
Iterative Methods via Locally Evolving Set Process

Baojian Zhou^{1,2} * Yifan Sun³ Reza Babanezhad Harikandeh⁴ Xingzhi Guo³

Deqing Yang^{1,2}

Yanghua Xiao²

¹ the School of Data Science, Fudan University,

² Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University

³ Department of Computer Science, Stony Brook University, ⁴ Samsung SAIT AI Lab.

Abstract

Given the damping factor α and precision tolerance ϵ , Andersen et al. [2] introduced Approximate Personalized PageRank (APPR), the *de facto local method* for approximating the PPR vector, with runtime bounded by $\Theta(1/(\alpha\epsilon))$ independent of the graph size. Recently, Fountoulakis & Yang [12] asked whether faster local algorithms could be developed using $\tilde{O}(1/(\sqrt{\alpha}\epsilon))$ operations. By noticing that APPR is a local variant of Gauss-Seidel, this paper explores the question of *whether standard iterative solvers can be effectively localized*. We propose to use the *locally evolving set process*, a novel framework to characterize the algorithm locality, and demonstrate that many standard solvers can be effectively localized. Let $\overline{\text{vol}}(\mathcal{S}_t)$ and $\overline{\gamma}_t$ be the running average of volume and the residual ratio of active nodes \mathcal{S}_t during the process. We show $\overline{\text{vol}}(\mathcal{S}_t)/\overline{\gamma}_t \leq 1/\epsilon$ and prove APPR admits a new runtime bound $\tilde{O}(\overline{\text{vol}}(\mathcal{S}_t)/(\alpha\overline{\gamma}_t))$ mirroring the actual performance. Furthermore, when the geometric mean of residual reduction is $\Theta(\sqrt{\alpha})$, then there exists $c \in (0, 2)$ such that the local Chebyshev method has runtime $\tilde{O}(\overline{\text{vol}}(\mathcal{S}_t)/(\sqrt{\alpha}(2-c)))$ without the monotonicity assumption. Numerical results confirm the efficiency of this novel framework and show up to a hundredfold speedup over corresponding standard solvers on real-world graphs.

1 Introduction

Personalized PageRank (PPR) vectors are key tools for graph problems such as clustering [2, 3, 30, 36, 54, 57], diffusion [10, 14, 15, 29], random walks [25, 32, 44], neural net training [7, 27, 20, 21], and many others [17, 48]. The Approximate PPR (APPR) [2] and its many variants [6, 9, 13, 37] efficiently approximate PPR vectors by exploring the neighbors of a specific node at each time, only requiring access to a tiny part of the graph – hence the number of operations needed is independent of graph size. These local solvers are well-suited for large-scale graphs in modern graph data analysis. Specifically, let \mathbf{A} and \mathbf{D} be the adjacency and degree matrices of a graph \mathcal{G} , respectively. Given a source node s and the damping factor $\alpha \in (0, 1)$, this paper studies local solvers for the linear system

$$(\mathbf{I} - (1 - \alpha)(\mathbf{I} + \mathbf{A}\mathbf{D}^{-1})/2) \boldsymbol{\pi} = \alpha \mathbf{e}_s, \quad (1)$$

where \mathbf{e}_s is the standard basis of s and $\boldsymbol{\pi}$ is the PPR vector [2, 12, 37]. Given the error tolerance ϵ , a local solver needs to find $\hat{\boldsymbol{\pi}}$ such that $\|\mathbf{D}^{-1}(\hat{\boldsymbol{\pi}} - \boldsymbol{\pi})\|_\infty \leq \epsilon$ without accessing the entire graph \mathcal{G} .²

*Corresponding to: Baojian Zhou, bjzhou@fudan.edu.cn

²Local methods for solving Equ. (1) can be naturally extended to other linear systems defined on \mathcal{G} .

Andersen et al. [2] proposed the local APPR algorithm, which pushes large residuals to neighboring nodes until all residuals are small. Its runtime is upper bounded by $\Theta(1/(\alpha\epsilon))$ independent of graph size. Based on a variational characterization of Equ. (1), Fountoulakis et al. [13] reformulated the problem as optimizing a quadratic objective plus ℓ_1 -regularization and later asked [12] whether there exists a local solver with runtime $\tilde{O}(1/(\sqrt{\alpha}\epsilon))$. This corresponds to an accelerated rate since α is the strongly convex parameter. Recently, Martínez-Rubio et al. [37] provided a method based on a nested subspace pursuit strategy, and the corresponding iteration complexity is bounded by $\tilde{O}(|\mathcal{S}^*|/\sqrt{\alpha})$ where \mathcal{S}^* is the support of the optimal solution. This bound deteriorates to $\tilde{O}(n/\sqrt{\alpha})$ when the solution is dense, with n representing the number of nodes in \mathcal{G} , which could be less favorable than that of standard solvers under similar conditions. Moreover, the nested computational structure provides a constant factor overhead, which could be significant in practice.

The bound analysis of the above local methods critically depends on the monotonicity properties of the designed algorithms. These requirements may hinder the development of simpler and faster local linear solvers that lack such monotonicity properties. Specifically, the runtime analysis of APPR relies on the non-negativity and decreasing monotonicity of residuals. Conversely, the runtime bounds developed in Fountoulakis et al. [13] and Martínez-Rubio et al. [37] depend on the monotonicity of variable updates, ensuring that the sparsity of intermediate variables increases monotonically.

Our contributions. Based on a refined analysis of APPR, our starting point is to demonstrate that APPR is a local variant of Gauss-Seidel Successive Overrelaxation (GS-SOR) that can be treated as an evolving set process.³ This insight leads us to explore whether standard solvers can be effectively localized to solve Equ. (1). To develop faster local methods with improved local bounds, we propose a novel *locally evolving set process* framework inspired by its stochastic counterpart [38]. This framework enables the development of faster local methods and circumvents the monotonicity requirement barrier in runtime complexity analysis in the existing literature. For example, our analysis of the local Chebyshev method does not depend on the monotonicity of residual or the active node sets processed. Specifically,

- As a core tool, we propose an algorithm framework based on the *locally evolving set process*. We show that APPR is a local variant of GS-SOR using this process. This framework is powerful enough to facilitate the development of new local solvers. Specifically, standard gradient descent (GD) can be effectively localized for solving this problem and admits $\Theta(1/(\alpha\epsilon))$ runtime bound.
- This local evolving set process provides a novel way to characterize the algorithm locality; hence, new runtime bounds can be derived. Let $\overline{\text{vol}}(\mathcal{S}_t)$ and $\overline{\gamma}_t$ be the running average of volume and the residual ratio of active nodes \mathcal{S}_t during the process; we prove the ratio $\overline{\text{vol}}(\mathcal{S}_t)/\overline{\gamma}_t$ serving as a lower bound of $1/\epsilon$. We further show both APPR and local GD have $\tilde{O}(\overline{\text{vol}}(\mathcal{S}_t)/(\alpha\overline{\gamma}_t))$ runtime bound mirroring the actual performance of these two methods.
- Using our framework, we show there exists $c \in (0, 2)$ such that both the localized Chebyshev and Heavy-Ball methods admit runtime bound $\tilde{O}(\overline{\text{vol}}(\mathcal{S}_t)/(\sqrt{\alpha}(2-c)))$ with the assumption that the geometric mean of active ratio factors is $\Theta(\sqrt{\alpha})$. Importantly, our analysis does not require any monotonicity property. The technical novelty is that we effectively characterize residuals of these two methods by using second-order difference equations with parameterized coefficients.
- We demonstrate, over 17 large graphs, that these localized methods can significantly accelerate their standard counterparts by a large margin. Furthermore, our proposed LOCSOR, LOCCH, and LOCHB are significantly faster than APPR and ℓ_1 -based solvers on two huge-scale graphs.

Paper structure. We begin by clarifying notations and reviewing APPR in Sec. 2. Sec. 3 introduces the locally evolving set process. Sec. 4 presents localized Chebyshev and Heavy-Ball methods along with our novel techniques. We discuss open questions in Sec. 5. Experiments and conclusions are covered in Sec. 6 and 7, respectively. *Detailed related works and all missing proofs are included in the Appendix. Our code is available at <https://github.com/baojian/LocalCH>.*

2 Notations and Preliminaries

Notations. We consider an undirected simple graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{1, 2, \dots, n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ with $|\mathcal{E}| = m$ are the node and edge sets, respectively. The set of neighbors of v is denoted as

³The local variant of GS-SOR is defined in Appendix B.1

$\mathcal{N}(v) \subseteq \mathcal{V}$. The adjacency matrix \mathbf{A} of \mathcal{G} assigns unit weight $a_{u,v} = 1$ if $(u, v) \in \mathcal{E}$ and 0 otherwise. The v -th entry of the degree matrix \mathbf{D} is $d_v = |\mathcal{N}(v)|$. Given $\mathcal{S} \subseteq \mathcal{V}$, we define the volume of \mathcal{S} as $\text{vol}(\mathcal{S}) \triangleq \sum_{v \in \mathcal{S}} d_v$. The support of $\mathbf{x} \in \mathbb{R}^n$ is the set of nonzero indices $\text{supp}(\mathbf{x}) \triangleq \{v : x_v \neq 0, v \in \mathcal{V}\}$. The eigendecomposition of $\mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2} = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^\top$ where each column of \mathbf{V} is an eigenvector and $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ with $1 = \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \geq -1$.

2.1 Revisiting Anderson’s APPR and its local runtime bound

We use (\mathbf{p}, \mathbf{z}) for solving Equ. (1) while use (\mathbf{x}, \mathbf{r}) for solving Equ. (3) or Equ. (4). With the initial setting $\mathbf{p} \leftarrow \mathbf{0}, \mathbf{z} \leftarrow \mathbf{e}_s$, APPR obtains a local estimate of $\boldsymbol{\pi}$ denoted as \mathbf{p} by using a sequence of PUSH operations defined as

$$\text{APPR}(\alpha, \epsilon, s, \mathcal{G}) : \quad \text{Repeat } (\mathbf{p}, \mathbf{z}) \leftarrow \text{PUSH}(u, \alpha, \mathbf{p}, \mathbf{z}) \text{ Until } \forall v, z_v < \epsilon d_v; \quad \text{Return } \mathbf{p}. \quad (2)$$

Algo. 1 PUSH($u, \alpha, \mathbf{p}, \mathbf{z}$)

- 1: $\nu = z_u$
- 2: $p_u \leftarrow p_u + \alpha \cdot \nu$
- 3: $z_u \leftarrow (1 - \alpha) \cdot \nu / 2$
- 4: **for** $v \in \mathcal{N}(u)$ **do**
- 5: $z_v \leftarrow z_v + \frac{(1-\alpha)\nu}{2d_u}$
- 6: **Return** (\mathbf{p}, \mathbf{z})

At each repeat step $(\mathbf{p}, \mathbf{z}) \leftarrow \text{PUSH}(u, \alpha, \mathbf{p}, \mathbf{z})$, it synchronously updates both \mathbf{p} and residual \mathbf{z} whenever there exists an active node $u \in \mathcal{V}$ (a node with a large residual, i.e., $z_u \geq \epsilon d_u$). Specifically, for each active u , it updates p_u, z_u , and z_v for $v \in \mathcal{N}(u)$ by using a PUSH operator illustrated on the left. It stops when no active nodes are left. APPR can be implemented locally so that the runtime is independent of \mathcal{G} . In particular, $\text{vol}(\text{supp}(\mathbf{p}))$ is locally bounded, demonstrating the sparsity effect. We restate the existing main results as follows.

Lemma 2.1 (Runtime bound of APPR [2]). *Given $\alpha \in (0, 1)$ and the precision $\epsilon \leq 1/d_s$ for node $s \in \mathcal{V}$ with $\mathbf{p} \leftarrow \mathbf{0}, \mathbf{z} \leftarrow \mathbf{e}_s$ at the initial, APPR($\alpha, \epsilon, s, \mathcal{G}$) defined in (2) returns an estimate \mathbf{p} of $\boldsymbol{\pi}$. There exists a real implementation of (2) (e.g., Algo. 2) such that the runtime $\mathcal{T}_{\text{APPR}}$ satisfies*

$$\mathcal{T}_{\text{APPR}} \leq \Theta(1/(\alpha\epsilon)).$$

Furthermore, the estimate $\hat{\boldsymbol{\pi}} := \mathbf{p}$ satisfies $\|\mathbf{D}^{-1}(\hat{\boldsymbol{\pi}} - \boldsymbol{\pi})\|_\infty \leq \epsilon$ and $\text{vol}(\text{supp}(\hat{\boldsymbol{\pi}})) \leq 2/((1 - \alpha)\epsilon)$.

The main argument for proving Lemma 2.1 is critically based on: 1) $\mathbf{z} \geq \mathbf{0}$ and $\|\mathbf{z}\|_1$ decreases during the updates; 2) for each active $u, z_u \geq \epsilon d_u$, implying that $\|\mathbf{z}\|_1$ is decreased by at least $\alpha\epsilon d_u$, consecutively leading to $\sum_u d_u \leq 1/(\alpha\epsilon)$.

2.2 Problem reformulation

To approximate the PPR vector $\boldsymbol{\pi}$, the original linear system in Equ. (1) can be reformulated as an equivalent symmetric version defined as

$$\mathbf{Q}\mathbf{x} = \mathbf{b}, \quad \text{with } \mathbf{Q} \triangleq \mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2} \text{ and } \mathbf{b} \triangleq \frac{2\alpha}{(1+\alpha)} \mathbf{D}^{-1/2} \mathbf{e}_s, \quad (3)$$

where again \mathbf{e}_s is the standard basis of s , and \mathbf{Q} is a symmetric positive-definite M -matrix with all eigenvalues in $[\frac{2\alpha}{1+\alpha}, \frac{2}{1+\alpha}]$. To solve Equ. (3) is equivalent to solving a quadratic problem

$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathbb{R}^n} \left\{ f(\mathbf{x}) \triangleq \frac{1}{2} \mathbf{x}^\top \mathbf{Q} \mathbf{x} - \mathbf{x}^\top \mathbf{b} \right\}, \quad (4)$$

where f is strongly convex with condition number $1/\alpha$. Indeed, Equ. (3) is a symmetrized version of Equ. (1) and has a unique solution $\mathbf{x}^* = \mathbf{Q}^{-1} \mathbf{b}$. The PPR vector $\boldsymbol{\pi}$ can be recovered from \mathbf{x}^* by $\boldsymbol{\pi} = \mathbf{D}^{1/2} \mathbf{x}^*$. It is convenient to denote estimate of $\boldsymbol{\pi}$ as $\boldsymbol{\pi}^{(t)} \triangleq \mathbf{D}^{1/2} \mathbf{x}^{(t)}$. Given $\mathbf{x}^{(t)}$, we define the residual $\mathbf{r}^{(t)} \triangleq \mathbf{b} - \mathbf{Q}\mathbf{x}^{(t)}$. If $\mathbf{x}^{(t)}$ is returned by a local solver for solving either Equ. (3) or Equ. (4), we then equivalently require $\|\mathbf{D}^{-1/2}(\mathbf{x}^{(t)} - \mathbf{x}^*)\|_\infty \leq \epsilon$. Hence, it is enough to have a stop condition $\|\mathbf{D}^{-1/2} \mathbf{r}^{(t)}\|_\infty \leq 2\alpha\epsilon/(1 + \alpha)$ for local solvers of Equ. (3) and Equ. (4).⁴

Fountoulakis et al. [13] demonstrated that APPR is equivalent to a coordinate descent solver for minimizing f in Equ. (4) and introduced an ISTA-style solver by minimizing $f(\mathbf{x}) + \epsilon\alpha \|\mathbf{D}^{1/2} \mathbf{x}\|_1$, which provides a method with runtime bound $\tilde{\mathcal{O}}(1/(\epsilon\alpha))$ for achieving the same estimation guarantee of APPR. On one hand, one may note that the runtime bound $\Theta(1/(\alpha\epsilon))$ provided in Lemma 2.1 becomes less valuable when $\epsilon \leq 1/m$; on the other hand, all previous local variants [6, 9, 13, 37]

⁴See a justification in Appendix B.

of APPR are critically based on some monotonicity property. This limitation could impede the development of faster local methods that might violate the monotonicity assumption. The following two sections present the techniques and tools to address these challenges.

3 Local Methods via Evolving Set Process

Our investigation begins with the *locally evolving set process*, as inspired by the stochastic counterpart [38]. The process reveals that APPR is essentially a local variant of GS-SOR. We then show how to use this process to build faster local solvers based on GS-SOR. We further develop a *local parallelizable* gradient descent with runtime $\Theta(1/(\alpha\epsilon))$.

3.1 Locally evolving set process

Given α, ϵ, s , and \mathcal{G} , a local solver for Equ. (3) keeps track of an *active set* $\mathcal{S}_t \subset \mathcal{V}$ at each iteration t . That is, only nodes in \mathcal{S}_t are used to update \mathbf{x} or \mathbf{r} . The next set \mathcal{S}_{t+1} is determined by current \mathcal{S}_t and an associated local solver \mathcal{A} . We define this process as the following local evolving set system.

Definition 3.1 (Locally evolving set process). Given a parameter configuration $\theta \triangleq (\alpha, \epsilon, s, \mathcal{G})$, and a local iterative method \mathcal{A} , the locally evolving set process generates a sequence of $(\mathcal{S}_t, \mathbf{x}^{(t)}, \mathbf{r}^{(t)})$ representing as the following dynamic system

$$(\mathcal{S}_{t+1}, \mathbf{x}^{(t+1)}, \mathbf{r}^{(t+1)}) = \Phi_\theta(\mathcal{S}_t, \mathbf{x}^{(t)}, \mathbf{r}^{(t)}, \mathcal{A}), \quad \forall t \geq 0, \quad (5)$$

where $\mathcal{S}_{t+1} \subseteq \mathcal{S}_t \cup (\cup_{u \in \mathcal{S}_t} \mathcal{N}(u))$ and we denote the active set $\mathcal{S}_t = \{u_1, u_2, \dots, u_{|\mathcal{S}_t|}\}$. The set \mathcal{S}_t is maintained via a queue data structure. We say this process *converges* when the last set $\mathcal{S}_T = \emptyset$ if there exists such T ; the generated sequence of active nodes are

$$(\mathcal{S}_0, \mathbf{x}^{(0)}, \mathbf{r}^{(0)}) \rightarrow (\mathcal{S}_1, \mathbf{x}^{(1)}, \mathbf{r}^{(1)}) \rightarrow (\mathcal{S}_2, \mathbf{x}^{(2)}, \mathbf{r}^{(2)}) \rightarrow \dots \rightarrow (\mathcal{S}_T = \emptyset, \mathbf{x}^{(T)}, \mathbf{r}^{(T)}).$$

The runtime of the local solver, \mathcal{A} for this whole local process, is then defined as ⁵

$$\mathcal{T}_\mathcal{A} \triangleq \sum_{t=0}^{T-1} \text{vol}(\mathcal{S}_t).$$

The framework of this set process provides a new way to design local methods. Furthermore, it helps to analyze the convergence and runtime bound of local solvers by characterizing the sequences $\{\text{vol}(\mathcal{S}_t)\}$, and $\{\|\mathbf{r}^{(t)}\|\}$ generated by Φ_θ . To analyze a new runtime bound, for $T \geq 1$, we define the average of the volume of active node sets $\{\text{vol}(\mathcal{S}_t)\}$ and active ratio sequence $\{\gamma_t\}$ as

$$\overline{\text{vol}}(\mathcal{S}_T) \triangleq \frac{1}{T} \sum_{t=0}^{T-1} \text{vol}(\mathcal{S}_t), \quad \bar{\gamma}_T \triangleq \frac{1}{T} \sum_{t=0}^{T-1} \left\{ \gamma_t \triangleq \frac{\sum_{i=1}^{|\mathcal{S}_t|} |\sqrt{d_{u_i}} r_{u_i}^{(t+\Delta_i)}|}{\|\mathbf{D}^{1/2} \mathbf{r}^{(t)}\|_1} \right\}, \quad (6)$$

where Δ_i is a smaller time magnitude. We define $\Delta_i = (i-1)/|\mathcal{S}_t|$ for the analysis of APPR and LOCSOR while $\Delta_i = 0$ for LOGGD in our later analysis. In the rest, we denote $\mathcal{I}_T = \text{supp}(\mathbf{r}^{(T)})$.

These two metrics $\overline{\text{vol}}(\mathcal{S}_T)$ and $\bar{\gamma}_T$ characterize the locality of local methods. To demonstrate this local process, Fig. 1 shows $\text{vol}(\mathcal{S}_t)$ of APPR peaks at the early stage, and the active ratio decreases as the active volume diminishes. The quantity $\overline{\text{vol}}(\mathcal{S}_T)/\bar{\gamma}_T$ is strictly smaller than $1/\epsilon$, indicating that it could serve as a better factor in the runtime analysis.

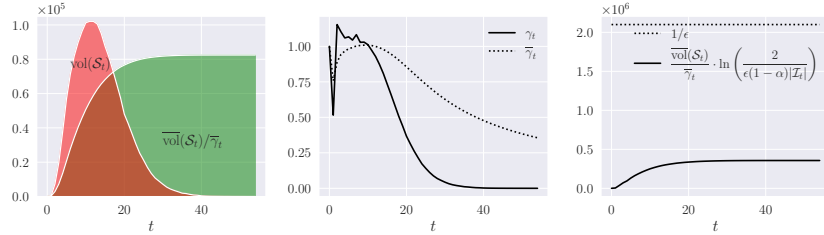


Figure 1: Runtime of APPR in the locally evolving set process on the *com-dblp* graph with $s = 0$, $\alpha = 0.1$, and $\epsilon = 1/m$. The red region of the left figure is $\mathcal{T}_{\text{APPR}}$. The right two figures show active ratios and $\overline{\text{vol}}(\mathcal{S}_T)/\bar{\gamma}_T \leq 1/\epsilon$.

⁵In practice, $\mathcal{T}_\mathcal{A} := \sum_{t=0}^{T-1} (\text{vol}(\mathcal{S}_t) + |\mathcal{S}_t|)$ where we ignore $|\mathcal{S}_t|$ for simplicity as $\text{vol}(\mathcal{S}_t)$ dominates $|\mathcal{S}_t|$.

3.2 APPR via locally evolving set process

We first demonstrate how this locally evolving set process can represent APPR. For solving Equ. (1), the set $\mathcal{S}_0 = \{s\}$ and the queue-based of APPR (see Algo. 2 in Appendix A) naturally forms a sequence of active sets from $\mathcal{S}_0 = \{s\}$ to $\mathcal{S}_T = \emptyset$, hence converging. Active nodes u in queue satisfy $z_u \geq \epsilon d_u$. To delineate successive iterations \mathcal{S}_t and \mathcal{S}_{t+1} , one can insert $*$ at the beginning of \mathcal{S}_t . After processing \mathcal{S}_t , it serves as an indicator for the next iteration. The star $*$ is reinserted into the queue iteratively until the queue is empty. We use a slightly different notation for presenting tuple $(\mathcal{S}_t, \mathbf{p}^{(t)}, \mathbf{z}^{(t)})$ to consistent with Sec. 2.1 and write out such evolving process as follows

$$\boxed{\begin{aligned} \Phi_\theta(\mathcal{S}_t, \mathbf{p}^{(t)}, \mathbf{z}^{(t)}, \mathcal{A} = \text{APPR}) : & \text{ for } u_i \text{ in } \mathcal{S}_t := \{u_1, u_2, \dots, u_{|\mathcal{S}_t|}\} \text{ do} \\ & \mathbf{p}^{(t+\Delta_i+1)} \leftarrow \mathbf{p}^{(t+\Delta_i)} + \alpha z_{u_i}^{(t+\Delta_i)} \mathbf{e}_{u_i}, \quad \Delta_i := (i-1)/|\mathcal{S}_t| \\ & \mathbf{z}^{(t+\Delta_i+1)} \leftarrow \mathbf{z}^{(t+\Delta_i)} - \frac{(1+\alpha)}{2} z_{u_i}^{(t+\Delta_i)} \mathbf{e}_{u_i} + \frac{(1-\alpha)}{2} z_{u_i}^{(t+\Delta_i)} \mathbf{A} \mathbf{D}^{-1} \mathbf{e}_{u_i} \end{aligned}} \quad (7)$$

The following lemma establishes the equivalence between APPR and the local variant of GS-SOR method (see Appendix B.1) and provides a new evolving-based bound.

Lemma 3.2 (New local evolving-based bound for APPR). *Let $\mathbf{M} = \alpha^{-1}(\mathbf{I} - \frac{1-\alpha}{2}(\mathbf{I} + \mathbf{A} \mathbf{D}^{-1}))$ and $\mathbf{s} = \mathbf{e}_s$. The linear system $\mathbf{M} \boldsymbol{\pi} = \mathbf{s}$ is equivalent to Equ. (1). Given $\mathbf{p}^{(0)} = \mathbf{0}$, $\mathbf{z}^{(0)} = \mathbf{e}_s$ with $\omega \in (0, 2)$, the local variant of GS-SOR (15) for $\mathbf{M} \boldsymbol{\pi} = \mathbf{s}$ can be formulated as*

$$\mathbf{p}^{(t+\Delta_i+1)} \leftarrow \mathbf{p}^{(t+\Delta_i)} + \frac{\omega z_{u_i}^{(t+\Delta_i)}}{M_{u_i u_i}} \mathbf{e}_{u_i}, \quad \mathbf{z}^{(t+\Delta_i+1)} \leftarrow \mathbf{z}^{(t+\Delta_i)} - \frac{\omega z_{u_i}^{(t+\Delta_i)}}{M_{u_i u_i}} \mathbf{M} \mathbf{e}_{u_i},$$

where u_i is an active node in \mathcal{S}_t satisfying $z_{u_i} \geq \epsilon d_{u_i}$ and $\Delta_i = (i-1)/|\mathcal{S}_t|$. Furthermore, when $\omega = \frac{1+\alpha}{2}$, this method reduces to APPR given in (7), and there exists a real implementation (Aglo. 2) of APPR such that the runtime $\mathcal{T}_{\text{APPR}}$ is bounded, that is

$$\mathcal{T}_{\text{APPR}} \leq \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\alpha \bar{\gamma}_T} \ln \frac{C_T}{\epsilon}, \text{ where } \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\hat{\gamma}_T} \leq \frac{1}{\epsilon}, C_T = \frac{2}{(1-\alpha)|\mathcal{I}_T|}, \hat{\gamma}_T \triangleq \frac{1}{T} \sum_{t=0}^{T-1} \left\{ \frac{\sum_{i=1}^{|\mathcal{S}_t|} |z_{u_i}^{(t+\Delta_i)}|}{\|\mathbf{z}^{(t)}\|_1} \right\}.$$

3.3 Faster local variant of GS-SOR

Lemma 3.2 points to the sub-optimality of APPR, as GS-SOR allows for a larger ω . For solving Equ. (3), since APPR essentially serves as a local variant of GS-SOR, we can develop a faster local variant based SOR. To extend this method to solve Equ. (3), we propose a local GS-SOR based on an evolving set process, namely LOCSOR, as the following

$$\boxed{\begin{aligned} \Phi_\theta(\mathcal{S}_t, \mathbf{x}^{(t)}, \mathbf{r}^{(t)}, \mathcal{A} = \text{LOCSOR}) : & \text{ for } u_i \text{ in } \mathcal{S}_t := \{u_1, \dots, u_{|\mathcal{S}_t|}\} \text{ and do} \\ & \mathbf{x}^{(t+\Delta_i+1)} \leftarrow \mathbf{x}^{(t+\Delta_i)} + \omega r_{u_i}^{(t+\Delta_i)} \mathbf{e}_{u_i}, \quad \Delta_i = (i-1)/|\mathcal{S}_t| \\ & \mathbf{r}^{(t+\Delta_i+1)} \leftarrow \mathbf{r}^{(t+\Delta_i)} - \omega r_{u_i}^{(t+\Delta_i)} \mathbf{e}_{u_i} + \frac{(1-\alpha)\omega}{1+\alpha} r_{u_i}^{(t+\Delta_i)} \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2} \mathbf{e}_{u_i} \end{aligned}} \quad (8)$$

When $\omega \in (0, 1]$, the residual \mathbf{r} is still nonnegative and monotonically decreasing, we establish the convergence of LOCSOR stated in the following theorem.

Theorem 3.3 (Runtime bound of LOCSOR ($\omega = 1$)). *Given the configuration $\theta = (\alpha, \epsilon, s, \mathcal{G})$ with $\alpha \in (0, 1)$ and $\epsilon \leq 1/d_s$ and let $\mathbf{r}^{(T)}$ and $\mathbf{x}^{(T)}$ be returned by LOCSOR defined in (8) for solving Equ. (3). There exists a real implementation of (8) such that the runtime $\mathcal{T}_{\text{LOCSOR}}$ is bounded by*

$$\frac{1+\alpha}{2} \cdot \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\alpha \bar{\gamma}_T} \left(1 - \frac{\|\mathbf{D}^{1/2} \mathbf{r}^{(T)}\|_1}{\|\mathbf{D}^{1/2} \mathbf{r}^{(0)}\|_1} \right) \leq \mathcal{T}_{\text{LOCSOR}} \leq \frac{1+\alpha}{2} \cdot \min \left\{ \frac{1}{\alpha \epsilon}, \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\alpha \bar{\gamma}_T} \ln \frac{C}{\epsilon} \right\}$$

where $\overline{\text{vol}}(\mathcal{S}_T)$ and $\bar{\gamma}_T$ are defined in (6) and $C = \frac{1+\alpha}{(1-\alpha)|\mathcal{I}_T|}$ with $\mathcal{I}_T = \text{supp}(\mathbf{r}^{(T)})$. Furthermore, $\overline{\text{vol}}(\mathcal{S}_T)/\bar{\gamma}_T \leq 1/\epsilon$ and the local estimate $\hat{\boldsymbol{\pi}} := \mathbf{D}^{1/2} \mathbf{x}^{(T)}$ satisfies $\|\mathbf{D}^{-1}(\hat{\boldsymbol{\pi}} - \boldsymbol{\pi})\|_\infty \leq \epsilon$.

Our new evolving bound $\tilde{\mathcal{O}}(\overline{\text{vol}}(\mathcal{S}_T)/(\alpha\bar{\gamma}_T))$ mirroring the actual performance of APPR and empirically much smaller than $\Theta(1/(\alpha\epsilon))$ as illustrated in Fig. 2. Our lower bounds are quite effective when ϵ is relatively large, while our upper bound is better than Anderson’s when ϵ is small. When $\epsilon \ll \Theta(1/m)$, this new bound is superior to both $\mathcal{O}(1/(\alpha\epsilon))$ and $\tilde{\mathcal{O}}(1/(\sqrt{\alpha}\epsilon))$. This superiority is evident when compared to algorithms like ISTA or FISTA [5] to minimize the ℓ_1 -regularization of f for obtaining an approximate solution of Equ. (3). Additionally, when $\omega \in (1, 2)$ and recalling that \mathbf{Q} is an M -matrix, the standard analysis of SOR shows that the spectral norm of the iteration matrix must be larger than $|\omega - 1|$. Hence, $0 < \omega < 2$ if and only if global SOR converges [55]. When ω^* is optimal (the point that the spectral radius of the iteration matrix is minimized), we have the following result.

Corollary 3.4. *Let $\omega = \omega^* \triangleq 2/(1 + \sqrt{1 - (1 - \alpha)^2/(1 + \alpha)^2})$ and $\mathcal{S}_t = \mathcal{V}, \forall t \geq 0$ during the updates, the global version of LOCSOR has the following convergence bound*

$$\|\mathbf{r}^{(t)}\|_2 \leq \frac{2}{(1 + \alpha)\sqrt{d_s}} \left(\frac{1 - \sqrt{\alpha}}{1 + \sqrt{\alpha}} + \epsilon_t \right)^t,$$

where ϵ_t are small positive numbers with $\lim_{t \rightarrow \infty} \epsilon_t = 0$.

Asymptotically, when $\epsilon_t = o(\sqrt{\alpha})$, then the runtime of global LOCSOR is $\tilde{\mathcal{O}}(m/\sqrt{\alpha})$ where $\tilde{\mathcal{O}}$ hides $\log 1/\epsilon$. The main difficulty of analyzing the *optimal* local LOCSOR is that the nonnegativity and monotonicity of $\mathbf{r}^{(t)}$ do not hold. Instead, by using a parameterized second-order difference equation, we develop new techniques based on the Chebyshev method detailed in Sec. 4.

3.4 Parallelizable local gradient descent

One disadvantage of LOCSOR is its limited potential for parallelization. The standard GD $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \nabla f(\mathbf{x}^{(t)})$ (step size = 1), in contrast, is easy to parallelize across the coordinates of the update. Instead of updating \mathbf{r} and \mathbf{x} synchronously per-coordinate, we propose the following

$$\Phi_\theta(\mathcal{S}_t, \mathbf{x}^{(t)}, \mathbf{r}^{(t)}, \mathcal{A} = \text{LocGD}) : \mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} + \mathbf{r}_{\mathcal{S}_t}^{(t)}, \mathbf{r}^{(t+1)} \leftarrow \mathbf{r}^{(t)} - \mathbf{Q}\mathbf{r}_{\mathcal{S}_t}^{(t)} \quad (9)$$

Every coordinate in \mathcal{S}_t is updated in parallel at iteration t . Interestingly, LocGD exhibits nonnegativity and monotonicity properties, and its runtime complexity is similar to that of LOCSOR, as stated in the following theorem (To remind, $\Delta_i = 0$ for $\bar{\gamma}_T$ of LocGD in Equ. (6)).

Theorem 3.5 (Runtime bound of LocGD). *Given the configuration $\theta = (\alpha, \epsilon, s, \mathcal{G})$ with $\alpha \in (0, 1)$ and $\epsilon \leq 1/d_s$ and let $\mathbf{r}^{(T)}$ and $\mathbf{x}^{(T)}$ be returned by LocGD defined in (9) for solving Equ. (4). There exists a real implementation of (9) such that the runtime $\mathcal{T}_{\text{LocGD}}$ is bounded by*

$$\frac{1 + \alpha}{2} \cdot \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\alpha\bar{\gamma}_T} \left(1 - \frac{\|\mathbf{D}^{1/2}\mathbf{r}^{(T)}\|_1}{\|\mathbf{D}^{1/2}\mathbf{r}^{(0)}\|_1} \right) \leq \mathcal{T}_{\text{LocGD}} \leq \frac{1 + \alpha}{2} \cdot \min \left\{ \frac{1}{\alpha\epsilon}, \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\alpha\bar{\gamma}_T} \ln \frac{C}{\epsilon} \right\},$$

where $C = (1 + \alpha)/((1 - \alpha)|\mathcal{I}_T|)$, $\mathcal{I}_T = \text{supp}(\mathbf{r}^{(T)})$. Furthermore, $\overline{\text{vol}}(\mathcal{S}_T)/\bar{\gamma}_T \leq 1/\epsilon$ and the estimate $\hat{\boldsymbol{\pi}} := \mathbf{D}^{1/2}\mathbf{x}^{(T)}$ satisfies $\|\mathbf{D}^{-1}(\hat{\boldsymbol{\pi}} - \boldsymbol{\pi})\|_\infty \leq \epsilon$.

Note that $\bar{\gamma}_T$ of LocGD is empirically smaller than that of LOCSOR. Hence, LocGD is empirically slower than LOCSOR by only a small constant factor (e.g., twice as slow), a finding consistent with observations of their standard counterparts [19]. Nonetheless, LocGD is much simpler and more amenable to parallelization on platforms such as GPUs compared to APPR.

4 Accelerated Local Iterative Methods

This section presents our key contributions where we propose faster local methods based on the Chebyshev method for solving Equ. (3) and the Heavy-Ball (HB) method for Equ. (4).

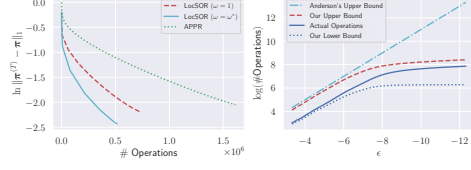


Figure 2: Comparison of runtime between APPR and LOCSOR (left) and runtime bounds (right) as a function of ϵ . We used the same setting as in Fig. 1.

