

Entrywise Approximate Solutions for SDDM Systems in Almost-Linear Time

Angelo Farfan*

Mehrdad Ghadiri†

Junzhao Yang‡

Abstract

We present an algorithm that given any invertible symmetric diagonally dominant M-matrix (SDDM), i.e., a principal submatrix of a graph Laplacian, \mathbf{L} and a nonnegative vector \mathbf{b} , computes an entrywise approximation to the solution of $\mathbf{L}\mathbf{x} = \mathbf{b}$ in $\tilde{O}(mn^{o(1)})$ time with high probability, where m is the number of nonzero entries and n is the dimension of the system.

*dstaff@mit.edu

†mehrdadg@mit.edu

‡junzhaoy@andrew.cmu.edu

Contents

1	Introduction	1
1.1	Preliminaries	2
1.2	Technical Overview	4
1.3	Related Work	8
2	Low-Diameter Cover	9
2.1	Probability Distance	9
2.2	Constructing a Cover with Low Probability Distance Diameter	11
3	The Almost-Linear-Time Algorithm	17
3.1	The Threshold Decay Framework	17
3.2	Errors Corresponding to Partial Linear Systems	19
3.3	The Efficient Partial Solver	20
3.4	The Almost-Linear-Time SDDM Solver	22
3.4.1	Bounding the sum of sizes	23
3.4.2	Maintaining the vector $\widehat{\mathbf{b}}^{(t)}$	25
3.4.3	Maintaining $I^{(t)}$ and $H^{(t)}$	25
3.4.4	Wrap-up	26
4	Conclusion	28

1 Introduction

Graph Laplacians have been a central subject of study in computer science over the past few decades due to their many connections to graphs and random walks. The seminal work of [ST14] presented the first near-linear time (in number of nonzeros of the matrix) algorithm for computing the solution of system $\mathbf{L}\mathbf{x} = \mathbf{b}$, where \mathbf{L} is a graph Laplacian or more generally a principal submatrix of a graph Laplacian, i.e., symmetric diagonally dominant M-matrices (SDDM). Many follow-up works contributed to improving many aspects of such near-linear solvers such as simplifying the algorithm [KOSZ13, LS13, KS16], improving the polylogarithmic factors in the running time [KMST10, KMP11, LS13, PS14, KMP14, CKM⁺14, JS21, KS16], and developing parallel solvers with polylogarithmic depth and near-linear work [BGK⁺11, PS14, KLP⁺16].

All of these algorithms have focused on normwise error bounds and output a solution $\hat{\mathbf{x}}$ such that

$$\left\| \hat{\mathbf{x}} - \mathbf{L}^\dagger \mathbf{b} \right\|_{\mathbf{L}} \leq \epsilon \cdot \left\| \mathbf{L}^\dagger \mathbf{b} \right\|_{\mathbf{L}}.$$

As pointed out recently by [GNV26], since the entries of the solution can vary exponentially, such normwise error bounds cannot recover small entries unless the error parameter ϵ is exponentially small in n , the size of the matrix. Since these algorithms are all iterative preconditioning approaches, they require about $\log(1/\epsilon)$ iterations and the numbers used in the algorithm need $\log(\kappa/\epsilon)$ bits of precision to guarantee convergence, where κ is the condition number of the matrix. Taking ϵ exponentially small means the number of bit operations, which we refer to as the running time throughout the paper, of the algorithm is about $O(mn^2)$. [GNV26] improved on this by presenting an algorithm with a running time of $\tilde{O}(m\sqrt{n}\log^2(U\epsilon^{-1}\delta^{-1}))$, where U is a bound on the magnitude of entries and $1 - \delta$ is the probability of success, that for any invertible SDDM matrix \mathbf{L} produces $\hat{\mathbf{x}}$ such that

$$e^{-\epsilon}(\mathbf{L}^{-1}\mathbf{b})_i \leq \hat{\mathbf{x}}_i \leq e^{\epsilon}(\mathbf{L}^{-1}\mathbf{b})_i, \text{ for all } i \in [n],$$

which we denote by $\hat{\mathbf{x}} \approx_{\epsilon} \mathbf{L}^{-1}\mathbf{b}$. This raises the question of whether solving a linear system with the stronger notion of entrywise approximation is possible in almost-linear time. We answer this question in the affirmative.

Theorem 1.1. *There exists a randomized algorithm **SDDMSOLVE** such that, for any $\delta \in (0, 1)$ and any $\epsilon > (nU)^{-2^{\sqrt{\log n}}}$ (i.e., not exponentially small), any invertible SDDM matrix $\mathbf{L} \in \mathbb{Z}^{n \times n}$ with m nonzero integer entries in $[-U, U]$ and a nonnegative vector $\mathbf{b} \in \mathbb{Z}^n$ with integer entries in $[0, U]$, with probability at least $1 - \delta$, computes the entrywise approximate solution*

$$\tilde{\mathbf{x}} \approx_{\epsilon} \mathbf{L}^{-1}\mathbf{b}$$

whose entries are represented by $O(\log(nU/\epsilon))$ -bit floating points, using¹

$$\tilde{O}(m2^{O(\sqrt{\log n})} \log(U) \log^2(U\epsilon^{-1}\delta^{-1}))$$

bit operations.

Our algorithm first constructs a low-diameter cover by solving a collection of linear systems with normwise error bounds on random right-hand side vectors. We then use this low-diameter cover to *predict* the scale of the entries of the solution in an adaptive manner, thereby achieving an entrywise approximate solution.

¹ \tilde{O} suppresses the $\log \log(U\epsilon^{-1}\delta^{-1})$ factors.

In the next subsection, we introduce our notation and review the relevant results from the literature. Equipped with these preliminaries, we provide an overview of our algorithm and the main ideas behind the proof of [Theorem 1.1](#) in [Section 1.2](#). We discuss related work in [Section 1.3](#). In [Section 2](#), we define our low-diameter cover and present an algorithm for constructing it with high probability. Finally, [Section 3](#) presents our almost-linear-time algorithm for solving SDDM linear systems with entrywise approximation guarantees, together with the proofs of correctness and bit-complexity bounds. We conclude in [Section 4](#) with a discussion of open problems and future directions.

1.1 Preliminaries

Notation. We denote $\{1, 2, \dots, n\}$ by $[n]$. For an undirected graph $G = (V, E, w)$ and a vertex $v \in V$, let $N(v)$ denote the set of neighbors of v . For a subset of vertices $S \subseteq V$, we denote by $G(S)$ the subgraph of G induced by S . We use \tilde{O} notation to suppress polylogarithmic factors in n and polyloglog factors in $U\epsilon^{-1}\delta^{-1}$. Formally, $\tilde{O}(f) = O(f \cdot \log(n \cdot \log(U\epsilon^{-1}\delta^{-1})))$. We also note that $\text{poly log } n$ factors are at most $2^{O(\sqrt{\log n})}$.

We use bold lowercase letters to denote vectors and bold uppercase letters to denote matrices. The vector of all ones is denoted by $\mathbf{1}$, and the i -th standard basis vector by $\mathbf{e}^{(i)}$. We do not explicitly specify their dimensions when they are clear from context. For a matrix \mathbf{M} , we denote its (i, j) -entry by \mathbf{M}_{ij} . If S and T are subsets of its row and column indices, respectively, then \mathbf{M}_{ST} denotes the submatrix formed by the entries $\{(i, j) : i \in S, j \in T\}$.

For $\mathbf{v} \in \mathbb{R}^n$, let $\text{diag}(\mathbf{v}) \in \mathbb{R}^{n \times n}$ denote the diagonal matrix with the entries of \mathbf{v} on its diagonal. For $\mathbf{M} \in \mathbb{R}^{n \times n}$, let $\text{diag}(\mathbf{M}) \in \mathbb{R}^{n \times n}$ denote the diagonal matrix obtained from \mathbf{M} by zeroing out all off-diagonal entries. The Moore–Penrose pseudoinverse of \mathbf{M} is denoted by \mathbf{M}^\dagger .

Definition 1.1 (RDDL and SDDM Matrices). *A matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ is called an L -matrix if $\mathbf{M}_{ij} \leq 0$ for all $i \neq j$ and $\mathbf{M}_{ii} > 0$ for all $i \in [n]$. It is row diagonally dominant (RDD) if for every $i \in [n]$,*

$$|\mathbf{M}_{ii}| \geq \sum_{j \in [n] \setminus \{i\}} |\mathbf{M}_{ij}|.$$

We call \mathbf{M} RDDL if it is both an L -matrix and row diagonally dominant. An SDDM matrix is an invertible symmetric RDDL matrix.

In this paper, we focus exclusively on SDDM matrices. An equivalent characterization is that such matrices are principal submatrices of undirected graph Laplacians. We refer to the corresponding graph as the *associated graph* of the SDDM matrix, defined as follows.

Definition 1.2 (Associated Graph of an RDDL Matrix). *For an RDDL matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$, we define the associated weighted graph $G = ([n+1], E, w)$ by adding a dummy vertex $n+1$. For every pair $i, j \in [n]$ with $i \neq j$, we set the edge weight $w(i, j) = -\mathbf{M}_{ij}$. Additionally, we set the edge weight $w(i, n+1) = \sum_{j \in [n]} \mathbf{M}_{ij}$.*

The associated graph of an SDDM matrix is defined analogously. The solution of a linear system with an SDDM matrix is closely connected to random walks on its associated graph, which we define next.

Random walks. Given an undirected weighted graph $G = ([n], E, w)$ with edge weights $w \in \mathbb{R}_{\geq 0}^E$, a random walk on G moves from a vertex to one of its neighbors independently of all previous steps.

If the walk is currently at vertex i , it moves to $j \in N(i)$ in the next step with probability

$$\mathbf{A}_{ij} := \frac{w(i, j)}{\sum_{k \in N(i)} w(i, k)}.$$

In other words, \mathbf{A} is the transition matrix of the Markov chain associated with G . For unweighted graphs, this simplifies to a uniform transition probability $1/|N(i)|$ for each neighbor.

Definition 1.3 (Escape Probability). *The escape probability $\mathbb{P}(s, t, p)$ denotes the probability that a random walk starting at vertex s visits vertex t before vertex p .*

The following result characterizes the relationship between the entries of the inverse of an invertible SDDM matrix and the escape probabilities in its associated graph.

Lemma 1.4 (Lemma 2.7 of [GN26]). *Let $\mathbf{L} \in \mathbb{R}^{n \times n}$ be an invertible RDDM matrix. Then, for all $s, t \in [n]$,*

$$\mathbb{P}(s, t, n+1) = \frac{(\mathbf{L}^{-1})_{st}}{\mathbf{L}_{tt}(\mathbf{L}^{-1})_{tt}}.$$

Furthermore, $(\mathbf{L}^{-1})_{tt} > 0$ for every $t \in [n]$.

We next recall a few linear algebraic notations and definitions that we use throughout the paper.

Norms. The ℓ_1 -norm and ℓ_∞ -norm of a vector $\mathbf{v} \in \mathbb{R}^n$ are defined as $\|\mathbf{v}\|_1 := \sum_{i \in [n]} |\mathbf{v}_i|$ and $\|\mathbf{v}\|_\infty := \max_{i \in [n]} |\mathbf{v}_i|$, respectively. The induced ℓ_1 and ℓ_∞ norms for a matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ are defined as

$$\begin{aligned} \|\mathbf{M}\|_1 &:= \max_{\mathbf{v} \in \mathbb{R}^n: \|\mathbf{v}\|_1=1} \|\mathbf{M}\mathbf{v}\|_1, \\ \|\mathbf{M}\|_\infty &:= \max_{\mathbf{v} \in \mathbb{R}^n: \|\mathbf{v}\|_\infty=1} \|\mathbf{M}\mathbf{v}\|_\infty. \end{aligned}$$

Both vector and induced norms satisfy the triangle inequality and consistency property; for example, $\|\mathbf{A}\mathbf{B}\|_\infty \leq \|\mathbf{A}\|_\infty \cdot \|\mathbf{B}\|_\infty$. We also define $\|\mathbf{A}\|_{\max} := \max_{i,j} |\mathbf{A}_{ij}|$. The *energy norm* with respect to a positive definite matrix \mathbf{L} is defined as

$$\|\mathbf{v}\|_{\mathbf{L}} := \sqrt{\mathbf{v}^\top \mathbf{L} \mathbf{v}}.$$

Schur complement. Given a block matrix

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix},$$

the *Schur complement* of \mathbf{M} with respect to \mathbf{A} is $\text{Sc}(\mathbf{M}, \mathbf{A}) = \mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B}$, assuming \mathbf{A} is invertible. If both $\mathbf{S} := \text{Sc}(\mathbf{M}, \mathbf{A})$ and \mathbf{A} are invertible, then \mathbf{M} is invertible and

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}\mathbf{S}^{-1}\mathbf{C}\mathbf{A}^{-1} & -\mathbf{A}^{-1}\mathbf{B}\mathbf{S}^{-1} \\ -\mathbf{S}^{-1}\mathbf{C}\mathbf{A}^{-1} & \mathbf{S}^{-1} \end{bmatrix}. \quad (1)$$

Solving SDDM linear systems with normwise error bounds. The following result by Spielman and Teng [ST14] was a breakthrough, providing a near-linear-time algorithm for solving SDDM linear systems with normwise error guarantees.

Lemma 1.5 (Theorem 5.5 of [ST14]). *Let $\mathbf{L} \in \mathbb{R}^{n \times n}$ be an SDDM matrix with m nonzero entries, and let $\mathbf{b} \in \mathbb{R}^n$ have integer entries in $[-U, U]$. For any $\epsilon > 0$ and $\delta \in (0, 1)$, there exists an algorithm that, with probability at least $1 - \delta$ and using $\tilde{O}(m \log(U\epsilon^{-1}\delta^{-1}) \log(\epsilon^{-1}))$ bit operations, computes a vector $\tilde{\mathbf{x}}$ such that*

$$\|\tilde{\mathbf{x}} - \mathbf{L}^\dagger \mathbf{b}\|_{\mathbf{L}} \leq \epsilon \cdot \|\mathbf{L}^\dagger \mathbf{b}\|_{\mathbf{L}}.$$

Moreover, the entries of $\tilde{\mathbf{x}}$ can be represented using $O(n \log(U\epsilon^{-1}))$ bits in fixed-point arithmetic.

The above result bounds the error with respect to the energy norm. For our purposes, it is often more convenient to work with the Euclidean norm. **Corollary 1.7**, which follows directly from **Lemma 1.5** together with the following bound on the condition number of invertible SDDM matrices, provides such an error guarantee in the Euclidean norm.

