Journal of Machine Learning Research 1 (2000) 1-48 Submitted 4/00; Published 10/00

Computing the log-determinant of symmetric, diagonally dominant matrices in near-linear time WORKING DRAFT

Timothy Hunter TJHUNTER@EECS.BERKELEY.EDU
Ahmed El Alaoui ELALAOUI@EECS.BERKELEY.EDU
Alexandre M. Bayen BAYEN@BERKELEY.EDU
Department of Electrical Engineering and Computer Sciences
University of California
Berkeley, CA 94720-1776, USA

Editor: Editor

Outline

1. Sampling log-determinants
   (a) Bernstein inequality
   (b) Algo preconditioned Monte Carlo
   (c) Sampling theorem
   (d) Corollary with SDD
   (e) Using approximate inversion
2. General method for SDD and Laplacian systems
   (a) Determinant of a Laplacian
   (b) Using a tree Laplacian as preconditioner
   (c) Ultra-sparsifiers
   (d) Bounds using stretch

1. Introduction

We consider the problem of computing the determinant of symmetric, diagonally dominant (SDD) matrices, i.e. real symmetric matrices A for which:

$$A_{ii} \geq \sum_{j \neq i} |A_{ij}|$$

The set of all such matrices of size $n \times n$ is denoted SDD_n, and the set of all symmetric real matrices is called $\mathcal{S}_n$. Call m the number of non-zero entries in A. We are interested in computing the determinant of sparse matrices, i.e. matrices for which $m \ll n^2$.

The best exact algorithm known for computing the determinant of general matrices, the Cholesky factorization, runs in a cubic complexity $O\left(n^3\right)$. Computing the factorization can be sped up for a few specific patterns such as trees, but no algorithm has been shown to work in a generic way for SDD_n, let alone general symmetric matrices. We present an algorithm that returns an

approximation of the logarithm of the determinant in time quasi-linear with the number of non-zero entries of A. More specifically, we show that our algorithm, `UltraLogDet`, computes an ϵ-approximation of the logarithm of the log-determinant with high probability and in expected time[1]:

$$\tilde{O}\left(m\epsilon^{-1}\log^3 n\log^2\left(\frac{n\kappa\left(B\right)}{\epsilon}\right)\right)$$

where κ_A is the condition number of A. This algorithm builds upon the work of Spielmann and Teng on *ultra-sparsifiers* Spielman and Teng (2009), and it critically exploits the recent improvements from Koutis, Miller and Peng Koutis et al. (2010). This is to our knowledge the first algorithm that presents a nearly linear complexity which depends neither on the condition number of A (except through a log-term) nor on a specific pattern for the non-zero coefficients of A.

The high complexity of the algorithm transpires through the large exponent of $\log\log n$. However, our algorithm will directly benefit from any improvement on ultra-sparsifiers. Given the considerable practical importance of such preconditioners, we expect some fast improvements in this area. Also, the bulk of the work is performed in a Monte Carlo procedure that is straightforward to parallelize. Furthermore, we also present simpler, non-optimal algorithms that compute upper and lower bounds of the logarithm of the determinant, and that may be of more immediate practical interest.

Background There are two approaches in numerical linear algebra to approximately compute a determinant (or the log of the determinant): by performing a (partial) Cholesky factorization of A, or by considering the trace of some power series.

As mentioned above, the Cholesky factorization performs a decomposition of the form: $A = PLDL^TP^T$ with P a permutation matrix, L a low-triangular matrix with 1 on the diagonal and D a diagonal matrix of non-negative coefficients. Then the log-determinant of A is simply[2]:

$$\log\left|A\right| = \sum_i \log D_{ii}$$

The complexity of dense Cholesky factorization for dense matrices is $O\left(n^3\right)$. Unfortunately, Cholesky factorization usually does not gain much from the knowledge of the sparsity pattern due to the *fill-in problem* (see Meurant (1999), section 3.2). There is one case, though, for which Cholesky factorization is efficient: if the sparsity pattern of A is a tree, then performing Cholesky factorization takes $O\left(n\right)$ time, and the matrix L is a banded matrix Liu (1990). If the sparsity pattern of A is not a tree, however, this advantageous decomposition does not hold anymore.

When the matrix A is close to the identity, more precisely when the spectral radius of $M = A - I$ is less than 1, one can use the remarkable Martin expansion of the log-determinant Martin (1992):

$$\log\left|A\right| = \mathrm{Tr}\left(\log A\right) \tag{1}$$

where $\log A$ is the matrix logarithm defined by the series expansion:

$$\log A = \sum_{i=0}^{\infty}\frac{\left(-1\right)^i}{i+1}M^i \tag{2}$$

The determinant can then be computed by a sum of traces of the power of M, and the rate of convergence of this series is driven by the spectral radius M. This line of reasoning has lead researchers to look for decompositions of A of the form $A = U + V$ with the determinant of U being easier to compute and $U^{-1}V + I$ having a small spectral radius. Then $\log\left|A\right| = \log\left|U\right| + \log\left|U^{-1}V + I\right|$. The most common decomposition U, V is in terms of block diagonal and off-diagonal

1. We use the notation $\tilde{O}$ to hide a factor at most $(\log\log n)^8$
2. We will use the $|\cdot|$ operator to denote the determinant, it will be clear from the context that it is different from the absolute value.

terms, which can then use Hadamard inequalities on the determinant to bound the error Ipsen and Lee (2006). Diagonal blocks also have the advantage of having determinants easy to compute. However, this approach requires some strong assumptions on the condition number of A, which may not hold in practice.

The trace approach is driven by *spectral properties* (the condition number) while the Cholesky approach is driven by *graphical* properties (the non-zero pattern). We propose to combine these two approaches by decomposing the problem with one component that is close to a tree (and is more amenable to Cholesky methods), and one component that has a bounded condition number. Our solution is to use a *spectral sparsifier* introduced by Spielman in Spielman and Teng (2008).

Applications. The problem of estimating determinants has important applications in spatial data analysis, statistical physics and statistics. In spatial statistics, it is often convenient to interpolate measurements in a 2-, 3- or 4-dimensional volume using a sparse Gaussian process, a technique known in the geospatial community as *kriging* **??**. Computing the optimal parameters of this Gaussian process involves repeated evaluations of the partition function, which is a log-determinant. In this context, a diagonally dominant matrix for the Gram matrix of the process corresponds to distant interactions between points of measure (which is verified in some contexts, see **?**). Determinants also play a crucial role in quantum physics and in theoretical physics. The wave function of a system of multiple fermion particles is an antisymmetric function which can be described as a determinant (Slatter determinant, **??**). In the theory of quantum chromodynamics (QCD), the interaction between particles can be discretized on a lattice, and the energy level of particles is the determinant of some functional operators over this lattice **?**. It is itself a very complex problem because of the size of the matrices involved for any non-trivial problem, for which the number of variables is typically in the millions **?**. In this setting, the restriction to diagonally dominant matrices can be interpreted as an interaction between relatively massive particles **?**, or as a bound on the propagation of interactions between sites in the lattice **?**.

For these reasons, computing estimates of the log-determinant has been an active problem in physics and statistics. In particular, the Martin expansion presented in Equation 1 is extensively used in quantum physics Ipsen and Lee (2006), and it can be combined with sampling method to estimate the trace of a matrix series (Zhang et al. (2008),McCourt (2008),Zhang and Leithead (2007)). Another different line of research has worked on bounds on the values of the determinant itself. This is deeply connected to simplifying statistical models using variational methods. Such a relaxation using a message-passing technique is presented in Wainwright and Jordan (2006). Our method is close in spirit to Reuksen's work Reusken (2002) by the use of a preconditioner. However, Reuksen considers preconditioners based on a clever approximation of the Cholesky decomposition, and its interaction with the eigenvalues of the complete matrix is not well understood. Using simpler methods based on sampling, we are able to carefully control the spectrum of the reminder, which in turn leads to strong convergence guarantees.

Main results. We first present some general results about the preconditioning of determinants. Consider $A \in SDD_n$ invertible, and some other matrix $B \in SDD_n$ that is close to A in the spectral sense. All the results of this article stem from observing that:

$$\begin{aligned} \log|A| &= \log|B| + \log\left|B^{-1}A\right| \\ &\log|B| + \mathrm{Tr}\left(\log\left(B^{-1}A\right)\right) \end{aligned}$$

The first section is concerned with estimating the reminder term $\mathrm{Tr}\left(\log\left(B^{-1}A\right)\right)$ using the Martin expansion. The exact inverse B^{-1} is usually not available, but we are given instead a linear operator C that is an $\epsilon-$approximation of B^{-1}, for example using a conjugate gradient method. We show in Section 2 that if the precision of this approximation is high enough, we can estimate the reminder with high probability and with a reasonable number of calls to the operator C (this sentence will be made precise in the rather technical Theorem 6). Using this general framework, the subsequent Section 3.1 shows that spectral sparsifiers make excellent preconditioners that are close enough to

A and so that computing the Martin expansion is not too expansive. In particular, we build upon the recursive structure of Spielman-Teng ultra-sparsifiers to obtain our main result:

Theorem 1 *On input $A \in SDD_n$ with m non-zeros, $\epsilon > n^{-1}$, $\eta > 0$, the algorithm `UltraLogDet` returns a scalar z so that:*

$$\mathbb{P}\left[\left|z - n^{-1}\log|A|\right| > \epsilon\right] \leq \eta$$

and this algorithm completes in expected time $\tilde{O}\left(m\epsilon^{-1}\log^2 n \log^2\left(\frac{\kappa(B)}{\epsilon}\right)\log\left(\eta^{-1}\right)\right)$.

