact, the corresponding probing vectors can be used to determine all nonzero entries of B_ϵ, not just its diagonal entries. As we are only interested in $\mathcal{D}(B_\epsilon)$, the standard probing technique would yield many more probing vectors than required.

6 Sequence of Linear Systems

The previous section described a procedure for finding the probing matrix, V_s, corresponding to a given A. Based on this V_s, we can now compute $X_s := B_\epsilon V_s$. If $X_s := [x_1, x_2, \ldots, x_s]$ and $V_s := [v_1, v_2, \ldots, v_s]$, the columns of X_s can be obtained by solving the linear systems

$$Ax_j = v_j, \quad j = 1, 2, \ldots, s. \tag{12}$$

By assuming that $\mathcal{D}^{-1}(V_s V_s^T) = I$, Proposition 3.1 yields $\mathcal{D}(B_\epsilon) := \mathcal{D}(X_s V_s^T)$. When $s \ll n$, solving Eq. (12) is obviously much less expensive than solving the full sequence of n linear systems to obtain the inverse by a straightforward use of equations.

A sequence of linear systems, such as (12), can be solved by a direct or iterative method. For large dimensions of A, Krylov-subspace methods are among the most popular methods in use. We apply GMRES [36] to solve Eq. (12) in our experiments, but any other Krylov-subspace methods could be used instead, such as QMR [18], BiCGSTAB [43], IDR(s) [40], or one of their variants. The method of choice usually depends on the dimensions and the condition of the matrix and the available memory storage, among others.

If A is not (nearly) diagonally dominant, the convergence of Krylov-subspace methods may deteriorate. To remedy this, it is common practice to incorporate a preconditioner into the method, i.e., we solve

$$M^{-1}Ax_j = M^{-1}v_j, \quad j = 1, 2, \ldots, s, \tag{13}$$

instead of Eq. (12), where $M \in \mathbb{C}^{n \times n}$ denotes a preconditioner, see, e.g., [35] for details. The preconditioner of choice depends on the dimensions and the condition of the matrix and the available memory storage, but it is also influenced by the number of probing vectors, s, and the parallel implementation.

7 Implementation and Further Considerations

In the previous sections, the components of the probing method are discussed with the aim of computing $\mathcal{D}(B)$. The resulting algorithm of the method is presented in Algorithm 2.

The bulk of the computations in Algorithm 2 usually involves solving the linear systems in Lines 2 and 7. The exact complexity of each solve depends on the choice of the Krylov-subspace method and its preconditioner, the sparsity pattern of A, and the number of required iterations for convergence, among other things. In the best case, solving each linear system has a complexity of $\mathcal{O}(\gamma n)$, where $\gamma > 0$ is related to the number of required iterations. The whole sequence of linear systems in Line 7 can then be solved with $\mathcal{O}(\gamma n s)$ flops.

The pattern $\mathcal{S}(A^p)$ depends on the decay property of B, and can be determined efficiently. If m denotes the maximum degree of each vertex of the underlying graph associated with A, then computing $\mathcal{S}(A^p)$ requires at most $m^p n$ operations. Therefore, the cost is linear in n with a prefactor m^p that can be large.

Moreover, it is not necessary to store the full matrix X_s explicitly, as X_s is only used to construct $\mathcal{D}\left(X_s V_s^T\right)$. As V_s consists of just n nonzero entries, $\mathcal{D}\left(X_s V_s^T\right)$ can be determined efficiently as it were an inner product of two vectors, requiring $\mathcal{O}(n)$ flops.

Algorithm 2 has low memory requirements. In addition to the required memory storage for the specific Krylov-subspace method and its preconditioner, we only have to store the colors of the n vertices, the n nonzero entries of V_s, and the entries of X_s involved in the computation of $\mathcal{D}\left(X_s V_s^T\right)$.