Lemma 1.6 (Corollary 2.5 of [GNY26]). *Let \mathbf{L} be an $n \times n$ invertible SDDM matrix with integer entries in $[-U, U]$. Then*

$$\|\mathbf{L}^{-1}\|_2 = \frac{1}{\lambda_{\min}(\mathbf{L})} < n^2.$$

Corollary 1.7 (Corollary 3.2 of [GNY26]). *Let $\mathbf{L} \in \mathbb{R}^{n \times n}$ be an invertible SDDM matrix with m nonzero entries, and let $\mathbf{b} \in \mathbb{R}^n$ have integer entries in $[-U, U]$. For any $\epsilon > 0$ and $\delta \in (0, 1)$, there exists an algorithm that, with probability at least $1 - \delta$ and using $\tilde{O}(m \log^2(U\epsilon^{-1}\delta^{-1}))$ bit operations, computes a vector $\tilde{\mathbf{x}}$ such that*

$$\|\tilde{\mathbf{x}} - \mathbf{L}^\dagger \mathbf{b}\|_2 \leq \epsilon \|\mathbf{b}\|_2.$$

Moreover, the entries of $\tilde{\mathbf{x}}$ can be represented using $O(\log(nU\epsilon^{-1}))$ bits in fixed-point arithmetic.

1.2 Technical Overview

Our approach builds on the *threshold decay (TD)* framework introduced in [GNY26]. We first recall the main components of this framework and then explain how we adapt it to obtain an almost-linear-time algorithm for solving SDDM linear systems with entrywise approximation guarantees.

Threshold decay (TD) framework. The TD framework relies on three key observations:

1. A solution with normwise error guarantees (for a polynomially small error parameter) also provides entrywise approximations for the large coordinates of the solution. Moreover, every solution to a linear system has at least one large entry.
2. Given a subset of entries $F \subseteq [n]$, if we substitute an entrywise approximation for the entries in F into the system, the resulting smaller linear system is only mildly affected. More precisely, substituting an $\exp(\epsilon)$ -entrywise approximation for F changes the solution on $[n] \setminus F$ by at most a multiplicative factor of $\exp(\epsilon)$.
3. Removing small entries of the solution introduces only a limited error. Specifically, if $C \subseteq [n]$ satisfies $(\mathbf{L}^{-1}\mathbf{b})_i < \theta$ for all $i \in C$, then with $F = [n] \setminus C$,

$$\|[(\mathbf{L}_{FF})^{-1}\mathbf{b}_F; \mathbf{0}] - \mathbf{L}^{-1}\mathbf{b}\|_2 \leq n^3 U \theta.$$

The first two observations imply that normwise guarantees can be leveraged to recover entry-wise approximations for some coordinates of the solution, which can then be substituted back to form a smaller system. Repeating this process allows us to iteratively recover approximations for progressively smaller entries. The third observation shows that if we knew in advance which entries were small, we could safely remove them and focus computation on a reduced system that preserves accuracy for the large entries.

These insights naturally motivate the following iterative algorithm. We maintain three disjoint subsets of indices partitioning $[n]$: (i) a *solved set* P containing entries whose entrywise approximations have been computed, (ii) an *inactive set* Q of entries deemed small enough to omit, and (iii) an *active set* R of remaining entries participating in the current linear system. At each iteration, the algorithm maintains a threshold parameter θ and proceeds as follows:

1. Add to R all entries in Q whose magnitude is at least θ . Update Q as $Q \leftarrow [n] \setminus (R \cup P)$.
2. Solve the linear system with coefficient matrix \mathbf{L}_{RR} and recover entrywise approximations for the entries in R that are large relative to θ . Move these entries from R to P .
3. Substitute the newly solved entries into the system to obtain an updated right-hand-side vector \mathbf{b} .
4. Reduce the threshold θ by a polynomial factor in nU and repeat.

We remark that the first step is the most technically challenging and constitutes our main contribution. To make the algorithm efficient, we must restrict the system solve to the active set R . However, this restriction provides no direct information about the large entries in Q . The core difficulty is to identify these large entries without explicitly solving on Q . Our key idea is to achieve this through carefully designed *predictions*.

TD with prediction. Achieving almost-linear running time requires keeping the active set small on average across iterations. This, in turn, depends on having good *a priori* predictions for the magnitudes of the solution entries. A notable instance where such predictions are available is the computation of the inverse of an SDDM matrix (see [GN26]). Specifically, for any pair (i, j) with $\mathbf{L}_{ij} \neq 0$, one has

$$\frac{1}{nU} \mathbf{L}^{-1} \mathbf{e}^{(j)} \leq \mathbf{L}^{-1} \mathbf{e}^{(i)} \leq nU \mathbf{L}^{-1} \mathbf{e}^{(j)},$$

implying that the i -th column of \mathbf{L}^{-1} provides reliable predictions for the j -th column. Using this property, [GN26] obtained an $\tilde{O}(mn)$ -time algorithm for computing the inverse of an SDDM matrix. Once one column of \mathbf{L}^{-1} is computed, each subsequent column can be obtained in $\tilde{O}(m)$ time.

However, this prediction only applies when inverting the entire matrix, i.e., solving $O(n)$ systems. For a single system, we must establish an *upper bound* on the small entries in order to identify which entries can be safely ignored when extracting the large ones. Directly using the inequality gives only *lower bounds* on the small entries, making it useless here.

Low-diameter (LD) cover. Obtaining accurate predictions for a single linear system requires a more sophisticated approach. In this paper, we obtain such predictions by first computing a *low-diameter cover* for the graph associated with \mathbf{L} , where we define the distance between two vertices i and j in terms of their corresponding entry in the inverse:

$$D_{\mathbf{L}}(i, j) := -\log_{nU}((\mathbf{L}^{-1})_{ij}) + 2.$$

An important property of this distance function is that it satisfies the triangle inequality.

We construct a collection of pairs (V, W) of *inner and outer balls*, where $V \subseteq W \subseteq [n]$, satisfying the following properties:

- (i) Every vertex is contained in at least one inner ball.
- (ii) Each vertex is contained in at most $n^{o(1)}$ outer balls.
- (iii) For any pair i, j within the same outer ball, $D_L(i, j) \leq n^{o(1)}$.
- (iv) If i lies in an inner ball and j lies outside its corresponding outer ball, then $D_L(i, j) \geq n^{\omega(1)}$.

We next explain how such a low-diameter cover can be leveraged to solve the linear system in almost-linear time, and subsequently describe how to compute such a cover efficiently.

Almost linear-time solver using LD cover. In iteration t of the TD framework, let $P^{(t)}$ denote the solved set where the entrywise approximations of the entries have been computed. We now solve on the reduced system on $[n] \setminus P^{(t)}$ (obtained by plugging in the solved entries in $P^{(t)}$)

$$\mathbf{L}_{[n] \setminus P^{(t)}, [n] \setminus P^{(t)}} \mathbf{x} = \hat{\mathbf{b}}^{(t)}$$

where $\hat{\mathbf{b}}^{(t)} \approx \mathbf{b}_{[n] \setminus P^{(t)}} - \mathbf{L}_{[n] \setminus P^{(t)}, P^{(t)}} \tilde{\mathbf{x}}^{(t)}$ is the right-hand-side vector of the reduced system.

It can be shown that (see Lemma 3.8 of [GNY26]) the nonzero entries of $\hat{\mathbf{b}}^{(t)}$ differ by at most a polynomial (in U) factor. This allows us to invoke the near-linear-time solver of [ST14] to compute a solution with normwise error guarantees in each iteration.

Let $I^{(t)}$ denote the set of nonzero entries of $\hat{\mathbf{b}}^{(t)}$. The entries indexed by $I^{(t)}$ in the solution $(\mathbf{L}_{[n] \setminus P^{(t)}, [n] \setminus P^{(t)}})^{-1} \hat{\mathbf{b}}^{(t)}$ are large—this follows directly from a simple application of Lemma 1.4. We use $I^{(t)}$ as a seed to construct the active set, ensuring that the inactive set contains only small entries of the solution.

We define the *boundary-expanded set* (see Definition 3.4) of $I^{(t)}$ as the union of inner balls whose corresponding outer balls intersect $I^{(t)}$. Due to the definition of the LD cover, if a vertex $u \in [n]$ does not belong to the boundary-expanded set of $I^{(t)}$, the corresponding outer ball of the inner ball that contains u does not intersect $I^{(t)}$, then the probability distance between u and any $v \in I^{(t)}$ is at least $n^{\omega(1)}$. This implies that the entry at u is small and can be omitted in iteration t of the TD framework. Therefore, the active set in iteration t , denoted by $H^{(t)}$, is defined as the intersection of $[n] \setminus P^{(t)}$ and the boundary-expanded set of $I^{(t)}$.

Moreover, we show that the total size of all $H^{(t)}$'s satisfies

$$\sum_t |H^{(t)}| \leq n^{1+o(1)}.$$

By properties of the TD framework, every vertex is added to and deleted from the active set exactly once. When a vertex u is added to the active set, the distance d to the solved set is bounded by the radius of an outer ball in the LD cover, i.e., $n^{o(1)}$. This gives a lower bound on the u -th entry of the solution $(\mathbf{L}^{-1} \mathbf{b})_u \geq \theta \cdot (nU)^{-d}$, where θ is the current threshold. Since the threshold is reduced by a polynomial factor in nU in each iteration, the vertex will be removed from the active set and added to the solved set after $O(d) = n^{o(1)}$ iterations, concluding the bound on total size of $H^{(t)}$'s.

Maintaining the sets $I^{(t)}$ and $H^{(t)}$ requires additional care to ensure almost-linear running time:

1. It can be shown that the right-hand-side vector $\hat{\mathbf{b}}^{(t)}$ undergoes only $O(m + n)$ updates in total. Hence, the sets $I^{(t)}$ can be maintained in $O(m + n)$ time overall.
2. The set $H^{(t)}$ is maintained by keeping a counter for each pair in the LD cover that tracks the number of vertices in $I^{(t)}$ contained in the corresponding outer ball. Then $H^{(t)}$ is the union of the inner balls of all pairs with nonzero counters. Since the number of updates to $I^{(t)}$ is bounded and each vertex is covered by a limited number of outer balls, these counters can be updated in almost-linear time. Moreover, since

$$\sum_t |H^{(t)}| \leq n^{1+o(1)},$$

and each vertex is covered by at most $n^{o(1)}$ outer balls, using the property of our TD framework, the total time to construct all sets $H^{(t)}$ is at most $n^{1+o(1)}$ over the course of the algorithm.

Constructing the LD cover. To explain the high-level ideas of our algorithm for constructing the LD cover, we assume exact solutions to linear systems. However, in [Section 2](#), we carefully address bit complexity and the errors introduced when solving linear systems approximately.

Consider the solution to the linear system $\mathbf{L}\mathbf{x} = \mathbf{e}^{(i)}$. Let

$$V = \{j \in [n] : (\mathbf{L}^{-1}\mathbf{e}^{(i)})_j \geq 2^{-\sqrt{\log n}}\} \quad \text{and} \quad W = \{j \in [n] : (\mathbf{L}^{-1}\mathbf{e}^{(i)})_j \geq 4^{-\sqrt{\log n}}\}.$$

Then $V \subseteq W$. By the triangle inequality for $D_{\mathbf{L}}$, one can verify that for the inner ball V and the outer ball W , properties (iii) and (iv) of the LD cover are satisfied. Moreover, i is contained in V , and therefore property (i) is satisfied for i . Thus, one may construct a pair of balls for each $i \in [n]$ by considering the large entries of the solution to $\mathbf{L}\mathbf{x} = \mathbf{e}^{(i)}$.

This naïve approach encounters two main issues: (1) it requires solving n different linear systems; and (2) each vertex may belong to too many outer balls. To address these issues, we instead attempt to construct outer/inner ball pairs for multiple vertices simultaneously by solving a linear system with right-hand side $\mathbf{1}_S = \sum_{i \in S} \mathbf{e}^{(i)}$. By linearity, the large entries of the solution $\mathbf{L}\mathbf{x} = \mathbf{1}_S$ roughly give the *union* of balls for $i \in S$. An important observation is that if i is far (in $D_{\mathbf{L}}$ distance) from all $j \in S \setminus \{i\}$, then the pair of balls corresponding to i form an isolated connected component from the union of the rest, and therefore can be recovered from the solution.

To obtain such an S , we select a random subset of $[n]$, where each vertex is included independently. Moreover, instead of using the fixed thresholds $2^{-\sqrt{\log n}}$ and $4^{-\sqrt{\log n}}$ for the inner and outer balls, we consider multiple thresholds that are exponential in a set of distance parameters d_i to ensure that vertices in S remain sufficiently far from one another.

Therefore, we construct the LD cover by solving a *collection of linear systems* with coefficient matrix \mathbf{L} and random right-hand-side vectors, and by recovering inner/outer balls from the large entries of the resulting solutions. We consider $O(\sqrt{\log n})$ different probabilities p_j (for selecting vertices to set S) and $O(\sqrt{\log n})$ different distance thresholds d_i in a nested loop. For each pair (d_i, p_j) , we run $n^{o(1)}$ independent iterations. In each iteration, every vertex u is included in S independently with probability p_j , forming a random subset S . We then compute a normwise approximate solution with error parameter exponential in d_i to the linear system

$$\mathbf{L}\mathbf{x} = \mathbf{1}_S.$$

Let T denote the set of vertices corresponding to the *large* entries (i.e., entries larger than $\exp(-d_i/2)$) in the resulting solution. If a connected component of the subgraph induced by T

contains exactly one vertex of S , then we construct a pair of inner and outer balls (V, W) on that connected component such that the distance between any vertex inside the inner ball and any vertex outside the outer ball is at least $n^{\omega(1)}$, and the distance between any two vertices within the outer ball is at most $n^{o(1)}$. We construct inner/outer ball pairs only from such connected components and add them to our cover.

This constraint on the connected components ensures that the vertex in S contained in that component is far from all other vertices in S . Consequently, the construction satisfies properties (iii) and (iv) of the LD cover.

We show that for every vertex $u \in [n]$, there exists a pair (d_i, p_j) such that, with probability at least $n^{-o(1)}$, the random set S contains a vertex near u (that is, within its corresponding inner ball) and that vertex is the unique element of S in its connected component of the subgraph induced by T , which is our requirement for constructing inner/outer balls as explained above. The repeated iterations in the inner loop amplify the probability of this event. Consequently, with high probability, every vertex is contained in at least one inner ball. Moreover, since in each iteration we only include disjoint outer balls, each vertex is covered by at most $n^{o(1)}$ outer balls. Hence, properties (i) and (ii) also hold.