A note on scaling. Unlike other common characteristics of linear operators, the determinant and the log-determinant are very sensitive to dimensionality. We will follow the approach of Reuksen Reusken (2002) and consider the *regularized log-determinant* $f(A) = n^{-1}\log|A|$ instead of the log-determinant. The regularized determinant has appealing properties with respect to dimensionality. In particular, its sensitivity to perturbations does not increase with the dimensionality, but only depends on spectral properties of the operator A. For example, calling $\lambda_{\min}$ and $\lambda_{\max}$ the minimum and maximum eigenvalues of A, respectively:

$$\log\lambda_{\min} \leq f(A) \leq \log\lambda_{\max}$$
$$|f(A+\epsilon I) - f(A)| \leq \epsilon\left\|A^{-1}\right\|_2 + O\left(\epsilon^2\right)$$

The last inequality in particular shows that any perturbation to $\log|A|$ will be in the order $O(n)$, and so that all the interesting log-determinants in practice will be dominated by some $O(n)$.

The rest of the article is structured as follows. In the next section, we present some results about estimating the log-determinant from a truncated expansion. These results will justify the use of *preconditioners* to compute the determinant of a matrix. The techniques developed by Spielman et al. work on the Laplacians of weighted graphs. Section 3 introduces some new concepts to expand the notion of determinants to Laplacian matrices, and presents a few straightforward results in the relations between graph Laplacians and SDD matrices. Section 3.2 will use these new concepts to introduce a first family of preconditioners based on low-stretch spanning trees. Finally, Section 3.3 contains the proof of our main result, an algorithm to compute determinants in near-linear time.

2. Preconditioned log-determinants

We begin by a close inspection of a simple sampling algorithm to compute log-determinants, presented first in Barry and Pace (1999). We will first present some error bounds on this algorithm that expand on bounds previously presented in Bai et al. (1996) and Barry and Pace (1999). This section considers general symmetric matrices and does not make assumptions about diagonal dominance.

Consider a real symmetric matrix $S \in \mathcal{S}_n^+$ such that its spectral radius is less than 1: $0 \preceq S \preceq (1-\delta)I$ for some $\delta \in (0,1)$. Our goal is to compute $\log|I-S|$ up to precision ϵ and with high probability. From the Martin expansion:

$$\log|I-S| = -\mathrm{Tr}\left(\sum_{k=1}^{\infty}\frac{1}{k}S^k\right) \tag{3}$$

This series of traces can be estimated by Monte Carlo sampling, up to precision ϵ with high probability, by truncating the series and by replacing the exact trace evaluation by $x^T S^k x$ for some suitably chosen random variables x. In order to bound the errors, we will bound the large deviation errors using the following Bernstein inequality:

Lemma 2 *Let $X_1 \cdots X_U$ be independent random variables with $\mathbb{E}[X_u] = 0$, $|X_u| < c$. Call $\sigma^2 = \frac{1}{U}\sum_u Var(X_u)$, then for all $\epsilon > 0$:*

$$\mathbb{P}\left[\frac{1}{U}\sum_u X_u \geq \epsilon\right] \leq \exp\left(-\frac{U\epsilon^2}{2\sigma^2 + 2c\epsilon/3}\right)$$

The proof of this result can be found in a lecture note from Peter Bartlett's class (stat241B/spring 2003). We can adapt some results from Barry and Pace (1999) to prove this bound on the deviation from the trace.

Lemma 3 *Consider $H \in \mathcal{S}_n$ with the assumption $\lambda_{min} I_n \preceq H \preceq \lambda_{max} I$. Consider p vectors sampled from the standard Normal distribution: $\mathbf{u}_i \sim \mathcal{N}(\mathbf{0}, I_n)$ for $i = 1 \cdots p$. Then for all $\epsilon > 0$:*

$$\mathbb{P}\left[\left|\frac{1}{p}\sum_{i=1}^{p}\frac{\mathbf{u}_i^T H \mathbf{u}_i}{\mathbf{u}_i^T \mathbf{u}_i} - \frac{1}{n}Tr(H)\right| \geq \epsilon\right] \leq \exp\left(-\frac{p\epsilon^2}{2n^{-1}\left(\lambda_{\max}-\lambda_{\min}\right)^2 + \left(\lambda_{\max}-\lambda_{\min}\right)\epsilon}\right)$$

Proof The distribution of $\mathbf{u}_i$ is invariant through a rotation, so we can consider H diagonal. We assume without loss of generality that $H = \operatorname{diag}(\lambda_1 ... \lambda_n)$. Again without loss of generality, we assume that $\lambda'_{\max} = \lambda_{\max} - \lambda_{\min}$ and $\lambda'_{\min} = 0$ (by considering $H' = H - \lambda_{\min} I$). Call $V_i = \frac{\mathbf{u}_i^T H \mathbf{u}_i}{\mathbf{u}_i^T \mathbf{u}_i} - n^{-1}\mathrm{Tr}(H)$. Using results from Barry and Pace (1999), we have: $|V_i| \leq \lambda_{\max} - \lambda_{\min}$, $\mathbb{E}[V_i] = 0$ and:

$$\mathrm{Var}(V_i) = \frac{2}{n^2}\sum_{i=1}^{n}\left(\lambda_i - n^{-1}\mathrm{Tr}(H)\right)^2$$

Each of the variables V_i is independent (see Barry and Pace (1999)), so invoking Lemma 2 gives:

$$\mathbb{P}\left[\frac{1}{p}\sum_{i=1}^{p} V_i \geq \epsilon\right] \leq \exp\left(-\frac{p\epsilon^2}{2\sigma^2 + 2\left(\lambda_{\max}-\lambda_{\min}\right)\epsilon/3}\right)$$

with $\sigma^2 = \frac{2}{n^2}\sum_{i=1}^{n}\left(\lambda_i - n^{-1}\mathrm{Tr}(H)\right)^2 \leq \frac{2}{n^2}\sum_{i=1}^{n}\left(\lambda_{\max}-\lambda_{\min}\right)^2 = 2n^{-1}\left(\lambda_{\max}-\lambda_{\min}\right)^2$. Dropping the factor $2/3$, we get our result. ■

The previous lemma shows that if the eigenspectrum of a matrix is bounded, we can obtain a Hoeffding bound on the error done by sampling the trace. Furthermore, the convergence of the series 3 is also determined by the extremal eigenvalues of S. If we truncate the series (3), we can bound the truncation error using the extremal eigenvalues. We formalize this intuition in the following theorem, which is adapted from the main Theorem in Barry and Pace (1999). While that main Theorem in Barry and Pace (1999) only considered a confidence interval based on the covariance properties of Gaussian distribution, we generalize this result to a more general Bernstein bound.

Theorem 4 *Consider $S \in \mathcal{S}_n^+$ with $0 \preceq S \preceq (1-\delta) I$ for some $\delta \in (0,1)$. Call $y = n^{-1}\log|I - S|$ the quantity to estimate, and consider $\mathbf{u}_i \sim \mathcal{N}(\mathbf{0}, I_n)$ for $i = 1 \cdots p$ all independent. Call $\hat{y}_{p,l}$ an estimator of the truncated series of l elements computed by sampling the trace using p samples:*

$$\hat{y}_{p,l} = \frac{1}{p}\sum_{j=1}^{p}\sum_{k=1}^{l}\frac{1}{k}\frac{\mathbf{u}_j^T S^k \mathbf{u}_j}{\mathbf{u}_j^T \mathbf{u}_j}$$

Given $\epsilon > 0$ and $\eta \in (0,1)$, the $\hat{y}_{p,l}$ approximates y up to precision ϵ with probability at least $1-\eta$ by choosing $p \geq 8\left(\frac{1}{\epsilon} + \frac{1}{n\epsilon^2}\right)\log\left(\eta^{-1}\right)\log^2\left(\delta^{-1}\right)$ and $l \geq 2\delta^{-1}\log\left(\frac{n}{\delta\epsilon}\right)$:

$$\mathbb{P}\left[|y - \hat{y}_{p,l}| \geq \epsilon\right] \leq \eta$$

The proof of this result is detailed in Appendix A.

From this theoem we derive two results that justify the notion of preconditioners for determinants: one for exact preconditioners and one for approximate preconditioners. The corresponding algorithm, which we call `PreconditionedLogDetMonteCarlo`, is presented in Algorithm 2.

Corollary 5 *Let $A \in \mathcal{S}_n^+$ and $B \in \mathcal{S}_n^+$ be positive definite matrices so that B is a $\kappa-$approximation of A:*

$$A \preceq B \preceq \kappa A \tag{4}$$

Given $\epsilon > 0$ and $\eta \in (0,1)$, the algorithm `PreconditionedLogDetMonteCarlo` *computes $\frac{1}{n}\log\left|B^{-1}A\right|$ up to precision ϵ with probability greater than $1-\eta$, by performing $\mathcal{O}\left(\kappa\left(\frac{1}{\epsilon}+\frac{1}{n\epsilon^2}\right)\log^2(\kappa)\log\left(\frac{n\kappa}{\epsilon}\right)\log\left(\eta^{-1}\right)\right)$ vector inversions from B and vector multiplies from A.*

The proof of this corollary is presented in Appendix A. Usually, computing the exact inverse by an SDD matrix is too expensive. We can instead extend the previous result to consider a black box procedure that approximatively computes $B^{-1}x$. If the error introduced by the approximate inversion is small enough, the result from the previous corollary still holds. This is what the following theorem establishes:

Theorem 6 *Consider $A, B \in \mathcal{S}_n^+$ positive definite with B a $\kappa-$approximation of A with $\kappa \geq 2$. Furthermore, assume there exists a linear operator C so that for all $y \in \mathbb{R}^n$, C returns an $\nu-$approximation of $B^{-1}y$:*

$$\left\|C(y) - B^{-1}y\right\|_B \leq \nu \left\|B^{-1}y\right\|_B$$

Given $\eta \in (0,1)$ and $\epsilon > 0$, if $\nu \leq \min\left(\frac{\epsilon}{8\kappa^3\kappa(B)}, \frac{1}{2\kappa}\right)$, then the algorithm `ApproxPreconditionedLogDetMonteCarlo` *returns a scalar z so that:*

$$\mathbb{P}\left[\left|z - n^{-1}\log\left|B^{-1}A\right|\right| \geq \epsilon\right] \leq \eta$$

by performing $64\kappa\left(\frac{1}{\epsilon}+\frac{1}{n\epsilon^2}\right)\log^2(\kappa)\log\left(\frac{n\kappa}{\epsilon}\right)\log\left(\eta^{-1}\right)$ vector calls to the operator C and vector multiplies from A.

