Sparse Cholesky Precision Matrix Estimation

Kingsley Yeon
email yeon@uchicago.edu

March 19, 2025

Abstract

We introduce a novel algorithm for estimating high-dimensional precision matrices by exploiting a sparse Cholesky factorization. By reparameterizing the precision matrix P as LL^T, with L a lower–triangular matrix having positive diagonal entries, the maximum likelihood estimation is reformulated as an unconstrained optimization problem. We rigorously derive error bounds for the estimator via the delta method, showing that each column of L is estimated at the $\sqrt{m}$–rate with an asymptotic covariance given by the inverse of the per–observation Hessian. This theoretical framework establishes a solid foundation for inference on the entries of L and, consequently, on the precision matrix itself.

In parallel, we propose an efficient, randomized strategy to assess the estimation error. Leveraging stochastic trace estimation combined with the Lanczos algorithm, our approach rapidly approximates the Kullback–Leibler divergence between the true Gaussian distribution and its approximation induced by the estimated Cholesky factor. This method, which requires only a modest number of matrix–vector products, serves as a fast diagnostic tool for large–scale problems. Numerical experiments corroborate that the empirical performance of the estimator aligns well with the derived asymptotic error bounds, while the randomized error–checking algorithm reliably quantifies the approximation quality in practice.

1 Introduction

In many multivariate settings, we observe data $u_1, u_2, \ldots, u_m \in \mathbb{R}^n$ drawn independently from a Gaussian distribution $\mathcal{N}(0, \Sigma)$. Our objective is to estimate the precision matrix $P = \Sigma^{-1}$. A common approach is to reparameterize P via its Cholesky factorization, namely,

$$P = LL^T,$$

where L is a lower–triangular matrix with strictly positive diagonal entries. This parametrization not only guarantees positive definiteness but also simplifies the likelihood function.

The log-likelihood function for the Gaussian model is given by

$$\ell(P) = \frac{m}{2} \log\det(P) - \frac{1}{2} \sum_{i=1}^{m} u_i^T P u_i.$$

Substituting $P = LL^T$ and noting that $\det(P) = (\det(L))^2 = \prod_{j=1}^{n} L_{jj}^2$, we have

$$\log\det(P) = 2 \sum_{j=1}^{n} \log L_{jj}.$$

Thus, the log-likelihood becomes

$$\ell(L) = m \sum_{j=1}^{n} \log L_{jj} - \frac{1}{2} \sum_{i=1}^{m} u_i^T L L^T u_i.$$

Since maximizing the likelihood is equivalent to minimizing the negative log-likelihood, we define the loss function as

$$f(L) = -\ell(L) = -m \sum_{j=1}^{n} \log L_{jj} + \frac{1}{2} \sum_{i=1}^{m} u_i^T L L^T u_i.$$

Exploiting the associativity of matrix multiplication, we can rewrite the quadratic term as

$$u_i^T L L^T u_i = \|L^T u_i\|^2 = \sum_{j=1}^{n} (l_j^T u_i)^2,$$

where l_j denotes the jth column of L. We obtain the loss function in the form

$$f(L) = \sum_{j=1}^{n} \left[-m \log L_{jj} + \frac{1}{2} \sum_{i=1}^{m} (l_j^T u_i)^2 \right].$$

This formulation is central to our maximum likelihood estimation procedure, as it converts the problem of estimating the precision matrix P into an unconstrained optimization problem over the entries of the Cholesky factor L.

2 Optimization and Delta method for asymptotics

We begin with the loss

$$f(L) = \sum_{j=1}^{n} \left[-m \log L_{jj} + \frac{1}{2} \sum_{i=1}^{m} \left(l_j^T u_i \right)^2 \right],$$

where for each $j = 1, \ldots, n$ the jth column l_j of the lower-triangular matrix L (with $L_{jj} > 0$) parameterizes the precision matrix via

$$P = LL^T,$$

and the observations u_i are drawn from a Gaussian distribution $N(0, \Sigma)$ with $\Sigma = P^{-1}$.

