Incremental Computation of Pseudo-Inverse of Laplacian

Gyan Ranjan, Zhi-Li Zhang and Dan Boley

Department of Computer Science & Engineering, University of Minnesota, Minneapolis, USA,
granjan@cs.umn.edu, zhzhang@cs.umn.edu,boley@cs.umn.edu

Abstract. A divide-and-conquer based approach for computing the Moore-Penrose pseudo-inverse of the combinatorial Laplacian matrix ($\mathbf{L}^+$) of a simple, undirected graph is proposed. The nature of the underlying sub-problems is studied in detail by means of an elegant interplay between $\mathbf{L}^+$ and the effective resistance distance (Ω). Closed forms are provided for a novel *two-stage* process that helps compute the pseudo-inverse incrementally. Analogous scalar forms are obtained for the converse case, that of structural regress, which entails the breaking up of a graph into disjoint components through successive edge deletions. The scalar forms in both cases, show absolute element-wise independence at all stages, thus suggesting potential parallelizability. Analytical and experimental results are presented for dynamic (time-evolving) graphs as well as large graphs in general (representing real-world networks). An order of magnitude reduction in computational time is achieved for dynamic graphs; while in the general case, our approach performs better in practice than the standard methods, even though the worst case theoretical complexities may remain the same: an important contribution with consequences to the study of online social networks.

1 Introduction

The combinatorial Laplacian matrix of a graph finds use in various aspects of structural analysis. The eigen spectrum of the Laplacian determines significant topological characteristics of the graph, such as minimal cuts, clustering and the number of spanning trees [5,13,18]. Likewise, the Moore-Penrose pseudo-inverse and the sub-matrix inverses of the Laplacian have evoked great interest in recent years. Their applications span fields as diverse as probability and mathematical chemistry, collaborative recommendation systems and social networks, epidemiology and robustness of networks and inter-dependent networks [15,16,17,31,21,14,24,23]. A brief discussion of the specific applications is provided for reference in a subsequent section (cf. §6). Despite such versatility, the pseudo-inverse and the sub-matrix inverses of the Laplacian suffer a practical handicap. These matrices are notoriously expensive to compute. The standard matrix factorization and inversion based methods employed to compute them [4,31] incur an $O(n^3)$ computational time, n being the order of the graph (number of vertices in the graph). This clearly impedes their utility particularly when the graphs are either dynamic, i.e. changing with time, or simply of large orders, i.e. have millions of nodes. Online social networks (OSN), typically represented as graphs, qualify on both counts. With time, the number of users as well as the relationships between them changes, thus requiring regular re-computations. In physical networks such as data/communication networks and power grids, dynamics in networ topologies may rise from link/node failures and repairs. For OSNs and other networks with millions or more of nodes, an $O(n^3)$ computational cost is clearly prohibitive. An approach for incremental updates is imperative, particularly given that such changes, in most cases, may be local in nature.

In this work, we provide a novel divide-and-conquer based approach for computing the Moore-Penrose pseudo-inverse of the Laplacian for an undirected graph. This, in turn, determines all of its sub-matrix inverses as well. The divide operation in our approach entails determining an arbitrary *connected bi-partition* of the graph $G(V, E)$ — a cut of the graph that is made up of exactly two connected sub-graphs (say $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$) — by deleting κ edges from it. As $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ are simple and connected themselves, the pseudo-inverse of their Laplacians, when computed, constitute solutions to two independent sub-problems. Better still, they can be computed in parallel (given two machines instead of one).

In the conquer step, we recombine these independent solutions in an iterative manner by re-introducing the edges in the cut set one at a time to reconstruct the original graph incrementally. Clearly, this process yields a sequence of intermediate connected spanning sub-graphs of G, (say $\{G_1, G_2\} \rightarrow G_3 \rightarrow G_4 \rightarrow ... \rightarrow G_{\kappa+2}$), where $G_{\kappa+2} = G(V, E)$. The first transition $\{G_1, G_2\} \rightarrow G_3$ represents a point of *singularity* whence the disjoint components $\{G_1, G_2\}$ get connected through a *bridge* edge to yield G_3, a sub-graph with exactly one component. We call this stage the *first join* in our process. Post the first join, all intermediate sub-graphs from G_4 to $G_{\kappa+2}$ are obtained by introducing an edge in a sub-graph that is already connected. We call these atomic steps *edge firings* (details in a subsequent section).

We then show that the pseudo-inverse of the Laplacian for any intermediate sub-graph in this sequence is determined entirely in terms of the pseudo-inverse of the Laplacian for its predecessor. Our results, presented in an element-wise scalar form, reveal several interesting properties of the sub-problems. First and foremost, if n be the order of the graph $G(V, E)$, then the cost incurred at each intermediate stage is $O(n^2)$ if the solution to the sub-problems for the immediate predecessor is known. Therefore, the cost of computing the pseudo-inverse for $G(V, E)$ is $O(\kappa \cdot n^2)$, if the pseudo-inverses for $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ are known. Secondly, using these forms, each element of the pseudo-inverse for an intermediate graph can be computed independent of the other elements. Hence, given multiple machines, the overall computational time is reduced further through parallelization. Moreover, we obtain similar closed form solutions for the case of structural regress of the graph, i.e. when vertices or edges are deleted from it. A straightforward consequence is that the pseudo-inverses for dynamic time-evolving graphs can now be updated when a node joins or leaves the network or an edge (a relationship) appears/disappears in it at an $O(n^2)$ cost overall (as $\kappa << n$).

Last but not least, we use these insights to compute the pseudo-inverses of the Laplacians of large real-world networks from the domain of online social networks. Real-world networks, and social ones in particular, are reported to have some notable characteristics such as edge sparsity, *power-law* and scale-free degree distributions [3], *small-world* characteristics [30] etc. Given these properties, we note that interesting algorithms (heuristics) can be developed for fast and parallel computations for the general case based on our divide-and-conquer strategy. Thus, even though the theoretical worst case costs stay at $O(n^3)$ for general graphs, the practical gains are significant enough to warrant attention. We discuss both analytical and experimental aspects of these in detail in the subsequent sections.