The proof of this result is detailed in Appendix A. While the overall bound looks the same, the constant (taken away by the $O()$ notation) is twice as large as Corollary 5.

This last theorem shows that we can compute a good approximation of the log-determinant if the preconditioner B: (a) is close to A in the spectral sense, and (b) can be approximately inverted and the error introduced by the approximate inversion can be controlled. This happens to be the case for symmetric, diagonally dominant matrices.

Algorithm **UltraLogDet**(A,ϵ,η):
If A is of a small size (<100), directly compute $\mathrm{ld}(A)$ with a dense Cholesky factorization.
Compute B =**IncrementalSparsify**(A)
Compute D, A' =**PartialCholesky**(B)
$\eta \leftarrow \min\left(\frac{\epsilon}{8\kappa^3\kappa(B)}, \frac{1}{2\kappa}\right)$
$p \leftarrow 8\left(\frac{1}{\epsilon}+\frac{1}{n\epsilon^2}\right)\log\left(\eta^{-1}\right)\log^2\left(\delta^{-1}\right)$
$l \leftarrow \delta^{-1}\log\left(\frac{2}{\epsilon\delta}\right)$
Compute s =**PreconditionedLogDetMonteCarlo**(B, A, η, p, l)
Return $s + \log|D|$ +**UltraLogDet**(A',ϵ,η)

Algorithm 1: Sketch of the main algorithm

3. Ultra-sparsifiers as determinant preconditioners

3.1 Making the problem laplacian

From now on, we consider all matrices to be in SDD_n. The techniques we will develop work on Laplacian matrices instead of SDD matrices. An SDD matrix is positive semi-definite while a Laplacian matrix is always singular, since its nullspace is spanned by $\mathbf{1}$. We generalize the definition of the determinant to handle this technicality.

Algorithm **PreconditionedLogDetMonteCarlo**(B,A,η,p,l):
$y \leftarrow 0$
for j from 1 to p:
  Sample $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, I)$
  $\mathbf{v} \leftarrow \mathbf{u}/\|\mathbf{u}\|$
  $z \leftarrow 0$
  for k from 1 to l:
    $\mathbf{v} \leftarrow B^{-1}A\mathbf{v}$ up to precision η
    $z \leftarrow z + k^{-1}\mathbf{v}^T\mathbf{u}$
  $y \leftarrow y - p^{-1}z$
Return y

Algorithm 2: PreconditionedLogDetMonteCarlo

Definition 7 Pseudo-log-determinant (PLD): *Let $A \in \mathcal{S}^{n+}$ be a non-null positive semi-definite matrix. The pseudo-log-determinant is defined by the sum of the logarithms of all the positive eigenvalues:*

$$ld(A) = \sum_{\lambda_i > 0} \log(\lambda_i)$$

where λ_i are the eigenvalues of A.

The interest of the PLD lies in the connection between SDD matrices and their associated Laplacian. Recall that an SDD matrix can be transformed into a Laplacian (which has all off-diagonal terms non-positive, and for which the sum of each row and each column is zero). As underlined in Gremban (1996), solving an equation involving an SDD can be turned into solving an equation with an SDD of twice the same size in which all the off-diagonal terms are non-positive. This transform is also useful for computing determinants. The following lemma shows that computing the log-determinant of an SDD with positive off-diagonal entries can be turned into computing the log-determinants of two SDD matrices with non-positive off-diagonal entries.

Lemma 8 *Consider the decomposition of a SDD matrix into a positive and a negative sum:*

$$A = P - N$$

with $P_{ij} \leq 0$ for $i \neq j$ and $N_{ij} \leq 0$ for all i, j so that $|A_{ij}| = |P_{ij}| + |N_{ij}|$. Then the matrices $\begin{pmatrix} P & N \\ N & P \end{pmatrix} \in SDD_{2n}$ and $P + N \in SDD_n$ have all the off-diagonal coefficients non-positive, and we have the relation:

$$\log\left|\begin{pmatrix} P & N \\ N & P \end{pmatrix}\right| = \log|P + N| + \log|P - N|$$

Proof Call $B = N + P$ and $C = \begin{pmatrix} P & N \\ N & P \end{pmatrix}$. Consider μ an eigenvalue of A, and x an associated eigenvector: $Ax = \mu x$. Then:

$$\begin{pmatrix} P & N \\ N & P \end{pmatrix}\begin{pmatrix} x \\ -x \end{pmatrix} = \begin{pmatrix} Ax \\ -Ax \end{pmatrix} = \mu\begin{pmatrix} x \\ -x \end{pmatrix}$$

which shows that the vector $\begin{pmatrix} x \\ -x \end{pmatrix}$ is an eigenvector of C with associated eigenvalue μ . Similarly, if η is an eigenvector of B with associated eigenvalue y, then η is also an eigenvector of C with associated eigenvalue $\begin{pmatrix} y \\ y \end{pmatrix}$. Since C is exactly of size $2n$, the set of eigenvalues of C is exactly the

concatenation of sets of eigenvalues of A and B. Thus $\log|C| = \sum_i \log\mu_i + \sum_i \log\eta_i = \log|A| + \log|B|$.

■

Furthermore, computing the determinant of SDD matrices with non-negative off-diagonal entries is equivalent to computing the PLD of Laplacian. Recall that a Laplacian matrix can be constructed from any SDD with non-positive off-diagonal entries matrix by adding a "ground" node (see Gremban's thesis Gremban (1996), Chapter 4).

Definition 9 Augmented Laplacian of an SDD matrix. *Let $A \in SDD_n$. Let L, D be the decomposition of A into L a Laplacian matrix and D a (non-negative) diagonal matrix such that $A = L + D$. The augmented matrix of A is:*

$$L_A = \begin{pmatrix} A & -\mathbf{d} \\ -\mathbf{d}^T & h \end{pmatrix}$$

with $\mathbf{d} \in \mathbb{R}^n$: $\mathbf{d}_i = D_{ii}$ and $h = \sum_i D_{ii}$.

From now on, given any $A \in SDD_n$, we will implicitly define a Laplacian L_A associated to it, using the augmented Laplacian. This process is also reversible: to any Laplacian L we can associate a unique positive definite matrix F_L (up to a permutation), and this transform preserves eigenvalues and matrix inequalities. We call this process "floating" of the Laplacian, by analogy to the "grounding" in the electrical sense of the SDD matrix as a Laplacian.

Definition 10 Floating a Laplacian. *Consider L a Laplacian matrix. Call F_L the matrix formed by removing the last row and the last column from L.*

The following lemma shows that the Laplacian matrix overdetermines a system, and that no information is lost by floating it.

Lemma 11 *Consider Z a (weighted) Laplacian matrix of a connected graph, then:*

1. *Idempotence: $L_{F_Z} = Z$*
2. *The eigenvalues of F_Z are the positive eigenvalues of Z, and the corresponding eigenvectors for F_Z are the same eigenvectors, truncated by the last coefficient.*
3. *$ld(Z) = \log|F_Z|$*
4. *Given Z_1, Z_2 Laplacian matrices, we have $Z_1 \succeq Z_2 \Rightarrow F_{Z_1} \succeq F_{Z_2}$*

The proof of this lemma is in Appendix B.

Using the floating and the grounding procedures, we can transform the PLD of a Laplacian matrix into the log-determinant of a SDD matrix. The following proposition establishes this equivalence.

Lemma 12 *Let $A \in SDD_n$, and L_A its augmented Laplacian.*

- *$ld(L_A) = \log|A|$*
- *Let $A, B \in SDD_n$. $ld(L_A L_B) = ld(L_A) + ld(L_B)$ and $ld\left(L_B^+ L_A\right) = \log\left|B^{-1}A\right|$ with $L_A^+$the pseudo inverse of L_A.*

Proof The proofs of this lemma are straightforward and are contained in Appendix B. ■

A Laplacian matrix can be considered either for its graphical properties, or for its algebraic properties. Recent results has shown a deep connection between these two aspects, and let us

develop a general framework for computing determinants: given $G \in SDD_n$, we consider the graph associated to its Laplacian L_G, through the grounding procedure. Using graphical properties of L_G, we can construct a weighted subgraph $L_H \subseteq L_G$ for which the PLD is easier to compute and that is a good approximation of $L_G$in the spectral sense. We can float the subgraph L_H and apply results of Section 2 to approximate the reminder with high probability. More precisely:

$$\begin{aligned}\log|G| &= \mathrm{ld}(L_H) - \log|F_{L_H}| + \log|G| \\ &= \mathrm{ld}(L_H) + \log\left|F_{L_H}^{-1}G\right|\end{aligned}$$

The first term $\mathrm{ld}(L_H)$ is usually easier to compute by considering the graphical properties of L_H, while the reminder $\log\left|F_{L_H}^{-1}G\right|$ is approximated by sampling. Preconditioner graphs L_H are typically efficient to factorize using Cholesky factorization, and close enough to G that the sampling procedure from the previous section can be applied to compute $\log\left|F_{L_H}^{-1}G\right|$. We will see how to adapt Spielman and Teng's remarkable work on *ultra-sparsifiers* to produce good preconditioners H for the determinant.

3.2 A first preconditioner

While the results in this section are not the main claims of this paper, we hope they will provide some intuition, and an easier path towards an implementation.