For a fixed column j, the loss contribution is

$$f_j(l_j) = -m \log L_{jj} + \frac{1}{2} \sum_{i=1}^{m} \left(l_j^T u_i \right)^2.$$

Its gradient (with respect to the free parameters in l_j) is

$$g_j(l_j) = -\frac{m}{L_{jj}} e_j + \sum_{i=1}^{m} (l_j^T u_i)\, u_i,$$

where e_j is the unit vector with 1 in the jth coordinate.

Taking expectations (and noting that the u_i are i.i.d.) we have

$$E\big[g_j(l_{0,j})\big] = -\frac{m}{L_{0,jj}} e_j + m\, E\big[(l_{0,j}^T u) u\big].$$

Because the covariance of u is Σ, we obtain

$$E\big[(l_{0,j}^T u_i) u_i\big] = E\big[u_i\, u_i^T\big]\, l_{0,j} = \Sigma\, l_{0,j}.$$

Setting the expected score to zero gives

$$-\frac{m}{L_{0,jj}}\,e_j + m\,\Sigma\,l_{0,j} = 0 \quad \Longrightarrow \quad \Sigma\,l_{0,j} = \frac{1}{L_{0,jj}}\,e_j.$$

Now, to see the implication for the quadratic form $l_{0,j}^T \Sigma\, l_{0,j}$, we multiply both sides on the left by $l_{0,j}^T$:

$$l_{0,j}^T \Sigma\, l_{0,j} = \frac{1}{L_{0,jj}}\, l_{0,j}^T e_j.$$

Since L is lower–triangular, its jth column $l_{0,j}$ has $L_{0,jj}$ as its jth (first nonzero) entry, so that

$$l_{0,j}^T e_j = L_{0,jj}.$$

Thus,

$$l_{0,j}^T \Sigma\, l_{0,j} = \frac{L_{0,jj}}{L_{0,jj}} = 1.$$

Define the per-observation score contribution as

$$s_i = (l_{0,j}^T u_i) u_i.$$

Then the full score is

$$g_j(l_{0,j}) = \sum_{i=1}^{m} \Big[s_i - E[s_i] \Big],$$

with

$$E[s_i] = \Sigma\, l_{0,j} = \frac{1}{L_{0,jj}}\, e_j.$$

By the central limit theorem, the aggregated (centered) score satisfies

$$\sqrt{m}\,\frac{1}{m}\, g_j(l_{0,j}) \xrightarrow{d} N(0, \Omega_j),$$

with

$$\Omega_j = \mathrm{Var}\big(s_i\big) = E\big[(l_{0,j}^T u_i)^2\, u_i u_i^T\big] - \big(\Sigma\, l_{0,j}\big)\big(\Sigma\, l_{0,j}\big)^T.$$

Using Isserlis' (or Wick's) theorem for Gaussian random vectors, one can show that

$$E\big[(l_{0,j}^T u_i)^2\, u_i u_i^T\big] = \Big(l_{0,j}^T \Sigma\, l_{0,j}\Big)\Sigma + 2\,\Sigma\, l_{0,j} l_{0,j}^T \Sigma.$$

Because the identification condition gives $l_{0,j}^T \Sigma\, l_{0,j} = 1$ and since

$$\Sigma\, l_{0,j} l_{0,j}^T \Sigma = \frac{1}{L_{0,jj}^2}\, e_j e_j^T,$$

we deduce that

$$\Omega_j = \Sigma + 2\,\frac{1}{L_{0,jj}^2}\, e_j e_j^T - \frac{1}{L_{0,jj}^2}\, e_j e_j^T = \Sigma + \frac{1}{L_{0,jj}^2}\, e_j e_j^T.$$

On the other hand, the Hessian (second derivative) of the per–observation loss for column j is computed by differentiating the score. The quadratic term contributes

$$\frac{\partial^2}{\partial l_j \partial l_j^T}\,\frac{1}{2}(l_j^T u_i)^2 = u_i u_i^T,$$

so that summing over i and adding the contribution from the $-m \log L_{jj}$ term (whose second derivative is $\frac{m}{L_{jj}^2}\, e_j e_j^T$) we have

$$H_j = \sum_{i=1}^{m} u_i u_i^T + \frac{m}{L_{jj}^2}\, e_j e_j^T.$$

Dividing by m (to obtain the per–observation Hessian) yields

$$\mathcal{H}_j = \Sigma + \frac{1}{L_{0,jj}^2}\, e_j e_j^T.$$

Notice that this is identical to Ω_j.