1.3 Related Work

Solving linear systems with entrywise approximation guarantees dates back to the 1980s and 1990s. From a theoretical point of view, Higham [Hig90] observed that fast matrix multiplication methods such as Strassen’s [Str69] cannot achieve entrywise approximations for matrix multiplication (which, via standard reductions, can be used for solving linear systems) due to their inherent reliance on subtractions, even when applied to nonnegative matrices.

On the other hand, empirical studies demonstrated that Gaussian elimination is significantly more stable when computing the stationary distribution of Markov chains [Hey87, GTH85, HP84] compared to its application to general matrices. These studies attributed the improved stability to the fact that Gaussian elimination, when applied to SDDM matrices, does not require subtractions and is therefore highly compatible with floating-point arithmetic.

Subsequent works further analyzed the error bounds and the number of floating-point operations required for computing the stationary distribution [O’C93] and for performing LU factorization [O’C96] with entrywise approximation guarantees. All of these algorithms require cubic time. Very recently, [GNY26] studied this problem for both SDDM and RDDDL matrices with the goal of designing subcubic-time algorithms. In particular, for solving a linear system with an SDDM matrix, they presented an algorithm with running time $\tilde{O}(m\sqrt{n}\log^2(U\epsilon^{-1}\delta^{-1}))$. Moreover, for inverting an SDDM matrix, they provided an algorithm with running time $\tilde{O}(mn\log^2(U\epsilon^{-1}\delta^{-1}))$.

The recent work of Gao, Kyng, and Spielman [GKS23] has demonstrated the superior practical performance of near-linear-time Laplacian solvers. We hope that our approach can further advance practical solvers, particularly in settings where edge weights vary widely—for example, in solving linear systems that arise as subroutines in interior-point methods for flow problems.

There also exist algorithms in the literature for decomposing graphs into components with small diameter or covering them with low-diameter components, with applications to shortest-path problems and parallel SDDM solvers [Coh98, BGK⁺11]. However, these low-diameter decompositions and covers differ substantially from ours. To begin with, our covers are defined using pairs of inner and outer balls, with the guarantee that vertices inside an inner ball are far from those outside the corresponding outer ball. In contrast, the decompositions and covers in the literature are defined using single clusters or balls of small diameter. Moreover, our notion of distance is based on the

probability distance

$$D_{\mathbf{L}}(i, j) := -\log_{nU}((\mathbf{L}^{-1})_{ij}) + 2,$$

as opposed to the standard (weighted or unweighted) graph distance. Most importantly, our access to the distance $D_{\mathbf{L}}(\cdot, \cdot)$ is highly restricted since it requires computing entries of the inverse. We address this limitation via a carefully designed oracle query, obtained by solving the system on a random vector with normwise error guarantees.

2 Low-Diameter Cover

In this section, we introduce our definition of a low-diameter cover and present an algorithm for constructing it with high probability using SDDM solvers with normwise error bounds as a subroutine. We note that the existence of such low-diameter covers is nontrivial, and the proof of correctness of our algorithm also serves as a constructive proof of their existence.

The diameters of the components in our cover are measured with respect to a distance function that we call the *probability distance*. We define this notion and establish some of its useful properties in [Section 2.1](#). In [Section 2.2](#), we formally define our low-diameter cover and present an algorithm for constructing it.

2.1 Probability Distance

The following distance function, defined in terms of the entries of the inverse of an SDDM matrix, forms the backbone of our low-diameter cover construction.

Definition 2.1 (Probability distance). *Let $\mathbf{L} \in \mathbb{Z}^{n \times n}$ be an invertible SDDM matrix with integer entries in $[-U, U]$. We define the probability distance with respect to \mathbf{L} as*

$$D_{\mathbf{L}}(i, j) := -\log_{nU}((\mathbf{L}^{-1})_{ij}) + 2, \quad \text{for all } i, j \in [n].$$

The following result establishes some basic properties of the probability distance, including the triangle inequality and the fact that the distance between two adjacent vertices is small.

Claim 2.2 (Properties of the probability distance). *Let $\mathbf{L} \in \mathbb{Z}^{n \times n}$ be an invertible SDDM matrix with integer entries in $[-U, U]$. Then, for all $i, j, k \in [n]$, the probability distance $D_{\mathbf{L}}$ satisfies:*

1. $D_{\mathbf{L}}(i, i) \geq 0$.
2. $D_{\mathbf{L}}(i, j) = D_{\mathbf{L}}(j, i)$.
3. $D_{\mathbf{L}}(i, k) \leq D_{\mathbf{L}}(i, j) + D_{\mathbf{L}}(j, k)$.
4. For adjacent vertices i, j (i.e., $\mathbf{L}_{ij} \neq 0$), we have $D_{\mathbf{L}}(i, j) \leq 4$.

Proof. The first property follows from [Lemma 1.6](#) which states that the largest eigenvalue of \mathbf{L}^{-1} is at most n^2 . The second property holds because \mathbf{L} is symmetric. For the fourth property, let $\mathbf{D} := \text{diag}(\mathbf{L})$ and $\mathbf{A} = \mathbf{D}^{-1}(\mathbf{D} - \mathbf{L})$ and i, j be adjacent. By [Lemma 1.4](#)

$$\begin{aligned} (\mathbf{L}^{-1})_{ij} &= \mathbb{P}(i, j, n+1) \mathbf{L}_{jj} (\mathbf{L}^{-1})_{jj} \\ &= \mathbb{P}(i, j, n+1) \mathbf{L}_{jj} ((\mathbf{D}(\mathbf{I} - \mathbf{A}))^{-1})_{jj} \\ &= \mathbb{P}(i, j, n+1) \mathbf{L}_{jj} \mathbf{D}_{jj}^{-1} ((\mathbf{I} - \mathbf{A})^{-1})_{jj} \\ &= \mathbb{P}(i, j, n+1) \mathbf{L}_{jj} \mathbf{D}_{jj}^{-1} (\mathbf{I} + \mathbf{A} + \mathbf{A}^2 + \cdots)_{jj} \end{aligned}$$

$$\geq U^{-2},$$

where the inequality holds because $\mathbb{P}(i, j, n+1), \mathbf{D}_{jj}^{-1} \geq U^{-1}$ and $\mathbf{L}_{jj}, (\mathbf{I} + \mathbf{A} + \mathbf{A}^2 + \dots)_{jj} \geq 1$. Therefore

$$D_{\mathbf{L}}(i, j) = -\log_{nU}((\mathbf{L}^{-1})_{ij}) + 2 \leq -\log_{nU}(U^{-2}) + 2 \leq 4.$$

Now we prove the third property, i.e., the triangle inequality. By [Lemma 1.6](#),

$$(\mathbf{L}^{-1})_{jj} \leq n^2.$$

Therefore since $\mathbf{L}_{jj} \leq U$,

$$\log_{nU}(\mathbf{L}_{jj}(\mathbf{L}^{-1})_{jj}) \leq 2. \quad (2)$$

Now note that due to Markov property,

$$\mathbb{P}(i, j, n+1) \cdot \mathbb{P}(j, k, n+1) \leq \mathbb{P}(i, k, n+1).$$

Therefore

$$(\mathbb{P}(i, j, n+1) \cdot \mathbf{L}_{jj}(\mathbf{L}^{-1})_{jj}) \cdot (\mathbb{P}(j, k, n+1) \cdot \mathbf{L}_{kk}(\mathbf{L}^{-1})_{kk}) \leq (\mathbb{P}(i, k, n+1) \cdot \mathbf{L}_{kk}(\mathbf{L}^{-1})_{kk}) \cdot (\mathbf{L}_{jj}(\mathbf{L}^{-1})_{jj}).$$

Thus by [Lemma 1.4](#),

$$(\mathbf{L}^{-1})_{i,j} \cdot (\mathbf{L}^{-1})_{j,k} \leq (\mathbf{L}^{-1})_{i,k} \cdot (\mathbf{L}_{jj}(\mathbf{L}^{-1})_{jj}).$$

Taking the logarithm and using [Eq \(2\)](#), we have

$$\log(\mathbf{L}^{-1})_{i,j} + \log(\mathbf{L}^{-1})_{j,k} \leq \log(\mathbf{L}^{-1})_{i,k} + \log(\mathbf{L}_{jj}(\mathbf{L}^{-1})_{jj}) \leq \log(\mathbf{L}^{-1})_{i,k} + 2.$$

Therefore

$$D_{\mathbf{L}}(i, j) + D_{\mathbf{L}}(j, k) = -\log(\mathbf{L}^{-1})_{i,j} - \log(\mathbf{L}^{-1})_{j,k} + 4 \geq -\log(\mathbf{L}^{-1})_{i,k} - 2 + 4 = D_{\mathbf{L}}(i, k).$$

□

Finally, we conclude this part by showing that the principal submatrices of an invertible SDDM matrix \mathbf{L} induce larger distances than \mathbf{L} itself. We will use this property in our almost-linear-time algorithm to establish that if two vertices are far from each other with respect to \mathbf{L} , then they remain far with respect to any principal submatrix of \mathbf{L} .

Lemma 2.3 (Monotonicity of distance). *Let $\mathbf{L} \in \mathbb{Z}^{n \times n}$ be an invertible SDDM matrix, and let $\mathbf{L}_{S,S}$ denote the principal submatrix indexed by $S \subseteq [n]$. Then, for all $i, j \in S$, we have*

$$D_{\mathbf{L}}(i, j) \leq D_{\mathbf{L}_{S,S}}(i, j).$$

Proof. Let $\mathbf{D} := \text{diag}(\mathbf{L})$ and

$$\mathbf{A} := \mathbf{D}^{-1}(\mathbf{D} - \mathbf{L}) = \mathbf{I} - \mathbf{D}^{-1}\mathbf{L},$$

which represents the transition matrix of the associated graph to \mathbf{L} . Then $\mathbf{L} = \mathbf{D}(\mathbf{I} - \mathbf{A})$. Therefore, $\mathbf{L}^{-1} = (\mathbf{I} - \mathbf{A})^{-1}\mathbf{D}^{-1}$. We also have $\mathbf{L}_{S,S} = \mathbf{D}_{S,S}(\mathbf{I} - \mathbf{A})_{S,S}$ and $(\mathbf{L}_{S,S})^{-1} = ((\mathbf{I} - \mathbf{A})_{S,S})^{-1}(\mathbf{D}_{S,S})^{-1}$. By invertibility of \mathbf{L} ,

$$(\mathbf{I} - \mathbf{A})^{-1} = \mathbf{I} + \mathbf{A} + \mathbf{A}^2 + \dots,$$

$$(\mathbf{I}_{S,S} - \mathbf{A}_{S,S})^{-1} = \mathbf{I} + \mathbf{A}_{S,S} + (\mathbf{A}_{S,S})^2 + \dots$$

Consequently, for any $i, j \in S$ we have

$$(\mathbf{L}^{-1})_{ij} = \frac{1}{\mathbf{L}_{jj}} \sum_{k=0}^{\infty} (\mathbf{A}^k)_{ij}, \quad ((\mathbf{L}_{S,S})^{-1})_{ij} = \frac{1}{\mathbf{L}_{jj}} \sum_{k=0}^{\infty} ((\mathbf{A}_{S,S})^k)_{ij}.$$

It follows from the nonnegativity of \mathbf{A} that $(\mathbf{A}^k)_{ij} \geq ((\mathbf{A}_{S,S})^k)_{ij}$ for all k , and thus $(\mathbf{L}^{-1})_{ij} \geq ((\mathbf{L}_{S,S})^{-1})_{ij}$. The result then follows from the definition of the distance. \square

2.2 Constructing a Cover with Low Probability Distance Diameter

In this section, we first present the definition of our low-diameter cover, which is characterized by three parameters: r_{in} , r_{out} , and α . The parameter α serves as an upper bound on the number of outer balls that cover a single vertex. r_{in} is an upper bound on the diameter of the outer balls, and consequently also on the diameter of the inner balls. Finally, r_{out} provides a lower bound on the distance between vertices inside an inner ball and those outside its corresponding outer ball.

Definition 2.4 ($(r_{\text{in}}, r_{\text{out}}, \alpha)$ -cover). *Given an n -by- n invertible SDDM matrix \mathbf{L} with integer entries in $[-U, U]$, we say a set of pairs $\mathcal{C} = \{(V_1, W_1), (V_2, W_2), \dots, (V_k, W_k)\}$, with $V_i, W_i \subseteq [n]$, for all $i \in [k]$, is a $(r_{\text{in}}, r_{\text{out}}, \alpha)$ -cover for \mathbf{L} if it satisfies the following where $D_{\mathbf{L}}$ is a probability distance with respect to \mathbf{L} .*

1. $V_i \subseteq W_i \subseteq [n]$ for all $i \in [k]$.
2. For every $u \in [n]$, there exists $i \in [k]$ such that $u \in V_i$.
3. For every $u \in [n]$, $|\{i \in [k] : u \in W_i\}| \leq \alpha$.
4. For any $i \in [k]$, and any $u, v \in W_i$, $D_{\mathbf{L}}(u, v) \leq r_{\text{in}}$.
5. For any $i \in [k]$, and any $u \in V_i$, $v \in [n] \setminus W_i$, $D_{\mathbf{L}}(u, v) > r_{\text{out}}$.

We colloquially refer to V_i 's as inner balls and W_i 's as outer balls.

The following result states that a low-diameter cover with suitable values of the parameters r_{in} , r_{out} , and α exists and can be computed in almost-linear time with high probability. The remainder of this section is devoted to proving this theorem.

Theorem 2.1. *Let $U > 32$ and $n > 10$. There exists an algorithm that given $\delta \in (0, 1)$ and an n -by- n invertible SDDM matrix \mathbf{L} with integer entries in $[-U, U]$, with probability at least $1 - \delta$, computes an $(r_{\text{in}}, r_{\text{out}}, \alpha)$ -cover \mathcal{C} for \mathbf{L} with*

$$r_{\text{in}} = 2^{2\ell+1}, \quad r_{\text{out}} = 2^{\ell-2}, \quad \alpha = 6\ell^2 \cdot 16^\ell \cdot \lceil \log(n/\delta) \rceil, \quad \ell = \lceil \sqrt{\log n} \rceil + 3$$

in $m2^{O(\sqrt{\log n})} \log(U) \log(\delta^{-1}) \log(U\delta^{-1})$ bit operations.