Algorithm 2: Probing Method for Computing $\mathcal{D}(B)$

input : $A \in \mathbb{C}^{n\times n}$ that is sparse, nonsingular, and complex-symmetric
output: $\mathcal{D}(B) \in \mathbb{C}^{n\times n}$

1 Select ϵ
2 Solve $Ax_j = e_j$ for a given j by a preconditioned Krylov-subspace method
3 Determine p from x_j
4 Color the vertices in the adjacency graph associated with $\mathcal{S}(A^p)$
5 Construct $V_s := [v_1, v_2, \ldots, v_s]$ from Eq. (11)
6 **for** $j \leftarrow 1$ **to** s **do**
7 Solve $Ax_j = v_j$ by a preconditioned Krylov-subspace method
8 Construct $X_s := [x_1, x_2, \ldots, x_s]$
9 Compute $\mathcal{D}(B_\epsilon) := \mathcal{D}\left(X_s V_s^T\right)$
10 Set $\mathcal{D}(B) := \mathcal{D}(B_\epsilon)$

The probing method can be parallelized in several ways. The most obvious way is to solve the linear systems in the sequence (Line 6–7) in parallel, where each processor solves one or more linear systems. This would significantly reduce the computing time of the probing method, as solving the sequence of linear systems is computationally most demanding in the algorithm. Another option is to split each matrix and vector across the available processors, so that each matrix or vector operation can be performed by distributing the work over the processors. Based on this architecture, solving a linear system can be readily parallelized (see, e.g., [35]), $\mathcal{S}(A^p)$ can be computed efficiently, parallel graph coloring algorithms can be found in [19], and the construction of and computations with V_s can be easily parallelized.

We note that decreasing ϵ in B_ϵ generally leads to a larger p for $\mathcal{S}(A^p)$, more required colors for the graph coloring, and a larger s in V_s. It will also result in a better approximation for $\mathcal{D}(B)$. However, it is usually not possible to verify if $\mathcal{D}(B_\epsilon)$ for a given ϵ is a sufficiently accurate approximation of $\mathcal{D}(B)$. Ideally, we want to evaluate $||I - B_\epsilon A||$, but this is obviously not practical, as most off-diagonal entries of B_ϵ are not involved in Algorithm 2. This is, however, not a drawback that is unique to the probing method, as this issue appears in other methods for computing $\mathcal{D}(B)$. If accuracy plays a critical role, a (rather expensive) way to evaluate the accuracy of $\mathcal{D}(B_\epsilon)$ is by applying Algorithm 2 twice, where in the second run Lines 1–3 could be replaced by the line 'Take p from the first run, and set $p := p + 1$'.

8 Numerical Experiments

To illustrate the probing method for computing $\mathcal{D}(B)$, we perform numerical experiments for two different applications: covariance matrices used for uncertainty quantification and imaginary Green's functions used in the dynamic mean field theory. The code is written in Fortran 90, and the computations are performed on a sequential LINUX machine (Dual-Core 2.6 GHz AMD Opteron processor with 8GB of memory).

The performance of the probing method is compared to a Krylov-subspace method that solves the full sequence of n linear systems. The latter method is denoted by Full Krylov. In both approaches, the Krylov-subspace method is GMRES [36] with a diagonal scaling as preconditioner. We choose the relative termination tolerance of GMRES as 10^{-8}. For the probing method, we take $\epsilon = 10^{-10}$. Moreover, the quantities that are measured in the experiments are given in Table 1.

8.1 Uncertainty Quantification: Covariance Matrices

Uncertainty quantification (UQ) in risk analysis has become a key issue in, e.g., geology, signal processing, and portfolio management. In this setting, inverse covariance matrices hold a central role. A crucial question in data analysis for risk management is the degree of confidence that we can have in the quality of data. A highly useful measure of this quality is provided by the diagonal entries of the inverse covariance matrix, see also [2] and the references therein.