Now, by a first–order Taylor expansion (the delta method), if $\hat{l}_j$ is an estimator of $l_{0,j}$ that (approximately) satisfies

$$g_j(\hat{l}_j) = 0 \quad \text{and} \quad g_j(\hat{l}_j) \approx g_j(l_{0,j}) + H_j\,(\hat{l}_j - l_{0,j}),$$

then

$$\hat{l}_j - l_{0,j} \approx -H_j^{-1}\, g_j(l_{0,j}).$$

Since

$$\sqrt{m}\,\frac{1}{m}\, g_j(l_{0,j}) \xrightarrow{d} N(0, \Omega_j),$$

it follows that by the delta method

$$\sqrt{m}\,(\hat{l}_j - l_{0,j}) \xrightarrow{d} N\Big(0,\, H_j^{-1}\,\Omega_j\, H_j^{-1}\Big).$$

Because $H_j/m \to \mathcal{H}_j$ and we have shown that $\Omega_j = \mathcal{H}_j$, the asymptotic covariance simplifies to

$$\sqrt{m}\,(\hat{l}_j - l_{0,j}) \xrightarrow{d} N\Big(0,\, \mathcal{H}_j^{-1}\Big)$$

or, equivalently,

$$\hat{l}_j - l_{0,j} = O_P\Big(\frac{1}{\sqrt{m}}\Big) \quad \text{with} \quad \operatorname{Var}\big(\sqrt{m}\,(\hat{l}_j - l_{0,j})\big) = \mathcal{H}_j^{-1}.$$

Therefore, using the delta method we conclude that each column l_j of the Cholesky factor L is estimated at a $\sqrt{m}$–rate, with the asymptotic distribution

$$\sqrt{m}\Big(\hat{l}_j - l_{0,j}\Big) \xrightarrow{d} N\left(0, \Big(\Sigma + \frac{1}{L_{0,jj}^2}\, e_j e_j^T\Big)^{-1}\right).$$

Because the per–observation Hessian $\mathcal{H}_j = \Sigma + \frac{1}{L_{0,jj}^2}\, e_j e_j^T$ matches the asymptotic covariance of the score, the asymptotic variance of $\hat{l}_j$ is given by the inverse of this matrix. This result provides a basis for constructing confidence intervals and hypothesis tests for the entries of L in large samples.

2.1 Result

In our simulation, we first generate a 15-dimensional precision matrix using a Gaussian kernel, compute its Cholesky factor L_{true}, and obtain the corresponding covariance matrix Σ. We then focus on estimating the second column ($j = 1$) of L. For that column, only the entries from the 2nd row onward (i.e. L_{jj} and below) are free parameters. Using a Newton method that minimizes the loss

$$f(L) = \sum_{j=1}^{n} \left[-m \log L_{jj} + \frac{1}{2} \sum_{i=1}^{m} \big(l_j^T u_i\big)^2 \right],$$

we estimate the free parameter vector z for this column based on $m = 10{,}000$ samples drawn from $N(0, \Sigma)$. Repeating the experiment over 500 Monte Carlo replications, we compute the scaled estimation error $\sqrt{m}(\hat{z} - z_0)$. The delta method predicts that this scaled error is asymptotically normal with covariance matrix

$$V_{\text{pred}} = \mathcal{H}^{-1},$$

where

$$\mathcal{H} = \Sigma_{\text{sub}} + \frac{1}{L_{0,jj}^2} e_1 e_1^T.$$

Figure 1 displays a set of subplots—one for each coordinate of the estimated vector z. In each subplot, the histogram shows the empirical distribution of the scaled error for that coordinate, and the red curve overlays the theoretical normal density using the corresponding predicted standard deviation. The close agreement between the histograms and the overlaid curves provides visual evidence that our simulation results align with the asymptotic distribution derived via the delta method.

