Math. Program., Ser. B (2019) 174:553–573
https://doi.org/10.1007/s10107-017-1214-8

FULL LENGTH PAPER

Variational perspective on local graph clustering

Kimon Fountoulakis[1] · **Farbod Roosta-Khorasani**[1] · **Julian Shun**[2] · **Xiang Cheng**[2] · **Michael W. Mahoney**[1]

Received: 15 March 2017 / Accepted: 24 November 2017 / Published online: 2 December 2017

Abstract Modern graph clustering applications require the analysis of large graphs and this can be computationally expensive. In this regard, local spectral graph clustering methods aim to identify well-connected clusters around a given "seed set" of reference nodes without accessing the entire graph. The celebrated Approximate Personalized PageRank (APPR) algorithm in the seminal paper by Andersen et al. (in: FOCS '06 proceedings of the 47th annual IEEE symposium on foundations of computer science, pp 475–486, 2006) is one such method. APPR was introduced and motivated purely from an algorithmic perspective. In other words, there is no a priori notion of objective function/optimality conditions that characterizes the steps taken by APPR. Here, we derive a novel variational formulation which makes explicit the actual optimization problem solved by APPR. In doing so, we draw connections between

A preliminary version of this work appeared with the title "Exploiting Optimization for Local Graph Clustering" as a technical report [9].

✉ Kimon Fountoulakis
kfount@berkeley.edu

Farbod Roosta-Khorasani
farbod@stat.berkeley.edu

Julian Shun
jshun@eecs.berkeley.edu

Xiang Cheng
x.cheng@berkeley.edu

Michael W. Mahoney
mmahoney@stat.berkeley.edu

1 Department of Statistics, UC Berkeley, Berkeley, CA 94720, USA

2 Department of Electrical Engineering and Computer Science, UC Berkeley, Berkeley, CA 94720, USA

the local spectral algorithm of Andersen et al. (2006) and an iterative shrinkage-thresholding algorithm (ISTA). In particular, we show that, appropriately initialized ISTA applied to our variational formulation can recover the sought-after local cluster in a time that only depends on the number of non-zeros of the optimal solution instead of the entire graph. In the process, we show that an optimization algorithm which apparently requires accessing the entire graph, can be made to behave in a completely local manner by accessing only a small number of nodes. This viewpoint builds a bridge across two seemingly disjoint fields of graph processing and numerical optimization, and it allows one to leverage well-studied, numerically robust, and efficient optimization algorithms for processing today's large graphs.

Keywords Local spectral graph clustering · Variational formulation · Approximate Personalized PageRank · Iterative shrinkage-thresholding

Mathematics Subject Classification 05C85 · 90C35 · 65K10

1 Introduction

Modern graph clustering applications require the analysis of large graphs [14,17]. However, in many cases, large sizes of recent graph data have rendered the applications of classical "global" approaches, i.e., those that require access to the entire graph, e.g., [3,12,13,16,22], rather impractical. The requirement to access the entire graph is indeed very undesirable. This is so since, the running time of these global algorithms typically increases with the size of the entire graph. This computational challenge sparked the development of more recent methods [1,2,15,19,23,25] that are *local* and only require access to a small portion of the graph. More specifically, given a "target" cluster, such local methods find a "nearby" cluster that sufficiently overlaps with the target and also has certain similar mathematical properties. Unlike global methods, the running time of these local alternatives depends only on the size of the output cluster or on the size of an input seed set of reference nodes, both of which can be significantly smaller than the entire graph. This property makes local graph clustering methods more applicable for today's large-scale graphs. In addition, many real-world graphs tend to have "good" small/medium size local clusters, as opposed to "good" large ones [14,17], making the application of such local algorithms even more appealing in practice.[1]

Approximate Personalized PageRank (APPR) algorithm, first introduced in the seminal paper [1], has been the cornerstone of local spectral graph clustering algorithms. APPR is a semi-supervised approximation algorithm for finding local partitions in a graph, and it does so by approximately solving the PageRank linear system, followed by rounding the approximate solution (see Sect. 3 for more details). Heuristic modifications of APPR have also been proposed which have successfully aimed at

[1] In between global and local algorithms, there is a class of *locally-biased algorithms*, e.g., [18], whose running time depends on the entire graph, however, the solution is locally-biased toward some input seed set of reference nodes. We don't consider them in this paper.

improving its performance, e.g., those that use different rules to update the iterates and/or to terminate iterations [11]. However, APPR was introduced and motivated purely from an algorithmic perspective. As a result, its output is solely determined by the operations of the algorithm applied to the data. In other words, there is no a priori notion of objective function/optimality conditions that characterizes the steps taken by APPR. As a result, it is often difficult to precisely quantify how such heuristic modifications affect the theoretical guarantees and the running time of APPR. *Our main objective here is to bridge this gap between APPR's theory and its heuristic modifications*. We do this by finding the *explicit variational formulation* of the local graph clustering problem, which is only implicitly considered in APPR. This viewpoint indeed decouples the combinatorial properties of the graph from the characteristics of the optimization algorithm used to solve the new formulation. More importantly, we will demonstrate that by using a popular optimization algorithm, namely iterative shrinkage-thresholding algorithm (ISTA), [24], and with proper initialization, one can indeed guarantee similar local properties as those of APPR. The "big-picture" objective of this work is to build a bridge between two seemingly disjoint fields of graph processing and numerical optimization. It is hoped that once this viewpoint is extended to other graph processing problems, faster and more efficient algorithms emerge as a result.

In light of the aforementioned goals, our contributions can be summarized as follows. In comparison to APPR in which the properties of the local/sparse solutions and those of the employed algorithms are tightly coupled, we propose a variational formulation in the form of ℓ_1-regularized PageRank (PR) that decouples the locality/sparsity of the solution from properties of the algorithm. In other words, if there exists a local solution for the original clustering problem, then any optimization algorithm applied to the proposed variational formulation outputs the same local solution. We then make explicit why the optimality conditions of the proposed ℓ_1-regularized PageRank problem imply the special termination criterion of APPR, and thus its solution provides the same combinatorial guarantees as in [1].

Although any optimization method applied to our proposed formulation naturally produces the same output, what differentiates between them is their running time. As a result, we present an algorithm based on iterative shrinkage-thresholding algorithm (ISTA) [4] that solves the ℓ_1-regularized PR problem, while maintaining a running time in the order of the volume of nodes/non-zeros in the optimal solution (i.e., *independent* of the size of the graph). We show that the considered algorithm only requires access to the graph in a localized manner, and hence enjoys similar locality properties as the original APPR.

Finally, by taking advantage of the local nature of iterations, we carefully implement the proposed algorithm in C++ and illustrate a few numerical experiments on several large-scale real graphs.

The rest of this paper is organized as follows. Notation used throughout the paper is introduced in Sect. 2. Section 3 provides a brief introduction to APPR and, in doing so, motivates our intentions in this paper. Our variational formulation is derived in Sect. 4. The application of ISTA for solving this variational formulation is considered in Sect. 5. This is then followed by numerical simulations on a few real graph data in Sect. 6. Conclusions and further thoughts are gathered in Sect. 7.

2 Notation and assumptions

Throughout the paper, vectors are denoted by bold lowercase letters, e.g., $\mathbf{q}$, and matrices are denoted by regular upper case letters, e.g., A. The ith coordinate of a vector $\mathbf{q}$ is denoted by $\mathbf{q}(i)$ or $[\mathbf{q}]_i$, depending on which is less cumbersome in a given formula. Iteration counter is denoted by k and is placed as subscripts, e.g., $\mathbf{q}_k$ denotes the vector corresponding to k^{th} iteration.The dot-product between two vectors is denoted by $\langle \mathbf{p}, \mathbf{q} \rangle = \mathbf{p}^T \mathbf{q}$. The vector of all ones and the vector whose i^{th} coordinate is one and zero elsewhere are denoted by $\mathbf{e}$ and $\mathbf{e}_i$, respectively. The square root of a vector is taken component-wise, i.e., $\mathbf{q}^{1/2} := [\mathbf{q}(1)^{1/2}, \ldots, \mathbf{q}(n)^{1/2}]$.

We assume that we are given an undirected graph $\mathscr{G}$ with no self-loops, whose number of nodes and edges are denoted by n and m, respectively.

