CPSC 490 : Senior Thesis December 15, 2015

Iterative Algorithms for Lipschitz Learning on Graphs

Advisor: Dan Spielman *Xiao Shi*

Abstract. This paper considers the problem of computing the Lex-Minimizer, where one is given a finite graph along with fixed values at some vertices and needs to compute a value assignment to the rest of the vertices such that the assignment lexicographically minimizes the infinity norm of the value differences across edges (in other words, to compute a value assignment that is as "smooth" as possible). This paper discusses the structural properties of the Lex-Minimizer, presents a few iterative algorithms which are based on these properties, and proves the convergence of these algorithms for computing the Lex-Minimizer.

Contents

1. Introduction 2
2. Properties of Lex-Minimizer 2
2.1. Max-Min Gradient Averaging Property 2
2.2. Linearity 3
3. Iterative Algorithms for Computing Lex-Minimizer 4
3.1. IterLex for Uniformly Weighted Graphs 4
3.2. IterLex for General Graphs 4
4. Related Work 5
4.1. Wide Usage of Lex-Minimizers and IterLex Algorithms 5
4.2. Alternative Formulations 5
4.3. Related Results 6
5. Convergence of IterLex (on Uniformly Weighted Graphs) 7
5.1. Continuity 8
5.2. Monotonicity 9
5.3. Proof of Convergence 10
6. Convergence of IterLex (on General Graphs) 11
6.1. Continuity 12
6.2. Monotonicity 14
7. Future Directions 15
7.1. Rate of Convergence 15
7.2. Bounding Errors 15
8. Acknowledgment 16
References 16

1. Introduction

We consider the *Lipschitz extension problem* on finite undirected graphs. Consider a weighted undirected graph $G = (V, E, \ell)$ and values $\boldsymbol{v_0} : T \mapsto \mathbb{R}$ on a subset T of its vertices. We only consider positive weights $\ell : E \mapsto \mathbb{R}_+$ and view the weights as indications of lengths of edges–shorter lengths indicate greater similarity.

Our goal is to assign values to every vertex $x \in V - T$ so that the values assigned (on all vertices) are as *smooth* as possible across edges. We will define *smoothness* precisely in later sections.

Throughout this paper, let $n = |V|$. As the vertices in T has fixed values, we call them *terminal* or *boundary* vertices. Another way of looking at the terminal values is to view $\boldsymbol{v_0} \in \mathbb{R}^n$ as a function $\boldsymbol{v_0} : V \mapsto \mathbb{R} \cup \{\perp\}$, where $\boldsymbol{v_0}(x) = \perp$ if and only if $x \notin T$. Given a vector $\boldsymbol{v} : V \mapsto \mathbb{R}$. We write $\boldsymbol{v}|T = \boldsymbol{v_0}|T$ say that $\boldsymbol{v}$ *extends* $\boldsymbol{v_0}$ if for every $x \in T$, $\boldsymbol{v}(x) = \boldsymbol{v_0}(x)$.

Inf-Minimizers. A minimal Lipschitz extension of $\boldsymbol{v_0}$ is a vector $\boldsymbol{v} : V \mapsto \mathbb{R}$ that minimizes

$$\max_{(x,y)\in E} \big(\ell(x,y)\big)^{-1} \big|\boldsymbol{v}(x) - \boldsymbol{v}(y)\big| \tag{1}$$

subject to $\boldsymbol{v}(x) = \boldsymbol{v_0}(x)$ for all $x \in T$.

As (1) is equivalent to the infinity norm of $\big(\ell(x,y)\big)^{-1} \big|\boldsymbol{v}(x) - \boldsymbol{v}(y)\big|$ across edges, we call a vector $\boldsymbol{v}$ that minimizes this objective an *Inf-Minimizer*.

Lex-Minmizer. Inf-minimizers are not unique, so among them, we seek vectors that minimizes the second-largest absolute value of $\big(\ell(x,y)\big)^{-1} \big|\boldsymbol{v}(x) - \boldsymbol{v}(y)\big|$ across edges, and then the third-largest given that, and so on. This way we obtain an *absolutely minimal Lipschitz extension* (AMLE) of $\boldsymbol{v_0}$. We call such a vector $\boldsymbol{v}^*$ the *Lex-Minizer*, which we have proven to be unique [KRSS15].

In this paper, we discuss some structural properties of Lex-Minimizer in Section 2; we then present a few iterative algorithms to calculate the Lex-Minimizer in Section 3; finally, we prove the convergence of these algorithms in Sections 5 and 6.

2. Properties of Lex-Minimizer

It turns out that the lex-minimizer has very nice structural properties, especially on uniformly weighted graphs.

2.1. **Max-Min Gradient Averaging Property.**

Definition 2.1 (Gradient due to an assignment)**.** *Given a vector $\boldsymbol{v} : V \mapsto \mathbb{R} \cup \{\perp\}$, $(x, y) \in E$, where $\boldsymbol{v}(x) \neq \perp$ and $\boldsymbol{v}(y) \neq \perp$. We define the gradient on (x, y) due to $\boldsymbol{v}$ to be*

$$\mathsf{grad}[\boldsymbol{v}](x, y) = \frac{\boldsymbol{v}(x) - \boldsymbol{v}(y)}{\ell(x, y)}.$$

Note that $\mathsf{grad}[\boldsymbol{v}](x, y) = -\mathsf{grad}[\boldsymbol{v}](y, x)$. We drop $[\boldsymbol{v}]$ when the context is clear.

We also define the gradient along a path $P = \langle p_1 \to p_2 \to \ldots \to p_k \rangle$, where $p_i \in V$ and $(p_i, p_{i+1}) \in E$, as

$$\mathsf{grad}[\boldsymbol{v}](P) = \frac{\boldsymbol{v}(p_1) - \boldsymbol{v}(p_k)}{\sum_{i=1}^{k-1} \ell(p_i, p_{i+1})}.$$

Note that $\mathsf{grad}[\boldsymbol{v}] \left(\langle p_1 \to p_2 \to \ldots \to p_k \rangle \right) = -\mathsf{grad}[\boldsymbol{v}] \left(\langle p_k \to p_{k-1} \to \ldots \to p_1 \rangle \right)$.