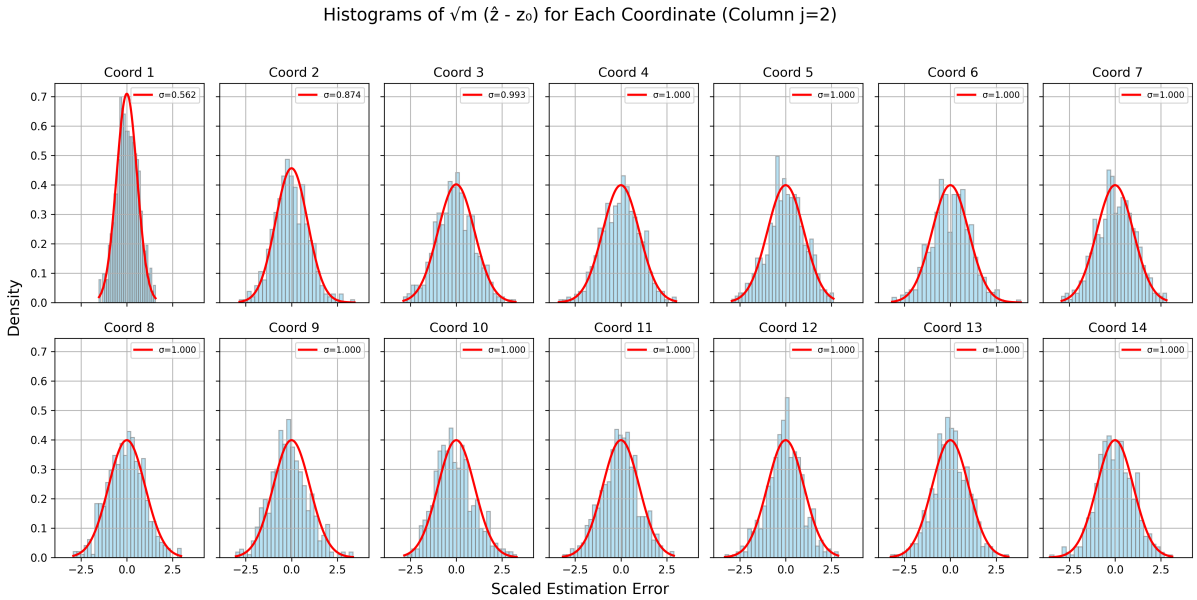

Figure 1: Histograms of the scaled estimation errors, $\sqrt{m}(\hat{z} - z_0)$. The simulation is based on 500 Monte Carlo replications with a $m = 10\,000$ sample size per trial.

3 Sparsity Pattern

To find an appropriate sparsity pattern for the Cholesky factorization of a precision matrix, one effective strategy is to first order the variables using a reverse-maximin criterion, whereby each variable is sequentially selected based on being maximally distant from those already chosen, thereby capturing the intrinsic geometric structure of the data and assigning each variable a local length scale. The sparsity pattern is then determined by retaining only those entries in the lower-triangular factor for which the distance between corresponding data points is less than a threshold proportional to the local length scale, modulated by a tuning parameter. This method leverages the screening effect typical of many kernel matrices, ensuring that only the most significant conditional dependencies are preserved. For a comprehensive treatment of this

sparsity selection process, including theoretical error bounds and computational complexity considerations, refer to [6].

4 Computational Complexity

Assume that for each column j the number of nonzero entries is roughly d (with $d \ll n$). Under this sparsity assumption, the computational cost per column is significantly reduced compared to the dense case. For each observation u_i, computing the inner product $l_j^T u_i$ involves only $O(d)$ operations rather than $O(n)$. Consequently, evaluating the gradient over m observations costs $O(m\,d)$ per column, and when accounting for the additional multiplications inherent in forming the gradient terms, the cost becomes $O(m\,d^2)$.