The remainder of the paper is organized into the following sections: we begin by introducing the preliminaries of our work — the pseudo-inverse and the sub-matrix inverses of the Laplacian along with their properties; and the interplay of the pseudo-inverse and the effective resistance distance — in §2. In §3, we describe our divide-and-conquer strategy involving connected bi-partitions and the *two-stage* process for computing the Moore-Penrose pseudo-inverse of the Laplacian. Relevant scalar forms are presented in each case. In §4, we establish the same closed forms for a graph in regress i.e. deleting edges one at a time until the graph breaks into two. We then apply the divide-and-conquer methodology to compute the pseudo-inverses for dynamically changing graphs as well as those of real world networks in §5. We provide a brief overview of related literature, focusing on specific application scenarios of L^+ in §6. The paper is finally concluded in §7 with a summary of results and a discussion of potential future works.

2 The Laplacian, Sub-Matrix Inverses and A Distance Function

In this section, we provide a brief introduction to the set of matrices studied in this work, namely, the combinatorial Laplacian of a graph ($\mathbf{L}$), its Moore-Penrose pseudo-inverse ($\mathbf{L}^+$) and the set of sub-matrix inverses of $\mathbf{L}$ (§2.1). We then demonstrate how all the sub-matrix inverses of the Laplacian can be computed in terms of the pseudo-inverse in §2.2. Finally, in §2.3 we describe the relationship between the *effective resistance distance*, a Euclidean metric, and the elements of the Moore-Penrose pseudo-inverse of the Laplacian — an equivalence that we exploit to great advantage in the rest of this work.

2.1 The Laplacian and its Moore-Penrose Pseudo-Inverse

Let $G(V, E)$ be a simple, connected and undirected graph. We denote by $n = |V(G)|$ the number of nodes/vertices in G, also called the *order* of the graph G, and by $m = |E(G)|$ the number of links/edges.

The adjacency matrix of $G(V, E)$ is defined as $\mathbf{A} \in \Re^{n \times n}$, with elements $[\mathbf{A}]_{xy} = a_{xy} = a_{yx} = [\mathbf{A}]_{yx} = w_{ij}$, if $x \neq y$ and $e_{xy} \in E(G)$ is an edge; 0 otherwise. Here, the weight of the edge w_{ij} is a measure of *affinity* between nodes i and j. Clearly, $\mathbf{A}$ is real and symmetric. The degree matrix $\mathbf{D}$, is a diagonal matrix where $[\mathbf{D}]_{xx} = d_{xx} = d(x) = \sum_{y \in V(G)} a_{xy}$, is the weighted degree of node $x \in V(G)$; the sum of all edge weights (affinities) emanating from x. Also, $vol(G) = \sum_{x \in V(G)} d(x)$, is called the *volume* of the graph G — the sum total of affinities between all pairs of vertices in G. The combinatorial Laplacian of the graph is then given by:

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{1}$$

It is easy to see, from the definition in (1) above, that the Laplacian $\mathbf{L}$ is a real, symmetric and doubly-centered matrix (each row/column sum is 0). More importantly, $\mathbf{L}$ admits an eigen decomposition of the form $\mathbf{L} = \mathbf{\Phi \Lambda \Phi'}$ where the columns of $\mathbf{\Phi}$ constitute the set of orthogonal eigen vectors of $\mathbf{L}$ and $\mathbf{\Lambda}$ is a diagonal matrix with $[\Lambda]_{ii} = \lambda_i : 1 \leq i \leq n$; being the n eigen values of $\mathbf{L}$. It is well established that for a connected undirected graph $G(V, E)$, $\mathbf{L}$ is positive semi-definite i.e. it has a unique smallest eigen value $\lambda_1 = 0$. The rest of the $n-1$ eigen values are all positive. Thus, $\mathbf{L}$ is rank deficient ($rank(\mathbf{L}) = n - 1 < n$) and consequently singular. Its inverse, in the usual sense, does not exist.

However, the Moore-Penrose pseudo-inverse of $\mathbf{L}$, denoted henceforth by $\mathbf{L}^+$, does exist and is unique [4]. Like $\mathbf{L}$, $\mathbf{L}^+$ is also real, symmetric, doubly centered and positive semi-definite. Moreover, the eigen decomposition of $\mathbf{L}^+$ is given by $\mathbf{L}^+ = \Phi \Lambda^+ \Phi'$, with the same set of orthogonal eigen-vectors as that of $\mathbf{L}$. The set of eigen values of $\mathbf{L}^+$, given by the diagonal of the matrix $\mathbf{\Lambda}^+$, is composed of $\lambda_1^+ = 0$ and the reciprocals of the positive eigen-values of $\mathbf{L}$. We denote by l_{xy}^+, the element in the x^{th} row and y^{th} column of $\mathbf{L}^+$ (a convention followed for all matrices henceforth). We emphasize that even when the matrix $\mathbf{L}$ is sparse (which is the case with real world networks), $\mathbf{L}^+$ is always a full matrix. In fact, for a connected graph, all the elements of $\mathbf{L}^+$ are non-zero.