Definition 2.2 (Max-min gradient averaging property)**.** *The vector $\boldsymbol{v} : V \mapsto \mathbb{R}$ which extends $\boldsymbol{v_0}$ satisfies max-min gradient averaging property if for every non-terminal vertex $x \in V - T$,*

$$\max_{(x,y) \in E} \mathsf{grad}[\boldsymbol{v}](y, x) = - \min_{(x,y) \in E} \mathsf{grad}[\boldsymbol{v}](y, x) \tag{2}$$

Theorem 2.3. *Given $(G, \boldsymbol{v_0})$, the Lex-Minimizer $\boldsymbol{v}^*$ which extends $\boldsymbol{v_0}$ satisfies the max-min gradient averaging property. Moreover, it is the* ***unique*** *assignment extending $\boldsymbol{v_0}$ that satisfies this property with respect to $(G, \boldsymbol{v_0})$.*

Proof. See the proof of Theorem 3.10 in Appendix A.3 in [KRSS15]. □

Corollary 2.4 (For uniformly weighted graphs)**.** *Given a* uniformly *weighted graph $G(V, E, \ell)$ and $\boldsymbol{v_0}$, (i.e., $\ell(e) = c$ for all $e \in E$, where c is a postive constant,) the Lex-Minimizer $\boldsymbol{v}^*$ which extends $\boldsymbol{v_0}$ has the following property: for all $x \in G - T$,*

$$\boldsymbol{v}^*(x) = \frac{1}{2} \left(\max_{(x,y) \in E} \boldsymbol{v}^*(y) + \min_{(x,z) \in E} \boldsymbol{v}^*(z) \right) \tag{3}$$

2.2. Linearity.

Theorem 2.5 (Linearity of Lex-Minimizers)**.** *Given a graph G and two sets of terminal values $\boldsymbol{v_0}$ and $\hat{\boldsymbol{v}}_{\boldsymbol{0}}$, let $\boldsymbol{v}^*$ be the Lex-Minimizer that extends $\boldsymbol{v_0}$ and $\hat{\boldsymbol{v}}^*$ be the Lex-Minimizer that extends $\hat{\boldsymbol{v}}_{\boldsymbol{0}}$, if $\boldsymbol{v_0}$ and $\hat{\boldsymbol{v}}_{\boldsymbol{0}}$ satisfies*

$$\hat{\boldsymbol{v}}_{\boldsymbol{0}}(x) = a\boldsymbol{v_0}(x) + b, \forall x \in T, a \in \mathbb{R}, b \in \mathbb{R},$$

then for every vertex $y \in V$,

$$\hat{\boldsymbol{v}}^*(y) = a\boldsymbol{v}^*(y) + b.$$

Proof. Construct $\boldsymbol{v} \in \mathbb{R}^n$ such that for every vertex $y \in V$, $\boldsymbol{v}(y) = a\boldsymbol{v}^*(y) + b$. $\boldsymbol{v}$ obviously extends $\boldsymbol{v_0}$.

For every non-terminal vertex $x \in V - T$,

$$\begin{aligned}
&\max_{(x,y)\in E} \mathsf{grad}[\boldsymbol{v}](y,x) \\
=&\max_{(x,y)\in E} \frac{\boldsymbol{v}(y)-\boldsymbol{v}(x)}{\ell(y,x)} \\
=&\max_{(x,y)\in E} \frac{a\boldsymbol{v}^*(y)+b-\big(a\boldsymbol{v}^*(x)+b\big)}{\ell(y,x)} \\
=&a\max_{(x,y)\in E} \frac{\boldsymbol{v}^*(y)-\boldsymbol{v}^*(x)}{\ell(y,x)} \\
=&a\max_{(x,y)\in E} \mathsf{grad}[\boldsymbol{v}^*](y,x)
\end{aligned}$$

Similarly, we can show that

$$\min_{(x,y)\in E} \mathsf{grad}[\boldsymbol{v}](y,x) = a\min_{(x,y)\in E} \mathsf{grad}[\boldsymbol{v}^*](y,x).$$

Since $\boldsymbol{v}^*$ is a Lex-Minimizer, which must satisfy the max-min gradient averaging property: $\max_{(x,y)\in E} \mathsf{grad}[\boldsymbol{v}^*](y,x) = -\min_{(x,y)\in E} \mathsf{grad}[\boldsymbol{v}^*](y,x)$. Hence, $\max_{(x,y)\in E} \mathsf{grad}[\boldsymbol{v}](y,x) = -\min_{(x,y)\in E} \mathsf{grad}[\boldsymbol{v}](y,x)$, *i.e.*, $\boldsymbol{v}$ also satisfies the max-min gradient averaging property. By Theorem 2.3, it must be the Lex-Minimizer that extends $\boldsymbol{v_0}$. By uniqueness of the Lex-Minimizer, $\boldsymbol{v} = \hat{\boldsymbol{v}}^*$.

□

3. Iterative Algorithms for Computing Lex-Minimizer

Theorem 2.3 and Corollary 2.4 naturally give rise to a few iterative algorithms for compute Lex-Minimizers.

3.1. **IterLex for Uniformly Weighted Graphs.** Initially assign a value (say 0) for any vertex $u \notin T$, then simply iteratively take the average for the max and min neighbors until we get to an assignment with acceptable error (the termination criterion could be within absolute error ϵ, or a ϵ-approximation, etc.). More formally, see Algorithm 1.

3.2. **IterLex for General Graphs.** Analogously for general graphs, we could take the weighted average of the max and min neighbors. However, Theorem 2.3 permits two methods of picking the max and min neighbors. As specified below, Version 1 selects the neighbors whose respective gradients to the current vertex are the maximum and mininum; Version 2 selects the *pair* of neighbors whose gradient between them are the maximum.

Algorithm 1 IterLex for Uniformly Weighted Graphs

```
1: $\boldsymbol{v_1}|T \leftarrow \boldsymbol{v_0}|T$    ▷ shorthand for copying values for vertices in $T$
2: for all $x \in V - T$ do
3:     $\boldsymbol{v_1}(x) \leftarrow c$
4: end for    ▷ initial assignment, $c$ is a constant
5: $t \leftarrow 0$
6: while termination criterion is not met do
7:     $\boldsymbol{v_{t+1}}|T \leftarrow \boldsymbol{v_0}|T$
8:     for all $x \in V - T$ do
9:         $\boldsymbol{v_{t+1}}(x) = \frac{1}{2}\left(\max_{(x,y)\in E} \boldsymbol{v_t}(y) + \min_{(x,z)\in E} \boldsymbol{v_t}(x)\right)$
10:    end for
11:    $t \leftarrow t + 1$
12: end while
```

Algorithm 2 IterLex for General Graphs (Version 1)

```
1: $\boldsymbol{v_1}|T \leftarrow \boldsymbol{v_0}|T$
2: for all $x \in V - T$ do
3:     $\boldsymbol{v_1}(x) \leftarrow c$
4: end for    ▷ initial assignment, $c$ is a constant
5: $t \leftarrow 0$
6: while termination criterion is not met do
7:     $\boldsymbol{v_{t+1}}|T \leftarrow \boldsymbol{v_0}|T$
8:     for all $x \in V - T$ do
9:         $y \leftarrow \arg\max_{(x,y)\in E} \mathsf{grad}[\boldsymbol{v_t}](y, x)$
10:        $z \leftarrow \arg\min_{(x,z)\in E} \mathsf{grad}[\boldsymbol{v_t}](z, x)$
11:        $\boldsymbol{v_{t+1}}(x) \leftarrow \left(\ell(y,x)\boldsymbol{v_t}(z)+\ell(z,x)\boldsymbol{v_t}(y)\right)/\left(\ell(y,x)+\ell(z,x)\right)$
12:    end for
13:    $t \leftarrow t + 1$
14: end while
```

4. Related Work

4.1. **Wide Usage of Lex-Minimizers and IterLex Algorithms.** As Lex-Minimizers provide the "smoothest" extension, they are used in various areas such as image processing and machine learning ([GEL11], [ADE14], [EDLL14]). Some examples include image inpainting, image scaling, and cluster recognition.