We present a first preconditioner that is not optimal, but that will motivate our results for stronger preconditioners: a tree that spans the graph G. Every graph has a low-stretch spanning tree, as discovered by Alon et al. Alon et al. (1995). The bound of Alon et al. was then improved by Abraham et al. Abraham et al. (2008). We restate their main result.

Lemma 13 *(Theorem 1 from Abraham et al. (2008)). Consider L_G a weighted graph. There exists a tree L_T that is a subgraph of L_G so that:*

$$L_G \preceq L_T \preceq \kappa L_G$$

with $\kappa = Cm\log n\log\log n\left(\log\log\log n\right)^3$ for some constant $C > 0$.

Proof This follows directly from Abraham et al. (2008). L_T is a subgraph of L_G (with the same weights on the edges), so $L_T \preceq L_G$ (see Spielman and Teng (2009) for example for a proof of this fact). Furthermore, we have $L_T \preceq \mathrm{st}_G(T)\, L_G$. This latter inequality is result of Boman and Hendrickson Boman et al. (2004) cited by Spielman et al. in Spielman (2010) that we generalize in Lemma 20. ■

Trees enjoy a lot of convenient properties for Gaussian elimination. The Cholesky factorization of a tree can be computed in linear time, and furthermore this factorization has a linear number of non-zero elements Spielman and Teng (2009). This factorization can be expressed as:

$$L_T = PLDL^TP^T$$

where P is a permutation matrix, L is a lower-triangular matrix with the diagonal being all ones, and D a diagonal matrix in which all the elements but the last one are positive, the last element being 0. These well-known facts about trees are presented in Spielman and Teng (2009). Once the Cholesky factorization of the tree is performed, the log-determinant of the original graph is an immediate by-product:

$$\log|L_T| = \sum_{i=1}^{n-1}\log D_{ii}$$

Furthermore, computing L_T^+x also takes $O(n)$ computations by forward-backward substitution. Applying Corollary 5 gives immediately:

Theorem 14 *Let G be a graph with n vertices and m edges. Its PLD can be computed up to a precision of ϵ and with high probability in time $\tilde{O}\left(mn\left(\epsilon^{-1}+\frac{1}{n\epsilon^2}\right)(\log m)^3(\log n)^2\log\epsilon^{-1}\right)$.*

Proof We assume G is connected, hence $m \geq n$ and $\log\kappa = \tilde{\mathcal{O}}(m\log n)$. ■

This bound may be of independent interest since it requires relatively little machinery to compute, and it is an improvement already for graphs with small vertex degree ($m = \mathcal{O}\left(n^{1+o(1)}\right)$) over the Cholesky factorization of G. Also, note that the PLD of the tree constructed above provides an upper bound to the log-determinant of G since $L_G \preceq L_T$. We will see in Subsection 3.4 that we can compute a non-trivial lower bound as well.

3.3 Incremental sparsifiers

We can do better and achieve near-linear time by using ultra-sparsifiers. The main insight of our result is that the class preconditioners presented by Spielman and Teng are based on incomplete Cholesky factorization, and hence have a determinant that is relatively easy to compute, and furthermore that they are excellent spectral preconditioners, so the procedure `PreconditionedLogDetMonteCarlo` is efficient to apply. We reintroduce some concepts presented in Koutis et al. (2010) to present a self-contained result. The following paragraphs are well-known facts about Spielman-Teng preconditioners and have been presented in Koutis et al. (2010); Spielman and Teng (2009).

The central idea to the Spielman-Teng preconditioner is to sample $O(n)$ edges from the graph A, to form a subgraph B that is close to a tree (hence it is easy to compute some partial Cholesky factorization), yet it is close to the original A is the spectral sense ($A \preceq B \preceq \kappa A$), thanks to the additional edges. The partial Cholesky factorization is computed using the `GreedyElimination` algorithm presented in Koutis et al. (2010). In order for this section to be self-contained, we include here the main results of Section 4 in Spielman and Teng (2009).

Consider the Laplacian matrix L_B of the subgraph B. There exists an algorithm that computes the partial Cholesky factorization:

$$L_B = PLCL^TP^T$$

where:

- P is a permutation matrix
- L is a non-singular, low triangular matrix of the form

  $$L = \begin{pmatrix} L_{1,1} & 0 \\ L_{2,1} & I_{n_1} \end{pmatrix}$$

  with the diagonal of $L_{1,1}$ being all ones.
- C has the form

  $$C = \begin{pmatrix} D_{n-n_1} & 0 \\ 0 & L_{A_1} \end{pmatrix}$$

  and every row and column of L_{A_1} has at least 3 non-zero coefficients. Furthermore, L_{A_1} is itself Laplacian and:

  $$\mathrm{ld}(L_G) = \sum_1^{n-n_1} \log D_{ii} + \mathrm{ld}(L_{A_1})$$

The exact algorithm that achieves this factorization is called `GreedyElimination` and is presented in Koutis et al. (2010). Using this factorization, the PLD of the original Laplacian L_A is:

$$\begin{aligned} \mathrm{ld}\left(L_A\right) &= \mathrm{ld}\left(L_B\right) + \mathrm{ld}\left(B^+A\right) \\ &= \sum_{1}^{n-n_1} \log D_{ii} + \mathrm{ld}\left(A_1\right) + \mathrm{ld}\left(B^+A\right) \end{aligned} \tag{5}$$

Thus, we are left with solving a smaller problem A_1, and we approximate the value of $\mathrm{ld}\left(B^+A\right)$ using the algorithm `SampleLogDet`. ST preconditioners are appealing for this task: they guarantee that A_1 is substantially smaller than A, so the recursion completes in $O\left(\log\log n\right)$ steps. Furthermore, computing the vector product B^+Ax is itself efficient (in can be done approximated in near-linear time), so we can apply Theorem 6. We formalize the notion of chain of preconditioners by reintroducing some material from Koutis et al. (2010).

Definition 15 *Definition 7.1 from Koutis et al. (2010). $\kappa\left(n\right)$-good chain. $\mathcal{C} = \{A_1 = A, B_1, A_2, \ldots A_d\}$ a chain of graphs with n_i and m_i the number of vertices and edges of A_i. $\mathcal{C}$ is $\kappa\left(n\right)$-good for A if:*

1. $A_i \preceq B_i \preceq \kappa\left(n_i\right) A_i$
2. $A_{i+1} =$ *`GreedyElimination`*(B_i)
3. $m_i/m_{i+1} \geq c_r\sqrt{\kappa\left(n_i\right)}$ *for some constant c_r.*

Good chains exist, as found by Koutis, Miller and Peng:

Lemma 16 *(Lemma 7.3 from Koutis et al. (2010)) Given a graph A,* *`BuildChain`*(A, p) *from Koutis et al. (2010) produces a $\tilde{O}\left(\log^4 n\right)$-good chain for A with probability $1-p$. The algorithm runs in time*

$$\tilde{O}\left(\left(m\log n + n\log^2 n\right)\log\left(1/p\right)\right)$$

These chains furthermore can be used a good preconditioners for conjugate gradient and lead to near-linear algorithms for approximate inversion (Lemma 7.2 from Koutis et al. (2010)). This remarkable result has been significantly strenghtened in the previous years, so that SDD systems can be considered to be solved in (expected) linear time.

Lemma 17 *(Theorem 4.6 from **?**). Given $A \in SDD_n$ with m non-zero entries, $b \in \mathbb{R}^n$ and $\nu > 0$, a vector x such that $\left\|x - A^+b\right\|_A < \nu\left\|A^+b\right\|_A$ can be computed in expected time $\tilde{O}\left(m\log n\log\left(1/\nu\right)\right)$.*

It should now become clear how we can combine a $\kappa\left(n\right)$-good chain with the Algorithm `PreconditionedLogDetMonteC` We start by building a chain. The partial Cholesky factorizations at each step of the chain provide an upper bound to $\mathrm{ld}\left(A\right)$. We then refine this upper bound by running `PreconditionedLogDetMonteCarlo` at each state of the chain to approximate $\mathrm{ld}\left(B_i^+A_i\right)$ with high probability. The complete algorithm is presented in Algorithm 1. We now have all the tools required to prove Theorem 1.

Proof of Theorem 1. Using Lemma 16, consider $\mathcal{C} = \{A_1 = A, B_1, A_2, \ldots A_d\}$ a $\tilde{O}\left(\log^4 n\right)$-good chain for L_A, with $d = \mathcal{O}\left(\log\log n\right) = \tilde{O}\left(1\right)$. We use Equation 5 to compute the Cholesky terms for each stage of the chain, and we are left with estimating $\tilde{\mathcal{O}}\left(1\right)$ reminders $\mathrm{ld}\left(B_i^+A_i\right)$. By construction, $A_i \preceq B_i \preceq \kappa\left(n_i\right)A_i$ and by Lemma 17, there exists an operator C so that $\left\|C\left(b\right) - A^+b\right\|_A < \nu\left\|A^+b\right\|_A$ for all b with a choice of relative precision $\nu = \frac{\epsilon}{16\kappa^3\kappa(B)}$. By Theorem 6, each reminder $\mathrm{ld}\left(B_i^+A_i\right)$ can be approximated to precision ϵ with probability at least $1-\eta$ using Algorithm **??**. Furthermore, this algorithm works in expected time $\tilde{O}\left(m\log n\log\left(1/\nu\right)\kappa\left(\frac{1}{\epsilon}+\frac{1}{n\epsilon^2}\right)\log\left(\frac{n\kappa}{\nu}\right)\log^2\left(\kappa\right)\log\left(\eta^{-1}\right)\right) = \tilde{O}\left(m\left(\frac{1}{\epsilon}+\frac{1}{n\epsilon^2}\right)\log^2 n\log^2\left(\frac{\kappa(B)}{\epsilon}\right)\log\left(\eta^{-1}\right)\right)$. By a union bound, the result also holds on the sum of all the approximations of the reminders. We can simplify this bound a little by assuming that $\epsilon \geq n^{-1}$, which then becomes $\tilde{O}\left(m\epsilon^{-1}\log^2 n\log^2\left(\frac{\kappa(B)}{\epsilon}\right)\log\left(\eta^{-1}\right)\right)$.