The set of nodes of the graph is denoted by $\mathscr{V}$. By $j \sim i$ we mean that j is a neighbor of i and vice-versa. For a set of nodes S, the relation $j \sim S$ indicates that a node j is a neighbor of at least one node in S, $\mathrm{vol}(S) := \sum_{i \in S} d_i$ and d_i is the number of edges of node i, i.e., the degree of node i. We reserve $\mathbf{d}$ to be the vector whose components are degrees of the nodes, i.e., $\mathbf{d}(i) = d_i$. Matrices A and D denote, respectively, the adjacency matrix and the diagonal degree matrix of $\mathscr{G}$. Recall that the i^{th} diagonal element of D is given by d_i. For

$$Q := D^{-1/2} \left\{ D - \frac{1-\alpha}{2}(D + A) \right\} D^{-1/2},$$

we define

$$f(\mathbf{q}) := \frac{1}{2}\langle \mathbf{q}, Q\mathbf{q} \rangle - \alpha \langle \mathbf{s}, D^{-1/2}\mathbf{q} \rangle, \tag{1}$$

where $\mathbf{s}$ is a given distribution over the nodes also known as teleportation distribution, and α is a positive constant. For $S \subseteq [n]$ where $[n] = \{1, 2, \ldots, n\}$, let $I_S \in \mathbb{R}^{n \times |S|}$ be a $\mathbb{R}^{n \times |S|}$ matrix whose columns are taken from those of the $\mathbb{R}^{n \times n}$ identity matrix indexed by S. Further, we define $\nabla_S f(\mathbf{q}) := I_S^T \nabla f(\mathbf{q})$, $Q_S := I_S^T Q I_S$, and $\mathbf{d}_S := \mathrm{diag}(I_S^T D I_S)$, where "diag$(\cdot)$" extracts the diagonal of the input matrix and returns it as a vector. We also define the support set of a vector $\mathbf{q}$ as the index set of its non-zero elements, i.e., $\mathrm{supp}(\mathbf{q}) := \{i \in [n] \mid \mathbf{q}(i) \neq 0\}$. One can easily see that function ∇f is 1-Lipschitz continuous w.r.t. ℓ_2 norm, that is, the largest eigenvalue of Q is smaller or equal to 1. To prove this note that $Q = \alpha I + \frac{1-\alpha}{2}\mathcal{L}$, where $\mathcal{L} = I - D^{-1/2} A D^{-1/2}$ is the symmetric normalized Laplacian matrix. Using the fact that the largest eigenvalue of $\mathcal{L}$ is bounded by 2 and the latter definition of Q we obtain the result. Furthermore, note that this condition implies that $\forall \mathbf{p}, \mathbf{q} \in \mathbb{R}^n$

$$\|\nabla f(\mathbf{p}) - \nabla f(\mathbf{q})\|_2 \leq \|\mathbf{p} - \mathbf{q}\|_2,$$

which also implies

$$f(\mathbf{p}) \leq f(\mathbf{q}) + \langle \nabla f(\mathbf{q}), \mathbf{p} - \mathbf{q} \rangle + \frac{1}{2}\|\mathbf{p} - \mathbf{q}\|_2^2.$$

3 Background and motivation

Suppose n denotes the total number of nodes. A simplified version of PageRank (PR) algorithm [20] amounts to computing the stationary solution of

$$\mathbf{p}_{k+1}(j) = \sum_{i \sim j} \mathbf{p}_k(i)/d_i,$$

where each node is modeled as a node of a graph, and the components of the vector $\mathbf{p} \in \mathbb{R}^n$ represent the "popularity" of these n nodes. Usually the "popularity" is encoded as a probability mass distributed over all the nodes, i.e., the vector $\mathbf{p}$ is like a probability mass function where $\mathbf{p} \geq \mathbf{0}$ and $\mathbf{e}^T\mathbf{p} = 1$. As a result, operationally, the simplified PR algorithm iteratively transfers probability mass around the graph by adding to a node's assigned probability and taking the equivalent amount from its neighbors. The stationary vector corresponding to this iterative operation is the degrees vector $\mathbf{d}$. In Linear Algebra's jargon, the above simplified version of the PR algorithm amounts to the computation of the principal eigenvector of a large and sparse matrix, AD^{-1}, often referred to as transition matrix, i.e.,

$$AD^{-1}\mathbf{p} = \mathbf{p}.$$

This simplified version of the PR algorithm has several disadvantages. A particular issue arise when some node is isolated and lacks edges to other nodes, in which case, the above procedure is not well-defined, i.e., the node's degree is zero. This type of nodes are often referred to as "dangling nodes" and an elegant way to handle such situations was proposed in [8]. As a result, for simplicity's sake, we assume that the dangling nodes are dealt with in a proper way and hence, $d_i > 0, \forall i \in [n]$.

The second disadvantage is that the convergence to the principal eigenvector of AD^{-1} requires the transition matrix to be aperiodic and irreducible, i.e., the smallest eigenvalue of AD^{-1} is in absolute value less than 1, and matrix $(AD^{-1})^t$ is component-wise positive for some t. The former issue can be resolved by considering the lazy random walk matrix, $W = (I + AD^{-1})/2$ instead of AD^{-1}, while for the latter, one can consider a convex combination of the form

$$\alpha \mathbf{s}\mathbf{e}^T + (1 - \alpha)W, \tag{2}$$

where $\alpha \in (0, 1)$ is the "teleportation" parameter and $\mathbf{s}$ is a given distribution over the nodes also known as teleportation distribution. The principal eigenvector of matrix (2) is known as the PR vector [20]. The celebrated PageRank (PR) vector was initially developed in [20] to rank websites/nodes according to their "popularity".

Initially, $\mathbf{s}$ was set to have uniform probability distribution over all the nodes. However, "personalized" distributions became popular [10] which assign non-uniform probability mass in favor of certain nodes and, as a result, one seeks to obtain personalized principal eigenvectors of matrix (2). For example, after arbitrarily ordering the nodes of $\mathcal{G}$, consider an input node, say i, and a vector $\mathbf{s} \in \mathbb{R}^n$ such that $\mathbf{s}(i) = 1$ and zero elsewhere. For a lazy random walk matrix, $W = (I + AD^{-1})/2$, finding the

principal eigenvector of (2) which also satisfies $\mathbf{e}^T\mathbf{p} = 1$ and $\mathbf{p} \geq \mathbf{0}$, is equivalent to the solution of the linear system

$$\mathbf{p} = \alpha\mathbf{s} + (1-\alpha)W\mathbf{p}. \tag{3}$$

This approach is known as Personalized PageRank (PPR), and in fact, has become the ubiquitous tool for ranking web pages, social and information network analysis, recommendation systems, analysis of biology, neuroscience and physics networks; see [10] for an excellent review of PR and PPR as well as their applications.

Approximate Personalized PageRank (APPR), was first introduced in the seminal work of [1]. As it appears from its name, APPR is an approximate version of PPR which boils down to approximately solving the linear system (3) using a particular iterative scheme and a specifically chosen early stopping criterion. In fact, it can be shown that APPR's original algorithm is, indeed, an iterative coordinate solver for the linear system (3). To see this, let us first define the residual vector as $\mathbf{r} := (I - (1-\alpha)W)\mathbf{p} - \alpha\mathbf{s}$. An iterative coordinate solver applied to (3) updates the current approximate solution at iteration k according to $\mathbf{p}_{k+1} = \mathbf{p}_k - \mathbf{r}_k(i)\mathbf{e}_i$. As a result, the residual vector has the following recursive representation

$$\mathbf{r}_{k+1} = \mathbf{r}_k - \mathbf{r}_k(i)\mathbf{e}_i + \frac{1-\alpha}{2}(I + AD^{-1})\mathbf{r}_k(i)\mathbf{e}_i. \tag{4}$$

Algorithm 1 gives an overview of such iterative coordinate solver with a particular stopping criterion. From the definitions of D and A, it can easily be seen that Steps 5, 6, and 7 practically implement the recursive relation (4).

Algorithm 1 Coordinate solver (APPR) for (3)

1: **Initialize**: $\rho > 0$, $\mathbf{p}_0 = \mathbf{0}$, thus $\mathbf{r}_0 = -\alpha\mathbf{s}$
2: **while** $\|D^{-1}\mathbf{r}_k\|_\infty > \rho\alpha$ **do**
3: Choose an i such that $\mathbf{r}_k(i) < -\alpha d_i \rho$
4: $\mathbf{p}_{k+1}(i) = \mathbf{p}_k(i) - \mathbf{r}_k(i)$
5: $\mathbf{r}_{k+1}(i) = \frac{1-\alpha}{2}\mathbf{r}_k(i)$
6: For each j such that $j \sim i$ set

$$\mathbf{r}_{k+1}(j) = \mathbf{r}_k(j) + \frac{1-\alpha}{2d_i}A_{ij}\mathbf{r}_k(i)$$

7: For each j such that $j \nsim i$ set $\mathbf{r}_{k+1}(j) = \mathbf{r}_k(j)$
8: $k = k+1$
9: **end while**
10: **return** $\mathbf{p}_k$

Now, by defining $\tilde{\mathbf{r}}_k := -(1/\alpha)\mathbf{r}_k$ and replacing $\mathbf{r}_k$ with $\tilde{\mathbf{r}}_k$ in Algorithm 1 we obtain APPR algorithm in exactly the same form as described in [1, Section 3]. This indeed shows that APPR is an iterative coordinate solver for the PPR linear system (3).

It is, in fact, easy to see that Algorithm 1 solves the optimization problem "min $f(\mathbf{q})$", where f is defined as in (1). To see this, note that the residual in

Algorithm 2 Coordinate descent solver for "min $f(\mathbf{q})$"

Initialize: $\rho > 0$, $\mathbf{q}_0 = \mathbf{0}$, thus $\nabla f(\mathbf{q}_0) = -\alpha D^{-1/2}\mathbf{s}$
while $\|D^{-1/2}\nabla f(\mathbf{q}_k)\|_\infty > \rho\alpha$ **do**
  Choose an i such that $\nabla_i f(\mathbf{q}_k) < -\alpha\rho d_i^{1/2}$
  $\mathbf{q}_{k+1}(i) = \mathbf{q}_k(i) - \nabla_i f(\mathbf{q}_k)$
  $\nabla_i f(\mathbf{q}_{k+1}) = \frac{1-\alpha}{2}\nabla_i f(\mathbf{q}_k)$
  For each j such that $j \sim i$ set

$$\nabla_j f(\mathbf{q}_{k+1}) = \nabla_j f(\mathbf{q}_k) + \frac{(1-\alpha)}{2d_i^{1/2}d_j^{1/2}} A_{ij} \nabla_i f(\mathbf{q}_k)$$

  For each j that $j \nsim i$ set $\nabla_j f(\mathbf{q}_{k+1}) = \nabla_j f(\mathbf{q}_k)$
  $k = k + 1$
end while
return $\mathbf{p}_k := D^{1/2}\mathbf{q}_k$

Algorithm 1 can be written in terms of the scaled gradient of function f. In particular, since

$$\nabla f(\mathbf{q}) = D^{-1/2}\left\{D - \frac{1-\alpha}{2}(D+A)\right\}D^{-1/2}\mathbf{q} - \alpha D^{-1/2}\mathbf{s},$$

we have $D^{1/2}\nabla f(\mathbf{q}) = \mathbf{r}$, where $\mathbf{q} := D^{-1/2}\mathbf{p}$. Using $D^{1/2}\nabla f(\mathbf{q}) = \mathbf{r}$ we can rewrite Algorithm 1 as a coordinate descent method for minimizing f as in Algorithm 2.