A straightforward approach for computing $\mathbf{L}^+$ is through the eigen-decomposition of $\mathbf{L}$, followed by an inversion of its non-zero eigen values, and finally reassembling the matrix as discussed above. In practice, however, mathematical software, such as MATLAB, use singular value decomposition to compute the pseudo-inverse of matrices (cf. *pinv* in the standard library). This general SVD based method does not exploit the special structural properties of $\mathbf{L}$ and incurs $O(n^3)$ computational time, n being the number of nodes in the graph. An alternative exists [11,31] specifically for computing $\mathbf{L}^+$ for a simple, connected, undirected graph. A $rank(1)$ perturbation of the matrix $\mathbf{L}$ makes it invertible. $\mathbf{L}^+$ can then be computed from this perturbed matrix as follows:

$$\mathbf{L}^+ = \left(\mathbf{L} + \frac{1}{n}\mathbf{J} \right)^{-1} - \frac{1}{n}\mathbf{J} \tag{2}$$

where $\mathbf{J} \in \Re^{n \times n}$ is a matrix of all $1's$. Although the theoretical cost for this method is also $O(n^3)$, in practice it works faster for graphs of arbitrary orders and edge densities than the standard *pinv* method. However, for applying such a method to *dynamically evolving* graphs in which small local modifications (e.g., adding or deleting an edge or a node), repeated computations of matrix inverse incur undue heavy computational costs. In what follows, we show that the computation of the Moore-Penrose pseudo-inverse of the Laplacian can be done in a divide-and-conquer fashion. Our method allows efficient incremental updates of $\mathbf{L}^+$ for dynamically changing graphs, without having to compute $\mathbf{L}^+$ all over again. Moreover, computing $\mathbf{L}^+$ for large graphs becomes feasible, in principle, through parallelization of (smaller) independent sub-problems over multiple machines, which can then be re-combined at $O(n^2)$ cost per edge across a division (details in a subsequent section). But first we need to establish a few more preliminary results to further motivate our study.

2.2 Sub-Matrix Inverses of L

As described in the previous section, the combinatorial Laplacian $\mathbf{L}$ of a connected graph $G(V, E)$, is singular and thus non-invertible. However, given that its rank is $n-1$, any $n-1$ combination of columns (or rows) of $\mathbf{L}$ constitutes a linearly independent set. Hence, any $(n-1 \times n-1)$ sub-matrix of $\mathbf{L}$ is invertible. Indeed, the

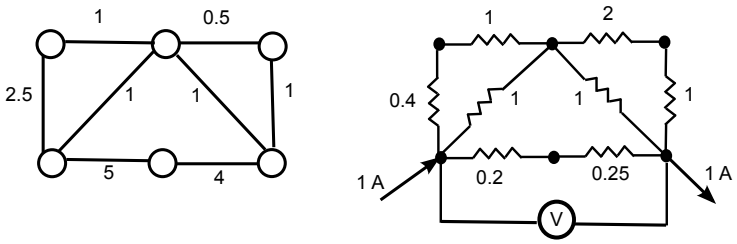


Fig. 1. A simple graph G and its EEN.

inverses of such $(n-1 \times n-1)$ sub-matrices are made use of in several graph analysis problems: enumerating the spanning trees and spanning forests of the graph [16], determining the random-walk betweenness of the nodes of the graph [21], to name a few. However, the cost of computing an $(n-1 \times n-1)$ sub-matrix inverse is still $O(n^3)$. To compute all such sub-matrix inverses amounts to a time complexity of $O(n^4)$. In the following, we show how they can be computed efficiently through $\mathbf{L}^+$.

Theorem 1 *Let $\mathbf{L}(\{\overline{n}\},\{\overline{n}\})$ be an $(n-1 \times n-1)$ sub-matrix of $\mathbf{L}$ formed by removing the n^{th} row and n^{th} column of $\mathbf{L}$. Then $\forall (x,y) \in V(G) \times V(G)$:*

$$[\mathbf{L}(\{\overline{n}\},\{\overline{n}\})^{-1}]_{xy} = l^+_{xy} - l^+_{xn} - l^+_{ny} + l^+_{nn} \tag{3}$$

The result in Theorem 1 above expresses, in scalar form, the general element (x^{th} row, y^{th} column) of the inverse of the sub-matrix $\mathbf{L}(\{\overline{n}\},\{\overline{n}\})$ in terms of the elements of $\mathbf{L}^+$, as claimed. As the choice of the n^{th} row and column is arbitrary, we can see that the result holds in general for any $(n-1 \times n-1)$ sub-matrix (permuting the rows and columns of $\mathbf{L}$ as per need). The cost of computing $\mathbf{L}(\{\overline{n}\},\{\overline{n}\})^{-1}$ for a given vertex n is $O(n^2)$. Therefore, all sub-matrix inverses can be computed in $O(n^3)$ time from $\mathbf{L}^+$, which itself can be computed in $O(n^3)$ time, even if the standard methods are used. This is clearly an order of magnitude improvement. Henceforth, we focus entirely on $\mathbf{L}^+$.

2.3 The Effective Resistance Distance and $\mathbf{L}^+$

An interesting analogy exists between graphs and resistive electrical circuits [12,17]. Given a simple, connected and undirected graph $G(V,E)$, the equivalent electrical network (EEN) of the graph can be formed by replacing each edge $e_{ij} \in E(G)$, of weight w_{ij} with an electrical resistance $\omega_{ij} = w_{ij}^{-1}$ *ohm* (cf. Fig. 1). A distance function can then be defined between any pair of nodes $(x,y) \in V(G) \times V(G)$ in the resulting EEN as follows:

Definition 1 *Effective Resistance (Ω_{xy}): The voltage developed between nodes x and y, when a unit current is injected at node x and is withdrawn at node y.*

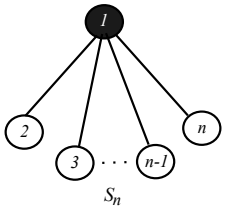
It is well established that the square root of the effective resistance distance ($\sqrt{\Omega_{xy}}$) is a Euclidean metric with interesting applications [17]. Amongst other things, it determines the expected length of random commutes between node pairs in the graph: $C_{xy} = vol(G)\ \Omega_{xy}$, [9,29]. More importantly, Ω_{xy} can be expressed in terms of the elements of $\mathbf{L}^+$ as follows:

$$\Omega_{xy} = l^+_{xx} + l^+_{yy} - l^+_{xy} - l^+_{yx} \tag{4}$$

We now invert the elegant form in (4) to derive an important result in the following lemma which gives us the general term of $\mathbf{L}^+$ in terms of the distance function Ω.