4.1 Local Chebyshev method

Compared with GS and GD, the standard Chebyshev method offers optimal acceleration in solving Equ. (3). Following existing techniques (e.g., see d’Aspremont et al. [11]), we show there exists an upper runtime bound $\tilde{\mathcal{O}}(m/\sqrt{\alpha})$ to meeting the stopping condition where $\tilde{\mathcal{O}}$ hides $\log 1/\epsilon$ (we presented it in Theorem C.6). Hence, the Chebyshev method is one of the optimal first-order linear solvers for solving Equ. (3). However, localizing Chebyshev poses greater challenges due to the additional momentum vector involved in updating $\mathbf{x}^{(t)}$. Our key observation is that *if a substantial reduction in the magnitudes of $\mathbf{r}^{(t)}$ is required within a subset of \mathcal{S}_t , then the corresponding momentum coordinates are likely to possess significant acceleration energy*. Intuitively, a viable strategy involves localizing both the residual and momentum vectors. For $t \geq 1$, denote the “momentum” vector as $\mathbf{\Delta}^{(t)} := \mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}$ and $\delta_{t:t+1} = \delta_t \delta_{t+1}$, we propose the localized Chebyshev as the following

$$\boxed{\begin{aligned} & \Phi_\theta(\mathcal{S}_t, \mathbf{x}^{(t)}, \mathbf{r}^{(t)}, \mathcal{A} = \text{LOCCH}) : \\ & \hat{\mathbf{x}}^{(t)} \leftarrow (1 + \delta_{t:t+1})\mathbf{r}_{\mathcal{S}_t}^{(t)} + \delta_{t:t+1}\mathbf{\Delta}_{\mathcal{S}_t}^{(t)}, \quad \delta_{t+1} = \left(2\frac{1+\alpha}{1-\alpha} - \delta_t\right)^{-1} \\ & \mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} + \hat{\mathbf{x}}^{(t)}, \quad \mathbf{r}^{(t+1)} \leftarrow \mathbf{r}^{(t)} - \hat{\mathbf{x}}^{(t)} + \frac{1-\alpha}{1+\alpha}\mathbf{W}\hat{\mathbf{x}}^{(t)}, \end{aligned}} \quad (10)$$

where $t \geq 1$ with the initials $\mathbf{x}^{(0)} = \mathbf{0}$, $\mathbf{x}^{(1)} = \mathbf{r}^{(0)}$, $\delta_0 = 0$, $\delta_1 = (1 - \alpha)/(1 + \alpha)$, and $\mathbf{W} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ is normalized adjacency matrix. Our key strategy for analyzing (10) is to rewrite the updates of $\mathbf{r}^{(t)}$ as a nonhomogeneous second-order difference equation (see details in Lemma C.8)

$$\mathbf{r}^{(t+1)} - 2\delta_{t+1}\mathbf{W}\mathbf{r}^{(t)} + \delta_{t:t+1}\mathbf{r}^{(t-1)} = \sum_{j=0}^t \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^t \delta_{r:r+1} \mathbf{Q}\mathbf{r}_{\mathcal{S}_j}^{(j)} \right), \quad (11)$$

where we denote $\mathcal{S}_{j,t} = \mathcal{S}_j \cap \dots \cap \mathcal{S}_{t-1} \cap \bar{\mathcal{S}}_t$ given $t \geq j \geq 0$ where $\bar{\mathcal{S}}_t = \mathcal{V} \setminus \mathcal{S}_t$. In the rest, we define $\tilde{\alpha} = (1 - \sqrt{\alpha})/(1 + \sqrt{\alpha})$ and recall the eigendecomposition of $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$. Based on the above Equ. (11), we have the following key lemma.

Lemma 4.1. *Given $t \geq 1$, $\mathbf{x}^{(0)} = \mathbf{0}$, $\mathbf{x}^{(1)} = \mathbf{r}^{(0)}$. The residual $\mathbf{r}^{(t)}$ of LOCCH defined in (10) can be expressed as the following*

$$\mathbf{V}^\top \mathbf{r}^{(t)} = \delta_{1:t} \mathbf{Z}_t \mathbf{V}^\top \mathbf{r}^{(0)} + \delta_{1:t} t \mathbf{u}_{0,t} + 2 \sum_{k=1}^{t-1} \delta_{k+1:t} (t-k) \mathbf{u}_{k,t},$$

where \mathbf{Z}_t is a diagonal matrix such that $\|\mathbf{Z}_t\|_2 \leq 1$ and

$$\mathbf{u}_{k,t} = \begin{cases} \sum_{j=1}^{t-1} \frac{\delta_{2;j}}{t} \mathbf{H}_{j,t} \left(\mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{0,j}}^{(0)} & \text{if } k = 0 \\ \sum_{j=k}^{t-1} \frac{\delta_{k+1;j}}{(t-k)} \mathbf{H}_{j,t} \left(\frac{1+\alpha}{1-\alpha} \mathbf{I} - \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{k,j}}^{(k)} & \text{if } k \geq 1, \end{cases}$$

where $\mathbf{H}_{k,t}$ is a diagonal matrix such that $\|\mathbf{H}_{k,t}\|_2 \leq t - k$.

This key lemma essentially captures the process of residual reduction $\mathbf{r}^{(t)}$ of LOCCH. Specifically, given current iteration t , we define the *running residual reduction rate* for $\mathbf{r}^{(k)}$ with $k = 0, 1, 2, \dots, t-1$ of step t as $\beta_{k,t}$, that is,

$$\beta_{k,t} \triangleq \frac{\|\mathbf{u}_{k,t}\|_2}{\|\mathbf{r}^{(k)}\|_2}, \quad \beta_k \triangleq \max_t \beta_{k,t}. \quad (12)$$

Note that

$$\beta_{k,t} \leq \underbrace{\frac{2\|\mathbf{r}_{\mathcal{S}_k}^{(k)}\|_2}{(1-\alpha)\|\mathbf{r}^{(k)}\|_2}}_{\approx \mathcal{O}(\epsilon)} + \underbrace{\sum_{j=k+1}^{t-1} \frac{4\tilde{\alpha}^{j-k}(t-j)}{(1-\alpha)(t-k)} \frac{\|\mathbf{r}_{\mathcal{S}_{k,j}}^{(k)}\|_2}{\|\mathbf{r}^{(k)}\|_2}}_{\leq \frac{4\tilde{\alpha}}{(1-\alpha)(1-\tilde{\alpha})}},$$

where whether the last term can be even smaller depends on $\|\mathbf{r}_{\mathcal{S}_{k,j}}^{(k)}\|_2$ for $\mathcal{S}_{k,j} = \mathcal{S}_k \cap \dots \cap \mathcal{S}_{j-1} \cap \bar{\mathcal{S}}_j$. However, we notice that the running geometric mean $\bar{\beta}_t \triangleq (\prod_{j=0}^{t-1} (1 + \beta_j))^{1/t}$ is even smaller in practice. Based on these observations and the assumption on $\bar{\beta}_t$, we establish the following theorem.

Theorem 4.2 (Runtime bound of LOCCH). *Given the configuration $\theta = (\alpha, \epsilon, s, \mathcal{G})$ with $\alpha \in (0, 1)$ and $\epsilon \leq 1/d_s$ and let $\mathbf{r}^{(T)}$ and $\mathbf{x}^{(T)}$ be returned by LOCCH defined in (10) for solving Equ. (3). For $t \geq 1$, the residual magnitude $\|\mathbf{r}^{(t)}\|_2$ has the following convergence bound*

$$\|\mathbf{r}^{(t)}\|_2 \leq \delta_{1:t} \prod_{j=0}^{t-1} (1 + \beta_j) y_t,$$

where y_t is a sequence of positive numbers solving $y_{t+1} - 2y_t + y_{t-1}/((1 + \beta_{t-1})(1 + \beta_t)) = 0$ with $y_0 = y_1 = \|\mathbf{r}^{(0)}\|_2$. Suppose the geometric mean $\bar{\beta}_t \triangleq (\prod_{j=0}^{t-1} (1 + \beta_j))^{1/t}$ of β_t be such that $\bar{\beta}_t = 1 + \frac{c\sqrt{\alpha}}{1-\sqrt{\alpha}}$ where $c \in [0, 2)$. There exists a real implementation of (10) such that the runtime $\mathcal{T}_{\text{LocCH}}$ is bounded by

$$\mathcal{T}_{\text{LocCH}} \leq \Theta \left(\frac{(1 + \sqrt{\alpha}) \overline{\text{vol}}(\mathcal{S}_T)}{\sqrt{\alpha}(2 - c)} \ln \frac{2y_T}{\epsilon} \right).$$

Golub & Overton [18] considered the approximate Chebyshev method by assuming that the inexact residual is sufficiently smaller than $\epsilon \|\mathbf{r}^{(t)}\|_2$, where ϵ must be small enough to ensure convergence. However, this assumption is overly stringent for our case. The novelty of our analysis lies in a more elegant treatment of a parameterized second-order difference equation, allowing us to circumvent this assumption. The nested APGD(ϵ), namely ASPR proposed in Martínez-Rubio et al. [37] has runtime complexity $\tilde{\mathcal{O}}(|\mathcal{S}^*| \overline{\text{vol}}(\mathcal{S}^*) / \sqrt{\alpha} + |\mathcal{S}^*| \overline{\text{vol}}(\mathcal{S}^*))$ where \mathcal{S}^* is

the optimal support of $\arg \min_{\mathbf{x}} \{f(\mathbf{x}) + \epsilon \alpha \|\mathbf{D}^{1/2} \mathbf{x}\|_1\}$ and $\overline{\text{vol}}(\mathcal{S}^*) = \text{nnz}(\mathbf{Q}_{\mathcal{S}^*, \mathcal{S}^*})$. Although it is difficult to compare our bound to this, one limitation of ASPR is that it assumes to call APGD(ϵ) $\mathcal{O}(|\mathcal{S}^*|)$ times to finish in the worst case. However, our iteration complexity is $\tilde{\mathcal{O}}(1/(\sqrt{\alpha}(2 - c)))$. Asymptotically, $c = o(\sqrt{\alpha})$ ($\epsilon \rightarrow 0$), our complexity is $\tilde{\mathcal{O}}(1/\sqrt{\alpha})$ could be better than $\tilde{\mathcal{O}}(|\mathcal{S}^*|/\sqrt{\alpha})$. Fig. 3 presents a preliminary study on ASPR, indicating that it requires more operations than APPR.

We conclude our analysis by presenting a similar result for the local Heavy Ball (HB). Note the HB method is the one when $\delta_t \delta_{t+1} \rightarrow \tilde{\alpha}^2$ where $\tilde{\alpha} = (1 - \sqrt{\alpha})/(1 + \sqrt{\alpha})$. Hence, it has similar convergence analyses as to LOCCH shown in Theorem D.8. The LOCHB has the following updates

$$\begin{aligned} \Phi(\mathcal{S}_t; s, \epsilon, \alpha, \mathcal{G}, \mathcal{A} = \text{LOCHB}) : \\ \hat{\mathbf{x}}^{(t)} \leftarrow (1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_t}^{(t)} + \tilde{\alpha}^2 \mathbf{\Delta}_{\mathcal{S}_t}^{(t)} \\ \mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} + \hat{\mathbf{x}}^{(t)}, \quad \mathbf{r}^{(t+1)} \leftarrow \mathbf{r}^{(t)} - \hat{\mathbf{x}}^{(t)} + \frac{1-\alpha}{1+\alpha} \mathbf{W} \hat{\mathbf{x}}^{(t)}. \end{aligned} \quad (13)$$

5 Generalization and Open Problems

Our framework can be applied to various local methods for large-scale linear systems. Extensions of this framework to other linear systems are detailed in Tab. 2 of Appendix E. More broadly, we consider the feasibility of local methods for solving $\mathbf{Q}\mathbf{x} = \mathbf{b}$, where \mathbf{b} is a sparse vector ($|\text{supp}(\mathbf{b})| \ll n$) and \mathbf{Q} is a positive definite, graph-induced matrix with bounded eigenvalues. This leads us to question whether all standard iterative methods can be effectively localized, raising two key questions

1. Given a graph-induced matrix \mathbf{Q} and its spectral radius $\rho(\mathbf{Q}) < 1$, a standard solver \mathcal{A} , and the corresponding local evolving process $\Phi_\theta(\mathcal{S}_t, \mathbf{x}^{(t)}, \mathbf{r}^{(t)}, \text{LOC}\mathcal{A})$, does a localized version of \mathcal{A} (over \mathcal{S}_t) converge and have local runtime bounds?
2. Based on current analysis, Theorem 4.2 relies on the geometric mean of residual reduction on $\|\mathbf{r}^{(k)}\|_2$ being small. How feasible is acceleration within locality constraints? Specifically, a stronger bound could be established for solving Equ. (3) via LOCHB and LOCCH, with a graph-independent bound of

$$\mathcal{T}_{\text{Loc}\mathcal{A}} = \Theta \left(\frac{\overline{\text{vol}}(\mathcal{S}_T)}{\sqrt{\alpha} \tilde{\gamma}_T} \ln \frac{C}{\epsilon} \right), \text{ where } C \text{ a graph-independent constant.}$$

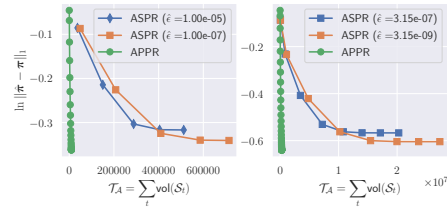


Figure 3: Comparison of runtime between APPR and ASPR. The setting is the same as in Fig. 1. Left $\epsilon = 10^{-4}$ while $\frac{1}{n}$ for right.

Additionally, this work primarily focuses on using first-order neighbors at each iteration. An area for future exploration is generalizing to higher-order neighbors to determine if this leads to faster or more efficient methodologies, which remains an open question.

6 Experiments

We conduct experiments over 17 graphs to solve (3) and explore the local clustering task. We address the following questions: 1) Can iterative solvers be effectively localized? 2) How does the performance of accelerated local methods compare to non-accelerated ones? 3) Can our proposed methods reduce the number of operations required for local clustering? ⁶

Baselines. We consider four baselines: 1) Conjugate Gradient Method (CGM) as a benchmark to compare local and non-local methods; 2) ISTA, the local method proposed by Fountoulakis et al. [13]; 3) FISTA, the momentum-based local algorithm proposed by Hu [22]; and 4) APPR, the classic local method proposed by Andersen et al. [2]. All methods are implemented in Python 3.10 with the numba library [33].

Efficiency of localized algorithms. To compare local solvers to their standard counterparts, we set $\alpha = 0.1$, randomly select 50 nodes from each graph to serve as e_s in (3), and run standard GD, SOR, HB, and CH solvers along with their local counterparts: LOC GD, LOC SOR, LOC HB, and LOC CH. We measure the efficiency by the *speedup*, defined as the ratio between the runtime of the standard and local solver. The range of ϵ is $\epsilon \in [\frac{\alpha}{2(1+\alpha)d_s}, 10^{-4}/n]$. The results, presented in Fig. 4, clearly indicate that our design demonstrates significant speedup, especially around $\epsilon = 1/n$. Remarkably, they still show better performance even when $\epsilon \approx 10^{-4}/n$ (Fig. 5). These results suggest that local solvers are preferred over non-local ones when the precision requirement is in this range.

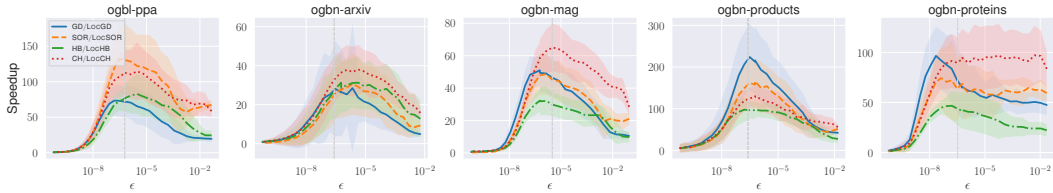


Figure 4: The speedup of local solvers as a function of ϵ . The vertical line is $\epsilon = 1/n$.

Comparison with local baselines and CGM. We next compare our three accelerated methods with four baselines. Fig. 5 presents the ℓ_1 -estimation error in terms of the number of operations (quantified as $t \cdot \overline{\text{vol}}(S_t)$) executed. It is evident that our three solvers use significantly fewer operations compared to CGM and the other three local methods. Again, due to maintaining a nondecreasing set of active nodes, ISTA and FISTA require more operations than the locally evolving set process. Ours are more efficient than APPR, where $r^{(0)} = e_s$ is used.

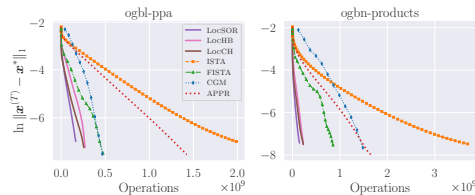


Figure 5: Estimation error as a function of operations required. ($\epsilon = 10^{-4}/n$)

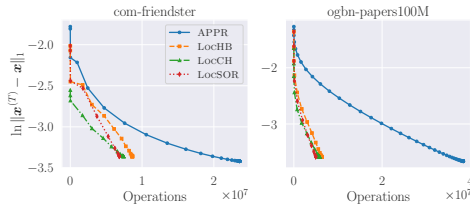


Figure 6: Performance on large-scale graphs.

papers100M has 111M nodes and 1.6B edges while com-friendster has 65M nodes with 1.8B edges. Results are shown in Fig. 6; compared with current default local methods, it is several times faster, especially on ogbn-papers100M.

Efficiency in terms of α and huge-graph tests. We demonstrate the performance of local solvers in terms of different α ranging from 0.005 to 0.25. Interestingly, in Fig. 13, LOC GD show faster convergence when α is small; this may be because of the advantages of monotonicity properties, which is not present in the accelerated methods. However, in other regions of α , accelerated methods are faster. We also tested local solvers on two large-scale graphs where papers100M has 111M nodes and 1.6B edges while com-friendster has 65M nodes with 1.8B edges. Results are shown in Fig. 6; compared with current default local methods, it is several times faster, especially on ogbn-papers100M.

⁶Additional experimental results, setups, and algorithm parameters are provided in Appendix F.

Case study on local clustering.

Following the experimental setup in Fountoulakis et al. [13], we consider the task of local clustering on 15 graphs. As partially demonstrated in Tab. 1, compared with APPR and FISTA, LocSOR uses the least operations and is the fastest, demonstrating the advantages of our proposed local solvers.

Table 1: Operations/runtime comparison on local clustering.

\mathcal{G}	Operations			Run Time (Seconds)		
	APPR	LocSOR	FISTA	APPR	LocSOR	FISTA
\mathcal{G}_1	6.9e+05	6.5e+04	5.7e+05	0.127	0.043	0.093
\mathcal{G}_2	6.7e+05	8.9e+04	4.4e+05	0.362	0.125	0.308
\mathcal{G}_3	4.3e+05	3.5e+04	2.9e+05	0.069	0.014	0.042
\mathcal{G}_4	5.7e+05	7.6e+04	4.4e+05	0.357	0.175	0.229
\mathcal{G}_5	5.4e+05	9.0e+04	5.0e+05	0.072	0.055	0.084

7 Limitations and Conclusion

Our proposed algorithms may have the following limitations: 1) When α is small, the acceleration effect partially disappears, as observed in Fig. 13. This may be due to the limitations of global counterparts, where the residual may not decrease early; 2) Our new accelerated bound for LocCH depends on an empirically reasonable assumption of residual reduction but lacks theoretical justification.

We propose using a new locally evolving set process framework to characterize algorithm locality and demonstrate that several standard iterative solvers can be effectively localized, significantly speeding up current local solvers. Our local methods could be efficiently implemented into GPU architecture to accelerate the training of GNNs such as APPNP [27] and PPRGo [7]. We also offer open problems in developing faster local methods. It is worth exploring whether subsampling active nodes stochastically or using different queue strategies (priority rather than FIFO) could help speed up the framework further. It also remains interesting to see how to design local algorithms for conjugate direction-based methods such as CGM.

Acknowledgments and Disclosure of Funding

The authors would like to thank the anonymous reviewers for their helpful comments. The work of Baojian Zhou is sponsored by Shanghai Pujiang Program (No. 22PJ1401300) and the National Natural Science Foundation of China (No. KRH2305047). The work of Deqing Yang is supported by Chinese NSF Major Research Plan No.92270121.

References

- [1] Alon, N., Rubinfeld, R., Vardi, S., and Xie, N. Space-efficient local computation algorithms. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms (SODA)*, pp. 1132–1139. SIAM, 2012.
- [2] Andersen, R., Chung, F., and Lang, K. Local graph partitioning using PageRank vectors. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2006.
- [3] Andersen, R., Gharan, S. O., Peres, Y., and Trevisan, L. Almost optimal local graph clustering using evolving sets. *Journal of the ACM (JACM)*, 63(2):1–31, 2016.
- [4] Anikin, A., Gasnikov, A., Gornov, A., Kamzolov, D., Maximov, Y., and Nesterov, Y. Efficient numerical methods to solve sparse linear equations with application to PageRank. *Optimization Methods and Software*, pp. 1–29, 2020.
- [5] Beck, A. and Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2:183–202, 2009.
- [6] Berkhin, P. Bookmark-coloring algorithm for personalized pagerank computing. *Internet Mathematics*, 3(1):41–62, 2006.
- [7] Bojchevski, A., Klicpera, J., Perozzi, B., Kapoor, A., Blais, M., Rózemberczki, B., Lukasik, M., and Günnemann, S. Scaling graph neural networks with approximate pagerank. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, 2020.
- [8] Boyd, S., Diaconis, P., and Xiao, L. Fastest mixing markov chain on a graph. *SIAM review*, 46(4):667–689, 2004.
- [9] Chen, Z., Guo, X., Zhou, B., Yang, D., and Skiena, S. Accelerating personalized PageRank vector computation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 262–273, 2023.
- [10] Chung, F. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences (PNAS)*, 104(50):19735–19740, 2007.
- [11] d’Aspremont, A., Scieur, D., Taylor, A., et al. Acceleration methods. *Foundations and Trends® in Optimization*, 5(1-2):1–245, 2021.
- [12] Fountoulakis, K. and Yang, S. Open problem: Running time complexity of accelerated ℓ_1 -regularized PageRank. In *Conference on Learning Theory (COLT)*, 2022.
- [13] Fountoulakis, K., Roosta-Khorasani, F., Shun, J., Cheng, X., and Mahoney, M. W. Variational perspective on local graph clustering. *Mathematical Programming*, 174(1):553–573, 2019.
- [14] Fountoulakis, K., Wang, D., and Yang, S. p-norm flow diffusion for local graph clustering. In *ICML*, 2020.
- [15] Gasteiger, J., Weissenberger, S., and Günnemann, S. Diffusion improves graph learning. In *Advances in neural information processing systems (NeurIPS)*, 2019.
- [16] Gleich, D. and Mahoney, M. Anti-differentiating approximation algorithms: A case study with min-cuts, spectral, and flow. In *International Conference on Machine Learning*, pp. 1018–1025. PMLR, 2014.
- [17] Gleich, D. F. PageRank beyond the web. *siam REVIEW*, 57(3):321–363, 2015.

- [18] Golub, G. H. and Overton, M. L. The convergence of inexact chebyshev and richardson iterative methods for solving linear systems. *Numerische Mathematik*, 53(5):571–593, 1988.
- [19] Golub, G. H. and Van Loan, C. F. *Matrix computations (4th Edition)*. JHU press, 2013.
- [20] Guo, X., Zhou, B., and Skiena, S. Subset node representation learning over large dynamic graphs. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 516–526, 2021.
- [21] Guo, X., Zhou, B., and Skiena, S. Subset node anomaly tracking over large dynamic graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 475–485, 2022.
- [22] Hu, C. Local graph clustering using l_1 -regularized PageRank algorithms. Master’s thesis, University of Waterloo, 2020.
- [23] Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.
- [24] Johnson, R. and Zhang, T. Graph-based semi-supervised learning and spectral kernel design. *IEEE Transactions on Information Theory*, 54(1):275–288, 2008.
- [25] Kapralov, M., Lattanzi, S., Nouri, N., and Tardos, J. Efficient and local parallel random walks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- [26] Klamkin, M. and Newman, D. J. Extensions of the weierstrass product inequalities. *Mathematics Magazine*, 43(3):137–141, 1970.
- [27] Klicpera, J., Bojchevski, A., and Günnemann, S. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations (ICLR)*, 2019.
- [28] Kloster, K. and Gleich, D. F. A nearly-sublinear method for approximating a column of the matrix exponential for matrices from large, sparse networks. In *Algorithms and Models for the Web Graph: 10th International Workshop, WAW 2013, Cambridge, MA, USA, December 14-15, 2013, Proceedings 10*, pp. 68–79. Springer, 2013.
- [29] Kloster, K. and Gleich, D. F. Heat kernel based community detection. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, 2014.
- [30] Kloumann, I. M. and Kleinberg, J. M. Community membership identification from small seed sets. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, 2014.
- [31] Koutis, I., Miller, G. L., and Peng, R. A nearly- $m \log n$ time solver for sdd linear systems. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 590–598. IEEE, 2011.
- [32] Łacki, J., Mitrović, S., Onak, K., and Sankowski, P. Walking randomly, massively, and efficiently. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pp. 364–377, 2020.
- [33] Lam, S. K., Pitrou, A., and Seibert, S. Numba: A llvm-based python jit compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, pp. 1–6, 2015.
- [34] Leskovec, J., Lang, K. J., Dasgupta, A., and Mahoney, M. W. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009.
- [35] Leventhal, D. and Lewis, A. S. Randomized methods for linear constraints: convergence rates and conditioning. *Mathematics of Operations Research*, 35(3):641–654, 2010.

- [36] Mahoney, M. W., Orecchia, L., and Vishnoi, N. K. A local spectral method for graphs: With applications to improving graph partitions and exploring data graphs locally. *The Journal of Machine Learning Research (JMLR)*, 13(1):2339–2365, 2012.
- [37] Martínez-Rubio, D., Wirth, E., and Pokutta, S. Accelerated and sparse algorithms for approximate personalized PageRank and beyond. In *Proceedings of Thirty Sixth Conference on Learning Theory (COLT)*, volume 195 of *Proceedings of Machine Learning Research*, pp. 2852–2876. PMLR, 2023.
- [38] Morris, B. and Peres, Y. Evolving sets and mixing. In *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing (STOC)*, pp. 279–286, New York, NY, USA, 2003. Association for Computing Machinery.
- [39] Nutini, J., Schmidt, M., Laradji, I., Friedlander, M., and Koepke, H. Coordinate descent converges faster with the gauss-southwell rule than random selection. In *ICML*, pp. 1632–1641. PMLR, 2015.
- [40] Polyak, B. T. *Introduction to optimization*. New York, Optimization Software., 1987.
- [41] Rakhlin, A. and Sridharan, K. Efficient online multiclass prediction on graphs via surrogate losses. In *Artificial Intelligence and Statistics*, pp. 1403–1411. PMLR, 2017.
- [42] Rubinfeld, R. and Shapira, A. Sublinear time algorithms. *SIAM Journal on Discrete Mathematics*, 25(4):1562–1588, 2011.
- [43] Saad, Y. *Iterative methods for sparse linear systems*. SIAM, 2003.
- [44] Schaub, M. T., Benson, A. R., Horn, P., Lippner, G., and Jadbabaie, A. Random walks on simplicial complexes and the normalized hodge 1-laplacian. *SIAM Review*, 62(2):353–391, 2020.
- [45] Spielman, D. A. Algorithms, graph theory, and linear equations in laplacian matrices. In *Proceedings of the International Congress of Mathematicians 2010 (ICM 2010) (In 4 Volumes) Vol. I: Plenary Lectures and Ceremonies Vols. II–IV: Invited Lectures*, pp. 2698–2722. World Scientific, 2010.
- [46] Spielman, D. A. and Teng, S.-H. 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.
- [47] Stević, S. Bounded solutions to nonhomogeneous linear second-order difference equations. *Symmetry*, 9(10):227, 2017.
- [48] Teng, S.-H. et al. Scalable algorithms for data and network analysis. *Foundations and Trends® in Theoretical Computer Science*, 12(1–2):1–274, 2016.
- [49] Tseng, P. and Yun, S. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117:387–423, 2009.
- [50] Tu, S., Venkataraman, S., Wilson, A. C., Gittens, A., Jordan, M. I., and Recht, B. Breaking locality accelerates block gauss-seidel. In *ICML*, pp. 3482–3491. PMLR, 2017.
- [51] Vishnoi, N. K. et al. $Lx = b$. *Foundations and Trends in Theoretical Computer Science*, 8(1–2): 1–141, 2013.
- [52] Wang, J.-K., Lin, C.-H., and Abernethy, J. D. A modular analysis of provable acceleration via Polyak’s momentum: Training a wide ReLU network and a deep linear network. In *ICML*, pp. 10816–10827. PMLR, 2021.
- [53] Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., and Weinberger, K. Simplifying graph convolutional networks. In *ICML*, pp. 6861–6871. PMLR, 2019.
- [54] Yin, H., Benson, A. R., Leskovec, J., and Gleich, D. F. Local higher-order graph clustering. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining (KDD)*, 2017.

- [55] Young, D. M. *Iterative solution of large linear systems*. Elsevier, 2014.
- [56] Zeng, H., Zhang, M., Xia, Y., Srivastava, A., Malevich, A., Kannan, R., Prasanna, V., Jin, L., and Chen, R. Decoupling the depth and scope of graph neural networks. *Advances in Neural Information Processing Systems*, 34:19665–19679, 2021.
- [57] Zhou, B., Sun, Y., and Harikandeh, R. B. Fast online node labeling for very large graphs. In *ICML*, pp. 42658–42697. PMLR, 2023.

Appendix / supplemental material

A	Notations and Proof of Lemma 2.1	16
A.1	List of Notations	16
A.2	Proof of Lemma 2.1	16
B	Local Iterative Methods via Evolving Set Process	18
B.1	Local Variant of GS-SOR and Proof of Lemma 3.2	18
B.2	LOC-SOR and Proof of Theorem 3.3	20
B.3	Optimal GS-SOR and Proof of Corollary 3.4	22
B.4	LOC-GD and Proof of Theorem 3.5	23
C	Local Chebyshev Method - LOCCH	25
C.1	Nonhomogeneous of Second-order Difference Equation	25
C.2	Properties on Ratio of Chebyshev Polynomials	28
C.3	Standard Chebyshev (CH) Method and Proof of Theorem C.6	29
C.4	Residual Updates of LOCCH and Proof of Lemma 4.1	30
C.5	Convergence of LOCCH and Proof of Theorem 4.2	34
C.6	Implementation of LOCCH	37
D	Local Heavy-Ball Method - LOCHB	37
D.1	Standard HB and Proof Theorem D.2	37
D.2	Residual Updates of LOCHB and Proof of Theorem D.6	40
D.3	Convergence of LOCHB of Proof of Theorem D.8	43
D.4	Implementation of LOCHB	45
E	Instances of Sparse Linear Systems	45
E.1	Table of Popular Graph-induced Linear Systems	45
F	Experimental Details and Missing Results	46
F.1	Datasets and Preprocessing	46
F.2	Problems Settings and Baseline Methods	46
F.3	Full results of Fig. 15 4 5 6	47
F.4	Results on local clustering	52
G	Related work	53

A Notations and Proof of Lemma 2.1

A.1 List of Notations

In the rest of the appendix, we use the following notations:

	Description
\mathcal{G}	An undirected connected simple graph with unit weights.
\mathbf{A}	The adjacency matrix of \mathcal{G} .
\mathbf{D}	The diagonal degree matrix of \mathcal{G} .
\mathbf{W}	The normalized Laplacian matrix $\mathbf{W} \triangleq \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}$.
$\mathbf{V} \mathbf{\Lambda} \mathbf{V}^\top$	The eigendecomposition of \mathbf{W} is given by $\mathbf{W} = \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2} = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^\top$, where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$, with $1 = \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \geq -1$. When $\lambda_n = -1$, \mathcal{G} is a bipartite graph.
\mathbf{Q}	The underlying matrix of Equ. (3) is $\mathbf{Q} = \mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}$.
$\mathbf{r}^{(t)}$	Given an estimate $\mathbf{x}^{(t)}$, the residual is defined as $\mathbf{r}^{(t)} \triangleq \mathbf{b} - \mathbf{Q} \mathbf{x}^{(t)}$.
$\tilde{\mathbf{r}}^{(t)}$	The $\mathbf{D}^{1/2}$ -shifted residual is defined as $\tilde{\mathbf{r}}^{(t)} = \mathbf{D}^{1/2} \mathbf{r}^{(t)}$.
α	The damping factor α , which lies in the interval $(0, 1)$.
$\tilde{\alpha}$	A pre-defined constant $\tilde{\alpha} = (1 - \sqrt{\alpha}) / (1 + \sqrt{\alpha})$.
$\mathcal{T}_{\mathcal{A}}$	The total runtime of a local algorithm \mathcal{A} .
$T_t(x)$	For $t \geq 1$, T_t denotes the Chebyshev polynomial of the first kind, defined as $T_{t+1}(x) = 2xT_t(x) - T_{t-1}(x)$, with $T_0(x) = 1$, and $T_1(x) = x$.
δ_t	The ratio of T_{t-1} and T_t , i.e., $\delta_t = T_{t-1}(\frac{1+\alpha}{1-\alpha}) / T_t(\frac{1+\alpha}{1-\alpha})$, and $\delta_{t+1} = (2\frac{1+\alpha}{1-\alpha} - \delta_t)^{-1}$, with $\delta_1 = \frac{1-\alpha}{1+\alpha}$.
$\delta_{1:t}$	The product of all δ_t , i.e., $\delta_{1:t} \triangleq \prod_{i=1}^t \delta_i$. By default, we set $\delta_{0:1} = 0$.
$\mathcal{S}_{1:t}$	The intersection of all t -ordered sets, i.e., $\mathcal{S}_{1:t} = \mathcal{S}_1 \cap \mathcal{S}_2 \cap \dots \cap \mathcal{S}_t$. By default, we set $\mathcal{S}_{t:t-1} = \mathcal{V}$.
$\bar{\mathcal{S}}_t$	The complement of \mathcal{S}_t , i.e., $\bar{\mathcal{S}}_t = \mathcal{V} \setminus \mathcal{S}_t$.
$\mathcal{S}_{j,t}$	$\mathcal{S}_{j,t} = \mathcal{S}_j \cap \mathcal{S}_{j+1} \cap \dots \cap \mathcal{S}_{t-1} \cap \bar{\mathcal{S}}_t = \mathcal{S}_{j:t-1} \cap \bar{\mathcal{S}}_t$. By default, we set $\mathcal{S}_{t,t} = \bar{\mathcal{S}}_t$.

A.2 Proof of Lemma 2.1

Lemma 2.1 (Runtime bound of APPR [2]). *Given $\alpha \in (0, 1)$ and the precision $\epsilon \leq 1/d_s$ for node $s \in \mathcal{V}$ with $\mathbf{p} \leftarrow \mathbf{0}$, $\mathbf{z} \leftarrow \mathbf{e}_s$ at the initial, APPR($\mathcal{G}, \epsilon, \alpha, s$) defined in (2) returns an estimate \mathbf{p} of $\boldsymbol{\pi}$. There exists a real implementation of (2) (e.g., Algo. 2) such that the runtime $\mathcal{T}_{\text{APPR}}$ satisfies*

$$\mathcal{T}_{\text{APPR}} \leq \Theta(1/(\alpha\epsilon)).$$

Furthermore, the estimate $\hat{\boldsymbol{\pi}} := \mathbf{p}$ satisfies $\|\mathbf{D}^{-1}(\hat{\boldsymbol{\pi}} - \boldsymbol{\pi})\|_\infty \leq \epsilon$ and $\text{vol}(\text{supp}(\hat{\boldsymbol{\pi}})) \leq 2/((1-\alpha)\epsilon)$.