In this subsection, $A = [a_{ij}]$ denotes a covariance matrix with entries computed via a decaying covariance function, see [37, Eq. (28)] and [33, Chapter 4.2]. This positive-definite function is based on a uniform 2-D

Quantity	Explanation
CPU	Computing time in seconds
Iter	Average number of required GMRES iterations for each solve
p	Approximate matrix power satisfying $\mathcal{S}(B_\epsilon) \approx \mathcal{S}(A^p)$
s	Number of probing vectors
Error	Difference of $\mathcal{D}(B)$ between full Krylov and probing method
RDD	Percentage of rows that are diagonally dominant
NNZ	Number of nonzero entries in A

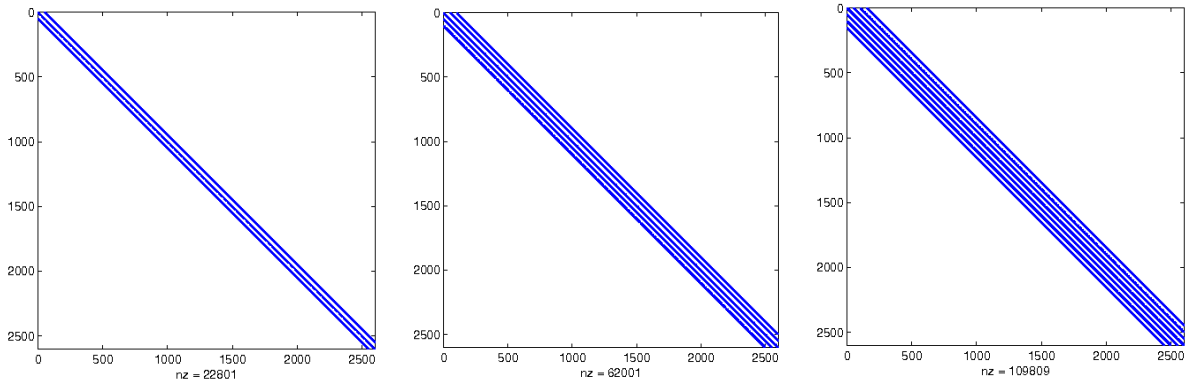
Table 1: Quantities used in the numerical experiments.

grid with $\sqrt{n}$ grid points in each direction, the Euclidean distance between grid points i and j (denoted by $d(i,j)$), the compact threshold $\alpha > 0$, and the function smoothness $\beta > 0$, i.e.,

$$a_{i,j} := \begin{cases} \left(1 - \frac{d(i,j)}{\alpha}\right)^\beta, & \text{if } d(i,j) \leq \alpha; \\ 0, & \text{otherwise.} \end{cases}$$

The resulting matrix A is sparse, as the covariance between points is zero when their distance exceeds α.

Typical sparsity patterns of A can be found in Figure 4. We aim at computing $\mathcal{D}(B)$ based on A for various values of n, α, and β.



(a) $\alpha = 2$ (NNZ = 9,801). (b) $\alpha = 3$ (NNZ = 62,001). (c) $\alpha = 4$ (NNZ = 159,201).

Figure 4: Typical sparsity patterns of A for $n = 51^2$.

8.1.1 Results for various values of n

We first start with fixing the parameter set $(\alpha, \beta) = (3, 5)$, and varying the grid size, n. The results of this experiment can be found in Table 2.

Increasing n hardly changes the condition of matrix A and the structure of B, so that the number of iterations and probing vectors is the same for all n. Therefore, for larger n, the probing method obviously becomes superior to the full Krylov method by considering the computational cost. Moreover, we observe from Table 2 that the solutions are sufficiently accurate.

8.1.2 Results for various values of α and β

In the next test run, we fix the grid to $n = 51^2$, and vary (α, β). The results can be found in Table 3.

From Table 3, we notice that the probing method requires significantly less computing time compared with the full Krylov method. The difference in performance becomes larger when GMRES requires more iterations to converge. Note that the probing method also works appropriately for problems in which A is not diagonally dominant, although the number of probing vectors can be large for these cases.