Lemma 1 $\forall (x,y,z) \in V(G) \times V(G) \times V(G)$:

$$l^+_{xy} = \frac{1}{2n}\left[\sum_{z=1}^{n}(\Omega_{xz} + \Omega_{zy} - \Omega_{xy})\right] - \frac{1}{2n^2}\sum_{x=1}^{n}\sum_{y=1}^{n}\Omega_{xy} \tag{5}$$



0.16	-0.04	-0.04	-0.04	-0.04
-0.04	0.76	-0.24	-0.24	-0.24
-0.04	-0.24	0.76	-0.24	-0.24
-0.04	-0.24	-0.24	0.76	-0.24
-0.04	-0.24	-0.24	-0.24	0.76

$\mathrm{L}^{+}_{S_5}$

Fig. 2. The Star Graph: Pre-computed $\mathbf{L}^{+}_{S_n}$ for $n = 5$.

The RHS in Lemma 1 above is composed of two terms: a triangle inequality of effective resistances [29] and a double summand over all pairwise effective resistances in the EEN. It is easy to see that the double-summand simply reduces to a scalar multiple of the trace of $\mathbf{L}^{+}$ ($Tr(\mathbf{L}^{+}) = \sum_{z=1}^{n} l^{+}_{zz}$). Thus the functional half that determines the elements of $\mathbf{L}^{+}$, is the triangle inequality of the effective resistances, while the double summand contributes an additive constant to all the entries of $\mathbf{L}^{+}$. We illustrate the utility of this result, with the help of two kinds of graphs on the extremal ends of the connectedness spectrum: the star and the clique.[1]

The Star A star of order n is a tree with exactly one vertex of degree $n-1$, referred to as the *root*, and $n-1$ pendant vertices each of degree 1, called *leaves*, (cf. Fig. 2). By definition, a singleton isolated vertex is also a degenerate star albeit with no leaves. It is easy to see that S_n, being a tree, is the most sparse connected graph of order n (with exactly $n-1$ edges). Also, S_n is the most compact tree of its order (lowest diameter). In the following, we show how $\mathbf{L}^{+}_{S_n}$ can be computed using the result of Lemma 1.

Corollary 1 *For a star graph S_n of order n, with node v_1 as root and nodes $\{v_2, v_3, ..., v_n\}$ as leaves, $\mathbf{L}^{+}_{S_n}$ is given by:*

$$l^{+}_{11} = \frac{n-1}{n^2} \qquad \textit{and} \qquad \forall x : 2 \leq x \leq n, \quad l^{+}_{1x} = l^{+}_{x1} = -\frac{1}{n^2} \tag{6}$$

$$\forall x : 2 \leq x \leq n, \quad l^{+}_{xx} = \frac{n^2-n-1}{n^2} \qquad \textit{and} \qquad \forall x \neq y : 2 \leq x, y \leq n, \quad l^{+}_{xy} = l^{+}_{yx} = -\frac{n+1}{n^2} \tag{7}$$

The Clique On the other end of the connectedness spectrum lies the clique. A clique K_n of order n is a complete graph with $\frac{n(n-1)}{2}$ edges. Clearly, the clique is the densest possible graph of order n, as there is a direct edge between any pair of vertices in it. It is also the most compact graph of its order (lowest diameter). Then,

Corollary 2 *For a clique K_n of order n, $\mathbf{L}^{+}_{K_n}$ is given by:*

$$\forall x : 1 \leq x \leq n, \quad l^{+}_{xx} = \frac{n-1}{n^2} \textit{ and } \forall x \neq y : 1 \leq x, y \leq n, \quad l^{+}_{xy} = l^{+}_{yx} = -\frac{1}{n^2} \tag{8}$$

The results in the corollaries presented above are not just illustrative examples. They are also of interest from a computational point of view, particularly when the graph under study is an unweighted one. Both stars and cliques can occur as motif sub-graphs in any given graph. Indeed, for any non-trivial connected simple graph of order $n \geq 3$, there is at least one sub-graph that is a star. Selecting any vertex i with $d(i) \geq 2$, and conducting a one-hop breadth first search, generates a star sub-graph. Cliques, though not so universal, also occur in real world networks (e.g. citation networks). Therefore, in any divide-and-conquer methodology, both stars and cliques are likely to be found at some stage. We have already established that

[1] The graphs in these examples are assumed to be unweighted, i.e. all edges have a unit resistance/conductance.

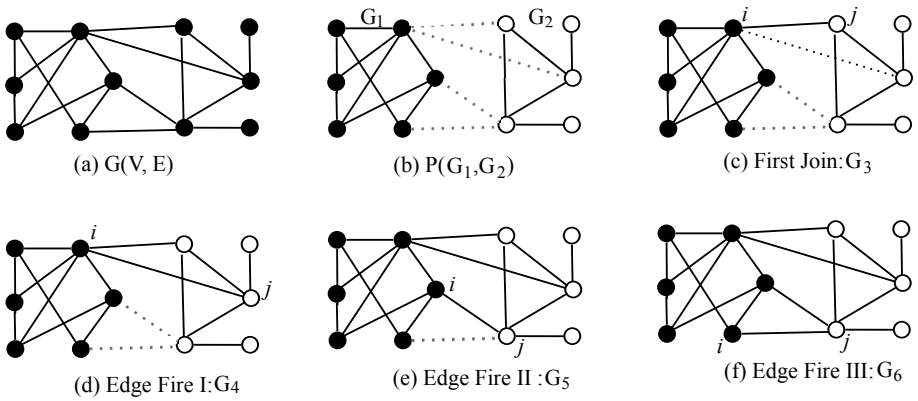


Fig. 3. Divide-and-Conquer: Connected bi-partition of a graph and the two-stage process: first join followed by three edge firings. The dotted lines represent the edges that are not part of the intermediate graph at that stage.

the cost of computing $\mathbf{L}^+_{S_n}$ and $\mathbf{L}^+_{K_n}$ is $O(1)$ (as they are determined entirely by n) and hence the solution to such a sub-problem, when found, is obtained at the lowest possible cost — a true practical gain.

