SIAM J. MATRIX ANAL. APPL.
Vol. 23, No. 3, pp. 799–818

# APPROXIMATION OF THE DETERMINANT OF LARGE SPARSE SYMMETRIC POSITIVE DEFINITE MATRICES*

ARNOLD REUSKEN†

**Abstract.** This paper is concerned with the problem of approximating $\det(A)^{1/n}$ for a large sparse symmetric positive definite matrix $A$ of order $n$. It is shown that an efficient solution of this problem is obtained by using a sparse approximate inverse of $A$. The method is explained and theoretical properties are discussed. The method is ideal for implementation on a parallel computer. Numerical experiments are described that illustrate the performance of this new method and provide a comparison with Monte Carlo–type methods from the literature.

**Key words.** determinant, sparse approximate inverse, preconditioning

**AMS subject classifications.** 6F10, 65F10, 65F50

**PII.** S089547980036869X

## 1. Introduction.

Throughout this paper, $A$ denotes a real symmetric positive definite matrix of order $n$ with eigenvalues

$$0 < \lambda_1 \le \lambda_2 \le \cdots \le \lambda_n \ .$$

In a number of applications, for example, in lattice quantum chromodynamics [14, 8, 16, 17] certain functions of the determinant of $A$, such as $\det(A)^{\frac{1}{2}}$ or $\ln(\det(A))$, are of interest. It is well known (cf. also section 2) that for large $n$ the function $A \to \det(A)$ has poor scaling properties and can be very ill-conditioned for certain matrices $A$. In this paper we consider the function

$$d: \ A \to \det(A)^{\frac{1}{n}} \ . \tag{1.1}$$

A few basic properties of this function are discussed in section 2. In this paper we present a new method for approximating $d(A)$ for large sparse matrices $A$. The method is based on using a matrix which is in a certain sense close to $A^{-1}$ and for which the determinant can be computed with low computational costs. One popular method for approximating $A$ is based on the construction of an incomplete Cholesky factorization. This incomplete factorization is often used as a preconditioner when solving linear systems with matrix $A$. In this paper we use another preconditioning technique, namely, that of factorized sparse approximate inverses (cf. [1, 7, 10, 12]). With such a method a lower triangular matrix $G_E$ with a prescribed sparsity structure $E$ can be constructed such that $G_E A G_E^T$ is in a certain sense close to the identity. We then use $\det(G_E)^{-2/n} = \prod_{i=1}^n (G_E)_{ii}^{-2/n}$ as an approximation for $d(A)$. In section 3 we explain the construction of $G_E$ and discuss theoretical properties of this sparse approximate inverse. For example, such a sparse approximate inverse can be shown to exist for any symmetric positive definite $A$ and has an interesting *optimality property* related to $d(A)$. From this optimality property it immediately follows that $d(A) \le \det(G_E)^{-2/n}$ holds and that the approximation of $d(A)$ by $\det(G_E)^{-2/n}$ becomes

*Received by the editors March 2, 2000; accepted for publication (in revised form) by N. J. Higham August 17, 2001; published electronically January 23, 2002.
http://www.siam.org/journals/simax/23-3/36869.html

†Institut für Geometrie und Praktische Mathematik, RWTH Aachen, Templergraben 55, D-52056 Aachen, Germany (reusken@igpm.rwth-aachen.de).

better if we take a larger sparsity pattern $E$. Besides this optimality property the method we present has two other interesting properties. The method is *ideal for a parallel implementation* and has *very low storage requirements*.

To make a comparison with other methods for approximating $d(A)$ we describe two known Monte Carlo–type methods (from [3] and [16]). We present results of a few numerical experiments. In these experiments the new method and the Monte Carlo methods are applied to a few model examples of large sparse symmetric positive definite matrices.

## 2. Preliminaries.

In this section we discuss a few elementary properties of the function $d$. We give a comparison between the conditioning of the function $d$ and of the function $A \to d(A)^n = \det(A)$. We use the notation $\| \cdot \|_2$ for the Euclidean norm, and $\kappa(A) = \lambda_n/\lambda_1$ denotes the spectral condition number of $A$. The trace of the matrix $A$ is denoted by $\mathrm{tr}(A)$.

Lemma 2.1. *Let $A$ and $A + \delta A$ be symmetric positive definite matrices of order $n$. The following inequalities hold:*

$$\lambda_1 \leq d(A) \leq \lambda_n \ , \tag{2.1a}$$

$$d(A) \leq \frac{1}{n}\mathrm{tr}(A) \ , \tag{2.1b}$$

$$\Big|\frac{d(A+\delta A) - d(A)}{d(A)}\Big| \leq \kappa(A)\frac{\|\delta A\|_2}{\|A\|_2} \ . \tag{2.1c}$$

*Proof.* The result in (2.1a) follows from

$$\lambda_1 \leq \left(\prod_{i=1}^{n} \lambda_i\right)^{\frac{1}{n}} \leq \lambda_n \ .$$

The result in (2.1b) follows from the inequality between the geometric and arithmetic mean:

$$d(A) = \left(\prod_{i=1}^{n} \lambda_i\right)^{\frac{1}{n}} \leq \frac{1}{n}\sum_{i=1}^{n}\lambda_i = \frac{1}{n}\mathrm{tr}(A) \ .$$

Now note that

$$\frac{d(A+\delta A) - d(A)}{d(A)} = \big(\det(I + A^{-1}\delta A)\big)^{\frac{1}{n}} - 1 = \left(\prod_{i=1}^{n}(1 + \lambda_i(A^{-1}\delta A))\right)^{\frac{1}{n}} - 1 \ .$$

From $\lambda_i(A^{-1}\delta A) \leq \|A^{-1}\|_2\|\delta A\|_2$ it follows that

$$\left(\prod_{i=1}^{n}(1 + \lambda_i(A^{-1}\delta A))\right)^{\frac{1}{n}} - 1 \leq \left(\prod_{i=1}^{n}(1 + \|A^{-1}\|_2\|\delta A\|_2)\right)^{\frac{1}{n}} - 1 = \|A^{-1}\|_2\|\delta A\|_2 \ .$$

Using $1 + \lambda_i(A^{-1}\delta A) > 0$ and $\lambda_i(A^{-1}\delta A) \geq -\|A^{-1}\|_2\|\delta A\|_2$ we obtain

$$\left(\prod_{i=1}^{n}(1 + \lambda_i(A^{-1}\delta A))\right)^{\frac{1}{n}} - 1 \geq \left(\prod_{i=1}^{n}\max\{0,\ 1 - \|A^{-1}\|_2\|\delta A\|_2\}\right)^{\frac{1}{n}} - 1 \geq -\|A^{-1}\|_2\|\delta A\|_2.$$

Thus we have

$$\Big|\frac{d(A+\delta A)-d(A)}{d(A)}\Big| \leq \|A^{-1}\|_2\|\delta A\|_2 = \kappa(A)\frac{\|\delta A\|_2}{\|A\|_2}\ ,$$

and the result in (2.1c) is proved. ☐

The result in (2.1c) shows that the function $d(A)$ is well-conditioned for matrices $A$ which have a not-too-large condition number $\kappa(A)$.

We now briefly discuss the difference in conditioning between the functions $A \to d(A)$ and $A \to \det(A)$. For any symmetric positive definite matrix $B$ of order $n$ we have

$$d'(A)B := \lim_{t\to 0}\frac{d(A+tB)-d(A)}{t} = \frac{d(A)}{n}\mathrm{tr}(A^{-1}B)\ .$$

From the Courant–Fischer eigenvalue characterization, we obtain, for all $i$, $\lambda_i(A^{-1}B) \leq \lambda_i(A^{-1})\|B\|_2$. Hence

$$\|d'(A)\|_2 := \max_{B\text{ is SPD}}\frac{|d'(A)B|}{\|B\|_2} = \frac{d(A)}{n}\max_{B\text{ is SPD}}\frac{\mathrm{tr}(A^{-1}B)}{\|B\|_2} \leq \frac{d(A)}{n}\mathrm{tr}(A^{-1})\ ,$$

with equality for $B = I$. Thus for the condition number of the function $d$ we have

$$\frac{\|A\|_2\|d'(A)\|_2}{d(A)} = \frac{1}{n}\|A\|_2\mathrm{tr}(A^{-1}) \leq \kappa(A)\ . \tag{2.2}$$

Note that for the diagonal matrix $A = \mathrm{diag}(A_{ii})$ with $A_{11} = 1,\ A_{ii} = \alpha$ for $i > 1$, the inequality in (2.2) is sharp if $0 < \alpha \ll 1$ and $n$ is large. For this $A$ and with $\delta A = \varepsilon I$, $0 < \varepsilon \ll \alpha$, the bound in (2.1c) is sharp, too.