Proof. We first prove that the output of **LOWDIAMCONSTRUCT** satisfies all the properties in **Definition 2.4** with probability at least $1 - \delta$. With probability at least $1 - \delta/2$, all the linear systems in Step 4(a)ii of **LOWDIAMCONSTRUCT** are solved correctly. Therefore for the rest of the proof, we assume that we have the error guarantee of Step 4(a)ii.

LowDiamConstruct: Constructing a low-diameter cover.

Input: \mathbf{L} , an $n \times n$ invertible SDDM matrix with integer entries in $[-U, U]$, and $\delta \in (0, 1)$

Output: $\{(V_i, W_i)\}_i$ with properties stated in [Definition 2.4](#)

1. Let $\ell = \lceil \sqrt{\log n} \rceil + 3$, for $i \in [\ell]$, $d_i = \frac{4^\ell}{2^{i-1}}$, $p_i = \min\{\frac{1}{n} \cdot 2^{\ell(i-1)}, 1\}$.
2. Let M be the smallest power of two larger than nU .
3. Set $\mathcal{C} = \emptyset$.
4. For $i \in [\ell]$, $j \in [\ell]$:
 - (a) Repeat for $B := 6 \cdot 16^\ell \cdot \lceil \log(n/\delta) \rceil$ times:
 - i. Sample a set S by including each element of $[n]$ independently with probability p_j .
 - ii. Compute a vector \mathbf{y} such that with probability $1 - \delta/(2\ell^2 B)$ satisfies that $\|\mathbf{y} - \mathbf{L}^{-1} \mathbf{1}_S\|_2 \leq M^{-2d_i}$. E.g., using [Corollary 1.7](#).
 - iii. Let $T = \{k \in [n] : \mathbf{y}_k \geq M^{-d_i/2}\}$ and G_T be the induced subgraph on T of the graph corresponding to \mathbf{L} .
 - iv. For each $v \in S$, let C_v be the connected component of G_T that contains v . If for all $w \in S \setminus \{v\}$, $C_v \cap C_w = \emptyset$, then we set

$$V = \{k \in C_v : \mathbf{y}_k \geq M^{-d_i/4} - M^{-2d_i}\}$$
 and

$$W = \{k \in C_v : \mathbf{y}_k \geq M^{-(d_i/2)+2}\},$$
 and add (V, W) to \mathcal{C} .
5. Return \mathcal{C}

Figure 1: low-diameter construction

1. Since $U > 32$ and $n > 10$, we have $d_i \geq 32$ and $M > 320$. Therefore, $M^{-d_i/4} - M^{-2d_i} < M^{-(d_i/2)+2}$, and

$$\{k \in C_u : \mathbf{y}_k \geq M^{-d_i/4} - M^{-2d_i}\} \subseteq \{k \in C_u : \mathbf{y}_k \geq M^{-(d_i/2)+2}\},$$

for all $(V, W) \in \mathcal{C}$ we have $V \subseteq W$.

2. Let $u \in [n]$. First we show that there exists $i \in [\ell]$ such that

$$\left| \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-2d_i}\} \right| \leq 8^\ell \left| \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_i/4}\} \right|. \quad (3)$$

Suppose this does not hold for any $i \in [\ell]$. Then since $8d_{i+3} = 2d_{i+1} = d_i$, for $i \in [\ell - 3]$, we have

$$\left| \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_i}\} \right| > 8^\ell \left| \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_{i+3}}\} \right|.$$

Therefore

$$\left| \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_1}\} \right| > 8^{\ell \cdot (\lceil \ell/3 \rceil - 1)} \left| \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_\ell}\} \right|.$$

Now note that

$$8^{\ell \cdot (\lceil \ell/3 \rceil - 1)} = 2^{3\ell \cdot (\lceil \ell/3 \rceil - 1)} \geq 2^{\ell^2 - 3\ell} \geq 2^{\log n} = n,$$

where the last inequality follows from

$$\ell^2 - 3\ell = \ell \cdot \lceil \sqrt{\log n} \rceil > \log n.$$

Thus

$$\left| \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_1}\} \right| > n \cdot \left| \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_\ell}\} \right|. \quad (4)$$

Moreover since $(\mathbf{L}^{-1} \mathbf{e}^{(u)})_u \geq 1/U$, $|\{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_\ell}\}| \geq 1$. Additionally

$$\left| \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_1}\} \right| \leq n.$$

However this contradicts (4). Therefore there exists $i \in [\ell]$ such that

$$\left| \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-2d_i}\} \right| \leq 8^\ell \left| \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_i/4}\} \right|.$$

Now since $p_1 = \frac{1}{n}$ and $p_n = 1$, there exists $j \in [\ell - 1]$, such that $p_j < 1$

$$\frac{1}{p_j \cdot 2^\ell} \leq \frac{1}{p_{j+1}} \leq |\{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-2d_i}\}| \leq \frac{1}{p_j}. \quad (5)$$

Therefore for the i and j that satisfy (3) and (5), respectively, for any iteration of the inner loop (Step 4a of **LOWDIAMCONSTRUCT**), we have

$$\frac{1}{2^\ell} \leq \mathbb{E} \left[\left| S \cap \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-2d_i}\} \right| \right] \leq 1.$$

Let $\beta = |\{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-2d_i}\}|$. We have

$$\begin{aligned} \mathbb{P} \left[\left| S \cap \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-2d_i}\} \right| = 1 \right] &= \beta p_j (1 - p_j)^{\beta - 1} \\ &\geq \beta p_j e^{-p_j(\beta - 1)/(1 - p_j)} \\ &\geq \beta p_j e^{-p_j \beta / (1 - p_j)} \\ &\geq 2^{-\ell} e^{-1}. \end{aligned}$$

Therefore if S_1, \dots, S_{2^ℓ} is the sampled set in 2^ℓ different iterations of the inner loop (Step 4a of **LOWDIAMCONSTRUCT**), with probability at least $1 - e^{-e^{-1}} \geq 0.3$, there is a $q \in [2^\ell]$ such that

$$\left| S_q \cap \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-2d_i}\} \right| = 1.$$

Now condition on $\left| S_q \cap \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-2d_i}\} \right| = 1$, the probability that

$$\left| S_q \cap \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_i/4}\} \right| = 1$$

is

$$\frac{|\{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_i/4}\}|}{|\{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-2d_i}\}|} \geq \frac{1}{8^\ell},$$

where the inequality follows from (3). Therefore with probability at least $(1 - e^{-1}) \cdot 0.3 \geq \frac{1}{6}$ over 16^ℓ iterations of the inner loop, we pick at least one S such that

$$\left| S \cap \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_i/4}\} \right| = \left| S \cap \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-2d_i}\} \right| = 1. \quad (6)$$

Therefore over $B = 6 \cdot 16^\ell \cdot \lceil \log(2n/\delta) \rceil$ iterations, with probability at least $1 - \frac{\delta}{2n}$, we pick at least one such S .

Let v be the only element of $S \cap \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_i/4}\}$. Therefore

$$M^{-d_i/4} \leq (\mathbf{L}^{-1} \mathbf{e}^{(u)})_v \leq \sum_{k \in S} (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k = (\mathbf{L}^{-1} \mathbf{1}_S)_u \leq \mathbf{y}_u + M^{-2d_i}, \quad (7)$$

where the last inequality follow from $\|\mathbf{y} - \mathbf{L}^{-1} \mathbf{1}_S\|_2 \leq M^{-2d_i}$. Thus $\mathbf{y}_u \geq M^{-d_i/4} - M^{-2d_i}$ and $u \in V$. We now show that for all $w \in S \setminus \{v\}$, $C_v \cap C_w = \emptyset$. Since $U > 64$,

$$1 \leq \log_{nU}(M) \leq 1.2, \text{ and } 0.8 \leq \log_M(nU) \leq 1. \quad (8)$$

Note that by (6), since $\{v\} = S \cap \{k \in [n] : (\mathbf{L}^{-1} \mathbf{e}^{(u)})_k \geq M^{-d_i/4}\}$, for any $w' \in S \setminus \{v\}$, we have $(\mathbf{L}^{-1} \mathbf{e}^{(u)})_{w'} < M^{-2d_i}$, and therefore

$$-\log_M((\mathbf{L}^{-1})_{ww'}) > 2d_i,$$

and

$$D_{\mathbf{L}}(u, w') = -\log_{nU}((\mathbf{L}^{-1})_{ww'}) + 2 = \frac{-\log_M((\mathbf{L}^{-1})_{ww'})}{\log_M(nU)} + 2 \geq 2d_i + 2,$$

where the inequality follows from (8). By (7), $M^{-d_i/4} \leq (\mathbf{L}^{-1})_{uv}$, and thus

$$\frac{d_i}{4} \geq -\log_M((\mathbf{L}^{-1})_{uv}).$$

Therefore by (8),

$$\begin{aligned} 0.3d_i + 2 &= 1.2 \cdot \frac{d_i}{4} + 2 \geq -\log_{nU}(M) \log_M((\mathbf{L}^{-1})_{uv}) + 2 \\ &= -\log_{nU}((\mathbf{L}^{-1})_{uv}) + 2 = D_{\mathbf{L}}(u, v). \end{aligned} \quad (9)$$

For the sake of contradiction, suppose there exists $w \in S \setminus \{v\}$ with $w \in C_v$. Moreover let $v = k_0, k_1, \dots, k_{p-1}, k_p = w$ be an arbitrary path between v and w that goes through C_v , i.e., $k_0, \dots, k_p \in C_v$. By triangle inequality (Claim 2.2), for all $g \in [p]$, we have

$$D_{\mathbf{L}}(u, k_g) \leq D_{\mathbf{L}}(u, k_{g-1}) + D_{\mathbf{L}}(k_{g-1}, k_g).$$

Moreover by Claim 2.2, since for all $g \in [p]$, k_{g-1} and k_g are adjacent, $D_{\mathbf{L}}(k_{g-1}, k_g) \leq 4$. Therefore since $D_{\mathbf{L}}(u, v) \leq 0.3d_i + 2$, $D_{\mathbf{L}}(u, w) \geq 2d_i + 2$, and $d_i \geq 8$, there exists $q \in [p]$ such that

$$1.2 \cdot d_i + 2 \leq D_{\mathbf{L}}(u, k_q) \leq 1.2 \cdot d_i + 6$$

Therefore for all $w' \in S \setminus \{v\}$

$$2d_i + 2 \leq D_{\mathbf{L}}(u, w') \leq D_{\mathbf{L}}(u, k_q) + D_{\mathbf{L}}(k_q, w') \leq 1.2 \cdot d_i + 6 + D_{\mathbf{L}}(k_q, w'),$$

which gives

$$D_{\mathbf{L}}(k_q, w') \geq 0.8d_i - 4.$$

Moreover by triangle inequality

$$1.2 \cdot d_i + 2 \leq D_{\mathbf{L}}(u, k_q) \leq D_{\mathbf{L}}(u, v) + D_{\mathbf{L}}(v, k_q) \leq 0.3d_i + 2 + D_{\mathbf{L}}(v, k_q),$$

which gives

$$D_{\mathbf{L}}(v, k_q) \geq 0.9d_i.$$

Therefore

$$\begin{aligned} -\log_M(\mathbf{L}^{-1}\mathbf{1}_S)_{k_q} &= \frac{-\log_{nU}(\mathbf{L}^{-1}\mathbf{1}_S)_{k_q}}{\log_{nU}(M)} \geq \frac{-\log_{nU}(|S| \cdot \max_{w' \in S}(\mathbf{L}^{-1})_{k_q w'})}{\log_{nU}(M)} \\ &\geq \frac{-1 + \min_{w' \in S} D_{\mathbf{L}}(k_q, w') - 2}{\log_{nU}(M)} \geq \frac{0.8d_i - 7}{1.2} \geq 0.66d_i - 6. \end{aligned}$$

Thus

$$(\mathbf{L}^{-1}\mathbf{1}_S)_{k_q} \leq M^{-0.66d_i+6}.$$

Now since $U > 32$ and $n > 10$, we have $d_i \geq 32$, $M > 320$, and

$$\mathbf{y}_{k_q} \leq M^{-0.66d_i+6} + M^{-2d_i} < M^{-d_i/2}.$$

Therefore k_q is not in T and is not in the connected component of v . Therefore by this contradiction for all $w \in S \setminus \{v\}$, $C_v \cap C_w = \emptyset$.

3. This holds because in each iteration of the loop (i), we pick at most one outer ball W that contains a vertex u . Therefore each vertex is included in at most $6\ell^2 \cdot 16^\ell \cdot \lceil \log(n/\delta) \rceil$ outer balls.
4. For the last two parts, let $(V, W) \in \mathcal{C}$ be an inner ball, outer ball pair and i, j be their corresponding iterand in the outer loop. Moreover let v be the unique element in S that is inside the connected component C_v of G_T . Therefore

$$V = \{k \in C_v : \mathbf{y}_k \geq M^{-d_i/4} - M^{-2d_i}\}$$

and

$$W = \{k \in C_v : \mathbf{y}_k \geq M^{(-d_i/2)+2}\}.$$

Let $y \in W$. Then we have

$$(\mathbf{L}^{-1}\mathbf{1}_S)_y + M^{-2d_i} \geq \mathbf{y}_y \geq M^{(-d_i/2)+2}.$$

Moreover

$$(\mathbf{L}^{-1}\mathbf{1}_S)_y = (\mathbf{L}^{-1}\mathbf{e}^{(v)})_y + \sum_{w \in S \setminus \{v\}} (\mathbf{L}^{-1}\mathbf{e}^{(w)})_y.$$

Next, we show, for $w \in S \setminus \{v\}$,

$$(\mathbf{L}^{-1}\mathbf{e}^{(w)})_y < M^{-d_i/2}. \tag{10}$$

This is because for any $o \in [n] \setminus \{w\}$,

$$\sum_{q \in [n]} \mathbf{L}_{oq} (\mathbf{L}^{-1} \mathbf{e}^{(w)})_q = 0$$

Therefore

$$\sum_{q \in [n] \setminus \{o\}} \frac{-\mathbf{L}_{oq}}{\mathbf{L}_{oo}} (\mathbf{L}^{-1} \mathbf{e}^{(w)})_q = (\mathbf{L}^{-1} \mathbf{e}^{(w)})_o$$

Since $\sum_{q \in [n] \setminus \{o\}} \frac{-\mathbf{L}_{oq}}{\mathbf{L}_{oo}} \leq 1$, either $(\mathbf{L}^{-1} \mathbf{e}^{(w)})_o$ is equal to $(\mathbf{L}^{-1} \mathbf{e}^{(w)})_q$ for all q that are adjacent to o or for one neighbor q , $(\mathbf{L}^{-1} \mathbf{e}^{(w)})_q > (\mathbf{L}^{-1} \mathbf{e}^{(w)})_o$. Continuing this argument inductively, one can see that if $(\mathbf{L}^{-1} \mathbf{e}^{(w)})_y \geq M^{-d_i/2}$, then there is a path from y to w where for all vertices q on this path $(\mathbf{L}^{-1} \mathbf{e}^{(w)})_q \geq M^{-d_i/2}$. In this case $y \in C_w$ and therefore since y is also in C_v , we have $C_v \cap C_w \neq \emptyset$ which is a contradiction.