3.4 Stretch bounds on preconditioners

How good is the estimate provided by the preconditioner? Intuitively, this depends on how well the preconditioner L_H approximates the graph L_G. This notion of quality of approximation can be formalized by the notion of *stretch*. This section presents a deterministic bound on the PLD of L_G based on the PLD of L_H and the stretch of G relative to H. This may be useful in practice as it gives a (tight) interval for the PLD before performing any Monte-Carlo estimation of the residual.

The stretch of a graph is usually defined with respect to a (spanning) tree. In our analysis, it is convenient and straightforward to generalize this definition to arbitrary graphs. To our knowledge, this straightforward extension is not considered in the litterature, so we feel compelled to properly introduce it.

Definition 18 Generalized stretch. *Consider $\mathcal{V}$ a set of vertices, $G = (\mathcal{V}, \mathcal{E}_G)$, $H = (\mathcal{V}, \mathcal{E}_H)$ connected graphs over the same set of vertices, and L_G, L_H their respective Laplacians. The stretch of G with respect to H is the sum of the effective resistances of each edge of graph G with repect to graph H,*

$$st_H(G) = \sum_{(u,v)\in\mathcal{E}_G} L_G(u,v)\,(\mathcal{X}_u - \mathcal{X}_v)^T L_H^+ (\mathcal{X}_u - \mathcal{X}_v)$$

with $\mathcal{X}_u \in \mathbb{R}^n$ the unit vector that is 1 at position u, and zero otherwise.

If the graph H is a tree, this is a standard definition of stretch, because the effective resistance $(\mathcal{X}_u - \mathcal{X}_v)^T L_H^+ (\mathcal{X}_u - \mathcal{X}_v)$ between vertices u and v is the sum of all resistances over the unique path between u and v (see Lemma 2.4 in Spielman and Woo (2009)). Furthermore, the arguments to prove Theorem 2.1 in Spielman and Woo (2009) carry over to our definition of stretch. For the sake of completeness, we include this result:

Lemma 19 *(Straightforward generalization of Theorem 2.1 in Spielman and Woo (2009)) Let $G = (\mathcal{V}, \mathcal{E}_G)$, $H = (\mathcal{V}, \mathcal{E}_H)$ be connected graphs over the same set of vertices, and L_G, L_H their respective Laplacians. Then:*

$$st_H(G) = Tr\left(L_H^+ L_G\right)$$

with L_H^+ the pseudo-inverse of L_H.

Proof

We denote $E(u,v)$ the Laplacian unit matrix that is 1 in position u,v: $E(u,v) = (\mathcal{X}_u - \mathcal{X}_v)(\mathcal{X}_u - \mathcal{X}_v)^T$. This is the same arguments as the original proof:

$$\begin{aligned}
\mathrm{Tr}\left(L_H^+ L_G\right) &= \sum_{(u,v)\in\mathcal{E}_G} L_G(u,v)\,\mathrm{Tr}\left(E(u,v)\,L_H^+\right) \\
&= \sum_{(u,v)\in\mathcal{E}_G} L_G(u,v)\,\mathrm{Tr}\left((\mathcal{X}_u - \mathcal{X}_v)(\mathcal{X}_u - \mathcal{X}_v)^T L_H^+\right) \\
&= \sum_{(u,v)\in\mathcal{E}_G} L_G(u,v)\,(\mathcal{X}_u - \mathcal{X}_v)^T L_H^+ (\mathcal{X}_u - \mathcal{X}_v) \\
&= \mathrm{st}_H(G)
\end{aligned}$$

■

A consequence is $\mathrm{st}_H(G) \geq \mathrm{Card}(\mathcal{E}_G) \geq n-1$ for connected G and H with $L_G \succeq L_H$. A number of properties of the stretch extend to general graphs using the generalized stretch. In particular, the stretch inequality (Lemma 8.2 in Spielman and Teng (2009)) can be generalized to arbitrary graphs (instead of spanning trees).

Lemma 20 *Let $G = (\mathcal{V}, \mathcal{E}_G)$, $H = (\mathcal{V}, \mathcal{E}_H)$ be connected graphs over the same set of vertices, and L_G, L_H their respective Laplacians. Then:*

$$L_G \preceq st_H(G) L_H$$

Proof The proof is very similar to that of Lemma 8.2 in Spielman and Woo (2009), except that the invocation of Lemma 8.1 is replaced by invoking Lemma 27 in Appendix B. The Laplacian G can be written as a linear combination of edge Laplacian matrices:

$$L_G = \sum_{e \in \mathcal{E}_G} \omega_e L(e) = \sum_{(u,v) \in \mathcal{E}_G} \omega_{(u,v)} (\mathcal{X}_u - \mathcal{X}_v) (\mathcal{X}_u - \mathcal{X}_v)^T$$

and using Lemma 27 gives:

$$(\mathcal{X}_u - \mathcal{X}_v) (\mathcal{X}_u - \mathcal{X}_v)^T \preceq (\mathcal{X}_u - \mathcal{X}_v)^T L_H^+ (\mathcal{X}_u - \mathcal{X}_v) L_H$$

By summing all the edge inequalities, we get:

$$\begin{aligned} L_G &\preceq \sum_{(u,v) \in \mathcal{E}_G} \omega_{(u,v)} (\mathcal{X}_u - \mathcal{X}_v)^T L_H^+ (\mathcal{X}_u - \mathcal{X}_v) L_H \\ &\preceq \mathrm{st}_H(G) L_H \end{aligned}$$

■

This bound is remarkable as it relates any pair of (connected) graphs, as opposed to spanning trees or subgraphs. An approximation of the generalized stretch can be quickly computed using a construct detailled in **?**, as we will see below. We now introduce the main result of this section: a bound on the PLD of L_G using the PLD of L_H and the stretch.

Theorem 21 *Let $G = (\mathcal{V}, \mathcal{E}_G)$, $H = (\mathcal{V}, \mathcal{E}_H)$ be connected graphs over the same set of vertices, and L_G, L_H their respective Laplacians. Assuming $L_H \preceq L_G$, then:*

$$ld(L_H) + (n-1) \log \left(\frac{st_H(G)}{n-1} \right) \geq ld(L_G) \geq ld(L_H) + \log(st_H(G) - n + 2) \tag{6}$$

This bound is tight.

Proof This is an application of Jensen's inequality on $\mathrm{ld}\left(L_H^+ L_G\right)$. We have $\mathrm{ld} L_G = \mathrm{ld} L_H + \mathrm{ld}\left(L_H^+ G\right)$ and $\mathrm{ld}\left(L_H^+ G\right) = \mathrm{ld}\left(\sqrt{L_H}^+ L_G \sqrt{L_H}^+\right)$ with $\sqrt{T}$ the matrix square root of T. From Lemma 28, we have the following inequality:

$$\begin{aligned} \mathrm{ld}\left(\sqrt{L_H}^+ L_G \sqrt{L_H}^+\right) &\leq (n-1) \log \left(\frac{\mathrm{Tr}\left(\sqrt{L_H}^+ L_G \sqrt{L_H}^+\right)}{n-1} \right) \\ &= (n-1) \log \left(\frac{\mathrm{Tr}\left(L_H^+ L_G\right)}{n-1} \right) \\ &= (n-1) \log \left(\frac{\mathrm{st}_H(G)}{n-1} \right) \end{aligned}$$

The latter equality is an application of Lemma 19.

The lower bound is slightly more involved. Call λ_i the positive eigenvalues of $\sqrt{L_H}^+ L_G \sqrt{L_H}^+$ and $\sigma = \mathrm{st}_H(G)$. We have $1 \leq \lambda_i$ from the asumption $L_H \preceq L_G$. By definition: $\mathrm{ld}\left(L_H^+ L_G\right) = \sum_i \log \lambda_i$. Furthermore, we know from Lemma 19 that $\sum_i \lambda_i = \sigma$. The upper and lower bounds on λ_i give:

$$\mathrm{ld}\left(L_H^+ L_G\right) \geq \begin{array}{c} \min \sum_i \log \lambda_i \\ 1 \leq \lambda_i \\ \sum_i \lambda_i = \sigma \end{array}$$

Since there are precisely $n-1$ positive eigenvalues λ_i, one can show that the minimization problem above has a unique minimum which is $\log(\sigma - n + 2)$.

To see that, consider the equivalent problem of minimizing $\sum_i \log(1+u_i)$ under the constraints $\sum_i u_i = \sigma - (n-1)$ and $u_i \geq 0$. Note that:

$$\sum_i \log(1+u_i) = \log\left(\prod_i [1+u_i]\right) = \log\left(1 + \sum_i u_i + \mathrm{Poly}(u)\right)$$

with $\mathrm{Poly}(u) \geq 0$ for all $u_i \geq 0$, so we get: $\sum_i \log(1+u_i) \geq \log(1+\sum_i u_i)$ and this inequality is tight for $u_1 = \sigma - (n-1)$ and $u_{i\geq 2} = 0$. Thus the vector $\lambda^* = (\sigma - n + 2, 1 \cdots 1)^T$ is (a) a solution to the minimization problem above, and (b) the objective value of any feasible vector λ is higher or equal. Thus, this is the solution (unique up to a permutation). Hence we have $\mathrm{ld}\left(L_H^+ L_G\right) \geq \sum_i \log \lambda_i^* = \log(\sigma - n + 2)$.