For $\tilde{d}(A) = \det(A) = d(A)^n$ the condition number is given by

$$\frac{\|A\|_2\|\tilde{d}'(A)\|_2}{\tilde{d}(A)} = \frac{\|A\|_2 n d(A)^{n-1}\|d'(A)\|_2}{d(A)^n} = \|A\|_2\mathrm{tr}(A^{-1})\ , \tag{2.3}$$

i.e., $n$ times larger than the condition number in (2.2). The condition numbers for $d$ and $\tilde{d}$ give an indication of the sensitivity if the perturbation $\|\delta A\|_2$ is sufficiently small. Note that the bound in (2.1c) is valid for arbitrary symmetric positive definite perturbations $\delta A$. The bound shows that even for larger perturbations the function $d(A)$ is well-conditioned at $A$ if $\kappa(A)$ is not too large. For the function $\tilde{d}(A)$ the effect of relatively large perturbations can be much worse than for the asymptotic case ($\delta A \to 0$), which is characterized by the condition number in (2.3). Consider, for example, for $0 < \varepsilon < \frac{1}{2}$ a perturbation $\delta A = \varepsilon A$, i.e., $\|\delta A\|_2/\|A\|_2 = \varepsilon$. Then

$$\frac{\tilde{d}(A+\delta A)-\tilde{d}(A)}{\tilde{d}(A)} = (1+\varepsilon)^n - 1 \geq e^{\frac{1}{2}n\varepsilon} - 1\ ,$$

which is very large if, for example, $\varepsilon = 10^{-3},\ n = 10^5$.

The results in this section show that the numerical approximation of the function $d(A)$ can be considered to be an easier task than the numerical approximation of $A \to \det(A)$.

*Remark* 2.2. The results on conditioning derived above and the fact that in the analysis of the sparse approximate inverse the function $d(A)$ plays a natural role (cf.

section 3) are the main motivation for considering $d(A)$ instead of $A \to \det(A)$. Of course, an algorithm for approximating $d(A)$ yields an approximation for $\det(A)$ or $\ln(\det(A))$, too. Note that the functions $x \to x^n$ and $x \to \ln(x^n)$ have condition numbers $n$ and $1/\ln(x)$, respectively. Hence, if $\hat{d}$ is an approximation of $d(A)$ with relative error $|\hat{d} - d(A)|/d(A) \leq$ eps, then it follows that $|\hat{d}^n - \det(A)|/\det(A) \lesssim n\,\text{eps}$ and $|\ln(\hat{d}^n) - \ln(\det(A))|/|\ln(\det(A))| \lesssim \text{eps}/|\ln(d(A))|$.

## 3. Sparse approximate inverse.

In this section we explain and analyze the construction of a sparse approximate inverse of the matrix $A$. Let $A = LL^T$ be the Cholesky factorization of $A$, i.e., $L$ is lower triangular and $L^{-1}AL^{-T} = I$. Note that $d(A) = d(L)^2 = \prod_{i=1}^n L_{ii}^{2/n}$. We will construct a sparse lower triangular approximation $G$ of $L^{-1}$ and approximate $d(A)$ by $d(G)^{-2} = \prod_{i=1}^n G_{ii}^{-2/n}$. The construction of a sparse approximate inverse that we use in this paper was introduced in [10, 11, 12] and can also be found in [1]. Some of the results derived in this section are also presented in [1].

### 3.1. Introduction.

We first introduce some notation. Let $E \subset \{(i,j) \mid 1 \leq i,j \leq n\}$ be a given sparsity pattern. By $\#E$ we denote the number of elements in $E$. Let $S_E$ be the set of $n \times n$ matrices for which all entries are set to zero if the corresponding index is *not* in $E$:

$$S_E = \{M \in \mathbb{R}^{n\times n} \mid M_{ij} = 0 \text{ if } (i,j) \notin E\} \ .$$

For $1 \leq i \leq n$ let $E_i = E \cap \{(i,j) \mid 1 \leq j \leq n\}$. If $n_i := \#E_i > 0$, we use the representation

$$E_i = \{(i,j_1),(i,j_2),\ldots,(i,j_{n_i})\}, \quad 1 \leq j_1 < j_2 < \cdots < j_{n_i} \leq n \ . \tag{3.1}$$

For $n_i > 0$ we define the projection

$$P_i : \mathbb{R}^n \to \mathbb{R}^{n_i} \ , \quad P_i(x_1,x_2,\ldots,x_n)^T = (x_{j_1},x_{j_2},\ldots,x_{j_{n_i}})^T \ . \tag{3.2}$$

Note that the matrix

$$P_i A P_i^T : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i}$$

is symmetric positive definite. To facilitate the analysis below, we first discuss the construction of a approximate sparse inverse $M_E \in S_E$ in a general framework. For $M_E \in S_E$ we use the representation

$$M_E = \begin{bmatrix} m_1^T \\ m_2^T \\ \vdots \\ m_n^T \end{bmatrix} \ , \quad m_i \in \mathbb{R}^n \ .$$

Note that if $n_i = 0$, then $m_i^T = (0,0,\ldots,0)$.

For given $A, B \in \mathbb{R}^{n\times n}$ with $A$ symmetric positive definite, we consider the following problem:

$$\text{determine } M_E \in S_E \text{ such that } (M_E A)_{ij} = B_{ij} \text{ for all } (i,j) \in E \ . \tag{3.3}$$

In (3.3) we have $\#E$ equations to determine $\#E$ entries in $M_E$. We first give two basic lemmas which will play an important role in the analysis of the sparse approximate inverse defined in (3.9) below.

LEMMA 3.1. *The problem* (3.3) *has a unique solution $M_E \in S_E$. If $n_i > 0$, then the $i$th row of $M_E$ is given by $m_i^T$ with*

$$m_i = P_i^T(P_iAP_i^T)^{-1}P_ib_i \ , \tag{3.4}$$

*where $b_i^T$ is the $i$th row of $B$.*

*Proof.* The equations in (3.3) can be represented as

$$(m_i^TA)_{j_k} = (b_i^T)_{j_k} \text{ for all } i \text{ with } n_i > 0 \text{ and all } k = 1, 2, \dots, n_i \ ,$$

where $m_i^T$ is the $i$th row of $M_E$. Consider an $i$ with $n_i > 0$. Note that $M_E \in S_E$, and hence $P_i^TP_im_i = m_i$. For the unknown entries in $m_i$ we obtain the system of equations

$$(AP_i^TP_im_i)_{j_k} = (b_i)_{j_k} \ , \quad k = 1, 2, \dots, n_i \ ,$$

which is equivalent to

$$P_iAP_i^TP_im_i = P_ib_i \ .$$

The matrix $P_iAP_i^T$ is symmetric positive definite and thus $m_i$ must satisfy

$$P_im_i = (P_iAP_i^T)^{-1}P_ib_i \ .$$

Using $P_i^TP_im_i = m_i$ we obtain the result in (3.4). The construction in this proof shows that the solution is unique. □

Below we use the Frobenius norm, denoted by $\|\cdot\|_F$:

$$\|B\|_F^2 = \sum_{i,j=1}^n B_{ij}^2 = \mathrm{tr}(BB^T) \ , \quad B \in \mathbb{R}^{n\times n}. \tag{3.5}$$

LEMMA 3.2. *Let $A = LL^T$ be the Cholesky factorization of $A$ and let $M_E \in S_E$ be the unique solution of* (3.3). *Then $M_E$ is the unique minimizer of the functional*

$$M \to \|(B - MA)L^{-T}\|_F^2 = \mathrm{tr}((B - MA)A^{-1}(B - MA)^T), \quad M \in S_E. \tag{3.6}$$

*Proof.* Let $e_i$ be the $i$th basis vector in $\mathbb{R}^n$. Take $M \in S_E$. The $i$th rows of $M$ and $B$ are denoted by $m_i^T$ and $b_i^T$, respectively. Now note

$$\mathrm{tr}((B - MA)A^{-1}(B - MA)^T) = \sum_{i=1}^n e_i^T(BA^{-1}B^T - MB^T - BM^T + MAM^T)e_i$$

$$= \mathrm{tr}(BA^{-1}B^T) + \sum_{i=1}^n(-2m_i^Tb_i + m_i^TAm_i) \ . \tag{3.7}$$

The minimum of the functional (3.6) is obtained if in (3.7) we minimize the functionals

$$m_i \to -2m_i^Tb_i + m_i^TAm_i \ , \quad m_i \in \mathcal{R}(P_i^T) \tag{3.8}$$

for all $i$ with $n_i > 0$. If we write $m_i = P_i^T\hat{m}_i$ , $\hat{m}_i \in \mathbb{R}^{n_i}$, then for $n_i > 0$ the functional (3.8) can be rewritten as

$$\hat{m}_i \to -2\hat{m}_i^TP_ib_i + \hat{m}_i^TP_iAP_i^T\hat{m}_i \ , \quad \hat{m}_i \in \mathbb{R}^{n_i}.$$

The unique minimum of this functional is obtained for $\hat{m}_i = (P_iAP_i^T)^{-1}P_ib_i$, i.e., $m_i = P_i^T(P_iAP_i^T)^{-1}P_ib_i$ for all $i$ with $n_i > 0$. Using Lemma 3.1 it follows that $M_E$ is the unique minimizer of the functional (3.6). □

**3.2. Sparse approximate inverse for approximating $d(A)$.** We now introduce the sparse approximate inverse that will be used as an approximation for $L^{-1}$. For this we chose a lower triangular pattern $E^l \subset \{(i,j) \mid 1 \le j \le i \le n\}$ and we assume that $(i,i) \in E^l$ for all $i$. The sparse approximate inverse is constructed in two steps:

$$1. \quad \hat{G}_{E^l} \in S_{E^l} \text{ such that } (\hat{G}_{E^l}A)_{ij} = \delta_{ij} \text{ for all } (i,j) \in E^l \ , \tag{3.9a}$$

$$2. \qquad G_{E^l} := (\mathrm{diag}(\hat{G}_{E^l}))^{-\frac{1}{2}}\hat{G}_{E^l} \ . \tag{3.9b}$$