To conclude, we have demonstrated that there exists a relationship between the elements of $\mathbf{L}^+$ and the pairwise effective resistances in the graph $G(V, E)$, that yields interesting closed form solutions for the pseudo-inverse for special graphs such as stars and cliques. In the subsequent sections, we demonstrate that it can be used to compute $\mathbf{L}^+$ for general graphs as well, incrementally, in a divide-and-conquer fashion.

3 From Two to One: Computing $\mathbf{L}^+$ by Partitions

In this section, we present our main result – the computation of the Moore-Penrose pseudo-inverse $\mathbf{L}^+$ of the Laplacian by means of graph bi-partitions. We first lay out a *two-stage* process — the *first join* followed by *edge firings* — that underpins our methodology. We then provide specific closed form solutions.

Note, due to space limitations, we omit the proofs from this paper. Interested readers can find the proofs for all the lemmas, theorems and corollaries in the `arXiv` version [25].

3.1 Connected Bi-Partitions of a Graph and the Two-Stage Process

In order to compute the Moore-Penrose pseudo-inverse of the Laplacian of a simple, connected, undirected and unweighted graph $G(V, E)$ by parts, we must first establish that the problem can be decomposed into two, or more, sub-problems that can be solved independently. The solutions to the independent sub-problems can then be combined to obtain the overall result. We start by introducing a few notations.

Definition 2 *Connected Bi-partition ($P = (G_1, G_2)$): A cut of the graph G which contains exactly two mutually exclusive and exhaustive connected sub-graphs G_1 and G_2.*

Fig.3(a-b), shows a graph $G(V, E)$ and a connected bi-partition $P(G_1, G_2)$, obtained from the graph $G(V, E)$ by removing the set of dotted edges shown. Each partition $P(G_1, G_2)$ has certain defining characteristics in terms of the set of vertices as well the set of edges in the graph. Let, $V_1(G_1)$ and $V_2(G_2)$ be the mutually exclusive and exhaustive subsets of $V(G)$ i.e. $V_1(G_1) \cap V_2(G_2) = \phi$ and $V_1(G_1) \cup V_2(G_2) = V(G)$. Similarly, let $E_1(G_1)$ and $E_2(G_2)$ be the sets of edges in the respective sub-graphs G_1 and G_2 of P and $E(G_1, G_2)$, the set of edges that *violate* the partition $P(G_1, G_2)$ i.e. have one end in G_1 and the other in G_2. Thus, $E_1(G_1) \cap E_2(G_2) = E_1(G_1) \cap E_1(G_1, G_2) = E(G_1, G_2) \cap E_2(G_2) = \phi$ and $E_1(G_1) \cup E(G_1, G_2) \cup E_2(G_2) = E(G)$. We denote by $\mathcal{P}(G)$ the set of all such connected bi-partitions of G.

It is easy to see that for an arbitrary connected bi-partition $P(G_1, G_2) \in \mathcal{P}(G)$ both G_1 and G_2 are themselves simple, connected, undirected and unweighted graphs. Hence, the discussion in §2 is applicable

in its entirety to the sub-graphs G_1 and G_2 independently. Note then that $\mathbf{L}^+_{G_1}$ and $\mathbf{L}^+_{G_2}$, the Moore-Penrose pseudo-inverse of the Laplacians of the sub-graphs G_1 and G_2, must, by definition, exist. The pair $\{\mathbf{L}^+_{G_1}, \mathbf{L}^+_{G_2}\}$, constitutes the solution to two independent sub-problems represented in the set $\{G_1, G_2\}$. All that remains to be shown now is that $\{\mathbf{L}^+_{G_1}, \mathbf{L}^+_{G_2}\}$ can indeed be combined to obtain $\mathbf{L}^+_G$. It is this aspect of the methodology, that we call the *two-stage* process, as explained in detail below.

The original graph $G(V, E)$ can be thought of, in some sense, as a bringing together of the disjoint spanning sub-graphs G_1 and G_2, by means of introducing the edges of the set $E(G_1, G_2)$. Starting from G_1 and G_2, we iterate over the set of edges in $E(G_1, G_2)$ in the following fashion (cf. Fig. 3 for a visual reference). Let $e_{ij} \in E(G_1, G_2) : i \in V_1(G_1), j \in V_2(G_2)$, of weight w_{ij} and resistance $\omega_{ij} = w_{ij}^{-1}$ *ohm*, be an arbitrary edge chosen during the first iteration as shown in Fig. 3(c). We call this step the *first join* in our two-stage process, whereafter G_1 and G_2 come together to give an intermediate connected spanning sub-graph (say $G_3(V_3, E_3)$). The first join represents a point of singularity in the reconstruction process, particularly from the perspective of the effective resistance distance. Note that before the first join, the effective resistance distance between an arbitrary pair of nodes $(x, y) \in V(G) \times V(G)$ is infinity, if $x \in V_1(G_1)$ and $y \in V_2(G_2)$, as there is no path connecting x and y. However, once the first edge e_{ij} has been introduced during the first join, this discrepancy no longer exists and all pairwise effective resistances are finite. Precisely, if $\Omega^{G_1} : V_1(G_1) \times V_1(G_1) \to \Re^+$ and $\Omega^{G_2} : V_2(G_2) \times V_2(G_2) \to \Re^+$, be the pairwise effective resistances defined over the sub-graphs G_1 and G_2, the following holds: if $x, y \in V_1(G_1)$, $\Omega^{G_3}_{xy} = \Omega^{G_1}_{xy}$; if $x, y \in V_2(G_2)$, $\Omega^{G_3}_{xy} = \Omega^{G_2}_{xy}$; and if $x \in V_1(G_1)$ & $y \in V_2(G_2)$, $\Omega^{G_1}_{xi} + \omega_{ij} + \Omega^{G_2}_{jy}$. Needless to say, this is a critical step in the process as we need finite values of effective resistances in order to exploit the result in Lemma 1. Hereafter, we can combine the solutions to the independent sub-problems, i.e. $\mathbf{L}^+_{G_1}$ and $\mathbf{L}^+_{G_2}$, to obtain $\mathbf{L}^+_{G_3}$. Indeed, we obtain an elegant scalar form with interesting properties (details in subsequent sections).