Proof. To find an upper bound of $\mathcal{T}_{\text{APPR}}$, we add a time index for all active nodes u_1, u_2, \dots, u_t processed in APPR of Algo. 2, a real implementation of (2). The parameter t is the number of active nodes processed, and u_i is the node dequeued at time i . So, updates of \mathbf{p} and \mathbf{z} are from $(\mathbf{0}, \mathbf{e}_s) = (\mathbf{p}^{(0)}, \mathbf{z}^{(0)})$ to $(\mathbf{p}^{(t)}, \mathbf{z}^{(t)})$ as follows:

$$(\mathbf{p}^{(0)}, \mathbf{z}^{(0)}) \xrightarrow{u_1} (\mathbf{p}^{(1)}, \mathbf{z}^{(1)}) \xrightarrow{u_2} (\mathbf{p}^{(2)}, \mathbf{z}^{(2)}) \dots \xrightarrow{u_t} (\mathbf{p}^{(t)}, \mathbf{z}^{(t)}).$$

For each active u_i , the updates of (\mathbf{p}, \mathbf{z}) by definition can be represented as

$$\mathbf{p}^{(i)} = \mathbf{p}^{(i-1)} + \alpha z_{u_i}^{(i-1)} \cdot \mathbf{e}_{u_i}, \quad \mathbf{z}^{(i)} = \mathbf{z}^{(i-1)} - \frac{(1+\alpha)z_{u_i}^{(i-1)}}{2} \mathbf{e}_{u_i} + \frac{(1-\alpha)z_{u_i}^{(i-1)}}{2} \mathbf{A} \mathbf{D}^{-1} \mathbf{e}_{u_i}.$$

Since $\mathbf{z}^{(0)} = \mathbf{e}_s \geq \mathbf{0}$, then $\mathbf{z}^{(i)} \geq \mathbf{0}$ and $\mathbf{p}^{(i)} \geq \mathbf{0}$ for all i by induction. Note that $\|\mathbf{A}\mathbf{D}^{-1}\mathbf{e}_{u_i}\|_1 = 1$, we have the following relation from the updates of \mathbf{z}

$$\|\mathbf{z}^{(i)}\|_1 = \|\mathbf{z}^{(i-1)}\|_1 - \alpha z_{u_i}^{(i-1)} \iff z_{u_i}^{(i-1)} = \frac{\|\mathbf{z}^{(i-1)}\|_1 - \|\mathbf{z}^{(i)}\|_1}{\alpha}.$$

Note $\epsilon d_{u_i} \leq z_{u_i}^{(i-1)}$ for each active u_i . Summing the above equation over $i = 1, 2, \dots, t$, we have

$$\sum_{i=1}^t \epsilon d_{u_i} \leq \sum_{i=1}^t z_{u_i}^{(i-1)} = \sum_{i=1}^t \left(\frac{\|\mathbf{z}^{(i-1)}\|_1 - \|\mathbf{z}^{(i)}\|_1}{\alpha} \right) = \frac{\|\mathbf{z}^{(0)}\|_1 - \|\mathbf{z}^{(t)}\|_1}{\alpha} \leq \frac{\|\mathbf{z}^{(0)}\|_1}{\alpha} = \frac{1}{\alpha},$$

where note $\|\mathbf{z}^{(0)}\|_1 = \|\mathbf{e}_s\|_1 = 1$ by the initial condition. Since $\sum_{i=1}^t d_{u_i}$ exactly captures the number of operations needed, the runtime of Algo. 2 is then bounded as

$$\mathcal{T}_{\text{APPR}} = \Theta \left(\sum_{i=1}^t d_{u_i} \right) \leq \Theta \left(\frac{1}{\alpha \epsilon} \right).$$

To check the quality of estimate \mathbf{p} , using the updates of $\mathbf{p}^{(i)}$ and summing over all i , we have

$$\mathbf{p}^{(t)} = \alpha \sum_{i=1}^t z_{u_i}^{(i-1)} \mathbf{e}_{u_i} = \alpha \underbrace{\left(\frac{(1+\alpha)}{2} \mathbf{I} - \frac{(1-\alpha)}{2} \mathbf{A}\mathbf{D}^{-1} \right)^{-1}}_{\mathbf{\Pi}} \sum_{i=1}^t \left(\mathbf{z}^{(i-1)} - \mathbf{z}^{(i)} \right) = \boldsymbol{\pi} - \mathbf{\Pi} \mathbf{z}^{(t)},$$

where $\mathbf{\Pi}$ is the PPR matrix. The above gives us $\boldsymbol{\pi} - \mathbf{p}^{(t)} = \mathbf{\Pi} \cdot \mathbf{z}^{(t)}$. Since \mathcal{G} is undirected, the $\mathbf{\Pi}$ matrix satisfies $\pi_v[u] = (d_u/d_v)\pi_u[v]$ where $\pi_v[u]$ is the u -th element of PPR vector of sourcing node v . Consider each u -th element of $\mathbf{\Pi} \cdot \mathbf{z}^{(t)}$

$$(\mathbf{\Pi} \cdot \mathbf{z}^{(t)})_u = \sum_{v \in \mathcal{V}} z_v^{(t)} \cdot \pi_v[u] = \sum_{v \in \mathcal{V}} z_v^{(t)} \cdot \frac{d_u}{d_v} \pi_u[v] \leq \epsilon d_u \sum_{v \in \mathcal{V}} \pi_u[v] = \epsilon d_u,$$

where the last equality is due to $\sum_{v \in \mathcal{V}} \pi_u[v] = 1$. Hence, $(\boldsymbol{\pi} - \mathbf{p}^{(t)})_u = (\mathbf{\Pi} \cdot \mathbf{z}^{(t)})_u \leq \epsilon d_u$, which indicates $\|\mathbf{D}^{-1}(\mathbf{p}^{(t)} - \boldsymbol{\pi})\|_\infty \leq \epsilon$. To see the bound of $\text{vol}(\text{supp}(\mathbf{p}^{(t)}))$, note for any $u \in \text{supp}(\mathbf{p}^{(t)})$, it was an active node and there was at least $\tilde{z}_u(1-\alpha)/2$ remain in u -th entry of $\mathbf{z}^{(t)}$ where we denote \tilde{z}_u as the residual before the last push operation of node u ; hence

$$\sum_{u \in \text{supp}(\mathbf{p})} d_u \leq \sum_{u \in \text{supp}(\mathbf{p})} \frac{\tilde{z}_u}{\epsilon} = \sum_{u \in \text{supp}(\mathbf{p})} \frac{\tilde{z}_u(1-\alpha)/2}{\epsilon(1-\alpha)/2} \leq \frac{\sum_{u \in \text{supp}(\mathbf{p})} \tilde{z}_u}{\epsilon(1-\alpha)/2} \leq \frac{2}{(1-\alpha)\epsilon}.$$

□

Algo. 2 APPR($\alpha, \epsilon, s, \mathcal{G}$) via FIFO Queue

```

1: Initialize:  $\mathbf{p} \leftarrow \mathbf{0}, \mathbf{z} \leftarrow \mathbf{e}_s, \mathcal{Q} \leftarrow \{*, s\}, t = -1$ 
2: while true do
3:    $u \leftarrow \mathcal{Q}.\text{dequeue}()$ 
4:   if  $u == *$  then
5:     if  $\mathcal{Q} = \emptyset$  then
6:       break
7:      $t \leftarrow t + 1$  // Starting time of  $\mathcal{S}_t$ 
8:      $\mathcal{Q}.\text{enqueue}(*)$  // Marker for next  $\mathcal{S}_{t+1}$ 
9:     continue
10:   $\tilde{z} \leftarrow z_u$ 
11:   $p_u \leftarrow p_u + \alpha \cdot \tilde{z}$ 
12:   $z_u \leftarrow \tilde{z} \cdot (1 - \alpha)/2$ 
13:  for  $v \in \mathcal{N}(u)$  do
14:     $z_v \leftarrow z_v + \frac{(1-\alpha)}{2} \cdot \frac{\tilde{z}}{d_u}$ 
15:    if  $z_v \geq \epsilon d_v$  and  $v \notin \mathcal{Q}$  then
16:       $\mathcal{Q}.\text{enqueue}(v)$ 
17:    if  $z_u \geq \epsilon d_u$  and  $u \notin \mathcal{Q}$  then
18:       $\mathcal{Q}.\text{enqueue}(u)$ 
19: return  $\mathbf{p}$ 

```

Indeed, Lemma 2.1 is a special case of Theorem 1 in [2]. The proof outlined above adheres to the key strategy demonstrated in that theorem, which involves exploring the monotonicity and nonnegativity of \mathbf{z} .

The real implementation of APPR, as shown in Algo. 2, presents a typical queue-based method. It has monotonic properties during the updates of \mathbf{p} and \mathbf{z} (Lines 10-16 of Algo. 2). It also holds element-wise that $\mathbf{p} \geq \mathbf{0}$ and $\mathbf{z} \geq \mathbf{0}$. The operations of $\mathcal{Q}.\text{enqueue}(u)$, $\mathcal{Q}.\text{dequeue}()$, and $v \notin \mathcal{Q}$ are all in $\mathcal{O}(1)$. Line 4 to Line 9 is to design the marker for distinguishing between \mathcal{S}_t and \mathcal{S}_{t+1} . If all active nodes are processed and no more active nodes are added into \mathcal{Q} , then \mathcal{Q} will be empty, and finally, the algorithm returns an estimate \mathbf{p} of $\boldsymbol{\pi}$.

B Local Iterative Methods via Evolving Set Process

Justification of an equivalent condition. We make the justification of an equivalent stop condition for solving (3). Note we require a local solver to return an estimate $\hat{\pi}$ satisfies

$$\|D^{-1}(\hat{\pi} - \pi)\|_{\infty} \leq \epsilon \quad (14)$$

Since we define $Qx = b$ as $(I - \frac{1-\alpha}{1+\alpha}D^{-1/2}AD^{-1/2})x = \frac{2\alpha}{1+\alpha}D^{-1/2}e_s$. With $r^{(t)} = b - Qx^{(t)}$ and the stop condition

$$\|D^{-1/2}r^{(t)}\|_{\infty} \leq \frac{2\alpha\epsilon}{1+\alpha}$$

ensures the estimate $\hat{\pi} = D^{1/2}x^{(t)}$ satisfies (14). To see this, since $\pi = D^{1/2}x^*$, we have

$$\begin{aligned} \|D^{-1}(\hat{\pi} - \pi)\|_{\infty} &= \|D^{-1/2}(x^{(t)} - x^*)\|_{\infty} \\ &= \|D^{-1/2}(Q^{-1}b - Q^{-1}r^{(t)} - Q^{-1}b)\|_{\infty} \\ &= \|D^{-1/2}Q^{-1}D^{1/2}D^{-1/2}r^{(t)}\|_{\infty} \\ &\leq \|D^{-1/2}Q^{-1}D^{1/2}\|_{\infty} \cdot \|D^{-1/2}r^{(t)}\|_{\infty} \\ &\leq \|D^{-1/2}Q^{-1}D^{1/2}\|_{\infty} \cdot \frac{2\alpha\epsilon}{1+\alpha} \end{aligned}$$

where $D^{-1/2}Q^{-1}D^{1/2} = (I - \frac{1-\alpha}{1+\alpha}D^{-1}A)^{-1} = \sum_{i=0}^{\infty} (\frac{1-\alpha}{1+\alpha}D^{-1}A)^i$. This leads to

$$\begin{aligned} \|D^{-1}(\hat{\pi} - \pi)\|_{\infty} &\leq \left\| \sum_{i=0}^{\infty} (\frac{1-\alpha}{1+\alpha}D^{-1}A)^i \right\|_{\infty} \cdot \frac{2\alpha\epsilon}{1+\alpha} \\ &\leq \frac{1+\alpha}{2\alpha} \cdot \frac{2\alpha\epsilon}{1+\alpha} \\ &= \epsilon. \end{aligned}$$

B.1 Local Variant of GS-SOR and Proof of Lemma 3.2

The Gauss-Seidel Successive Over-Relaxation (GS-SOR) solver (see Section 11.2.7 of Golub & Van Loan [19]) for the linear system $M\pi = s$ via the following forward substitution

for i **in** $\mathcal{V} := \{1, 2, \dots, n\}$ **do** :

$$p_i^{(t+1)} = \omega \left(s_i - \sum_{j=1}^{i-1} M_{ij}p_j^{(t+1)} - \sum_{j=i+1}^n M_{ij}p_j^{(t)} \right) / M_{ii} + (1 - \omega)p_i^{(t)},$$

where p is updated from $p^{(t)}$ to $p^{(t+1)}$. When the relaxation parameter $\omega = 1$, GS-SOR reduces to the standard GS method. Equivalently, let $\Delta_i = (i - 1)/n$ for $i = 1, 2, \dots, n$, then GS-SOR updates can be sequentially represented as

for i **in** $\mathcal{V} := \{1, 2, \dots, n\}$ **do** :

$$p^{(t+\Delta_{i+1})} \leftarrow p^{(t+\Delta_i)} + \frac{\omega}{M_{ii}} \left(s_i - \sum_{j=1}^{i-1} M_{ij}p_j^{(t+\Delta_i)} - \sum_{j=i}^n M_{ij}p_j^{(t+\Delta_i)} \right) \cdot e_i.$$

Therefore, it is natural to define the following local variant of GS-SOR.

Definition B.1 (Local variant of GS-SOR). Consider the linear system $M\pi = s$. For $t \geq 0$, we are given an active node set $\mathcal{S}_t = \{u_1, u_2, \dots, u_{|\mathcal{S}_t|}\}$ and let $\Delta_i = (i - 1)/|\mathcal{S}_t|$ for $i = 1, 2, \dots, |\mathcal{S}_t|$, providing $\omega \in (0, 2)$, it is natural to define the *local variant* of GS-SOR as follows:

for u_i **in** $\mathcal{S}_t := \{u_1, u_2, \dots, u_{|\mathcal{S}_t|}\}$ **do** :

$$p^{(t+\Delta_{i+1})} \leftarrow p^{(t+\Delta_i)} + \frac{\omega}{M_{u_i u_i}} \left(s_{u_i} - \sum_{j=1}^{i-1} M_{u_i u_j} p_{u_j}^{(t+\Delta_i)} - \sum_{j=i}^n M_{u_i u_j} p_{u_j}^{(t+\Delta_i)} \right) \cdot e_{u_i}, \quad (15)$$

where $\mathcal{S}_t \subseteq \mathcal{V}$. When $\omega = 1$ and $\mathcal{S}_t = \mathcal{V} = \{1, 2, \dots, n\}$, it reduces to the standard GS.

We use the above definition to show APPR is a local variant of GS-SOR as the following.

Lemma 3.2 (New local evolving-based bound for APPR). *Let $\mathbf{M} = \alpha^{-1}(\mathbf{I} - \frac{1-\alpha}{2}(\mathbf{I} + \mathbf{A}\mathbf{D}^{-1}))$ and $\mathbf{s} = \mathbf{e}_s$. The linear system $\mathbf{M}\boldsymbol{\pi} = \mathbf{s}$ is equivalent to Equ. (1). Given $\mathbf{p}^{(0)} = \mathbf{0}$, $\mathbf{z}^{(0)} = \mathbf{e}_s$ with $\omega \in (0, 2)$, the local variant of GS-SOR (15) for $\mathbf{M}\boldsymbol{\pi} = \mathbf{s}$ can be formulated as*

$$\mathbf{p}^{(t+\Delta_{i+1})} \leftarrow \mathbf{p}^{(t+\Delta_i)} + \frac{\omega z_{u_i}^{(t+\Delta_i)}}{M_{u_i u_i}} \mathbf{e}_{u_i}, \quad \mathbf{z}^{(t+\Delta_{i+1})} \leftarrow \mathbf{z}^{(t+\Delta_i)} - \frac{\omega z_{u_i}^{(t+\Delta_i)}}{M_{u_i u_i}} \mathbf{M} \mathbf{e}_{u_i},$$

where u_i is an active node in \mathcal{S}_t satisfying $z_{u_i} \geq \epsilon d_{u_i}$ and $\Delta_i = (i-1)/|\mathcal{S}_t|$. Furthermore, when $\omega = \frac{1+\alpha}{2}$, this method reduces to APPR given in (7), and there exists a real implementation (Aglo. 2) of APPR such that the runtime $\mathcal{T}_{\text{APPR}}$ is bounded by

$$\mathcal{T}_{\text{APPR}} \leq \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\alpha \hat{\gamma}_T} \ln \frac{C_T}{\epsilon}, \quad \text{where } \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\hat{\gamma}_T} \leq \frac{1}{\epsilon}, \quad C_T = \frac{2}{(1-\alpha)|\mathcal{I}_T|}, \quad \hat{\gamma}_T \triangleq \frac{1}{T} \sum_{t=0}^{T-1} \left\{ \frac{\sum_{i=1}^{|\mathcal{S}_t|} |z_{u_i}^{(t+\Delta_i)}|}{\|\mathbf{z}^{(t)}\|_1} \right\}.$$

Proof. Note that $\mathbf{M}\boldsymbol{\pi} = \mathbf{s}$ is equivalent to Equ. (1). We first rewrite $\mathbf{p}^{(t+\Delta_{i+1})}$ in terms of the residual \mathbf{z} . The residual \mathbf{z} at time $t + \Delta_i$ can be written as $\mathbf{z}^{(t+\Delta_i)} = \mathbf{s} - \mathbf{M}\mathbf{p}^{(t+\Delta_i)}$. Note $s_{u_i} - \sum_{j=1}^{i-1} M_{u_i u_j} p_{u_j}^{(t+\Delta_i)} - \sum_{j=i}^n M_{u_i u_j} p_{u_j}^{(t+\Delta_i)} = (\mathbf{s} - \mathbf{M}\mathbf{p}^{(t+\Delta_i)})_{u_i} = z_{u_i}^{(t+\Delta_i)}$. Then, the updates of local GS-SOR defined in (15) can be rewritten as $\mathbf{p}^{(t+\Delta_{i+1})} = \mathbf{p}^{(t+\Delta_i)} + \frac{\omega}{M_{u_i u_i}} z_{u_i}^{(t+\Delta_i)} \cdot \mathbf{e}_{u_i}$. Hence, the updates of $\mathbf{z}^{(t+\Delta_i)}$ can be written as

$$\mathbf{z}^{(t+\Delta_{i+1})} = \mathbf{s} - \mathbf{M}\mathbf{p}^{(t+\Delta_{i+1})} = \mathbf{s} - \mathbf{M} \left(\mathbf{p}^{(t+\Delta_i)} + \frac{\omega z_{u_i}^{(t+\Delta_i)}}{M_{u_i u_i}} \cdot \mathbf{e}_{u_i} \right) = \mathbf{z}^{(t+\Delta_i)} - \frac{\omega z_{u_i}^{(t+\Delta_i)}}{M_{u_i u_i}} \mathbf{M} \mathbf{e}_{u_i}.$$

Note the diagonal element $M_{u_i u_i} = (1+\alpha)/(2\alpha)$. Hence, when $\omega = \frac{1+\alpha}{2}$, we have

$$\begin{aligned} \mathbf{p}^{(t+\Delta_{i+1})} &= \mathbf{p}^{(t+\Delta_i)} + \alpha z_{u_i}^{(t+\Delta_i)} \cdot \mathbf{e}_{u_i} \\ \mathbf{z}^{(t+\Delta_{i+1})} &= \mathbf{z}^{(t+\Delta_i)} - \alpha z_{u_i}^{(t+\Delta_i)} \mathbf{M} \mathbf{e}_{u_i} = \mathbf{z}^{(t+\Delta_i)} - z_{u_i}^{(t+\Delta_i)} \left(\frac{1+\alpha}{2} \mathbf{I} - \frac{1-\alpha}{2} \mathbf{A}\mathbf{D}^{-1} \right) \mathbf{e}_{u_i}. \end{aligned}$$

The above updates match APPR's evolving set process formulation in (7). The rest is to show a new runtime bound. Adding ℓ_1 -norm on both sides of the above equation, then note $\|\mathbf{z}^{(t+\Delta_{i+1})}\|_1 = \|\mathbf{z}^{(t+\Delta_i)}\|_1 - \alpha z_{u_i}^{(t+\Delta_i)}$ for $i = 1, 2, \dots, |\mathcal{S}_t|$. We have

$$\|\mathbf{z}^{(t+1)}\|_1 = \left(1 - \frac{\alpha \sum_{i=1}^{|\mathcal{S}_t|} z_{u_i}^{(t+\Delta_i)}}{\|\mathbf{z}^{(t)}\|_1} \right) \|\mathbf{z}^{(t)}\|_1 = (1 - \alpha \beta_t) \|\mathbf{z}^{(t)}\|_1 = \prod_{i=0}^t (1 - \alpha \beta_t) \|\mathbf{z}^{(0)}\|_1,$$

where we define $\beta_t := \frac{\sum_{i=1}^{|\mathcal{S}_t|} |z_{u_i}^{(t+\Delta_i)}|}{\|\mathbf{z}^{(t)}\|_1}$. Let $t = T-1$, we have

$$\ln \frac{\|\mathbf{z}^{(T)}\|_1}{\|\mathbf{z}^{(0)}\|_1} = \sum_{t=0}^{T-1} \ln(1 - \alpha \beta_t) \leq - \sum_{t=0}^{T-1} \alpha \beta_t \quad \Rightarrow \quad T \leq \frac{1}{\alpha \hat{\gamma}_T} \ln \frac{\|\mathbf{z}^{(0)}\|_1}{\|\mathbf{z}^{(T)}\|_1},$$

where the first inequality is due to $\ln(1+x) \leq x$ for $x > -1$. For each nonzero node $u \in \mathcal{I}_T = \{z_u^{(T)} : z_u^{(T)} \neq 0, u \in \mathcal{V}\}$, consider the last time t' that it was altered. Then, either the alteration came from u being an active node, with $z_u^{(t')} \geq d_u \epsilon$, and after the PUSH operation it became $z_u^{(t'')} \geq \frac{(1-\alpha)d_u}{2} \epsilon$; or the alteration came from a neighboring node $v_u \in \mathcal{N}(u)$ pushing its mass onto u , which ensures that $z_u^{(t'')} \geq \frac{(1-\alpha)}{2d_{v_u}} z_{v_u}^{(t')} \geq \frac{(1-\alpha)}{2} \epsilon$. These two cases provide a lower bound of $\frac{1-\alpha}{2} \epsilon$. Hence, $\|\mathbf{z}^{(T)}\|_1 \geq \frac{\epsilon(1-\alpha)|\mathcal{I}_T|}{2}$, which leads to the corresponding constant C_T .

To see the lower bound of $1/\epsilon$, note $\epsilon d_{u_i} \leq z_{u_i}^{(t+\Delta_i)}$ for all $i = 1, 2, \dots, |\mathcal{S}_t|$. Then we have

$$\begin{aligned} \epsilon \text{vol}(\mathcal{S}_t) &\leq \sum_{i=1}^{|\mathcal{S}_t|} z_{u_i}^{(t+\Delta_i)} \\ &= \beta_t \|\mathbf{z}^{(t)}\|_1 \\ &\leq \beta_t, \end{aligned}$$

where we defined $\beta_t := \frac{\sum_{i=1}^{|\mathcal{S}_t|} |z_{u_i}^{(t+\Delta_i)}|}{\|\mathbf{z}^{(t)}\|_1}$ and the last inequality is due to the monotonic decreasing of $\|\mathbf{z}^{(t)}\|_1$, i.e., $1 \geq \|\mathbf{z}^{(0)}\|_1 \geq \dots \geq \|\mathbf{z}^{(T)}\|_1$. Applying the above inequality for all $t = 0, 1, 2, \dots, T-1$, it leads to

$$\begin{aligned} \epsilon \operatorname{vol}(\mathcal{S}_t) &\leq \beta_t \\ \Rightarrow \epsilon \sum_{t=0}^{T-1} \operatorname{vol}(\mathcal{S}_t) &\leq \sum_{t=0}^{T-1} \beta_t \\ \Rightarrow \frac{\overline{\operatorname{vol}}(\mathcal{S}_T)}{\hat{\gamma}_T} &\leq \frac{1}{\epsilon}, \end{aligned}$$

where the last derivation is from the fact that $\hat{\gamma}_T = \frac{1}{T} \left\{ \sum_{t=0}^{T-1} \beta_t := \frac{\sum_{i=1}^{|\mathcal{S}_t|} |z_{u_i}^{(t+\Delta_i)}|}{\|\mathbf{z}^{(t)}\|_1} \right\}$. \square

Remark B.2. The connection between APPR and the Gauss-Seidel is not new [28, 29, 16, 9]. Our work is the first work that has linked APPR and the Gauss-Seidel with a locally evolving set process.

B.2 LOCSOR and Proof of Theorem 3.3

In this subsection, recall we defined $\tilde{\mathbf{r}}^{(t)} = \mathbf{D}^{1/2} \mathbf{r}^{(t)}$.

Lemma B.3 (Local iteration complexity of LOCSOR ($\omega \leq 1$)). *Denote $\mathcal{S}_t = \{u_1, u_2, \dots, u_{|\mathcal{S}_t}|\}$ as the active node set at the t -th iteration. When $\omega \in (0, 1]$, all vectors $\tilde{\mathbf{r}}^{(t)} \geq 0$ are nonnegative and magnitudes are decreasing $\|\tilde{\mathbf{r}}^{(t+1)}\|_1 < \|\tilde{\mathbf{r}}^{(t)}\|_1$. Let T be the total number of iterations needed. Then, at iteration T , we have*

$$T \in \frac{(1+\alpha)}{2\alpha\omega\bar{\gamma}_T} \left[1 - \frac{\|\tilde{\mathbf{r}}^{(T)}\|_1}{\|\tilde{\mathbf{r}}^{(0)}\|_1}, \ln \frac{\|\tilde{\mathbf{r}}^{(0)}\|_1}{\|\tilde{\mathbf{r}}^{(T)}\|_1} \right], \quad \bar{\gamma}_T \triangleq \frac{1}{T} \sum_{t=0}^{T-1} \left\{ \gamma_t \triangleq \sum_{i=1}^{|\mathcal{S}_t|} \frac{\tilde{r}_{u_i}^{(t+\Delta_i)}}{\|\tilde{\mathbf{r}}^{(t)}\|_1} \right\}, \quad (16)$$

where $\bar{\gamma}_t = t^{-1} \sum_{\tau=0}^{t-1} \gamma_\tau$ is the mean of active ratio factors defined in Equ. (6).

Proof. Recall $u_i \in \mathcal{S}_t = \{u_1, \dots, u_{|\mathcal{S}_t}|\}$ and $\Delta_i = \frac{i-1}{|\mathcal{S}_t|}$, LOCSOR in Algo. 3 updates

$$\begin{aligned} \mathbf{x}^{(t+\Delta_{i+1})} &= \mathbf{x}^{(t+\Delta_i)} + \omega r_{u_i}^{(t+\Delta_i)} \cdot \mathbf{e}_{u_i} \\ \mathbf{r}^{(t+\Delta_{i+1})} &= \mathbf{r}^{(t+\Delta_i)} - \omega r_{u_i}^{(t+\Delta_i)} \cdot \mathbf{e}_{u_i} + \frac{(1-\alpha)\omega}{1+\alpha} r_{u_i}^{(t+\Delta_i)} \cdot \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2} \mathbf{e}_{u_i} \end{aligned}$$

Note $\mathbf{r} \geq \mathbf{0}$ during updates when $\omega \in (0, 1]$ and recall $\tilde{\mathbf{r}}^{(t)} = \mathbf{D}^{1/2} \mathbf{r}^{(t)}$, we have

$$\begin{aligned} \|\tilde{\mathbf{r}}^{(t+\Delta_{i+1})} + \omega \tilde{r}_{u_i}^{(t+\Delta_i)} \cdot \mathbf{e}_{u_i}\|_1 &= \|\tilde{\mathbf{r}}^{(t+\Delta_i)} + \frac{(1-\alpha)\omega}{1+\alpha} \tilde{r}_{u_i}^{(t+\Delta_i)} \cdot \mathbf{A} \mathbf{D}^{-1} \mathbf{e}_{u_i}\|_1 \\ \|\tilde{\mathbf{r}}^{(t+\Delta_{i+1})}\|_1 + \omega \tilde{r}_{u_i}^{(t+\Delta_i)} &= \|\tilde{\mathbf{r}}^{(t+\Delta_i)}\|_1 + \frac{(1-\alpha)\omega}{1+\alpha} \tilde{r}_{u_i}^{(t+\Delta_i)}. \end{aligned}$$

Summing over the above equations over u_i , we have

$$\|\tilde{\mathbf{r}}^{(t+1)}\|_1 = \|\tilde{\mathbf{r}}^{(t)}\|_1 - \frac{2\alpha\omega}{1+\alpha} \sum_{i=1}^{|\mathcal{S}_t|} \tilde{r}_{u_i}^{(t+\Delta_i)} = \left(1 - \frac{2\alpha\omega}{1+\alpha} \underbrace{\sum_{i=1}^{|\mathcal{S}_t|} \frac{\tilde{r}_{u_i}^{(t+\Delta_i)}}{\|\tilde{\mathbf{r}}^{(t)}\|_1}}_{\gamma_t} \right) \|\tilde{\mathbf{r}}^{(t)}\|_1. \quad (17)$$

Given $\{x_i\}_{i=0}^{T-1}$ and $x_i \in (0, 1)$, the Weierstrass product inequality provides $1 - \sum_{i=0}^{T-1} x_i \leq \prod_{i=0}^{T-1} (1 - x_i)$. By using this inequality, we continue to have a lower bound of T as the following

$$1 - \sum_{t=0}^{T-1} \frac{2\alpha\omega\gamma_t}{1+\alpha} \leq \prod_{t=0}^{T-1} \left(1 - \frac{2\alpha\omega\gamma_t}{1+\alpha} \right) = \frac{\|\tilde{\mathbf{r}}^{(T)}\|_1}{\|\tilde{\mathbf{r}}^{(0)}\|_1} \Rightarrow \frac{(1+\alpha)(1 - \|\tilde{\mathbf{r}}^{(T)}\|_1/\|\tilde{\mathbf{r}}^{(0)}\|_1)}{2\alpha\omega\bar{\gamma}_T} \leq T.$$

To get upper bound of γ_t , note each active residual $\tilde{r}_{u_i}^{(t+\Delta_i)}$ pushes at most $\frac{(1-\alpha)\omega}{(1+\alpha)}$ times magnitude to $\tilde{\mathbf{r}}_{u_{i+1}}$, $\tilde{\mathbf{r}}_{u_{i+2}}$, and $\tilde{\mathbf{r}}_{u_{|\mathcal{S}_t}|}$; hence, $\sum_{j=i}^{|\mathcal{S}_t|} \tilde{r}_{u_j}^{(t+\Delta_j)}$ will increase by at most $\tilde{r}_{u_i}^{(t+\Delta_i)} \cdot \frac{(1-\alpha)\omega}{(1+\alpha)} \leq \tilde{r}_{u_i}^{(t+\Delta_i)}$ in total. Hence, overall u_i , we have

$$\|\tilde{\mathbf{r}}_{\mathcal{S}_t}^{(t)}\|_1 = \sum_{i=1}^{|\mathcal{S}_t|} \tilde{r}_{u_i}^{(t)} \leq \sum_{i=1}^{|\mathcal{S}_t|} \tilde{r}_{u_i}^{(t+\Delta_i)} \leq 2\|\tilde{\mathbf{r}}_{\mathcal{S}_t}^{(t)}\|_1.$$

We reach the following lower and upper bounds of γ_t , $\frac{\|\tilde{\mathbf{r}}_{\mathcal{S}_t}^{(t)}\|_1}{\|\tilde{\mathbf{r}}^{(t)}\|_1} \leq \gamma_t := \sum_{i=1}^{|\mathcal{S}_t|} \frac{\tilde{r}_{u_i}^{(t+\Delta_i)}}{\|\tilde{\mathbf{r}}^{(t)}\|_1} \leq \frac{2\|\tilde{\mathbf{r}}_{\mathcal{S}_t}^{(t)}\|_1}{\|\tilde{\mathbf{r}}^{(t)}\|_1}$. To check the upper bound of T , from Equ. (17), $\|\tilde{\mathbf{r}}^{(T)}\|_1 = \prod_{t=0}^{T-1} \left(1 - \frac{2\alpha\omega\gamma_t}{1+\alpha}\right) \|\tilde{\mathbf{r}}^{(0)}\|_1$ and

$$\ln \frac{\|\tilde{\mathbf{r}}^{(T)}\|_1}{\|\tilde{\mathbf{r}}^{(0)}\|_1} = \sum_{t=0}^{T-1} \ln \left(1 - \frac{2\alpha\omega\gamma_t}{1+\alpha}\right) \leq - \sum_{t=0}^{T-1} \frac{2\alpha\omega\gamma_t}{1+\alpha} \Rightarrow T \leq \frac{(1+\alpha)}{2\alpha\omega\bar{\gamma}_T} \ln \frac{\|\tilde{\mathbf{r}}^{(0)}\|_1}{\|\tilde{\mathbf{r}}^{(T)}\|_1},$$

where the first inequality is due to $\ln(1+x) \leq x$ for $x > -1$. \square

Theorem 3.3 (Runtime bound of LOCSOR ($\omega = 1$)). *Given the configuration $\theta = (\alpha, \epsilon, s, \mathcal{G})$ with $\alpha \in (0, 1)$ and $\epsilon \leq 1/d_s$ and let $\mathbf{r}^{(T)}$ and $\mathbf{x}^{(T)}$ be returned by LOCSOR defined in (8) for solving Equ. (3). There exists a real implementation of (8) such that the runtime $\mathcal{T}_{\text{LocSOR}}$ is bounded by*

$$\frac{1+\alpha}{2} \cdot \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\alpha\bar{\gamma}_T} \left(1 - \frac{\|\mathbf{D}^{1/2}\mathbf{r}^{(T)}\|_1}{\|\mathbf{D}^{1/2}\mathbf{r}^{(0)}\|_1}\right) \leq \mathcal{T}_{\text{LocSOR}} \leq \frac{1+\alpha}{2} \cdot \min \left\{ \frac{1}{\alpha\epsilon}, \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\alpha\bar{\gamma}_T} \ln \frac{C}{\epsilon} \right\}$$

where $\overline{\text{vol}}(\mathcal{S}_T)$ and $\bar{\gamma}_T$ are defined in (6) and $C = \frac{1+\alpha}{(1-\alpha)^{|\mathcal{I}_T|}}$ with $\mathcal{I}_T = \text{supp}(\mathbf{r}^{(T)})$. Furthermore, $\overline{\text{vol}}(\mathcal{S}_T)/\bar{\gamma}_T \leq 1/\epsilon$ and the local estimate $\hat{\boldsymbol{\pi}} := \mathbf{D}^{1/2}\mathbf{x}^{(T)}$ satisfies $\|\mathbf{D}^{-1}(\hat{\boldsymbol{\pi}} - \boldsymbol{\pi})\|_\infty \leq \epsilon$.