The construction of $G_{E^l}$ in (3.9) was first introduced in [10]. A theoretical background for this factorized sparse inverse is given in [12]. The approximate inverse $\hat{G}_{E^l}$ in (3.9a) is of the form (3.3) with $B = I$. From Lemma 3.1 it follows that in (3.9a) there is a unique solution $\hat{G}_{E^l}$. Note that because $E^l$ is lower triangular and $(i,i) \in E^l$ we have $n_i = \#E^l > 0$ for all $i$ and $j_{n_i} = i$ in (3.1). Hence it follows from Lemma 3.1 that the $i$th row of $\hat{G}_{E^l}$, denoted by $g_i^T$, is given by

$$\begin{aligned} g_i &= P_i^T(P_iAP_i^T)^{-1}P_ie_i, \qquad i = 1,2,\ldots,n, \\ &= P_i^T(P_iAP_i^T)^{-1}\hat{e}_i, \quad \text{with } \ \hat{e}_i = (0,\ldots,0,1)^T \in \mathbb{R}^{n_i} \ . \end{aligned} \tag{3.10}$$

The $i$th entry of $g_i$, i.e., $e_i^Tg_i$, is given by $\hat{e}_i^T(P_iAP_i^T)^{-1}\hat{e}_i$, which is strictly positive because $P_iAP_i^T$ is symmetric positive definite. Hence $\mathrm{diag}(\hat{G}_{E^l})$ contains only strictly positive entries and the second step (3.9b) is well-defined. Define $\hat{g}_i = P_ig_i$. The sparse approximate inverse $\hat{G}_{E^l}$ in (3.9a) can be computed by solving the low-dimensional symmetric positive definite systems

$$P_iAP_i^T\hat{g}_i = \hat{e}_i := (0,\ldots,1)^T, \quad i = 1,2,\ldots,n. \tag{3.11}$$

For the approximation of $d(A)$ we propose to use $d(G_{E^l})^{-2}$. Due to

$$d(G_{E^l})^{-2} = d(\hat{G}_{E^l})^{-1} = \prod_{i=1}^{n}(\hat{G}_{E^l})_{ii}^{-\frac{1}{n}}$$

we only need the diagonal entries of $\hat{G}_{E^l}$. In the systems $P_iAP_i^T\hat{g}_i = \hat{e}_i$ we then only have to compute the last entry of $\hat{g}_i$, i.e., $(\hat{g}_i)_{n_i}$. If these systems are solved using the Cholesky factorization $P_iAP_i^T =: L_iL_i^T$ ($L_i$ lower triangular) we only need the $(n_i,n_i)$ entry of $L_i$, since $(\hat{g}_i)_{n_i} = (L_i)_{n_in_i}^{-2}$ and thus

$$d(G_{E^l})^{-2} = \prod_{i=1}^{n}(L_i)_{n_in_i}^{\frac{2}{n}} \ .$$

This leads to the following algorithm.

Algorithm 3.3. *Let $A \in \mathbb{R}^{n\times n}$ and a lower triangular pattern $E^l$ be given.*

```
For $i = 1,\ldots,n$ do:
    1. Construct the matrix $A_i := P_iAP_i^T \in \mathbb{R}^{n_i\times n_i}$.
    2. Compute the Cholesky factorization $A_i = L_iL_i^T$ and set $\gamma_i := (L_i)_{n_in_i}$.
End. Compute
```

$$\prod_{i=1}^{n}\gamma_i^{\frac{2}{n}} \ . \tag{3.12}$$

**3.3. Analysis of the method.** We now derive some interesting properties of the sparse approximate inverse as in (3.9). We start with a minimization property of $\hat{G}_{E^l}$.

THEOREM 3.4. *Let $A = LL^T$ be the Cholesky factorization of $A$ and $D := \operatorname{diag}(L)$, $\hat{L} := LD$. $\hat{G}_{E^l}$ as in* (3.9a) *is the unique minimizer of the functional*

$$G \to \|(I - G\hat{L})D^{-1}\|_F^2 = \operatorname{tr}((I - G\hat{L})D^{-2}(I - G\hat{L})^T), \quad G \in S_{E^l}. \tag{3.13}$$

*Proof.* The construction of $\hat{G}_{E^l}$ in (3.9a) is as in (3.3) with $E = E^l$, $B = I$. Hence Lemma 3.2 is applicable with $B = I$. It follows that $\hat{G}_{E^l}$ is the unique minimizer of

$$G \to \|(I - GA)L^{-T}\|_F^2 \ , \quad G \in S_{E^l} \ . \tag{3.14}$$

Decompose $L^{-T}$ as $L^{-T} = D^{-1} + R$ with $R$ strictly upper triangular. We then obtain

$$\begin{aligned} \|(I - GA)L^{-T}\|_F^2 &= \|(I - GLL^T)L^{-T}\|_F^2 = \|D^{-1} + R - GL\|_F^2 \\ &= \|D^{-1} - GL\|_F^2 + \|R\|_F^2 = \|(I - G\hat{L})D^{-1}\|_F^2 + \|R\|_F^2 \ . \end{aligned}$$

Hence the minimizers in (3.14) and (3.13) are the same. □

*Remark* 3.5. From the result in Theorem 3.4 we see that in a scaled Frobenius norm (scaling with $D^{-1}$) $\hat{G}_{E^l}$ is the optimal approximation of $\hat{L}^{-1}$ in the set $S_{E^l}$, in the sense that $\hat{G}_{E^l}\hat{L}$ is closest to the identity. A seemingly more natural minimization problem is

$$\min_{G \in S_{E^l}} \|I - GL\|_F \ , \tag{3.15}$$

i.e., we directly approximate $L^{-1}$ (instead of $\hat{L}^{-1}$) and do not use the scaling with $D^{-1}$. The minimization problem (3.15) is of the form as in Lemma 3.2 with $B = L^T$, $E = E^l$. Hence the unique minimizer in (3.15), denoted by $\tilde{G}_{E^l}$, must satisfy (3.3) with $B = L^T$:

$$(\tilde{G}_{E^l}A)_{ij} = L_{ji} \quad \text{for all } (i,j) \in E^l \ . \tag{3.16}$$

Because $E^l$ contains only indices $(i,j)$ with $i \geq j$ and $L_{ji} = 0$ for $i > j$, it follows that $\tilde{G}_{E^l} \in S_{E^l}$ must satisfy

$$(\tilde{G}_{E^l}A)_{ij} = \begin{cases} 0 & \text{if } i \neq j, \\ L_{ii} & \text{if } i = j \end{cases} \qquad \text{for all } (i,j) \in E^l \ . \tag{3.17}$$

This is similar to the system of equations in (3.9a), which characterizes $\hat{G}_{E^l}$. However, in (3.17) one needs the values $L_{ii}$, which in general are not available. Hence opposite to the minimization problem related to the functional (3.13) the minimization problem (3.15) is in general not solvable with acceptable computational costs. □

The following lemma will be used in the proof of Theorem 3.8.

LEMMA 3.6. *Let $\hat{G}_{E^l}$ be as in* (3.9a). *Decompose $\hat{G}_{E^l}$ as $\hat{G}_{E^l} = \hat{D}(I - \hat{L})$, with $\hat{D}$ diagonal and $\hat{L}$ strictly lower triangular. Define $E^l_- := E^l \setminus \{(i,i) \mid 1 \leq i \leq n\}$. Then $\hat{L}$ is the unique minimizer of the functional*

$$L \to \operatorname{tr}((I - L)A(I - L^T)) \ , \qquad L \in S_{E^l_-} \ , \tag{3.18}$$

*and also of the functional*

$$L \to \det[\operatorname{diag}((I-L)A(I-L^T))]\ , \qquad L \in S_{E_-^l}\ . \tag{3.19}$$

*Furthermore, for $\hat{D}$ we have*

$$\hat{D} = [\operatorname{diag}((I-\hat{L})A(I-\hat{L}^T))]^{-1}\ . \tag{3.20}$$

*Proof.* From the construction in (3.9a) it follows that

$$((I-\hat{L})A)_{ij} = 0 \quad \text{for all } (i,j) \in E_-^l\ ,$$

i.e., $\hat{L} \in S_{E_-^l}$ is such that $(\hat{L}A)_{ij} = A_{ij}$ for all $(i,j) \in S_{E_-^l}$. This is of the form (3.3) with $B = A$, $E = E_-^l$. From Lemma 3.2 we obtain that $\hat{L}$ is the unique minimizer of the functional

$$L \to \operatorname{tr}((A-LA)A^{-1}(A-LA)^T) = \operatorname{tr}((I-L)A(I-L^T))\ , \quad L \in S_{E_-^l}\ ,$$

i.e., of the functional (3.18). From the proof of Lemma 3.2, with $B = A$, it follows that the minimization problem

$$\min_{L \in S_{E_-^l}} \operatorname{tr}((I-L)A(I-L^T))$$

decouples into separate minimization problems (cf. (3.8)) for the rows of $L$:

$$\min_{l_i \in \mathcal{R}(P_i^T)} \{-2l_i^T a_i + l_i^T A l_i\} \tag{3.21}$$

for all $i$ with $n_i > 0$. Here $l_i^T$ and $a_i^T$ are the $i$th rows of $L$ and $A$, respectively. The minimization problem corresponding to (3.19) is

$$\min_{L \in S_{E_-^l}} \prod_{i=1}^n ((I-L)A(I-L^T))_{ii} = \min_{L \in S_{E_-^l}} \prod_{i=1}^n (A_{ii} - 2l_i^T a_i + l_i^T A l_i)\ .$$

This decouples into the same minimization problems as in (3.21). Hence the functionals in (3.18) and (3.19) have the same minimizer.