4.2. **Alternative Formulations.** Lex-Minimizers have a lot of alternative formulations and related problems. It is the limit of the solutions to p-Laplacian

Algorithm 3 IterLex for General Graphs (Version 2)

```
1: $\boldsymbol{v_1}|T \leftarrow \boldsymbol{v_0}|T$
2: for all $x \in V - T$ do
3:     $\boldsymbol{v_1}(x) \leftarrow c$
4: end for                                   ▷ initial assignment, $c$ is a constant
5: $t \leftarrow 0$
6: while termination criterion is not met do
7:     $\boldsymbol{v_{t+1}}|T \leftarrow \boldsymbol{v_0}|T$
8:     for all $x \in V - T$ do
9:         $(y,z) \leftarrow \arg\max_{(x,y)\in E,(x,z)\in E} \mathsf{grad}[\boldsymbol{v_t}](y,z)$
10:        $\boldsymbol{v_{t+1}}(x) \leftarrow \big(\ell(y,x)\boldsymbol{v_t}(z)+\ell(z,x)\boldsymbol{v_t}(y)\big)/\big(\ell(y,x)+\ell(z,x)\big)$
11:    end for
12:    $t \leftarrow t+1$
13: end while
```

minimization problems as $p \to \infty$, namely, the vectors that solve

$$\min_{\boldsymbol{v}:V\mapsto\mathbb{R},\boldsymbol{v}|_T=\boldsymbol{v_0}|_T} \sum_{(x,y)\in E} \big(\ell(x,y)\big)^{-p} \left|\boldsymbol{v}(x)-\boldsymbol{v}(y)\right|^p . \tag{4}$$

Given that all weights are positive, (4)$\geq$ 0. Zero is actually achievable by solutions to p-harmonic functions. (See [And09] for details.) It follows that the Lex-Minimizer is also the limit of the solutions to p-harmonic functions.

The continuous counterpart of Lex-Minimization on finite graphs is infinity-Laplacian (Δ_∞) equations. [PSSW08] studies the ∞-Laplacians and relate them to tug-of-war games. [Obe05] discretizes Δ_∞ as AMLE (*i.e.*, Lex-Minimizers) and proves that the solutions of AMLE converges to the solution of ∞-Laplacians as the discretization becomes more and more granular. Lex-Minimizers are also closely related to the Dirichlet problems, simple stochastic games (also affectionately known as Anne Condon's games in the distributed systems community)... the list goes on.

4.3. **Related Results.** Before the algorithm that [KRSS15] proposed, the community has been using the IterLex algorithms, often without careful analysis. In fact, their convergence was an open question. The main contributions of this paper is a proof of convergence of IterLex Algorithms 1 and 3.

Andersson in his Master Thesis [And09] formulated Lex-Minimizer as the limit of solutions to p-harmonic functions. He presented an iterative algorithm for p-harmonic functions on uniformly weighted graphs, and proved its convergence. Unfortunately his proof breaks down when $p = \infty$.

[LLP+99] formulated the Lex-Minimization problem as calculating Richman costs. This is a special case of the Lipschitz extension problem, as it considers directed (but unweighted) graphs and has only two terminals. Lazarus et al. proved convergence of the iterative algorithm and showed an exponential rate of convergence.

5. Convergence of IterLex (on Uniformly Weighted Graphs)

In order to show the convergence of Algorithm 1, we formulate the original problem and the algorithm slightly differently.

Given undirected graph $G(V, E, \ell)$ that is uniformly weighted, let $\mathcal{H}(V)$ be the Hilbert space of all real valued functions (assignments) on the vertices of the graph, *i.e.*, each function $\boldsymbol{v} \in \mathcal{H}(V) : V \mapsto \mathbb{R}$ assigns a real value $\boldsymbol{v}(x)$ to each vertex $x \in V$.

The Set of Possible Assignments. Consider the set of potential assignments

$$K = \left\{ \boldsymbol{v} \in \mathcal{H}(V) : \boldsymbol{v}|_T = \boldsymbol{v_0}|_T; \forall x \in V, m \leq \boldsymbol{v}(x) \leq M \right\},$$

where $m = \min_{x \in T} \boldsymbol{v_0}(x)$ and $M = \max_{x \in T} \boldsymbol{v_0}(x)$.

K does not include all possible assignment that extends $\boldsymbol{v_0}$, but every assignment in K extends $\boldsymbol{v_0}$. Furthermore, it follows from Theorem 2.3 and the iteration procedure in Algorithm 1 that for any vertex $x \in V$, the value of x is always contained in $[m, M]$. Therefore, $\forall t \geq 0, \boldsymbol{v_t} \in K$.

If we identify $\mathcal{H}(V)$ to be $\mathbb{R}^n$, we notice that $K \subset \mathcal{H}(V)$ is a convex and compact subset of $\mathbb{R}^n$.

Update Functions. We then consider the iterations in Algorithm 1 to be update functions on assignments $\boldsymbol{v_t}$.

The *global update function* $F : K \mapsto K$ is thus

$$F(\boldsymbol{v})(x) = \begin{cases} \frac{1}{2}\left(\max_{(x,y)\in E} \boldsymbol{v}(y) + \min_{(x,z)\in E} \boldsymbol{v}(z)\right) & \text{if } x \notin T; \\ \boldsymbol{v}(x) & \text{if } x \in T. \end{cases}$$

Composition of Local Update Functions. Perceivably, we may choose not to update vertices altogether based on their values from the previous iteration. Instead, we can update the vertices one by one in each iteration.

Consider $F_i : K \mapsto K$, the local update function on vertex i, which takes the average of i's max and min neighbors and leaves values at other vertices intact:

$$F_i(\boldsymbol{v})(x) = \begin{cases} \frac{1}{2}\left(\max_{(x,y)\in E} \boldsymbol{v}(y) + \min_{(x,z)\in E} \boldsymbol{v}(z)\right) & \text{if } x = i \text{ and } x \notin T; \\ \boldsymbol{v}(x) & \text{otherwise.} \end{cases}$$

Given a permutation of the vertices $\pi \in S_n$, where S_n is the symmetric group of the n vertices. Then the composition of n local update functions can be a candidate for our update function:

$$F^{\pi}(\boldsymbol{v}) = F_{\pi(n)} \circ F_{\pi(n-1)} \circ \ldots \circ F_{\pi(1)}(\boldsymbol{v}).$$

Note that the global update function is not the same as the composition of local update functions, as the max-min neighbors might change as we apply the local updates one by one.