Method	Quantity	n		
		21^2	51^2	81^2
Full Krylov	CPU	2	77	518
	Iter	13	13	13
Probing	CPU	3	12	29
	Iter	13	13	13
	p	7	7	7
	s	289	289	289
	Error	5.1E-9	1.2E-8	1.7E-8
	RDD	100%	100%	100%
	NNZ	9,801	62,001	159,201

Table 2: Results of the UQ experiment for $(\alpha, \beta) = (3, 5)$ and various grid size.

Method	Quantity	(α, β)				
		(2, 4)	(3, 5)	(3, 3)	(4, 5)	(3, 4)
Full Krylov	CPU	29	77	174	187	108
	Iter	9	13	24	20	17
Probing	CPU	2	12	34	81	19
	Iter	9	13	24	20	17
	p	5	7	9	9	8
	s	49	289	441	999	361
	Error	7.2E-7	1.2E-8	1.8E-8	1.4E-8	1.8E-8
	RDD	100%	100%	15%	15%	7.6%
	NNZ	22,801	62,001	62,001	109,809	62,001

Table 3: Results of the UQ experiment for $n = 51^2$ and various values of α and β.

8.2 Dynamic Mean Field Theory: Imaginary Time Green's Function

Dynamic mean field theory (DMFT) has recently emerged as a very powerful and reliable tool in physics for the investigation of lattice models of correlated electrons in a quantum many-body system, see [17] and the references therein. For the inhomogeneous DMFT method, the computation of the diagonal of the inverse of a large collection of sparse matrices is required. These inverses correspond to problems corresponding to the real and imaginary time Green's functions, respectively. Both problems are related by the chemical potential, μ, and are solved for different frequencies, ω. A typical form of a matrix A is

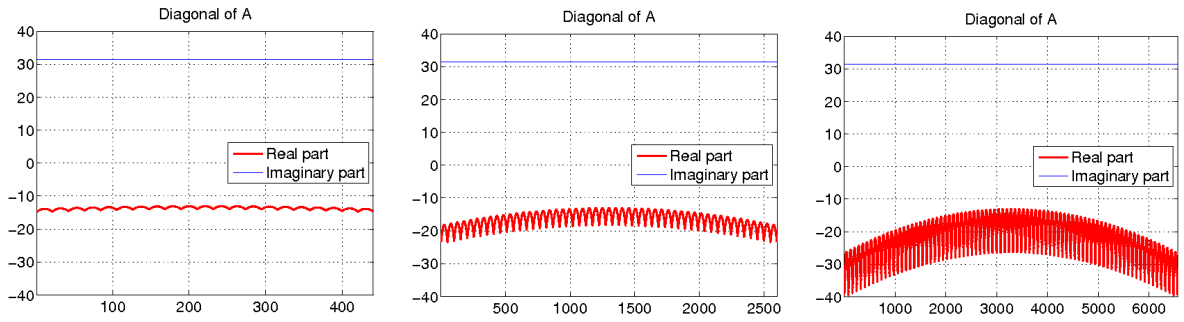
$$A := H + D, \quad H \in \{0,1\}^{n\times n},\ D \in \mathbb{C}^{n\times n}. \tag{14}$$

The matrix H is known as a 2-D hopping matrix, which has a similar sparsity pattern as a standard 2-D Laplacian matrix. The entries of the main diagonal of H are zero, and the entries of the four nonzero off-diagonals are equal to one. The matrix D is a sum of various diagonal matrices, and varies for each frequency. In the specific case of the imaginary time Green's function, we have

$$D = (\mu + i\omega)I - V - \Sigma, \quad \mu, \omega \in \mathbb{R},\ V \in \mathbb{R}^{n\times n},\ \Sigma \in \mathbb{C}^{n\times n}. \tag{15}$$

where V and Σ represent the trap potential and local self-energy, respectively. The form of matrix D associated with the real time Green's function is different, and is not further considered here. A detailed explanation and analysis can be found in [17].