The Hessian is assembled by accumulating contributions from each observation in the d-dimensional subspace, incurring $O(m\,d^2)$ operations. In each Newton iteration, the subsequent update step involves solving a linear system of size $d \times d$, which typically requires $O(d^3)$ operations. Thus, each Newton iteration for a single column has a complexity of

$$O\big(m\,d^2 + d^3\big).$$

Since the algorithm proceeds column–wise for $j = 1, \ldots, n$, the total cost per full sweep (i.e., one Newton iteration for every column) is

$$\sum_{j=1}^{n} O\big(m\,d^2 + d^3\big) = O\big(m\,n\,d^2 + n\,d^3\big).$$

Let T denote the number of Newton iterations per column. In our experiments, Newton's method converges very rapidly—typically within 3–5 iterations—so that T is effectively a small constant that can be omitted from the asymptotic expression. Accordingly, the overall computational complexity of the algorithm is

$$O\Big(T\big(m\,n\,d^2 + n\,d^3\big)\Big) \approx O\big(m\,n\,d^2 + n\,d^3\big).$$

An additional advantage of the proposed method is that the algorithm is inherently parallelizable in n, the number of columns (or data dimensions). Since the computations for each column of the Cholesky factor L are independent, the cost associated with processing n columns can be distributed across multiple processors. This parallelism is particularly beneficial when n is large, as it allows the per-iteration computational cost to be reduced to that of processing a single column.

In practice, the hierarchy $m \gg n \gg d$ holds: m represents the number of observations and is very large compared to n, the full data dimension, while n is substantially larger than d, the effective sparsity level per column. Under these conditions, the dominant term in the computational complexity is $m\,n\,d^2$, which is effectively linear in m. Moreover, since the algorithm is parallelizable in n, the cost per processor primarily scales with m, making the method highly scalable even in high-dimensional settings.

5 Randomized Estimation of the KL Divergence

In the zero–mean Gaussian setting, let

$$p(x) = \mathcal{N}(0, \Sigma) \quad \text{and} \quad q(x) = \mathcal{N}(0, (LL^T)^{-1}),$$

where $\Sigma \in \mathbb{R}^{n\times n}$ is the true covariance matrix and $L \in \mathbb{R}^{n\times n}$ is a lower–triangular factor used to approximate the precision matrix via $(LL^T)^{-1}$. By definition, the Kullback–Leibler divergence is

$$D_{\mathrm{KL}}(p \parallel q) = \int_{\mathbb{R}^n} p(x)\, \log \frac{p(x)}{q(x)}\, dx.$$

For a zero–mean Gaussian, the density of p is

$$p(x) = \frac{1}{(2\pi)^{n/2}\det(\Sigma)^{1/2}} \exp\Big(-\frac{1}{2}x^T\Sigma^{-1}x\Big),$$

and, noting that the covariance of q is $(LL^T)^{-1}$ (so its precision is LL^T), the density of q is

$$q(x) = \frac{\det(LL^T)^{1/2}}{(2\pi)^{n/2}} \exp\Big(-\frac{1}{2}x^T(LL^T)x\Big).$$

Taking the ratio and its logarithm, we obtain

$$\log\frac{p(x)}{q(x)} = -\frac{1}{2}\ln\det(\Sigma) - \frac{1}{2}\ln\det(LL^T) - \frac{1}{2}\Big(x^T\Sigma^{-1}x - x^T(LL^T)x\Big).$$

Taking the expectation with respect to $p(x)$ and using the standard Gaussian identities

$$\mathbb{E}_p[x^T\Sigma^{-1}x] = \operatorname{tr}(\Sigma^{-1}\Sigma) = n, \quad \text{and} \quad \mathbb{E}_p[x^T(LL^T)x] = \operatorname{tr}(LL^T\Sigma),$$

one finds

$$D_{\mathrm{KL}}(p \parallel q) = -\frac{1}{2}\ln\det(\Sigma) - \frac{1}{2}\ln\det(LL^T) - \frac{1}{2}\Big[n - \operatorname{tr}(LL^T\Sigma)\Big].$$

Since $\ln\det(LL^T\Sigma) = \ln\det(LL^T) + \ln\det(\Sigma)$, the logarithmic terms combine to yield

$$D_{\mathrm{KL}}(p \parallel q) = \frac{1}{2}\Big[\operatorname{tr}(LL^T\Sigma) - \ln\det(LL^T\Sigma) - n\Big].$$

Defining the eigenvalues $\{\lambda_i\}_{i=1}^n$ of

$$A = LL^T\Sigma,$$

we can equivalently write

$$D_{\mathrm{KL}}(p \parallel q) = \frac{1}{2}\sum_{i=1}^{n}\Big(\lambda_i - \ln\lambda_i - 1\Big),$$

where we define

$$f(\lambda) = \lambda - \ln\lambda - 1.$$

Moreover, since the matrix

$$B = L^T\Sigma L$$

is similar to A (and thus shares the same eigenvalues), in settings where Σ is available only through matrix–vector products, we introduce the linear operator

$$\mathcal{A}(v) = L^T\Big[\Sigma\,(L\,v)\Big],$$

which encapsulates the action of B.