Lazy Update Function. Analogous to the lazy version of random walk, we can define a lazy version of the global update function $F^{\alpha} : K \mapsto K$ with parameter $0 \leq \alpha < 1$:

$$F^{\alpha}(\boldsymbol{v})(x) = \begin{cases} \alpha \boldsymbol{v}(x) + \frac{(1-\alpha)}{2}\left(\max_{(x,y)\in E} \boldsymbol{v}(y) + \min_{(x,z)\in E} \boldsymbol{v}(z)\right) & \text{if } x \notin T; \\ \boldsymbol{v}(x) & \text{if } x \in T. \end{cases}$$

Note that when $\alpha = 0$, F^{α} is exactly the same as F.

Lex-Minimizer as Fixed Point. Any fixed point $\boldsymbol{v}^{\dagger}$ of the global update function F (*i.e.*, $\boldsymbol{v}^{\dagger} \in K$ that satisfies $\boldsymbol{v}^{\dagger} = F(\boldsymbol{v}^{\dagger})$) corresponds exactly to the Lex-Minimizer as it satisfies the property in Corollary 2.4. From the existence and uniqueness of the Lex-Minimizer, the existence and uniqueness of fixed point follows. Since $\boldsymbol{v}^{\dagger} = \boldsymbol{v}^*$, we refer to the fixed point as $\boldsymbol{v}^*$ as well.

We can also show the existence of the fixed point using Brouwer's Fixed Point Theorem: every continuous function from a convex compact subset S of a Euclidean space to S itself has a fixed point. The map $\boldsymbol{v} \mapsto F(\boldsymbol{v})$ is continuous (See Section 5.1) and is from the convex compact subset $K \subset \mathbb{R}^n$ to itself, hence F has a fixed point. However, Brouwer's Fixed Point Theorem does not provide a construction process nor any indication of convergence using fixed point iteration.

5.1. **Continuity.** We now prove two important properties of the global update function–continuity and monotonicity–in order to show convergence of Algorithm 1.

Lemma 5.1 (Continuity of the global update function)**.** *The global update function F is continuous.*

Proof. Suppose we have $\boldsymbol{v}, \hat{\boldsymbol{v}} \in K$, such that $\|\boldsymbol{v} - \hat{\boldsymbol{v}}\|_{\infty} \leq \epsilon$, *i.e.*, for all $x \in V$, $\left|\boldsymbol{v}(x) - \hat{\boldsymbol{v}}(x)\right| \leq \epsilon$.

Since $\boldsymbol{v}, \hat{\boldsymbol{v}} \in K$, $\boldsymbol{v}|_T = \boldsymbol{v_0}|_T = \hat{\boldsymbol{v}}|_T$.

For each vertex $x \in V - T$, even though x may have different max (resp. min) neighbors in assignments $\boldsymbol{v}$ and $\hat{\boldsymbol{v}}$, the values of its max (resp. min) neighbors cannot be more than ϵ apart. Hence $\left|F(\boldsymbol{v})(x) - F(\hat{\boldsymbol{v}})(x)\right| \leq \epsilon$.

Thus the global update function is actually uniformly continuous:

$$\left\|F(\boldsymbol{v}) - F(\hat{\boldsymbol{v}})\right\|_{\infty} = \max_{x\in V} \left|F(\boldsymbol{v})(x) - F(\hat{\boldsymbol{v}})(x)\right| \leq \epsilon.$$

□

Corollary 5.2 (Continuity of variants of update functions)**.** *The local update function F_i, the composition of local update functions F^π, and the lazy global update function are all continuous.*

Proof. This follows from the same argument as in Lemma 5.1. Again assumee we have $\boldsymbol{v}, \hat{\boldsymbol{v}} \in K$, such that $\|\boldsymbol{v} - \hat{\boldsymbol{v}}\|_\infty \leq \epsilon$.

For the local update function, apply the argument in Lemma 5.1 to the vertex i. For the rest of the vertices, $\left|F(\boldsymbol{v})(x) - F(\hat{\boldsymbol{v}})(x)\right| \leq \epsilon$ follows directly from assumption.

Since F^π is a composition of continuous functions, it is itself continous.

For the lazy version, we obtain that for $x \in V - T$,

$$\left|F(\boldsymbol{v})(x) - F(\hat{\boldsymbol{v}})(x)\right| \leq (1-\alpha)\epsilon.$$

Since $\alpha \in [0,1)$,

$$\left\|F(\boldsymbol{v}) - F(\hat{\boldsymbol{v}})\right\|_\infty = \max_{x \in V} \left|F(\boldsymbol{v})(x) - F(\hat{\boldsymbol{v}})(x)\right| \leq \epsilon.$$

□

5.2. **Monotonicity.**

Lemma 5.3 (Monotonicity of the global update function)**.** *If $\boldsymbol{v}, \hat{\boldsymbol{v}} \in K$ satisfy $\boldsymbol{v}(x) \leq \hat{\boldsymbol{v}}(x)$ for all $x \in V$, then $F(\boldsymbol{v})(x) \leq F(\hat{\boldsymbol{v}})(x)$ for all $x \in V$.*

Proof. First note that terminal nodes always satisfy this property: if $x \in T$, by definition, $\boldsymbol{v}(x) = \hat{\boldsymbol{v}}(x) = \boldsymbol{v_0}(x)$ and $F(\boldsymbol{v})(x) = F(\hat{\boldsymbol{v}})(x) = \boldsymbol{v_0}(x)$.

Now we consider any non-terminal vertex $x \in V - T$.

Let $x^{\boldsymbol{v}}_{\max}$ be x's max-neighbor in $\boldsymbol{v}$, *i.e.*,

$$x^{\boldsymbol{v}}_{\max} = \arg\max_{(x,y)\in E} \boldsymbol{v}(y).$$

We then have

$$\max_{(x,y)\in E} \boldsymbol{v}(y) = \boldsymbol{v}(x^{\boldsymbol{v}}_{\max}) \leq \hat{\boldsymbol{v}}(x^{\boldsymbol{v}}_{\max}) \leq \max_{(x,y)\in E} \hat{\boldsymbol{v}}(y).$$

Similarly, let $x^{\hat{\boldsymbol{v}}}_{\min}$ be x's min-neighbor in $\hat{\boldsymbol{v}}$, *i.e.*,

$$x^{\hat{\boldsymbol{v}}}_{\min} = \arg\min_{(x,z)\in E} \hat{\boldsymbol{v}}(z).$$

We then have

$$\min_{(x,z)\in E} \boldsymbol{v}(z) \leq \boldsymbol{v}(x^{\hat{\boldsymbol{v}}}_{\min}) \leq \hat{\boldsymbol{v}}(x^{\hat{\boldsymbol{v}}}_{\min}) = \min_{(x,z)\in E} \hat{\boldsymbol{v}}(z).$$

Therefore,

$$F(\boldsymbol{v})(x) = \frac{1}{2}\left(\max_{(x,y)\in E} \boldsymbol{v}(y) + \min_{(x,z)\in E} \boldsymbol{v}(z)\right) \leq \frac{1}{2}\left(\max_{(x,y)\in E} \hat{\boldsymbol{v}}(y) + \min_{(x,z)\in E} \hat{\boldsymbol{v}}(z)\right) = F(\hat{\boldsymbol{v}})(x).$$

□

It follows from the same line of reasoning that the other variants of update functions are also monotonic.