Finally, note that if G is a tree and $H = G$ (G is its own spanning tree), then $\mathrm{st}_H(G) = n-1$, which gives an equality. ■

Note that Lemma 20 gives us $L_H \preceq L_G \preceq \mathrm{st}_H(G) L_H$ which implies $\mathrm{ld}(L_H) \leq \mathrm{ld}(L_G) \leq \mathrm{ld}(L_H) + n \log \mathrm{st}_H(G)$. The inequalities in Theorem 21 are stronger. Interestingly, it does not make assumption on the topology of the graphs (such as L_H being a subset of L_G). Research on conditioners has focused so far on low-stretch approximations that are subgraphs of the original graph. It remains to be seen if some better preconditioners can be found with stretches in $\mathcal{O}(n)$ by considering more general graphs. In this case, the machinery developped in Section 3 would not be necessary.

From a practical perspective, the stretch can be calculated also with respect to the number of non-zero entries.

Lemma 22 *Let $G = (\mathcal{V}, \mathcal{E}_G)$, $H = (\mathcal{V}, \mathcal{E}_H)$ be connected graphs over the same set of vertices, and L_G, L_H their respective Laplacians. Call $r = \max_e L_H(e) / \min_e L_H(e)$. Given $\epsilon > 0$, there exists an algorithm that returns a scalar y so that:*

$$(1-\epsilon)\, st_H(G) \leq y \leq (1+\epsilon)\, st_H(G)$$

with high probability and in expected time $\tilde{\mathcal{O}}\left(m\epsilon^{-2} \log(rn)\right)$.

Proof This is a straightforward consequence of Theorem 2 in **?**. Once the effective resistance of an edge can be approximated in time $O\left(\log n/\epsilon^2\right)$, we can sum it and weight it by the conductance in G for each edge.

■

Comments

Since the bulk of the computations are performed in estimating the residue PLD, it would be interesting to see if this could be bypassed using better bounds based on the stretch.

Also, even if this algorithm presents a linear bound, it requires a fairly advanced machinery (ST solvers) that may limit its practicality. Some heuristic implementation, for example based on algebraic multi-grid methods, could be a first step in this direction.

The authors are much indebted to Satish Rao and Jim Demmel for suggesting the original idea and their helpful comments on the draft of this article.

Appendix A: Proofs of Section 2

3.5 Proof of Theorem 4

Proof The proof of this theorem follows the proof of the Main Theorem in Barry and Pace (1999) with some slight modifications. Using triangular inequality:

$$|y - \hat{y}_{p,l}| \leq |\mathbb{E}[\hat{y}_{p,l}] - \hat{y}_{p,l}| + |y - \mathbb{E}[\hat{y}_{p,l}]|$$

Since S is upper-bounded by $(1-\delta) I$, we have for all $k \in \mathbb{N}$:

$$\left|\mathrm{Tr}\left(S^k\right)\right| \leq n (1-\delta)^k$$

Using again triangle inequality, we can bound the error with respect to the expected value:

$$\begin{aligned}
|y - \mathbb{E}[\hat{y}_{p,l}]| &= n^{-1} \left| \sum_{i=l+1}^{\infty} \frac{(-1)^i}{i} \mathrm{Tr}\left(S^k\right) \right| \\
&\leq n^{-1} \sum_{i=l+1}^{\infty} \frac{1}{i} \left|\mathrm{Tr}\left(S^k\right)\right| \\
&\leq \frac{1}{n(l+1)} \sum_{i=l+1}^{\infty} \left|\mathrm{Tr}\left(S^k\right)\right| \\
&\leq \frac{1}{l+1} \sum_{i=l+1}^{\infty} (1-\delta)^k \\
&\leq \frac{1}{l+1} \frac{(1-\delta)^{l+1}}{\delta} \\
&\leq \frac{(1-\delta)^{l+1}}{\delta}
\end{aligned}$$

And since $\delta \leq -\log(1-\delta)$, for a choice of $l \geq \delta^{-1} \log\left(\frac{2}{\epsilon\delta}\right)$, the latter part is less than $\epsilon/2$. We now bound the first part using Lemma 3. Call H the truncated series:

$$H = -\sum_{i=1}^{m} \frac{1}{i} S^i$$

This truncated series is upper-bounded by 0 (H is negative, semi-definite). The lowest eigenvalue of the truncated series can be lower-bounded in terms of δ:

$$H = -\sum_{i=1}^{m} \frac{1}{i} S^i \succeq -\sum_{i=1}^{m} \frac{1}{i} (1-\delta)^i I \succeq -\sum_{i=1}^{+\infty} \frac{1}{i} (1-\delta)^i I = (\log \delta) I$$

We can now invoke Lemma 3 to conclude:

$$\mathbb{P}\left[\left|\frac{1}{p} \sum_{i=1}^{p} \left(\mathbf{u}_i^T \mathbf{u}_i\right)^{-1} \mathbf{u}_i^T H \mathbf{u}_i - n^{-1} \mathrm{Tr}(H)\right| \geq \frac{\epsilon}{2}\right] \leq \exp\left(-\frac{p\epsilon^2}{8n^{-1}\left(\log(1/\delta)\right)^2 + 4\log(1/\delta)\,\epsilon}\right)$$

Thus, any choice of

$$p \geq 8\left(\frac{1}{\epsilon}+\frac{1}{n\epsilon^2}\right)\log\left(\eta^{-1}\right)\log^2\left(\delta^{-1}\right) \geq 4\log\left(\eta^{-1}\right)\epsilon^{-2}\left(2n^{-1}\log^2\left(\delta^{-1}\right)+\epsilon\log\left(\delta^{-1}\right)\right)$$

satisfies the inequality:$\exp\left(-\frac{p\epsilon^2}{8n^{-1}(\log\delta)^2+4\log(1/\delta)\epsilon}\right) \leq \eta$.

■

3.6 Proof of Corollary 5

Proof We introduce some notations that will prove useful for the rest of the article:

$$H = I - B^{-1}A$$

$$S = I - B^{-1/2}AB^{-1/2}$$

with $B^{-1/2}$ the inverse of the square root of the positive-definite matrix B[3]. The inequality 4 is equivalent to $\kappa^{-1}B \preceq A \preceq B$, or also:

$$\left(1-\kappa^{-1}\right)I \succeq I - B^{-1/2}AB^{-1/2} \succeq 0$$

$$\left(1-\kappa^{-1}\right)I \succeq S \succeq 0 \tag{7}$$

The matrix S is a contraction, and its spectral radius is determined by κ. Furthermore, computing the determinant of $B^{-1}A$ is equivalent to computing the determinant of $I - S$:

$$\begin{aligned}\log|I-S| &= \log\left|B^{-1/2}AB^{-1/2}\right| \\ &= \log|A| - \log|B| \\ &= \log\left|B^{-1}A\right| \\ &= \log|I-H|\end{aligned}$$

and invoking Theorem 4 gives us bounds on the number of calls to matrix-vector multiplies with respect to S. It would seem at this point that computing the inverse square root of B is required, undermining our effort. However, we can reorganize the terms in the series expansion to yield only

3. Given a real PSD matrix X, which can be diagonalized: $X = Q\Delta Q^T$ with Δ diagonal, and $\Delta_{ii} \geq 0$. Call $Y = Q\sqrt{\Delta}Q^T$ the square root of X, then $Y^2 = X$.

full inverses of B. Indeed, given $l \in \mathbb{N}^*$, consider the truncated series:

$$\begin{aligned}
y_l &= -\mathrm{Tr}\left(\sum_{i=1}^{l} \frac{1}{i} S^i\right) \\
&= -\sum_{i=1}^{l} \frac{1}{i} \mathrm{Tr}\left(S^i\right) \\
&= -\sum_{i=1}^{l} \frac{1}{i} \mathrm{Tr}\left(\sum_j \left(\begin{array}{c} j \\ i-j \end{array} \right) \left(B^{-1/2} A B^{-1/2}\right)^j\right) \\
&= -\sum_{i=1}^{l} \frac{1}{i} \sum_j \left(\begin{array}{c} j \\ i-j \end{array} \right) \mathrm{Tr}\left(\left(B^{-1/2} A B^{-1/2}\right)^j\right) \\
&= -\sum_{i=1}^{l} \frac{1}{i} \sum_j \left(\begin{array}{c} j \\ i-j \end{array} \right) \mathrm{Tr}\left(\left(B^{-1} A\right)^j\right) \\
&= -\sum_{i=1}^{l} \frac{1}{i} \mathrm{Tr}\left(\sum_j \left(\begin{array}{c} j \\ i-j \end{array} \right) \left(B^{-1} A\right)^j\right) \\
&= -\sum_{i=1}^{l} \frac{1}{i} \mathrm{Tr}\left(H^i\right)
\end{aligned}$$

Hence, the practical computation of the latter sum can be done on $A^{-1}B$. To conclude, if we compute $p = 8\left(\frac{1}{\epsilon} + \frac{1}{n\epsilon^2}\right)\log(n)\log^2(\kappa)$ truncated chains of length $l = 2\kappa \log\left(\frac{n}{\kappa\epsilon}\right)$, we get our result. This requires lp multiplies by A and inversions by B. ■

3.7 Proof of Theorem 6

We prove here the main result of Section 2. In the following, A and B are positive-definite matrices in $\mathcal{S}_n$, and B is a $\kappa-$approximation of A ($A \preceq B \preceq \kappa A$). The following notations will prove useful:

$$S = I - B^{-1/2} A B^{-1/2} \tag{8}$$

$$R = I - B^{-1} A \tag{9}$$

$$\varphi = \kappa^{-1}$$

Recall the definition of the matrix norm. Given $M \in \mathcal{S}_n^+$, $\|M\|_B = \max_{x \neq 0} \sqrt{\frac{x^T M x}{x^T B x}}$

Lemma 23 *S and R are contractions for the Euclidian and $B-$norms:*

$$\begin{aligned}
\|S\| &\leq 1 - \varphi \\
\|R\| &\leq 1 - \varphi \\
\|R\|_B &\leq (1-\varphi)^2
\end{aligned}$$

Proof Recall the definition of the matrix norm:$\|S\| = \max_{x^T x \leq 1} \sqrt{x^T S x}$. Since we know from Equation 7 that $S \preceq (1-\varphi) I$, we get the first inequality.