Proof. After the last iteration T , for each nonzero residual $\tilde{r}_u^{(T)} \neq 0, u \in \mathcal{I}_T$, there must be at least one update that happened at node u : Node u has a neighbor $v_u \in \mathcal{N}(u)$, which was active. This neighbor v_u pushed some residual $\frac{(1-\alpha)\tilde{r}_{v_u}^{(t')}}{(1+\alpha)d_{v_u}}$ to u where $t' < T$. Hence, for all $u \in \mathcal{I}_T$, we have

$$\|\tilde{\mathbf{r}}^{(T)}\|_1 = \sum_{u \in \mathcal{I}_T} \tilde{r}_u^{(T)} \geq \sum_{u \in \mathcal{I}_T} \frac{(1-\alpha)\tilde{r}_{v_u}^{(t')}}{(1+\alpha)d_{v_u}} \geq \sum_{u \in \mathcal{I}_T} \frac{(1-\alpha)2\alpha\epsilon d_{v_u}/(1+\alpha)}{(1+\alpha)d_{v_u}} = \epsilon|\mathcal{I}_T| \frac{2\alpha(1-\alpha)}{(1+\alpha)^2},$$

where the second inequality is because $\tilde{r}_{v_u}^{(t')}$ was active before the push operation. Applying the above lower bound of $\|\tilde{\mathbf{r}}^{(T)}\|_1$ to Equ. (16) of Lemma B.3 and note $\|\tilde{\mathbf{r}}^{(0)}\|_1 = 2\alpha/(1+\alpha)$, we obtain

$$\frac{\|\tilde{\mathbf{r}}^{(0)}\|_1}{\|\tilde{\mathbf{r}}^{(T)}\|_1} \leq \frac{\|\tilde{\mathbf{r}}^{(0)}\|_1}{\epsilon|\mathcal{I}_T| \cdot \frac{2\alpha(1-\alpha)}{(1+\alpha)^2}} = \frac{1+\alpha}{\epsilon(1-\alpha)|\mathcal{I}_T|} := \frac{C_1}{\epsilon}.$$

The rest is to prove an upper bound $1/(\alpha\epsilon)$ of $\mathcal{T}_{\text{LocSOR}}$. Recall that for any active node u , we have residual updates from Algo. 3 as the following

$$\mathbf{D}^{1/2}\mathbf{r}^{(t+1)} = \mathbf{D}^{1/2}\mathbf{r}^{(t)} - \omega\mathbf{r}_u^{(t)}\mathbf{D}^{1/2}\mathbf{e}_u + \frac{(1-\alpha)\omega r_u^{(t)}}{1+\alpha}\mathbf{A}\mathbf{D}^{-1}\mathbf{D}^{1/2}\mathbf{e}_u.$$

Move $-\omega r_u^{(t)}\mathbf{D}^{1/2}\mathbf{e}_u$ to the left and note $\|\mathbf{A}\mathbf{D}^{-1}\mathbf{D}^{1/2}\mathbf{e}_u\|_1 = \sqrt{d_u}$, we then obtain

$$\|\mathbf{D}^{1/2}\mathbf{r}^{(t+1)}\|_1 + \omega\sqrt{d_u}r_u^{(t)} = \|\mathbf{D}^{1/2}\mathbf{r}^{(t)}\|_1 + \frac{(1-\alpha)\omega}{1+\alpha}\sqrt{d_u}r_u^{(t)}.$$

Hence, for each active u , we have $\frac{2\alpha\omega\sqrt{d_u}r_u^{(t)}}{1+\alpha} = \|\mathbf{D}^{1/2}\mathbf{r}^{(t)}\|_1 - \|\mathbf{D}^{1/2}\mathbf{r}^{(t+1)}\|_1$. Summing them over all active nodes u and noticing $r_u^{(t)} \geq 2\alpha\epsilon\sqrt{d_u}/(1+\alpha)$ by the active condition. Note $\omega = 1$ and $\|\mathbf{D}^{1/2}\mathbf{r}^{(0)}\|_1 = \frac{2\alpha}{1+\alpha}$, we have run time bounded by

$$\mathcal{T}_{\text{LocSOR}} = \sum_u d_u \leq \left(\frac{1+\alpha}{2\alpha}\right)^2 \frac{\sum_t (\|\mathbf{D}^{1/2}\mathbf{r}^{(t)}\|_1 - \|\mathbf{D}^{1/2}\mathbf{r}^{(t+1)}\|_1)}{\omega\epsilon} \leq \frac{(1+\alpha)}{2\alpha\epsilon}.$$

Combining the above bound and the bound T shown in Lemma B.3, we prove the lower and upper bound of $\mathcal{T}_{\text{LocSOR}}$. To check the lower bound of $1/\epsilon$, i.e., $\overline{\text{vol}}(\mathcal{S}_T)/\bar{\gamma}_T \leq 1/\epsilon$, note $\frac{2\alpha\epsilon d_{u_i}}{1+\alpha} \leq \tilde{r}_{u_i}^{(t+\Delta_i)}$ for all $i = 1, 2, \dots, |\mathcal{S}_t|$. Then we have

$$\begin{aligned} \frac{2\alpha\epsilon}{1+\alpha} \text{vol}(\mathcal{S}_t) &\leq \sum_{i=1}^{|\mathcal{S}_t|} \tilde{r}_{u_i}^{(t+\Delta_i)} \\ &= \gamma_t \|\mathbf{D}^{1/2}\mathbf{r}^{(t)}\|_1 \\ &\leq \gamma_t \|\mathbf{D}^{1/2}\mathbf{r}^{(0)}\|_1 = \frac{2\alpha\gamma_t}{1+\alpha}, \end{aligned}$$

where the last inequality is due to the monotonic decreasing of $\|D^{1/2}\mathbf{r}^{(t)}\|_1$, i.e., $\frac{2\alpha}{1+\alpha} \geq \|D^{1/2}\mathbf{r}^{(0)}\|_1 \geq \dots \geq \|\mathbf{r}^{(T)}\|_1$. Applying the above inequality over all $t = 0, 1, 2, \dots, T-1$, it leads to

$$\begin{aligned} \epsilon \text{vol}(\mathcal{S}_t) &\leq \gamma_t \\ \Rightarrow \epsilon \sum_{t=0}^{T-1} \text{vol}(\mathcal{S}_t) &\leq \sum_{t=0}^{T-1} \gamma_t \\ \Rightarrow \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\bar{\gamma}_T} &\leq \frac{1}{\epsilon}. \end{aligned}$$

□

Algo. 3 **LocSOR**($\alpha, \epsilon, s, \mathcal{G}, \omega$) via FIFO Queue

```

1: Initialize:  $\mathbf{r} \leftarrow c\mathbf{e}_s$ ,  $\mathbf{x} \leftarrow \mathbf{0}$ ,  $c = \frac{2\alpha}{1+\alpha}$ ,  $t = -1$ 
2:  $\mathcal{Q} \leftarrow \{*, s\}$  // As we assume  $\epsilon \leq 1/d_s$ 
3: while true do
4:    $u \leftarrow \mathcal{Q}.\text{dequeue}()$ 
5:   if  $u == *$  then
6:     if  $\mathcal{Q} = \emptyset$  then
7:       break
8:      $t \leftarrow t + 1$  // Starting time of  $\mathcal{S}_t$ 
9:      $\mathcal{Q}.\text{enqueue}(*)$  // Marker for next  $\mathcal{S}_{t+1}$ 
10:    continue
11:     $\tilde{r} \leftarrow r_u$ 
12:    if  $|r_u| < c \cdot \epsilon d_u$  then
13:      continue
14:       $x_u \leftarrow x_u + \omega \cdot \tilde{r}$ 
15:       $r_u \leftarrow r_u - \omega \cdot \tilde{r}$ 
16:      for  $v \in \mathcal{N}(u)$  do
17:         $r_v \leftarrow r_v + \frac{(1-\alpha)\omega}{(1+\alpha)} \cdot \frac{\tilde{r}}{d_u}$ 
18:        if  $|r_v| \geq c \cdot \epsilon d_v$  and  $v \notin \mathcal{Q}$  then
19:           $\mathcal{Q}.\text{enqueue}(v)$ 
20:        if  $|r_u| \geq c \cdot \epsilon d_u$  and  $u \notin \mathcal{Q}$  then
21:           $\mathcal{Q}.\text{enqueue}(u)$ 
22: return  $\mathbf{x}$ 

```

The real queue-based implementation of LOC-SOR is presented in Algo. 3. It has monotonic and nonnegative properties during the updates of $\mathbf{r} \geq \mathbf{0}$ and $\mathbf{x} \geq \mathbf{0}$ when $\omega \in (0, 1]$. Same as APPR, the operations of $\mathcal{Q}.\text{enqueue}(u)$, $\mathcal{Q}.\text{dequeue}()$, and $v \notin \mathcal{Q}$ are all in $\mathcal{O}(1)$.

During the updates, one should note that the real vector \mathbf{r} presents $D^{1/2}\mathbf{r}^{(t)}$ while the vector \mathbf{x} is $D^{1/2}\mathbf{x}^{(t)}$. In this case, the original active node condition is implicitly shifting from $|r_u| \geq \frac{2\alpha\epsilon\sqrt{d_u}}{1+\alpha}$ to $\sqrt{d_u}|r_u| \geq \frac{2\alpha\epsilon d_u}{1+\alpha}$. We use this shifted active condition in Lines 11 and 13 and inactive condition in Line 5. When $\omega \in (1, 2)$, it is possible $|r_u| < c \cdot \epsilon d_u$ and LOC-SOR will ignore this inactive node u during the updates. This step makes sure $\mathcal{S}_t = \{u_i : |r_{u_i}^{(t+\Delta_i)}| \geq \frac{2\alpha\epsilon\sqrt{d_u}}{1+\alpha}\}$ during the updates.

B.3 Optimal GS-SOR and Proof of Corollary 3.4

We introduce the following standard result.

Lemma B.4 (Young [55], Section 12.2, Theorem 2.1). *Given the GS-SOR method for solving $\mathbf{Q}\mathbf{x} = \mathbf{b}$, if the underlying matrix \mathbf{Q} is a Stieltjes matrix and set relaxation parameter ω as*

$$\omega^* = \frac{2}{1 + \sqrt{1 - \rho(\mathbf{B})^2}} = 1 + \left(\frac{\rho(\mathbf{B})}{1 + \sqrt{1 - \rho(\mathbf{B})^2}} \right)^2, \quad (18)$$

where $\rho(\mathbf{B})$ is the largest eigenvalue (in magnitude) of $\mathbf{B} = \mathbf{I} - \text{diag}(\mathbf{Q})^{-1}\mathbf{Q}$, then

$$\omega^* - 1 \leq \rho(\mathbf{L}_{\omega^*}) \leq \sqrt{\omega^* - 1}, \quad (19)$$

where $\mathbf{L}_{\omega} := (\text{diag}(\mathbf{Q}) - \omega\mathbf{Q}_L)^{-1}(\omega\mathbf{Q}_U - (\omega - 1)\text{diag}(\mathbf{Q}))$ with $\mathbf{Q} = \text{diag}(\mathbf{Q}) - \mathbf{Q}_U - \mathbf{Q}_L$.

Corollary 3.4. *Let $\omega = \omega^* \triangleq 2/(1 + \sqrt{1 - (1 - \alpha)^2/(1 + \alpha)^2})$ and $\mathcal{S}_t = \mathcal{V}, \forall t \geq 0$, the global version of LocSOR has the following convergence bound*

$$\|\mathbf{r}^{(t)}\|_2 \leq \frac{2}{(1 + \alpha)\sqrt{d_s}} \left(\frac{1 - \sqrt{\alpha}}{1 + \sqrt{\alpha}} + \epsilon_t \right)^t, \quad (20)$$

where ϵ_t are small positive numbers with $\lim_{t \rightarrow \infty} \epsilon_t = 0$.

Proof. Recall $\mathbf{Q} = \mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}$ and we consider the underlying graph as simple which means \mathbf{A} has 0 diagonal. Hence, $\text{diag}(\mathbf{Q}) = \mathbf{I}$ and \mathbf{B} is defined as

$$\mathbf{B} = \frac{1-\alpha}{1+\alpha} \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}, \quad \rho(\mathbf{B}) = \frac{1-\alpha}{1+\alpha}.$$

Since \mathbf{Q} is a Stieltjes matrix, then Lemma B.4 gives a bound on the spectral radius of \mathbf{L}_ω as

$$\rho(\mathbf{L}_{\omega^*}) \leq \left(\frac{2}{1 + \sqrt{1 - \rho(\mathbf{B})^2}} - 1 \right)^{1/2} = \left(\frac{2(1+\alpha)}{1+\alpha + 2\sqrt{\alpha}} - 1 \right)^{1/2} = \frac{1 - \sqrt{\alpha}}{1 + \sqrt{\alpha}}.$$

Recall that Gelfand's formula states [52]: Given spectral radius $\rho(\mathbf{L}_{\omega^*}) := \max_{i \in [n]} |\lambda_i(\mathbf{L}_{\omega^*})|$, where $\lambda_i(\cdot)$ is the i -th eigenvalue, there exists a sequence $\{\epsilon_t\}_{t \geq 0}$ such that $\|\mathbf{L}_{\omega^*}^t\|_2 = (\rho(\mathbf{L}_{\omega^*}) + \epsilon_t)^t$ and $\lim_{t \rightarrow \infty} \epsilon_t = 0$. The standard SOR method is defined as the following

$$\begin{aligned} \mathbf{x}^{(t+1)} &= (\text{diag}(\mathbf{Q}) - \omega^* \mathbf{Q}_L)^{-1} \left(\omega^* \mathbf{b} + (\omega^* \mathbf{Q}_U - (\omega^* - 1) \text{diag}(\mathbf{Q})) \mathbf{x}^{(t)} \right) \\ &= \mathbf{L}_{\omega^*} \mathbf{x}^{(t)} + \omega^* (\text{diag}(\mathbf{Q}) - \omega^* \mathbf{Q}_L)^{-1} \mathbf{b}, \end{aligned}$$

Note $\mathbf{r}^{(t)} = \mathbf{Q} \mathbf{e}^{(t)} = \mathbf{Q} \mathbf{L}_{\omega^*}^t \mathbf{Q}^{-1} \mathbf{Q} \mathbf{e}^{(0)} = \mathbf{Q} \mathbf{L}_{\omega^*}^t \mathbf{Q}^{-1} \mathbf{r}^{(0)}$. We have

$$\begin{aligned} \|\mathbf{r}^{(t)}\|_2 &= \|\mathbf{Q} \mathbf{L}_{\omega^*}^t \mathbf{Q}^{-1} \mathbf{r}^{(0)}\|_2 \leq \|\mathbf{Q}\|_2 \|\mathbf{L}_{\omega^*}^t\|_2 \|\mathbf{Q}^{-1}\|_2 \|\mathbf{r}^{(0)}\|_2 \\ &\leq \frac{2}{1+\alpha} \cdot \|\mathbf{L}_{\omega^*}^t\|_2 \cdot \frac{1+\alpha}{2\alpha} \cdot \frac{2\alpha}{(1+\alpha)\sqrt{d_s}} = \frac{2\|\mathbf{L}_{\omega^*}^t\|_2}{(1+\alpha)\sqrt{d_s}}. \end{aligned}$$

□

To meet the stop condition, we require $|r_u^{(t)}| \leq \frac{2\alpha\epsilon\sqrt{d_u}}{1+\alpha}$. It is enough to make sure $\|\mathbf{r}^{(t)}\|_2 \leq \frac{2\alpha\epsilon}{(1+\alpha)\sqrt{d_s}}$. This leads to find t such that

$$\|\mathbf{r}^{(t)}\|_2 \leq \frac{2\|\mathbf{L}_{\omega^*}^t\|_2}{(1+\alpha)\sqrt{d_s}} \leq \frac{2\alpha\epsilon}{(1+\alpha)\sqrt{d_s}} \Leftrightarrow \left(\frac{1 - \sqrt{\alpha}}{1 + \sqrt{\alpha}} + \epsilon_t \right)^t \leq \alpha\epsilon.$$

When $\epsilon_t = o(\sqrt{\alpha})$, then the runtime of global LOCSOR is $\tilde{\mathcal{O}}(m/\sqrt{\alpha})$ where $\tilde{\mathcal{O}}$ hides $\log \frac{1}{\epsilon}$.

B.4 LOCGD and Proof of Theorem 3.5

The *local* gradient descent, namely LOCGD is to use $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \mathbf{r}_{\mathcal{S}_t}^{(t)}$ and $\mathbf{r}^{(t+1)} = \mathbf{r}^{(t)} - \mathbf{Q} \mathbf{r}_{\mathcal{S}_t}^{(t)}$, where $\mathcal{S}_t = \{u_i : |r_{u_i}^{(t+\Delta_i)}| \geq 2\alpha\epsilon\sqrt{d_u}/(1+\alpha)\}$ where $\Delta_i = 0$. Algo. 4 presents our actual implementation of LOCGD via FIFO Queue.

Algo. 4 LOCGD($\alpha, \epsilon, s, \mathcal{G}$) via FIFO Queue

```

1: Initialize:  $\mathbf{r} \leftarrow c\mathbf{e}_s, \mathbf{x} \leftarrow \mathbf{0}, c = \frac{2\alpha}{1+\alpha}$ 
2:  $\mathcal{Q} \leftarrow \{s\}$  // Assume  $\epsilon \leq \frac{1}{d_s}$ 
3:  $t = 0$ 
4: while  $\mathcal{Q} \neq \emptyset$  do
5:    $\mathcal{S}_t \leftarrow \emptyset$ 
6:   while  $\mathcal{Q} \neq \emptyset$  do
7:      $u \leftarrow \mathcal{Q}.\text{dequeue}()$ 
8:      $\mathcal{S}_t.\text{append}((u, r_u))$ 
9:      $x_u \leftarrow x_u + r_u$ 
10:     $r_u \leftarrow 0$ 
11:    for  $(u, \tilde{r}) \in \mathcal{S}_t$  do
12:      for  $v \in \mathcal{N}(u)$  do
13:         $r_v \leftarrow r_v + \frac{(1-\alpha)\tilde{r}}{(1+\alpha)d_u}$ 
14:        if  $|r_v| \geq c \cdot \epsilon d_v$  and  $v \notin \mathcal{Q}$  then
15:           $\mathcal{Q}.\text{enqueue}(v)$ 
16:     $t \leftarrow t + 1$ 
17: return  $\mathbf{x}, \mathbf{r}$ 

```

Algo. 4 presents LOCGD similar to the real queue-based implementation of LOCSOR. It has monotonic and nonnegative properties during the updates of $\mathbf{r} \geq \mathbf{0}$ and $\mathbf{x} \geq \mathbf{0}$. Again, the operations of $\mathcal{Q}.\text{enqueue}(u)$, $\mathcal{Q}.\text{dequeue}()$, and $v \notin \mathcal{Q}$ are all in $\mathcal{O}(1)$. During the updates, one should note that \mathbf{r} presents $\mathbf{D}^{1/2} \mathbf{r}^{(t)}$ while \mathbf{x} is $\mathbf{D}^{1/2} \mathbf{x}^{(t)}$. All shifted conditions are the same as of LOCSOR. The key advantage of LOCGD is that it is highly parallelizable, while LOCSOR is truly an online update, so it is hard to parallelize.

Lemma B.5 (Iterations of LOCGD). *With the initial $\mathbf{x}^{(0)} = \mathbf{0}$, $\mathbf{r}^{(0)} = \mathbf{b}$, $\mathcal{S}_0 = \text{supp}(\mathbf{r}^{(0)})$, denote $\tilde{\mathbf{r}}^{(t)} = \mathbf{D}^{1/2}\mathbf{r}^{(t)}$. LOCGD defined in (9) has the following properties: 1) $\mathbf{x}^{(t)} \geq \mathbf{0}$, $\mathbf{r}^{(t)} \geq \mathbf{0}$ and $\|\mathbf{r}^{(t)}\| \geq \|\mathbf{r}^{(t+1)}\|_1$; 2) The residual and estimation error satisfies*

$$\|\tilde{\mathbf{r}}^{(t+1)}\|_1 = \left(1 - \frac{2\alpha\gamma_t}{1+\alpha}\right) \|\tilde{\mathbf{r}}^{(t)}\|_1, \quad \gamma_t = \sum_{i=1}^{|\mathcal{S}_t|} \frac{\tilde{r}_{u_i}^{(t+\Delta_i)}}{\|\tilde{\mathbf{r}}^{(t)}\|_1}, \text{ where } \Delta_i = 0.$$

Proof. We first show $\mathbf{x}^{(t)} \geq \mathbf{0}$, $\mathbf{r}^{(t)} \geq \mathbf{0}$ are all nonnegative vectors during the updates when $\mathbf{b} \geq \mathbf{0}$. This can be seen from the induction. At the initial stage, $\mathbf{x}^{(0)} \geq \mathbf{0}$ and $\mathbf{r}^{(0)} = \mathbf{b} \geq \mathbf{0}$. Now assume that for any $t \geq 0$, $\mathbf{x}^{(t)} \geq \mathbf{0}$ and $\mathbf{r}^{(t)} \geq \mathbf{0}$. Then $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \mathbf{r}_{\mathcal{S}_k}^{(t)} \geq \mathbf{0}$, and $\mathbf{r}^{(t+1)} = \mathbf{r}_{\mathcal{S}_t}^{(t)} + \frac{1-\alpha}{1+\alpha}\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}\mathbf{r}_{\mathcal{S}_t}^{(t)} \geq \mathbf{0}$. Therefore, $\mathbf{x}^{(t)} \geq \mathbf{0}$ and $\mathbf{r}^{(t)} \geq \mathbf{0}$ for all t . Note $\tilde{\mathbf{r}}^{(t+1)} = \tilde{\mathbf{r}}_{\mathcal{S}_t}^{(t)} + \frac{1-\alpha}{1+\alpha}\mathbf{A}\mathbf{D}^{-1} \cdot \tilde{\mathbf{r}}_{\mathcal{S}_t}^{(t)}$ and since $\|\mathbf{A}\mathbf{D}^{-1}\tilde{\mathbf{r}}_{\mathcal{S}_t}^{(t)}\|_1 = \|\tilde{\mathbf{r}}_{\mathcal{S}_t}^{(t)}\|_1$, we will have

$$\|\tilde{\mathbf{r}}^{(t+1)}\|_1 = \left(1 - \frac{2\alpha}{1+\alpha} \frac{\|\tilde{\mathbf{r}}_{\mathcal{S}_t}^{(t)}\|_1}{\|\tilde{\mathbf{r}}^{(t)}\|_1}\right) \|\tilde{\mathbf{r}}^{(t)}\|_1, \text{ where } \gamma_t := \|\tilde{\mathbf{r}}_{\mathcal{S}_t}^{(t)}\|_1 / \|\tilde{\mathbf{r}}^{(t)}\|_1.$$

□

Then, we can bound the total residual as the following theorem.

Theorem 3.5 (Runtime bound of LOCGD). *Given the configuration $\theta = (\alpha, \epsilon, s, \mathcal{G})$ with $\alpha \in (0, 1)$ and $\epsilon \leq 1/d_s$ and let $\mathbf{r}^{(T)}$ and $\mathbf{x}^{(T)}$ be returned by LOCGD defined in (9) for solving Equ. (4). There exists a real implementation of (9) such that the runtime $\mathcal{T}_{\text{LocGD}}$ is bounded by*

$$\frac{1+\alpha}{2} \cdot \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\alpha\bar{\gamma}_T} \left(1 - \frac{\|\tilde{\mathbf{r}}^{(T)}\|_1}{\|\tilde{\mathbf{r}}^{(0)}\|_1}\right) \leq \mathcal{T}_{\text{LocGD}} \leq \frac{1+\alpha}{2} \cdot \min\left\{\frac{1}{\alpha\epsilon}, \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\alpha\bar{\gamma}_T} \ln \frac{C}{\epsilon}\right\},$$

where $C = (1+\alpha)/((1-\alpha)|\mathcal{I}_T|)$, $\mathcal{I}_T = \text{supp}(\mathbf{r}^{(T)})$. Furthermore, $\overline{\text{vol}}(\mathcal{S}_T)/\bar{\gamma}_T \leq 1/\epsilon$ and the estimate $\hat{\boldsymbol{\pi}} := \mathbf{D}^{1/2}\mathbf{x}^{(T)}$ satisfies $\|\mathbf{D}^{-1}(\hat{\boldsymbol{\pi}} - \boldsymbol{\pi})\|_\infty \leq \epsilon$.

Proof. We first show bound $1/(\alpha\epsilon)$. We first rearrange $\mathbf{r}^{(t+1)} = \mathbf{r}^{(t)} - \mathbf{Q}\mathbf{r}_{\mathcal{S}_t}^{(t)}$ into

$$\mathbf{D}^{1/2}\mathbf{r}^{(t+1)} + \mathbf{D}^{1/2}\mathbf{r}_{\mathcal{S}_t}^{(t)} = \mathbf{D}^{1/2}\mathbf{r}^{(t)} + \frac{1-\alpha}{1+\alpha}\mathbf{A}\mathbf{D}^{-1}\mathbf{D}^{1/2}\mathbf{r}_{\mathcal{S}_t}^{(t)}.$$

Note $\mathbf{r}^{(t)} \geq \mathbf{0}$ and $\|\mathbf{A}\mathbf{D}^{-1}\mathbf{D}^{1/2}\mathbf{r}_{\mathcal{S}_t}^{(t)}\|_1 = \|\mathbf{D}^{1/2}\mathbf{r}_{\mathcal{S}_t}^{(t)}\|_1$. Hence, it leads to

$$\|\mathbf{D}^{1/2}\mathbf{r}_{\mathcal{S}_t}^{(t)}\|_1 = \frac{1+\alpha}{2\alpha} \left(\|\mathbf{D}^{1/2}\mathbf{r}^{(t)}\|_1 - \|\mathbf{D}^{1/2}\mathbf{r}^{(t+1)}\|_1\right).$$

At each local iterative t , by the active node condition $2\alpha\epsilon\sqrt{d_u}/(1+\alpha) \leq r_u^{(t)}$, we have

$$\epsilon \text{vol}(\mathcal{S}_t) = \sum_{u \in \mathcal{S}_t} \epsilon d_u \leq \sum_{u \in \mathcal{S}_t} \frac{(1+\alpha)\sqrt{d_u}r_u^{(t)}}{2\alpha} = \frac{1+\alpha}{2\alpha} \|\mathbf{D}^{1/2}\mathbf{r}_{\mathcal{S}_t}^{(t)}\|_1.$$

Then the total run time of LOCGD presented in Algo. 4 is

$$\sum_{t=0}^{T-1} \text{vol}(\mathcal{S}_t) \leq \frac{1}{\epsilon} \left(\frac{1+\alpha}{2\alpha}\right)^2 \left(\|\mathbf{D}^{1/2}\mathbf{r}^{(0)}\|_1 - \|\mathbf{D}^{1/2}\mathbf{r}^{(T)}\|_1\right) \leq \frac{1+\alpha}{2\alpha\epsilon}.$$

Therefore, the total run time is at most $\mathcal{T}_{\text{LocGD}} := \sum_{t=0}^{T-1} \text{vol}(\mathcal{S}_t) \leq \frac{1+\alpha}{2\alpha\epsilon}$. For estimating the bounds of T , by the Weierstrass product inequality [26] and Lemma B.5, we use the similar argument made in Lemma B.3 and continue to have

$$\frac{(1+\alpha)}{2\alpha\bar{\gamma}_T} \left(1 - \frac{\|\tilde{\mathbf{r}}^{(T)}\|_1}{\|\tilde{\mathbf{r}}^{(0)}\|_1}\right) \leq T \leq \frac{(1+\alpha)}{2\alpha\bar{\gamma}_T} \ln \frac{\|\tilde{\mathbf{r}}^{(0)}\|_1}{\|\tilde{\mathbf{r}}^{(T)}\|_1},$$

Note that each nonzero $\tilde{r}_u^{(T)}$ has at least part of the magnitude from the push operation of an active node, say v_u at time $t' < T$. This means each nonzero of $\mathbf{D}^{1/2}\mathbf{r}^{(T)}$ satisfies

$$\tilde{r}_u^{(T)} \geq \frac{(1-\alpha)\tilde{r}_{v_u}^{(t')}}{(1+\alpha)d_{v_u}} \geq \frac{(1-\alpha) \cdot 2\alpha\epsilon d_{v_u}/(1+\alpha)}{(1+\alpha)d_{v_u}} = \frac{2\alpha(1-\alpha)\epsilon}{(1+\alpha)^2}, \text{ for } u \in \mathcal{I}_T.$$

Hence, we have $\|\tilde{\mathbf{r}}^{(T)}\|_1 \geq \frac{2\alpha(1-\alpha)\epsilon|\mathcal{I}_T|}{(1+\alpha)^2}$ and T is further bounded as

$$T \leq \frac{(1+\alpha)}{2\alpha\bar{\gamma}_T} \ln \frac{\|\tilde{\mathbf{r}}^{(0)}\|_1}{\frac{2\alpha(1-\alpha)\epsilon|\mathcal{I}_T|}{(1+\alpha)^2}} := \frac{(1+\alpha)}{2\alpha\bar{\gamma}_T} \ln \frac{C_T}{\epsilon}, \text{ where } C_T = \frac{(1+\alpha)}{(1-\alpha)|\mathcal{I}_T|}.$$

The lower bound of $1/\epsilon$, i.e., $\overline{\text{vol}}(\mathcal{S}_t)/\bar{\gamma}_T \leq 1/\epsilon$, directly follows a similar strategy of previous proof by noticing that $\|\mathbf{D}^{1/2}\mathbf{r}^{(t)}\|_1$ is monotonically decreasing. \square

Remark B.6. One may consider designing local methods based on Jacobi and Richardson's iterations. Indeed, these two methods have the same updates as standard GD. Recall the standard GD method to solve (4) is $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \mathbf{r}^{(t)}$, $\mathbf{r}^{(t+1)} = \mathbf{r}^{(t)} - \mathbf{Q}\mathbf{r}^{(t)}$. The Richardson's iteration is $\mathbf{x}^{(t+1)} = (\mathbf{I} - \omega\mathbf{W})\mathbf{x}^{(t)} + \omega\mathbf{b}$, i.e., $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \omega(\mathbf{W}\mathbf{x}^{(t)} - \mathbf{b})$. The optimal $\omega^* = 2/(\lambda_{\min} + \lambda_{\max})$ where $\lambda_{\min} = 2\alpha/(1+\alpha)$ and $\lambda_{\max} \leq 2/(1+\alpha)$. Hence one can choose $\omega = 1 \leq \omega^*$ [19]. It leads to $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \mathbf{r}^{(t)}$. One can get the same result for the Jacobi method.

C Local Chebyshev Method - LOCCH

C.1 Nonhomogeneous of Second-order Difference Equation

We begin by providing the solutions of the second-order nonhomogeneous equation as the following

Lemma C.1 (Stević [47]). *The solution of the second-order nonhomogeneous difference equation*

$$x_{t+1} + px_t + qx_{t-1} = f_t, \quad t = 1, 2, \dots \quad (21)$$

is characterized by the following two cases

$$x_t = \begin{cases} \frac{1}{\hat{\lambda}_2 - \hat{\lambda}_1} \left(\hat{\lambda}_1^t \left(\hat{\lambda}_2 x_0 - x_1 - \sum_{k=1}^t \frac{f_k}{\hat{\lambda}_1^k} \right) + \hat{\lambda}_2^t \left(x_1 - \hat{\lambda}_1 x_0 + \sum_{k=1}^t \frac{f_k}{\hat{\lambda}_2^k} \right) \right) & p^2 \neq 4q \\ \left(-\frac{p}{2} \right)^t \left(x_0 - \sum_{k=1}^t \frac{k f_k}{(-p/2)^{k+1}} \right) + t \left(-\frac{p}{2} \right)^{t-1} \left(x_1 - \left(-\frac{p}{2} \right) x_0 + \sum_{k=1}^t \frac{f_k}{(-p/2)^k} \right) & p^2 = 4q \end{cases},$$

where $\hat{\lambda}_1, \hat{\lambda}_2$ are two roots of $\lambda^2 + p\lambda + q = 0$, and the summation follows convention $\sum_{k=1}^0 \cdot = 0$.

Based on the above lemma, we have the following corollary

Corollary C.2 (Second-order nonhomogeneous equation). *Given $|a| \leq 1$, the second-order nonhomogeneous equation*

$$x_{t+1} - 2ax_t + x_{t-1} = f_t$$

has the following solution

$$x_t = \begin{cases} x_0 + t(x_1 - x_0) + \sum_{k=1}^t (t-k)f_k & \text{if } a = 1 \\ \frac{\sin(\theta t)x_1 - \sin(\theta(t-1))x_0}{\sin \theta} + \sum_{k=1}^t \frac{\sin(\theta(t-k))f_k}{\sin \theta} & \text{if } |a| < 1 \text{ where } \theta = \arccos(a) \\ (-1)^t(x_0 - t(x_0 + x_1)) + (-1)^t \left(\sum_{k=1}^t (-1)^{-k-1} (t-k)f_k \right) & \text{if } a = -1. \end{cases} \quad (22)$$

Proof. The first and last two cases can be directly followed from Lemma C.1. Since $\hat{\lambda}_1$ and $\hat{\lambda}_2$ are the two complex roots of $\lambda^2 - 2a\lambda + 1 = 0$, we write $\hat{\lambda}_1 = re^{i\theta} = r(\cos \theta + i \sin \theta)$ and $\hat{\lambda}_2 = re^{-i\theta} = r(\cos \theta - i \sin \theta)$. It indicates

$$\lambda^2 - 2a\lambda + 1 = (\lambda - re^{i\theta})(\lambda - re^{-i\theta}) = \lambda^2 - r(e^{i\theta} + e^{-i\theta})\lambda + r^2 = 0.$$

Since $1 > a^2$, $\hat{\lambda}_1 = a - i\sqrt{1-a^2}$, and $\mathbf{Re}(\hat{\lambda}_1)^2 + \mathbf{Im}(\hat{\lambda}_1)^2 = a^2 + (1-a^2) = 1$, then $r = 1$. Then $\theta = \arccos(\mathbf{Re}(\hat{\lambda}_1)) = \arccos(a)$, and $\sin(\theta) = \mathbf{Im}(\hat{\lambda}_1) = \sqrt{1-a^2}$. Finally, $\hat{\lambda}_1^t = e^{it\theta} = \cos(t\theta) + i \sin(t\theta)$, and

$$\begin{aligned}\hat{\lambda}_1 &= \cos \theta + i \sin \theta, & \hat{\lambda}_2 &= \cos \theta - i \sin \theta \\ \hat{\lambda}_1 \hat{\lambda}_2 &= e^{it\theta} e^{-it\theta} = 1 \\ \hat{\lambda}_2 - \hat{\lambda}_1 &= -2i \sin \theta \\ \hat{\lambda}_1^t &= \cos(\theta t) + i \sin(\theta t), & \hat{\lambda}_2^t &= \cos(\theta t) - i \sin(\theta t) \\ \hat{\lambda}_2^t - \hat{\lambda}_1^t &= e^{-it\theta} - e^{it\theta} = -2i \sin(\theta t). \\ \hat{\lambda}_1^{t-1} - \hat{\lambda}_2^{t-1} &= 2i \sin(\theta(t-1)) \stackrel{\hat{\lambda}_1 \hat{\lambda}_2 = 1}{=} \hat{\lambda}_1^t \hat{\lambda}_2 - \hat{\lambda}_2^t \hat{\lambda}_1\end{aligned}$$

Based on these, we have

$$\begin{aligned}x_t &= \frac{1}{\hat{\lambda}_2 - \hat{\lambda}_1} \left(\hat{\lambda}_1^t \left(\hat{\lambda}_2 x_0 - x_1 - \sum_{k=1}^t \frac{f_k}{\hat{\lambda}_1^k} \right) + \hat{\lambda}_2^t \left(x_1 - \hat{\lambda}_1 x_0 + \sum_{k=1}^t \frac{f_k}{\hat{\lambda}_2^k} \right) \right) \\ &= \frac{\hat{\lambda}_1^t (\hat{\lambda}_2 x_0 - x_1) + \hat{\lambda}_2^t (x_1 - \hat{\lambda}_1 x_0)}{\hat{\lambda}_2 - \hat{\lambda}_1} + \frac{1}{\hat{\lambda}_2 - \hat{\lambda}_1} \left(-\hat{\lambda}_1^t \sum_{k=1}^t \frac{f_k}{\hat{\lambda}_1^k} + \hat{\lambda}_2^t \sum_{k=1}^t \frac{f_k}{\hat{\lambda}_2^k} \right) \\ &= \frac{(2i \sin(\theta(t-1))x_0 - 2i \sin(\theta t)x_1)}{-2i \sin \theta} + \frac{1}{-2i \sin \theta} \left(-\hat{\lambda}_1^t \sum_{k=1}^t \frac{f_k}{\hat{\lambda}_1^k} + \hat{\lambda}_2^t \sum_{k=1}^t \frac{f_k}{\hat{\lambda}_2^k} \right) \\ &= \frac{\sin(\theta t)x_1 - \sin(\theta(t-1))x_0}{\sin \theta} + \frac{1}{-2i \sin \theta} \sum_{k=1}^t (\hat{\lambda}_2^{t-k} - \hat{\lambda}_1^{t-k}) f_k \\ &= \frac{\sin(\theta t)x_1 - \sin(\theta(t-1))x_0}{\sin \theta} + \frac{1}{-2i \sin \theta} \sum_{k=1}^t -2i \sin(\theta(t-k)) f_k \\ &= \frac{\sin(\theta t)x_1 - \sin(\theta(t-1))x_0}{\sin \theta} + \frac{\sum_{k=1}^t \sin(\theta(t-k)) f_k}{\sin \theta}.\end{aligned}$$

□

Lemma C.3. For $|\lambda_i| \leq 1, i = 1, 2, \dots, n$, equations $y_i^{(t+1)} - 2\lambda_i y_i^{(t)} + y_i^{(t-1)} = 0$ have the following solutions

$$y_i^{(t)} = \begin{cases} y_i^{(0)} + t(y_i^{(1)} - y_i^{(0)}) & \text{if } \lambda_i = 1 \\ \frac{\sin(\theta_i t) y_i^{(1)} - \sin(\theta_i(t-1)) y_i^{(0)}}{\sin(\theta_i)} & \text{if } |\lambda_i| < 1 \text{ where } \theta_i = \arccos(\lambda_i) \\ (-1)^t (z_{i,0} - t(y_i^{(0)} + y_i^{(1)})) & \text{if } \lambda_i = -1. \end{cases} \quad (23)$$

Furthermore, when $y_i^{(1)} = \lambda_i y_i^{(0)}$ for $i = 1, 2, \dots, n$, then solutions can be simplified as

$$y_i^{(t)} = \begin{cases} y_i^{(0)} & \text{if } \lambda_i = 1 \\ y_i^{(0)} \cos(\theta_i t) & \text{if } |\lambda_i| < 1 \text{ where } \theta_i = \arccos(\lambda_i) \\ z_{i,0} (-1)^t & \text{if } \lambda_i = -1. \end{cases} \quad (24)$$

Proof. The first part is a consequence of Corollary C.2 by letting $f_t = 0$. To see the second identity of this lemma, note that

$$\frac{(\lambda_i \sin(\theta_i t) - \sin(\theta_i(t-1)))}{\sin \theta_i} = \cos(\theta_i t).$$

Indeed, by expanding $\sin(\theta_i(t-1))$, we have

$$\begin{aligned}\sin(\theta_i(t-1)) &= \sin(\theta_i t - \theta_i) \\ &= \sin(\theta_i t) \cos(-\theta_i) + \cos(\theta_i t) \sin(-\theta_i) \\ &= \sin(\theta_i t) \cos(\theta_i) - \cos(\theta_i t) \sin(\theta_i) \\ &= \lambda_i \sin(\theta_i t) - \cos(\theta_i t) \sin(\theta_i)\end{aligned}$$

Hence, when $y_i^{(1)} = \lambda_i y_i^{(0)}$, we have

$$\frac{\sin(\theta_i t) y_i^{(1)} - \sin(\theta_i(t-1)) y_i^{(0)}}{\sin(\theta_i)} = \frac{(\lambda_i \sin(\theta_i t) - \sin(\theta_i(t-1))) y_i^{(0)}}{\sin \theta_i} = \cos(\theta_i t) y_i^{(0)}.$$

□

Lemma C.4. Given $t \geq 1$, $|\lambda_i| \leq 1$, the n second-order difference equations

$$y_i^{(t+1)} - 2\lambda_i y_i^{(t)} + y_i^{(t-1)} = h_{i,t}, \quad i = 1, 2, \dots, n.$$

have the following solutions

$$y_i^{(t)} = \begin{cases} y_i^{(0)} + t(y_i^{(1)} - y_i^{(0)}) + \sum_{k=1}^{t-1} (t-k) h_{i,k} & \text{if } \lambda_i = 1 \\ \frac{\sin(\theta_i t) y_i^{(1)} - \sin(\theta_i(t-1)) y_i^{(0)}}{\sin(\theta_i)} + \sum_{k=1}^{t-1} \frac{\sin(\theta_i(t-k))}{\sin \theta_i} h_{i,k} & \text{if } |\lambda_i| < 1 \text{ where } \theta_i = \arccos(\lambda_i) \\ (-1)^t (z_{i,0} - t(y_i^{(0)} + y_i^{(1)})) + \sum_{k=1}^{t-1} (-1)^{t-k-1} (t-k) h_{i,k} & \text{if } \lambda_i = -1. \end{cases} \quad (25)$$

Furthermore, with initial conditions $y_i^{(1)} = \lambda_i y_i^{(0)}$, $y_i^{(t)}$ can be simplified as

$$y_i^{(t)} = \begin{cases} y_i^{(0)} + \sum_{k=1}^{t-1} (t-k) h_{i,k} & \text{if } \lambda_i = 1 \\ \cos(\theta_i t) y_i^{(0)} + \sum_{k=1}^{t-1} \frac{\sin(\theta_i(t-k))}{\sin \theta_i} h_{i,k} & \text{if } |\lambda_i| < 1 \text{ where } \theta_i = \arccos(\lambda_i) \\ (-1)^t y_i^{(0)} + \sum_{k=1}^{t-1} (-1)^{t-k-1} (t-k) h_{i,k} & \text{if } \lambda_i = -1. \end{cases} \quad (26)$$

Proof. The first part is a consequence of Corollary C.2 by letting $f_t = h_{i,t}$. We prove the second part by considering three cases of λ_i as

- **Case 1.** When $\lambda_i = 1$, we have $y_i^{(t+1)} - 2y_i^{(t)} + y_i^{(t-1)} = h_{i,t}$. For $t \geq 1$, the solution the above is

$$y_i^{(t)} = y_i^{(0)} + t(y_i^{(1)} - y_i^{(0)}) + \sum_{k=1}^{t-1} (t-k) h_{i,k} = y_i^{(0)} + \sum_{k=1}^{t-1} (t-k) h_{i,k},$$

where the second equation is due to $y_i^{(1)} = \lambda_1 y_i^{(0)} = y_i^{(0)}$.