Let $J = \operatorname{diag}((I-\hat{L})A(I-\hat{L}^T))$. Using the construction of $\hat{G}_{E^l}$ in (3.9a) we obtain

$$\begin{aligned}
\hat{D}_{ii}^2 J_{ii} &= (\hat{D}(I-\hat{L})A(I-\hat{L}^T)\hat{D})_{ii} = (\hat{G}_{E^l} A \hat{G}_{E^l}^T)_{ii} \\
&= \sum_{k=1}^n (\hat{G}_{E^l}A)_{ik}(\hat{G}_{E^l})_{ik} = \sum_{k=1,(i,k)\in E^l}^n \delta_{ik}(\hat{G}_{E^l})_{ik} \\
&= (\hat{G}_{E^l})_{ii} = \hat{D}_{ii}\ .
\end{aligned}$$

Hence $\hat{D}_{ii} = J_{ii}^{-1}$ holds for all $i$, i.e., (3.20) holds. □

COROLLARY 3.7. *From* (3.20) *it follows that* $\operatorname{diag}(\hat{G}_{E^l}A\hat{G}_{E^l}) = \operatorname{diag}(\hat{G}_{E^l})$ *and thus, using* (3.9b), *we obtain*

$$\operatorname{diag}(G_{E^l}AG_{E^l}) = I \tag{3.22}$$

*for the approximate inverse* $G_{E^l}$.

The following theorem gives a main result in the theory of approximate inverses. It was first derived in [12]. A proof can be found in [1], too.

THEOREM 3.8. *Let* $G_{E^l}$ *be the approximate inverse in* (3.9). *Then* $G_{E^l}$ *is the unique minimizer of the functional*

$$G \to \frac{\frac{1}{n}\mathrm{tr}(GAG^T)}{\det(GAG^T)^{\frac{1}{n}}}\ , \qquad G \in S_{E^l}\ . \tag{3.23}$$

*Proof.* For $G \in S_{E^l}$ we use the decomposition $G = D(I - L)$, with $D$ diagonal and $L \in S_{E^l_-}$. Furthermore, for $L \in S_{E^l_-}$, $J_L := \mathrm{diag}((I - L)A(I - L^T))$. Now note

$$\frac{\frac{1}{n}\mathrm{tr}(GAG^T)}{\det(GAG^T)^{\frac{1}{n}}} = \det(A)^{-\frac{1}{n}}\frac{\frac{1}{n}\mathrm{tr}(D(I-L)A(I-L^T)D)}{\det(G^2)^{\frac{1}{n}}} = \det(A)^{-\frac{1}{n}}\frac{\frac{1}{n}\mathrm{tr}(D^2J_L)}{\det(D^2)^{\frac{1}{n}}}$$

$$= \det(A)^{-\frac{1}{n}}\frac{\frac{1}{n}\mathrm{tr}(D^2J_L)}{\det(D^2J_L)^{\frac{1}{n}}}\det(J_L)^{\frac{1}{n}} \geq \det(A)^{-\frac{1}{n}}\det(J_L)^{\frac{1}{n}}\ . \tag{3.24}$$

The inequality in (3.24) follows from the inequality between the arithmetic and geometric mean: $\frac{1}{n}\sum_{i=1}^n \alpha_i \geq (\prod_{i=1}^n \alpha_i)^{1/n}$ for $\alpha_i \geq 0$.

For $\hat{G}_{E^l}$ in (3.9a) we use the decomposition $\hat{G}_{E^l} = \hat{D}(I-\hat{L})$. For the approximate inverse $G_{E^l}$ we then have $G_{E^l} = (\mathrm{diag}(\hat{G}_{E^l}))^{-\frac{1}{2}}\hat{G}_{E^l} = \hat{D}^{\frac{1}{2}}(I - \hat{L})$. From (3.19) of Lemma 3.6 it follows that $\det(J_L) \geq \det(J_{\hat{L}})$ for all $L \in S_{E^l_-}$. Furthermore, from (3.20) of Lemma 3.6 we obtain that for $G_{E^l} = \hat{D}^{\frac{1}{2}}(I - \hat{L})$ we have $(\hat{D}^{\frac{1}{2}})^2 J_{\hat{L}} = I$ and thus equality in (3.24) for $G = G_{E^l}$. We conclude that $G_{E^l}$ is the unique minimizer of the functional in (3.23). □

*Remark* 3.9. The quantity

$$K(A) = \frac{\frac{1}{n}\mathrm{tr}(A)}{\det(A)^{\frac{1}{n}}}$$

can be seen as a nonstandard condition number (cf. [1, 10]). Properties of this quantity are given in [1, Theorem 13.5]. One elementary property is

$$1 \leq K(A) \leq \frac{\lambda_n}{\lambda_1} = \kappa(A)\ .$$

COROLLARY 3.10. *For the approximate inverse* $G_{E^l}$ *as in* (3.9) *we have* (*cf.* (3.22))

$$1 \leq K(G_{E^l}AG_{E^l}^T) = \frac{1}{\det(G_{E^l}AG_{E^l}^T)^{\frac{1}{n}}}\ ,$$

*i.e.*,

$$d(A) \leq \det(G_{E^l}^2)^{-\frac{1}{n}} = \prod_{i=1}^n (G_{E^l})_{ii}^{-\frac{2}{n}} = \prod_{i=1}^n (\hat{G}_{E^l})_{ii}^{-\frac{1}{n}} = \prod_{i=1}^n \gamma_i^{\frac{2}{n}}\ , \tag{3.25}$$

*where* $\gamma_i$ *is as in* (3.12). *Let* $\tilde{E}^l$ *be a lower triangular sparsity pattern that is larger than* $E^l$, *i.e.*, $E^l \subset \tilde{E}^l \subset \{(i,j) \mid 1 \leq j \leq i \leq n\}$. *From the optimality result in Theorem* 3.8 *it follows that*

$$1 \leq K(G_{\tilde{E}^l}AG_{\tilde{E}^l}^T) \leq K(G_{E^l}AG_{E^l}^T)\ . \tag{3.26}$$

In the following remark we summarize the main properties of the new method for approximating $d(A)$ that is formulated in Algorithm 3.3.

*Remark* 3.11. The method of approximating $d(A)$ by $d(G_{E^l})^{-2} = d(\hat{G}_{E^l})^{-1}$ boils down to choosing a sparsity pattern $E^l$ and computing the Cholesky decomposition of the low-dimensional matrices $A_i$ in step 2 of Algorithm 3.3. We note the following related to this algorithm:

1. The sparse approximate inverse exists for every symmetric positive definite $A$. Note that such an existence result does not hold for the incomplete Cholesky factorization.

2. The construction of the matrices $A_i = P_i A P_i^T$ and the computation of the Cholesky factorization $A_i = L_i L_i^T$ can be realized for all $i$ *in parallel*. Hence the method has a very high potential for parallelism.

3. If for a given $i$ the number $\gamma_i = (L_i)_{n_i n_i}$ in (3.12) has been computed the matrices $A_i$ and $L_i$ are not needed anymore. Hence the storage requirements for the method are very low.

4. The sparse approximate inverse has an optimality property related to the determinant: The functional $G \to K(GAG^T)$, $G \in S_{E^l}$, is minimal for $G_{E^l}$. From this the inequality (3.25) and the monotonicity result (3.26) follow.

5. From (3.25) it follows that $\prod_{i=1}^n \gamma_i^{\frac{2}{n}}$ is an upper bound for $d(A)$. $\square$

## 4. Monte Carlo methods for approximating $d(A)$.

In this section we describe two methods for approximating $d(A)$ that are known from the literature. Both methods are based on the following proposition [9, 3].

PROPOSITION 4.1. *Let $H$ be a symmetric matrix of order $n$ with $\mathrm{tr}(H) \neq 0$. Let $V$ be the discrete random variable which takes the values $1$ and $-1$ each with probability $0.5$ and let $z$ be a vector of $n$ independent samples from $V$. Then $z^T H z$ is an unbiased estimator of $\mathrm{tr}(H)$:*

$$E(z^T H z) = \mathrm{tr}(H)$$

*and*

$$var(z^T H z) = 2 \sum_{i \neq j} h_{ij}^2 \ .$$

Using the identity

$$d(A) = \det(A)^{\frac{1}{n}} = \exp\left(\frac{1}{n} \mathrm{tr}\ln(A)\right)$$

leads to the following Monte Carlo algorithm.

ALGORITHM 4.2.

```
For $j = 1, 2, \ldots, M$
    1. Generate $z_j \in \mathbb{R}^n$ with entries which are uniformly distributed in $(0,1)$.
    2. If $(z_j)_i < 0.5$, then $(z_j)_i := -1$, otherwise $(z_j)_i := 1$.
    3. Compute an approximation
(4.1)        $d_j \approx z_j^T \ln(A) z_j$ .
End. Compute
```

$$\hat{d}_M(A) = \exp\left(\frac{1}{n}\frac{1}{M}\sum_{j=1}^{M} d_j\right) \ .$$

In the following two subsections we describe methods for computing the approximation $d_j \approx z_j^T \ln(A) z_j$ in (4.1).