In this subsection, we perform experiments for a fixed set of parameters (Bose level is 19.36 eV and temperature is $T = 10$ K) and a fixed frequency $\omega = \pi T$. The exact expression of μ, V and Σ follows from the underlying DMFT theory and simulation, and the resulting diagonal of D, and, therefore, the resulting diagonal of A is presented in Figure 5 for various values of n. Hence, the matrix A associated with the imaginary time Green's function is complex-symmetric and diagonally dominant.



(a) $n = 21^2$. (b) $n = 51^2$. (c) $n = 81^2$.

Figure 5: The real and imaginary parts of $\mathcal{D}(A)$ in the DMFT experiment for various values of n.

8.2.1 Results for various values of n

In the first test run, we compute $\mathcal{D}(B)$ for the matrix A based on Figure 5. The results of this experiment can be found in Table 4.

Increasing n does not significantly change the condition of matrix A and the structure of B, implying that the number of iterations and probing vectors does not vary much for different n. Consequently, the probing method outperforms the full Krylov method in terms of computing time, as the number of linear systems to be solved for the Full Krylov method obviously grows for increasing n, whereas s does not differ much for various values of n.

Method	Quantity	n		
		21^2	51^2	81^2
Full Krylov	CPU	0.6	20	130
	Iter	7	7	6
Probing	CPU	1	1	1
	Iter	7	7	7
	p	4	4	3
	s	24	24	18
	Error	2.3E-8	5.5E-8	6.3E-7
	RDD	100%	100%	100%
	NNZ	2,121	12,801	32,481

Table 4: Results of the DMFT experiment for various values of n.

8.2.2 Results for various scalings of $\mathcal{D}(A)$

In the next test run, we take the same setting as above, but we now fix the grid to $n = 51^2$ and scale $\mathcal{D}(A)$ with a parameter $0.1 \leq \sigma \leq 1$. We note that choosing $\sigma < 0.1$ leads to an indefinite matrix A for which the maximum number of probing vectors (i.e., $s = n$) would be required. A scaling of $\mathcal{D}(A)$ can be associated with a different DMFT setting in which the condition of the resulting matrix A is varied. The results of the experiment can be found in Table 5.

As can be observed in Table 5, choosing a smaller σ leads to an increased number of both the probing vectors and the GMRES iterations, while the accuracy of the solution becomes worse. The accuracy could be increased by choosing a more stringent ϵ, but this seems to be only required for an A that is not diagonally dominant. For a diagonally dominant A, the probing method is superior to the full Krylov method, while the accuracy is comparable.

Method	Quantity	σ		
		1	0.5	0.1
Full Krylov	CPU	20	27	132
	Iter	7	9	27
Probing	CPU	1	2	11
	Iter	7	9	27
	p	4	5	13
	s	24	32	133
	Error	5.5E-8	3.4E-7	4.7E-5
	RDD	100%	100%	7.7%
	NNZ	12,801	12,801	12,801

Table 5: Results of the DMFT experiment for $n = 51^2$ and various scalings of the main diagonal of A.

9 Conclusions and Future Work

A probing method is presented to extract the diagonal entries of the inverse of a matrix. Appropriate probing vectors are constructed by exploiting the pattern of a sparsified matrix inverse and coloring the underlying adjacency graph. A preconditioned Krylov-subspace method is applied to solve the resulting sequence of linear systems efficiently, and the diagonal of the matrix inverse can then be readily extracted. We have shown that the resulting method is a simple and powerful approach from both a theoretical and numerical point of view.

Future research should explore ideas to improve the construction of probing vectors and preconditioners for the Krylov-subspace method. For example, we alluded earlier to the problem of updating the probing matrix, when it turns out that the pattern of sparsified inverse has been miscalculated and must be updated (e.g., taking one more power of A to approximate the inverse). Finally, the problem of extracting diagonals of inverses of indefinite matrices is much harder, because of the absence of any decay of the entries of the inverse. Alternative methods are needed for this case.