Therefore

$$(\mathbf{L}^{-1} \mathbf{e}^{(v)})_y \geq M^{-(d_i/2)+2} - nM^{-d_i/2} - M^{-2d_i} \geq M^{-(d_i/2)+1}.$$

Thus

$$\begin{aligned} D_{\mathbf{L}}(v, y) &= -\log_{nU}((\mathbf{L}^{-1} \mathbf{e}^{(v)})_y) + 2 = -\log_M((\mathbf{L}^{-1} \mathbf{e}^{(v)})_y) \log_{nU} M + 2 \\ &\leq \left(\frac{d_i}{2} - 1\right) \log_{nU} M + 2 \\ &\leq 0.6d_i - 1.2 + 2 \leq 0.65d_i, \end{aligned} \tag{11}$$

where the penultimate inequality follows from (8). Now by triangle inequality (Claim 2.2) for $y, y' \in W$,

$$D_{\mathbf{L}}(y, y') \leq D_{\mathbf{L}}(v, y) + D_{\mathbf{L}}(v, y') \leq 1.3d_i.$$

Therefore since the largest d_i is 4^ℓ , the result follows.

5. Let $y \in V$ and $z \in [n] \setminus W$. Similar to the proof of the previous property, we can show that for all $w \in S \setminus \{v\}$, $(\mathbf{L}^{-1} \mathbf{e}^{(w)})_y < M^{-d_i/2}$ (see (10)). Moreover since $y \in V$, we have

$$(\mathbf{L}^{-1} \mathbf{1}_S)_y + M^{-2d_i} \geq \mathbf{y}_y \geq M^{-d_i/4} - M^{-2d_i}.$$

Therefore since $(\mathbf{L}^{-1} \mathbf{1}_S)_y = (\mathbf{L}^{-1} \mathbf{e}^{(v)})_y + \sum_{w \in S \setminus \{v\}} (\mathbf{L}^{-1} \mathbf{e}^{(w)})_y$,

$$(\mathbf{L}^{-1} \mathbf{e}^{(v)})_y \geq M^{-d_i/4} - nM^{-d_i/2} - 2M^{-2d_i} \geq M^{-d_i/4+0.1}.$$

Thus

$$\begin{aligned} D_{\mathbf{L}}(v, y) &= -\log_{nU}((\mathbf{L}^{-1} \mathbf{e}^{(v)})_y) + 2 = -\log_M((\mathbf{L}^{-1} \mathbf{e}^{(v)})_y) \log_{nU} M + 2 \\ &\leq \left(\frac{d_i}{4} - 0.1\right) \log_{nU} M + 2 \\ &\leq 0.3d_i - 0.08 + 2 \leq 0.36d_i, \end{aligned} \tag{12}$$

where the penultimate inequality follows from (8). Now by triangle inequality (Claim 2.2) and (12),

$$D_{\mathbf{L}}(v, z) \leq D_{\mathbf{L}}(v, y) + D_{\mathbf{L}}(y, z) \leq 0.36d_i + D_{\mathbf{L}}(y, z).$$

Again, by an argument similar to the previous part for (10), we have $(\mathbf{L}^{-1}\mathbf{e}^{(v)})_z < M^{-d_i/2}$; otherwise, z would belong to C_v and thus to W , which is a contradiction. Therefore by (8),

$$D_{\mathbf{L}}(v, z) = -\log_{nU}((\mathbf{L}^{-1}\mathbf{e}^{(v)})_z) + 2 = \frac{-\log_M((\mathbf{L}^{-1}\mathbf{e}^{(v)})_z)}{\log_M(nU)} + 2 > \frac{d_i}{2} + 2,$$

which gives

$$D_{\mathbf{L}}(y, z) \geq 0.14d_i + 2.$$

Thus since the smallest d_i is $2^{\ell+1}$, the result follows.

We now bound the bit complexity and failure probability of the algorithm. The algorithm performs a total of $2^{O(\sqrt{\log n})} \log(n) \log(n/\delta)$ iterations. In each iteration, we solve a linear system that takes

$$\tilde{O}(m \log(M^{4^\ell}) \log(UM^{4^\ell} \delta^{-1} \ell^2 B)) = m 2^{O(\sqrt{\log n})} \log(U) \log(U \delta^{-1})$$

time. Moreover, in each iteration we compute connected components, which takes $O(m)$ time. Therefore, the total bit complexity of the algorithm is bounded by

$$m 2^{O(\sqrt{\log n})} \log(U) \log(\delta^{-1}) \log(U \delta^{-1}).$$

For the failure probability, note that since the total number of iterations is $\ell^2 B$, and each linear-system solve succeeds with probability at least $1 - \delta/(2\ell^2 B)$, by a union bound all linear systems are solved correctly with probability at least $1 - \delta/2$. Furthermore, as argued for the second property, we obtain a desirable set S for each vertex with probability at least $1 - \delta/(2n)$ over the B iterations of the inner loop for the corresponding i and j . Hence, with probability at least $1 - \delta/2$, we select a desirable S for all vertices throughout the algorithm, ensuring that all vertices are covered by the resulting low-diameter cover. By another union bound, the algorithm therefore succeeds with probability at least $1 - \delta$. □

3 The Almost-Linear-Time Algorithm

In this section, we present our algorithm for solving SDDM linear systems with entrywise approximation guarantees in almost-linear time and prove the main theorem of the paper ([Theorem 1.1](#)). To this end, we first recall the threshold decay (TD) framework from [\[GNY26\]](#), its requirements, and the guarantees it provides in [Section 3.1](#). We then derive bounds on the error of solutions to partial linear systems solved within the TD framework in [Section 3.2](#). Next, in [Section 3.3](#), we show that our low-diameter cover can be used to construct the partial solvers required in the iterations of the TD framework. Finally, we provide implementation details for achieving almost-linear running time and complete the proof in [Section 3.4](#).

3.1 The Threshold Decay Framework

Our almost-linear-time algorithm builds upon the TD framework proposed in [\[GNY26\]](#). In this framework, the linear system is solved by iteratively solving a sequence of partial linear systems, where each matrix is a principal submatrix of the original SDDM matrix, and the right-hand side vector is adaptively updated based on previously computed entrywise approximations to certain entries—see [THRESHOLDDecay](#).

ThresholdDecay: The algorithm framework for our solvers. Implementations are specified later. Only correctness is proved for the framework.

Input: \mathbf{L} : an $n \times n$ invertible RDDDL matrix with integer entries in $[-U, U]$,

\mathbf{b} : a nonnegative vector with integer entries in $[0, U]$,

ϵ : the target accuracy,

T : number of iterations (in this paper we always set it to n)

Assume that $U \geq 2, T \geq 10, \epsilon \in (0, 1)$

Output: vector $\tilde{\mathbf{x}}$ supported on some set $A \subseteq [n]$ such that $\tilde{\mathbf{x}}_A \approx_\epsilon (\mathbf{L}^{-1} \mathbf{b})_A$.

1. Initialize $S^{(0)} = [n]$, $\hat{\mathbf{b}}^{(0)} = \mathbf{b}$, and $\tilde{\mathbf{x}}^{(0)}$ to a zero dimensional vector. Let $\epsilon_L := \frac{\epsilon}{64T(nU)^2}$.
2. For $t = 0 \dots T$:
 - (a) Compute $\hat{\mathbf{x}}^{(t)}$ defined on $S^{(t)}$ such that $\|\hat{\mathbf{x}}^{(t)} - \mathbf{L}_{S^{(t)}, S^{(t)}}^{-1} \hat{\mathbf{b}}^{(t)}\|_2 \leq \epsilon_L \|\hat{\mathbf{b}}^{(t)}\|_2$.
 - (b) Let θ_t be the smallest power of two larger than $\frac{1}{4(nU)^2} \|\hat{\mathbf{b}}^{(t)}\|_1$.
 - (c) Let $F^{(t)} \subseteq S^{(t)}$ be the indices of all the entries in $\hat{\mathbf{x}}^{(t)}$ larger than or equal to θ_t , and let $S^{(t+1)} \leftarrow S^{(t)} \setminus F^{(t)}$.
 - (d) Let $\tilde{\mathbf{x}}^{(t+1)}$ be defined on $[n] \setminus S^{(t+1)}$ with $[\tilde{\mathbf{x}}_{[n] \setminus S^{(t)}}^{(t+1)}; \tilde{\mathbf{x}}_{F^{(t)}}^{(t+1)}] \leftarrow [\tilde{\mathbf{x}}^{(t)}; \hat{\mathbf{x}}_{F^{(t)}}^{(t)}]$.
 - (e) Create vector $\hat{\mathbf{b}}^{(t+1)}$ for the next iteration such that

$$\hat{\mathbf{b}}^{(t+1)} \approx_{\epsilon/(8T)} \mathbf{b}_{S^{(t+1)}} - \mathbf{L}_{S^{(t+1)}, [n] \setminus S^{(t+1)}} \tilde{\mathbf{x}}^{(t+1)} \quad (13)$$

3. Output set $A := [n] \setminus S^{(T)}$ and $\tilde{\mathbf{x}} := [\tilde{\mathbf{x}}_A; \tilde{\mathbf{x}}_{S^{(T)}}] = [\tilde{\mathbf{x}}^{(T)}; \mathbf{0}]$

Figure 2: The Threshold Decay framework.

Recall Threshold Decay. Let $\bar{\mathbf{x}} := \mathbf{L}^{-1} \mathbf{b}$ denote the solution vector. In Threshold Decay, $[n] = S^{(0)} \supset S^{(1)} \supset S^{(2)} \supset \dots$ are a chain of indices where $S^{(t)}$ is the set of indices for which an entrywise approximate has yet not been computed at iteration t . We invoke Laplacian solver on the set $S^{(t)}$ to solve $\hat{\mathbf{x}}^{(t)}$:

$$\|\hat{\mathbf{x}}^{(t)} - (\mathbf{L}_{S^{(t)}, S^{(t)}})^{-1} \hat{\mathbf{b}}^{(t)}\|_2 \leq \epsilon_L \|\hat{\mathbf{b}}^{(t)}\|_2 \quad (14)$$

The framework has the following correctness guarantees.

Lemma 3.1 (Lemma 3.7 of [GNY26]). *Given $\epsilon \in (0, 1)$, integer $T \geq 10$, $U \geq 2$, an $n \times n$ invertible RDDDL matrix \mathbf{L} with integer entries in $[-U, U]$, and a nonnegative vector \mathbf{b} with integer weights in $[0, U]$, let $\bar{\mathbf{x}} := \mathbf{L}^{-1} \mathbf{b}$ denote the exact solution vector, THRESHOLDDDECAY has the following guarantees for all $t \in [0, T]$:*

1. $\|\hat{\mathbf{b}}^{(t)}\|_1 \leq (nU)^{-t} \|\mathbf{b}\|_1$. Moreover, if $t > 0$, $\|\hat{\mathbf{b}}^{(t)}\|_1 \leq \frac{1}{(nU)} \|\hat{\mathbf{b}}^{(t-1)}\|_1$.
2. $\tilde{\mathbf{x}}^{(t)} \approx_{\epsilon t/(4T)} \bar{\mathbf{x}}_{[n] \setminus S^{(t)}}$. Note that we define $\tilde{\mathbf{x}}^{(0)}$ to be a zero dimensional vector.

3. For all $i \in S^{(t+1)}$, $\bar{x}_i < (nU)^{-2} \|\hat{\mathbf{b}}^{(t)}\|_1 \leq (nU)^{-(t+2)} \|\mathbf{b}\|_1$.

The first property implies that the threshold $\|\hat{\mathbf{b}}^{(t)}\|_1$ decreases over iterations. The second property ensures that the computed values are accurate entrywise approximations. The third property guarantees that large entries are solved and removed from the set $S^{(t)}$ in iteration t .

We invoke the following theorem from [GNY26] to establish the correctness of the framework. Note that in this paper we apply the framework only to SDDM matrices, although it generalizes to RDDDL matrices.

Theorem 3.1 (Correctness of the Threshold Decay framework, Theorem 3.1 of [GNY26]). *Given $\epsilon \in (0, 1)$, integer $T \geq 10$, $U \geq 2$, an $n \times n$ invertible RDDDL matrix \mathbf{L} with integer entries in $[-U, U]$, and a nonnegative vector \mathbf{b} with integer weights in $[0, U]$, THRESHOLDDECAY produces a vector $\tilde{\mathbf{x}}$ and a set $A \subseteq [n]$ such that $\tilde{\mathbf{x}}$ has support A , with the following properties hold:*

- For any $i \in [n]$ with $(\mathbf{L}^{-1}\mathbf{b})_i \geq (nU)^{-(T+1)} \|\mathbf{b}\|_1$, we have $i \in A$.
- Our output is an entrywise $\exp(\epsilon)$ -approximation on A , i.e., $\tilde{\mathbf{x}}_A \approx_\epsilon (\mathbf{L}^{-1}\mathbf{b})_A$.
- Specifically, when $T = n$, we have $\tilde{\mathbf{x}} \approx_\epsilon \mathbf{L}^{-1}\mathbf{b}$.

3.2 Errors Corresponding to Partial Linear Systems

In this subsection, we present results that bound the error for solutions to partial linear systems, i.e., when a subset of entries is set to zero. We will use these bounds to establish the correctness of our algorithm.