Following the first join, the remaining edges in $E(G_1, G_2)$, can now be introduced one at a time to obtain a sequence of intermediate graphs $(G_4 \to G_5 \to G_6)$ which finally ends in $G(V, E)$ (cf. Fig. 3(d-f)). We call this second stage of edge introductions, following the first join, *edge firing*. In terms of effective resistances, each edge firing simply creates parallel resistive connections, or alternative paths, in the graph. Algebraically, each edge firing is a $rank(1)$ perturbation of the Laplacian for the intermediate graph from the previous step. Thus, the Moore-Penrose pseudo-inverse of the Laplacians for the intermediate graph sequence $(G_4 \to G_5 \to G_6)$ can be obtained starting from $\mathbf{L}^+_{G_3}$ using standard perturbation methods [20] (details in subsequent sections).

To summarize, therefore, during the two-stage process we obtain a sequence of connected spanning sub-graphs of $G(V, E)$ starting from a partition $P(G_1, G_2) \in \mathcal{P}(G)$, performing the first join by arbitrarily selecting an edge $e_{ij} \in E(G_1, G_2)$, and then firing the remaining edges, one after the other, in any arbitrary order. The number of connected spanning sub-graphs of $G(V, E)$ constructed during the two-stage process is exactly $|E(G_1, G_2)|$ (= 4 for the example in Fig. 3). Note that, the order in which these sub-graphs are generated, is of no consequence whatsoever. Next, we use these insights to obtain $\mathbf{L}^+$ for the intermediate graphs in the sequence.

3.2 The Two-Stage Process and $\mathbf{L}^+$

We now present the closed form solutions for the Moore-Penrose pseudo-inverse of the Laplacians of the set of intermediate graphs obtained during the two-stage process.

The First Join Given, two simple, connected, undirected graphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ let $\mathbf{L}^+_{G_1}$ and $\mathbf{L}^+_{G_2}$, be the respective Moore-Penrose pseudo-inverses of their Laplacians. Also, let $n_1 = |V_1(G_1)|$ and $n_2 = |V_2(G_2)|$ be the orders of the two graphs. We denote by $l^{+(1)}_{xy}$ and $l^{+(2)}_{xy}$ respectively the general terms of the matrices $\mathbf{L}^+_{G_1}$ and $\mathbf{L}^+_{G_2}$. Next, let the *first join* between G_1 and G_2 be performed by introducing an edge e_{ij} between the graphs G_1 and G_2 to obtain $G_3(V_3, E_3)$; where $i \in V_1(G_1)$ and $j \in V_2(G_2)$. Clearly, $V_3(G_3) = V_1(G_1) \cup V_2(G_2)$ and $E_3(G_3) = E_1(G_1) \cup \{e_{ij}\} \cup E_2(G_2)$. Thus, $|V_3(G_3)| = n_3 = n_1 + n_2$ and $E_3(G_3) = m_3 = m_1 + 1 + m_2$. By convention, the vertices in $V_3(G_3)$ are labeled in the following order:

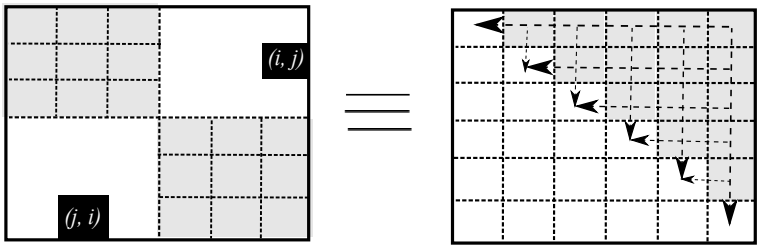


Fig. 4. The First Join: Scalar mapping $\left(\mathbf{L}^+_{G_1},\ \mathbf{L}^+_{G_2}\right)$ to $\mathbf{L}^+_{G_3}$. The grey blocs represent relevant elements in $\mathbf{L}^+_{G_1}$, $\mathbf{L}^+_{G_2}$ and $\mathbf{L}^+_{G_3}$. Arrows span the elements of the upper triangular of $\mathbf{L}^+_{G_3}$ that contribute to the respective diagonal element pointed to by the arrow head: $l^{+(3)}_{kk} = -\left(\sum_{i=1}^{k-1} l^{+(3)}_{ik} + \sum_{j=k+1}^{n} l^{+(3)}_{kj}\right)$.

the first n_1 vertices $\{1, 2, ..., n_1\}$ are retained, *as is*, from $V_1(G_1)$ and the remaining n_2 vertices are labelled $\{n_1+1, n_1+2, ..., n_1+n_2\}$ in order from $V_2(G_2)$. We denote by $\mathbf{L}^+_{G_3}$ the pseudo-inverse and $l^{+(3)}_{xy}$ its general term. Then,

Theorem 2 $\forall (x, y) \in V_3(G_3) \times V_3(G_3)$,

$$
\begin{aligned}
l^{+(3)}_{xy} &= l^{+(1)}_{xy} - \frac{n_2 n_3 \left(l^{+(1)}_{xi} + l^{+(1)}_{iy}\right) - n_2^2 \left(l^{+(1)}_{ii} + l^{+(2)}_{jj} + \omega_{ij}\right)}{n_3^2}, && if\ x, y \in V_1(G_1) \\
&= l^{+(2)}_{xy} - \frac{n_1 n_3 \left(l^{+(2)}_{xj} + l^{+(2)}_{jy}\right) - n_1^2 \left(l^{+(1)}_{ii} + l^{+(2)}_{jj} + \omega_{ij}\right)}{n_3^2}, && if\ x, y \in V_2(G_2) \\
&= \frac{n_3 \left(n_1 l^{+(1)}_{xi} + n_2 l^{+(2)}_{jy}\right) - n_1 n_2 \left(l^{+(1)}_{ii} + l^{+(2)}_{jj} + \omega_{ij}\right)}{n_3^2}, && if\ x \in V_1(G_1)\ \ \&\ \ y \in V_2(G_2)
\end{aligned}
$$