**4.1. Approximation of $z^T \ln(A) z$ using Chebyshev polynomials.** We describe a method that is presented in [16]. We assume that $A$ is scaled by a factor $0 < \frac{1}{b} \leq \frac{1}{\lambda_n}$. Then $\sigma(\frac{1}{b}A) \subset [\varepsilon, 1]$ holds with $0 < \varepsilon \leq \frac{\lambda_1}{b}$. For ease of notation this scaled matrix is denoted by $A$, too.

Let $T_k$, $k \geq 0$, be the Chebyshev polynomials on $[0,1]$:

$$T_{-1}(x) = 2x - 1, \quad T_0(x) = 1, \quad T_{k+1}(x) = (4x-2)T_k(x) - T_{k-1}(x) \quad \text{for} \quad k \geq 1 \ .$$

The method is based on the following expansion for $\ln x$:

$$\ln x = \sum_{k=0}^{m+1} b_k T_k\left(\frac{1-x}{1-\varepsilon}\right) + \delta \ln x \quad \text{for} \quad x \in [\varepsilon, 1], \tag{4.2}$$

$$|\delta| \leq 2e^{-2(m+1)\sqrt{\varepsilon}} \ . \tag{4.3}$$

We show that this result holds and derive a simple and cheap algorithm for the computation of the coefficients $b_k$. The starting point is the identity

$$\frac{1}{y}\Big(1 + \rho T_{m+1}\Big(\frac{1-y}{1-\varepsilon}\Big)\Big) = \sum_{k=0}^{m} c_k T_k\Big(\frac{1-y}{1-\varepsilon}\Big) \ , \quad y \in [\varepsilon, 1] \ , \tag{4.4}$$

with parameters $\rho$ and $c_k$, $0 \leq k \leq m$. With $z := \frac{1-y}{1-\varepsilon} \in [0,1]$ this is equivalent to

$$1 + \rho T_{m+1}(z) = (1 - (1-\varepsilon)z) \sum_{k=0}^{m} c_k T_k(z) \ . \tag{4.5}$$

Substituting $zT_k(z) = \frac{1}{4}T_{k+1}(z) + \frac{1}{2}T_k(z) + \frac{1}{4}T_{k-1}(z)$ in (4.5) and comparing the coefficients of $T_k$ on both sides of the equality results in a linear system of $m+2$ equations for the unknowns $c := (c_0, \ldots, c_m)^T$ and $\rho$. A simple calculation shows that the solution of this system is given by

$$c = \frac{4}{1-\varepsilon} B^{-1} e_1 \ , \tag{4.6}$$

$$\rho = -e_{m+1}^T B^{-1} e_1 \ , \tag{4.7}$$

with $e_1$ and $e_{m+1}$ the first and $(m+1)$st basis vector in $\mathbb{R}^{m+1}$, respectively, and

$$B = \begin{pmatrix} 2\gamma & -1 & & & & \\ -2 & 2\gamma & -1 & & \emptyset & \\ & -1 & 2\gamma & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & \emptyset & & \ddots & \ddots & -1 \\ & & & & -1 & 2\gamma \end{pmatrix} \in \mathbb{R}^{(m+1)\times(m+1)} \ , \quad \gamma := \frac{1+\varepsilon}{1-\varepsilon} \ .$$

Hence, the $LU$-decomposition of $B$ results in an efficient algorithm for computing the coefficients $c$ and $\rho$ in (4.6), (4.7). Elementary manipulations with difference equations yield explicit formulas for $B^{-1}e_1$. For example, for the last component of this vector one can derive the expression

$$-\rho = e_{m+1}^T B^{-1} e_1 = \frac{-2}{\lambda^{m+1} + \lambda^{-(m+1)}} , \quad \lambda := \gamma + \sqrt{\gamma^2 - 1} . \tag{4.8}$$

Such explicit expressions are given in [16] and offer an alternative (but probably somewhat less efficient) approach for computing $c$ and $\rho$.

From (4.4) and $|T_{m+1}(z)| \leq 1$ it follows that

$$-|\rho|\frac{1}{y} \leq \frac{1}{y} - \sum_{k=0}^{m} c_k T_k\Big(\frac{1-y}{1-\varepsilon}\Big) \leq |\rho|\frac{1}{y} , \qquad y \in [\varepsilon, 1] .$$

Integrating between $y = x \in [\varepsilon, 1]$ and $y = 1$ we obtain

$$|\rho| \ln x \leq -(1-\varepsilon)\sum_{k=0}^{m} c_k \int_0^{\frac{1-x}{1-\varepsilon}} T_k(z)\, dz - \ln x \leq -|\rho| \ln x , \quad x \in [\varepsilon, 1] . \tag{4.9}$$

Using $\int T_0 = \frac{1}{2}(T_0 + T_1)$, $\int T_1 = \frac{1}{8}(T_2 - T_0)$, $\int T_k = \frac{1}{4}(\frac{T_{k+1}}{k+1} - \frac{T_{k-1}}{k-1})$, $k \geq 2$, a straightforward computation yields

$$-(1-\varepsilon)\sum_{k=0}^{m} c_k \int_0^{\frac{1-x}{1-\varepsilon}} T_k(z)\, dz = \sum_{k=0}^{m+1} b_k T_k\Big(\frac{1-x}{1-\varepsilon}\Big) ,$$

$$\begin{aligned} \text{with} \quad b_k &= -\frac{1-\varepsilon}{4k} c_{k-1} , \quad k = m, m+1 , \\ b_k &= -\frac{1-\varepsilon}{4k}(c_{k-1} - c_{k+1}) , \quad 2 \leq k \leq m-1 , \\ b_1 &= -\frac{1-\varepsilon}{4}(2c_0 - c_2) , \\ b_0 &= -\sum_{k=1}^{m+1}(-1)^k b_k . \end{aligned} \tag{4.10}$$

Hence, using the values for the coefficients $c = (c_0, \ldots, c_m)^T$ from (4.6) the coefficients $b_k$ in (4.2) directly follow from (4.10). The bound on $\delta$ in (4.3) is a consequence of (4.9) and

$$|\rho| = \frac{2}{\lambda^{m+1} + \lambda^{-(m+1)}} \leq 2\lambda^{-(m+1)} \leq 2e^{-2(m+1)\sqrt{\varepsilon}} .$$

Now assume that the coefficients $b_k$ have been computed. For $z_j \in \mathbb{R}^n$ it follows that

$$z_j^T \ln(A) z_j \approx \sum_{k=0}^{m+1} b_k z_j^T T_k\Big(\frac{I-A}{1-\varepsilon}\Big) z_j =: d_j \tag{4.11}$$

can be used as an approximation in (4.1). The terms $z_j^T T_k\big(\frac{I-A}{1-\varepsilon}\big) z_j$ in (4.11) can be computed using the recursion for $T_k$. In our applications we have $n = \dim(A) \gg m$, and the costs for computing $d_j$ in (4.11) are dominated by the costs for the $m+1$ matrix-vector multiplications with the matrix $A$. These matrix-vector computations are easy to parallelize. Note, however, that the Monte Carlo Algorithm 4.2 and the computation of the sum in (4.11) are purely sequential processes.

**4.2. Approximation of $z^T \ln(A) z$ using quadrature.** In this subsection we recall the method from [3] for approximating $z^T \ln(A)z$, $z \in \mathbb{R}^n$. Let $Q^T \Lambda Q = A$ be the eigendecomposition of $A$ with $Q$ orthogonal, $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, $\lambda_1 \le \cdots \le \lambda_n$. For $z \in \mathbb{R}^n$ let $\tilde{z} = \frac{Qz}{\|z\|_2}$. Then we have

$$\frac{z^T \ln(A) z}{\|z\|_2^2} = \tilde{z}^T \ln(\Lambda)\tilde{z} = \sum_{i=1}^{n} \ln\lambda_i \, \tilde{z}_i{}^2 = \int_{\lambda_1}^{\lambda_n} \ln\lambda \, d\mu(\lambda) =: J \ , \tag{4.12}$$

where the measure $\mu(\lambda)$ is given by

$$\mu(\lambda) = \begin{cases} 0 & \text{if} \quad \lambda < \lambda_1 \ , \\ \sum_{j=1}^{i} \tilde{z}_j^2 & \text{if} \quad \lambda_i \le \lambda < \lambda_{i+1}, \qquad 1 \le i \le n-1 \ , \\ 1 & \text{if} \quad \lambda_n \le \lambda \ . \end{cases}$$