Acknowledgments

The authors are indebted to Jie Chen for providing us with the covariance matrix generator, and to Pierre Carrier for many fruitful discussions and his help with the DMFT code.

References

[1] Z. Bai, M. Fahey, and G. Golub, *Some large-scale matrix computation problems*, J. Comput. Appl. Math., 74 (1996), pp. 71–89.

[2] C. Bekas, A. Curioni, and I. Fedulova, *Low cost high performance uncertainty quantification*, in WHPCF '09: Proc. of the 2nd Workshop on High Performance Computational Finance, New York, NY, USA, 2009, ACM, pp. 1–8.

[3] C. Bekas, E. Kokiopoulou, and Y. Saad, *An estimator for the diagonal of a matrix*, Appl. Numer. Math., 57 (2007), pp. 1214–1229.

[4] M. Benzi and G. H. Golub, *Bounds for the entries of matrix functions with applications to preconditioning*, BIT Numer. Math., 39 (1999), pp. 417–438.

[5] M. Benzi and N. Razouk, *Decay bounds and $O(n)$ algorithms for approximating functions of sparse matrices*, Elec. Trans. Numer. Anal., 28 (2007), pp. 16–39.

[6] M. Ceriotti, T. D. Kühne, and M. Parrinello, *An efficient and accurate decomposition of the Fermi operator*, J. Chem. Phys., 129 (2008), p. 024707.

[7] T. F. Chan and T. P. Mathew, *The interface probing technique in domain decomposition*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 212–238.

[8] E. Chow, *A priori sparsity patterns for parallel sparse approximate inverse preconditioners*, SIAM Journal on Scientific Computing, 21 (2000), pp. 1804–1822.

[9] ——, *Parallel implementation and practical use of sparse approximate inverse preconditioners with a priori sparsity patterns*, Int. J. High Perf. Comput. Apps., 15 (2001), p. 56.

[10] E. Chow and Y. Saad, *Approximate inverse preconditioners via sparse-sparse iterations*, SIAM J. Sci. Comput., 19 (1998), pp. 995–1023.

[11] T. F. Coleman, B. S. Garbow, and J. J. More, *Fortran subroutines for estimating sparse Jacobian matrices*, ACM Trans. Math. Software, 10 (1984), pp. 346–347.

[12] T. F. Coleman and J. J. More, *Estimation of sparse Jacobian matrices and graph coloring problems*, SIAM J. Numer. Anal., 20 (1983), pp. 187–209.

[13] A. R. Curtis, M. J. D. Powell, and J. K. Reid, *On the estimation of sparse Jacobian matrices*, J. Inst. Math. Appl., 13 (1974), pp. 117–119.

[14] S. Demko, W. F. Moss, and P. W. Smith, *Decay rates for inverses of band matrices*, Math. Comp., 43 (1984), pp. 491–499.

[15] I. S. Duff, A. M. Erisman, and J. K. Reid, *Direct Methods for Sparse Matrices*, Oxford University Press, 1986.

[16] V. Eijkhout and B. Polman, *Decay rates of inverses of banded M-matrices that are near to Toeplitz matrices*, Lin. Alg. Appl., 109 (1988), pp. 247–277.

[17] J. Freericks, *Transport in Multilayered Nanostructures. The Dynamical Mean-Field Theory Approach*, Imperial College, London, 2006.

[18] R. W. Freund and N. M. Nachtigal, *QMR: a quasi-minimal residual method for non-Hermitian linear systems*, Numer. Math., 60 (1991), pp. 315–339.

[19] A. H. Gebremedhin, F. Manne, and A. Pothen, *What color is your Jacobian? graph coloring for computing derivatives*, SIAM Rev., 47 (2005), pp. 629–705.

[20] G. Golub and Z. Strakos, *Estimates in quadratic formulas*, Numer. Algor., 8 (1994), pp. 241–268.