Lemma 3.2. *Let \mathbf{b} be a nonnegative n -dimensional vector, and let P denote the set of its positive entries. If $T \subseteq [n] \setminus P$ and $D_{\mathbf{L}}(u, v) \geq d > 0$ for all $u \in P$ and $v \in T$, then*

$$\|(\mathbf{L}^{-1})_{-T, -T} \mathbf{b}_{-T} - (\mathbf{L}_{-T, -T})^{-1} \mathbf{b}_{-T}\|_2 < n^{2.5} U (nU)^{2-d} \|\mathbf{b}\|_1.$$

Proof. Let

$$\mathbf{L} = \begin{pmatrix} \mathbf{L}_{-T, -T} & \mathbf{L}_{-T, T} \\ \mathbf{L}_{T, -T} & \mathbf{L}_{T, T} \end{pmatrix},$$

and let $\mathbf{S} := \mathbf{L}_{T, T} - \mathbf{L}_{T, -T}(\mathbf{L}_{-T, -T})^{-1} \mathbf{L}_{-T, T}$ be the Schur complement. Then, by Eq (1),

$$\begin{aligned} (\mathbf{L}^{-1})_{T, -T} &= -\mathbf{S}^{-1} \mathbf{L}_{T, -T} (\mathbf{L}_{-T, -T})^{-1}, \\ (\mathbf{L}^{-1})_{-T, -T} &= (\mathbf{L}_{-T, -T})^{-1} + (\mathbf{L}_{-T, -T})^{-1} \mathbf{L}_{-T, T} \mathbf{S}^{-1} \mathbf{L}_{T, -T} (\mathbf{L}_{-T, -T})^{-1}. \end{aligned}$$

We can bound

$$\begin{aligned} \|(\mathbf{L}^{-1})_{-T, -T} \mathbf{b}_{-T} - (\mathbf{L}_{-T, -T})^{-1} \mathbf{b}_{-T}\|_2 &= \|(\mathbf{L}_{-T, -T})^{-1} \mathbf{L}_{-T, T} \mathbf{S}^{-1} \mathbf{L}_{T, -T} (\mathbf{L}_{-T, -T})^{-1} \mathbf{b}_{-T}\|_2 \\ &= \|(\mathbf{L}_{-T, -T})^{-1} \mathbf{L}_{-T, T} (\mathbf{L}^{-1})_{T, -T} \mathbf{b}_{-T}\|_2 \\ &\leq \|(\mathbf{L}_{-T, -T})^{-1}\|_2 \|\mathbf{L}_{-T, T}\|_{\infty \rightarrow 2} \|(\mathbf{L}^{-1})_{T, -T} \mathbf{b}_{-T}\|_\infty \\ &\leq n^2 \cdot \sqrt{n} U \cdot (nU)^{2-d} \|\mathbf{b}_{-T}\|_1 \end{aligned}$$

where we used Lemma 1.6 to bound $\|(\mathbf{L}_{-T, -T})^{-1}\|_2$, the fact that the maximum entry of \mathbf{L} is U to bound $\|\mathbf{L}_{-T, T}\|_{\infty \rightarrow 2}$, and the property of the probability distance to bound $\|(\mathbf{L}^{-1})_{T, -T}\|_{\max}$ in the last inequality. \square

The previous lemma provides an upper bound on the error incurred when removing a set of vertices that are sufficiently far—in terms of probability distance—from the support of the right-hand side vector. We now extend this result to quantify the total error in approximating the full solution of the linear system by zeroing out the corresponding entries for these far vertices. Intuitively, the corollary shows that vertices far from the nonzero entries of \mathbf{b} contribute negligibly to the solution of $\mathbf{L}\mathbf{x} = \mathbf{b}$, and their removal introduces only a small error.

Corollary 3.3. *Let $\mathbf{L} \in \mathbb{Z}^{n \times n}$ be an invertible SDDM matrix, and let \mathbf{b} be any nonnegative n -dimensional vector. Let P denote the set of indices of its positive entries, and let $T \subseteq [n] \setminus P$. If $D_{\mathbf{L}}(u, v) \geq d > 0$ for all $u \in P, v \in T$, then*

$$\left\| [(\mathbf{L}^{-1}\mathbf{b})_{-T}; (\mathbf{L}^{-1}\mathbf{b})_T] - [(\mathbf{L}_{-T,-T})^{-1}\mathbf{b}_{-T}; \mathbf{0}] \right\|_2 \leq (nU)^{-d+5} \|\mathbf{b}\|_2.$$

Proof. Since $\mathbf{b}_T = \mathbf{0}$, we have $(\mathbf{L}^{-1})_{-T,-T}\mathbf{b}_{-T} = (\mathbf{L}^{-1}\mathbf{b})_{-T}$. Applying Lemma 3.2 to \mathbf{b}_{-T} , we obtain

$$\left\| (\mathbf{L}_{-T,-T})^{-1}\mathbf{b}_{-T} - (\mathbf{L}^{-1}\mathbf{b})_{-T} \right\|_2 < (nU)^{4.5-d} \|\mathbf{b}_{-T}\|_1 \leq (nU)^{4.5-d} \sqrt{n} \|\mathbf{b}_{-T}\|_2. \quad (15)$$

Next, observe that

$$(\mathbf{L}^{-1}\mathbf{b})_T = (\mathbf{L}^{-1})_{T,P}\mathbf{b}_P + (\mathbf{L}^{-1})_{T,-P}\mathbf{b}_{-P} = (\mathbf{L}^{-1})_{T,P}\mathbf{b}_P.$$

Thus we have

$$\left\| (\mathbf{L}^{-1})_{T,P}\mathbf{b}_P \right\|_2 \leq nU \cdot (nU)^{2-d} \|\mathbf{b}_P\|_1 = (nU)^{3-d} \|\mathbf{b}_P\|_1.$$

□

3.3 The Efficient Partial Solver

In our **THRESHOLDDECAY** framework, we introduce the boundary-expanded set for the use of our efficient partial solver in the next section.

The boundary-expanded set. Recall that each step of **THRESHOLDDECAY**, we extract the large entries in the solution vector of the subsystem

$$\mathbf{L}_{S^{(t)}, S^{(t)}} \mathbf{x} = \widehat{\mathbf{b}}^{(t)}.$$

Suppose that the right-hand-side vector $\widehat{\mathbf{b}}^{(t)}$ has support $I \subseteq [n]$. The combinatorial interpretation of the solution is the sum of random walks on the subgraph $S^{(t)}$ starting from the vertices in I . By leveraging the low-diameter cover, we enlarge the set I to its *boundary-expanded set*, ensuring that all but a small weighted fraction of the random walks remain inside this set. This allows us to solve the subsystem restricted to the boundary-expanded set instead of $S^{(t)}$, incurring only a small additive error, which is sufficient to accurately recover the large entries.

Definition 3.4 (Boundary-expanded set). *Given a low-diameter cover $\mathcal{C} = \{(V_i, W_i)\}_i$ and a subset $I \subseteq [n]$, we define the boundary-expanded set of I with respect to \mathcal{C} as*

$$\text{Exp}_{\mathcal{C}}(I) := \{u \in [n] : (\exists (V_i, W_i) \in \mathcal{C})(u \in V_i \wedge |W_i \cap I| > 0)\}.$$

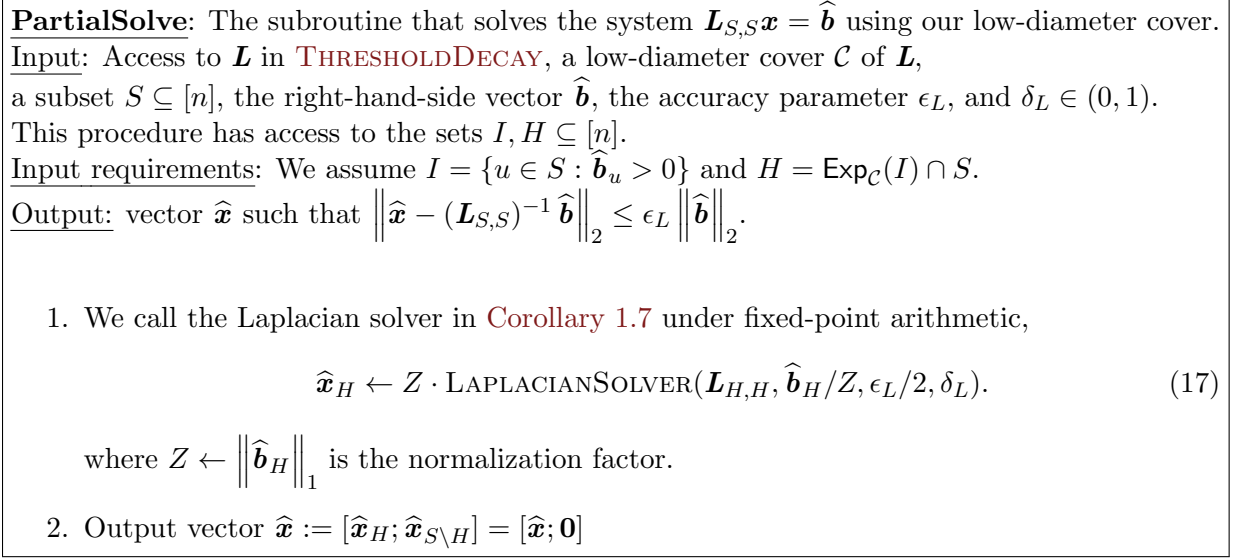


Figure 3: Solve the linear system restricted on a subset of vertices using the low-diameter cover

Note that $\text{Exp}_{\mathcal{C}}(I)$ is a superset of I that contains all the inner balls whose outer ball touches I . An equivalent definition of $\text{Exp}_{\mathcal{C}}(I)$ is

$$\text{Exp}_{\mathcal{C}}(I) := \bigcup_{(V_i, W_i) \in \mathcal{C}: |W_i \cap I| > 0} V_i. \quad (16)$$

We call standard Laplacian solvers on the boundary-expanded set $\text{Exp}_{\mathcal{C}}(I)$ so that the entries on I have good approximations.

The efficient partial solver. We present the main subroutine, **PARTIALSOLVE**, of our almost-linear-time algorithm that solves a partial system $\mathbf{L}_{S^{(t)}, S^{(t)}} \mathbf{x} = \hat{\mathbf{b}}^{(t)}$ in the t -th iteration of **THRESHOLDDECAY**.

We state our main lemma for the correctness of **PARTIALSOLVE**.

Lemma 3.5 (Correctness of **PARTIALSOLVE**). *For any $\mathbf{L}, \hat{\mathbf{b}}$, any I, H that satisfy the input requirements, any (r_1, r_2, s) -cover \mathcal{C} such that $r_2 \geq 5 + \log_{nU}(2/\epsilon_L)$, the returned vector $\hat{\mathbf{x}}$ of **PARTIALSOLVE** satisfies*

$$\|\hat{\mathbf{x}} - (\mathbf{L}_{S,S})^{-1}\hat{\mathbf{b}}\|_2 \leq \epsilon_L \|\hat{\mathbf{b}}\|_2$$

with probability $\geq 1 - \delta_L$.

Proof. Recall that $\hat{\mathbf{x}}$ is obtained from (17), which satisfies

$$\|\hat{\mathbf{x}}_H - (\mathbf{L}_{H,H})^{-1}\hat{\mathbf{b}}_H\|_2 \leq \frac{\epsilon_L}{2} \|\hat{\mathbf{b}}_H\|_2. \quad (18)$$

with probability at least $1 - \delta_L$ and $H = \text{Exp}_{\mathcal{C}}(I) \cap S$.

Fix any $u \in S \setminus H$, we have $u \notin \text{Exp}_C(I)$. By the definition of $\text{Exp}_C(I)$, it holds that

$$(\forall i)(u \notin V_i \vee |W_i \cap I| = 0).$$

By the definition of the clusters, there must exist an i_u such that $u \in V_{i_u}$. Thus, $|W_{i_u} \cap I| = 0$, implying that $D_L(u, v) > r_2$ for all $v \in I$, due to the last property [Definition 2.4](#). Note that the distance here is with respect to the Laplacian matrix of the entire graph. By [Lemma 2.3](#), we have $D_{L_{S,S}}(u, v) \geq D_L(u, v)$. Therefore, we have shown that

$$D_{L_{S,S}}(u, v) > r_2, \quad \forall u \in S \setminus H, v \in I. \quad (19)$$

We then apply [Corollary 3.3](#) to the base Laplacian matrix $L_{S,S}$ and the right-hand-side vector $\hat{\mathbf{b}}$ defined on the set S , with $S \setminus H$ being the target set, obtaining

$$\left\| \left((L_{S,S})^{-1} \hat{\mathbf{b}} \right)_H - (L_{H,H})^{-1} \hat{\mathbf{b}}_H \right\|_2 \leq (nU)^{-r_2+5} \left\| \hat{\mathbf{b}} \right\|_2 \quad (20)$$

$$\left\| \left((L_{S,S})^{-1} \hat{\mathbf{b}} \right)_{S \setminus H} \right\|_2 \leq (nU)^{-r_2+5} \left\| \hat{\mathbf{b}} \right\|_2 \quad (21)$$

Setting $r_2 \geq 5 + \log_{nU}(2/\epsilon_L)$, triangle inequality with [Eq \(18\)](#) and [Eq \(20\)](#) achieves

$$\left\| \hat{\mathbf{x}}_H - \left((L_{S,S})^{-1} \hat{\mathbf{b}} \right)_H \right\|_2 \leq \left(\frac{\epsilon_L}{2} + (nU)^{-r_2+5} \right) \left\| \hat{\mathbf{b}} \right\|_2 \leq \epsilon_L \left\| \hat{\mathbf{b}} \right\|_2. \quad (22)$$

Recall that $\hat{\mathbf{x}}_{S \setminus H} = \mathbf{0}$. Thus, [Eq \(21\)](#) achieves

$$\left\| \hat{\mathbf{x}}_{S \setminus H} - \left((L_{S,S})^{-1} \hat{\mathbf{b}} \right)_{S \setminus H} \right\|_2 \leq (nU)^{-r_2+5} \left\| \hat{\mathbf{b}} \right\|_2 < \epsilon_L \left\| \hat{\mathbf{b}} \right\|_2. \quad (23)$$

The correctness lemma then follows from combining [Eq \(22\)](#) and [Eq \(23\)](#). \square

3.4 The Almost-Linear-Time SDDM Solver

In this section, we present our almost-linear-time SDDM solver, [SDDMSOLVE](#), by plugging in [PARTIALSOLVE](#) into [THRESHOLDDECAY](#) with our pre-computed low-diameter cover from [LOWDIAM-CONSTRUCT](#). The correctness of our solver follows directly from the correctness of the subroutines ([Lemma 3.1](#), [Lemma 3.5](#), and [Theorem 2.1](#)). We focus on fast implementations and the resulting almost-linear running time bounds.

In iteration t , we call [PARTIALSOLVE](#) on the subsystem $L_{S^{(t)}, S^{(t)}}$ defined on the subset $S^{(t)}$. We use $I^{(t)}, H^{(t)}, \eta^{(t)}, \hat{\mathbf{x}}^{(t)}$ to denote $I, H, \eta, \hat{\mathbf{x}}$ of [PARTIALSOLVE](#) invoked in the t -th iteration. In the worst case, the sum of the sizes of $S^{(t)}$ over all iterations is $\Theta(n^2)$. Our partial solver works on the smaller subset $H^{(t)}$ instead of $S^{(t)}$. The following lemma shows that each vertex occurs in only a small number of sets $H^{(t)}$'s.