The second inequality is a consequence of Proposition 3.3 from Spielman and Teng (2009): A and B have the same nullspace and we have the linear matrix inequality $A \preceq B \preceq \kappa A$, which implies that the eigenvalues of $B^{-1}A$ lie between $\kappa^{-1} = \varphi$ and 1. This implies that the eigenvalues of $I - B^{-1}A$ are between 0 and $1 - \varphi$.

Recall the definition of the matrix norm induced by the B-norm over $\mathbb{R}^n$:

$$\begin{aligned}
\|R\|_B &= \max_{x \neq 0} \frac{\|Rx\|_B}{\|x\|_B} \\
&= \max_{\|x\|_B^2 \leq 1} \sqrt{x^T R^T B R x} \\
&= \max_{x^T B x \leq 1} \sqrt{x^T R^T B R x} \\
&= \max_{y^T y \leq 1} \sqrt{y^T B^{-1/2} R^T B R B^{-1/2} y}
\end{aligned}$$

and the latter expression simplifies:

$$\begin{aligned}
B^{-1/2} R^T B R B^{-1/2} &= B^{-1/2} \left(I - AB^{-1}\right) B \left(I - B^{-1}A\right) B^{-1/2} \\
&= \left(I - B^{-1/2} A B^{-1/2}\right) \left(I - B^{-1/2} A B^{-1/2}\right) \\
&= S^2
\end{aligned}$$

so we get:

$$\|R\|_B = \left\|S^2\right\| \leq \|S\|^2 \leq (1 - \varphi)^2$$

■

The approximation of the log-determinant is performed by computing sequences of power series $\left(R^k x\right)_k$. These chains are computed approximately by repeated applications of the R operator on the previous element of the chain, starting from a unit Gaussian distribution $x_0 \sim \mathcal{N}(0, \mathbf{I})$. We formalize the notion of an approximate chain.

Definition 24 Approximate power sequence. *Given a linear operator H, a start point $x^{(0)} \in \mathbb{R}^n$, and a positive-definite matrix D, we define an $\epsilon-$approximate power sequence as a sequence that does not deviate too much from the power sequence:*

$$\left\|x^{(k+1)} - Hx^{(k)}\right\|_D \leq \epsilon \left\|Hx^{(k)}\right\|_D$$

We now prove the following result that is quite intuitive: if the operator H is a contraction and if the relative error ϵ is not too great, the sum of all the errors on the chain is bounded.

Lemma 25 *Consider $\left(x^{(k)}\right)_k$ a $\nu-$approximate power sequence for the operator H and the norm D. Assuming that $\|H\|_D \leq 1 - \rho$ and that $2\nu \leq \rho \leq 1/2$, the total error is bounded by $\mathcal{O}\left(\rho^{-3}\nu\right)$:*

$$\sum_{k=0}^{\infty} \left\|x^{(k)} - H^k x^{(0)}\right\|_D \leq 4\nu\rho^{-3} \left\|x^{(0)}\right\|_D$$

Proof Call $\omega_k = \left\|x^{(k)} - H^k x^{(0)}\right\|_D$ and $\theta_k = \left\|Hx^{(k)}\right\|_D$. We are going to bound the rate of convergence of these two series. We have first using triangular inequality on the D norm and then the definition of the induced matrix norm.

$$\begin{aligned}
\theta_k &\leq \left\|Hx^{(k)} - H^k x^{(0)}\right\|_D + \left\|H^k x^{(0)}\right\|_D \\
&= \omega_k + \left\|H^k x^{(0)}\right\|_D \\
&\leq \omega_k + \|H\|_D^k \left\|x^{(0)}\right\|_D
\end{aligned}$$

We now bound the error on the ω_k sequence:

$$\begin{aligned}
\omega_{k+1} &= \left\| x^{(k+1)} - Hx^{(k)} + Hx^{(k)} - H^{k+1}x^{(0)} \right\|_D \\
&\leq \left\| Hx^{(k)} - H^{k+1}x^{(0)} \right\|_D + \left\| x^{(k+1)} - Hx^{(k)} \right\|_D \\
&\leq \|H\|_D \left\| x^{(k)} - H^k x^{(0)} \right\|_D + \nu \left\| Hx^{(k)} \right\|_D \\
&= \|H\|_D \omega_k + \nu\theta_k \\
&\leq \|H\|_D \omega_k + \nu \left(\omega_k + \|H\|_D^k \left\| x^{(0)} \right\|_D \right) \\
&\leq \left[(1-\rho)^2 + \nu \right] \omega_k + \nu (1-\rho)^{2k} \left\| x^{(0)} \right\|_D
\end{aligned}$$

Note that the inequality $(1-\rho)^2 + \nu \leq 1-\rho$ is equivalent to $\nu \leq \rho - \rho^2$. We can assume without loss of generality that $0 < \rho \leq 1/2$. Under this condition, $\rho/2 \leq \rho - \rho^2$. So the condition $2\nu \leq \rho$ implies $(1-\rho)^2 + \nu \leq 1-\rho$ which leads to:

$$\omega_{k+1} \leq (1-\rho)\omega_k + \nu (1-\rho)^{2k} \left\| x^{(0)} \right\|_D$$

By induction, one obtains:

$$\omega_k \leq \omega_1 (1-\rho)^k + \frac{\nu \left\| x^{(0)} \right\|_D}{\rho} (1-\rho)^{2k-1}$$

and since $\omega_1 = \left\| x^{(1)} - Hx^{(0)} \right\|_D \leq \nu \left\| Hx^{(0)} \right\|_D \leq \nu \|H\|_D \left\| x^{(0)} \right\|_D \leq \nu (1-\rho)^2 \left\| x^{(0)} \right\|_D$, we have a final bound that depends on ν:

$$\omega_k \leq \nu (1-\rho)^2 \left\| x^{(0)} \right\|_D (1-\rho)^k + \frac{\nu \left\| x^{(0)} \right\|_D}{\rho} (1-\rho)^{2k-1}$$

This is the sum of two geometric series, which give the bound:

$$\sum_k \omega_k \leq \nu\rho^{-1} \left[1 + \frac{1}{\rho^2 (1-\rho)^2} \right] \left\| x^{(0)} \right\|_D$$

Since $\rho \leq 1/2$, it implies $(1-\rho)^{-2} \leq 4$ and $1 \leq \rho^{-2}$, so we can further simplify:

$$\sum_k \omega_k \leq 4\nu\rho^{-3} \left\| x^{(0)} \right\|_D$$

■

We can use the bound on the norm of A to compute bound the error with a preconditioner:

Lemma 26 *Consider A, B with the same hypothesis as above, $x_0 \in \mathbb{R}^n$, $\nu \in (0, \varphi/2)$, $\varphi \leq 1/2$, and $(x_u)_u$ an $\nu-$approximate power sequence for the operator R with start vector x_0. Then:*

$$\left| \sum_{i=1}^{l} \frac{1}{i} x_0^T R^i x_0 - \sum_{i=1}^{l} \frac{1}{i} x_0^T x_i \right| \leq 4\nu\kappa^3 \kappa(B) \|x_0\|^2$$

where $\kappa(B)$ is the condition number of B.

Proof

Call $\hat{z}$ the truncated sequence:

$$\hat{z} = \sum_{i=1}^{l} \frac{1}{i} \left(x_0\right)^T x_i$$

This sequence is an approximation of the exact sequence z:

$$z = \sum_{i=1}^{l} \frac{1}{i} x_0^T R^i x_0$$

We now bound the error between the two sequences:

$$|\hat{z} - z| \leq \sum_{i=1}^{l} \frac{1}{i} \left|x_0^T \left(R^i x_0 - x_i\right)\right| \leq \sum_{i=1}^{l} \left|x_0^T \left(R^i x_0 - x_i\right)\right| \leq \sum_{i=1}^{l} \left|\left(B^{-1} x_0\right)^T B \left(R^i x_0 - x_i\right)\right|$$

Using the Cauchy-Schwartz inequality, we obtain:

$$\left|\left(B^{-1} x_0\right)^T B \left(R^i x_0 - x_i\right)\right| = \left|\left\langle B^{-1} x_0, R^i x_0 - x_i \right\rangle_B\right| \leq \left\|B^{-1} x_0\right\|_B \left\|R^i x_0 - x_i\right\|_B$$

so that we can bound the deviation using the bound from Lemma 25.

$$|\hat{z} - z| \leq \left\|B^{-1} x_0\right\|_B \sum_{i=1}^{l} \left\|R^i x_0 - x_i\right\|_B \leq 4\nu\kappa^3 \left\|B^{-1} x_0\right\|_B \|x_0\|_B \leq 4\nu\kappa^3 \kappa\left(B\right) \|x_0\|^2$$

where $\kappa\left(B\right)$ is the condition number of B. ■

We now have all the elements required for the proof of Theorem 6.