[21] G. H. Golub and G. Meurant, *Matrices, moments, and quadrature*, in Numerical Analysis 1993, D. F. Griffiths and G. A. Watson, eds., vol. 303, Pitman, Research Notes in Mathematics, 1994, pp. 105–1–6.

[22] A. S. Householder, *Theory of Matrices in Numerical Analysis*, Blaisdell Pub. Co., Johnson, CO, 1964.

[23] T. Huckle, *Approximate sparsity patterns for the inverse of a matrix and preconditioning*, Appl. Numer. Math., 30 (1999), pp. 291–303.

[24] M. Hutchinson, *A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines*, Commun. Stat. Simul. Comput., 19 (1990), pp. 433–450.

[25] M. Kubale, *Graph Colorings*, American Mathematical Society, Providence, Rhode Island, 2004.

[26] L. Lin, J. Lu, R. Car, and W. E, *Multipole representation of the Fermi operator with application to the electronic structure analysis of metallic systems*, Phys. Rev. B (Condensed Matter and Materials Physics), 79 (2009), pp. 115–133.

[27] L. Lin, J. Lu, L. Ying, R. Car, and W. E, *Fast algorithm for extracting the diagonal of the inverse matrix with application to the electronic structure analysis of metallic systems*, Commun. Math. Sci., 7 (2009), pp. 755–777.

[28] M. Luby, *A simple parallel algorithm for the maximal independent set problem*, SIAM J. Comput., 15 (1986), pp. 1036–1055.

[29] G. Meurant, *A review on the inverse of symmetric tridiagonal and block tridiagonal matrices*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 707–728.

[30] R. Nabben, *Decay rates of the inverse of nonsymmetric tridiagonal and band matrices*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 820–837.

[31] V. Pan and J. Reif, *Efficient parallel solution of linear systems*, in Proc. 17th ACM STOC, 1985, pp. 143–152.

[32] R.-S. Ran and T.-Z. Huang, *An inversion algorithm for a banded matrix*, Comp. Math. Appl., 58 (2009), pp. 1699–1710.

[33] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*, MIT Press, Cambridge, MA, 2006.

[34] H. Roder, R. Silver, D. Drabold, and J. Dong, *The kernel polynomial method for non-orthogonal electronic structure calculation of amorphous diamond*, Phys. Rev. B, 55 (1997), pp. 15382–15385.

[35] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, PA, USA, 2003. Second edition.

[36] Y. Saad and M. H. Schultz, *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 856–869.

[37] K. Schmitt, M. Anitescu, and D. Negrut, *Efficient sampling for spatial uncertainty quantification in multibody system dynamics applications*, Int. J. Numer. Meth. Eng., 80 (2009), pp. 537–564.

[38] R. B. Sidje and Y. Saad, *Rational approximation to the Fermi-Dirac function with applications in density functional theory*, Tech. Rep. umsi-2008-279, Minnesota Supercomputer Institute, University of Minnesota, 2008.

[39] C. Siefert and E. de Sturler, *Probing methods for saddle-point problems*, Elec. Trans. Numer. Anal., 22 (2006), pp. 163–183.

[40] P. Sonneveld and M. B. van Gijzen, *IDR(s): a family of simple and fast algorithms for solving large nonsymmetric linear systems*, SIAM J. Sci. Comput., 31 (2008), pp. 1035–1062.

[41] K. Takahashi, J. Fagan, and M.-S. Chin, *Formation of a sparse bus impedance matrix and its application to short circuit study*, in Proc. of the Eighth Inst. PICA Conf., Minneapolis, MN, IEEE Power Engineering Society, 1973, pp. 63–69.

[42] W.-P. Tang, *Toward an effective sparse approximate inverse preconditioner*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 970–986.

[43] H. A. van der Vorst, *BI-CGSTAB: a fast and smoothly converging variant of BI-CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 631–644.