- **Case 2.** When $\lambda_n = -1$ (\mathcal{G} is a bipartite graph), we have $y_i^{(t+1)} + 2y_i^{(t)} + y_i^{(t-1)} = h_{i,t}$. For $t \geq 1$, the solution is

$$y_i^{(t)} = (-1)^t y_i^{(0)} + (-1)^t \sum_{k=1}^{t-1} \frac{(t-k) h_{i,k}}{(-1)^{k+1}} = (-1)^t y_i^{(0)} + \sum_{k=1}^{t-1} (-1)^{t-k-1} (t-k) h_{i,k}.$$

- **Case 3.** When $|\lambda_i| < 1$, and define $\theta_i = \arccos(\lambda_i)$. We use a similar argument in Lemma C.3 and continue to have

$$\begin{aligned}y_i^{(t)} &= \frac{\sin(\theta_i t) y_i^{(1)} - \sin(\theta_i(t-1)) y_i^{(0)}}{\sin \theta_i} + \frac{\sum_{k=1}^{t-1} \sin(\theta_i(t-k)) h_{i,k}}{\sin \theta_i} \\ &= \frac{(\lambda_i \sin(\theta_i t) - \sin(\theta_i(t-1))) y_i^{(0)}}{\sin \theta_i} + \frac{\sum_{k=1}^{t-1} \sin(\theta_i(t-k)) h_{i,k}}{\sin \theta_i} \\ &= \cos(\theta_i t) y_i^{(0)} + \sum_{k=1}^{t-1} \frac{\sin(\theta_i(t-k))}{\sin \theta_i} h_{i,k}.\end{aligned}$$

□

C.2 Properties on Ratio of Chebyshev Polynomials

The next lemma presents the properties of Chebyshev polynomials.

Lemma C.5 (Chebyshev polynomial bound). *For $t \geq 1$, the Chebyshev polynomial of the first kind is defined recursively as*

$$T_{t+1}(x) = 2xT_t(x) - T_{t-1}(x) \quad \text{with} \quad T_0(x) = 1, \quad T_1(x) = x.$$

For $t \geq 1$, define $\delta_t = T_{t-1}(\frac{1+\alpha}{1-\alpha})/T_t(\frac{1+\alpha}{1-\alpha})$, then

1. $T_t(x = \frac{1+\alpha}{1-\alpha})$ and δ_t defines the following sequence

$$\delta_{t+1} = \left(2\frac{1+\alpha}{1-\alpha} - \delta_t\right)^{-1}, \quad \text{where } \delta_1 = \frac{1-\alpha}{1+\alpha}.$$

2. The closed-form $\delta_{1:t}$ can be upper bounded as

$$\delta_{1:t} = \frac{1}{T_t(\frac{1+\alpha}{1-\alpha})} = \frac{2}{\tilde{\alpha}^t + \tilde{\alpha}^{-t}} \leq 2 \left(\frac{1-\sqrt{\alpha}}{1+\sqrt{\alpha}}\right)^t.$$

3. Note $\delta_1 = T_0/T_1 = 1/x = \frac{1-\alpha}{1+\alpha}$, the sequence $\{\delta_t\}$ satisfies $\delta_t < 1, \forall t \geq 1$ and

$$1 = 2\delta_{t+1}x - \delta_t\delta_{t+1}, \quad t = 1, 2, \dots$$

4. Denote $\delta_{j:t} = \prod_{i=j}^t \delta_i$ for $t \geq j \geq 0$ and set the default value $\delta_{j:j-1} = 1$ for $j \geq 0$, then

$$\delta_{1:t}/(\delta_{1:k}) = \delta_{k+1:t} \leq 2\tilde{\alpha}^{t-k}, \quad \text{for } t \geq k \geq 0.$$

Proof. For the first item, let $x = \frac{1+\alpha}{1-\alpha}$, use the Chebyshev equation, we have

$$1 = 2 \left(\frac{1+\alpha}{1-\alpha}\right) \frac{T_t}{T_{t+1}} - \frac{T_{t-1}}{T_{t+1}} = 2 \left(\frac{1+\alpha}{1-\alpha}\right) \delta_{t+1} - \delta_t\delta_{t+1} \Rightarrow \delta_{t+1}^{-1} = 2 \left(\frac{1+\alpha}{1-\alpha}\right) - \delta_t.$$

For the second item, for all $t \geq 0$, if $\xi = \frac{x+x^{-1}}{2} \neq 0$, it is well known that the T_t can be rewritten as

$$T_t \left(\xi = \frac{x+x^{-1}}{2} \right) = \frac{x^t + x^{-t}}{2}.$$

For our problem, recall we defined $\tilde{\alpha} = \frac{1-\sqrt{\alpha}}{1+\sqrt{\alpha}}$ and using $x = \frac{1+\alpha}{1-\alpha} \neq 0$, one can verify that

$$x = \frac{1+\alpha}{1-\alpha} = \frac{\tilde{\alpha} + \tilde{\alpha}^{-1}}{2} \iff T_t \left(x = \frac{1+\alpha}{1-\alpha} \right) = T_t \left(\frac{\tilde{\alpha} + \tilde{\alpha}^{-1}}{2} \right) = \frac{\tilde{\alpha}^t + \tilde{\alpha}^{-t}}{2}$$

and

$$\prod_{j=1}^t \delta_j = \frac{T_0}{T_1} \cdot \frac{T_1}{T_2} \cdot \frac{T_2}{T_3} \cdots \frac{T_{t-1}}{T_t} = \frac{1}{T_t} = \frac{2}{\tilde{\alpha}^t + \tilde{\alpha}^{-t}} = \frac{2\tilde{\alpha}^t}{\tilde{\alpha}^{2t} + 1} \leq 2\tilde{\alpha}^t = 2 \left(\frac{1-\sqrt{\alpha}}{1+\sqrt{\alpha}}\right)^t.$$

For the third item, it is sufficient to show that $\delta_t \leq \frac{1}{x}$ for all $t \geq 1$, for $x = \frac{1+\alpha}{1-\alpha}$. This can be done recursively, since $\delta_1 = x$ and

$$\delta_{t+1} = \frac{1}{2x - \delta_t} \leq \frac{1}{2x - \frac{1}{x}} \stackrel{x \geq \frac{1}{x}}{\leq} \frac{1}{x}.$$

For the last item, note when $t \geq 1$ and $k \geq 1$, we have the following inequalities

$$\prod_{j=1}^t \delta_j \cdot \prod_{j=1}^k \delta_j^{-1} = \frac{2}{\tilde{\alpha}^t + \tilde{\alpha}^{-t}} \cdot \frac{\tilde{\alpha}^k + \tilde{\alpha}^{-k}}{2} = \frac{\tilde{\alpha}^k + \tilde{\alpha}^{-k}}{\tilde{\alpha}^t + \tilde{\alpha}^{-t}} = \tilde{\alpha}^{t-k} \frac{\tilde{\alpha}^{2k} + 1}{\tilde{\alpha}^{2t} + 1} \leq 2\tilde{\alpha}^{t-k},$$

where note $\frac{\tilde{\alpha}^{2k} + 1}{\tilde{\alpha}^{2t} + 1} \in [1, 2]$ for $t \geq k$. □

C.3 Standard Chebyshev (CH) Method and Proof of Theorem C.6

This subsection introduces the standard Chebyshev algorithm. Our following theorem is to prove the runtime complexity of the Chebyshev polynomial iteration for solving Equ. (3).

Theorem C.6 (Standard CH). *For $t \geq 1$, consider the Chebyshev polynomials to solve Equ. (3) as*

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + (1 + \delta_{t:t+1})\mathbf{r}^{(t)} + \delta_{t:t+1}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}), \quad \mathbf{r}^{(t+1)} = 2\delta_{t+1}\mathbf{W}\mathbf{r}^{(t)} - \delta_{t:t+1}\mathbf{r}^{(t-1)},$$

where $\mathbf{x}^{(0)} = \mathbf{0}$, $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \mathbf{r}^{(0)}$ and $\delta_{t+1} = (2\frac{1+\alpha}{1-\alpha} - \delta_t)^{-1}$ with $\delta_1 = \frac{1-\alpha}{1+\alpha}$. Assume $\epsilon < 1/d_s$, then the residual has the following convergence bound

$$\|\mathbf{r}^{(t)}\|_2 \leq 2 \left(\frac{1 - \sqrt{\alpha}}{1 + \sqrt{\alpha}} \right)^t \|\mathbf{b}\|_2.$$

Let the estimate be $\hat{\boldsymbol{\pi}} = \mathbf{D}^{1/2}\mathbf{x}^{(t)}$, the the runtime of CH for reaching $\|\mathbf{D}^{-1/2}\mathbf{r}^{(t)}\|_\infty \leq \frac{2\alpha\epsilon}{1+\alpha}$ with $\|\mathbf{D}^{-1}(\hat{\boldsymbol{\pi}} - \boldsymbol{\pi})\|_\infty \leq \epsilon$ guarantee is at most

$$\mathcal{T}_{\text{CH}} \leq \Theta \left(m \left\lceil \frac{1 + \sqrt{\alpha}}{2\sqrt{\alpha}} \ln \frac{2}{\epsilon} \right\rceil \right) = \tilde{\Theta} \left(\frac{m}{\sqrt{\alpha}} \right).$$

Proof. Recall eigendecomposition of $\mathbf{W} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top$ where $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n]$ and each \mathbf{v}_i is the eigenvector. For $t \geq 1$, the residual $\mathbf{r}^{(t)}$ can be written as n second-order difference equations as

$$\mathbf{V}^\top \mathbf{r}^{(t+1)} - 2\delta_{t+1}\boldsymbol{\Lambda}\mathbf{V}^\top \mathbf{r}^{(t)} + \delta_{t:t+1}\mathbf{V}^\top \mathbf{r}^{(t-1)} = \mathbf{0},$$

where each i -th element-wise equation of the above can be written as the following

$$\mathbf{v}_i^\top \mathbf{r}^{(t+1)} - 2\delta_{t+1}\lambda_i \mathbf{v}_i^\top \mathbf{r}^{(t)} + \delta_{t+1}\delta_t \mathbf{v}_i^\top \mathbf{r}^{(t-1)} = 0, \quad i = 1, 2, \dots, n.$$

Define $\mathbf{V}^\top \mathbf{r}^{(t)} = \delta_{1:t}\mathbf{y}^{(t)}$. Each component $\mathbf{v}_i^\top \mathbf{r}^{(t)}$ is $\mathbf{v}_i^\top \mathbf{r}^{(t)} = \delta_{1:t}y_i^{(t)}$ where $\mathbf{v}_i^\top \mathbf{r}^{(0)} := y_i^{(0)}$ by default. The above can be rewritten

$$\begin{aligned} \delta_{1:t+1}y_i^{(t+1)} - 2\delta_{t+1}\delta_{1:t}\lambda_i y_i^{(t)} + \delta_{t+1}\delta_t\delta_{1:t-1}y_i^{(t-1)} &= 0 \\ \iff y_i^{(t+1)} - 2\lambda_i y_i^{(t)} + y_i^{(t-1)} &= 0, \end{aligned} \quad (27)$$

where \iff follows from $\delta_{1:t+1} \neq 0$. Note $\mathbf{V}^\top \mathbf{r}^{(1)} = \frac{1-\alpha}{1+\alpha}\mathbf{V}^\top \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top \mathbf{r}^{(0)} = \delta_1\boldsymbol{\Lambda}\mathbf{V}^\top \mathbf{r}^{(0)}$, we have

$$\mathbf{V}^\top \mathbf{r}^{(0)} = \mathbf{y}^{(0)} \quad \Rightarrow \quad \mathbf{V}^\top \mathbf{r}^{(1)} = \delta_1\mathbf{y}^{(1)} = \delta_1\boldsymbol{\Lambda}\mathbf{V}^\top \mathbf{r}^{(0)} = \delta_1\boldsymbol{\Lambda}\mathbf{y}^{(0)},$$

where it follows from $\delta_1 \neq 0$. As $\mathbf{y}^{(1)} = \boldsymbol{\Lambda}\mathbf{y}^{(0)}$, follow Equ. (24) of Lemma C.3, $y_i^{(t)}$ has the solution

$$y_i^{(t)} = \begin{cases} y_i^{(0)} \cos(\theta_i t) & |\lambda_i| < 1 \\ y_i^{(0)} \lambda_i^t & |\lambda_i| = 1 \end{cases} \leq \begin{cases} |y_i^{(0)}| & |\lambda_i| < 1 \\ |y_i^{(0)}| & |\lambda_i| = 1 \end{cases},$$

where $\theta_i = \arccos(\lambda_i)$. We can write down $\mathbf{r}^{(t)}$ in terms of $\mathbf{y}^{(t)}$

$$\mathbf{V}^\top \mathbf{r}^{(t)} = \delta_{1:t}\mathbf{y}^{(t)} = \delta_{1:t}\mathbf{Z}_t\mathbf{y}^{(0)},$$

where \mathbf{Z}_t has two possible forms

$$\mathbf{Z}_t = \begin{cases} \mathbf{diag}(1, \dots, \cos(\theta_1 t), \dots, (-1)^t) & \text{for bipartite graphs} \\ \mathbf{diag}(1, \dots, \cos(\theta_1 t), \dots, \cos(\theta_n t)) & \text{for non-bipartite graphs.} \end{cases}$$

Hence, $\|\mathbf{Z}_t\|_2 \leq 1$ and $\|\mathbf{Z}_t\mathbf{y}^{(0)}\|_2 \leq \|\mathbf{y}^{(0)}\|_2$. We have

$$\|\mathbf{r}^{(t)}\|_2 = \|\mathbf{V}^\top \mathbf{r}^{(t)}\|_2 \leq \delta_{1:t}\|\mathbf{y}^{(0)}\|_2 \leq 2 \left(\frac{1 - \sqrt{\alpha}}{1 + \sqrt{\alpha}} \right)^t \|\mathbf{b}\|_2,$$

where the last inequality follows Lemma C.5 and note $\mathbf{z}^{(0)} = \mathbf{V}^\top \mathbf{r}^{(0)}$. To meet the stop condition of $\{|r_u^{(t)}| < 2\alpha\epsilon\sqrt{d_u}/(1+\alpha), u \in \mathcal{V}\} = \emptyset$, it is sufficient to choose a minimal integer t such that

$$2 \left(\frac{1 - \sqrt{\alpha}}{1 + \sqrt{\alpha}} \right)^t \|\mathbf{b}\|_2 < \frac{2\alpha\epsilon}{(1+\alpha)\sqrt{d_s}}.$$

To see this, if the above inequality is satisfied, then note for any node u , we have

$$|r_u^{(t)}| \leq \|\mathbf{r}^{(t)}\|_\infty \leq \|\mathbf{r}^{(t)}\|_2 \leq 2 \left(\frac{1 - \sqrt{\alpha}}{1 + \sqrt{\alpha}} \right)^t \|\mathbf{b}\|_2 < \frac{2\alpha\epsilon}{(1+\alpha)\sqrt{d_s}} \leq \frac{2\alpha\epsilon\sqrt{d_u}}{(1+\alpha)},$$

where all nodes are inactive. So, it gives $t \ln \left(\frac{1 - \sqrt{\alpha}}{1 + \sqrt{\alpha}} \right) \leq \ln \frac{\epsilon}{2}$ by noticing $\|\mathbf{b}\|_2 = 2\alpha/((1+\alpha)\sqrt{d_s})$.

It indicates $t \geq \left\lceil \ln \frac{2}{\epsilon} / \ln \left(\frac{1 + \sqrt{\alpha}}{1 - \sqrt{\alpha}} \right) \right\rceil$. As $\frac{1}{\ln(1+x)} \leq \frac{1+x}{x}$ for $x > 0$, it is sufficient to choose t

$$t = \left\lceil \frac{1 + \sqrt{\alpha}}{2\sqrt{\alpha}} \ln \frac{2}{\epsilon} \right\rceil.$$

□

C.4 Residual Updates of LOCCH and Proof of Lemma 4.1

We propose the following local Chebyshev iteration procedure

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + (1 + \delta_{t:t+1})\mathbf{r}_{S_t}^{(t)} + \delta_{t:t+1}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{S_t}.$$

Our next Lemma is to expanding $(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{S_t}$

Lemma C.7. For $t \geq 1$ with initials $\mathbf{x}^{(0)} = \mathbf{0}$ and $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \mathbf{r}_{S_0}^{(0)}$, the local Chebyshev iterative is the following

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + (1 + \delta_{t:t+1})\mathbf{r}_{S_t}^{(t)} + \delta_{t:t+1}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{S_t}.$$

Denote $\Delta^{(t)} = \mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}$, then $\Delta^{(t)} = \sum_{j=0}^{t-1} \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^{t-1} \delta_{r:r+1} \mathbf{r}_{S_{j:t-1}}^{(j)} \right)$, where $\delta_{0:1} = 0$, $\mathcal{S}_{0:t} = \mathcal{S}_0 \cap \mathcal{S}_1 \cap \dots \cap \mathcal{S}_t$ and $\delta_{j:j+1} = \delta_j \delta_{j+1}$. We have the following

$$(1 + \delta_{t:t+1})\mathbf{r}_{S_t}^{(t)} + \delta_{t:t+1}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{S_t} = \sum_{j=0}^t \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^t \delta_{r:r+1} \mathbf{r}_{S_{j,t}}^{(j)} \right),$$

where $\mathcal{S}_{j,t} \triangleq \mathcal{S}_{j:t-1} \cap \bar{\mathcal{S}}_t$.

Proof. Recall we defined $\delta_0 = 0$ so that $\delta_{0:1} = 0$. We prove this lemma by induction.

For $t = 1$, note $\delta_{0:1} = 0$ and the support of $\mathbf{r}^{(0)}$ is $\mathcal{S}_0 = \text{supp}(\mathbf{r}^{(0)})$, then

$$\mathbf{x}^{(1)} - \mathbf{x}^{(0)} = (1 + \delta_{0:1})\mathbf{r}^{(0)}.$$

For $t = 2$, note $\mathcal{S}_{1:1} = \mathcal{S}_1$ and $\mathcal{S}_{0:1} = \mathcal{S}_0 \cap \mathcal{S}_1$ by our notation, then

$$\begin{aligned} \mathbf{x}^{(2)} - \mathbf{x}^{(1)} &= (1 + \delta_{1:2})\mathbf{r}_{S_{1:1}}^{(1)} + \delta_{1:2}(\mathbf{x}^{(1)} - \mathbf{x}^{(0)})_{S_1} \\ &= (1 + \delta_{1:2})\mathbf{r}_{S_{1:1}}^{(1)} + (1 + \delta_{0:1})\delta_{1:2}\mathbf{r}_{S_{0:1}}^{(0)}. \end{aligned}$$

For $t = 3$, one can build $\mathbf{x}^{(3)} - \mathbf{x}^{(2)}$ based on $\mathbf{x}^{(2)} - \mathbf{x}^{(1)}$ and recall $\mathcal{S}_{2:2} = \mathcal{S}_2$, $\mathcal{S}_{1:2} = \mathcal{S}_1 \cap \mathcal{S}_2$

$$\begin{aligned} \mathbf{x}^{(3)} - \mathbf{x}^{(2)} &= (1 + \delta_{2:3})\mathbf{r}_{S_{2:2}}^{(2)} + \delta_{2:3}(\mathbf{x}^{(2)} - \mathbf{x}^{(1)})_{S_2} \\ &= (1 + \delta_{2:3})\mathbf{r}_{S_{2:2}}^{(2)} + \delta_{2:3} \left((1 + \delta_{1:2})\mathbf{r}_{S_{1:1}}^{(1)} + (1 + \delta_{0:1})\delta_{1:2}\mathbf{r}_{S_{0:1}}^{(0)} \right)_{S_2} \\ &= (1 + \delta_{2:3})\mathbf{r}_{S_{2:2}}^{(2)} + (1 + \delta_{1:2})\delta_{2:3}\mathbf{r}_{S_{1:2}}^{(1)} + (1 + \delta_{0:1})\delta_{1:2}\delta_{2:3}\mathbf{r}_{S_{0:2}}^{(0)}. \end{aligned}$$

For $t = 4$, we continue to have

$$\begin{aligned}
\mathbf{x}^{(4)} - \mathbf{x}^{(3)} &= (1 + \delta_{3:4})\mathbf{r}_{\mathcal{S}_{3:3}}^{(3)} + \delta_{3:4}(\mathbf{x}^{(3)} - \mathbf{x}^{(2)})_{\mathcal{S}_{3:3}} \\
&= (1 + \delta_{3:4})\mathbf{r}_{\mathcal{S}_{3:3}}^{(3)} + \delta_{3:4}((1 + \delta_{2:3})\mathbf{r}_{\mathcal{S}_{2:2}}^{(2)} + (1 + \delta_{1:2})\delta_{2:3}\mathbf{r}_{\mathcal{S}_{1:2}}^{(1)} \\
&\quad + (1 + \delta_{0:1})\delta_{1:2}\delta_{2:3}\mathbf{r}_{\mathcal{S}_{0:2}}^{(0)})_{\mathcal{S}_{3:3}} \\
&= (1 + \delta_{3:4})\mathbf{r}_{\mathcal{S}_{3:3}}^{(3)} + (1 + \delta_{2:3})\delta_{3:4}\mathbf{r}_{\mathcal{S}_{2:3}}^{(2)} + (1 + \delta_{1:2})\delta_{3:4}\delta_{2:3}\mathbf{r}_{\mathcal{S}_{1:3}}^{(1)} \\
&\quad + (1 + \delta_{0:1})\delta_{1:2}\delta_{2:3}\delta_{3:4}\mathbf{r}_{\mathcal{S}_{0:3}}^{(0)} \\
(\mathbf{x}^{(4)} - \mathbf{x}^{(3)})_{\overline{\mathcal{S}}_4} &= ((1 + \delta_{3:4})\mathbf{r}_{\mathcal{S}_{3:3}}^{(3)} + \delta_{3:4}(\mathbf{x}^{(3)} - \mathbf{x}^{(2)})_{\mathcal{S}_{3:3}})_{\overline{\mathcal{S}}_4}.
\end{aligned}$$

By induction $t \geq 1$,

$$\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)} = \sum_{j=0}^{t-1} \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^{t-1} \delta_{r:r+1} \mathbf{r}_{\mathcal{S}_{j:t-1}}^{(j)} \right),$$

where the convention notation $\sum_{i=1}^0 \cdot = 0$ and $\prod_{j=i+1}^i \cdot = 1$. To verify the inductive step, consider for $t + 1$, we have

$$\begin{aligned}
\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)} &= (1 + \delta_{t:t+1})\mathbf{r}_{\mathcal{S}_t}^{(t)} + \delta_{t:t+1}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\mathcal{S}_t} \\
&= (1 + \delta_{t:t+1})\mathbf{r}_{\mathcal{S}_t}^{(t)} + \delta_{t:t+1}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\mathcal{S}_t} \\
&= (1 + \delta_{t:t+1})\mathbf{r}_{\mathcal{S}_t}^{(t)} + \delta_{t:t+1} \left(\sum_{j=0}^{t-1} \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^{t-1} \delta_{r:r+1} \mathbf{r}_{\mathcal{S}_{j:t-1}}^{(j)} \right) \right)_{\mathcal{S}_t} \\
&= \sum_{j=0}^t \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^t \delta_{r:r+1} \mathbf{r}_{\mathcal{S}_{j:t}}^{(j)} \right).
\end{aligned}$$

To see the second equation, note

$$\begin{aligned}
&(1 + \delta_{t:t+1})\mathbf{r}_{\overline{\mathcal{S}}_t}^{(t)} + \delta_{t:t+1}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\overline{\mathcal{S}}_t} \\
&= (1 + \delta_{t:t+1})\mathbf{r}_{\overline{\mathcal{S}}_t}^{(t)} + \sum_{j=0}^{t-1} \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^t \delta_{r:r+1} \mathbf{r}_{\mathcal{S}_{j:t-1} \cap \overline{\mathcal{S}}_t}^{(j)} \right) \\
&= \sum_{j=0}^t \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^t \delta_{r:r+1} \mathbf{r}_{\mathcal{S}_{j,t}}^{(j)} \right),
\end{aligned}$$

where recall we denote $\mathcal{S}_{j,t} \triangleq \mathcal{S}_{j:t-1} \cap \overline{\mathcal{S}}_t$. □

Lemma C.8 (Local Chebyshev updates). *Given the updates of $\mathbf{x}^{(t+1)}$ as defined by LOCCH in (10), we have the following local updates*

$$\begin{cases} \mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + (1 + \delta_{t:t+1})\mathbf{r}_{\mathcal{S}_t}^{(t)} + \delta_{t:t+1}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\mathcal{S}_t} \\ \mathbf{r}^{(t+1)} - 2\delta_{t+1}\mathbf{W}\mathbf{r}^{(t)} + \delta_{t:t+1}\mathbf{r}^{(t-1)} = \sum_{j=0}^t \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^t \delta_{r:r+1} \mathbf{Q}\mathbf{r}_{\mathcal{S}_{j,t}}^{(j)} \right) \end{cases} \quad (28)$$

Proof. We only need to show the second equation of (28). The residual of LOCCH updates is

$$\begin{aligned}
\mathbf{r}^{(t+1)} &= \mathbf{b} - \mathbf{Q}\mathbf{x}^{(t+1)} \\
\mathbf{r}^{(t+1)} &= \mathbf{b} - \mathbf{Q}(\mathbf{x}^{(t)} + (1 + \delta_{t:t+1})\mathbf{r}_{\mathcal{S}_t}^{(t)} + \delta_{t:t+1}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\mathcal{S}_t}) \\
&= \mathbf{r}^{(t)} - (1 + \delta_{t:t+1})\mathbf{Q}\mathbf{r}_{\mathcal{S}_t}^{(t)} - \delta_{t:t+1}\mathbf{Q}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\mathcal{S}_t} \\
&\quad \underbrace{\mathbf{r}^{(t+1)} + (1 + \delta_{t:t+1})\mathbf{Q}\mathbf{r}^{(t)} + \delta_{t:t+1}\mathbf{Q}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}) - \mathbf{r}^{(t)}}_{\mathbf{u}} \\
&= \underbrace{(1 + \delta_{t:t+1})\mathbf{Q}\mathbf{r}_{\mathcal{S}_t}^{(t)} + \delta_{t:t+1}\mathbf{Q}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\mathcal{S}_t}}_{\text{small noisy part}}.
\end{aligned}$$

Note $\mathbf{Q}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}) = \mathbf{b} - \mathbf{Q}\mathbf{x}^{(t-1)} - (\mathbf{b} - \mathbf{Q}\mathbf{x}^{(t)}) = \mathbf{r}^{(t-1)} - \mathbf{r}^{(t)}$ and then \mathbf{u} becomes

$$\begin{aligned}
\mathbf{u} &= \mathbf{r}^{(t+1)} + (1 + \delta_{t:t+1})\mathbf{Q}\mathbf{r}^{(t)} + \delta_{t:t+1}(\mathbf{r}^{(t-1)} - \mathbf{r}^{(t)}) - \mathbf{r}^{(t)} \\
&= \mathbf{r}^{(t+1)} - 2\delta_{t:t+1}\mathbf{W}\mathbf{r}^{(t)} + \delta_{t:t+1}\mathbf{r}^{(t-1)},
\end{aligned}$$

where the last equality is due to $(1 + \delta_{t:t+1})\mathbf{Q}\mathbf{r}^{(t)} = (1 + \delta_{t:t+1})\mathbf{r}^{(t)} - 2\delta_{t:t+1}\mathbf{W}\mathbf{r}^{(t)}$ by noticing $(1 + \delta_t\delta_{t+1})\frac{1-\alpha}{1+\alpha} = 2\delta_{t:t+1}$ in Lemma C.5. Hence, we have the second equation. To see the noisy part, note by Lemma C.7

$$\begin{aligned}
(1 + \delta_{t:t+1})\mathbf{r}_{\mathcal{S}_t}^{(t)} + \delta_{t:t+1}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\mathcal{S}_t} &= (1 + \delta_{t:t+1})\mathbf{r}_{\mathcal{S}_t}^{(t)} + \sum_{j=0}^{t-1} \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^t \delta_{r:r+1} \mathbf{r}_{\mathcal{S}_{j,t}}^{(j)} \right) \\
(1 + \delta_{t:t+1})\mathbf{Q}\mathbf{r}_{\mathcal{S}_t}^{(t)} + \delta_{t:t+1}\mathbf{Q}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\mathcal{S}_t} &= \sum_{j=0}^t \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^t \delta_{r:r+1} \mathbf{Q}\mathbf{r}_{\mathcal{S}_{j,t}}^{(j)} \right).
\end{aligned}$$

□

Lemma 4.1 (Residual updates of LOCCH). *Given $t \geq 1$, $\mathbf{x}^{(0)} = \mathbf{0}$, $\mathbf{x}^{(1)} = \mathbf{x}^0 + \mathbf{r}_{\mathcal{S}_0}^{(0)}$. The residual $\mathbf{r}^{(t)}$ of LOCCH defined in Equ. (28) satisfies*

$$\mathbf{V}^\top \mathbf{r}^{(t)} = \delta_{1:t} \mathbf{Z}_t \mathbf{V}^\top \mathbf{r}^{(0)} + \delta_{1:t} t \mathbf{u}_{0,t} + 2 \sum_{k=1}^{t-1} \delta_{k+1:t} (t-k) \mathbf{u}_{k,t}, \quad (29)$$

where

$$\begin{aligned}
\mathbf{u}_{k,t} &= \begin{cases} \sum_{j=1}^{t-1} \delta_{2:j} \mathbf{H}_{j,t} \left(\mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{0,j}}^{(0)} / t & \text{if } k = 0 \\ \sum_{j=k}^{t-1} \left(\delta_{k+1:j} \mathbf{H}_{j,t} \left(\frac{1+\alpha}{1-\alpha} \mathbf{I} - \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{k,j}}^{(k)} \right) / (t-k) & \text{if } k \geq 1, \end{cases} \\
\mathbf{Z}_t &= \begin{cases} \text{diag}(1, \dots, \cos(\theta_1 t), \dots, (-1)^t) & \text{for bipartite graphs} \\ \text{diag}(1, \dots, \cos(\theta_1 t), \dots, \cos(\theta_n t)) & \text{for non-bipartite graphs,} \end{cases} \\
\mathbf{H}_{k,t} &= \begin{cases} \text{diag} \left(t-k, \dots, \frac{\sin(\theta_i(t-k))}{\sin \theta_i}, \dots, (-1)^{t-k-1} (t-k) \right) & \text{for bipartite graphs} \\ \text{diag} \left(t-k, \dots, \frac{\sin(\theta_i(t-k))}{\sin \theta_i}, \dots, \frac{\sin(\theta_n(t-k))}{\sin \theta_n} \right) & \text{for non-bipartite graphs.} \end{cases}
\end{aligned}$$

Proof. We first decompose the residual equation in (28) as

$$\begin{aligned}
\mathbf{V}^\top \mathbf{r}^{(t+1)} - 2\delta_{t+1} \mathbf{\Lambda} \mathbf{V}^\top \mathbf{r}^{(t)} + \delta_{t+1} \mathbf{V}^\top \mathbf{r}^{(t-1)} \\
= \underbrace{\sum_{j=0}^t \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^t (\delta_{r:r+1}) \left(\mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{j,t}}^{(j)} \right)}_{\mathbf{f}^{(t)}}.
\end{aligned}$$

Define $\mathbf{V}^\top \mathbf{r}^{(t)} = \delta_{1:t} \mathbf{y}^{(t)}$ and $\mathbf{V}^\top \mathbf{r}^{(0)} = \delta_{1:0} \mathbf{y}^{(0)} = \mathbf{y}^{(0)}$ by default. Then we have

$$\delta_{1:t+1} \mathbf{y}^{(t+1)} - 2\delta_{1:t+1} \mathbf{\Lambda} \mathbf{y}^{(t)} + \delta_{1:t+1} \mathbf{y}^{(t-1)} = \mathbf{f}^{(t)} \Rightarrow \mathbf{y}^{(t+1)} - 2\mathbf{\Lambda} \mathbf{y}^{(t)} + \mathbf{y}^{(t-1)} = \frac{\mathbf{f}^{(t)}}{\delta_{1:t+1}}.$$

Note $\mathbf{V}^\top \mathbf{r}^{(1)} = \delta_{1:1} \mathbf{y}^{(1)} = \mathbf{V}^\top \delta_{1:1} \mathbf{W} \mathbf{r}^{(0)} = \delta_{1:1} \mathbf{\Lambda} \mathbf{y}^{(0)}$, which indicates $\mathbf{y}^{(1)} = \mathbf{\Lambda} \mathbf{y}^{(0)}$. Then, by the Lemma C.4, each $y_i^{(t)}$ has the solution

$$y_i^{(t)} = \begin{cases} y_i^{(0)} + \sum_{k=1}^{t-1} (t-k) f_i^{(k)} / (\delta_{1:k+1}) & \text{if } \lambda_i = 1 \\ \cos(\theta_i t) y_i^{(0)} + \sum_{k=1}^{t-1} \frac{\sin(\theta_i(t-k))}{\sin \theta_i} f_i^{(k)} / (\delta_{1:k+1}) & \text{if } |\lambda_i| < 1 \\ (-1)^t y_i^{(0)} + \sum_{k=1}^{t-1} (-1)^{t-k-1} (t-k) f_i^{(k)} / (\delta_{1:k+1}) & \text{if } \lambda_i = -1. \end{cases}$$