For approximating the integral in (4.12) one can use a Gauss-type quadrature rule. Several possibilities are treated in [3]. Here we use a Gauss–Radau method:

$$Q_N := \sum_{j=1}^{N} \omega_j \ln\theta_j + \nu_\tau \ln\tau \ ,$$

where the node $\tau$ is prescribed. We will consider $\tau \approx \lambda_1$ and $\tau \approx \lambda_n$. The weights $\omega_j$, $\nu_\tau$ and the nodes $\theta_j$ are unknown and to be determined. It is well known that the nodes and weights in the Gauss quadrature can be computed using the Lanczos method (cf. [4]). The Gauss–Radau quadrature is treated in [5]. For $f(x) = \ln x$ we have $f^{(2N+1)}(x) > 0$ for all $x > 0$, and from [5] it then follows that if $\tau \le \lambda_1$ ($\tau \ge \lambda_n$), the approximation $Q_N$ is a lower bound (upper bound) for $J$. In [3] the following algorithm for approximating $J$ is proposed. We assume that $\nu_1 \le \lambda_1$ and $\nu_2 \ge \lambda_n$ are given.

```
ALGORITHM 4.3.
$x_0 = z/\|z\|_2, \quad x_{-1} = 0, \quad \gamma_0 = 0$;
For $k = 1, 2, \ldots$ do:
    1. $\alpha_k = x_{k-1}^T A x_{k-1}$.
    2. $r_k = Ax_{k-1} - \alpha_k x_{k-1} - \gamma_{k-1} x_{k-2}$.
    3. $\gamma_k = \|r_k\|_2$.
    4. Let
```

$$T_k = \begin{pmatrix} \alpha_1 & \gamma_1 & & \emptyset & \\ \gamma_1 & \alpha_2 & \gamma_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \gamma_{k-1} \\ & \emptyset & & \gamma_{k-1} & \alpha_k \end{pmatrix} ,$$

$$\delta_m = \gamma_k^2 e_k^T (T_k - \nu_m I)^{-1} e_k, \quad m = 1, 2,$$

$$\hat{T}_k^{(m)} = \begin{pmatrix} T_k & \gamma_k e_k \\ \gamma_k e_k^T & \phi_m \end{pmatrix} , \quad \phi_m = \nu_m + \delta_m \ , \quad m = 1, 2 \ .$$

```
    5. Compute the eigenvalues $\theta_\ell^{(m)}$ and the first elements $\omega_\ell^{(m)}$ of the normalized
       eigenvectors of $\hat{T}_k^{(m)}$ $(m = 1, 2;\ 1 \le \ell \le k+1)$.
    6. $Q_k^{(m)} = \sum_{\ell=1}^{k+1} (\omega_\ell^{(m)})^2 \ln \theta_\ell^{(m)}$,  $m = 1, 2$.
    7. *If* $\frac{Q_k^{(2)} - Q_k^{(1)}}{|Q_k^{(1)}|} \le \mathrm{eps}$ *(user specified tolerance), then Stop.*
    8. $x_k = r_k/\gamma_k$.
*End. Compute*
```

$$d_z = \frac{1}{2}(Q_k^{(1)} + Q_k^{(2)})\|z\|_2^2 \ . \tag{4.13}$$

For $z = z_j$ as in Algorithm 4.2 the value $d_j := d_{z_j}$ from (4.13) is taken as the approximation in (4.1). As for the method in the previous subsection we have an outer (Monte Carlo) and inner iteration which are purely sequential operations. In our applications the dimensions of the eigenvalue problems that occur in Algorithm 4.3 are very small compared to $n = \dim(A)$, and the costs for one iteration in this algorithm are dominated by the matrix-vector multiplication with the matrix $A$.

Both in the algorithm in this subsection and in the algorithm in subsection 4.1 we need approximations of $\lambda_1$ and $\lambda_n$. It turns out that the performance of the algorithms is less sensitive to the accuracy of these approximations. In the numerical experiments we used a fixed (small) number of Lanczos iterations to compute these approximations.

## 5. Numerical experiments.

In this section we present some results of numerical experiments with the methods introduced in sections 3 and 4. All experiments are done using a MATLAB implementation.

*Experiment* 1 (discrete two-dimensional Laplacian). We consider the standard 5-point discrete Laplacian on a uniform square grid with $N$ mesh points in both directions, i.e., $n = N^2$. For this symmetric positive definite matrix the eigenvalues are known:

$$\lambda_{\nu\mu} = 4(N+1)^2 \left( \sin^2\left(\frac{\nu\pi}{2(N+1)}\right) + \sin^2\left(\frac{\mu\pi}{2(N+1)}\right)\right), \quad 1 \le \nu, \mu \le N \ . \tag{5.1}$$

For the choice of the sparsity pattern $E^l$ we use a simple approach:

$$E^l(k) := \{(i,j) \mid i \ge j \ \text{ and } \ (A^k)_{ij} \ne 0\} \ , \quad k = 1, 2, \dots \ . \tag{5.2}$$

We first describe some features of the methods for the case $N = 30$, $k = 2$, and after that we will vary $N$ and $k$. Let $A$ denote the discrete Laplacian for the case $N = 30$. For the matrices $A_i = P_i A P_i^T \in \mathbb{R}^{n_i \times n_i}$ $(i = 1, \dots, n)$ the dimensions $n_i$ are between 1 and 7; the mean of these dimensions is 6.7. Algorithm 3.3 yields an approximation,

$$d(G_{E^l(2)})^{-2} = d(\hat{G}_{E^l(2)})^{-1} = \prod_{i=1}^{n} \gamma_i^{\frac{2}{n}} = 3.2526\ 10^3$$

for $d(A) = 3.1379\ 10^3$. Hence the relative error is 3.5%. For the computation of the Cholesky factorizations $A_i = L_i L_i^T$, $i = 1, 2, \dots, n$, approximately 41 $10^3$ flops are needed (in the MATLAB implementation). If we compare this with the costs of one matrix-vector multiplication $A * x$ (8760 flops), denoted by MATVEC, it follows that for computing this approximation of $d(A)$, with error 3.5%, we need arithmetic work comparable to only 5 MATVEC. In Table 5.1 we give results for the discrete two-dimensional Laplacian with $N = 30$ $(n = 900)$, $N = 100$ $(n = 10000)$ and $N = 200$ $(n = 40000)$. We use the sparsity patterns $E^l(2)$ and $E^l(4)$. In the third column

TABLE 5.1
*Results for two-dimensional discrete Laplacian with $E^l = E^l(2)$.*

| $n$ | $d(A)$ | $d(G_{E^l(2)})^{-2}$ (error) | Costs for $d(G_{E^l(2)})^{-2}$ | $d(G_{E^l(4)})^{-2}$ (error) | Costs for $d(G_{E^l(4)})^{-2}$ |
|---|---|---|---|---|---|
| 900 | $3.138\ 10^3$ | $3.253\ 10^3$ (3.5%) | 5 MV | $3.177\ 10^3$ (1.2%) | 41 MV |
| 10000 | $3.292\ 10^4$ | $3.434\ 10^4$ (4.1%) | 5 MV | $3.347\ 10^3$ (1.6%) | 45 MV |
| 40000 | $1.300\ 10^5$ | $1.359\ 10^5$ (4.3%) | 5 MV | $1.323\ 10^3$ (1.7 %) | 46 MV |

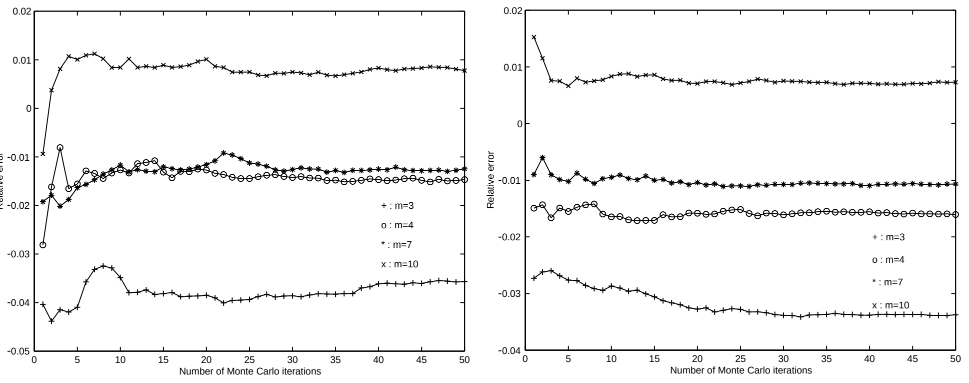

FIG. 5.1. *Algorithm* 4.2 *combined with the method from section* 4.1*:* $n = 10000$ *(left),* $n = 40000$ *(right).*

of this table we give the computed approximation of $d(A)$ and the corresponding relative error. In the fourth column we give the total arithmetic costs for the Cholesky factorization of the matrices $A_i$, $i = 1, 2, \dots, n$. In the columns 5 and 6 we give the results and corresponding arithmetic costs for the case with larger sparsity pattern $E^l(4)$.

Related to these numerical results we note the following. From the third and fourth column in Table 5.1 we see that using this method we can obtain an approximation of $d(A)$ with relative error only a few percent and arithmetic costs only a few MATVEC. Moreover, this efficiency hardly depends on the dimension $n$. Comparison of the third and fifth columns in Table 5.1 shows that the approximation significantly improves if we enlarge the pattern from $E^l(2)$ to $E^l(4)$. The corresponding arithmetic costs increase by a factor of about 9. This is caused by the fact that the mean of the dimensions of the systems $A_i$, $i = 1, 2, \dots, n$, increases from approximately 7 (for $E^l(2)$) to approximately 20 (for $E^l(4)$).