The above simple observation is a motivating factor behind our objective of deriving the exact variational formulation of APPR. However, before delving into the details of this derivation, let us briefly review the combinatorial guarantees of APPR, with respect to graph clustering. This is indeed important in light of our new variational formulation and the proposed algorithm for solving it. In particular, we will show that the optimality condition corresponding to this variational formulation, in fact, implies the special termination criterion of APPR, and hence, the proposed algorithm, upon termination, recovers a cluster with the same combinatorial guarantees as the solution of APPR.

Conductance is a widely used concept in graph clustering to measure the quality of a cluster. Loosely speaking, conductance of a cluster is defined as the ratio of its external over internal connectivities. Lower conductance translates to a better cluster since it implies the cluster is better connected internally than externally. More specifically, let w_{ij} be the weight of the edge between two neighbor nodes $i \sim j$. We define the conductance of a subset of nodes $S \subset \mathscr{V}$ as

$$\Phi(S) := \frac{\sum_{i\in S}\sum_{j\in\mathscr{V}\setminus S, j\sim i} w_{ij}}{\min\left(\mathrm{vol}(S), \mathrm{vol}(\mathscr{V}\setminus S)\right)}$$

and the minimum-conductance of a given graph $\mathscr{G}$ as

$$\Phi(\mathscr{G}) := \min_{S\subset\mathscr{V}} \Phi(S). \tag{5}$$

Given a target cluster C with conductance $\Phi(C) \leq \Omega(\varphi^2/\log m)$ and α set properly according to φ, a particular rounding algorithm is applied to the output of APPR which determines a set of nodes in the graph with conductance of at most φ. More precisely, let $\mathbf{p}_k$ be the output of APPR with input value α and let $\mathbf{r}_k$ be the residual of (3). According to [1, Theorem 5], the output of APPR can be used as an input to a rounding procedure (see [1, Section 2.2]) to produce clusters of low-conductance. The rounding procedure sorts the indices in $\text{supp}(\mathbf{p}_k)$ in decreasing order according to the values of the components of $D^{-1}\mathbf{p}_k$. Let $i_1, i_2, \ldots, i_{|H_k|}$ be the sorted indices, where $H_k = \text{supp}(\mathbf{p}_k)$. Using the sorted indices, the rounding procedure generates a collection of sets $\mathscr{S}_j := \{i_1, i_2, \ldots, i_j\}$ for each $j \in \{1, 2, \ldots, |H_k|\}$. Provided that there exists a subset of nodes, C, such that $\Phi(C) \leq \alpha/10$, $\text{vol}(C) \leq 2\text{vol}(\mathscr{G})/3$, $\mathbf{s}$ is initialized within nodes in C_α, where $C_\alpha \subseteq C$ satisfies $\text{vol}(C_\alpha) \geq \text{vol}(C)/2$, and $\rho = 1/(10\text{vol}(C))$ then [1, Theorem 5] implies that

$$\min_{j\in\{1,2,\ldots,|H_k|\}} \Phi(\mathscr{S}_j) \leq \sqrt{135\log(m)\alpha}.$$

This result is a local analogue of the Cheeger inequality [5] for PageRank vectors.

An undesirable side-effect of this rounding procedure is the lack of a lower bound on the volume of the output cluster. This, in particular, implies that it is possible to find a very small cluster. As a remedy, Andersen et al. [1, Section 6] introduces PageRank-Nibble procedure. Let $\phi \in [0, 1]$ be a parameter and assume that there exists $C \subset \mathscr{V}$ such that $\text{vol}(C) \leq \text{vol}(\mathscr{G})/2$ and $\Phi(C) \leq \phi^2/(22500\log^2(100m))$. PageRank-Nibble makes only a single call to APPR and uses its output to produce the rounded sets as before. However, [1, Theorem 7] suggests that if APPR is initialized with $\alpha = \phi^2/(225\log(100m^{1/2}))$ and $\mathbf{s}$ is set in C_α, then there exists some $b \in [1, \lceil\log m\rceil]$ such that if $\rho \leq (2^b 48\lceil\log m\rceil)^{-1}$, at least one set $\mathscr{S}_j$ satisfies $\Phi(\mathscr{S}_j) \leq \phi$, $2^{b-1} < \text{vol}(\mathscr{S}_j) < 2\text{vol}(\mathscr{G})/3$ and $\text{vol}(\mathscr{S}_j \cap C) > 2^{b-2}$.

4 Variational formulation

In this section we set out to derive the variational formulation characterizing APPR and discuss how we can view the approximate solution of (3) as the optimal solution of an ℓ_1-regularized problem.

A key observation which helps us derive the sought-after variational formulation is given by the following lemma. In particular, Lemma 1 shows that the iterates generated by Algorithm 2 with a particular initialization, have an interesting property, in that they all satisfy $\nabla f(\mathbf{q}_k) \leq 0\ \forall k$.

Lemma 1 *If Algorithm 2 is initialized with* $\mathbf{q}_0 = 0$ *and* $\mathbf{s} \geq 0$*, then* $\mathbf{q}_{k+1} \geq \mathbf{q}_k$ *and* $\nabla f(\mathbf{q}_k) \leq 0\ \forall k$.

Proof We will prove this statement by induction. Let us assume that at the k^{th} iteration we have $\mathbf{q}_k \geq 0$ and $\nabla f(\mathbf{q}_k) \leq 0$. Further, let assume that there exists coordinate i such that $\nabla_i f(\mathbf{q}_k) < -\rho\alpha d_i^{1/2}$, otherwise, the termination criterion is satisfied. Algorithm 2 chooses one coordinate which satisfies $\nabla_i f(\mathbf{q}_k) < -\rho\alpha d_i^{1/2}$. Then from Step 4 of

Algorithm 2 we have that $\mathbf{q}_{k+1} \geq \mathbf{q}_k$. Moreover, from Steps 5, 6, and 7, it follows that $\nabla_i f(\mathbf{q}_k) < \nabla_i f(\mathbf{q}_{k+1}) < 0$, $\nabla_j f(\mathbf{q}_{k+1}) < \nabla_j f(\mathbf{q}_k) \leq 0$ for each j such that $i \sim j$ and $\nabla_j f(\mathbf{q}_{k+1}) = \nabla_j f(\mathbf{q}_k) \leq 0$ for each j such that $i \nsim j$. Hence, $\nabla f(\mathbf{q}_{k+1}) \leq 0$. Let $\mathbf{q}_0 = 0$ and $\mathbf{s} \geq 0$. Then $\nabla f(\mathbf{q}_0) = -\alpha \mathbf{s} \leq 0$. We conclude that $\mathbf{q}_{k+1} \geq \mathbf{q}_k \geq 0$ and $\nabla f(\mathbf{q}_k) \leq 0 \; \forall k$. □

On the one hand, as argued in Sect. 3, Algorithm 2 is equivalent to the coordinate descent interpretation of APPR. On the other, Algorithm 2 terminates when

$$\|D^{-1/2} \nabla f(\mathbf{q}_k)\|_\infty \leq \rho\alpha, \tag{6}$$

which, since by Lemma 1 the gradient components at every iteration are all non-positive, is equivalent to

$$\nabla_i f(\mathbf{q}_k) \geq -\rho\alpha d_i^{1/2} \; \forall i. \tag{7}$$

Interestingly, the termination criterion (7) is related to the first-order optimality conditions of the following ℓ_1-regularized problem

$$\ell_1\text{-reg. PR:} \quad \boxed{\text{minimize} \quad \psi(\mathbf{q}) := \rho\alpha \|D^{1/2}\mathbf{q}\|_1 + f(\mathbf{q}).} \tag{8}$$

Let $\mathbf{q}_*$ denote the optimal solution of (8). The first-order optimality conditions of (8) can be written as

$$\nabla_i f(\mathbf{q}_*) = \begin{cases} -\rho\alpha d_i^{1/2} & \text{if } \mathbf{q}_*(i) > 0 \\ \rho\alpha d_i^{1/2} & \text{if } \mathbf{q}_*(i) < 0 \\ \in \rho\alpha d_i^{1/2}[-1, 1] & \text{if } \mathbf{q}_*(i) = 0. \end{cases} \tag{9}$$

Theorem 1, below, shows that the solution of (8) has the property that $\mathbf{q}_* \geq 0$. Therefore, the optimality conditions of problem (8) are equivalent to

$$\nabla_i f(\mathbf{q}_*) = \begin{cases} -\rho\alpha d_i^{1/2} & \text{if } \mathbf{q}_*(i) > 0 \\ \in \rho\alpha d_i^{1/2}[-1, 0] & \text{if } \mathbf{q}_*(i) = 0. \end{cases} \tag{10}$$

The formulation (8) is indeed a variational characterization of the APPR procedure as described by its coordinate descent representation in Algorithm 2. However, notice that the optimality conditions (10) imply the termination criterion (7) of APPR, but the converse is not necessarily true. This is because (7) does not distinguish between positive and zero components of $\mathbf{q}_*$. Moreover, depending on which coordinate is chosen at every iteration, APPR can yield a different output on multiple runs. In other words, the output solution depends completely on the setting of the algorithm. In contrast, ℓ_1-regularized PR formulation (8) decouples the locality/sparsity of the solution from properties of the algorithm, i.e., which nodes are chosen at every iteration. More specifically, if there exists a good local cluster, then *any optimization algorithm applied to ℓ_1-regularized PR obtains the same solution, and the differences merely boil down*

to running time and locality as opposed to the actual output solution. Note that in practice algorithms solve approximately the ℓ_1-regularized PR, therefore, small differences might exist among solutions of different algorithms. However, the longer that any convergent algorithm is run the closer its solution will be to the optimal solution of the ℓ_1-regularized PR problem.