Corollary 5.4 (Monotonicity of variants of update functions)**.** *The local update function F_i, the composition of local update functions F^π, and the lazy global update function are all monotonic.*

Note that we do not have *strict* monotonicity, *i.e.*, if f and g are two elements in K such that $f(u) < g(u)$ for all $u \in V - T$, we do *not* have $F(f)(u) < F(g)(u)$. The simplest counter example would be a path graph with 3 vertices, the two vertices on the end are terminals. Applying the update function will always give you the same value for the single non-terminal node, which is the average of the two terminals values. Fortunately, we do not need strict monotonicity to argue that Algorithm 1 converges to the Lex-Minimizer.

5.3. **Proof of Convergence.** Consider the sequence of assignments $\{\boldsymbol{v_t}\}$ we obtain by running Algorithm 1.

Initially, we start with $\boldsymbol{v_1}$. We set the terminal values accordingly: $\boldsymbol{v_1}|_T \leftarrow \boldsymbol{v_0}|_T$. We set all the non-terminal vertices to be $c \leftarrow m = \min_{x \in T} \boldsymbol{v_0}(x)$, where the constant c is defined in Algorithm 1 and m is the minimum among all the terminal values. We then keep applying the global update function:

$$\begin{aligned}
\boldsymbol{v_2} &= F(\boldsymbol{v_1}) \\
\boldsymbol{v_3} &= F\left(F(\boldsymbol{v_1})\right) = F(\boldsymbol{v_2}) \\
&\dots \\
\boldsymbol{v_t} &= F\left(\boldsymbol{v_{t-1}}\right) \\
&\dots
\end{aligned}$$

Definition 5.5 (Partial order on the set K)**.** *Given $\boldsymbol{v}, \hat{\boldsymbol{v}} \in K$, we write $\boldsymbol{v} \le \hat{\boldsymbol{v}}$ if and only if $\forall x \in V, \boldsymbol{v}(x) \le \hat{\boldsymbol{v}}(x)$.*

Remark 5.6. *Given this partial order, note that $\boldsymbol{v_1}$ is the "minimal" assignment in the set K, i.e., $\forall \boldsymbol{v} \in K$, we have $\boldsymbol{v_1} \le \boldsymbol{v}$.*

Lemma 5.7 (Boundedness of $\{\boldsymbol{v_t}\}$)**.** *$\forall t \ge 1$, $\boldsymbol{v_t} \le \boldsymbol{v}^*$, where $\boldsymbol{v}^* = F(\boldsymbol{v}^*)$ is the Lex-Minimizer.*

Proof. We show this by induction on t. The base case $\boldsymbol{v_1} \le \boldsymbol{v}^*$ follows from Remark 5.6. Now assume $\boldsymbol{v_{t-1}} \le \boldsymbol{v}^*$. By Lemma 5.3, $F(\boldsymbol{v_{t-1}}) \le F(\boldsymbol{v}^*)$. Hence,

$$\boldsymbol{v_t} = F(\boldsymbol{v_{t-1}}) \le F(\boldsymbol{v}^*) = \boldsymbol{v}^*.$$

□

Lemma 5.8 (Monotonicity of $\{\boldsymbol{v_t}\}$)**.** *$\forall t > 1$, $\boldsymbol{v_{t-1}} \le \boldsymbol{v_t}$.*

Proof. We again induct on t. The base case $t = 2$, $\boldsymbol{v_1} \leq \boldsymbol{v_2}$ follows from Remark 5.6. Now assume for $t > 2$, $\boldsymbol{v_{t-2}} \leq \boldsymbol{v_{t-1}}$. By Lemma 5.3, $F(\boldsymbol{v_{t-2}}) \leq F(\boldsymbol{v_{t-1}})$. Hence,

$$\boldsymbol{v_{t-1}} = F(\boldsymbol{v_{t-2}}) \leq F(\boldsymbol{v_{t-1}}) = \boldsymbol{v_t}.$$

□

Theorem 5.9 (Convergence of IterLex)**.** $\lim_{t\to\infty} \boldsymbol{v_t} \to \boldsymbol{v}^*$.

Proof. Lemma 5.7 and Lemma 5.8 state that for any vertex $x \in V$, the sequence $\{\boldsymbol{v_t}(x)\}$ is monotonically non-decreasing and upper-bounded by $\boldsymbol{v}^*(x)$. Hence, by the Monotone Convergence Theorem (of real numbers), the sequence $\{\boldsymbol{v_t}(x)\}$ converges to $\sup_{t\geq 1} \{\boldsymbol{v_t}(x)\}$. Therefore the sequence $\{\boldsymbol{v_t}\}$ *pointwise* converges to a limit $\tilde{\boldsymbol{v}}$, where $\tilde{\boldsymbol{v}}(x) = \sup_{t\geq 1} \{\boldsymbol{v_t}(x)\}$ for all $x \in V$.

Now we show that the limit $\tilde{\boldsymbol{v}} = \boldsymbol{v}^*$, the Lex-Minimizer.

Since $F(\cdot)$ is continuous on K and $K \subset R^n$ is Hausdorff, we can interchange limit and function application. Hence,

$$F(\tilde{\boldsymbol{v}}) = F(\lim_{t\to\infty} \boldsymbol{v_t}) = \lim_{t\to\infty} F(\boldsymbol{v_t}) = \lim_{t\to\infty} \boldsymbol{v_{t+1}} = \tilde{\boldsymbol{v}}.$$

$\tilde{\boldsymbol{v}}$ is a fixed point of F, but since we know the unique fixed point is the Lex-Minimizer, we must have $\tilde{\boldsymbol{v}} = \boldsymbol{v}^*$. □

This tells us that Algorithm 1 *pointwise* converges to the Lex-Minimizer $\boldsymbol{v}^*$ in the limit.

6. Convergence of IterLex (on General Graphs)

The convergence of Algorithm 3 follows from a similar argument. However, the argument for the continuity and monotonicity is more complicated.