We also applied the Monte Carlo Algorithm 4.2, with $M = 50$, to this problem. If, for the approximation of $z_j^T \ln(A) z_j$ in (4.1), we use the approach based on Chebyshev polynomials, we obtain the results in Figure 5.1. It turns out that the bound in (4.3) is very pessimistic and should not be used to determine a value for the parameter $m$. In the experiments we used the values $m = 3, 4, 7, 10$. Note that the arithmetic costs in the inner Chebyshev iteration (4.11) are comparable to $m + 1$ MATVEC. From Figure 5.1 we see that for a relative error of approximately 1.5% it suffices to take 10–15 Monte Carlo iterations with $m = 4$. The arithmetic costs are then

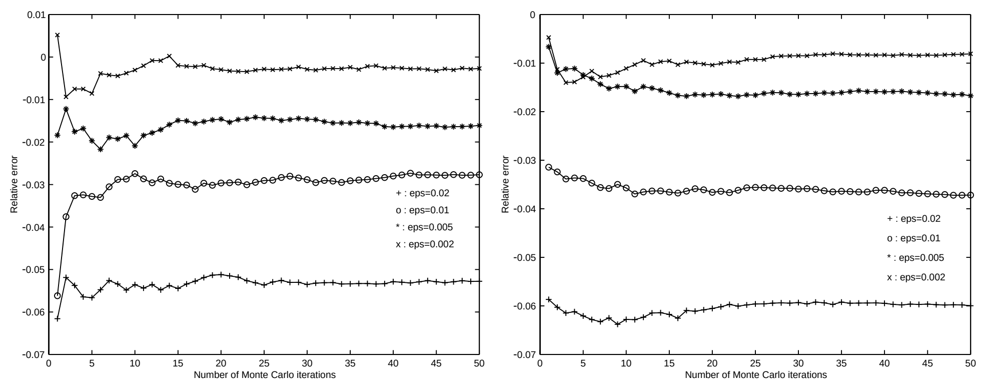

FIG. 5.2. *Algorithm* 4.2 *combined with the method from section* 4.2: $n = 10000$ *(left)*, $n = 40000$ *(right)*.

roughly 50–75 MATVEC. In Figure 5.2 results are shown if $z_j^T \ln(A) z_j$ in (4.1) is approximated using Algorithm 4.3. We used different tolerances in step 7 in this algorithm: eps $= 0.02$, $0.01$, $0.005$, $0.002$. The corresponding total number of matrix-vector multiplications is 188, 250, 306, 449 (for $n = 10000$) and 200, 250, 350, 501 (for $n = 40000$). We observe that for a relative error of approximately 1.5% about 10–15 Monte Carlo iterations with eps $= 0.005$ are sufficient. The arithmetic costs are then roughly 60–95 MATVEC.

Note that both Monte Carlo methods (in Figures 5.1 and 5.2) perform similarly. In both methods we need estimates for the extreme eigenvalues of the matrix $A$. We used the known values of these extreme eigenvalues given in (5.1).

*Experiment* 2 (MATLAB random sparse matrix). The sparsity structure of the matrices considered in Experiment 1 is very regular. In this experiment we consider matrices with a pattern of nonzero entries that is very irregular. We used the MATLAB generator ($\textsc{sprand}(n, n, 2/n)$) to generate a matrix $B$ of order $n$ with approximately $2n$ nonzero entries. These are uniformly distributed random entries in $(0, 1)$. The matrix $B^T B$ is then sparse symmetric positive semidefinite. In the generic case this matrix has many eigenvalues zero. To obtain a positive definite matrix we generated a random vector $d$ with all entries chosen from a uniform distribution on the interval $(0, 1)$ ($d := \textsc{rand}(n, 1)$). As a test matrix we used $A := B^T B + \mathrm{diag}(d)$. We performed numerical experiments similar to those in Experiment 1 above. We consider only the case with sparsity pattern $E^l = E^l(2)$. Results obtained with Algorithm 3.3 are shown in Table 5.2. From these results it is clear that for this random matrix $A$ the approximation of $d(A)$ based on the sparse approximate inverse is much better than for the discrete Laplacian in Experiment 1. This is related to the fact that for the random matrices considered in this example the preconditioned matrix $G_{E^l} A G_{E^l}$ turns out to be very well-conditioned.

We also apply the same Monte Carlo methods as discussed in Experiment 1 to these matrices. To allow a fair comparison we first rescaled the matrix $A$ with a diagonal matrix $D$ such that the absolute row sums of the matrix $DAD$ are all equal to one. Estimates of the extreme eigenvalues that are needed in these algorithms are obtained by applying 20 iterations of the Lanczos method (with starting vector $(1, \ldots, 1)^T$). The performance of these methods is similar for the three cases $n =$ 900, 10000, 40000. In Figure 5.3 we show the results for the case $n = 10000$.

Table 5.2
*Results for MATLAB random sparse matrices with $E^l = E^l(2)$.*

| $n$ | $d(A)$ | $d(G_{E^l})^{-2}$ (error) | Costs for $d(G_{E^l})^{-2}$ |
|---|---|---|---|
| 900 | 0.82453 | 0.82521 ($8.2\ 10^{-4}$) | 23 MV |
| 10000 | 0.80985 | 0.81053 ($8.4\ 10^{-4}$) | 18 MV |

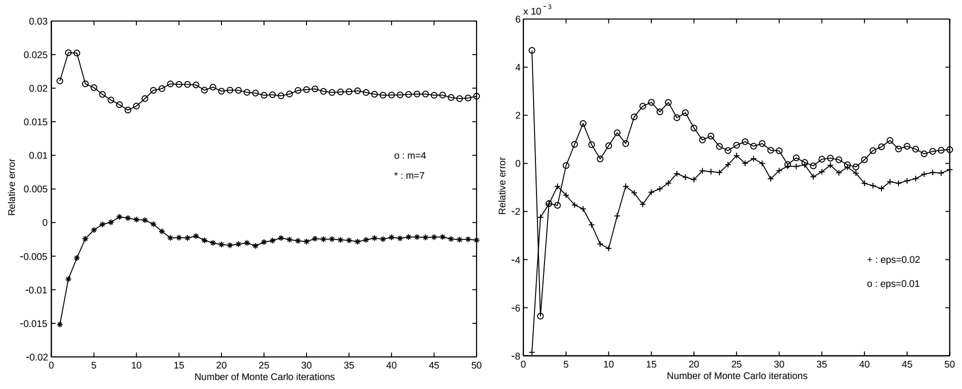

Fig. 5.3. *Algorithm* 4.2 *combined with the methods from section* 4.1 *(left) and from section* 4.2 *(right).*

For the Monte Carlo method using the approach based on Chebyshev polynomials the result after 20 iterations and $m = 7$ (cf. Figure 5.3, left) has a relative error $\approx 0.001$. For computing this result, approximately 180 MATVEC are needed. If we use the Gauss–Radau quadrature (cf. Figure 5.3, right) with eps $= 0.02$, then after 20 Monte Carlo iterations the result also has a relative error $\approx 0.001$. The total costs are about 100 MATVEC. Hence, in this example the method based on the sparse approximate inverse is more efficient than the Monte Carlo methods.

*Experiment* 3 (quantum chromodynamics (QCD) type matrix). In this experiment we consider a complex Hermitian positive definite matrix with a regular sparsity structure. This matrix is motivated by applications from the QCD field. In QCD simulations the determinant of the so-called Wilson fermion matrix is of interest. These matrices and some of their properties are discussed in [14, 13]. The Wilson fermion matrix $A = I - \kappa D$ describes a nearest neighbor coupling with periodic boundary conditions on a four-dimensional regular space-time lattice with lattice sites

$$\Omega_N = \{ (x_1, x_2, x_3, x_4) \mid x_i = 1, \ldots, n_i, \quad n_i = 2^{N_i} \} .$$

The so-called hopping matrix $D$ has the form

$$D_{x,y} = \sum_{\mu=1}^{4} \big((I - \gamma_\mu) \otimes U_\mu(x)\big)\delta_{x,y-e_\mu} + \big((I + \gamma_\mu) \otimes U_\mu^H(x - e_\mu)\big)\delta_{x,y+e_\mu} , \tag{5.3}$$

where $x, y$ are lattice sites from $\Omega_N$, $e_\mu$ is the $\mu$th basisvector in $\mathbb{R}^4$, and $\delta_{x,y} = 1$ (0) if $x = y$ ($x \neq y$). The matrices $I \pm \gamma_\mu \in \mathbb{C}^{4\times 4}$ are projectors onto two-dimensional subspaces and the matrices $U_\mu(x) \in \mathbb{C}^{3\times 3}$ are from $SU(3)$ (see [13] for

TABLE 5.3
*Results for QCD type matrix with $E^l = E^l(2)$.*

| $n$ | $d(A)$ | $d(G_{E^l})^{-2}$ (error) | Costs for $d(G_{E^l})^{-2}$ |
|---|---|---|---|
| 1024 | 0.8032 | 0.8248 (2.7%) | 22 MV |
| 4096 | 0.8037 | 0.8254 (2.7%) | 21 MV |