Proof

Consider $\mathbf{u}_j \sim \mathcal{N}\left(0, I_n\right)$ for $j = 1 \cdots p$, and $x_{i,j} = \begin{cases} \mathbf{u}_j / \left\|\mathbf{u}_j\right\| & i = 0 \\ x_{i-1,j} - C\left(A x_{i-1,j}\right) & i > 0 \end{cases}$

Call

$$z_{p,l} = \frac{1}{p} \sum_{j=1}^{p} \sum_{i=1}^{l} \frac{1}{i} \left(x_{0,j}\right)^T x_{i,j}$$

$$\hat{y}_{p,l} = \frac{1}{p} \sum_{j=1}^{p} \sum_{k=1}^{l} \frac{1}{k} \left(x_{0,j}\right)^T S^k x_{0,j}$$

By construction, $\left(x_{i,j}\right)_i$ is an $\nu-$approximate chain for the operator R. Applying Lemma 26, we get:

$$|z_{p,l} - \hat{y}_{p,l}| \quad \leq \quad 4\nu\kappa^3\kappa\left(B\right) \left[\frac{1}{p} \sum_{j=1}^{p} \left\|x_{0,j}\right\|^2\right] = 4\nu\kappa^3\kappa\left(B\right)$$

since $\left\|x_{0,j}\right\|^2 = 1$, which gives us a deterministic bound. Consider $\nu \leq \min\left(\frac{\epsilon}{8\kappa^3\kappa(B)}, \frac{1}{2\kappa}\right)$. Then $|z_{p,l} - \hat{y}_{p,l}| \leq \epsilon/2$. Furthermore:

$$|z_{p,l} - y| \leq |z_{p,l} - \hat{y}_{p,l}| + |y - \hat{y}_{p,l}|$$

and $\mathbb{P}\left[|y - \hat{y}_{p,l}| \geq \epsilon/2\right] \leq \eta$ for a choice of $p \geq 16\left(\frac{1}{\epsilon} + \frac{1}{n\epsilon^2}\right) \log\left(\eta^{-1}\right) \log^2\left(\delta^{-1}\right)$ and $l \geq 4\kappa \log\left(\frac{n}{\delta\epsilon}\right)$. Hence, we get our bound result of $O\left(pl\right) = O\left(\kappa\left(\frac{1}{\epsilon} + \frac{1}{n\epsilon^2}\right) \log\left(\eta^{-1}\right) \log^2\left(\delta^{-1}\right) \log\left(\frac{n}{\delta\epsilon}\right)\right)$.

■

Appendix B: Proofs of section 3.1

We put here the proofs that pertain to Section 3.1.

3.8 Properties of the generalized Laplacian

Proof of Lemma 11.

Proof The first statement is obvious from the construction of the grounded Laplacian.

Statment (2) is a direct consequence of the fact that $F_Z = PZP^T$ with $P = (I_n\, 0)$.

Then the third statement is a simple consequence of statement 2, as $\mathrm{ld}\,(Z) = \sum_i \log \lambda_i$ with $(\lambda_i)_i$ the $n-1$ positive eigenvalues of Z.

Statement (4) is straightforward after observing that the floating procedure is a linear transform from $\mathcal{S}_n$ to $\mathcal{S}_{n-1}$, so it preserves the matrix inequalities.

■

Proof of Lemma 12.

Proof Consider $A, B \in SDD_n$ positive definite. Let $P = (I_n\, \mathbf{0})$ be a projection matrix over the first n columns of L_A. If λ is an eigenvalue of A with associated eigenvector x, it is also an eigenvalue of PL_AP^T with the same eigenvector. Thus $\mathrm{ld}\,(L_A) = \log|A|$. Using the floating procedure, the second statement is straightforward.

■

3.9 Technical lemmas for Theorem 1

This lemma generalizes Lemma 8.1 in Spielman and Teng (2009).

Lemma 27 *Consider $A \in \mathcal{S}_n$ positive semi-definite, and $x \in \mathbb{R}^n$. Then $xx^T \preceq \left(x^T A^+ x\right) A$*

Proof Without loss of generality, consider $x^T x = 1$. Consider the eigenvalue decomposition of A: $A = \sum_i \lambda_i u_i u_i^T$. Since $(u_i)_i$ is an orthonormal basis of $\mathbb{R}^n$, we only need to establish that $\left(u_i^T x\right)^2 \leq \left(x^T A^+ x\right) u_i^T A u_i$ for all i. The latter term can be simplified:

$$\begin{aligned}
\left(x^T A^+ x\right) u_i^T A u_i &= \left(x^T \left[\sum_j \lambda_j^{-1} u_j u_j^T\right] x\right) \lambda_i \\
&= \lambda_i \sum_j \lambda_j^{-1} \left(u_j^T x\right)^2 \\
&\geq \left(u_i^T x\right)^2
\end{aligned}$$

which is the inequality we wanted.

■

Lemma 28 *Jensen inequality for the matrix logarithm. Let $A \in \mathcal{S}_n$ be a positive semi-definite matrix with p positive eigenvalues. Then*

$$ld\,(A) \leq p \log\left(\frac{Tr\,(A)}{p}\right)$$

Proof This is a direct application of Jensen's inequality. Call $(\lambda_i)_i$ the positive eigenvalues of A. Then $\mathrm{ld}\,(A) = \sum_i \log \lambda_i$. By concavity of the logarithm:

$$\sum_i \log \lambda_i \leq p \log\left(\frac{\sum \lambda_i}{p}\right) = p \log\left(\frac{\mathrm{Tr}\,(A)}{p}\right)$$

■

References

Ittai Abraham, Yair Bartal, and Ofer Neiman. Nearly tight low stretch spanning trees. *Foundations of Computer ...*, pages 781–790, 2008. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4691010.

Noga Alon, RM Karp, D Peleg, and Douglas West. A graph-theoretic game and its application to the k-server problem. *SIAM Journal on Computing*, 24(1):78–100, 1995. doi: 10.1137/S0097539792224474. URL http://epubs.siam.org/doi/abs/10.1137/S0097539792224474 http://epubs.siam.org/doi/pdf/10.1137/S0097539792224474.

Zhaojun Bai, Mark Fahey, and Gene H. Golub. Some large-scale matrix computation problems. *Journal of Computational and Applied ...*, 74(1):71–89, 1996. doi: 10.1.1.56.8150. URL http://www.sciencedirect.com/science/article/pii/0377042796000180.

Ronald Paul Barry and R. Kelley Pace. Monte Carlo estimates of the log determinant of large sparse matrices. *Linear Algebra and its Applications*, 1999. URL http://www.sciencedirect.com/science/article/pii/S002437959710009X.

Erik G. Boman, Doron Chen, Bruce Hendrickson, and Sivan Toledo. Maximum-weight-basis preconditioners. *Numerical Linear Algebra with Applications*, 11(89):695–721, October 2004. ISSN 1070-5325. doi: 10.1002/nla.343. URL http://doi.wiley.com/10.1002/nla.343.

Keith D. Gremban. *Combinatorial Preconditioners for Sparse, Symmetric, Diagonally Dominant Linear Systems*. PhD thesis, Carnegie Mellon University, 1996. URL www.cs.cmu.edu/ glmiller/Publications/GrembanPHD.ps.gz.

Ilse C F Ipsen and Dean J Lee. Determinant approximations. *Numerical Linear Algebra with Applications (under ...*, (X), 2006. URL http://www.ncsu.edu/crsc/reports/ftp/pdf/crsc-tr03-30.pdf.

Ioannis Koutis, Gary L Miller, and Richard Peng. Approaching optimality for solving SDD linear systems. pages 1–16, 2010.

J Liu. The Role of Elimination Trees in Sparse Factorization. *SIAM Journal on Matrix Analysis and Applications*, 11(1):134–172, 1990. doi: 10.1137/0611010. URL http://epubs.siam.org/doi/abs/10.1137/0611010.

R J Martin. Approximations to the determinant term in Gaussian maximum likelihood estimation of some spatial models. *Communications in Statistics-Theory and Methods*, 22(1):189–205, 1992.

M McCourt. A Stochastic Simulation for Approximating the log-Determinant of a Symmetric Positive Definite Matrix. *compare*, 2:1–10, 2008. URL http://www.thefutureofmath.com/mathed/logdet.pdf.

Gérard A Meurant. *Computer Solution of Large Linear Systems*. North-Holland: Amsterdam, 1999.

Arnold Reusken. Approximation of the Determinant of Large Sparse Symmetric Positive Definite Matrices. *SIAM Journal on Matrix Analysis and Applications*, 23(3):799, 2002. ISSN 08954798. doi: 10.1137/S089547980036869X. URL http://link.aip.org/link/SJMAEL/v23/i3/p799/s1&Agg=doi.

Daniel A Spielman. Spectral sparsification of graphs. *Arxiv preprint arXiv:0808.4134*, 2008. URL http://arxiv.org/abs/0808.4134.

Daniel A Spielman. Algorithms , Graph Theory , and Linear Equations in Laplacian Matrices. Technical report, Proceedings of the International Congress of Mathematicians, Hyderabad, India, 2010.

Daniel A Spielman and Shang-Hua Teng. A Local Clustering Algorithm for Massive Graphs and its Application to Nearly-Linear Time Graph Partitioning. 2008.

Daniel A Spielman and Shang-Hua Teng. Nearly-Linear Time Algorithms for Preconditioning and Solving Symmetric , Diagonally Dominant Linear Systems. pages 1–48, 2009.

Daniel A Spielman and Jaeoh Woo. A Note on Preconditioning by Low-Stretch Spanning Trees. pages 1–4, 2009.

M.J. Wainwright and M.I. Jordan. Log-determinant relaxation for approximate inference in discrete Markov random fields. *IEEE Transactions on Signal Processing*, 54 (6):2099–2109, June 2006. ISSN 1053-587X. doi: 10.1109/TSP.2006.874409. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1634807.

Y. Zhang and W. E. Leithead. Approximate implementation of the logarithm of the matrix determinant in Gaussian process regression. *Journal of Statistical Computation and Simulation*, 77(4):329–348, April 2007. ISSN 0094-9655. doi: 10.1080/10629360600569279. URL http://www.tandfonline.com/doi/abs/10.1080/10629360600569279.

Yunong Zhang, W.E. Leithead, D.J. Leith, and L. Walshe. Log-det approximation based on uniformly distributed seeds and its application to Gaussian process regression. *Journal of Computational and Applied Mathematics*, 220(1-2): 198–214, October 2008. ISSN 03770427. doi: 10.1016/j.cam.2007.08.012. URL http://linkinghub.elsevier.com/retrieve/pii/S0377042707004360.