Now we consider an weighted undirected graph $G(V, E, \ell)$, where $\ell : E \mapsto \mathbb{R}_+$, and an initial set of terminal values $\boldsymbol{v_0} : V \mapsto \mathbb{R}^n \cup \{\perp\}$. We still let the set of possible assignments be K, and the Lex-Minimizer be $\boldsymbol{v}^*$.

Definition 6.1 (Weighted average of two neighbors)**.** *For any non-terminal vertex $x \in V - T$ and two of its neighbors y and z, i.e., $(x, y) \in E, (x, z) \in E$, we define the weighted average of the two neighbors in assignment $\boldsymbol{v}$ as*

$$w_{\boldsymbol{v}}(y, z) \triangleq \frac{\ell(x, y)\boldsymbol{v}(z) + \ell(x, z)\boldsymbol{v}(y)}{\ell(x, y) + \ell(x, z)}.$$

We drop the subscript when the context is clear.

Remark 6.2. $\forall \boldsymbol{v} \in K, x \in V, (x, y) \in E, (x, z) \in E$, $w_{\boldsymbol{v}}(z, y) = w_{\boldsymbol{v}}(y, z)$.

Remark 6.3. *Consider any assignment $\boldsymbol{v} \in K$, any vertex $x \in V - T$ and its three neighbors u, y, z. If $\mathsf{grad}[\boldsymbol{v}]\left(\langle y \to x \to u\rangle\right) \leq \mathsf{grad}[\boldsymbol{v}]\left(\langle y \to x \to z\rangle\right)$, or equivalently $\mathsf{grad}[\boldsymbol{v}]\left(\langle u \to x \to y\rangle\right) \geq \mathsf{grad}[\boldsymbol{v}]\left(\langle z \to x \to y\rangle\right)$, then $w(y, u) \geq w(y, z)$.*

Definition 6.4 (Weighted global update function)**.**

$$\tilde{F}(\boldsymbol{v})(x) = w_{\boldsymbol{v}}(y,z), \textit{ where } (y,z) = \underset{(x,y)\in E,(x,z)\in E}{\arg\max} \ \mathsf{grad}[\boldsymbol{v}]\left(\langle y \to x \to z\rangle\right).$$

6.1. **Continuity.**

Lemma 6.5 (Continuity of weighted global update function)**.** *The weighted global update function $\tilde{F}$ is continuous.*

Proof. Suppose we have $\boldsymbol{v}, \hat{\boldsymbol{v}} \in K$, such that $\|\boldsymbol{v}-\hat{\boldsymbol{v}}\|_\infty \le \epsilon$, *i.e.*, for all $x \in V$, $\left|\boldsymbol{v}(x)-\hat{\boldsymbol{v}}(x)\right| \le \epsilon$. Consider any non-terminal vertex $x \in V - T$. We first observe that if vertices y and z are the neighbors of x, we have

$$\left|w_{\boldsymbol{v}}(y,z) - w_{\hat{\boldsymbol{v}}}(y,z)\right| \le \frac{\ell(x,y)\left|\boldsymbol{v}(z)-\hat{\boldsymbol{v}}(z)\right| + \ell(x,z)\left|\boldsymbol{v}(y)-\hat{\boldsymbol{v}}(y)\right|}{\ell(x,y)+\ell(x,z)} \le \epsilon. \tag{5}$$

In this proof, we use $\mathsf{grad}[\cdot](y,z)$ solely in the context of (and therefore depends on) x, *i.e.*, for any $\boldsymbol{u} \in K$, we overload the notation

$$\mathsf{grad}[\boldsymbol{u}](y,z) \triangleq \mathsf{grad}[\boldsymbol{u}](\langle y \to x \to z\rangle) = \frac{\boldsymbol{u}(y)-\boldsymbol{u}(z)}{\ell(x,y)+\ell(x,z)}.$$

We then further observe that

$$\left|\mathsf{grad}[\boldsymbol{v}](y,z) - \mathsf{grad}[\hat{\boldsymbol{v}}](y,z)\right| \le \frac{\left|\boldsymbol{v}(z)-\hat{\boldsymbol{v}}(z)\right| + \left|\boldsymbol{v}(y)-\hat{\boldsymbol{v}}(y)\right|}{\ell(x,y)+\ell(x,z)} \le \frac{2\epsilon}{\ell(x,y)+\ell(x,z)}$$

. We write the righthand side as $\Delta_{y,z}$ for short.

Let vertices α and β be the neighbors of x that maximize $\mathsf{grad}[\boldsymbol{v}](\beta,\alpha)$; let vertices γ and δ be the neighbors of x that maximize $\mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)$.

By our previous observation,

$$\left|\mathsf{grad}[\boldsymbol{v}](\delta,\gamma) - \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)\right| \le \frac{2\epsilon}{\ell(x,\delta)+\ell(x,\gamma)} = \Delta_{\delta,\gamma}, \tag{6}$$

$$\left|\mathsf{grad}[\boldsymbol{v}](\beta,\alpha) - \mathsf{grad}[\hat{\boldsymbol{v}}](\beta,\alpha)\right| \le \frac{2\epsilon}{\ell(x,\beta)+\ell(x,\alpha)} = \Delta_{\beta,\alpha}. \tag{7}$$

Therefore,

$$\begin{aligned}
&\mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma) \\
\le&\mathsf{grad}[\boldsymbol{v}](\delta,\gamma) + \Delta_{\delta,\gamma} && \big(\text{By } (6)\big) \\
\le&\mathsf{grad}[\boldsymbol{v}](\beta,\alpha) + \Delta_{\delta,\gamma} && \big(\text{Since } (\beta,\alpha) \text{ maximize } \mathsf{grad}[\boldsymbol{v}](\cdot,\cdot)\big) \\
\le&\mathsf{grad}[\hat{\boldsymbol{v}}](\beta,\alpha) + \Delta_{\beta,\alpha} + \Delta_{\delta,\gamma} && \big(\text{By } (7)\big)
\end{aligned}$$

Hence

$$\mathsf{grad}[\hat{\boldsymbol{v}}](\beta,\alpha) \le \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma) \le \mathsf{grad}[\hat{\boldsymbol{v}}](\beta,\alpha) + \Delta_{\beta,\alpha} + \Delta_{\delta,\gamma}. \tag{8}$$

Now we want to relate their weighted averages. Recall that

$$w_{\hat{\boldsymbol{v}}}(\gamma,\delta) = \hat{\boldsymbol{v}}(\gamma) + \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)\cdot \ell(\gamma,x) = \hat{\boldsymbol{v}}(\delta) - \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)\cdot \ell(\delta,x).$$