Use \mathbf{Z}_t and $\mathbf{H}_{k,t}$, we write the solution of the second-order difference equation as

$$\mathbf{y}^{(t)} = \mathbf{Z}_t \mathbf{y}^{(0)} + \sum_{k=1}^{t-1} \mathbf{H}_{k,t} \mathbf{f}^{(k)} / (\delta_{1:k+1})$$

$$\mathbf{V}^\top \mathbf{r}^{(t)} = \delta_{1:t} \mathbf{y}^{(t)} = \delta_{1:t} \mathbf{Z}_t \mathbf{V}^\top \mathbf{r}^{(0)} + \delta_{1:t} \sum_{k=1}^{t-1} \mathbf{H}_{k,t} \mathbf{f}^{(k)} / (\delta_{1:k+1})$$

$$\mathbf{V}^\top \mathbf{r}^{(t)} = \delta_{1:t} \mathbf{Z}_t \mathbf{V}^\top \mathbf{r}^{(0)} + \sum_{k=1}^{t-1} \delta_{k+2:t} \mathbf{H}_{k,t} \sum_{j=0}^k \left((1 + \delta_{j:j+1}) \prod_{r=j+1}^k (\delta_{r:r+1}) \left(\mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{j,k}}^{(j)} \right).$$

Note $1 + \delta_{j:j+1} = 2\delta_{j+1} \frac{1+\alpha}{1-\alpha}$, $j \geq 1$, then $(1 + \delta_{j:j+1}) \prod_{r=j+1}^k (\delta_{r:r+1}) = 2 \frac{1+\alpha}{1-\alpha} \delta_{j+1:k} \delta_{j+1:k+1}$. Then, we have

$$\begin{aligned} \mathbf{V}^\top \mathbf{r}^{(t)} &= \delta_{1:t} \mathbf{Z}_t \mathbf{V}^\top \mathbf{r}^{(0)} + \sum_{k=1}^{t-1} \delta_{k+2:t} \mathbf{H}_{k,t} \left(\delta_{2:k} \delta_{1:k+1} \left(\mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{0,k}}^{(0)} \right) \\ &\quad + 2 \sum_{k=1}^{t-1} \delta_{k+2:t} \mathbf{H}_{k,t} \sum_{j=1}^k \left(\delta_{j+1:k} \delta_{j+1:k+1} \left(\frac{1+\alpha}{1-\alpha} \mathbf{I} - \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{j,k}}^{(j)} \right) \\ &= \delta_{1:t} \mathbf{Z}_t \mathbf{V}^\top \mathbf{r}^{(0)} + \delta_{1:t} \sum_{k=1}^{t-1} \delta_{2:k} \mathbf{H}_{k,t} \left(\mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{0,k}}^{(0)} \\ &\quad + 2 \sum_{k=1}^{t-1} \delta_{k+2:t} \mathbf{H}_{k,t} \sum_{j=1}^k \left(\delta_{j+1:k} \delta_{j+1:k+1} \left(\frac{1+\alpha}{1-\alpha} \mathbf{I} - \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{j,k}}^{(j)} \right), \end{aligned}$$

where the last term can be expanded as

$$\begin{aligned} &\sum_{k=1}^{t-1} \delta_{k+2:t} \mathbf{H}_{k,t} \sum_{j=1}^k \left(\delta_{j+1:k} \delta_{j+1:k+1} \underbrace{\left(\frac{1+\alpha}{1-\alpha} \mathbf{I} - \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{j,k}}^{(j)}}_{\mathbf{w}_{\mathcal{S}_{j,k}}^{(j)}} \right) = \\ &\delta_{3:t} \mathbf{H}_{1,t} \delta_{2:1} \delta_{2:2} \mathbf{w}_{\mathcal{S}_{1,1}}^{(1)} + \\ &\delta_{4:t} \mathbf{H}_{2,t} \delta_{2:2} \delta_{2:3} \mathbf{w}_{\mathcal{S}_{1,2}}^{(1)} + \delta_{4:t} \mathbf{H}_{2,t} \delta_{3:2} \delta_{3:3} \mathbf{w}_{\mathcal{S}_{2,2}}^{(2)} + \\ &\delta_{5:t} \mathbf{H}_{3,t} \delta_{2:3} \delta_{2:4} \mathbf{w}_{\mathcal{S}_{1,3}}^{(1)} + \delta_{5:t} \mathbf{H}_{3,t} \delta_{3:3} \delta_{3:4} \mathbf{w}_{\mathcal{S}_{2,3}}^{(2)} + \delta_{5:t} \mathbf{H}_{3,t} \delta_{4:3} \delta_{4:4} \mathbf{w}_{\mathcal{S}_{3,3}}^{(3)} + \\ &\delta_{6:t} \mathbf{H}_{4,t} \delta_{2:4} \delta_{2:5} \mathbf{w}_{\mathcal{S}_{1,4}}^{(1)} + \delta_{6:t} \mathbf{H}_{4,t} \delta_{3:4} \delta_{3:5} \mathbf{w}_{\mathcal{S}_{2,4}}^{(2)} + \delta_{6:t} \mathbf{H}_{4,t} \delta_{4:4} \delta_{4:5} \mathbf{w}_{\mathcal{S}_{3,4}}^{(3)} + \delta_{6:t} \mathbf{H}_{4,t} \delta_{5:4} \delta_{5:5} \mathbf{w}_{\mathcal{S}_{4,4}}^{(4)} + \\ &\delta_{7:t} \mathbf{H}_{5,t} \delta_{2:5} \delta_{2:6} \mathbf{w}_{\mathcal{S}_{1,5}}^{(1)} + \delta_{7:t} \mathbf{H}_{5,t} \delta_{3:5} \delta_{3:6} \mathbf{w}_{\mathcal{S}_{2,5}}^{(2)} + \delta_{7:t} \mathbf{H}_{5,t} \delta_{4:5} \delta_{4:6} \mathbf{w}_{\mathcal{S}_{3,5}}^{(3)} + \delta_{7:t} \mathbf{H}_{5,t} \delta_{5:5} \delta_{5:6} \mathbf{w}_{\mathcal{S}_{4,5}}^{(4)} \\ &\quad + \delta_{7:t} \mathbf{H}_{5,t} \delta_{6:5} \delta_{6:6} \mathbf{w}_{\mathcal{S}_{5,5}}^{(5)} + \\ &\vdots \\ &= \sum_{k=1}^{t-1} \delta_{k+1:t} \sum_{j=k}^{t-1} \left(\delta_{k+1:j} \mathbf{H}_{j,t} \left(\frac{1+\alpha}{1-\alpha} \mathbf{I} - \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{k,j}}^{(k)} \right). \end{aligned}$$

Here, we denote $\delta_{k+1:k} = 1$. The final iterative update is

$$\begin{aligned} \mathbf{V}^\top \mathbf{r}^{(t)} &= \delta_{1:t} \mathbf{Z}_t \mathbf{V}^\top \mathbf{r}^{(0)} + \underbrace{\delta_{1:t} t \sum_{j=1}^{t-1} \delta_{2:j} \mathbf{H}_{j,t} \left(\mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{S_{0,j}}^{(0)}}_{\mathbf{u}_{0,t}} / t \\ &+ 2 \sum_{k=1}^{t-1} \delta_{k+1:t} (t-k) \underbrace{\sum_{j=k}^{t-1} \left(\delta_{k+1:j} \mathbf{H}_{j,t} \left(\frac{1+\alpha}{1-\alpha} \mathbf{I} - \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{S_{k,j}}^{(k)} \right)}_{\mathbf{u}_{k,t}} / (t-k). \end{aligned}$$

□

C.5 Convergence of LOCCH and Proof of Theorem 4.2

Corollary C.9. *Let β_k be lower bound of residual reduction satisfies $\|\mathbf{u}_{k,t}\|_2 \leq \beta_k \|\mathbf{r}^{(k)}\|_2$, then the upper bound of $\|\mathbf{r}^{(t)}\|_2$ can be characterized as*

$$\|\mathbf{r}^{(t)}\|_2 \leq \delta_{1:t} \prod_{j=0}^{t-1} (1 + \beta_j) y_t, \text{ where } y_{t+1} - 2y_t + \frac{y_{t-1}}{(1 + \beta_{t-1})(1 + \beta_t)} = 0, \quad (30)$$

where $y_0 = y_1 = \|\mathbf{r}^0\|_2$.

Proof. Since $\|\mathbf{u}_{k,t}\|_2 \leq \beta_k \|\mathbf{r}^{(k)}\|_2$, the final iterative updates (29) can be bounded as

$$\begin{aligned} \mathbf{V}^\top \mathbf{r}^{(t)} &= \delta_{1:t} \mathbf{Z}_t \mathbf{V}^\top \mathbf{r}^{(0)} + \delta_{1:t} t \mathbf{u}_{0,t} + 2 \sum_{k=1}^{t-1} \delta_{k+1:t} (t-k) \mathbf{u}_{k,t} \\ \|\mathbf{r}^{(t)}\|_2 &\leq \delta_{1:t} \|\mathbf{r}^{(0)}\|_2 + \delta_{1:t} t \beta_0 \|\mathbf{r}^{(0)}\|_2 + 2 \sum_{k=1}^{t-1} \delta_{k+1:t} (t-k) \beta_k \|\mathbf{r}^{(k)}\|_2 \\ \|\mathbf{r}^{(t)}\|_2 - 2 \sum_{k=1}^{t-1} \delta_{k+1:t} (t-k) \beta_k \|\mathbf{r}^{(k)}\|_2 &\leq \delta_{1:t} (1 + t\beta_0) \|\mathbf{r}^{(0)}\|_2, \end{aligned} \quad (31)$$

where $t = 0, 1, \dots, T$. These $T + 1$ (including a trivial one where $\|\mathbf{r}^{(0)}\|_2 \leq \|\mathbf{r}^{(0)}\|_2$) inequalities shown in Equ. (31) form a system of linear inequality matrix as

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ -z_{21} & 1 & 0 & \cdots & 0 \\ -z_{31} & -z_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -z_{T1} & -z_{T2} & -z_{T3} & \cdots & 1 \end{pmatrix}}_{\mathbf{I} - \mathbf{Z}_L} \begin{pmatrix} \|\mathbf{r}^{(1)}\|_2 \\ \|\mathbf{r}^{(2)}\|_2 \\ \|\mathbf{r}^{(3)}\|_2 \\ \vdots \\ \|\mathbf{r}^{(T-1)}\|_2 \end{pmatrix} \leq \begin{pmatrix} \delta_{1:1} (1 + 1\beta_0) \|\mathbf{r}^{(0)}\|_2 \\ \delta_{1:2} (1 + 2\beta_0) \|\mathbf{r}^{(0)}\|_2 \\ \delta_{1:3} (1 + 3\beta_0) \|\mathbf{r}^{(0)}\|_2 \\ \vdots \\ \delta_{1:T} (1 + T\beta_0) \|\mathbf{r}^{(0)}\|_2 \end{pmatrix} := \mathbf{c},$$

where $(\mathbf{Z}_L)_{tk} = 2\delta_{k+1:t} (t-k)\beta_k$ for $t = 2, 3, \dots, T$ and $k = 1, 2, \dots, t-1$. Assume that $\mathbf{N} \in \mathbb{R}^{T \times T}$ is a strictly lower triangular matrix, then we know the established formula $(\mathbf{I} + \mathbf{N})^{-1} = \mathbf{I} + \sum_{k=1}^{T-1} (-1)^k \mathbf{N}^k$. Hence, we have the following

$$(\mathbf{I} - \mathbf{Z}_L)^{-1} = \mathbf{I} + \sum_{k=1}^{T-1} \mathbf{Z}_L^k.$$

Given that $(\mathbf{I} - \mathbf{Z}_L)^{-1} \geq \mathbf{0}$, then we obtain an upper-bound of
$$\begin{pmatrix} \|\mathbf{r}^{(1)}\|_2 \\ \|\mathbf{r}^{(2)}\|_2 \\ \vdots \\ \|\mathbf{r}^{(T)}\|_2 \end{pmatrix} \leq \mathbf{z} \triangleq (\mathbf{I} - \mathbf{Z}_L)^{-1} \mathbf{c}.$$

It leads to the following new second-order difference equation

$$z_t - 2 \sum_{k=1}^{t-1} \delta_{k+1:t} (t-k) \beta_k z_k = \delta_{1:t} (1 + t\beta_0) z_0, \quad \text{for } t = 1, 2, \dots, T,$$

where the initial value of $z_0 = \|\mathbf{r}^{(0)}\|_2$. Following the argument in Theorem 1 of Golub & Overton [18], we construct a second-order homogeneous equation for z_t as

$$\begin{aligned} z_t &= \delta_{1:t} (1 + t\beta_0) z_0 + 2 \sum_{k=1}^{t-1} \delta_{k+1:t} (t-k) \beta_k z_k \\ z_{t+1} &= \delta_{1:t+1} (1 + (t+1)\beta_0) z_0 + 2 \sum_{k=1}^t \delta_{k+1:t+1} (t+1-k) \beta_k z_k \\ \delta_{t+1} z_t &= \delta_{1:t+1} (1 + t\beta_0) z_0 + 2 \sum_{k=1}^t \delta_{k+1:t+1} (t-k) \beta_k z_k \\ z_{t+1} - \delta_{t+1} z_t &= \delta_{1:t+1} \beta_0 z_0 + 2 \sum_{k=1}^t \delta_{k+1:t+1} \beta_k z_k, \end{aligned} \quad (32)$$

where Equ. (32) is obtained by the difference between the second equation and the third equation. Similarly,

$$\begin{aligned} z_{t-1} &= \delta_{1:t-1} (1 + (t-1)\beta_0) z_0 + 2 \sum_{k=1}^{t-2} \delta_{k+1:t-1} (t-k-1) \beta_k z_k \\ \delta_{t:t+1} z_{t-1} &= \delta_{1:t+1} (1 + (t-1)\beta_0) z_0 + 2 \sum_{k=1}^{t-2} \delta_{k+1:t+1} (t-k-1) \beta_k z_k \\ \delta_{t+1} z_t - \delta_{t:t+1} z_{t-1} &= \delta_{1:t+1} \beta_0 z_0 + 2 \sum_{k=1}^{t-1} \delta_{k+1:t+1} \beta_k z_k, \end{aligned} \quad (33)$$

where Equ. (33) is obtained by the difference of the first two. Hence, Equ. (32) – (33) gives us

$$z_{t+1} - 2(1 + \beta_t) \delta_{t+1} z_t + \delta_{t:t+1} z_{t-1} = 0, \quad \text{for } t = 1, 2, \dots, T,$$

where two initials are $z_0 = \|\mathbf{r}^{(0)}\|_0$ and $z_1 = \delta_1 (1 + \beta_0) \|\mathbf{r}^{(0)}\|_2$. Let $z_t = \delta_{1:t} \hat{z}_t$, then

$$\hat{z}_{t+1} - 2(1 + \beta_t) \hat{z}_t + \hat{z}_{t-1} = 0,$$

where two initials are $\hat{z}_0 = z_0 = \|\mathbf{r}^{(0)}\|_0$ and $\hat{z}_1 = (1 + \beta_0) \|\mathbf{r}^{(0)}\|_2$. We finish the proof by setting $\prod_{j=0}^{t-1} (1 + \beta_j) y_t = \hat{z}_t$. \square

Remark C.10. Key points of the above proof strategy largely follow from Golub & Overton [18]. However, different from the original technique, we generalize the strategy to a parameterized version.

Lemma C.11. *Given $\beta_j \geq 0$, the following second-order difference equation*

$$x_{t+1} - 2(1 + \beta_t) x_t + x_{t-1} = 0.$$

has the following solution

$$x_t = \prod_{j=0}^{t-1} (1 + \beta_j) y_t,$$

where $y_{t+1} - 2y_t + \frac{y_{t-1}}{(1+\beta_{t-1})(1+\beta_t)} = 0$ with $y_0 = x_0$ and $y_1 = x_1/(1 + \beta_0)$.

Proof. Assume $x_t = \left(-\frac{1}{2}\right)^t \prod_{j=0}^{t-1} (-2(1 + \beta_j)) y_t$. Then, following the equation, we have

$$\begin{aligned} & \left(-\frac{1}{2}\right)^{t+1} \prod_{j=0}^t (-2(1 + \beta_j)) y_{t+1} \\ & - 2(1 + \beta_t) \left(-\frac{1}{2}\right)^t \prod_{j=0}^{t-1} (-2(1 + \beta_j)) y_t + \left(-\frac{1}{2}\right)^{t-1} \prod_{j=0}^{t-2} (-2(1 + \beta_j)) y_{t-1} = 0. \end{aligned}$$

Since $\beta_j \geq 0$, the term $\prod_{j=0}^t (-2(1 + \beta_j)) \neq 0$, we divide it on both sides to have

$$\left(-\frac{1}{2}\right)^{t+1} y_{t+1} + \left(-\frac{1}{2}\right)^t y_t + \left(-\frac{1}{2}\right)^{t-1} \frac{1}{4(1 + \beta_{t-1})(1 + \beta_t)} y_{t-1} = 0.$$

Hence, it is simplified into $y_{t+1} - 2y_t + \frac{y_{t-1}}{(1 + \beta_{t-1})(1 + \beta_t)} = 0$. To make a simplification on x_t , we prove the lemma. \square

Theorem 4.2 (Runtime bound of LOCCH). *Given the configuration $\theta = (\alpha, \epsilon, s, \mathcal{G})$ with $\alpha \in (0, 1)$ and $\epsilon \leq 1/d_s$ and let $\mathbf{r}^{(T)}$ and $\mathbf{x}^{(T)}$ be returned by LOCCH defined in (10) for solving Equ. (3). For $t \geq 1$, the residual magnitude $\|\mathbf{r}^{(t)}\|_2$ has the following convergence bound*

$$\|\mathbf{r}^{(t)}\|_2 \leq \delta_{1:t} \prod_{j=0}^{t-1} (1 + \beta_j) y_t,$$

where y_t is a sequence of positive numbers solving $y_{t+1} - 2y_t + y_{t-1}/((1 + \beta_{t-1})(1 + \beta_t)) = 0$ with $y_0 = y_1 = \|\mathbf{r}^{(0)}\|_2$. Suppose the geometric mean $\bar{\beta}_t \triangleq (\prod_{j=0}^{t-1} (1 + \beta_j))^{1/t}$ of β_t be such that $\bar{\beta}_t = 1 + \frac{c\sqrt{\alpha}}{1 - \sqrt{\alpha}}$ where $c \in [0, 2)$. There exists a real implementation of (9) such that the runtime $\mathcal{T}_{\text{LocCH}}$ is bounded by

$$\mathcal{T}_{\text{LocCH}} \leq \Theta \left(\frac{(1 + \sqrt{\alpha}) \text{vol}(\mathcal{S}_T)}{\sqrt{\alpha}(2 - c)} \ln \frac{2y_T}{\epsilon} \right).$$

Proof. The convergence bound of $\mathbf{r}^{(t)}$ directly follows from Corollary C.9. Since we assume that there exists $c \in [0, 2)$ such that $\prod_{j=0}^{t-1} (1 + \beta_j) \leq \left(1 + \frac{c\sqrt{\alpha}}{1 - \sqrt{\alpha}}\right)^t$. Then multiplying both sides by $\tilde{\alpha}^t$, we have

$$\tilde{\alpha}^t \prod_{j=0}^{t-1} (1 + \beta_j) \leq \left(1 - \frac{(2 - c)\sqrt{\alpha}}{1 + \sqrt{\alpha}}\right)^t.$$

Then we have

$$\begin{aligned} \|\mathbf{r}^{(t)}\|_2 & \leq \delta_{1:t} \prod_{j=0}^{t-1} (1 + \beta_j) y_t \stackrel{\delta_{1:t} \leq 2\tilde{\alpha}^t}{\leq} 2\tilde{\alpha}^t \bar{\beta}_t^t y_t \leq \epsilon \\ t \ln \left(\frac{1 - \sqrt{\alpha}}{1 + \sqrt{\alpha}} \left(\prod_{j=0}^{t-1} (1 + \beta_j) \right)^{1/t} \right) & \leq \ln \left(\frac{\epsilon}{2y_t} \right) \\ t & \geq \left\lceil \frac{\ln \left(\frac{2y_t}{\epsilon} \right)}{\ln \left(\frac{1 + \sqrt{\alpha}}{(1 - \sqrt{\alpha}) \bar{\beta}_t} \right)} \right\rceil \end{aligned}$$

Since $\bar{\beta}_t = (\prod_{j=0}^{t-1} (1 + \beta_j))^{1/t}$, and by using $\frac{1+x}{x} \geq \frac{1}{\ln(1+x)}$ and letting $x = \frac{1 + \sqrt{\alpha}}{(1 - \sqrt{\alpha}) \bar{\beta}_t} - 1 > 0$, then t can be lower bounded further by

$$t \geq \left\lceil \frac{1 + \sqrt{\alpha}}{1 + \sqrt{\alpha} - (1 - \sqrt{\alpha}) \bar{\beta}_t} \ln \left(\frac{2y_t}{\epsilon} \right) \right\rceil \geq \left\lceil \frac{\ln \left(\frac{2y_t}{\epsilon} \right)}{\ln \left(\frac{1 + \sqrt{\alpha}}{(1 - \sqrt{\alpha}) \bar{\beta}_t} \right)} \right\rceil.$$

Since we assumed $\beta_t = (1 + \frac{c\sqrt{\alpha}}{1-\sqrt{\alpha}})$, which means $1 \leq \bar{\beta}_t = (1 + \frac{c\sqrt{\alpha}}{1-\sqrt{\alpha}})$, so $\bar{\beta}_t \in [1, \frac{1+\sqrt{\alpha}}{1-\sqrt{\alpha}}]$. Then, we find such an upper bound of t so that LOCCH converges.

$$t = \left\lceil \frac{1 + \sqrt{\alpha}}{1 + \sqrt{\alpha} - (1 - \sqrt{\alpha})\bar{\beta}_t} \ln \left(\frac{2yt}{\epsilon} \right) \right\rceil = \left\lceil \frac{1 + \sqrt{\alpha}}{(2-c)\sqrt{\alpha}} \ln \left(\frac{2yt}{\epsilon} \right) \right\rceil.$$

□

C.6 Implementation of LOCCH

We present the implementation of LOCCH as follows: Recall the sequence $\delta_{t+1} = (2\frac{1+\alpha}{1-\alpha} - \delta_t)^{-1}$, $t = 1, 2, \dots$ with $\delta_1 = \frac{1-\alpha}{1+\alpha}$. Denote $\tilde{\mathbf{x}}^{(t)} \triangleq \mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}$, $\mathbf{\Delta}^{(t)} := (1 + \delta_{t:t+1})\mathbf{r}^{(t)} + \delta_{t:t+1}\tilde{\mathbf{x}}^{(t)}$, we have

$$\begin{aligned} \mathbf{x}^{(t+1)} &= \mathbf{x}^{(t)} + (1 + \delta_{t:t+1})\mathbf{r}_{S_t}^{(t)} + \delta_{t:t+1}\tilde{\mathbf{x}}_{S_t}^{(t)} = \mathbf{x}^{(t)} + \mathbf{\Delta}_{S_t}^{(t)} \\ \mathbf{r}^{(t+1)} &= \mathbf{b} - \mathbf{Q} \left(\mathbf{x}^{(t)} + (1 + \delta_{t:t+1})\mathbf{r}_{S_t}^{(t)} + \delta_{t:t+1}\tilde{\mathbf{x}}_{S_t}^{(t)} \right) = \mathbf{r}^{(t)} - \mathbf{Q}\mathbf{\Delta}_{S_t}^{(t)} \\ \tilde{\mathbf{x}}^{(t+1)} &= \tilde{\mathbf{x}}^{(t)} + \mathbf{\Delta}_{S_t}^{(t)} - \mathbf{\Delta}_{S_{t-1}}^{(t-1)}. \end{aligned}$$

- When $t = 0$, we have $\mathbf{x}^{(0)} = \mathbf{0}$, $\mathbf{r}^{(0)} = \mathbf{b}$, $\tilde{\mathbf{x}}^{(0)} = \mathbf{0}$, $\mathbf{\Delta}^{(0)} = \mathbf{r}^{(0)}$.
- When $t = 1$, we have $\mathbf{x}^{(1)} = \mathbf{r}_{S_0}^{(0)}$, $\mathbf{r}^{(1)} = \frac{1-\alpha}{1+\alpha}\mathbf{W}\mathbf{\Delta}_{S_0}^{(0)}$, $\tilde{\mathbf{x}}^{(1)} = \mathbf{r}_{S_0}^{(0)}$, $\mathbf{\Delta}^{(1)} = (1 + \delta_{1:2})\mathbf{r}^{(1)} + \delta_{1:2}\tilde{\mathbf{x}}^{(1)}$.
- When $t \geq 1$, we can recursively calculate the following vectors

$$\begin{aligned} \mathbf{x}^{(t+1)} &= \mathbf{x}^{(t)} + \mathbf{\Delta}_{S_t}^{(t)} \\ \mathbf{r}^{(t+1)} &= \mathbf{r}^{(t)} - \mathbf{\Delta}_{S_t}^{(t)} + \frac{1-\alpha}{1+\alpha}\mathbf{W}\mathbf{\Delta}_{S_t}^{(t)} \\ \tilde{\mathbf{x}}^{(t+1)} &= \tilde{\mathbf{x}}^{(t)} + \mathbf{\Delta}_{S_t}^{(t)} - \mathbf{\Delta}_{S_{t-1}}^{(t-1)}. \end{aligned}$$

Therefore, at per-iteration, we only need to save sub-vectors $\mathbf{\Delta}_{S_t}$ and $\mathbf{\Delta}_{S_{t-1}}$ and update \mathbf{x} locally.

D Local Heavy-Ball Method - LOCHB

D.1 Standard HB and Proof Theorem D.2

Lemma D.1 (The standard HB updates). *The updates $\mathbf{x}^{(t)}$ and $\mathbf{r}^{(t)}$ of the HB method for solving Equ. (4) can be written as*

$$\begin{aligned} \mathbf{x}^{(t+1)} &= \mathbf{x}^{(t)} + (1 + \tilde{\alpha}^2)\mathbf{r}^{(t)} + \tilde{\alpha}^2(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}) \\ \mathbf{r}^{(t+1)} &= 2\tilde{\alpha}\mathbf{W}\mathbf{r}^{(t)} - \tilde{\alpha}^2\mathbf{r}^{(t-1)}. \end{aligned}$$

The residual updates can be rewritten as a second-order homogeneous equation

$$\mathbf{y}^{(t+1)} - 2\mathbf{\Lambda}\mathbf{y}^{(t)} + \mathbf{y}^{(t-1)} = \mathbf{0}, \quad \forall t = 1, 2, 3, \dots$$

where $\mathbf{y}^{(t)}$ is such that $\mathbf{r}^{(t)} = \tilde{\alpha}^t\mathbf{V}\mathbf{y}^{(t)}$, $t \geq 0$ with $\mathbf{y}^{(0)} = \mathbf{V}^\top\mathbf{r}^{(0)} = \mathbf{V}^\top\mathbf{b}$.

Proof. We follow the standard Polyak's heavy-ball method [40] as

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \eta_\alpha \nabla f(\mathbf{x}^{(t)}) + \eta_\beta (\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}),$$

where $\nabla f(\mathbf{x}^{(t)}) = \mathbf{Q}\mathbf{x}^{(t)} - \mathbf{b}$ and $\eta_\alpha = 4/(\sqrt{2/(1+\alpha)} + \sqrt{2\alpha/(1+\alpha)})^2 = 2(1+\alpha)/(1+\sqrt{\alpha})^2 = 1 + \tilde{\alpha}^2$ and $\eta_\beta = (\sqrt{2/(1+\alpha)} - \sqrt{2\alpha/(1+\alpha)})^2/(\sqrt{2/(1+\alpha)} + \sqrt{2\alpha/(1+\alpha)})^2 = \tilde{\alpha}^2$. Hence, it leads to the following updates

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + (1 + \tilde{\alpha}^2)\mathbf{r}^{(t)} + \tilde{\alpha}^2(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}).$$

Inserting

$$\mathbf{r}^{(t)} = \mathbf{Q}(\mathbf{x}^* - \mathbf{x}^{(t)}) = \mathbf{b} - \left(\mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{W} \right) \mathbf{x}^{(t)} = \mathbf{b} - \left(\mathbf{I} - \frac{2\tilde{\alpha}}{1+\tilde{\alpha}^2} \mathbf{W} \right) \mathbf{x}^{(t)}$$

then $\mathbf{x}^{(t+1)} = 2\tilde{\alpha} \mathbf{W} \mathbf{x}^{(t)} - \tilde{\alpha}^2 \mathbf{x}^{(t-1)} + (1 + \tilde{\alpha}^2) \mathbf{b}$ and since

$$\mathbf{Q} \mathbf{W} = \left(\mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{W} \right) \mathbf{W} = \mathbf{W} \left(\mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{W} \right) = \mathbf{W} \mathbf{Q}$$

So

$$\begin{aligned} \mathbf{r}^{(t+1)} &= -\mathbf{Q}(\mathbf{x}^{(t+1)} - \mathbf{x}^*) \\ &= -2\tilde{\alpha} \mathbf{Q} \mathbf{W} \mathbf{x}^{(t)} + \tilde{\alpha}^2 \mathbf{Q} \mathbf{x}^{(t-1)} + (\mathbf{I} - (1 + \tilde{\alpha}^2) \mathbf{Q}) \mathbf{b} \\ &= -2\tilde{\alpha} \mathbf{Q} \mathbf{W} \mathbf{x}^{(t)} + \tilde{\alpha}^2 \mathbf{Q} \mathbf{x}^{(t-1)} + (-\tilde{\alpha}^2 + 2\tilde{\alpha} \mathbf{W}) \mathbf{b} \\ &= 2\tilde{\alpha} \mathbf{W} \mathbf{r}^{(t)} - \tilde{\alpha}^2 \mathbf{r}^{(t-1)} \end{aligned}$$

Using $\mathbf{r}^{(t)} = \tilde{\alpha}^t \mathbf{V} \mathbf{y}^{(t)}$, $t \geq 0$

$$\tilde{\alpha}^{t+1} \mathbf{V} \mathbf{y}^{(t+1)} = 2\tilde{\alpha} \mathbf{W} \tilde{\alpha}^t \mathbf{V} \mathbf{y}^{(t)} - \tilde{\alpha}^2 \tilde{\alpha}^{t-1} \mathbf{V} \mathbf{y}^{(t-1)} \Rightarrow \mathbf{V} \mathbf{y}^{(t+1)} = 2\mathbf{W} \mathbf{V} \mathbf{y}^{(t)} - \mathbf{V} \mathbf{y}^{(t-1)}.$$

As $\mathbf{W} = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^\top$ and $\mathbf{V}^\top = \mathbf{V}^{-1}$ is orthogonal matrix, we continue to have

$$\mathbf{V}^\top \mathbf{V} \mathbf{y}^{(t+1)} - 2\mathbf{V}^\top \mathbf{V} \mathbf{\Lambda} \mathbf{V}^\top \mathbf{V} \mathbf{y}^{(t)} + \mathbf{V}^\top \mathbf{V} \mathbf{y}^{(t-1)} = \mathbf{0} \Rightarrow \mathbf{y}^{(t+1)} - 2\mathbf{\Lambda} \mathbf{y}^{(t)} + \mathbf{y}^{(t-1)} = \mathbf{0}.$$

□

Theorem D.2 (Convergence analysis of Heavy-Ball (HB)). *To solve the minimization problem in Equ. (4), we propose the following standard HB updates as*

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + (1 + \tilde{\alpha}^2) \mathbf{r}^{(t)} + \tilde{\alpha}^2 (\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}), \quad \mathbf{r}^{(t+1)} = 2\tilde{\alpha} \mathbf{W} \mathbf{r}^{(t)} - \tilde{\alpha}^2 \mathbf{r}^{(t-1)},$$

where the initial condition is $\mathbf{x}^{(0)} = \mathbf{0}$, $\mathbf{r}^{(0)} = \mathbf{b}$, $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \mathbf{\Gamma} \mathbf{r}^{(0)}$, $\mathbf{r}^{(1)} = \mathbf{b} - \mathbf{Q} \mathbf{x}^{(1)}$. Then there exists a constant τ such that the total iteration complexity to reach the stop condition $\{u : |r_u| \leq \epsilon d_u, u \in \mathcal{V}\} = \emptyset$ is

$$t = \left\lceil \frac{1 + \sqrt{\alpha}}{2\sqrt{\alpha}} \ln \frac{C_t \|\mathbf{r}^{(0)}\|_2}{\epsilon} \right\rceil,$$

where $C_t = 1$ if $\mathbf{\Gamma} = \mathbf{Q}^{-1}(\mathbf{I} - \tilde{\alpha} \mathbf{W})$ (ideal case); $C_t = \max \left\{ \frac{1+\tilde{\alpha}^{-1}}{\sqrt{1-\lambda_2^2}}, 1 + (1 + \tilde{\alpha}^{-1})t \right\}$ if $\mathbf{\Gamma} = \mathbf{0}$ (practical case); and $C_t = \frac{2}{\sqrt{1-\lambda_2^2}}$ if $\mathbf{\Gamma} = \frac{(1-\tilde{\alpha})(1+\alpha)}{2} \mathbf{I}$ (\mathcal{G} is not bi-partite graph).

Proof. Recall $\mathbf{W} = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^\top$ and then $\mathbf{V}^\top \mathbf{r}^{(t+1)} = 2\tilde{\alpha} \mathbf{\Lambda} \mathbf{V}^\top \mathbf{r}^{(t)} - \tilde{\alpha}^2 \mathbf{V}^\top \mathbf{r}^{(t-1)}$. By Lemma D.1, we have

$$\mathbf{y}^{(t+1)} - 2\mathbf{\Lambda} \mathbf{y}^{(t)} + \mathbf{y}^{(t-1)} = \mathbf{0},$$

where we obtained n second-order difference equations

$$y_i^{(t+1)} - 2\lambda_i y_i^{(t)} + y_i^{(t-1)} = 0, \quad \forall i = 1, 2, \dots, n.$$

Follow the Lemma C.3, Equ. (23) has the solution

$$y_i^{(t)} = \begin{cases} \frac{\sin(\theta_i t) y_i^{(1)} - \sin(\theta_i (t-1)) y_i^{(0)}}{\sin(\theta_i)} & |\lambda_i| < 1 \text{ where } \theta_i = \arccos(\lambda_i) \\ (y_i^{(0)} + (y_i^{(1)} - \lambda_i y_i^{(0)}) t) \lambda_i^t & |\lambda_i| = 1, \end{cases} \quad (34)$$

where in the case of $|\lambda_i| < 1$. We consider the three cases of $\mathbf{\Gamma}$

- **Ideal Case:** We can eliminate t in (34), when $\mathbf{y}^{(1)} = \mathbf{\Lambda}\mathbf{y}^{(0)}$, we get $y_i^{(1)} = \lambda_i y_i^{(0)}$, and then $y_i^{(t)}$ can be simplified into

$$y_i^{(t)} = \begin{cases} \frac{(\lambda_i \sin(\theta_i t) - \sin(\theta_i(t-1)))y_i^{(0)}}{\sin(\theta_i)} & |\lambda_i| < 1 \\ y_i^{(0)} \lambda_i^t & |\lambda_i| = 1 \end{cases} = \begin{cases} y_i^{(0)} \cos(\theta_i t) & |\lambda_i| < 1 \\ y_i^{(0)} \lambda_i^t & |\lambda_i| = 1 \end{cases} \leq \begin{cases} |y_i^{(0)}| & |\lambda_i| < 1 \\ |y_i^{(0)}| & |\lambda_i| = 1 \end{cases}.$$

In this case, $\mathbf{\Gamma}$ needs to be $\mathbf{\Gamma} = \mathbf{Q}^{-1}(\mathbf{I} - \tilde{\alpha}\mathbf{W})$. Therefore, we have

$$\|\mathbf{V}^\top \mathbf{r}^{(t)}\|_2 = \|\mathbf{r}^{(t)}\|_2 = \tilde{\alpha}^t \|\mathbf{y}^{(t)}\|_2 \leq \tilde{\alpha}^t \|\mathbf{y}^{(0)}\|_2 = \tilde{\alpha}^t \|\mathbf{V}^\top \mathbf{r}^{(0)}\|_2 = \tilde{\alpha}^t \|\mathbf{r}^{(0)}\|_2.$$

- **Practical Case:** Just letting $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} = \mathbf{0}$, we have $\tilde{\alpha}\mathbf{y}^{(1)} = \mathbf{y}^{(0)}$, then

$$y_i^{(t)} = \begin{cases} \frac{\tilde{\alpha}^{-1} \sin(\theta_i t) - \sin(\theta_i(t-1))}{\sin(\theta_i)} y_i^{(0)} & |\lambda_i| < 1 \\ (1 + (\tilde{\alpha}^{-1} - \lambda_i)t) y_i^{(0)} \lambda_i^t & |\lambda_i| = 1 \end{cases} \leq \max \left\{ \frac{1 + \tilde{\alpha}^{-1}}{\sqrt{1 - \lambda_2^2}}, 1 + (1 + \tilde{\alpha}^{-1})t \right\} |y_i^{(0)}|,$$

where $\theta_i = \arccos(\lambda_i)$.