The proposed optimization formulation (8) is motivated by [11, Theorem 3]. However, by drawing a clear connection between the termination criterion of APPR, (7), and the first-order optimality conditions of ℓ_1-regularized PR, (10), we get a much simpler formulation than the one presented in [11]. In particular, unlike the formulation of [11], problem (8) does not require any additional tuning parameters other than the ones used for APPR, nor does it introduce any constraints, such as non-negativity. More importantly, the formulation in [11] only implies the sparsity of the final solution as opposed to the intermediate iterates produced by any iterative procedure applied to solve the corresponding optimization problem. In sharp contrast, in Sect. 5, we will show that the application of properly initialized ISTA to our formulation (8) maintains sparsity for all generated iterates, a property which is crucial to obtaining a local algorithm.

5 Algorithm

As mentioned before, an advantage of the variational formulation (8) is that it decouples the properties of the obtained solution from the applied algorithm. This allows for application of any optimization algorithm. However, among all options, we need to find methods that, like APPR, enjoy *locality* properties, in that they only require access to small portion of the graph. In doing so, in this section, we investigate the application of ISTA for solving (8) and study its theoretical properties such as locality and running time. The adaptation of ISTA to our particular problem is depicted in Algorithm 3.

The main computational advantage of APPR is that, APPR never requires access to the entire graph and iterations are performed efficiently which makes the application of APPR very appealing for modern large graphs. Interestingly, we now show that Algorithm 3, which incorporates a presumably global optimization routine such as ISTA, exhibits this desired locality property while inheriting the fast convergence properties of ISTA.

Theorem 1 shows the equivalence between Algorithm 3 and ISTA, and more importantly, establishes the desired locality property. In particular, part (iii) of Theorem 1 states that if Algorithm 3 is initialized properly, then despite the fact that the set S_k changes at every iteration (Step 3 of Algorithm 3), its size, $|S_k|$, indeed *never grows larger* than the total number of non-zeros of the optimal solution. As such, in the worst case where one might update all the coordinates in S_k at every iteration, the per-iteration cost depends only on the sparsity of the final solution vector, as opposed to the size of the full graph.

Theorem 1 *Let* $\mathbf{q}_*$ *be the optimal solution of* (8) *and consider* $\rho > 0$ *and a vector* $\mathbf{s} \geq 0$ *such that* $\langle \mathbf{e}, \mathbf{s} \rangle = 1$ *and* $\|\mathbf{s}\|_\infty \geq \rho$. *Algorithm 3 has the following properties.*

(i) *Algorithm 3 is equivalent to ISTA in [4],*

Algorithm 3 ISTA-equivalent solver for (8)

1: **Initialize**: $\epsilon \in (0,1), \alpha > 0, \mathbf{q}_0 = 0, \rho > 0, \mathbf{s}$ such that $\langle \mathbf{e}, \mathbf{s} \rangle = 1$ and $\mathbf{s} \geq \mathbf{0}$, set $\nabla f(\mathbf{q}_0) = -\alpha D^{-1/2}\mathbf{s}$.

2: **while** $\|D^{-1/2}\nabla f(\mathbf{q}_k)\|_\infty > (1+\epsilon)\rho\alpha$ **do**

3: Set $S_k := \{i \in [n] \mid \mathbf{q}_k(i) - \nabla_i f(\mathbf{q}_k) \geq \rho\alpha d_i^{1/2}\}$

4: $\Delta\mathbf{q}_k := -(\nabla_{S_k} f(\mathbf{q}_k) + \rho\alpha \mathbf{d}_{S_k}^{1/2})$ and $\mathbf{q}_{k+1}(S_k) = \mathbf{q}_k(S_k) + \Delta\mathbf{q}_k$

5: For each $i \in S_k$ set

$$\nabla_i f(\mathbf{q}_{k+1}) = -\rho\alpha d_i^{1/2} - \frac{1-\alpha}{2}[I_{S_k}\Delta\mathbf{q}_k]_i - \frac{1-\alpha}{2d_i^{1/2}} \sum_{l\sim i, l\in S_k} \frac{A_{il}[I_{S_k}\Delta\mathbf{q}_k]_l}{d_l^{1/2}}$$

6: For each $j \notin S_k$ such that $j \sim S_k$ set

$$\nabla_j f(\mathbf{q}_{k+1}) = \nabla_j f(\mathbf{q}_k) - \frac{1-\alpha}{2d_j^{1/2}} \sum_{l\sim j, l\in S_k} \frac{A_{jl}[I_{S_k}\Delta\mathbf{q}_k]_l}{d_l^{1/2}}$$

7: For each $j \notin S_k$ such that $j \nsim S_k$ set

$$\nabla_j f(\mathbf{q}_{k+1}) = \nabla_j f(\mathbf{q}_k)$$

8: $k = k+1$

9: **end while**

10: **return** $\mathbf{p}_k := D^{1/2}\mathbf{q}_k$

(ii) $S_k \subseteq S_{k+1} \subseteq supp(\mathbf{q}_*)\ \forall k$,
(iii) $|S_k| \leq |S_{k+1}| \leq |supp(\mathbf{q}_*)|, \forall k$,
(iv) $0 \leq \mathbf{q}_k \leq \mathbf{q}_{k+1}, \forall k$, *which implies that* $\mathbf{q}_* \geq 0$, *since* $\mathbf{q}_k \to \mathbf{q}_*$ *as* $k \to \infty$.
(v) $\nabla f(\mathbf{q}_k) \leq 0$, *and moreover* $\nabla_i f(\mathbf{q}_k) \leq -\rho\alpha d_i^{1/2}\ \forall i \in S_k$ *and* $\nabla_i f(\mathbf{q}_k) > -\rho\alpha d_i^{1/2}\ \forall i \in [n]\backslash S_k\ \forall k$.

Proof Define

$$\tilde{f}(\mathbf{q}; \mathbf{q}_k) := f(\mathbf{q}_k) + \langle \mathbf{q} - \mathbf{q}_k, \nabla f(\mathbf{q}_k)\rangle + \frac{1}{2}\|\mathbf{q} - \mathbf{q}_k\|_2^2,$$
$$\tilde{\psi}(\mathbf{q}; \mathbf{q}_k) := \rho\alpha\|D^{1/2}\mathbf{q}\|_1 + \tilde{f}(\mathbf{q}; \mathbf{q}_k).$$

It is easy to see that

$$\arg\min_{\mathbf{q}} \tilde{\psi}(\mathbf{q}; \mathbf{q}_k) = \arg\min_{\mathbf{q}} \rho\alpha\|D^{1/2}\mathbf{q}\|_1 + \frac{1}{2}\|\mathbf{q} - (\mathbf{q}_k - \nabla f(\mathbf{q}_k))\|_2^2,$$

and hence

$$\mathbf{q}(i) = \mathbf{prox}_{\rho\alpha d_i^{1/2}\|.\|_1}(\mathbf{q}_k(i) - \nabla_i f(\mathbf{q}_k)),$$

where **prox** is the proximal operator [21]. Now let us define the sets

$$\begin{aligned} S_k &:= \{i \in [n] \mid \mathbf{q}_k(i) - \nabla_i f(\mathbf{q}_k) \geq \rho\alpha d_i^{1/2}\}, \\ \widehat{S}_k &:= \{i \in [n] \mid -\rho\alpha d_i^{1/2} < \mathbf{q}_k(i) - \nabla_i f(\mathbf{q}_k) < \rho\alpha d_i^{1/2}\}, \\ \widetilde{S}_k &:= \{i \in [n] \mid \mathbf{q}_k(i) - \nabla_i f(\mathbf{q}_k) \leq -\rho\alpha d_i^{1/2}\}. \end{aligned} \tag{11}$$

For convenience, below, we rewrite ISTA from [4]. To show that Algorithms 3

Algorithm 4 ISTA for (8)

1: **Initialize**: $\rho > 0$, $\mathbf{q}_0 = 0$, thus $\nabla f(\mathbf{q}_0) = -\alpha D^{-1/2}\mathbf{s}$
2: **while** termination criteria are not satisfied **do**
3: $\mathbf{q}_{k+1}(i) = \mathbf{prox}_{\rho\alpha d_i^{1/2}\|.\|_1}(\mathbf{q}_k(i) - \nabla_i f(\mathbf{q}_k))$, $\forall i$, whose closed-form solution is given by

$$\mathbf{q}_{k+1}(i) = \begin{cases} \mathbf{q}_k(i) - (\nabla_i f(\mathbf{q}_k) + \rho\alpha d_i^{1/2}) & \text{if } i \in S_k \\ \mathbf{q}_k(i) - (\nabla_i f(\mathbf{q}_k) - \rho\alpha d_i^{1/2}) & \text{if } i \in \widetilde{S}_k \\ 0 & \text{if } i \in \widehat{S}_k. \end{cases}$$