The result in Theorem 2 is interesting for several reasons. First and foremost, it clearly shows that the general term of $\mathbf{L}^+_{G_3}$, is a linear combination of the elements of $\mathbf{L}^+_{G_1}$ and $\mathbf{L}^+_{G_2}$. This was indeed our principal claim. Secondly, $\forall (x, y) \in V_3(G_3) \times V_3(G_3)$, each individual $l^{+(3)}_{xy}$ can be computed independent of the others (barring symmetry, i.e. $l^{+(3)}_{xy} = l^{+(3)}_{yx}$, which we shall discuss shortly). They are determined entirely by the specific elements from the i^{th} and j^{th} columns of the matrices $\mathbf{L}^+_{G_1}$ and $\mathbf{L}^+_{G_2}$, depending upon the membership of x and y in the disjoint graphs. This implies that all $l^{+(3)}_{xy}$ can be computed in parallel, as long as we have the relevant elements of $\mathbf{L}^+_{G_1}$ and $\mathbf{L}^+_{G_2}$.

From a cost point of view, the first join requires $O(1)$ computations per element in $\mathbf{L}^+_{G_3}$ — constant number of $\{+,\ -,\ \times,\ /\}$ operations — if $\{\mathbf{L}^+_{G_1},\ \mathbf{L}^+_{G_2}\}$ is given *a priori*. The common term in the numerator, i.e. $(l^{+(1)}_{ii} + l^{+(2)}_{jj} + \omega_{ij})$, is an invariant for the elements of $\mathbf{L}^+_{G_3}$ and need only be computed once. This term is simply a linear multiple of the change in trace:

$$\Delta(Tr) = Tr(\mathbf{L}^+_{G_3}) - \left[Tr(\mathbf{L}^+_{G_1}) + Tr(\mathbf{L}^+_{G_2})\right] \tag{9}$$

For details see the proof of Lemma 2 in [25]. Therefore, we achieve an overall cost of $O(n_3^2)$ for the first join. Last but not the least, we need to compute and store only the upper triangular of $\mathbf{L}^+_{G_3}$. Owing to the symmetry of $\mathbf{L}^+_{G_3}$, the lower triangular is determined automatically. As for the diagonal elements, they come without any additional cost as a result of $\mathbf{L}^+_{G_3}$ being doubly-centered (cf. Fig. 4).

Firing an Edge We now look at the second stage that of *firing an edge* in a connected graph. Given a simple, connected, undirected graph $G_1(V_1, E_1)$, let $e_{ij} \notin E_1(G_1)$ be *fired* to obtain $G_2(V_2, E_2)$. Clearly, $V_2(G_2) = V_1(G_1)$ and $E_2(G_2) = E_1(G_1) \cup \{e_{ij}\}$. Continuing with our convention, we denote by $\mathbf{L}^+_{G_1}$ and $\mathbf{L}^+_{G_2}$ the Moore-Penrose pseudo-inverses of the respective Laplacians. Then,

Theorem 3 *$\forall (x, y) \in V_2(G_2) \times V_2(G_2)$,*

$$l^{+(2)}_{xy} = l^{+(1)}_{xy} - \frac{\left(l^{+(1)}_{xi} - l^{+(1)}_{xj}\right)\left(l^{+(1)}_{iy} - l^{+(1)}_{jy}\right)}{\omega_{ij} + \Omega^{G_1}_{ij}} \tag{10}$$

where $\Omega^{G_1}_{ij}$ is the effective resistance distance between nodes i and j in the graph $G_1(V_1, E_1)$ — an invariant $\forall (x, y) \in V_3(G_3) \times V_3(G_3)$ that is determined entirely by the end-points of the edge e_{ij} being fired. Once again, we observe that the general term of $\mathbf{L}^+_{G_2}$ is a linear combination of the elements of $\mathbf{L}^+_{G_1}$ and requires $O(1)$ computations per element in $\mathbf{L}^+_{G_2}$ — constant number of $\{+,\ -,\ \times,\ /\}$ operations — if $\mathbf{L}^+_{G_1}$ is given *a priori*. The rest of the discussion from the preceding sub-section on first join — element-wise independence and upper triangular sufficiency — holds *as is* for this stage too. However, before concluding this section, we extend the result of Theorem 3 to the pairwise effective resistances themselves in the following corollary.

Corollary 3 *$\forall (x, y) \in V_2(G_2) \times V_2(G_2)$,*

$$\Omega^{G_2}_{xy} = \Omega^{G_1}_{xy} - \frac{\left[\left(\Omega^{G_1}_{xj} - \Omega^{G_1}_{xi}\right) - \left(\Omega^{G_1}_{jy} - \Omega^{G_1}_{iy}\right)\right]^2}{4(\omega_{ij} + \Omega^{G_1}_{ij})} \tag{11}$$

The result above is interesting in its own right. Note that computing Ω^{G_2} when the edge density of a graph increases (or the expected commute times in random walks between nodes), is pertinent to many application scenarios [7,9,14,19,26]. Corollary 3 gives us a way of computing these distances directly without having to compute $\mathbf{L}^+_{G_2}$ first.

To conclude, therefore, we have established that the Moore-Penrose pseudo-inverses of the Laplacians of all the intermediate graphs, generated during the two-stage process, are incrementally computable from the solutions at the preceding stage, on an element-to-element basis. We shall return to specific applications of these results to dynamic (time-evolving) graphs and large graphs in general, in a subsequent section. But first, for the sake of completeness, we present the case of structural regress.