Lemma 3.6. *In [SDDMSOLVE](#), for any $u \in [n]$, it holds that $\sum_{t=0}^T \mathbf{1}[u \in H^{(t)}] \leq r_{in}$.*

Note that by [Theorem 2.1](#), we have that $r_{in} = 2^{O(\sqrt{\log n})}$ is small. We first prove the lemma in the next section. To get the almost-linear running time, we also discuss the efficient maintenance of the boundary-expanded set $\text{Exp}_C(S^{(t)})$ in [PARTIALSOLVE](#), as well as fast implementations of the updates to the vectors and sets in [THRESHOLDDECAY](#).

SDDMSolve: The entrywise approximate linear system solver for SDDM matrices based on **THRESHOLDDECAY** and **PARTIALSOLVE**, with fast implementations.

Input: $\mathbf{L}, \mathbf{b}, \epsilon, \kappa$ of **THRESHOLDDECAY**, where \mathbf{L} is an invertible SDDM matrix. $\delta \in (0, 1)$.

WLOG, assume \mathbf{L} is irreducible.

Output: vector $\tilde{\mathbf{x}}$ such that $\tilde{\mathbf{x}} \approx_{\epsilon} \mathbf{L}^{-1}\mathbf{b}$.

1. Produce a low-diameter cover $\mathcal{C} \leftarrow \text{LOWDIAMCONSTRUCT}(\mathbf{L}, \delta/2)$.

2. Run **THRESHOLDDECAY**, with vectors $\hat{\mathbf{b}}^{(t)}, \tilde{\mathbf{x}}^{(t)}, \hat{\mathbf{x}}^{(t)}$ represented by $O(\log(nU/\epsilon))$ -bit floating-point numbers.

Step 2a: We maintain $I^{(t)} := \{u \in S^{(t)} : \hat{\mathbf{b}}_u^{(t)} > 0\}$ and $H^{(t)} := \text{Exp}_{\mathcal{C}}(I^{(t)}) \cap S^{(t)}$ efficiently over the iterations. In iteration t :

(a) $\hat{\mathbf{x}}^{(t)} \leftarrow \text{PARTIALSOLVE}(\mathbf{L}, \mathcal{C}, S^{(t)}, \hat{\mathbf{b}}^{(t)}, \epsilon_L, \delta/(2n), H^{(t)}, I^{(t)})$.

Step 2b, 2c, 2d: Work only on the subset of indices $H^{(t)}$; ignore the entries on $S^{(t)}$.

Step 2e: Initialize n -dimensional vector $\hat{\mathbf{v}}^{(0)} = 0$. In iteration t :

(a) Compute $\hat{\mathbf{v}}^{(t+1)}$ defined on $S^{(t+1)}$, using in-place updates to $\hat{\mathbf{v}}^{(t)}$,

$$\hat{\mathbf{v}}^{(t+1)} \leftarrow \hat{\mathbf{v}}_{S^{(t+1)}}^{(t)} - \mathbf{L}_{S^{(t+1)}, F^{(t)}} \hat{\mathbf{x}}_{F^{(t)}}^{(t)}. \quad (24)$$

(b) Then, we have an implicit representation for $\hat{\mathbf{b}}^{(t+1)}$, that is,

$$\hat{\mathbf{b}}^{(t+1)} := \mathbf{b}_{S^{(t+1)}} + \hat{\mathbf{v}}^{(t+1)}. \quad (25)$$

Return the vector $\tilde{\mathbf{x}}$ produced by **THRESHOLDDECAY**.

Figure 4: The entrywise approximate linear system solver for SDDM matrices in almost-linear-time

3.4.1 Bounding the sum of sizes

In this section, we bound the sum of sizes of $H^{(t)}$ over all iterations t . We need the following lemma from [GNY26].

Lemma 3.7 (Lemma 3.14 of [GNY26]). *Let \mathbf{L} be an $n \times n$ invertible SDDM matrix with integer weights in $[-U, U]$, and \mathbf{b} be a nonnegative vector of dimension n . For any $i, j \in [n]$ with $\mathbf{L}_{ij} \neq 0$, we have*

$$(nU)^{-1} \cdot (\mathbf{L}^{-1}\mathbf{b})_j \leq (\mathbf{L}^{-1}\mathbf{b})_i \leq nU \cdot (\mathbf{L}^{-1}\mathbf{b})_j.$$

We also need the following generalized version of the lemma, using our probability distance (defined in Definition 2.1).

Lemma 3.8. *Let \mathbf{L} be an $n \times n$ invertible SDDM matrix with integer weights in $[-U, U]$, and \mathbf{b} be a nonnegative vector of dimension n . For any $i, j \in [n]$ with $D_{\mathbf{L}}(i, j) \leq d$, we have*

$$(nU)^{-d} \cdot (\mathbf{L}^{-1}\mathbf{b})_j \leq (\mathbf{L}^{-1}\mathbf{b})_i \leq (nU)^d \cdot (\mathbf{L}^{-1}\mathbf{b})_j$$

Proof. Fix any $k \in [n]$. By the triangle inequality from [Claim 2.2](#), we have $D_{\mathbf{L}}(i, k) \leq D_{\mathbf{L}}(i, j) + D_{\mathbf{L}}(j, k)$, and thus

$$(\mathbf{L}^{-1})_{ik} \geq (\mathbf{L}^{-1})_{ij}(\mathbf{L}^{-1})_{jk}(nU)^{-2} \geq (nU)^{-d+2} \cdot (\mathbf{L}^{-1})_{jk}(nU)^{-2} = (nU)^{-d} \cdot (\mathbf{L}^{-1})_{jk}. \quad (26)$$

Note that $(\mathbf{L}^{-1}\mathbf{b})_i$ and $(\mathbf{L}^{-1}\mathbf{b})_j$ can be written as linear combinations:

$$(\mathbf{L}^{-1}\mathbf{b})_i = \sum_{k=1}^n (\mathbf{L}^{-1})_{ik} \mathbf{b}_k, \quad (\mathbf{L}^{-1}\mathbf{b})_j = \sum_{k=1}^n (\mathbf{L}^{-1})_{jk} \mathbf{b}_k.$$

Since $\mathbf{b}_k \geq 0$ for all k , the inequality $(\mathbf{L}^{-1}\mathbf{b})_i \geq (nU)^{-d} \cdot (\mathbf{L}^{-1}\mathbf{b})_j$ follows by taking the same linear combination of both sides of (26). The other direction of the lemma follows by symmetry. \square

Proof of Lemma 3.6. We fix any $u \in [n]$. Suppose that in iteration t , u is contained in the set

$$H^{(t)} = \text{Exp}_{\mathcal{C}}(I^{(t)}) \cap S^{(t)}.$$

We thus have $u \in S^{(t)}$ and $u \in \text{Exp}_{\mathcal{C}}(I^{(t)})$. By the definition of boundary-expanded set, there exists $(V, W) \in \mathcal{C}$ such that $u \in V$ and $|W \cap I^{(t)}| > 0$. Let v be any vertex in $W \cap I^{(t)}$. By the definition of $I^{(t)}$, we have $\widehat{\mathbf{b}}_v^{(t)} > 0$. Let $\bar{\mathbf{x}} := \mathbf{L}^{-1}\mathbf{b}$. We now lower bound $\bar{\mathbf{x}}_v$.

By step 2e of [THRESHOLDDECAY](#), we have

$$\widehat{\mathbf{b}}^{(t)} \approx_{\epsilon/(8T)} \mathbf{b}_{S^{(t)}} - \mathbf{L}_{S^{(t)}, [n] \setminus S^{(t)}} \tilde{\mathbf{x}}^{(t)}.$$

Therefore, either $\mathbf{b}_v > 0$, or there exists $w \in [n] \setminus S^{(t)}$ such that $|\mathbf{L}_{v,w}| > 0$ and $\tilde{\mathbf{x}}_w^{(t)} > 0$. In the first case, $\bar{\mathbf{x}}_v \geq \mathbf{b}_v \geq 1$. In the second case, by [Lemma 3.7](#),

$$\bar{\mathbf{x}}_v \geq (nU)^{-1} \bar{\mathbf{x}}_w \approx_{\epsilon t/(4T)} \tilde{\mathbf{x}}_w \geq \theta_{t-1} = \frac{1}{4(nU)^2} \left\| \widehat{\mathbf{b}}^{(t-1)} \right\|_1.$$

where the entrywise approximation follows from the second property of [Lemma 3.1](#), and the last inequality follows from step 2c of [THRESHOLDDECAY](#). We thus obtain the following lower bound (where we define $\widehat{\mathbf{b}}^{(-1)}$ as the all-ones vector to handle the first case)

$$\bar{\mathbf{x}}_v \geq \frac{1}{8(nU)^2} \left\| \widehat{\mathbf{b}}^{(t-1)} \right\|_1.$$

Since u, v are in the same outer ball W in the low-diameter cover \mathcal{C} , by the fourth property of [Definition 2.4](#), we have $D_{\mathbf{L}}(u, v) \leq r_{\text{in}}$. Therefore, by [Lemma 3.8](#),

$$\bar{\mathbf{x}}_u \geq \bar{\mathbf{x}}_v \cdot (nU)^{-r_{\text{in}}} \geq (nU)^{-r_{\text{in}}-3} \left\| \widehat{\mathbf{b}}^{(t-1)} \right\|_1. \quad (27)$$

If u survives after $r_{\text{in}} + 1$ iterations, then $u \in S^{(t+r_{\text{in}}+1)}$ and by [Lemma 3.1](#),

$$\bar{\mathbf{x}}_u < (nU)^{-2} \left\| \widehat{\mathbf{b}}^{(t+r_{\text{in}})} \right\|_1 < (nU)^{-3-r_{\text{in}}} \left\| \widehat{\mathbf{b}}^{(t-1)} \right\|_1$$

where we repeatedly applied property 1 of [Lemma 3.1](#) for the second inequality. This contradicts with (27). Therefore, for any $u \in [n]$, it must survive at most r_{in} iterations, and thus

$$\sum_{t=0}^T \mathbf{1}[u \in H^{(t)}] \leq r_{\text{in}}.$$

\square

3.4.2 Maintaining the vector $\hat{\mathbf{b}}^{(t)}$

The implementation of step 2e (stated in **SDDMSOLVE**) is the same implementation as in **PREDICTIVESDDMSOLVER** of [GN26]. This maintains the vector $\hat{\mathbf{b}}^{(t)}$ efficiently over the iterations by storing an extra vector $\hat{\mathbf{v}}^{(t)}$.

Note that $F^{(t)} = S^{(t)} \setminus S^{(t+1)}$ is the set of removed vertices in iteration t . We update $\hat{\mathbf{b}}^{(t)}$ on the edges from $F^{(t)}$ to $S^{(t+1)}$, i.e., the nonzero entries of $\mathbf{L}_{S^{(t+1)}, F^{(t)}}$. Each edge only contributes once over all iterations, so the total number of arithmetic operation is $O(n + m)$. The detailed proof appears in the proof of Lemma 3.13 of [GN26].

Claim 3.9 (Lemma 3.13 of [GN26]). *The implementation of Step 2e in **SDDMSOLVE** runs in a total of $O((n + m) \log(nU/\epsilon))$ bit operations. There are only $O(n + m)$ total entry updates to the vector $\hat{\mathbf{v}}^{(t)}$ and $\hat{\mathbf{b}}^{(t)}$.*

Furthermore, we need to track the norms of the vector $\hat{\mathbf{b}}^{(t)}$, including $\|\hat{\mathbf{b}}^{(t)}\|_1$ and $\|\hat{\mathbf{b}}^{(t)}\|_2$, for use in other steps of the algorithm. This can be implemented efficiently as well.

Claim 3.10. *In **SDDMSOLVE**, for any constant $c > 0$, we can maintain $\|\hat{\mathbf{b}}^{(t)}\|_1$ and $\|\hat{\mathbf{b}}^{(t)}\|_2$ up to multiplicative error of $(\epsilon/(nU))^c$ with total $O((n + m) \log(nU/\epsilon) \log n)$ bit operations.*

Proof. By **Claim 3.9**, there are only $O(n + m)$ total updates to the vector $\hat{\mathbf{v}}^{(t)}$. Observe that it suffices to compute the sum of p -th powers over the index set $S^{(t)}$, since

$$\|\hat{\mathbf{b}}^{(t)}\|_p^p = \sum_{i \in S^{(t)}} (\mathbf{b}_i + \mathbf{v}_i^{(t)})^p.$$

We can maintain this sum while updating $S^{(t)}$ and $\mathbf{v}^{(t)}$. There are n vertex removals from $S^{(t)}$ and $O(n + m)$ entry updates to $\mathbf{v}^{(t)}$. However, directly subtracting the sum when removing vertices from $S^{(t)}$ can cause numerical issues due to cancelation. To avoid this, we use dynamic data structures that maintain partial sums. A segment tree supports $O(\log n)$ update and query time per operation. We conclude the claim by using an $O(\log(nU/\epsilon))$ -bit floating-point number for each node of the segment tree. \square

3.4.3 Maintaining $I^{(t)}$ and $H^{(t)}$

In this section, we discuss how the set $I^{(t)}$ and $H^{(t)}$ are maintained efficiently. Recall the input requirements for **PARTIALSOLVE**, $I^{(t)} = \{u \in S^{(t)} : \hat{\mathbf{b}}_u^{(t)} > 0\}$, and $H^{(t)} = \text{Exp}_C(I^{(t)}) \cap S^{(t)}$.

Maintaining $I^{(t)}$. Note that there is no numerical issue here. Since $\hat{\mathbf{b}}^{(t)}$ is constructed with multiplicative approximation guarantees (and represented using floating points), it is easy to distinguish a nonzero value from a zero, as no cancelation occurs throughout our algorithm.

By **Claim 3.9**, there are only $O(n + m)$ total entry updates to $\hat{\mathbf{b}}^{(t)}$. Consequently, maintaining $I^{(t)}$ alongside $\hat{\mathbf{b}}^{(t)}$ is straightforward in total time $O(n + m)$.

$I^{(t)}$ has a more desirable property, as shown in the next claim.