4: Calculate new gradient $\nabla f(\mathbf{q}_{k+1})$.
5: $k = k + 1$
6: **end while**
7: **return** $\mathbf{p}_k := D^{1/2}\mathbf{q}_k$

and 4 are equivalent, it suffices to show that $\widetilde{S}_k = \emptyset, \forall k$. We will prove the result by induction. Let us assume that at iteration k we have a $\mathbf{q}_k \geq 0$, $\nabla f(\mathbf{q}_k) \leq 0$ and $\nabla_i f(\mathbf{q}_k) \leq -\rho\alpha d_i^{1/2}\ \forall i \in S_k$. As a result of the first two assumptions, we have $\widetilde{S}_k = \emptyset$ and $S_k \cup \widehat{S}_k = [n]$. Hence, Step 3 of ISTA Algorithm 4 can be simplified as

$$\mathbf{q}_{k+1}(i) = \begin{cases} \mathbf{q}_k(i) - (\nabla_i f(\mathbf{q}_k) + \rho\alpha d_i^{1/2}) & \text{if } i \in S_k \\ 0 & \text{if } i \in \widehat{S}_k \end{cases}. \tag{12}$$

Define $\Delta\mathbf{q}_k := -I_{S_k}^T\left(\nabla f(\mathbf{q}_k) + \rho\alpha D^{1/2}\mathbf{e}\right)$, where I_{S_k} is defined in Sect. 2. Consequently, at iteration k, the new gradient components are updated as follows

$$\nabla_i f(\mathbf{q}_{k+1}) = \begin{cases} -\rho\alpha d_i^{1/2} - \frac{1-\alpha}{2}[I_{S_k}\Delta\mathbf{q}_k]_i - \frac{1-\alpha}{2d_i^{1/2}}\sum_{l\sim i, l\in S_k}\frac{A_{il}[I_{S_k}\Delta\mathbf{q}_k]_l}{d_l^{1/2}}, & i \in S_k \\ \nabla_i f(\mathbf{q}_k) - \frac{1-\alpha}{2d_i^{1/2}}\sum_{l\sim i, l\in S_k}\frac{A_{il}[I_{S_k}\Delta\mathbf{q}_k]_l}{d_l^{1/2}}, & i \in \widehat{S}_k \text{ and } i \sim S_k \\ \nabla_i f(\mathbf{q}_k), & i \in \widehat{S}_k \text{ and } i \nsim S_k, \end{cases} \tag{13}$$

where A is the adjacency matrix of the given graph. Equation (13) is obtained by using $\nabla f(\mathbf{q}_{k+1}) = \nabla f(\mathbf{q}_k) - I_{S_k}\Delta\mathbf{q}_k - \frac{1-\alpha}{2}I_{S_k}\Delta\mathbf{q}_k - \frac{1-\alpha}{2}D^{-1/2}AD^{-1/2}I_{S_k}\Delta\mathbf{q}_k$ and the definition of $\Delta\mathbf{q}_k$. By induction hypothesis and noticing that $\Delta\mathbf{q}_k \geq 0$ and $A_{i,l} \geq 0, \forall i, l$, it is easy to see that by (12), we have $\mathbf{q}_{k+1} \geq 0$, and by (13), we get

$\nabla f(\mathbf{q}_{k+1}) \leq 0$. Hence, it follows that $\widetilde{S}_{k+1} = \emptyset$. In addition, for any $i \in S_k$, we get $\nabla_i f(\mathbf{q}_{k+1}) \leq -\rho\alpha d_i^{1/2}$ and, as such, $i \in S_{k+1}$. In other words, once an index i enters the set S_k at iteration k, it will continue to stay in that set for all subsequent iterations, and so we always have $\mathbf{q}_{k+1}(i) \geq \mathbf{q}_k(i)$. As a result we obtain $S_k \subseteq S_{k+1}$ and $|S_k| \leq |S_{k+1}|$. The only indices entering S_{k+1} are those from $\widehat{S}_k$ that are also neighbors of S_k. To prove this use that $\widetilde{S}_k = \emptyset\ \forall k$, therefore the only coordinates that can enter in S_k come from $\widehat{S}_k$. In addition from (12) we have that $[\mathbf{q}_k]_i = 0\ \forall i \in \widehat{S}_k$ and from (13) we have that neighbors of S_k that are also in $\widehat{S}_k$ get their partial derivatives updated. Therefore, using the definition of S_k in (11) only the neighbors of S_k that are also in $\widehat{S}_k$ might enter S_k, since the rest of the coordinates in $i \in \widehat{S}_k$ have $[\mathbf{q}_k]_i = 0$ and also do not get their partial derivatives updated. In this case, suppose that $i \in \widehat{S}_k \cap S_{k+1}$. By (12), we have $\mathbf{q}_{k+1}(i) = 0$, which combined with the definition of S_{k+1}, yields $\nabla_i f(\mathbf{q}_{k+1}) \leq -\rho\alpha d_i^{1/2}$. As a result, we have $\nabla_i f(\mathbf{q}_{k+1}) \leq -\rho\alpha d_i^{1/2}, \forall i \in S_{k+1}$. All is left to do is to start the iterations with the proper initial conditions, so that the base case of the induction holds. Set ρ small enough that $\|\mathbf{s}\|_\infty \geq \rho$. Now since $\mathbf{s} \geq 0$, by choosing $\mathbf{q}_0 = 0$, we have that $\nabla f(\mathbf{q}_0) = -\alpha D^{-1/2}\mathbf{s} \leq 0$ and $\nabla_i f(\mathbf{q}_0) \leq -\rho\alpha d_i^{1/2}$ $\forall i \in S_0$. In addition, such a choice of $\mathbf{q}_0$, (12) as well as the decreasing nature of $\widehat{S}_k$ imply that $\mathbf{q}_{k+1} \geq \mathbf{q}_k, \forall k$. Since $\mathbf{q}_{k+1} \geq \mathbf{q}_k\ \forall k$ and $\mathbf{q}_k \to \mathbf{q}_*$ then Algorithm 3 will update only coordinates that are in $\text{supp}(\mathbf{q}_*)$. To prove this note that if a coordinate in $\mathbf{q}_k$ becomes positive it will remain positive because $\mathbf{q}_{k+1} \geq \mathbf{q}_k$. Since $\mathbf{q}_k \to \mathbf{q}_*$ it must be that only coordinates in $\text{supp}(\mathbf{q}_*)$ will become positive in $\mathbf{q}_k$ for some k. Thus, we have that $S_k \subseteq \text{supp}(\mathbf{q}_*)$ and $|S_k| \leq |\text{supp}(\mathbf{q}_*)|\ \forall k$. Finally, notice that $\nabla_i f(\mathbf{q}_k) > -\rho\alpha d_i^{1/2}\ \forall i \in [n]\backslash S_k\ \forall k$. This can be proved by using $[n]\backslash S_k = \widehat{S}_k \cup \widetilde{S}_k$, $\widetilde{S}_k = \emptyset$, $\mathbf{q}_k \geq 0\ \forall k$ and using the definition of $\widetilde{S}_k$ in (11). □

Let

$$\mathscr{S}_* := \text{supp}(\mathbf{q}_*), \tag{14}$$

be the support of the optimal solution. In the following theorem, we give an upper bound for $\text{vol}(\mathscr{S}_*)$ which is, in turn, used in Theorem 3 to derive the worst-case running time of Algorithm 3.

Theorem 2 *We have that $\text{vol}(\mathscr{S}_*) \leq \|\mathbf{s}\|_1/\rho$, where ρ is the regularization parameter of the ℓ_1-regularized PageRank (8).*

Proof From (v) in Theorem 1 we have that $\nabla_i f(\mathbf{q}_k) \leq -\rho\alpha d_i^{1/2}\ \forall i \in S_k$ for any iteration k. Multiplying both sides of the latter by $-d_i^{1/2}$ and summing over all nodes in S_k yields

$$\sum_{i \in S_k} -d_i^{1/2}\nabla_i f(\mathbf{q}_k) \geq \rho\alpha\text{vol}(S_k),$$

which implies that

$$\|D^{1/2}\nabla f(\mathbf{q}_k)\|_1 \geq \rho\alpha\text{vol}(S_k). \tag{15}$$

We will now prove that $\|D^{1/2}\nabla f(\mathbf{q}_k)\|_1$ decreases monotonically as k increases. From Step 4 of Algorithm 3, we have $\mathbf{q}_{k+1} = \mathbf{q}_k + I_{S_k}\Delta\mathbf{q}_k$. As a result, from (1), it follows that

$$\begin{aligned}\nabla f(\mathbf{q}_{k+1}) &= Q\mathbf{q}_{k+1} - \alpha D^{-1/2}\mathbf{s}\\ &= Q\mathbf{q}_k + QI_{S_k}\Delta\mathbf{q}_k - \alpha D^{-1/2}\mathbf{s}\\ &= \nabla f(\mathbf{q}_k) + QI_{S_k}\Delta\mathbf{q}_k\\ &= \nabla f(\mathbf{q}_k) + \left(\alpha I + \frac{(1-\alpha)}{2}\left(I - D^{-1/2}AD^{-1/2}\right)\right) I_{S_k}\Delta\mathbf{q}_k.\end{aligned}$$