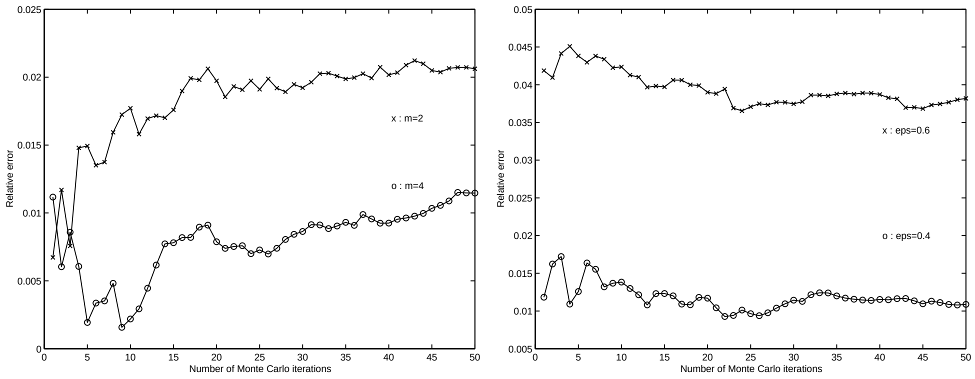

FIG. 5.4. *Algorithm* 4.2 *combined with the methods from section* 4.1 *(left) and from section* 4.2 *(right).*

details). Usually, these matrices $U_\mu(x)$ are generated randomly. In this model the matrix $D$ has a block structure with blocks $D_{x,y} \in \mathbb{C}^{12\times 12}$, $x, y \in \Omega_N$. Here we consider a very simple variant of this model. We take $\gamma_\mu = 0$, $I = 1$, $U_\mu(x) = \exp(2i\pi\alpha_\mu(x))$, where $\alpha_\mu(x)$ is chosen from a uniform distribution on the interval $(0, 1)$. Hence the couplings $D_{x,y}$ in (5.3) are complex scalars. Note that the matrix $D$ is Hermitian. Due to the randomly generated functions $\alpha_\mu(x)$ the couplings $D_{x,y}$ show a strong fluctuation as a function of $x$ and $y$. In QCD simulations the parameter $\kappa$ is taken such that the Wilson fermion matrix $A$ is positive definite and close to singular. In the experiment here we computed the largest eigenvalue $\rho_D$ of $D$ (using the MATLAB function EIGS) and set $\kappa := (1.01\,\rho_D)^{-1}$. We performed numerical experiments as in Experiment 1 with $E^l = E^l(2)$ for two cases: $(n_1, n_2, n_3, n_4) = (4, 4, 8, 8)$ and $(n_1, n_2, n_3, n_4) = (8, 8, 8, 8)$. The results are presented in Table 5.3. We also used the Monte Carlo methods. As in Experiment 2 we applied 20 Lanczos iterations to obtain estimates for the extreme eigenvalues. The results are shown in Figure 5.4. From this figure we see that after 20 Monte Carlo iterations using the method from subsection 4.1 with $m = 2$ the result has a relative error of about 2%. For computing this result approximately 80 MATVEC are needed. Using the method from subsection 4.2 with eps $= 0.4$ the result after 20 Monte Carlo iterations has a relative error of about 1%. The total costs for computing this result are about 80 MATVEC.

Note that in all three experiments the performance of the methods hardly depends on the dimension $n$. In all measurements for the arithmetic costs we did not take into account the costs of determining the sparsity pattern $E^l(k)$ and of building the matrices $P_iAP_i^T$.

We conclude that at least for these few model problems the new method can compete, even on a sequential machine, with the two Monte Carlo methods proposed in the literature. We believe that in a (massively) parallel environment the method based on the sparse approximate inverse can be expected to be much more efficient than the Monte Carlo techniques because the former is ideally suited for a parallel implementation.

*Remark* 5.1. In this paper we do not discuss the topic of error estimation. For the Monte Carlo method, error estimation techniques are treated in [3]. Related to the method based on the sparse approximate inverse (Algorithm 3.3) we briefly discuss one possible technique for a posteriori error estimation. From (3.25) we have the a priori error bound

$$\frac{d(A)}{d(G_{E^l})^{-2}} \leq 1 \; .$$

The exact error is given by

$$\frac{d(A)}{d(G_{E^l})^{-2}} = d(G_{E^l} A G_{E^l}^T) = d(\mathcal{E}_{E^l}) \; ,$$

where $\mathcal{E}_{E^l} := G_{E^l} A G_{E^l}^T$ is a sparse symmetric positive definite matrix. For ease of presentation we assume that the pattern $E^l$ is sufficiently large such that $\rho(I - \mathcal{E}_{E^l}) < 1$ holds. In [12] it is proved that if $A$ is an $M$-matrix or a (block) $H$-matrix, then this condition is satisfied for every lower triangular pattern $E^l$. For the exact error we obtain, using a Taylor expansion of $\ln(I - B)$ for $B \in \mathbb{R}^{n \times n}$ with $\rho(B) < 1$ (see [6]),

$$\begin{aligned} d(\mathcal{E}_{E^l}) &= \exp\left(\frac{1}{n}\ln(\det(\mathcal{E}_{E^l}))\right) = \exp\left(\frac{1}{n}\mathrm{tr}(\ln(\mathcal{E}_{E^l}))\right) \\ &= \exp\left(\frac{1}{n}\mathrm{tr}(\ln(I - (I - \mathcal{E}_{E^l})))\right) = \exp\left(-\frac{1}{n}\mathrm{tr}\left(\sum_{k=1}^{\infty} \frac{(I - \mathcal{E}_{E^l})^k}{k}\right)\right) \; . \end{aligned} \tag{5.4}$$

Hence, an error estimation can be based on estimates for the partial sums $S_m := \sum_{k=1}^{m} \frac{1}{k}\mathrm{tr}((I - \mathcal{E}_{E^l})^k)$. The construction of $G_{E^l}$ is such that $\mathrm{diag}(\mathcal{E}_{E^l}) = I$ (cf. (3.22)) and thus $\mathrm{tr}(\mathcal{E}_{E^l}) = n$ and $S_1 = 0$. For $S_2$ we have

$$S_2 = \frac{1}{2}\mathrm{tr}((I - \mathcal{E}_{E^l})^2) = \frac{1}{2}\mathrm{tr}(I - 2\mathcal{E}_{E^l} + \mathcal{E}_{E^l}^2) = -\frac{1}{2}n + \frac{1}{2}\mathrm{tr}(\mathcal{E}_{E^l}^2) \; . \tag{5.5}$$

For approximating the trace quantity $\mathrm{tr}(\mathcal{E}_{E^l}^2)$ in $S_2$ we can use the following Monte Carlo algorithm based on Proposition 4.1.

ALGORITHM 5.2.

```
For $j = 1, 2, \ldots, M$
    1. Generate $z_j \in \mathbb{R}^n$ with entries which are uniformly distributed in $(0, 1)$.
    2. If $(z_j)_i < 0.5$, then $(z_j)_i := -1$, otherwise $(z_j)_i := 1$.
    3. $y_j := \mathcal{E}_{E^l} z_j$,   $\alpha_j := y_j^T y_j$.
End.
```

This then yields

$$\hat{S}_2 := -\frac{1}{2}n + \frac{1}{2M}\sum_{j=1}^{M} \alpha_j \tag{5.6}$$

as an approximation for $S_2$. The corresponding error estimate is given by

$$E_2 = \exp\left(-\frac{1}{n}\hat{S}_2\right). \tag{5.7}$$

It turns out that, at least in our experiments, this technique yields satisfactory results. One clear disadvantage of this approach is that the matrix $G_{E^l}$ must be available (and thus stored). Note that for the computation of the approximation $d(G_{E^l})^{-2}$ of $d(A)$ we do not have to store the matrix $G_{E^l}$.

**Acknowledgment.** The author thanks the referees for a number of valuable comments that considerably improved the paper.

## REFERENCES

[1] O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, New York, 1994.
[2] Z. Bai and G. H. Golub, *Bounds on the trace of the inverse and the determinant of symmetric positive definite matrices*, Ann. Numer. Math., 4 (1997), pp. 29–38.
[3] Z. Bai, M. Fahey, and G. H. Golub, *Some large scale matrix computation problems*, J. Comput. Appl. Math., 74 (1996), pp. 71–89.
[4] P. Davis and P. Rabinowitz, *Methods of Numerical Integration*, Academic Press, New York, 1984.
[5] G. H. Golub, *Some modified matrix eigenvalue problems*, SIAM Rev., 15 (1973), pp. 318–334.
[6] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, 1996.
[7] M. J. Grote and T. Huckle, *Parallel preconditioning with sparse approximate inverses*, SIAM J. Sci. Comput., 18 (1997), pp. 838–853.
[8] M. Hasenbusch, *Speeding up finite step-size updating of full QCD on the lattice*, Phys. Rev. D, 59 (1999); available online as article 054505 from http://prd.aps.org/.
[9] M. Hutchinson, *A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines*, Commun. Statist. Simulation Comput., 18 (1989), pp. 1059–1076.
[10] I. E. Kaporin, *An alternative approach to estimating the convergence rate of the CG method*, in Numerical Methods and Software, Yu. A. Kuznetsov, ed., Department of Numerical Mathematics, USSR Academy of Sciences, Moscow, 1990, pp. 55–72 (in Russian).
[11] L. Yu. Kolotilina and A. Yu. Yeremin, *On a family of two-level preconditionings of the incomplete block factorization type*, Soviet J. Numer. Anal. Math. Modelling, 1 (1986), pp. 293–320.
[12] L. Yu. Kolotilina and A. Yu. Yeremin, *Factorized sparse approximate inverse preconditionings.* I: *Theory*, SIAM J. Matrix Anal. Appl., 14 (1993), pp. 45–58.
[13] B. Medeke, *On algebraic multilevel preconditioners in lattice gauge theory*, in Numerical Challenges in Lattice Quantum Chromodynamics, Lecture Notes in Comput. Sci. and Engrg. 15, A. Frommer, T. Lippert, B. Medeke, and K. Schilling, eds., Springer-Verlag, Berlin, 2000, pp. 99–114.
[14] I. Montvay and G. Münster, *Quantum Fields on a Lattice*, Cambridge University Press, Cambridge, 1994.
[15] D. Pollard, *Convergence of Stochastic Processes*, Springer-Verlag, New York, 1984.
[16] J. Sexton and D. Weingarten, *Error estimate for the valence approximation and for a systematic expansion of full QCD*, Phys. Rev. D, 55 (1997), pp. 4025–4035.
[17] C. Thron, S. J. Dong, K. F. Liu, and H. P. Ying, *Padé-$Z_2$ estimator of determinants*, Phys. Rev. D, 57 (1998), pp. 1642–1653.