- **Non-bipartite graph Case:** When the graph is non-bipartite, we can eliminate t , as the following: We choose $\mathbf{\Gamma} = \tau\mathbf{I}$, we have

$$\begin{aligned} \mathbf{x}^{(1)} &= \mathbf{x}^{(0)} + \mathbf{\Gamma}\mathbf{r}^{(0)} = \mathbf{\Gamma}\mathbf{r}^{(0)}, & \mathbf{r}^{(1)} &= \mathbf{b} - \mathbf{Q}\mathbf{x}^{(1)} = \mathbf{r}^{(0)} - \tau\mathbf{Q}\mathbf{r}^{(0)} \\ \mathbf{V}^\top \mathbf{r}^{(1)} &= (1 - \tau)\mathbf{V}^\top \mathbf{r}^{(0)} + \frac{(1 - \alpha)\tau}{1 + \alpha} \mathbf{\Lambda}\mathbf{V}^\top \mathbf{r}^{(0)}, & \mathbf{v}_i^\top \mathbf{r}^{(1)} &= (1 - \tau + \frac{(1 - \alpha)\tau}{1 + \alpha} \lambda_i) \mathbf{v}_i^\top \mathbf{r}^{(0)} \end{aligned}$$

We have the following relations

$$\begin{aligned} \mathbf{v}_i^\top \mathbf{r}^{(0)} &= y_i^{(0)} \\ \mathbf{v}_i^\top \mathbf{r}^{(1)} &= \tilde{\alpha} y_i^{(1)} = (1 - \tau + \frac{(1 - \alpha)\tau}{1 + \alpha} \lambda_i) \mathbf{v}_i^\top \mathbf{r}^{(0)} = (1 - \tau + \frac{(1 - \alpha)\tau \lambda_i}{1 + \alpha}) y_i^{(0)}, \end{aligned}$$

To make t disappear when $\lambda_i = 1$, we need $y_i^{(0)} = y_i^{(1)}$, or

$$1 - \tau + \frac{(1 - \alpha)\tau}{1 + \alpha} = \frac{1 - \sqrt{\alpha}}{1 + \sqrt{\alpha}} \iff \tau = \frac{1 + \alpha}{\alpha + \sqrt{\alpha}}.$$

In this case, we have

$$|y_i^{(t)}| \leq \frac{2}{\sqrt{1 - \lambda_2^2}} |y_i^{(0)}|$$

To make sure the algorithm stops when the stop condition is met, it is enough for

$$\|\mathbf{r}^{(t)}\|_2 = \tilde{\alpha}^t \|\mathbf{y}^{(t)}\|_2 \leq \tilde{\alpha}^t C_t \|\mathbf{y}^{(0)}\|_2 = \tilde{\alpha}^t C_t \|\mathbf{V}^\top \mathbf{r}^{(0)}\|_2 = \tilde{\alpha}^t C_t \|\mathbf{r}^{(0)}\|_2 \leq \epsilon.$$

This means $C_t \|\mathbf{r}^{(0)}\|_2 \tilde{\alpha}^t \leq \epsilon$, which leads to the following

$$t = \left\lceil \frac{1 + \sqrt{\alpha}}{2\sqrt{\alpha}} \ln \frac{C_t \|\mathbf{r}^{(0)}\|_2}{\epsilon} \right\rceil.$$

□

Remark D.3. The constant that appears in the bound involves the second largest eigenvalue λ_2 of \mathbf{AD}^{-1} . It is deeply related to the mixing time of random walk [8] where the second largest eigenvalue determines the mixing time of the walk. A smaller absolute value of the second largest eigenvalue indicates that a random walk on the graph will mix (i.e., approach its steady-state distribution) more quickly. Our proof is partially inspired by d'Aspremont et al. [11] where we directly bound $\mathbf{r}^{(t)}$ instead of providing bound for $e^{(t)}$.

D.2 Residual Updates of LOCHB and Proof of Theorem D.6

Lemma D.4. *Let the local heavy-ball method be defined as*

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \mathbf{\Delta}_{\mathcal{S}_t}^{(t)}, \quad \mathbf{r}^{(t+1)} = \mathbf{r}^{(t)} - \mathbf{Q}\mathbf{\Delta}_{\mathcal{S}_t}^{(t)}, \quad \mathbf{\Delta}^{(t)} = (1 + \tilde{\alpha}^2)\mathbf{r}^{(t)} + \tilde{\alpha}^2(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}),$$

where $\mathbf{x}^{(0)} = \mathbf{0}$, $\mathbf{x}^{(1)} = \mathbf{\Gamma}\mathbf{r}^{(0)}$ and $\mathbf{\Gamma} = \text{diag}(\Gamma_1, \Gamma_2, \dots, \Gamma_n)$ is initial step size matrix. We have the following expanding sequence

$$\begin{aligned} \tilde{\alpha}^2(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\bar{\mathcal{S}}_t} &= (1 + \tilde{\alpha}^2) \sum_{i=1}^{t-1} \tilde{\alpha}^{2(t-i)} \mathbf{r}_{\mathcal{S}_{i:t-1} \cap \bar{\mathcal{S}}_t}^{(i)} + \tilde{\alpha}^{2t} \mathbf{\Gamma} \mathbf{r}_{\mathcal{S}_{0:t-1} \cap \bar{\mathcal{S}}_t}^{(0)}, \quad \forall t \geq 1 \\ \mathbf{\Delta}_{\bar{\mathcal{S}}_t}^{(t)} &= (1 + \tilde{\alpha}^2) \sum_{i=1}^t \tilde{\alpha}^{2(t-i)} \mathbf{r}_{\mathcal{S}_{i:t-1} \cap \bar{\mathcal{S}}_t}^{(i)} + \tilde{\alpha}^{2t} \mathbf{\Gamma} \mathbf{r}_{\mathcal{S}_{0:t-1} \cap \bar{\mathcal{S}}_t}^{(0)}, \quad \forall t \geq 1. \end{aligned}$$

Furthermore, we have the following sequence

$$\begin{aligned} \tilde{\alpha}^2(\mathbf{V}^\top - \frac{1-\alpha}{1+\alpha} \mathbf{\Lambda} \mathbf{V}^\top)(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\bar{\mathcal{S}}_t} &= \\ (1 + \tilde{\alpha}^2) \sum_{i=1}^{t-1} \tilde{\alpha}^{2(t-i)} (\mathbf{V}^\top - \frac{1-\alpha}{1+\alpha} \mathbf{\Lambda} \mathbf{V}^\top) \mathbf{r}_{\bar{\mathcal{S}}_{i,t}}^{(i)} + \mathbf{\Gamma} \tilde{\alpha}^{2t} (\mathbf{V}^\top - \frac{1-\alpha}{1+\alpha} \mathbf{\Lambda} \mathbf{V}^\top) \mathbf{r}_{\bar{\mathcal{S}}_{0,t}}^{(0)}. \end{aligned}$$

where we denote $\bar{\mathcal{S}}_{i,t} \triangleq \mathcal{S}_{i:t-1} \cap \bar{\mathcal{S}}_t$.

Proof. We assume all nonzeros in \mathbf{b} are active nodes at time $t = 0$ and $t = 1$, i.e., $\mathcal{S}_0 = \mathbf{r}^{(0)} = \text{supp}(\mathbf{b})$. The local updates can be expressed as

$$\begin{aligned} \mathbf{x}^{(t+1)} &= \mathbf{x}^{(t)} + (1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_t}^{(t)} + \tilde{\alpha}^2(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\mathcal{S}_t} \\ \mathbf{r}^{(t+1)} &= \mathbf{b} - \mathbf{Q}\mathbf{x}^{(t+1)} \\ &= \mathbf{r}^{(t)} - \underbrace{(1 + \tilde{\alpha}^2) \mathbf{Q} \mathbf{r}^{(t)} - \tilde{\alpha}^2 \mathbf{Q}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})}_{\text{original updates}} \\ &\quad + \underbrace{(1 + \tilde{\alpha}^2) \mathbf{Q} \mathbf{r}_{\bar{\mathcal{S}}_t}^{(t)} + \tilde{\alpha}^2 \mathbf{Q}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\bar{\mathcal{S}}_t}}_{\text{noisy with small magnitudes}} \\ &= 2\tilde{\alpha} \mathbf{W} \mathbf{r}^{(t)} - \tilde{\alpha}^2 \mathbf{r}^{(t-1)} + \underbrace{(1 + \tilde{\alpha}^2) \mathbf{Q} \mathbf{r}_{\bar{\mathcal{S}}_t}^{(t)} + \tilde{\alpha}^2 \mathbf{Q}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\bar{\mathcal{S}}_t}}_{\text{noisy with small magnitudes}} \\ &= 2\tilde{\alpha} \mathbf{W} \mathbf{r}^{(t)} - \tilde{\alpha}^2 \mathbf{r}^{(t-1)} + (1 + \tilde{\alpha}^2) (\mathbf{I} - \frac{1-\alpha}{1+\alpha} \mathbf{W}) \mathbf{r}_{\bar{\mathcal{S}}_t}^{(t)} + \tilde{\alpha}^2 \mathbf{Q}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\bar{\mathcal{S}}_t} \\ \mathbf{r}^{(t+1)} - 2\tilde{\alpha} \mathbf{W} \mathbf{r}^{(t)} + \tilde{\alpha}^2 \mathbf{r}^{(t-1)} &= (1 + \tilde{\alpha}^2) \mathbf{r}_{\bar{\mathcal{S}}_t}^{(t)} - 2\tilde{\alpha} \mathbf{W} \mathbf{r}_{\bar{\mathcal{S}}_t}^{(t)} + \tilde{\alpha}^2 \mathbf{Q}(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\bar{\mathcal{S}}_t} \end{aligned}$$

For $t \geq 1$, we can expand $\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}$ as the following

$$\begin{aligned}
\mathbf{x}^{(t+1)} &= \mathbf{x}^{(t)} + (1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_t}^{(t)} + \tilde{\alpha}^2 (\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\mathcal{S}_t} \\
\Delta^{(t)} = \mathbf{x}^{(t+1)} - \mathbf{x}^{(t)} &= (1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_t}^{(t)} + \tilde{\alpha}^2 ((1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_{t-1}^{(t-1)}}^{(t-1)} + \tilde{\alpha}^2 (\mathbf{x}^{(t-1)} - \mathbf{x}^{(t-2)})_{\mathcal{S}_{t-1}^{(t-1)}})_{\mathcal{S}_t} \\
&= (1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_t}^{(t)} + \tilde{\alpha}^2 (1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_{t-1:t}^{(t-1)}}^{(t-1)} + \tilde{\alpha}^4 (\mathbf{x}^{(t-1)} - \mathbf{x}^{(t-2)})_{\mathcal{S}_{t-1:t}} \\
&= (1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_t}^{(t)} + \tilde{\alpha}^2 (1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_{t-1:t}^{(t-1)}}^{(t-1)} + \tilde{\alpha}^4 ((1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_{t-2}^{(t-2)}}^{(t-2)} \\
&\quad + \tilde{\alpha}^2 (\mathbf{x}^{(t-2)} - \mathbf{x}^{(t-3)})_{\mathcal{S}_{t-2}^{(t-2)}})_{\mathcal{S}_{t-1:t}} \\
&= (1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_t}^{(t)} + \tilde{\alpha}^2 (1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_{t-1:t}^{(t-1)}}^{(t-1)} + \tilde{\alpha}^4 (1 + \tilde{\alpha}^2) \mathbf{r}_{\mathcal{S}_{t-2:t}^{(t-2)}}^{(t-2)} \\
&\quad + \tilde{\alpha}^6 (\mathbf{x}^{(t-2)} - \mathbf{x}^{(t-3)})_{\mathcal{S}_{t-2:t}} \\
&= (1 + \tilde{\alpha}^2) \sum_{i=t-2}^t \tilde{\alpha}^{2(t-i)} \mathbf{r}_{\mathcal{S}_{i:t}}^{(i)} + \tilde{\alpha}^6 (\mathbf{x}^{(t-2)} - \mathbf{x}^{(t-3)})_{\mathcal{S}_{t-2:t}} \\
&= (1 + \tilde{\alpha}^2) \sum_{i=1}^t \tilde{\alpha}^{2(t-i)} \mathbf{r}_{\mathcal{S}_{i:t}}^{(i)} + \tilde{\alpha}^{2t} (\mathbf{x}^{(1)} - \mathbf{x}^{(0)})_{\mathcal{S}_{1:t}} \\
&= (1 + \tilde{\alpha}^2) \sum_{i=1}^t \tilde{\alpha}^{2(t-i)} \mathbf{r}_{\mathcal{S}_{i:t}}^{(i)} + \tilde{\alpha}^{2t} \mathbf{\Gamma} \mathbf{r}_{\mathcal{S}_{1:t}}^{(0)}.
\end{aligned}$$

Note $\mathcal{S}_0 = \text{supp}(\mathbf{r}^{(0)})$, then $\mathbf{r}_{\mathcal{S}_{1:t}}^{(0)} = \mathbf{r}_{\mathcal{S}_{0:t}}^{(0)}$, we continue to have

$$\begin{aligned}
\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)} &= (1 + \tilde{\alpha}^2) \sum_{i=1}^{t-1} \tilde{\alpha}^{2(t-i-1)} \mathbf{r}_{\mathcal{S}_{i:t-1}}^{(i)} + \mathbf{\Gamma} \tilde{\alpha}^{2(t-1)} \mathbf{r}_{\mathcal{S}_{0:t-1}}^{(0)}, \quad \forall t \geq 1 \\
\tilde{\alpha}^2 (\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{\bar{\mathcal{S}}_t} &= (1 + \tilde{\alpha}^2) \sum_{i=1}^{t-1} \tilde{\alpha}^{2(t-i)} \mathbf{r}_{\mathcal{S}_{i:t-1} \cap \bar{\mathcal{S}}_t}^{(i)} + \mathbf{\Gamma} \tilde{\alpha}^{2t} \mathbf{r}_{\mathcal{S}_{0:t-1} \cap \bar{\mathcal{S}}_t}^{(0)}, \quad \forall t \geq 1
\end{aligned}$$

The rest follows readily. □

Lemma D.5 (The nonhomogeneous difference equation). *Given $\mathbf{y}^{(1)} = \mathbf{\Lambda} \mathbf{y}^{(0)}$, equations*

$$\mathbf{y}^{(t+1)} - 2\mathbf{\Lambda} \mathbf{y}^{(t)} + \mathbf{y}^{(t-1)} := \mathbf{f}^{(t)}.$$

have the following solutions

$$\mathbf{y}^{(t)} = \mathbf{Z}_t \mathbf{y}^{(0)} + \sum_{k=1}^{t-1} \mathbf{H}_{k,t} \mathbf{f}^{(k)},$$

where

$$\mathbf{Z}_t = \begin{cases} \mathbf{diag}(1, \dots, \cos(\theta_i t), \dots, (-1)^t) & \text{for bipartite graphs} \\ \mathbf{diag}(1, \dots, \cos(\theta_i t), \dots, \cos(\theta_n t)) & \text{for non-bipartite graphs,} \end{cases}$$

$$\mathbf{H}_{k,t} = \begin{cases} \mathbf{diag}\left(t-k, \dots, \frac{\sin(\theta_i(t-k))}{\sin \theta_i}, \dots, (-1)^{t-k-1} (t-k)\right) & \text{for bipartite graphs} \\ \mathbf{diag}\left(t-k, \dots, \frac{\sin(\theta_i(t-k))}{\sin \theta_i}, \dots, \frac{\sin(\theta_n(t-k))}{\sin \theta_n}\right) & \text{for non-bipartite graphs.} \end{cases}$$

Proof. This directly follows from Lemma C.4. □

Theorem D.6 (Representation of $\mathbf{r}^{(t)}$ for LOCHB). *Given $t \geq 1$, $\mathbf{x}^{(0)} = \mathbf{0}$ and $\mathbf{x}^{(1)} = \mathbf{\Gamma} \mathbf{r}_{S_0}^{(0)}$. The residual of $\mathbf{r}^{(t)}$ of LOCHB satisfies*

$$\begin{aligned} \mathbf{V}^\top \mathbf{r}^{(t)} &= \tilde{\alpha}^t \mathbf{Z}_t \mathbf{V}^\top \mathbf{r}^{(0)} + \underbrace{\tilde{\alpha}^t t \sum_{k=1}^{t-1} \tilde{\alpha}^{k-1} \mathbf{H}_{k,t} \mathbf{V}^\top \mathbf{Q} \mathbf{\Gamma} \mathbf{r}_{S_{0,k}}^{(0)}}_{\mathbf{u}_{0,t}} / t \\ &\quad + 2 \underbrace{\sum_{k=1}^{t-1} \tilde{\alpha}^{t-k} (t-k) \sum_{j=k}^{t-1} \tilde{\alpha}^{j-k} \mathbf{H}_{j,t} \left(\frac{1+\alpha}{1-\alpha} - \mathbf{\Lambda} \right) \mathbf{V}^\top \mathbf{r}_{S_{k,j}}^{(k)}}_{\mathbf{u}_{k,t}} / (t-k). \end{aligned}$$

Proof. Follow Lemma D.4, we have

$$\begin{aligned} \mathbf{r}^{(t+1)} - \mathbf{Q} \mathbf{\Delta}_{S_t}^{(t)} &= \mathbf{r}^{(t)} - \mathbf{Q} \mathbf{\Delta}^{(t)} \\ &= \mathbf{r}^{(t)} - (1 + \tilde{\alpha}^2) \mathbf{Q} \mathbf{r}^{(t)} - \tilde{\alpha}^2 \mathbf{Q} (\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}) \\ &= -\tilde{\alpha}^2 \mathbf{r}^{(t)} + 2\tilde{\alpha} \mathbf{W} \mathbf{r}^{(t)} - \tilde{\alpha}^2 \mathbf{Q} (\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}) \\ &= 2\tilde{\alpha} \mathbf{W} \mathbf{r}^{(t)} - \tilde{\alpha}^2 \mathbf{r}^{(t-1)} \end{aligned}$$

So we can write the updates of LOCHB as

$$\begin{aligned} \mathbf{r}^{(t+1)} - 2\tilde{\alpha} \mathbf{W} \mathbf{r}^{(t)} + \tilde{\alpha}^2 \mathbf{r}^{(t-1)} &= \mathbf{Q} \mathbf{\Delta}_{S_t}^{(t)} \\ &= (1 + \tilde{\alpha}^2) \sum_{i=1}^t \tilde{\alpha}^{2(t-i)} \mathbf{Q} \mathbf{r}_{S_{i,t}}^{(i)} + \tilde{\alpha}^{2t} \mathbf{Q} \mathbf{\Gamma} \mathbf{r}_{S_{0,t}}^{(0)}. \end{aligned}$$

Write $\mathbf{V}^\top \mathbf{r}^{(t)} = \tilde{\alpha}^t \mathbf{y}^{(t)}$, and note

$$\mathbf{r}^{(t)} = \tilde{\alpha}^t \mathbf{V} \mathbf{y}^{(t)}, \quad \mathbf{r}_{S_t}^{(t)} = \tilde{\alpha}^t \left(\mathbf{V} \mathbf{y}^{(t)} \right)_{S_t} \Rightarrow \mathbf{V}^\top \mathbf{r}_{S_t}^{(t)} = \tilde{\alpha}^t \mathbf{V}^\top \left(\mathbf{V} \mathbf{y}^{(t)} \right)_{S_t}.$$

Hence,

$$\begin{aligned} &\mathbf{V}^\top \mathbf{r}^{(t+1)} - 2\tilde{\alpha} \mathbf{V}^\top \mathbf{W} \mathbf{r}^{(t)} + \tilde{\alpha}^2 \mathbf{V}^\top \mathbf{r}^{(t-1)} \\ &= (1 + \tilde{\alpha}^2) \sum_{i=1}^t \tilde{\alpha}^{2(t-i)} \mathbf{V}^\top \mathbf{Q} \mathbf{r}_{S_{i,t}}^{(i)} + \tilde{\alpha}^{2t} \mathbf{V}^\top \mathbf{Q} \mathbf{\Gamma} \mathbf{r}_{S_{0,t}}^{(0)} \\ \Leftrightarrow &\tilde{\alpha}^{t+1} \mathbf{y}^{(t+1)} - 2\tilde{\alpha} \tilde{\alpha}^t \mathbf{W} \mathbf{y}^{(t)} + \tilde{\alpha}^2 \tilde{\alpha}^{t-1} \mathbf{y}^{(t-1)} \\ &= (1 + \tilde{\alpha}^2) \sum_{i=1}^t \tilde{\alpha}^{2(t-i)} \tilde{\alpha}^i \mathbf{Q} \mathbf{V}^\top \left(\mathbf{V}^\top \mathbf{y}^{(i)} \right)_{S_{i,t}} + \tilde{\alpha}^{2t} \mathbf{Q} \mathbf{\Gamma} \mathbf{V}^\top \left(\mathbf{V} \mathbf{y}^{(0)} \right)_{S_{0,t}} \\ \Leftrightarrow &\mathbf{y}^{(t+1)} - 2\mathbf{W} \mathbf{y}^{(t)} + \mathbf{y}^{(t-1)} \\ &= \underbrace{\sum_{i=1}^t (1 + \tilde{\alpha}^2) \tilde{\alpha}^{t-1} \tilde{\alpha}^{-i} \mathbf{Q} \mathbf{V}^\top \left(\mathbf{V} \mathbf{y}^{(i)} \right)_{S_{i,t}}}_{\mathbf{f}_i^{(t)}} + \underbrace{\tilde{\alpha}^{t-1} \mathbf{Q} \mathbf{\Gamma} \mathbf{V}^\top \left(\mathbf{V} \mathbf{y}^{(0)} \right)_{S_{0,t}}}_{\mathbf{f}_0^{(t)}}. \end{aligned}$$

Then, from Lemma D.5

$$\mathbf{y}^{(t)} = \mathbf{Z}_t \mathbf{y}^{(0)} + \sum_{k=1}^{t-1} \mathbf{H}_{k,t} \mathbf{f}^{(k)} \Leftrightarrow \mathbf{V}^\top \mathbf{r}^{(t)} = \tilde{\alpha}^t \mathbf{Z}_t \mathbf{V}^\top \mathbf{r}^{(0)} + \tilde{\alpha}^t \sum_{k=1}^{t-1} \mathbf{H}_{k,t} \mathbf{f}^{(k)}$$

so expanding the error term

$$\begin{aligned}
\tilde{\alpha}^t \mathbf{H}_{k,t} \mathbf{f}^{(k)} &= \sum_{i=1}^k \tilde{\alpha}^t \mathbf{H}_{k,t} \mathbf{f}_i^{(k)} + \tilde{\alpha}^t \mathbf{H}_{k,t} \mathbf{f}_0^{(k)} \\
&= (1 + \tilde{\alpha}^2) \tilde{\alpha}^{t+k-1-i} \sum_{i=1}^k \mathbf{H}_{k,t} \mathbf{Q} \mathbf{V}^\top (\mathbf{V} \mathbf{y}^{(i)})_{\mathcal{S}_{i,k}} \\
&\quad + \tilde{\alpha}^{t+k-1} \mathbf{H}_{k,t} \mathbf{Q} \mathbf{V}^\top (\mathbf{V} \mathbf{y}^{(0)})_{\mathcal{S}_{0,k}} \\
&= (1 + \tilde{\alpha}^2) \tilde{\alpha}^{t+k-1-2i} \sum_{i=1}^k \mathbf{H}_{k,t} \mathbf{Q} \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{i,k}}^{(i)} + \tilde{\alpha}^{t+k-1} \mathbf{H}_{k,t} \mathbf{Q} \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{0,k}}^{(0)} \\
&= 2 \sum_{i=1}^k \tilde{\alpha}^{t+k-2i} \mathbf{H}_{k,t} \left(\frac{1+\alpha}{1-\alpha} \right) \mathbf{Q} \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{i,k}}^{(i)} + \tilde{\alpha}^{t+k-1} \mathbf{H}_{k,t} \mathbf{Q} \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{0,k}}^{(0)} \\
\sum_{k=1}^{t-1} \tilde{\alpha}^t \mathbf{H}_{k,t} \mathbf{f}^{(k)} &= \sum_{k=1}^{t-1} \left(2 \sum_{i=1}^k \tilde{\alpha}^{t+k-2i} \mathbf{H}_{k,t} \left(\frac{1+\alpha}{1-\alpha} \right) \mathbf{Q} \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{i,k}}^{(i)} + \tilde{\alpha}^{t+k-1} \mathbf{H}_{k,t} \mathbf{Q} \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{0,k}}^{(0)} \right) \\
&= 2 \sum_{k=1}^{t-1} \sum_{j=k}^{t-1} \tilde{\alpha}^{t+j-2k} \mathbf{H}_{j,t} \left(\frac{1+\alpha}{1-\alpha} \right) \mathbf{Q} \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{k,j}}^{(k)} + \sum_{k=1}^{t-1} \tilde{\alpha}^{t+k-1} \mathbf{H}_{k,t} \mathbf{Q} \mathbf{V}^\top \mathbf{r}_{\mathcal{S}_{0,k}}^{(0)}
\end{aligned}$$

which when simplified, gives the relation proposed. \square

D.3 Convergence of LOCHB of Proof of Theorem D.8

Corollary D.7. *Define*

$$\beta_{k,t} \triangleq \|\mathbf{u}_{k,t}\|_2 / \|\mathbf{r}^{(k)}\|_2, \quad \beta_k \triangleq \max_t \beta_{k,t}.$$

Then the upper bound of $\|\mathbf{r}^{(t)}\|_2$ can be characterized as

$$\|\mathbf{r}^{(t)}\|_2 \leq \tilde{\alpha}^t \prod_{j=0}^{t-1} (1 + \beta_j) y_t, \quad (35)$$

where $y_{t+1} - 2y_t + y_{t-1} / ((1 + \beta_{t-1})(1 + \beta_t)) = 0$ where $y_0 = y_1 = \|\mathbf{r}^{(0)}\|_2$.

Proof. Since $\|\mathbf{u}_{k,t}\|_2 \leq \beta_k \|\mathbf{r}^{(k)}\|_2$, then given the final iterative updates (29)

$$\mathbf{V}^\top \mathbf{r}^{(t)} = \tilde{\alpha}^t \mathbf{Z}_t \mathbf{V}^\top \mathbf{r}^{(0)} + \tilde{\alpha}^t t \mathbf{u}_{0,t} + 2 \sum_{k=1}^{t-1} \tilde{\alpha}^{t-k} (t-k) \mathbf{u}_{k,t}$$

and since $\|\mathbf{Z}_t\|_2 \leq 1$ we can bound

$$\begin{aligned}
\|\mathbf{r}^{(t)}\|_2 &\leq \tilde{\alpha}^t \|\mathbf{r}^{(0)}\|_2 + \tilde{\alpha}^t t \beta_0 \|\mathbf{r}^{(0)}\|_2 + 2 \sum_{k=1}^{t-1} \tilde{\alpha}^{t-k-1} (t-k) \beta_k \|\mathbf{r}^{(k)}\|_2 \\
\iff \|\mathbf{r}^{(t)}\|_2 - 2 \sum_{k=1}^{t-1} \tilde{\alpha}^{t-k-1} (t-k) \beta_k \|\mathbf{r}^{(k)}\|_2 &\leq \tilde{\alpha}^t (1 + t\beta_0) \|\mathbf{r}^{(0)}\|_2, \quad (36)
\end{aligned}$$

where $t = 0, 1, \dots, T$. The rest just follows a similar strategy shown in Corollary C.9. We have the following inequalities

$$\mathbf{I} - \mathbf{V}_L := \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ -v_{21} & 1 & 0 & \cdots & 0 \\ -v_{31} & -v_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -v_{n1} & -v_{n2} & -v_{n3} & \cdots & 1 \end{pmatrix} \begin{pmatrix} \|\mathbf{r}^{(1)}\|_2 \\ \|\mathbf{r}^{(2)}\|_2 \\ \|\mathbf{r}^{(3)}\|_2 \\ \vdots \\ \|\mathbf{r}^{(T)}\|_2 \end{pmatrix} \leq \begin{pmatrix} \tilde{\alpha}^1 (1 + \beta_0) \|\mathbf{r}^{(0)}\|_2 \\ \tilde{\alpha}^2 (1 + 2\beta_0) \|\mathbf{r}^{(0)}\|_2 \\ \tilde{\alpha}^3 (1 + 3\beta_0) \|\mathbf{r}^{(0)}\|_2 \\ \vdots \\ \tilde{\alpha}^T (1 + T\beta_0) \|\mathbf{r}^{(0)}\|_2 \end{pmatrix} := \mathbf{c},$$

where $(\mathbf{V}_L)_{tk} = 2\tilde{\alpha}^{t-k}(t-k)\beta_k$. Denote each upper bound as $\tau_t = \|\mathbf{r}^{(t)}\|_2$, we will have

$$\begin{aligned}
\tau_t &= c_t + \sum_{k=1}^{t-1} v_{t,k} = \tilde{\alpha}^t(1+t\beta_0)\tau_0 + 2\sum_{k=1}^{t-1}(t-k)\tilde{\alpha}^{t-k}\beta_k\tau_k \\
\tau_{t+1} &= c_{t+1} + \sum_{k=1}^t v_{t+1,k} = \tilde{\alpha}^{t+1}(1+(t+1)\beta_0)\tau_0 + 2\sum_{k=1}^t(t+1-k)\tilde{\alpha}^{t-k+1}\beta_k\tau_k \\
&= \tilde{\alpha}^{t+1}(1+(t+1)\beta_0)\tau_0 + 2\sum_{k=1}^{t-1}(t-k)\tilde{\alpha}^{t-k+1}\beta_k\tau_k + 2\sum_{k=1}^t\tilde{\alpha}^{t-k+1}\beta_k\tau_k \\
\tilde{\alpha}\tau_t &= \tilde{\alpha}^{t+1}(1+t\beta_0)\tau_0 + 2\sum_{k=1}^{t-1}(t-k)\tilde{\alpha}^{t-k+1}\beta_k\tau_k \\
\tau_{t+1} - \tilde{\alpha}\tau_t &= \tilde{\alpha}^{t+1}\beta_0\tau_0 + 2\sum_{k=1}^t\tilde{\alpha}^{t-k+1}\beta_k\tau_k \\
\tau_{t-1} &= \tilde{\alpha}^{t-1}(1+(t-1)\beta_0)\tau_0 + 2\sum_{k=1}^{t-2}(t-k-1)\tilde{\alpha}^{t-k-1}\beta_k\tau_k \\
\tilde{\alpha}^2\tau_{t-1} &= \tilde{\alpha}^{t+1}(1+(t-1)\beta_0)\tau_0 + 2\sum_{k=1}^{t-1}(t-k)\tilde{\alpha}^{t-k+1}\beta_k\tau_k - 2\sum_{k=1}^{t-1}\tilde{\alpha}^{t-k+1}\beta_k\tau_k \\
\tilde{\alpha}\tau_t - \tilde{\alpha}^2\tau_{t-1} &= \tilde{\alpha}^{t+1}\beta_0\tau_0 + 2\sum_{k=1}^{t-1}\tilde{\alpha}^{t-k+1}\beta_k\tau_k.
\end{aligned}$$

$$\tau_{t+1} - 2\tilde{\alpha}\tau_t + \tilde{\alpha}^2\tau_{t-1} = 2\tilde{\alpha}\beta_t\tau_t$$

The above analysis finally leads to $\tau_{t+1} - 2(1+\beta_t)\tilde{\alpha}\tau_t + \tilde{\alpha}^2\tau_{t-1} = 0$. So for

$$\tilde{\alpha}^t \prod_{j=0}^{t-1} (1+\beta_j) y_t = \tau_t$$

we have

$$y_{t+1} - 2y_t + \frac{y_t}{(1+\beta_t)(1+\beta_{t-1})} = 0.$$

Following the same strategy in Corollary C.9, we obtain the upper bound. \square

Theorem D.8 (Convergence of LOCHB). *Let the geometric mean of β_t be $\bar{\beta}_t \triangleq \prod_{j=0}^{t-1} (1+\beta_j)^{1/t}$. Then the upper bound of $\|\mathbf{r}^{(t)}\|_2$ can be characterized as*

$$\|\mathbf{r}^{(t)}\|_2 \leq \tilde{\alpha}^t \prod_{j=0}^{t-1} (1+\beta_j) y_t, \quad (37)$$

where $y_{t+1} - 2y_t + y_{t-1}/((1+\beta_{t-1})(1+\beta_t)) = 0$ where $y_0 = y_1 = \|\mathbf{r}^{(0)}\|_2$. Assume that there exists a constant $c \in [0, 2)$ such that $\beta_t \leq 1 + \frac{c\sqrt{\alpha}}{1-\sqrt{\alpha}}$. Then the total number of iterations can be bounded as

$$T \leq \left\lceil \frac{1+\sqrt{\alpha}}{(2-c)\sqrt{\alpha}} \right\rceil \ln \left(\frac{y_t}{\epsilon} \right).$$

Then the total runtime \mathcal{T} is bounded by

$$\mathcal{T} \leq \Theta \left(\frac{(1+\sqrt{\alpha})\text{vol}(\mathcal{S}_T)}{(2-c)\sqrt{\alpha}} \ln \frac{y_t}{\epsilon} \right) = \tilde{\mathcal{O}} \left(\frac{\text{vol}(\mathcal{S}_T)}{(2-c)\sqrt{\alpha}} \right),$$

where $\tilde{\mathcal{O}}$ hides $\ln(y_t/\epsilon)$.

Proof. The first part follows from Corollary of D.7. The rest follows the same strategy of LOCCH as in Theorem 4.2. (See also Theorem 4.2.) \square

Table 2: Examples of sparse linear systems

Original Linear system	Our target $Q\mathbf{x} = \mathbf{b}$ $Q = \Lambda - \sigma D^{-1/2} A D^{-1/2}$	$[\mu, L]$	Ref.
$(I - \sigma W)\mathbf{x} = \alpha \mathbf{e}_s$ ^{1.}	$\Lambda = I, \sigma = 1 - \alpha$ $\mathbf{b} = \alpha \mathbf{e}_s$	$[\alpha, 2 - \alpha]$	[27] [53]
$(\alpha I + \frac{1-\alpha}{2} \mathcal{L})\mathbf{x} = \alpha D^{-1/2} \mathbf{e}_s$ ^{2.} $\mathcal{L} = I - D^{-1/2} A D^{-1/2}$	$\Lambda = I, \sigma = \frac{1-\alpha}{1+\alpha}$ $\mathbf{b} = 2\alpha D^{-1/2} \mathbf{s} / (1 + \alpha)$	$[\frac{2\alpha}{1+\alpha}, \frac{2}{1+\alpha}]$	[2] [13] [12] [37]
$(I - (1 - \alpha) A D^{-1}) \mathbf{y} = \alpha \mathbf{e}_s$ ^{3.}	$\Lambda = I, \sigma = 1 - \alpha$ $\mathbf{b} = \alpha D^{-1/2} \mathbf{e}_s, \alpha \in (0, 1)$ $\mathbf{x} = D^{-1/2} \mathbf{y}$	$[\alpha, 2 - \alpha]$	[7] [56]
$(\frac{\lambda}{n} I_n + D - A) \mathbf{y} = 2\lambda \mathbf{e}_s$ ^{4.}	$\Lambda = \frac{\lambda}{n} D^{-1} + I_n, \sigma = 1$ $\mathbf{b} = 2\lambda D^{-1/2} \mathbf{e}_s, \lambda \in [1, n]$ $\mathbf{x} = D^{1/2} \mathbf{y}$	$[\frac{\lambda}{nd_{\max}}, \frac{\lambda+2n}{n}]$	[41] [57]

D.4 Implementation of LOCHB

We present the implementation of LOCHB as follows: Recall the updates of LOCHB is

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + (1 + \tilde{\alpha}^2) \mathbf{r}_{S_t}^{(t)} + \tilde{\alpha}^2 (\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)})_{S_t}, \quad \mathbf{r}^{(t+1)} = 2\tilde{\alpha} \mathbf{W} \mathbf{r}^{(t)} - \tilde{\alpha}^2 \mathbf{r}^{(t-1)}.$$

The corresponding local updates are

$$\begin{aligned} \Delta^{(t)} &= (1 + \tilde{\alpha}^2) \mathbf{r}^{(t)} + \tilde{\alpha}^2 \tilde{\mathbf{x}}^{(t)} \\ \mathbf{x}^{(t+1)} &= \mathbf{x}^{(t)} + \Delta_{S_t}^{(t)} \\ \mathbf{r}^{(t+1)} &= \mathbf{r}^{(t)} - \Delta^{(t)} + \frac{1 - \alpha}{1 + \alpha} \mathbf{W} \Delta^{(t)} \\ \tilde{\mathbf{x}}^{(t+1)} &= \tilde{\mathbf{x}}^{(t)} + \Delta^{(t)} - \Delta^{(t-1)}, \end{aligned}$$

where if we choose $\mathbf{x}^{(0)} = \mathbf{x}^{(1)} = \tilde{\mathbf{x}}^{(1)} = \mathbf{0}, \mathbf{r}^{(0)} = \mathbf{r}^{(1)} = \mathbf{b}$ and $\Delta^{(0)} = \mathbf{0}$.