In the last inequality we used $Q = I - \frac{1-\alpha}{2}(I + D^{-1/2}AD^{-1/2}) = I + \frac{1-\alpha}{2}I - \frac{1-\alpha}{2}I - \frac{1-\alpha}{2}(I + D^{-1/2}AD^{-1/2}) = \alpha I + \frac{(1-\alpha)}{2}\left(I - D^{-1/2}AD^{-1/2}\right)$. Hence, we get

$$D^{1/2}\nabla f(\mathbf{q}_{k+1}) = D^{1/2}\nabla f(\mathbf{q}_k) + \alpha D^{1/2}I_{S_k}\Delta\mathbf{q}_k + \frac{(1-\alpha)}{2}(D-A)D^{-1/2}I_{S_k}\Delta\mathbf{q}_k,$$

which implies

$$\begin{aligned}\mathbf{e}^T D^{1/2}\nabla f(\mathbf{q}_{k+1}) &= \mathbf{e}^T D^{1/2}\nabla f(\mathbf{q}_k) + \alpha\mathbf{e}^T D^{1/2}I_{S_k}\Delta\mathbf{q}_k\\ &\quad + \frac{(1-\alpha)}{2}\mathbf{e}^T(D-A)D^{-1/2}I_{S_k}\Delta\mathbf{q}_k\\ &= \mathbf{e}^T D^{1/2}\nabla f(\mathbf{q}_k) + \alpha\mathbf{e}^T D^{1/2}I_{S_k}\Delta\mathbf{q}_k,\end{aligned}$$

where for the latter equality, we used the fact that $(D-A)\mathbf{e} = \mathbf{0}$.

From the proof of Theorem 1 we have that $\nabla f(\mathbf{q}_k) \leq \mathbf{0}$ and $\Delta\mathbf{q}_k \geq \mathbf{0}\ \forall k$. Hence, the last equality implies that

$$\|D^{1/2}\nabla f(\mathbf{q}_{k+1})\|_1 \leq \|D^{1/2}\nabla f(\mathbf{q}_k)\|_1.$$

Using the above inequality and $D^{1/2}\nabla f(\mathbf{q}_0) = -\alpha\mathbf{s}$ in (15) we get

$$\|s\|_1 \geq \rho\,\mathrm{vol}(S_k)\ \forall k.$$

Since $S_k \to \mathscr{S}_*$ as $k \to \infty$ then $\|s\|_1 \geq \rho\,\mathrm{vol}(\mathscr{S}_*)$. To prove this use the fact that Algorithm 3 is a convergent algorithm. Therefore, as Algorithm 3 converges to the optimal solution $\mathbf{q}_*$ then the set S_k converges to $\mathscr{S}_*$, i.e., S_k consists of the same elements as $\mathscr{S}_*$, thus inequality $\|s\|_1 \geq \rho\,\mathrm{vol}(S_k)\ \forall k$ holds for $\mathscr{S}_*$ as well, i.e., $\|s\|_1 \geq \rho\,\mathrm{vol}(\mathscr{S}_*)$. □

We are now ready to derive the overall iteration complexity and the total running time of Algorithm 3. For this, we will make use of strong convexity of f in (1). It is easy to see that f is α-strongly convex. Indeed, Q in (1) can be rewritten as $Q = \alpha I + (1-\alpha)\mathcal{L}/2$. Since $\mathcal{L} \succeq 0$, it follows that $Q \succeq \alpha I$. However, Theorem 1 guarantees that for each iteration of Algorithm 3, one has $\mathrm{supp}(\mathbf{q}_k) \subseteq \mathscr{S}_*\ \forall k$. Naturally, the function f, restricted to vectors with support in $\mathscr{S}_*$, has a better strong convexity

parameter. Let $\mathcal{L}_{\mathscr{S}_*}$ be the principal sub-matrix of the normalized graph Laplacian $\mathcal{L} = I - D^{-1/2}AD^{-1/2}$ by removing the rows and columns with indices in $V \setminus \mathscr{S}_*$. It is clear that such restricted strong convexity parameter, when restricted to all vectors $\mathbf{q}$ such that $\text{supp}(\mathbf{q}) \subseteq \mathscr{S}_*$, is $\alpha + (1-\alpha)\lambda_{min}(\mathcal{L}_{\mathscr{S}_*})/2$, which, if $\lambda_{min}(\mathcal{L}_{\mathscr{S}_*}) > 0$, is larger than α.

Now consider the local conductance constant, defined in [6] as

$$H(\mathscr{S}) := \min_{S \subset \mathscr{S}} \Phi(S).$$

Note this latter definition differs from (5) in that $H(\mathscr{S})$ measures the minimum conductance over all subsets of $\mathscr{S}$, as opposed to $\mathscr{V}$. Suppose $\mathscr{G}$ is connected and let $\|\mathbf{s}\|_1/\rho \leq \text{vol}(\mathscr{G})/2$, which, from Theorem 2, implies that $\text{vol}(\mathscr{S}_*) \leq \text{vol}(\mathscr{G})/2$. This is a reasonable assumption since, in the context of local graph clustering, it is not desired for the optimal support, $\mathscr{S}_*$, to have a volume larger than half of that of the whole graph, $\mathscr{G}$. In [6], a local Cheeger inequality is proved for the Dirichlet eigenvalue $\lambda_{min}(\mathcal{L}_{\mathscr{S}_*})$ of the induced subgraph on $\mathscr{S}_*$. For cases when such induced subgraph is connected, the lower bound given in [6] is in the form of

$$0 < \frac{(H(\mathscr{S}_*))^2}{2} \leq \lambda_{min}(\mathcal{L}_{\mathscr{S}_*}). \tag{16}$$

Luckily, it can be shown that, for any tolerance parameter in the termination condition, the optimal support $\mathscr{S}_*$ from Algorithm 3 corresponds to a connected induced subgraph of $\mathscr{G}$. Indeed, Step 4 of Algorithm 3 ensures that the procedure only touches the neighbors of the current non-zero nodes. Therefore, if the input reference set of nodes (captured by vector $\mathbf{s}$) corresponds to connected induced subgraphs of $\mathscr{G}$, the support of the output of Algorithm 3 and consequently $\mathscr{S}_*$ correspond to connected induced subgraphs of $\mathscr{G}$. Note that, in the cases where $\mathscr{G}$ is disconnected, the above reasoning still holds as long as ρ is chosen such that $\|\mathbf{s}\|_1/\rho \leq \text{vol}(\tilde{\mathscr{G}})/2$, where $\tilde{\mathscr{G}} \subset \mathscr{G}$ is the largest connected component of $\mathscr{G}$ that includes a reference node, i.e., a node i that satisfies $\mathbf{s}(i) \neq 0$ (otherwise, for the output of Algorithm 3, we might have $\mathscr{S}_* = \tilde{\mathscr{G}}$, which implies $\lambda_{min}(\mathcal{L}_{\mathscr{S}_*}) = 0$).

Thus, using (16), we can define the restricted strong convexity parameter of f as

$$\mu := \alpha + \frac{1-\alpha}{4}(H(\mathscr{S}_*))^2. \tag{17}$$

We are not aware of any better lower bound for $\lambda_{min}(\mathcal{L}_{\mathscr{S}_*})$. In fact, we believe that to lower bound this constant, one needs to make some strong assumptions about the target cluster that includes the reference node. As this is not our primary objective in this paper, we leave this for future work.

Using the restricted strong convexity parameter (17), Theorem 3 below gives the overall iteration complexity and total running time[2] of Algorithm 3.

[2] Iteration complexity refers to the worst-case number of iterations to satisfy the termination criterion and running time refers to the total amount of work, i.e., the per-iteration cost times iteration complexity.

Theorem 3 *Algorithm 3 with* $\|\mathbf{s}\|_\infty \geq \rho$ *requires at most*

$$T \in \mathcal{O}\left(\frac{1}{\mu}\log\left(\frac{2}{\epsilon^2\rho^2\alpha^2\min_j d_j}\right)\right), \tag{18}$$

iterations to converge to a solution that satisfies the termination criterion in Step 2, where μ *is as in* (17)*. Furthermore, the running time of Algorithm 3 is at most*

$$\mathcal{O}\left(\frac{(|\mathcal{S}_*| + \widehat{vol}(\mathcal{S}_*))}{\mu}\log\left(\frac{2}{\epsilon^2\rho^2\alpha^2\min_j d_j}\right)\right), \tag{19}$$

where $\mathcal{S}_*$ *is defined in* (14) *and* $\widehat{vol}(\mathcal{S}_*)$ *is the volume of* $\mathcal{S}_*$ *by assuming that the edges of the graph are unweighted, i.e., the sum of all neighbors for each node in* $\mathcal{S}_*$*. If we further suppose that* $|\mathcal{S}_*|, \widehat{vol}(\mathcal{S}_*) \in \mathcal{O}(vol(\mathcal{S}_*))$*, then using Theorem 2 and* $\|s\|_1 = 1$ (19) *simplifies to*

$$\mathcal{O}\left(\frac{2}{\rho\mu}\log\left(\frac{2}{\epsilon^2\rho^2\alpha^2\min_j d_j}\right)\right). \tag{20}$$