Now the problem is turned into efficiently estimating the trace of matrix functions; which is an area of extensive study [1, 3, 7]. A simple observation shows

$$\mathbb{E}[g_i^T A g_i] = \operatorname{tr}(A), \quad \text{for each } g_i \in \mathbb{R}^d.$$

This suggests a Monte Carlo method, which can be done via a bilinear form,

$$\operatorname{tr}(f(A)) \approx \frac{1}{m}\sum_{i=1}^{m} g_i^T f(A) g_i,$$

where A is accessible only through matrix-vector products and f is a smooth function defined on the spectrum of A. Hutchinson's original approach suggests using random ± 1 sign vectors for $g_1, \ldots, g_m$, while an earlier study by Girard proposes employing standard normal random variables. Below, we briefly derive and describe the simplest example of such an algorithm. For large-scale problems, readers are encouraged to consult the referenced works.

A randomized estimation of $\mathrm{tr}(f(B))$ can be performed by combining stochastic trace estimation (Girard–Hutchinson estimator) with the Lanczos and Golub-Welsch algorithm.

To derive the relationship between Gaussian quadrature and trace estimation, the key observation is that for any symmetric matrix M with eigen-decomposition $M = \sum_{j=1}^{n} \lambda_j \, q_j q_j^T$ and for any vector v (which can be expanded as $v = \sum_{j=1}^{n} \gamma_j \, q_j$), the bilinear form $v^T f(M) v$ can be written as

$$v^T f(M) v = \left(\sum_{j=1}^{n} \gamma_j \, q_j^T \right) \left(\sum_{k=1}^{n} f(\lambda_k) \, q_k q_k^T \right) \left(\sum_{l=1}^{n} \gamma_l \, q_l \right) = \sum_{j=1}^{n} \gamma_j^2 f(\lambda_j) = \int f(\lambda) \, d\mu(\lambda),$$

with $d\mu(\lambda) = \sum_{j=1}^{n} \gamma_j^2 \, \delta(\lambda - \lambda_j) d\lambda$. Gaussian quadrature approximates this integral by a weighted sum

$$\int f(\lambda) \, d\mu(\lambda) \approx \sum_{j=1}^{k} w_j \, f(\lambda_j),$$

where the nodes and weights are determined by constructing orthogonal polynomials with respect to $d\mu$. In practice, this is achieved via the Lanczos algorithm. For each random probe vector v_i (drawn from a suitable distribution such as Rademacher or Gaussian), the Lanczos process generates an orthonormal basis for the Krylov subspace $\mathcal{K}_k(\mathcal{A}, v_i)$ and produces a symmetric tridiagonal matrix T_i whose entries are the recurrence coefficients of the orthogonal polynomials associated with the measure $d\mu$. Diagonalizing T_i as

$$T_i = U_i \, \mathrm{diag}(\mu_{i1}, \ldots, \mu_{ik}) \, U_i^{-1},$$

and noting that T_i is symmetric so that $U_i^{-1} = U_i^T$, the eigenvalues $\{\mu_{ij}\}_{j=1}^{k}$ approximate those of $\mathcal{A}$ while the quadrature weights are given by

$$w_{ij} = (U_i(1, j))^2,$$

with $U_i(1, j)$ being the first component of the jth eigenvector. Consequently, a Gaussian quadrature rule yields the approximation

$$v_i^T f(\mathcal{A}) v_i \approx \sum_{j=1}^{k} w_{ij} \, f(\mu_{ij}).$$

By averaging these estimates over q random probe vectors, we obtain an approximation of $\mathrm{tr}(f(B))$, namely,

$$\widehat{\mathrm{tr}(f(B))} = \frac{1}{q} \sum_{i=1}^{q} \sum_{j=1}^{k} w_{ij} \, f(\mu_{ij}).$$