Claim 3.11. *For any iteration $t < T$ and any vertex $u \in S^{(t+1)}$, if $\hat{\mathbf{b}}_u^{(t)} > 0$, then it follows that $\hat{\mathbf{b}}_u^{(t+1)} > 0$.*

Proof. It follows directly from the construction of $\widehat{\mathbf{b}}^{(t)}$, (24) and (25), and the fact that $\mathbf{L}_{S^{(t+1)}, F^{(t)}}$ only has non-positive entries while $\widehat{\mathbf{x}}_{F^{(t)}}^{(t)}$ has positive entries. \square

The claim shows that each vertex u is inserted to $I^{(t)}$ only once. It is removed from $I^{(t)}$ only when u is no longer in the set $S^{(t)}$.

Maintaining $H^{(t)}$. By the equivalent definition of boundary-expanded set (16),

$$\text{Exp}_{\mathcal{C}}(I) = \bigcup_{(V_i, W_i) \in \mathcal{C}: |W_i \cap I| > 0} V_i = \bigcup_{u \in I} \left(\bigcup_{(V_i, W_i) \in \mathcal{C}: u \in W_i} V_i \right) = \bigcup_{u \in I} \text{Cores}_{\mathcal{C}}(u).$$

where $\text{Cores}_{\mathcal{C}}(u) := \bigcup_{(V_i, W_i) \in \mathcal{C}: u \in W_i} V_i$ is the union of cores of the balls containing u . We maintain the set $H^{(t)} = \text{Exp}_{\mathcal{C}}(I^{(t)}) \cap S^{(t)}$ as follows:

- Create a counter cnt_i for each cluster $i \in |\mathcal{C}|$, with initial value 0. Maintain the indices of positive counters $C_{>0} := \{i : \text{cnt}_i > 0\}$.
- Insert/Delete u in $I^{(t)}$: For each i such that $(V_i, W_i) \in \mathcal{C}$ and $u \in W_i$, increment/decrement cnt_i by 1 and update $C_{>0}$.
- Build $H^{(t)}$ in iteration t : Build the union set by removing duplicates

$$H^{(t)} \leftarrow \bigcup_{i \in C_{>0}} V_i.$$

By the third property of Definition 2.4 and that \mathcal{C} is an $(r_{\text{in}}, r_{\text{out}}, \alpha)$ -cover, each vertex u is contained in at most α outer balls in \mathcal{C} .

For the second step, since each vertex is inserted to $I^{(t)}$ only once, the total number of updates to the counters is bounded by $2n\alpha = n2^{O(\sqrt{\log n})}$.

For the third step, $H^{(t)}$ is constructed in time complexity $O(|H^{(t)}| \cdot \alpha)$, since each vertex u in $H^{(t)}$ has at most $\alpha - 1$ duplicates by the third property of \mathcal{C} .

The total time complexity for maintaining $H^{(t)}$ throughout the iterations is

$$\begin{aligned} O(n2^{O(\sqrt{\log n})}) + \sum_{t=0}^T O(|H^{(t)}| \cdot \alpha) &= O(n2^{O(\sqrt{\log n})}) + \alpha \cdot O\left(\sum_{t=0}^T |H^{(t)}|\right) \\ &= O(n2^{O(\sqrt{\log n})}) + \alpha \cdot n2^{O(\sqrt{\log n})} = n2^{O(\sqrt{\log n})}. \end{aligned}$$

where the second equality follows from Lemma 3.6. We summarize our discussions into the following claim.

Claim 3.12. *In SDDMSOLVE, $H^{(t)}$ can be maintained in time complexity $n2^{O(\sqrt{\log n})}$ throughout the iterations.*

3.4.4 Wrap-up

In this section, we put everything together to prove the main theorem of the paper.

Theorem 1.1. *There exists a randomized algorithm **SDDMSOLVE** such that, for any $\delta \in (0, 1)$ and any $\epsilon > (nU)^{-2^{\sqrt{\log n}}}$ (i.e., not exponentially small), any invertible SDDM matrix $\mathbf{L} \in \mathbb{Z}^{n \times n}$ with m nonzero integer entries in $[-U, U]$ and a nonnegative vector $\mathbf{b} \in \mathbb{Z}^n$ with integer entries in $[0, U]$, with probability at least $1 - \delta$, computes the entrywise approximate solution*

$$\tilde{\mathbf{x}} \approx_{\epsilon} \mathbf{L}^{-1} \mathbf{b}$$

whose entries are represented by $O(\log(nU/\epsilon))$ -bit floating points, using²

$$\tilde{O}(m2^{O(\sqrt{\log n})} \log(U) \log^2(U\epsilon^{-1}\delta^{-1}))$$

bit operations.

Proof. We start by recalling the parameters.

Parameters. In Step 1 of **SDDMSOLVE**, it follows from **Theorem 2.1** that \mathcal{C} is an $(r_{\text{in}}, r_{\text{out}}, \alpha)$ -cover with parameters as below, for $\ell := \lceil \sqrt{\log n} \rceil + 3$:

- Diameter: $r_{\text{in}} = 2^{2\ell+1} = 2^{O(\sqrt{\log n})}$.
- Outer gap: $r_{\text{out}} = 2^{\ell-2} = 2^{\Omega(\sqrt{\log n})}$.
- Size parameter: $\alpha = 6\ell^2 \cdot 16^{\ell} \cdot \lceil \log(n/\delta) \rceil = 2^{O(\sqrt{\log n})}$.

This cover is produced in $\tilde{O}(mn^{o(1)} \log(U\delta^{-1}))$ bit operations.

In Step 2 of **SDDMSOLVE**, we run **THRESHOLDDECAY** for $T = n$ iterations.

Correctness. We apply **Lemma 3.5** to show that **PARTIALSOLVE** satisfies the guarantees of Step 2a in **THRESHOLDDECAY**. It suffices to verify

$$5 + \log_{nU}(2/\epsilon_L) = 5 + \log_{nU}(128T(nU)^2/\epsilon) \leq 9 + \log_{nU}(1/\epsilon) \leq 9 + 2^{\sqrt{\log n}} < r_{\text{in}}$$

Since $r_{\text{in}} = 2^{2\ell+1}$, the inequality holds provided that $\epsilon \geq 1/\text{poly}(n)$. It then follows from the correctness of **THRESHOLDDECAY** (**Theorem 3.1**) that $\tilde{\mathbf{x}} \approx_{\epsilon} \mathbf{L}^{-1} \mathbf{b}$, concluding the correctness of our almost-linear-time solver.

About the probability correctness:

- Failure probability by using **LOWDIAMCONSTRUCT** is $\leq \delta/2$. We call this procedure once.
- Failure probability by using **PARTIALSOLVE** is $\leq \delta/(2n)$. We call this procedure once each iteration, so overall n times.

Union bound gives the failure probability of **SDDMSOLVE** to be $\leq \delta$, thus the algorithm succeeds with probability at least $1 - \delta$.

² \tilde{O} suppresses the $\log \log(U\epsilon^{-1}\delta^{-1})$ factors.

Bit complexity. By Lemma 3.6, for any $u \in [n]$, we have

$$\sum_{t=0}^T \mathbf{1}[u \in H^{(t)}] \leq r_{\text{in}} = 2^{O(\sqrt{\log n})},$$

and consequently, the total size of $H^{(t)}$ over all iterations is almost-linear in n , i.e.,

$$\sum_{t=0}^T |H^{(t)}| \leq n 2^{O(\sqrt{\log n})}. \quad (28)$$

By Claim 3.12, the set $H^{(t)}$ is maintained in $n 2^{O(\sqrt{\log n})}$ operations throughout the iterations.

Step 2a is the dominating step. We bound the total time of the Laplacian solvers we invoked. By Corollary 1.7, the Laplacian solver in iteration t solves the subsystem $\mathbf{L}_{H^{(t)}, H^{(t)}}$, and requires $\tilde{O}(\text{nnz}(\mathbf{L}_{H^{(t)}, H^{(t)}}) \log^2(U\epsilon^{-1}\delta^{-1}))$ bit operations. The sum of sparsity can be bounded by

$$\begin{aligned} \sum_{t=0}^T \text{nnz}(\mathbf{L}_{H^{(t)}, H^{(t)}}) &\leq \sum_{t=0}^T \sum_{u \in [n]} \mathbf{1}[u \in H^{(t)}] \cdot \deg(u) \\ &\leq \sum_{u \in [n]} \left(\sum_{t=0}^T \mathbf{1}[u \in H^{(t)}] \right) \cdot \deg(u) \leq m 2^{O(\sqrt{\log n})}. \end{aligned}$$

For **Steps 2b, 2c, 2d**, every vector is supported only on the set $H^{(t)}$. A straightforward implementation that iterates over $H^{(t)}$ gives a total of $\tilde{O}(2^{O(\sqrt{\log n})} \cdot n)$ bit operations. For example, in **Step 2c**, we need to compute the set $F^{(t)} \subseteq S^{(t)}$ such that

$$F^{(t)} = \{i \in S^{(t)} : \hat{x}_i^{(t)} \geq \theta_t\}.$$

We cannot afford enumerating over $S^{(t)}$; instead, we enumerate over $H^{(t)}$ and skip the indices in $S^{(t)} \setminus H^{(t)}$ since we know they are all zeros. By Eq (28), this step take a total of almost-linear time.

Lastly, by Claim 3.9 and Claim 3.10, **Step 2e** is implemented in $O((n + m) \log(nU/\epsilon))$ time. Putting everything together, we conclude that **SDDMSOLVE** runs in $\tilde{O}(m 2^{O(\sqrt{\log n})} \log^2(U\epsilon^{-1}\delta^{-1}))$ bit operations. The final running time follows by considering the running time of computing the low-diameter cover as well. \square

4 Conclusion

We presented an almost-linear-time algorithm for solving SDDM linear systems with entrywise approximation guarantees. A natural open question is whether one can achieve truly near-linear running time, that is, $O(m \text{poly} \log n)$. In this work, we focused on inputs given in fixed-point representation. However, as noted in [GN26], the case of floating-point inputs presents additional challenges due to possible reductions from shortest-path problems. Investigating algorithms for SDDM systems with floating-point inputs therefore remains an interesting direction for future research.

References

- [BGK⁺11] Guy E Blelloch, Anupam Gupta, Ioannis Koutis, Gary L Miller, Richard Peng, and Kanat Tangwongsan. Near linear-work parallel SDD solvers, low-diameter decomposition, and low-stretch subgraphs. In *Proceedings of the twenty-third annual ACM symposium on Parallelism in algorithms and architectures*, pages 13–22, 2011. 1, 8

- [CKM⁺14] Michael B Cohen, Rasmus Kyng, Gary L Miller, Jakub W Pachocki, Richard Peng, Anup B Rao, and Shen Chen Xu. Solving SDD linear systems in nearly $m \log^{1/2} n$ time. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 343–352, 2014. 1
- [Coh98] Edith Cohen. Fast algorithms for constructing t -spanners and paths with stretch t . *SIAM Journal on Computing*, 28(1):210–236, 1998. 8
- [GKS23] Yuan Gao, Rasmus Kyng, and Daniel A. Spielman. Robust and practical solution of Laplacian equations by approximate elimination. *CoRR*, abs/2303.00709, 2023. 8
- [GNY26] Mehrdad Ghadiri, Hoai-An Nguyen, and Junzhao Yang. Entrywise approximation for matrix inversion and linear systems. In *SODA*, 2026. 1, 3, 4, 5, 6, 8, 17, 18, 19, 23, 25, 28
- [GTH85] Winfried K Grassmann, Michael I Taksar, and Daniel P Heyman. Regenerative analysis and steady state distributions for Markov chains. *Operations Research*, 33(5):1107–1116, 1985. 8
- [Hey87] Daniel P Heyman. Further comparisons of direct methods for computing stationary distributions of Markov chains. *SIAM Journal on Algebraic Discrete Methods*, 8(2):226–232, 1987. 8
- [Hig90] Nicholas J Higham. Exploiting fast matrix multiplication within the level 3 BLAS. *ACM Transactions on Mathematical Software (TOMS)*, 16(4):352–368, 1990. 8
- [HP84] WJ Harrod and RJ Plemmons. Comparison of some direct methods for computing stationary distributions of Markov chains. *SIAM journal on scientific and statistical computing*, 5(2):453–469, 1984. 8
- [JS21] Arun Jambulapati and Aaron Sidford. Ultrasparse ultrasparsifiers and faster Laplacian system solvers. *ACM Transactions on Algorithms*, 2021. 1
- [KLP⁺16] Rasmus Kyng, Yin Tat Lee, Richard Peng, Sushant Sachdeva, and Daniel A Spielman. Sparsified Cholesky and multigrid solvers for connection Laplacians. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 842–850, 2016. 1
- [KMP11] Ioannis Koutis, Gary L Miller, and Richard Peng. A nearly- $m \log n$ time solver for SDD linear systems. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 590–598. IEEE, 2011. 1
- [KMP14] Ioannis Koutis, Gary L Miller, and Richard Peng. Approaching optimality for solving SDD linear systems. *SIAM Journal on Computing*, 43(1):337–354, 2014. 1
- [KMST10] Alexandra Kolla, Yury Makarychev, Amin Saberi, and Shang-Hua Teng. Subgraph sparsification and nearly optimal ultrasparsifiers. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 57–66, 2010. 1
- [KOSZ13] Jonathan A Kelner, Lorenzo Orecchia, Aaron Sidford, and Zeyuan Allen Zhu. A simple, combinatorial algorithm for solving SDD systems in nearly-linear time. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 911–920, 2013. 1

- [KS16] Rasmus Kyng and Sushant Sachdeva. Approximate Gaussian elimination for Laplacians - fast, sparse, and simple. In *FOCS*, pages 573–582. IEEE Computer Society, 2016. 1
- [LS13] Yin Tat Lee and Aaron Sidford. Efficient accelerated coordinate descent methods and faster algorithms for solving linear systems. In *2013 IEEE 54th annual symposium on foundations of computer science*, pages 147–156. IEEE, 2013. 1
- [O’C93] Colm Art O’Cinneide. Entrywise perturbation theory and error analysis for Markov chains. *Numerische Mathematik*, 65(1):109–120, 1993. 8
- [O’C96] Colm Art O’Cinneide. Relative-error bounds for the LU decomposition via the GTH algorithm. *Numerische Mathematik*, 73:507–519, 1996. 8
- [PS14] Richard Peng and Daniel A Spielman. An efficient parallel solver for SDD linear systems. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 333–342, 2014. 1
- [ST14] D. Spielman and S. Teng. Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *SIAM Journal on Matrix Analysis and Applications*, 35(3):835–885, 2014. Available at: <http://arxiv.org/abs/cs/0607105>. 1, 4, 6
- [Str69] Volker Strassen. Gaussian elimination is not optimal. *Numerische mathematik*, 13(4):354–356, 1969. 8