Proof Let the assumption about $\mathbf{s}$ from Theorem 1 hold. Then from Theorem 1 we have that $\mathbf{q}_k \geq 0\ \forall k$, i.e., we always remain in the the non-negative orthant. Denoting the restriction of $\psi(\mathbf{q})$ to $\mathbf{q} \geq 0$, by

$$\widehat{\psi}(\mathbf{q}) := \rho\alpha\mathbf{e}^T D^{1/2}\mathbf{q} + f(\mathbf{q}),$$

it follows that $\psi(\mathbf{q}) = \hat{\psi}(\mathbf{q})$ for all $\mathbf{q}$ in the non-negative orthant. From 1-Lipschitz continuity of ∇f w.r.t. ℓ_2 norm, it follows that $\widehat{\psi}$ is also smooth with the same parameter, i.e., 1. Hence, for any $\mathbf{q}_k$ from Algorithm 3, we have

$$\widehat{\psi}(\mathbf{q}) \leq \psi(\mathbf{q}_k) + (\mathbf{q} - \mathbf{q}_k)^T\nabla\widehat{\psi}(\mathbf{q}_k) + \frac{1}{2}\|\mathbf{q}_k - \mathbf{q}\|_2^2. \tag{21}$$

Since $\mathbf{q}_{k+1} \geq 0$ (see Theorem 1), $\mathbf{q}_{k+1} - \mathbf{q}_k = I_{S_k}\Delta\mathbf{q}_k$ and $\Delta\mathbf{q}_k = -\nabla_{S_k}\widehat{\psi}(\mathbf{q}_k)$ we have that

$$\psi(\mathbf{q}_{k+1}) \leq \psi(\mathbf{q}_k) - \frac{1}{2}\|\nabla_{S_k}\widehat{\psi}(\mathbf{q}_k)\|_2^2. \tag{22}$$

We have that f is μ-restricted strongly convex when restricted to all vectors $\mathbf{q}$ such that $\text{supp}(\mathbf{q}) \subseteq \mathcal{S}_*$, where $\mu := (\alpha + (1-\alpha)\lambda_{min}(\mathcal{L}_{\mathcal{S}_*})/2)$. Therefore, ψ is μ-restricted strongly convex as well and we have

$$\psi(\mathbf{q}_k) - \psi(\mathbf{q}_*) \leq \frac{1}{2\mu}\|g\|_2^2 \quad \forall g \in \partial\psi(\mathbf{q}_k),$$

where $\partial\psi(\mathbf{q}_k)$ is the sub-differential of ψ at $\mathbf{q}_k$. Notice that $I_{S_k}\nabla\widehat{\psi}_{S_k}(\mathbf{q}_k)$ is a valid sub-gradient of ψ at $\mathbf{q}_k$. This gives us

$$\psi(\mathbf{q}_k) - \psi(\mathbf{q}_*) \leq \frac{1}{2\mu}\|\nabla_{S_k}\widehat{\psi}(\mathbf{q}_k)\|_2^2. \tag{23}$$

Combining (22) and (23) and subtracting $\psi(\mathbf{q}_*)$ from both sides we get

$$\psi(\mathbf{q}_{k+1}) - \psi(\mathbf{q}_*) \leq (1-\mu)\,(\psi(\mathbf{q}_k) - \psi(\mathbf{q}_*)),$$

which implies linear convergence. Applying the last inequality recursively we get that Algorithm 3 requires at most $T \in \mathcal{O}((1/\mu)\log(1/\hat{\epsilon}))$ iterations to obtain a solution $\mathbf{q}_T$ such that $\psi(\mathbf{q}_T) - \psi(\mathbf{q}_*) \leq \hat{\epsilon}$.

From (22) we have that

$$\psi(\mathbf{q}_*) \leq \psi(\mathbf{q}_k) - \frac{1}{2}\|\nabla_{S_k}\widehat{\psi}(\mathbf{q}_k)\|_2^2 \quad \forall k.$$

Using the above and $\psi(\mathbf{q}_T) - \psi(\mathbf{q}_*) \leq \hat{\epsilon}$, we get $\|\nabla_{S_k}\widehat{\psi}(\mathbf{q}_T)\|_\infty^2 \leq 2\hat{\epsilon}$, which is equivalent to

$$-\rho\alpha - \left(\frac{2\hat{\epsilon}}{d_i}\right)^{1/2} \leq \frac{\nabla_i f(\mathbf{q}_T)}{d_i^{1/2}} \leq \rho\alpha + \left(\frac{2\hat{\epsilon}}{d_i}\right)^{1/2}$$

$\forall i \in S_k$. From Theorem 1 we have that $\nabla_i f(\mathbf{q}_T) > -\rho\alpha d_i^{1/2}$ $\forall i \in [n]\backslash S_k$. Let $\epsilon \in (0,1)$ be the accuracy parameter of Algorithm 3. As a result, by setting $\hat{\epsilon} := (\epsilon^2\rho^2\alpha^2 \min_j d_j)/2$ and using the fact that $\nabla f(\mathbf{q}_k) \leq 0$ $\forall k$ from Lemma 1, we get that after

$$T \in \mathcal{O}\left(\frac{1}{\mu}\log\left(\frac{2}{\epsilon^2\rho^2\alpha^2 \min_j d_j}\right)\right)$$

iterations the output of Algorithm 3 satisfies $-(1+\epsilon)\rho\alpha d_i^{1/2} \leq \nabla_i f(\mathbf{q}_T) \leq 0$ $\forall i$, which is the termination criterion in Step 2 of Algorithm 3.

From Theorem 1 we have that $S_k \subseteq \mathcal{S}_*$ and $|S_k| \leq |\mathcal{S}_*|$ $\forall k$. The set S_k in Step 3 of Algorithm 3 can be updated in $\mathcal{O}(\widehat{\mathrm{vol}}(S_{k-1}))$ operations, where $\widehat{\mathrm{vol}}(S_{k-1})$ is the volume of S_{k-1} by assuming that the edges of the graph are unweighted, i.e., the sum of all neighbors for each node in $\mathcal{S}_*$. The quantity $\widehat{\mathrm{vol}}(S_{k-1})$ is upper bounded by $\widehat{\mathrm{vol}}(\mathcal{S}_*)$. Therefore, Step 3 costs at most $\mathcal{O}(\widehat{\mathrm{vol}}(\mathcal{S}_*))$ operations. Step 4 of Algorithm 3 requires at most $\mathcal{O}(|\mathcal{S}_*|)$ operations. Similarly, Steps 5 and 6 require at most $\mathcal{O}(|\mathcal{S}_*|+\widehat{\mathrm{vol}}(\mathcal{S}_*))$ operations. Finally, Step 7 does not perform any computations. Putting the operations performed in all of the steps together, using the iteration complexity result in (18) and the result of Theorem 2, we get (19) and (20). □

Remark 1 The assumption $|\mathcal{S}_*|, \widehat{\mathrm{vol}}(\mathcal{S}_*) \in \mathcal{O}(\mathrm{vol}(\mathcal{S}_*))$ in the latter part of Theorem 3 holds for many types of graphs, e.g., unweighted. Indeed, such assumption

is commonly made in the related literature, including APPR in [1] and many others [2,15,19,23,25].

Remark 2 For unweighted graphs, according to Theorem 3, the worst-case running time of Algorithm 3 is $\mathcal{O}\left(\log\left(2/(\epsilon^2\rho^2\alpha^2)\right)/(\rho\mu)\right)$ (ignoring small terms and using $\|\mathbf{s}\|_1 \leq 1$), where μ was defined in (17). However, Andersen et al. [1, Theorems 1 and 5] state that the worst-case running time of APPR is $\mathcal{O}(1/(\rho\alpha))$. Despite the fact that $\mu \geq \alpha$, since (20) involves $H(\mathscr{S}_*)$ as well as a "log" factor, it is unfortunately difficult to directly compare the worst-case running time of Algorithm 3 with that of APPR.

It is possible to replace the output of APPR with the solution of (8) and still maintain the combinatorial guarantees for PageRank-Nibble as in [1, Theorem 7]; see also the discussion in Sect. 3. This can be shown using the fact that ISTA Algorithm 3 for ℓ_1-regularized PR satisfies the invariance property of APPR (see [1, Section 3]). Moreover, all algorithms at termination satisfy $\|D^{-1/2}\nabla f(q_k)\|_\infty \leq \rho\alpha$. The proof is identical to that of Theorem 7 in [1] and is, therefore, omitted. Relatedly, to ensure that the solutions of Algorithm 3 and APPR share the same theoretical clustering guarantees, the parameter ρ of Algorithm 3 must be set with respect to that of APPR. More specifically, let $\rho, \tilde{\rho} \in (0, 1)$ be the parameters of the ℓ_1-regularized PR problem (8) and APPR, respectively. Moreover, let the vector $\mathbf{s} \geq 0$ be chosen such that $\mathbf{s}(i) \geq \max(\rho, \tilde{\rho})$ for all i with $\mathbf{s}(i) \neq 0$, e.g., $\mathbf{s}(i) = 1$ for the reference node i and zero elsewhere. Then APPR algorithm at termination gives an output which satisfies (6) while Algorithm 3 is terminated when $\|D^{-1/2}\nabla f(\mathbf{q}_k)\|_\infty \leq (1+\epsilon)\rho\alpha$. Hence, one can set $\rho \leq \tilde{\rho}/(1+\epsilon)$ to ensure that the termination criterion of Algorithm 3 matches that of APPR; see Sect. 6 for numerical experiments.