Since $\mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma) \geq \mathsf{grad}[\hat{\boldsymbol{v}}](\beta,\gamma)$, we have

$$\begin{aligned}\hat{\boldsymbol{v}}(\beta) =& \hat{\boldsymbol{v}}(\gamma) + \mathsf{grad}[\hat{\boldsymbol{v}}](\beta,\gamma)\cdot\big(\ell(\beta,x)+\ell(\gamma,x)\big)\\ \leq& \hat{\boldsymbol{v}}(\gamma) + \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)\cdot\big(\ell(\beta,x)+\ell(\gamma,x)\big)\\ =& w_{\hat{\boldsymbol{v}}}(\gamma,\delta) + \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)\cdot\ell(\beta,x).\end{aligned} \tag{9}$$

Since $\mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma) \geq \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\alpha)$, we have

$$\begin{aligned}\hat{\boldsymbol{v}}(\alpha) =& \hat{\boldsymbol{v}}(\delta) - \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\alpha)\cdot\big(\ell(\alpha,x)+\ell(\delta,x)\big)\\ \geq& \hat{\boldsymbol{v}}(\delta) - \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)\cdot\big(\ell(\alpha,x)+\ell(\delta,x)\big)\\ =& w_{\hat{\boldsymbol{v}}}(\gamma,\delta) - \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)\cdot\ell(\alpha,x).\end{aligned} \tag{10}$$

Therefore,

$$\begin{aligned}w_{\hat{\boldsymbol{v}}}(\alpha,\beta) =& \hat{\boldsymbol{v}}(\beta) - \mathsf{grad}[\hat{\boldsymbol{v}}](\beta,\alpha)\cdot\ell(\beta,x)\\ \leq& w_{\hat{\boldsymbol{v}}}(\gamma,\delta) + \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)\cdot\ell(\beta,x) - \mathsf{grad}[\hat{\boldsymbol{v}}](\beta,\alpha)\cdot\ell(\beta,x) && \big(\text{By (9)}\big)\\ \leq& w_{\hat{\boldsymbol{v}}}(\gamma,\delta) + (\Delta_{\beta,\alpha}+\Delta_{\delta,\gamma})\cdot\ell(\beta,x). && \big(\text{By (8)}\big)\end{aligned}$$

Similarly, we obtain a lower bound,

$$\begin{aligned}w_{\hat{\boldsymbol{v}}}(\alpha,\beta) =& \hat{\boldsymbol{v}}(\alpha) + \mathsf{grad}[\hat{\boldsymbol{v}}](\beta,\alpha)\cdot\ell(\alpha,x)\\ \geq& w_{\hat{\boldsymbol{v}}}(\gamma,\delta) - \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)\cdot\ell(\alpha,x) + \mathsf{grad}[\hat{\boldsymbol{v}}](\beta,\alpha)\cdot\ell(\alpha,x) && \big(\text{By (10)}\big)\\ \geq& w_{\hat{\boldsymbol{v}}}(\gamma,\delta) - (\Delta_{\beta,\alpha}+\Delta_{\delta,\gamma})\cdot\ell(\alpha,x). && \big(\text{By (8)}\big)\end{aligned}$$

Let $\ell_{\min} = \min_{e\in E}\ell(e)$ and $\ell_{\max} = \max_{e\in E}\ell(e)$. As the graph G is fixed, $\ell_{\min}$ and $\ell_{\max}$ can be considered as constants. Then we have

$$\begin{aligned}&\big|w_{\hat{\boldsymbol{v}}}(\gamma,\delta) - w_{\hat{\boldsymbol{v}}}(\alpha,\beta)\big|\\ \leq&(\Delta_{\beta,\alpha}+\Delta_{\delta,\gamma})\cdot\max\big\{\ell(\alpha,x),\ell(\beta,x)\big\}\\ =&\left(\frac{2\epsilon}{\ell(x,\beta)+\ell(x,\alpha)}+\frac{2\epsilon}{\ell(x,\delta)+\ell(x,\gamma)}\right)\cdot\max\big\{\ell(\alpha,x),\ell(\beta,x)\big\}\\ \leq&\left(\frac{2\epsilon}{\ell_{\min}+\ell_{\min}}+\frac{2\epsilon}{\ell_{\min}+\ell_{\min}}\right)\cdot\ell_{\max}\\ \leq&\frac{2\ell_{\max}}{\ell_{\min}}\cdot\epsilon.\end{aligned} \tag{11}$$

It follows that

$$\begin{aligned}
&\left|\tilde{F}(\boldsymbol{v})(x) - \tilde{F}(\hat{\boldsymbol{v}})(x)\right| \\
\leq &\left|w_{\boldsymbol{v}}(\beta,\alpha) - w_{\hat{\boldsymbol{v}}}(\delta,\gamma)\right| \\
\leq &\left|w_{\boldsymbol{v}}(\beta,\alpha) - w_{\hat{\boldsymbol{v}}}(\beta,\alpha)\right| + \left|w_{\hat{\boldsymbol{v}}}(\beta,\alpha) - w_{\hat{\boldsymbol{v}}}(\delta,\gamma)\right| && \text{(triangular inequality)} \\
\leq &\epsilon + \frac{2\ell_{\max}}{\ell_{\min}} \cdot \epsilon && \big(\text{By (5) and (11)}\big) \\
= &\left(1 + \frac{2\ell_{\max}}{\ell_{\min}}\right) \cdot \epsilon.
\end{aligned}$$

Therefore,

$$\left\|\tilde{F}(\boldsymbol{v}) - \tilde{F}(\hat{\boldsymbol{v}})\right\|_\infty = \max_{x\in V}\left|\tilde{F}(\boldsymbol{v})(x) - \tilde{F}(\hat{\boldsymbol{v}})(x)\right| \leq \left(1 + \frac{2\ell_{\max}}{\ell_{\min}}\right) \cdot \epsilon.$$

$\tilde{F}$ is continuous.

□

6.2. **Monotonicity.**

Lemma 6.6 (Monotonicity of weighted global update function)**.** *If $\boldsymbol{v}, \hat{\boldsymbol{v}} \in K$ satisfy $\boldsymbol{v}(x) \geq \hat{\boldsymbol{v}}(x)$ for all $x \in V$, then $\tilde{F}(\boldsymbol{v})(x) \geq \tilde{F}(\hat{\boldsymbol{v}})(x)$ for all $x \in V$.*

Proof. Consider any non-terminal vertex $x \in V - T$.