E Instances of Sparse Linear Systems

E.1 Table of Popular Graph-induced Linear Systems

This section presents most commonly used graph-induced linear system as the following

$$\underbrace{\Lambda - \sigma D^{-1/2} A D^{-1/2}}_Q \mathbf{x} = \mathbf{b},$$

where Q is the generalized version of the perturbed normalized graph Laplacian matrix with perturbation parameter $\sigma > 0$, and \mathbf{b} is a sparse vector. A typical example of $Q = I - \frac{1-\alpha}{1+\alpha} D^{-1/2} A D^{-1/2}$ with $\Lambda = I$ and $\mathbf{b} = 2\alpha D^{-1/2} \mathbf{e}_s / (1 + \alpha)$

The detailed parameters are:

- 1. $W = \tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2}$ and $\tilde{A} = I + A$ is the adjacency matrix defined on $\mathcal{G}(\mathcal{V}, \mathcal{E})$ by adding self-loops for all nodes, and $\tilde{D} = I + D$ is defined as the augmented degree matrix by adding self-loops. $\alpha \in (0, 1)$ and usually $\alpha < 0.5$, The $I - (1 - \alpha)W$ is the perturbed augmented normalized Laplacian with perturbed parameter α .
- 2. $Q = D^{-1/2} (D - \frac{1-\alpha}{2} (D + A)) D^{-1/2} = \alpha I + \frac{1-\alpha}{2} \mathcal{L} > 0$ and $Q = \frac{1-\alpha}{1+\alpha} D^{-1/2} A D^{-1/2}$. This is known as the lazy random-walk version of PPR vectors.
- 3. $\mathbf{x} = \alpha (I - (1 - \alpha) A D^{-1})^{-1} \mathbf{e}_s$ This is the standard Personalized PageRank vectors widely used for graph embeddings and graph neural network designing [7]. It is also used for decoupling for large-scale GNNs [56].

Dataset ID	Dataset Name	n	m
\mathcal{G}_1	as-skitter	1,694,616	11,094,209
\mathcal{G}_2	cit-patent	3,764,117	16,511,740
\mathcal{G}_3	com-dblp	317,080	1,049,866
\mathcal{G}_4	com-lj	3,997,962	34,681,189
\mathcal{G}_5	com-orkut	3,072,441	117,185,083
\mathcal{G}_6	com-youtube	1,134,890	2,987,624
\mathcal{G}_7	ogbn-arxiv	169,343	1,157,799
\mathcal{G}_8	ogbn-mag	1,939,743	21,091,072
\mathcal{G}_9	ogbn-products	2,385,902	61,806,303
\mathcal{G}_{10}	ogbn-proteins	132,534	39,561,252
\mathcal{G}_{11}	soc-lj1	4,843,953	42,845,684
\mathcal{G}_{12}	soc-pokec	1,632,803	22,301,964
\mathcal{G}_{13}	wiki-talk	2,388,953	4,656,682
\mathcal{G}_{14}	ogbl-ppa	576,039	21,231,776
\mathcal{G}_{15}	wiki-en21	6,216,199	160,823,797
\mathcal{G}_{16}	com-friendster	65,608,366	1,806,067,135
\mathcal{G}_{17}	ogbn-papers100m	111,059,433	1,614,061,934

Table 3: Dataset Statistics

- 4. Graph kernel computation for online learning. Each computed vector serves as semi-supervised learning feature vectors [24] or as online node labeling learning vectors [41, 57]. Note the target linear system when $\sigma = 1$

$$D^{-1/2} \left(\frac{\lambda}{n} I_n + D - A \right) D^{-1/2} D^{1/2} \mathbf{y} = 2\lambda D^{-1/2} \mathbf{e}_s$$

$$\left(\frac{\lambda}{n} D^{-1} + I_n - D^{-1/2} A D^{-1/2} \right) D^{1/2} \mathbf{y} = 2\lambda D^{-1/2} \mathbf{e}_s.$$

Hence, we have $\Lambda = \frac{\lambda}{n} D^{-1} + I_n, \sigma = 1, \mathbf{b} = 2\lambda D^{-1/2} \mathbf{e}_s$.

F Experimental Details and Missing Results

F.1 Datasets and Preprocessing

Following Leskovec et al. [34], we treat all 17 graphs as undirected with unit weights. We remove self-loops and keep the largest connected component when the graph is disconnected. After preprocessing, the graphs range from 169,343 nodes in ogbn-proteins to 111,059,433 in ogbn-papers100M, as presented in Table 3.

F.2 Problems Settings and Baseline Methods

For solving Equation (3), we randomly select 50 source nodes s from each graph. The damping factor is fixed at 0.1, i.e., $\alpha = 0.1$ for all experiments, and it varies within the range $\{0.005, 0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$ for others. The ϵ is chosen from the range $[2\alpha/((1+\alpha)d_s), 10^{-4}/n]$.

For solving the local clustering problem, we follow the greedy strategy from Andersen et al. [2], where a local cluster is identified by examining the top magnitudes in PPR vectors. Specifically, we denote the boundary of \mathcal{S} as $\partial(\mathcal{S}) = \{(u, v) \in \mathcal{E} : u \in \mathcal{S}, v \notin \mathcal{S}\}$. The conductance of \mathcal{S} is defined as

$$\Phi(\mathcal{S}) \triangleq \frac{|\partial(\mathcal{S})|}{\min(\text{vol}(\mathcal{S}), 2m - \text{vol}(\mathcal{V} \setminus \mathcal{S}))}.$$

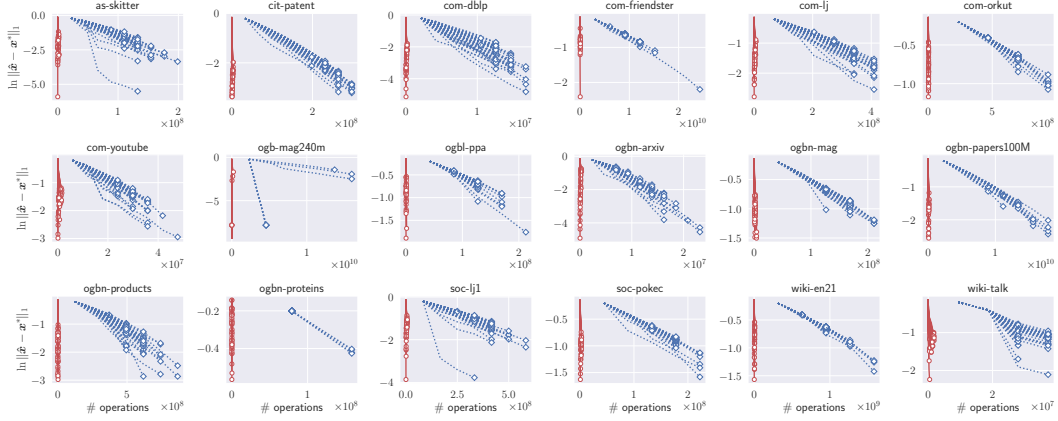


Figure 7: The LOCSOR method compared with CGM over 18 graphs.

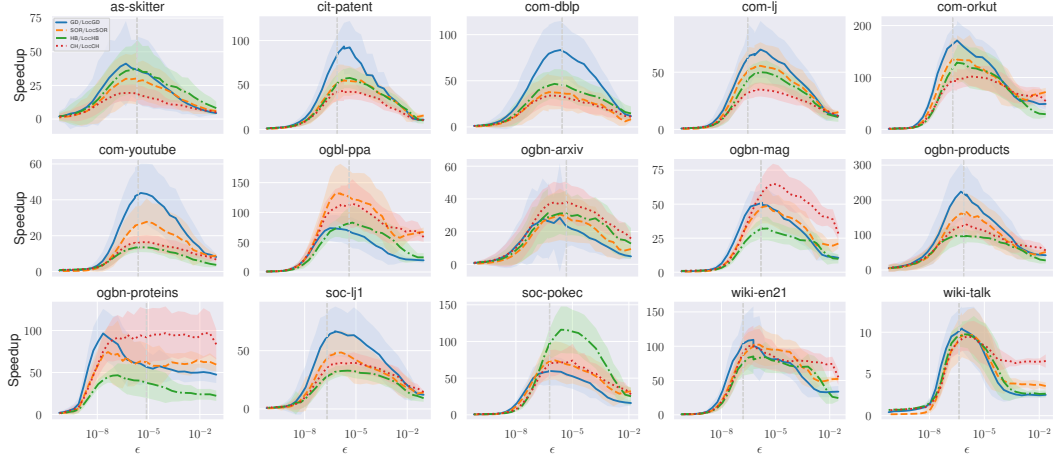


Figure 8: The speedup of local solvers compared with their standard counterparts.

The goal of local clustering is to obtain PPR vectors using these local methods and then apply clustering algorithms to find clusters with low conductance. For the sorting process, given the approximate PPR vector $\tilde{\pi}$, we sort $D^{-1/2}\tilde{\pi}$ in decreasing order of magnitudes. Let the ordered nodes be v_1, v_2, \dots, v_t ; the local clustering algorithm iteratively checks the conductance reduction by v_1, v_2, \dots, v_k where $k = 1, 2, \dots, t$, and after completing all checks, it returns a subset $v_1, v_2, \dots, v_{k'}$ that has the minimal conductance among all examined subsets.

Parameter settings of baselines. For the local ISTA method [13], the precision parameter is set to $\hat{\epsilon} = 0.5$ for all experiments. According to the algorithm’s description of ISTA, the corresponding ρ value is given by $\epsilon/(1 + \hat{\epsilon})$. For LOCSOR, the parameter ω is calculated as $2(1 + \alpha)/(1 + \sqrt{\alpha})^2$. For the local FISTA, as demonstrated in [22], we adopt the same settings as for ISTA and follow its implementation guidelines. We also include preliminary results on ASPR [37]. The algorithm incorporates a parameter, $\hat{\epsilon}$, to control the number of iterations in the nested Accelerated Projected Gradient Descent (APGD). We adjust $\hat{\epsilon}$ from low precision, $\hat{\epsilon} = 0.1/n$, to high precision, $\hat{\epsilon} = 10^{-4}/n$, to ensure the identification of a good approximation.

For our experiment, we used a server powered by an Intel(R) Xeon(R) Gold 5218R CPU, which features 40 cores (80 threads). The system is equipped with 256 GB of RAM.

E.3 Full results of Fig. 15 4 5 6

In all 15 graphs, we set $\alpha = 0.1$ and $\epsilon = 0.1/n$. For each of the testing graphs, we randomly select 50 nodes and run LOCSOR and CGM.

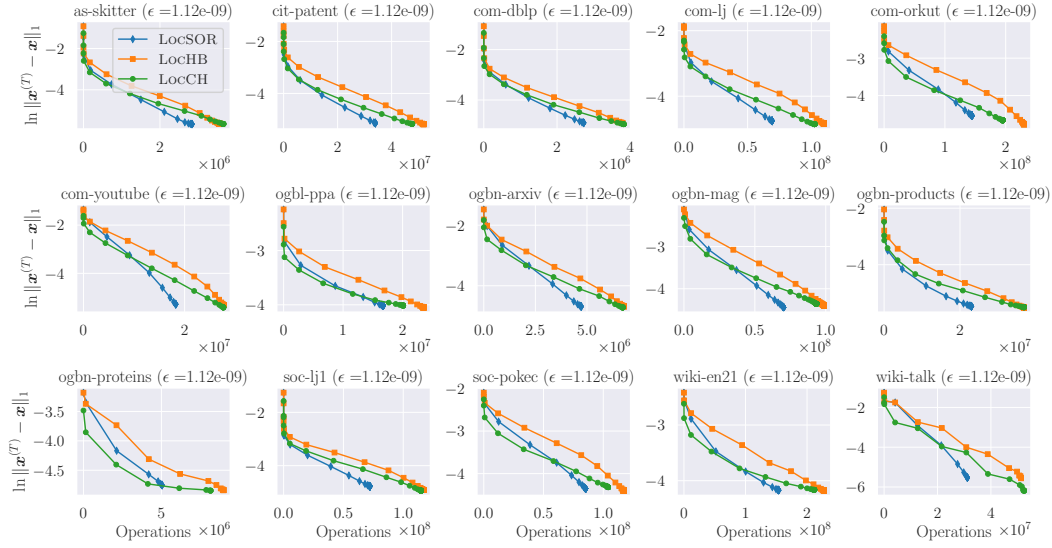


Figure 10: Comparison of three local solvers over 15 graphs.

Fig. 8 presents all speedup tests on 15 datasets. It is evident that these standard linear solvers can be localized effectively.

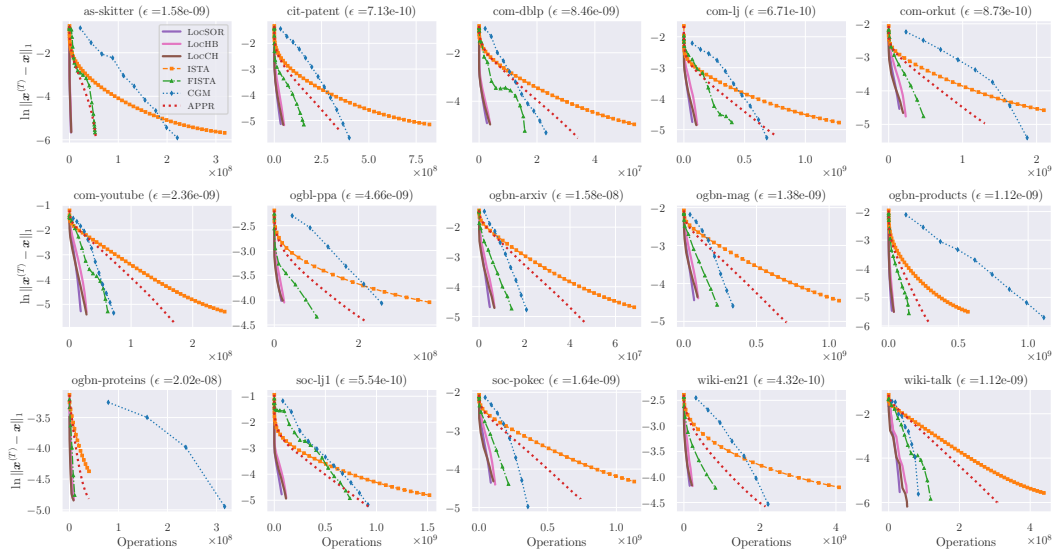


Figure 9: The estimation error reduction tests on 7 solvers including our LOCSOR, LOCHB, and LOCCH. The experiments were conducted on 15 datasets.

Fig. 9 presents the missing results on the estimation error reduction for 15 datasets. Compared with the global solver CGM, all local methods show significant speedup in the early stages. To compare our three local solvers, we zoom in on our results and present them in Fig. 10. Empirically, LOCSOR is the fastest algorithm when the parameter ω is chosen optimally.

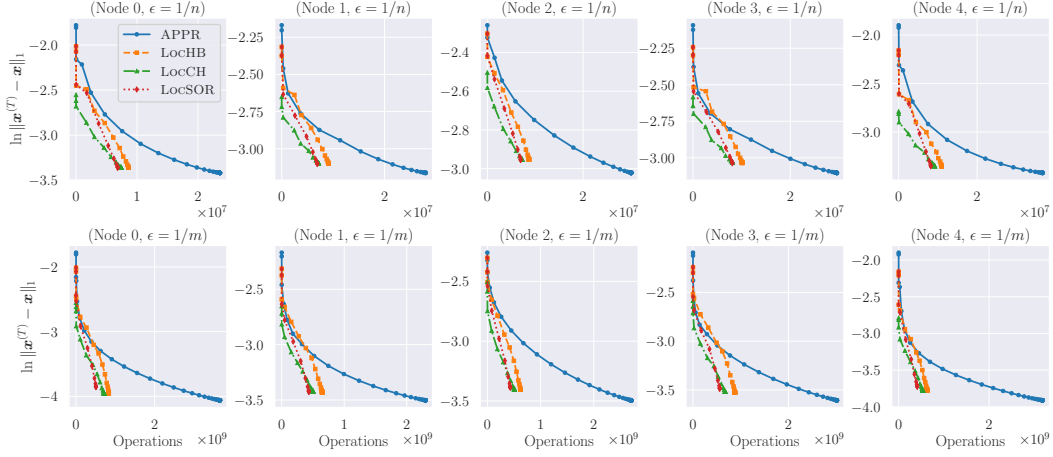


Figure 11: Estimation error as a function of the number of operations on com-friendster. We randomly select 5 different nodes and use $\epsilon = 1/n$ and $\epsilon = 1/m$.

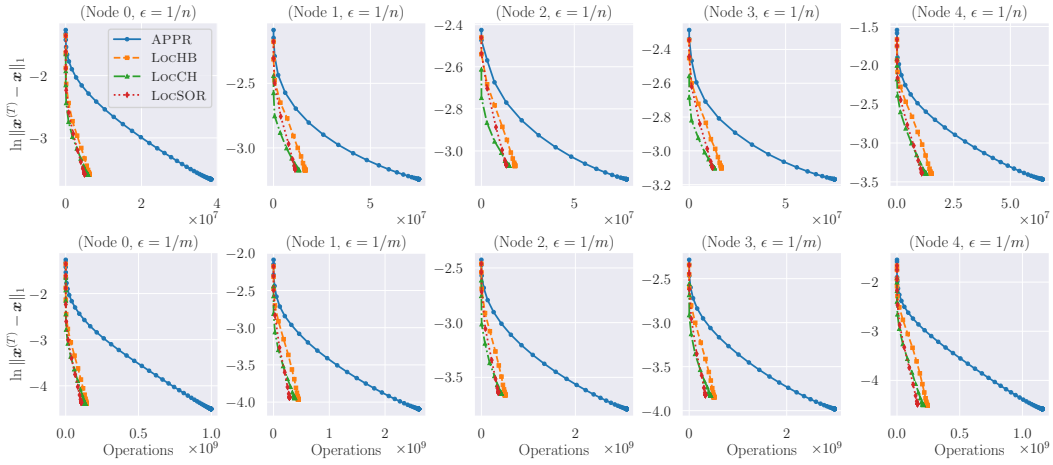


Figure 12: Estimation error as a function of the number of operations on ogbn-papers100m. We randomly select 5 different nodes and use $\epsilon = 1/n$ and $\epsilon = 1/m$.

Dataset	Run time (seconds)			Number of operations		
	LocSOR	LocCH	CGM	LocSOR	LocCH	CGM
as-skitter	0.350 ± 0.054	0.567 ± 0.095	2.626 ± 0.551	7.827e+05	1.026e+06	1.524e+08
cit-patent	0.966 ± 0.120	1.609 ± 0.189	14.660 ± 2.249	1.804e+06	2.298e+06	2.873e+08
com-dblp	0.068 ± 0.058	0.104 ± 0.059	0.469 ± 0.112	8.222e+04	1.166e+05	1.562e+07
com-friendster	15.29 ± 1.89	26.54 ± 3.52	508.50 ± 99.12	7.027e+07	8.063e+07	1.442e+10
com-lj	0.802 ± 0.148	1.410 ± 0.234	7.593 ± 2.361	2.604e+06	3.271e+06	4.122e+08
com-orkut	0.455 ± 0.158	0.815 ± 0.308	13.343 ± 7.951	2.965e+06	3.221e+06	9.220e+08
com-youtube	0.290 ± 0.070	0.501 ± 0.099	1.314 ± 0.257	7.617e+05	9.323e+05	3.561e+07
ogb-mag240m	85.14 ± 16.05	108.86 ± 7.994	549.51 ± 367.2	4.541e+07	5.820e+07	9.426e+09
ogbl-ppa	0.116 ± 0.019	0.202 ± 0.040	5.624 ± 1.108	6.117e+05	6.445e+05	1.682e+08
ogbn-arxiv	0.039 ± 0.060	0.058 ± 0.063	0.239 ± 0.104	1.158e+05	1.354e+05	1.473e+07
ogbn-mag	0.520 ± 0.079	0.921 ± 0.128	6.136 ± 0.974	1.804e+06	1.947e+06	2.050e+08
ogbn-papers100M	27.20 ± 3.94	41.99 ± 6.17	750.96 ± 110.45	8.893e+07	1.095e+08	1.604e+10
ogbn-products	0.695 ± 0.236	1.059 ± 0.249	31.907 ± 4.961	1.777e+06	2.385e+06	7.071e+08
ogbn-proteins	0.021 ± 0.057	0.025 ± 0.057	0.910 ± 0.306	7.941e+04	6.610e+04	1.488e+08
soc-ljl	1.040 ± 0.282	1.751 ± 0.487	7.102 ± 3.410	3.263e+06	4.045e+06	4.833e+08
soc-pokec	0.210 ± 0.027	0.368 ± 0.049	1.568 ± 0.207	2.020e+06	2.225e+06	2.160e+08
wiki-en21	1.436 ± 2.708	1.794 ± 0.215	19.658 ± 4.576	5.996e+06	6.659e+06	1.329e+09
wiki-talk	0.251 ± 0.049	0.455 ± 0.091	0.642 ± 0.071	1.090e+06	1.290e+06	4.284e+07

Table 4: Summary of runtime and operations for 15 datasets. ($\epsilon = 10^{-6}$)

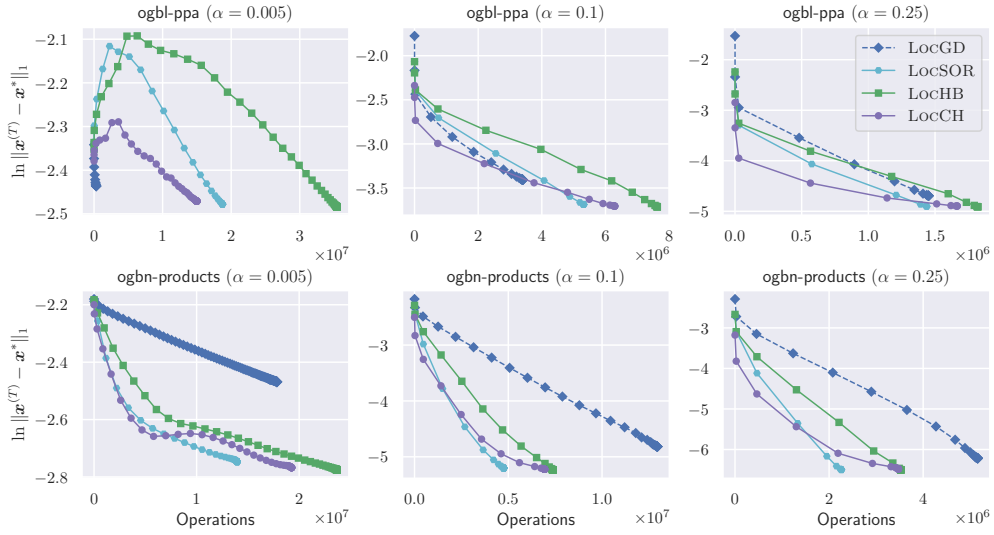


Figure 13: Estimation error as a function of operations needed. For $\alpha = 0.005$, $\alpha = 0.1$, and $\alpha = 0.25$.

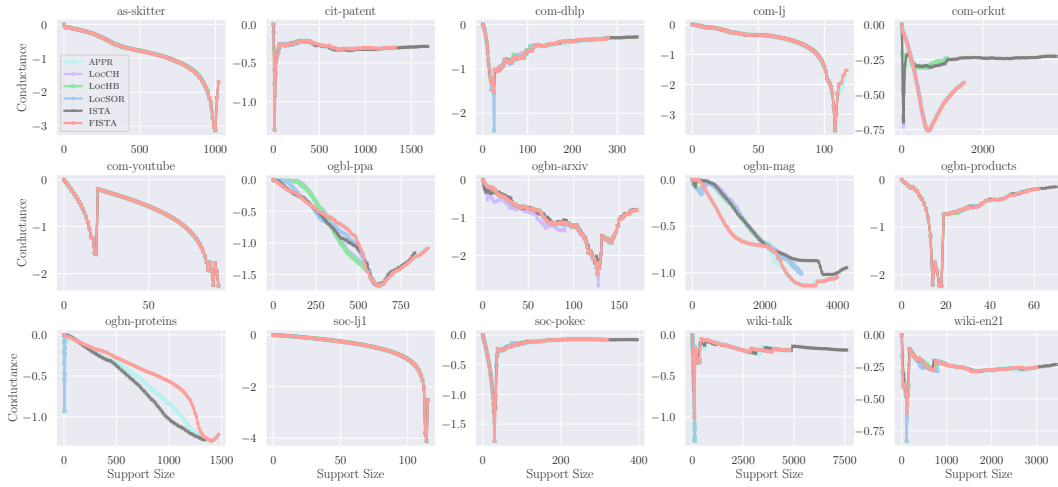


Figure 14: The graph conductance found by local graph clustering method using different local approximate methods. Experiments ran on 15 graphs. ($\epsilon = 10^{-6}$)

Dataset	APPR	LocCH	LoCHB	LocSOR	ISTA	FISTA
as-skitter	5.90e-04	5.90e-04	5.90e-04	5.90e-04	5.90e-04	5.90e-04
cit-patent	4.23e-02	4.23e-02	4.23e-02	4.23e-02	4.23e-02	4.23e-02
com-dblp	4.12e-03	4.12e-03	4.12e-03	4.12e-03	4.12e-03	4.12e-03
com-lj	2.94e-04	2.94e-04	2.94e-04	2.94e-04	2.94e-04	2.94e-04
com-orkut	1.75e-01	1.76e-01	1.76e-01	1.75e-01	1.76e-01	1.75e-01
com-youtube	5.46e-03	5.46e-03	5.46e-03	5.46e-03	5.46e-03	5.46e-03
ogbn-arxiv	2.11e-02	2.11e-02	2.11e-02	2.12e-02	2.14e-02	2.11e-02
ogbn-mag	3.18e-03	1.59e-03	3.18e-03	3.18e-03	4.74e-03	3.18e-03
ogbn-products	7.26e-02	1.02e-01	9.68e-02	9.65e-02	9.51e-02	7.24e-02
ogbn-proteins	5.92e-03	5.92e-03	5.92e-03	5.92e-03	5.92e-03	5.92e-03
soc-ljl	5.07e-02	1.18e-01	1.18e-01	1.18e-01	5.25e-02	5.07e-02
soc-pokec	7.80e-05	7.80e-05	7.80e-05	7.80e-05	7.80e-05	7.80e-05
wiki-talk	1.64e-02	1.64e-02	1.64e-02	1.64e-02	1.64e-02	1.64e-02
ogbl-ppa	5.13e-02	5.13e-02	5.13e-02	5.13e-02	5.13e-02	5.13e-02
wiki-en21	1.31e-01	1.31e-01	1.31e-01	1.31e-01	1.31e-01	1.31e-01

Table 5: The local conductance for six local solvers tested on 15 graphs datasets. ($\epsilon = 10^{-6}$)

Dataset	APPR	LocCH	LoCHB	LocSOR	ISTA	FISTA
as-skitter	0.127	0.147	0.144	0.043	0.323	0.093
cit-patent	0.362	0.516	0.457	0.125	0.939	0.308
com-dblp	0.069	0.033	0.028	0.014	0.297	0.042
com-lj	0.357	0.440	0.664	0.175	0.493	0.229
com-orkut	0.072	0.141	0.139	0.055	0.108	0.084
com-youtube	0.128	0.176	0.131	0.040	0.682	0.102
ogbl-ppa	0.102	0.047	0.091	0.027	0.146	0.102
ogbn-arxiv	0.042	0.013	0.014	0.006	0.237	0.032
ogbn-mag	0.068	0.137	0.091	0.035	0.253	0.090
ogbn-products	0.072	0.121	0.117	0.045	0.135	0.090
ogbn-proteins	0.005	0.005	0.005	0.002	0.005	0.004
soc-ljl	0.239	0.376	0.350	0.103	0.512	0.194
soc-pokec	0.067	0.096	0.059	0.033	0.222	0.051
wiki-talk	0.197	0.314	0.274	0.109	0.508	0.176
wiki-en21	0.121	0.290	0.279	0.106	0.197	0.147

Table 6: Runtime (seconds) for six local solvers tested on 15 graphs datasets. ($\epsilon = 10^{-6}$)

F.4 Results on local clustering

Dataset	APPR	LocCH	LocHB	LocSOR	ISTA	FISTA
as-skitter	6.9e+05	7.6e+04	8.1e+04	6.5e+04	2.9e+06	5.7e+05
cit-patent	6.7e+05	1.0e+05	1.1e+05	8.9e+04	2.3e+06	4.4e+05
com-dblp	4.3e+05	4.8e+04	5.0e+04	3.5e+04	1.9e+06	2.9e+05
com-lj	5.7e+05	8.9e+04	1.0e+05	7.6e+04	2.8e+06	4.4e+05
com-orkut	5.4e+05	8.9e+04	1.1e+05	9.0e+04	1.3e+06	5.0e+05
com-youtube	5.3e+05	6.5e+04	6.6e+04	5.7e+04	3.8e+06	4.4e+05
ogbl-ppa	6.7e+05	9.6e+04	1.1e+05	9.3e+04	1.5e+06	6.0e+05
ogbn-arxiv	6.8e+05	8.7e+04	9.7e+04	7.6e+04	2.7e+06	5.2e+05
ogbn-mag	5.2e+05	8.2e+04	9.4e+04	8.3e+04	2.4e+06	4.4e+05
ogbn-products	9.0e+05	1.1e+05	1.3e+05	9.8e+04	2.1e+06	7.6e+05
ogbn-proteins	5.3e+05	4.0e+04	5.5e+04	5.0e+04	6.8e+05	6.2e+05
soc-lj1	5.6e+05	8.9e+04	1.0e+05	7.7e+04	2.8e+06	4.4e+05
soc-pokec	5.9e+05	1.1e+05	1.3e+05	1.1e+05	2.5e+06	5.0e+05
wiki-talk	4.4e+05	5.5e+04	5.2e+04	4.6e+04	1.8e+06	3.4e+05
wiki-en21	5.2e+05	8.0e+04	9.7e+04	8.1e+04	1.8e+06	4.9e+05

Table 7: Operations Needed for six local solvers tested on 15 graphs datasets. ($\epsilon = 10^{-6}$)

G Related work

Many graph applications [2, 7, 29, 15, 27, 36, 30, 44, 25, 45, 51, 48] only require solving Eq. (1) approximately. The reasons could be either the most energies of π are among a small set of nodes forming small subgraphs, or one wants to study large graphs by checking them locally. Given a graph \mathcal{G} with n nodes and m edges, there are two main types of iterative solvers for Eq. (1) as follows:

Standard iterative methods. Methods for solving linear systems have been well-established over the past decades (see textbooks of Saad [43], Golub & Van Loan [19], Young [55]). The fastest linear solver for solving the symmetrized version of Eq. (1) is the Conjugate Gradient Method (CGM) with runtime complexity $\tilde{O}(m/\sqrt{\alpha})$ where m is the number of edges in the graph. It costs $\Theta(m)$ to access the entire graph at each iteration; hence, it is much slower than local solvers, as demonstrated in Fig. 15. The symmetric diagonally dominant (SDD) solvers advance CGM further to have complexity $\tilde{O}(m \log^c n \log(1/\epsilon))$ [31, 46]. Anikin et al. [4] considered the PageRank problem and proposed an algorithm with runtime depending on $\mathcal{O}(n)$. This paper focuses on local algorithms where the goal is to avoid the dominant factor m or n by avoiding the full Qx operation.

Local algorithms. Local solvers, in contrast to standard counterparts, leverage the fact that the energy of π lives in a small portion of the graph and hence do not require $\mathcal{O}(m)$ or $\mathcal{O}(n)$ per iteration. They are advantageous for huge-scale graphs demonstrated in Fig. 15. Andersen et al. [2] used APPR to obtain an approximate of π for local clustering. Quite similar algorithms were developed in Berkhin [6] (*bookmark-coloring* algorithm) and Kloster & Gleich [28] (Gauss-Southwell procedure).

Under the same stopping condition as APPR, Fountoulakis et al. [13] demonstrated that APPR is equivalent to coordinate descent via variational characterization, with a runtime of $\tilde{O}(1/(\alpha\epsilon))$ using ISTA where the monotonicity and conservation properties remain. Hence, it is nature to ask whether $\tilde{O}(1/(\sqrt{\alpha}\epsilon))$ could be achieved by FISTA [5] in Fountoulakis & Yang [12]. However, the difficulty is that FISTA violates the monotonicity property where the volume accessed of per-iteration cannot be bounded properly. To overcome this, Martínez-Rubio et al. [37] proposed a nested accelerated projected gradient descent (APGD) and gradually expanding solutions so that the monotonicity property still holds. However, nested APGD requires solving subproblems accurately, which in practice may be cumbersome if the precision requirement of the inner problem is too stringent. All current local methods rely on some monotonicity property of variables to guarantee locality, which does not exist in most accelerated frameworks; thus, developing an accelerated method that is guaranteed to preserve intermediate variable sparsity remains challenging.

It is worth mentioning that local methods are also closely related to sublinear time and local computational algorithms [42, 1]. From the optimization perspective, the equivalence between Gauss-Seidel and coordinate descent has been considered [49, 35, 39, 50] but does not focus on local analysis.

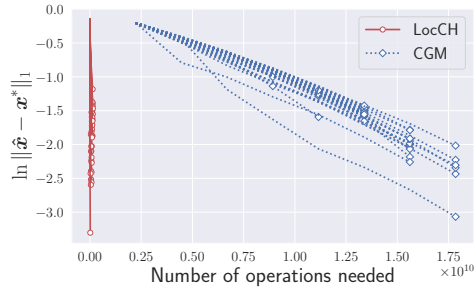


Figure 15: Comparison of the error reduction between the proposed LOCCH and the standard CGM on the papers100M dataset [23], in terms of the number of operations required.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: [The abstract and introduction clearly state the main claims made by the paper, including its key contributions and scope.](#)

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: [The accelerated algorithms, such as LOCCH and LOCHB, may exhibit instability when \$\alpha\$ is small. This limitation arises due to the inherent constraints of global methods.](#)

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory Assumptions and Proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: All assumptions are clearly stated.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental Result Reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We have provided our code for the review process and will make it publicly available upon publication.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: All datasets used in this study are publicly available.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental Setting/Details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: All parameter settings of our methods and baselines are included.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment Statistical Significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: For most of our results, we report the standard error over 50 random sampling nodes.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).

- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments Compute Resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: For our experiment, we used a server powered by an Intel(R) Xeon(R) Gold 5218R CPU, which features 40 cores (80 threads). The system is equipped with 256 GB of RAM.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code Of Ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The research conducted in the paper conforms in every respect with the NeurIPS Code of Ethics. The authors have thoroughly reviewed and adhered to the guidelines, ensuring that all aspects of their work align with ethical standards.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader Impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The advancements in accelerated algorithms like LOCCH and LOCHB can significantly enhance computational efficiency in various applications, contributing to faster and more effective solutions in fields such as data analysis, machine learning, and optimization.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: None.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [NA]

Justification: None.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New Assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: None.

Guidelines:

- The answer NA means that the paper does not release new assets.

- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and Research with Human Subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: No human subjects involved.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: No human subjects involved.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.