6 Experiments

In this section, we numerically demonstrate that ℓ_1-reg. PR problem achieves in practice similar graph cut guarantees as APPR. The experiments are performed on a single thread of a 64-core machine with four 2.4 GHz 16-core AMD Opteron 6278 processors. The implementations are written using C++ code and compiled with the g++ compiler version 4.8.0. We use a set of undirected, unweighted real-world graphs from the Stanford Network Analysis Project (http://snap.stanford.edu/data), whose sizes are shown in Table 1. We present the performance of greedy and heuristic versions of APPR and ISTA. In particular, in the following figures APPR GREEDY is Algorithm 2 where in step 3 we select the i'th coordinate with the largest partial derivative $\nabla_i f(q_k)$ in absolute value. APPR HEURISTIC is Algorithm 2 where we select approximately

Table 1 Graph inputs

Input graph	Num. vertices	Num. edges[a]
wiki-Talk	2,394,385	4,659,565
soc-LJ	4,847,571	42,851,237
cit-Patents	6,009,555	16,518,947
com-Orkut	3,072,627	117,185,083

[a]Number of unique undirected edges

Table 2 Number of non-zeros for the output solution p_k of each algorithm for the four experiments in Fig. 1

Input graph	APPR GREEDY	APPR HEUR.	ISTA
wiki-Talk	326	334	326
soc-LJ	159	159	159
cit-Patents	210	211	198
com-Orkut	447	448	442

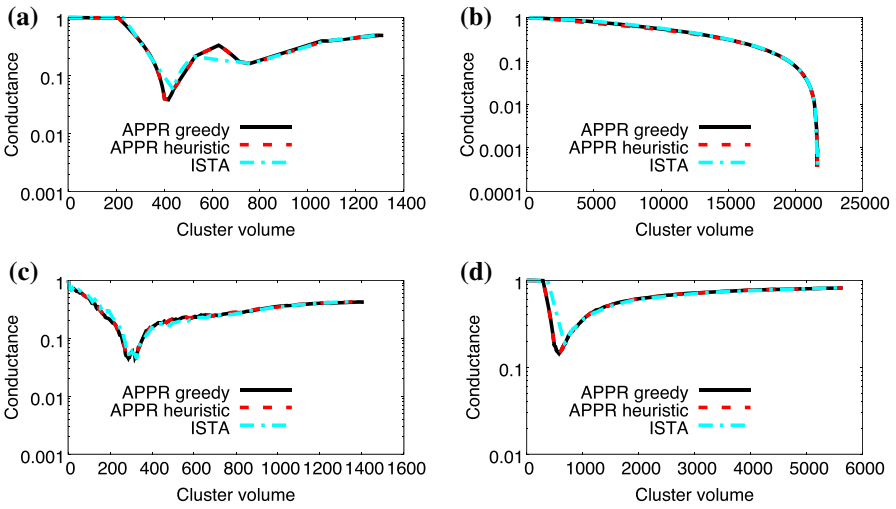

Fig. 1 Conductance versus cluster volume. The axes of all plots are in log-scale. This figure shows the conductance criterion for the clusters which are produced by the sweep procedure applied on the output of each algorithm. The volume of the clusters is shown in increasing size. **a** wiki-Talk, $\alpha = 0.1$, $\rho = 10^{-5}$, **b** soc-LJ, $\alpha = 0.1$, $\rho = 10^{-5}$, **c** cit-Patents, $\alpha = 0.1$, $\rho = 10^{-5}$ and **d** com-Orkut, $\alpha = 0.1$, $\rho = 10^{-5}$

the i'th coordinate with the largest $\nabla_i f(q_k)$ in absolute value. In particular, a priority queue of coordinates is maintained which initially contains the starting vertex only. On each iteration we select the highest-priority coordinate in the queue and update the coordinate and its neighbors accordingly. For each neighbor, insert it in the queue if it is above the threshold with priority equal to the chosen coordinate. Note that this is a heuristic because we select coordinates based on their priority when they are initially inserted in the queue, and do not update their priorities later on. It is important to mention that the heuristic versions of the algorithms are guaranteed to converge in theory but not with linear convergence rate. However, there exist examples where one can maintain the linear convergence rate, as discussed in Section 5 in [7].

For all experiments we set $s_v = 1$ and zero elsewhere, where the coordinate/node v is chosen based on a search of over 10^4 starting nodes. We used the starting vertex that gave the best conductance. We conduct all experiments by fixing $\alpha = 0.1$ and choose the ρ values empirically such that we get clusters with at least 100 nodes each. This agrees with the observations in [17] regarding the size of local clusters in large-scale graphs.

We use the same rounding procedure as the one described in Section 2.2 in [1] for the original APPR algorithm, which is based on the conductance criterion. In Fig. 1 we present the conductance criterion (y-axis) versus the volume of the clusters (x-axis) produced by the sweep procedure in increasing order. All algorithms obtain approximately the same conductance value after the rounding procedure. The number of non-zeros of the output for each algorithm is given in Table 2. Notice that the output of the ℓ_1-reg. PR problem, which is obtained by ISTA, has at most the same number of non-zeros as the greedy and the heuristic versions of APPR.

7 Conclusion

In this paper, we derived and studied a variational formulation of the celebrated local spectral clustering algorithm APPR in [1]. Through this explicit formulation, we argued that an existing state-of-the-art optimization algorithm, i.e., ISTA [24], can be applied in a way as to result in a strongly local algorithm, which only requires access to a small portion of the graph. In addition, we showed that the running time of this algorithm only depends on the volume of non-zeros of the solution, as opposed to the entire graph. From a broader perspective, we hope that this variational viewpoint serves as a bridge across two seemingly disjoint fields of graph processing and numerical optimization, and allows one to leverage well-studied, numerically robust, and efficient optimization algorithms for processing today's large graphs. For example, one might be able to apply a modification of accelerated ISTA, i.e. FISTA [24] to further improve upon the efficiency of local graph clustering algorithms. This can indeed be a direction for future research, which we plan to undertake.

Acknowledgements MM would like to thank the Army Research Office and the Defense Advanced Research Projects Agency for partial support of this work. JS was supported by the Miller Institute for Basic Research in Science at UC Berkeley. JS would also like to acknowledge the Miller Institute for Basic Research in Science at UC Berkeley.

References

1. Andersen, R., Chung, F., Lang, K.: Local graph partitioning using pagerank vectors. In: FOCS '06 Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, pp. 475–486 (2006)
2. Andersen, R., Lang, K.: An algorithm for improving graph partitions. In: SODA '08 Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 651–660 (2008)
3. Arora, S., Rao, S., Vazirani, U.: Expander flows, geometric embeddings and graph partitioning. J. ACM **56**(2), 5 (2009)
4. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. **2**(1), 183–202 (2009)
5. Cheeger, J.: A lower bound for the smallest eigenvalue of the Laplacian. In: Problems in Analysis, Papers Dedicated to Salomon Bochner, pp. 195–199. Princeton University Press (1969)
6. Chung, F.: Random walks and local cuts in graphs. Linear Algebra Appl. **423**, 22–32 (2007)
7. Dhillon, I.S., Ravikumar, P.K., Tewari, A.: Nearest neighbor based greedy coordinate descent. In: Advances in Neural Information Processing Systems 24 (NIPS 2011) (2011)
8. Eiron, N., McCurley, K.S., Tomlin, J.A.: Ranking the web frontier. In: Proceedings of the 13th International Conference on World Wide Web, pp. 309–318 (2004)

9. Fountoulakis, K., Cheng, X., Shun, J., Roosta-Khorasani, F., Mahoney, M.W.: Exploiting optimization for local graph clustering. Technical report. Preprint arXiv:1602.01886 (2016)
10. Gleich, D.F.: Pagerank beyond the web. SIAM Rev. **57**(3), 321–363 (2015)
11. Gleich, D.F., Mahoney, M.W.: Anti-differentiating approximation algorithms: a case study with min-cuts, spectral, and flow. In: Proceedings of the 31st International Conference on Machine Learning, pp. 1018–1025 (2014)
12. Grady, L., Schwartz, E.L.: Isoperimetric partitioning:a new algorithm for graph partitioning. SIAM J. Sci. Comput. **27**(6), 1844–1866 (2006)
13. Hall, K.M.: An r-dimensional quadratic placement algorithm. Manag. Sci. **17**(3), 219–229 (1970)
14. Jeub, L.G.S., Balachandran, P., Porter, M.A., Mucha, P.J., Mahoney, M.W.: Think locally, act locally: the detection of small, medium-sized, and large communities in large networks. Phys. Rev. E **91**(1), 012,821 (2015)
15. Kloster, K., Gleich, D.F.: Heat kernel based community detection. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1386–1395 (2014)
16. Leighton, T., Rao, S.: An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. In: 29th Annual Symposium on Foundations of Computer Science, pp. 422–431 (1988)
17. Leskovec, J., Lang, K.J., Dasgupta, A., Mahoney, M.W.: Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters. Internet Math. **6**(1), 29–123 (2011)
18. Mahoney, M.W., Orecchia, L., Vishnoi, N.K.: A local spectral method for graphs: with applications to improving graph partitions and exploring data graphs locally. J. Mach. Learn. Res. **13**, 2339–2365 (2012)
19. Orecchia, L., Zhu, Z.A.: Flow-based algorithms for local graph clustering. In: SODA '14 Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1267–1286 (2014)
20. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: bringing order to the web. Technical report, Stanford InfoLab (1999)
21. Parikh, N., Boyd, S.: Proximal algorithms. Found. Trends Optim. **1**(3), 123–231 (2013)
22. Pothen, A., Simon, H.D., Liou, K.P.: Partitioning sparse matrices with eigenvectors of graphs. SIAM J. Matrix Anal. Appl. **11**(3), 430–452 (1990)
23. Spielman, D.A., Teng, S.H.: A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. SIAM J. Sci. Comput. **42**(1), 1–26 (2013)
24. Sra, S., Nowozin, S., Wright, S.J.: Optimization for Machine Learning. MIT Press, Cambridge (2012)
25. Veldt, N., Gleich, D.F., Mahoney, M.W.: A simple and strongly-local flow-based method for cut improvement. Accepted to ICML (2016)