Thus, the KL divergence is estimated as

$$\widehat{D}_{\mathrm{KL}} = \frac{1}{2} \, \widehat{\mathrm{tr}(f(B))}.$$

Algorithm 1: Randomized Estimation of the KL Divergence

Input: Covariance operator Σ, Cholesky factor L, number of probes q, Lanczos steps k

Define $\mathcal{A}(v) = L^T\Big[\Sigma\,(L\,v)\Big]$;

for $i = 1$ *to* q **do**

  Generate a random probe vector z_i with entries in $\{-1, 1\}$;

  Normalize: $v_i \leftarrow z_i/\|z_i\|$;

  Run the Lanczos algorithm for k iterations with operator $\mathcal{A}$ starting from v_i to obtain the tridiagonal matrix T_i;

  Diagonalize T_i as

$$T_i = U_i\,\mathrm{diag}(\mu_{i1}, \ldots, \mu_{ik})\,U_i^{-1} \quad (\text{with } U_i^{-1} = U_i^T),$$

  to obtain eigenvalues $\{\mu_{ij}\}_{j=1}^k$ and eigenvector matrix U_i;

  Set quadrature weights $w_{ij} \leftarrow (U_i(1,j))^2$ for $j = 1, \ldots, k$;

  Compute the quadrature estimate $\eta_i \leftarrow \sum_{j=1}^k w_{ij}\,f(\mu_{ij})$, where $f(\lambda) = \lambda - \ln\lambda - 1$;

Estimate the trace: $\widehat{\mathrm{tr}(f(\mathcal{A}))} \leftarrow \dfrac{n}{q}\sum_{i=1}^q \eta_i$;

Output: Estimated KL divergence $\widehat{D}_{\mathrm{KL}} = \frac{1}{2}\,\widehat{\mathrm{tr}(f(\mathcal{A}))}$

The direct computation of $LL^T\Sigma$ would require $O(n^2)$ operations and an eigenvalue decomposition costing $O(n^3)$ in the worst case. In contrast, the above randomized algorithm employs only $q \cdot k$ matrix-vector products with Σ, where typically $q, k \ll n$ [4].

For example, the Lanczos algorithm yields an average relative error decaying like $k \sim O((\frac{\ln(n)}{k})^2)$, requiring only on the order of $\ln(n)$ iterations for a fixed tolerance. Additionally, the error bound of the Hutch estimator and algorithm 1 follows naturally from Hoeffding's inequality, as shown in [1].

In principle, we do not need to restrict the random trace estimation to just Girard–Hutchinson (Hutch) or Hutch++ [5], other matrix–vector based trace estimators [2] could be modified to our framework.

References

[1] Zhaojun Bai, Gark Fahey, and Gene Golub. Some large-scale matrix computation problems. *Journal of Computational and Applied Mathematics*, 74(1-2):71–89, 1996.

[2] Ethan N Epperly, Joel A Tropp, and Robert J Webber. Xtrace: Making the most of every sample in stochastic trace estimation. *SIAM Journal on Matrix Analysis and Applications*, 45(1):1–23, 2024.

[3] Rafael Díaz Fuentes, Marco Donatelli, Caterina Fenu, and Giorgio Mantica. Estimating the trace of matrix functions with application to complex networks. *Numerical Algorithms*, 92(1):503–522, 2023.

[4] Jacek Kuczyński and Henryk Woźniakowski. Estimating the largest eigenvalue by the power and lanczos algorithms with a random start. *SIAM journal on matrix analysis and applications*, 13(4):1094–1122, 1992.

[5] Raphael A Meyer, Cameron Musco, Christopher Musco, and David P Woodruff. Hutch++: Optimal stochastic trace estimation. In *Symposium on Simplicity in Algorithms (SOSA)*, pages 142–155. SIAM, 2021.

[6] Florian Schafer, Matthias Katzfuss, and Houman Owhadi. Sparse cholesky factorization by kullback–leibler minimization. *SIAM Journal on scientific computing*, 43(3):A2019–A2046, 2021.

[7] Shashanka Ubaru, Jie Chen, and Yousef Saad. Fast estimation of tr(f(a)) via stochastic lanczos quadrature. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1075–1099, 2017.