We again use overload the notation $\mathsf{grad}[\cdot](\cdot,\cdot)$, *i.e.*, for any $\boldsymbol{u} \in K$ and x's neighbors y and z,

$$\mathsf{grad}[\boldsymbol{u}](y,z) \triangleq \mathsf{grad}[\boldsymbol{u}](\langle y \to x \to z\rangle) = \frac{\boldsymbol{u}(y) - \boldsymbol{u}(z)}{\ell(x,y) + \ell(x,z)}.$$

Let vertices α and β be the neighbors of x that maximize $\mathsf{grad}[\boldsymbol{v}](\beta,\alpha)$; let vertices γ and δ be the neighbors of x that maximize $\mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)$. Therefore, $\boldsymbol{v}(\beta) \geq \boldsymbol{v}(\alpha)$, $\hat{\boldsymbol{v}}(\delta) \geq \hat{\boldsymbol{v}}(\gamma)$, $\mathsf{grad}[\boldsymbol{v}](\delta,\alpha) \leq \mathsf{grad}[\boldsymbol{v}](\beta,\alpha)$, and $\mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\alpha) \leq \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)$.

$$
\begin{aligned}
&\tilde{F}(\boldsymbol{v})(x) \\
=&w_{\boldsymbol{v}}(\beta,\alpha) \\
\geq&w_{\boldsymbol{v}}(\delta,\alpha) && \big(\text{by Remark 6.3 given } \mathsf{grad}[\boldsymbol{v}](\delta,\alpha) \leq \mathsf{grad}[\boldsymbol{v}](\beta,\alpha)\big) \\
=&\frac{\ell(\delta,x)\boldsymbol{v}(\alpha)+\ell(\alpha,x)\boldsymbol{v}(\delta)}{\ell(\delta,x)+\ell(\alpha,x)} \\
\geq&\frac{\ell(\delta,x)\hat{\boldsymbol{v}}(\alpha)+\ell(\alpha,x)\hat{\boldsymbol{v}}(\delta)}{\ell(\delta,x)+\ell(\alpha,x)} && \big(\boldsymbol{v}(\alpha)\geq\hat{\boldsymbol{v}}(\alpha), \boldsymbol{v}(\delta)\geq\hat{\boldsymbol{v}}(\delta), \ell(\cdot,\cdot)>0\big) \\
=&w_{\hat{\boldsymbol{v}}}(\delta,\alpha) \\
\geq&w_{\hat{\boldsymbol{v}}}(\delta,\gamma) && \big(\text{by Remark 6.3 given } \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\alpha) \leq \mathsf{grad}[\hat{\boldsymbol{v}}](\delta,\gamma)\big) \\
=&\tilde{F}(\hat{\boldsymbol{v}})(x)
\end{aligned}
$$

□

7. Future Directions

7.1. **Rate of Convergence.**

7.1.1. *Path Graphs.* Using spectral methods, without much effort, one can show that if G is an uniformly weighted path graph P_n, then

$$\|\boldsymbol{v_t}-\boldsymbol{v}^*\| \leq (1-\lambda)^t \|\boldsymbol{v_1}-\boldsymbol{v}^*\| \approx e^{-\lambda t}\|\boldsymbol{v_1}-\boldsymbol{v}^*\|,$$

where $\lambda = 1-\cos(\pi/n-1)$ is the spectral gap of the adjacency matrix $1/2\boldsymbol{A}_{P_{n-2}}$.

Therefore, to get $\|\boldsymbol{v_t}-\boldsymbol{v}^*\| \leq \epsilon\|\boldsymbol{v_0}-\boldsymbol{v}^*\|$, simply require $t \geq \frac{1}{\lambda}\log(1/\epsilon)$.

7.1.2. *General Graphs.* Although the IterLex algorithms might not be as fast as the ones presented in [KRSS15], their programming complexity is much lower. It would be very nice to obtain a theoretical bound for the rate of convergence in the general case.

7.2. **Bounding Errors.**

One way to exploit the monotonicity of the update functions is using it to estimate or bound the error $\|\boldsymbol{v_t}-\boldsymbol{v}^*\|$.

It follows from the above sections that given an assignment $\boldsymbol{v}\in K$ and any vertex $x\in V$, we have $\big|F(\boldsymbol{v})(x)-\boldsymbol{v}^*(x)\big| \geq \big|F(\boldsymbol{v})(x)-\boldsymbol{v}(x)\big|$, a lower bound on the error. An upper bound would be very useful.

Another possible way to exploit monotonicity is approaching $\boldsymbol{v}^*$ from both above and below. Specifically, when running Algorithm 1, concurrently run it on another set of assignments $\boldsymbol{v_t'}$, where $\boldsymbol{v_1'}$ is obtained by setting $c \leftarrow M = \max_{x\in T}\boldsymbol{v_0}(x)$. It's easy to show that $\{\boldsymbol{v_t'}\}$ is monotonically non-increasing and converges to $\boldsymbol{v}^*$. Therefore we have $\boldsymbol{v_t} \leq \boldsymbol{v}^* \leq \boldsymbol{v_t'}$ for every t. The maximum error is then bounded by $\|\boldsymbol{v_t'}-\boldsymbol{v_t}\|_\infty$.

8. Acknowledgment

I am indebted to my advisor Prof. Dan Spielman for his insights, patience, and support. Without him, this work would not have been possible. I would also like to thank Sushant Sachdeva, Rasmus Kyng, and Xinwei (David) Yao for their discussions and encouragement.

References

[ADE14] Sadia Alkama, Xavier Desquesnes, and Abderrahim Elmoataz. Infinity laplacian on graphs with gradient terms for image and data clustering. *Pattern Recognition Letters*, 41:65 – 72, 2014. Supervised and Unsupervised Classification Techniques and their Applications.

[And09] Karl T. Andersson. An iterative solution method for p-harmonic functions on finite graphs with an implementation. 2009. (Master Thesis).

[EDLL14] Abderrahim Elmoataz, Xavier Desquesnes, Zakaria Lakhdari, and Olivier Lzoray. Nonlocal infinity laplacian equation on graphs with applications in image processing and machine learning. *Mathematics and Computers in Simulation*, 102:153 – 163, 2014. 4th International Conference on Approximation Methods and Numerical Modeling in Environment and Natural Resources - {PART} {II}.

[GEL11] Mahmoud Ghoniem, Abderrahim Elmoataz, and Olivier Lezoray. Discrete infinity harmonic functions: Towards a unified interpolation framework on graphs. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 1361–1364, Sept 2011.

[KRSS15] Rasmus Kyng, Anup Rao, Sushant Sachdeva, and Daniel A. Spielman. Algorithms for lipschitz learning on graphs. *CoRR*, abs/1505.00290, 2015.

[LLP$^+$99] Andrew J. Lazarus, Daniel E. Loeb, James G. Propp, Walter R. Stromquist, and Daniel H. Ullman. Combinatorial games under auction play. *Games and Economic Behavior*, 27(2):229 – 264, 1999.

[Obe05] Adam M. Oberman. A convergent difference scheme for the infinity laplacian: construction of absolutely minimizing lipschitz extensions. *Math. Comput.*, 74(251):1217–1230, 2005.

[PSSW08] Yuval Peres, Oded Schramm, Scott Sheffield, and David B. Wilson. Tug-of-war and the infinity laplacian. 2008.