4 From One to Two: A Case of Regress

We now present analogous results in the opposite direction, that of structural regress of a graph through successive deletion of edges until the graph breaks into two. These results, similar in essence to those presented in the preceding section, are particularly significant with respect to dynamically evolving graphs that change with time (e.g. social networks). Once again, we have two cases to address with respect to edge deletions *viz.* (a) *Non-bridge edge*: an edge that upon deletion does not affect the connectedness of the graph (cf. §4.1); and (b) *Bridge-edge*: an edge that, when deleted, yields a connected bi-partition of the graph (cf. §4.2).

4.1 Deleting a Non-Bridge Edge

Given a simple, connected, undirected graph $G_1(V_1, E_1)$, let $e_{ij} \in E_1(G_1)$ be a *non-bridge* edge that is deleted to obtain $G_2(V_2, E_2)$. Clearly, $V_2(G_2) = V_1(G_1)$ and $E_2(G_2) = E_1(G_1) - \{e_{ij}\}$. Once again, we denote by $\mathbf{L}^+_{G_1}$ and $\mathbf{L}^+_{G_2}$ the Moore-Penrose pseudo-inverses of the respective Laplacians. Then,

Theorem 4 *$\forall (x, y) \in V_2(G_2) \times V_2(G_2)$,*

$$l^{+(2)}_{xy} = l^{+(1)}_{xy} + \frac{\left(l^{+(1)}_{xi} - l^{+(1)}_{xj}\right)\left(l^{+(1)}_{iy} - l^{+(1)}_{jy}\right)}{\omega_{ij} - \Omega^{G_1}_{ij}} \tag{12}$$

Note, as e_{ij} is a non-bridge edge, $\Omega_{ij}^{G_1} \neq 1$. In fact, given that $G_1(V_1, E_1)$ is connected, undirected and unweighted, we have: $0 < \Omega_{ij}^{G_1} < 1$. Also, as in the case of the two-stage process, we observe the same element-wise independence for $\mathbf{L}_{G_2}^+$ here as well. Once again, if the quantity of interest is Ω^{G_2} or pairwise expected commute times in random walks, we can simply use the following corollary.

Corollary 4 $\forall (x, y) \in V_2(G_2) \times V_2(G_2)$,

$$\Omega_{xy}^{G_2} = \Omega_{xy}^{G_1} + \frac{\left[\left(\Omega_{xj}^{G_1} - \Omega_{xi}^{G_1}\right) - \left(\Omega_{jy}^{G_1} - \Omega_{iy}^{G_1}\right)\right]^2}{4(\omega_{ij} - \Omega_{ij}^{G_1})} \tag{13}$$

4.2 Deleting a Bridge Edge

Finally, we deal with the case when a bridge edge is deleted from a graph, thus rendering it disconnected for the first time. This represents the point of singularity in the case of structural regress (analogous to the first join). Continuing with our convention, let $G_1(V_1, E_1)$ be a simple, connected, undirected graph with a bridge edge $e_{ij} \in E_1(G_1)$. Upon deleting e_{ij}, we obtain $G_2(V_2, E_2)$ and $G_3(V_3, E_3)$, two disjoint spanning sub-graphs of G_1. The orders of G_1, G_2 and G_3 are respectively given by n_1, n_2 and n_3, while $\mathbf{L}_{G_1}^+, \mathbf{L}_{G_2}^+$ and $\mathbf{L}_{G_3}^+$ are the respective pseudo-inverse matrices of their Laplacians. It is easy to see that $\Omega_{xy}^{G_1} = \Omega_{xy}^{G_2}$, if $x, y \in V_2(G_2)$, $\Omega_{xy}^{G_1} = \Omega_{xy}^{G_3}$, if $x, y \in V_3(G_3)$ and $\Omega_{xy}^{G_2 \times G_3} = \Omega_{xy}^{G_3 \times G_2} = \infty$, as G_1 and G_2 are disjoint. To obtain $\mathbf{L}_{G_2}^+$ and $\mathbf{L}_{G_3}^+$ from $\mathbf{L}_{G_1}^+$, we use the result in Lemma 1.

Theorem 5 $\forall (x, y) \in V_2(G_2) \times V_2(G_2)$ *and* $\forall (u, v) \in V_3(G_3) \times V_3(G_3)$,

$$l_{xy}^{+(2)} = l_{xy}^{+(1)} - \frac{n_2 \sum_{z \in V_2(G_2)} \left(l_{xz}^{+(1)} + l_{zy}^{+(1)}\right) - \sum_{x \in V_2(G_2)} \sum_{y \in V_2(G_2)} l_{xy}^{+(1)}}{n_2^2} \tag{14}$$

$$l_{uv}^{+(3)} = l_{uv}^{+(1)} - \frac{n_3 \sum_{w \in V_3(G_3)} \left(l_{uw}^{+(1)} + l_{wv}^{+(1)}\right) - \sum_{u \in V_3(G_3)} \sum_{v \in V_3(G_3)} l_{uv}^{+(1)}}{n_3^2} \tag{15}$$

Note also that $\mathbf{L}_{G_2}^+ \in \Re^{n_2 \times n_2}$ and $\mathbf{L}_{G_2}^+ \in \Re^{n_3 \times n_3}$. For convenience, and without loss of generality, we assume that the rows and columns of $\mathbf{L}_{G_1}^+ \in \Re^{n_1 \times n_1}$ have been pre-arranged in such a way that the first $(n_2 \times n_2)$ sub-matrix (upper-left) maps to the sub-graph G_2 and similarly the lower-right $(n_3 \times n_3)$ sub-matrix to G_3.

5 Bringing It Together: Algorithm, Complexity and Parallelization

The results obtained in §3 and §4 are summarized in Table 1. In this section, we bring together these results to bear on two important scenarios: (a) dynamic (time-evolving) graphs (cf. §5.1), and (b) real-world networks of large orders (cf. §5.2). In each case, we discuss the time complexity and parallelizability of our approach in detail.

5.1 Dynamic Graphs: Incremental Computation for Incremental Change

Dynamic graphs are often used to represent temporally changing systems. The most intuitively accessible example of such a system is an online social network (OSN). An OSN evolve not only in terms of order, through introduction and attrition of users with time, but also in terms of the social ties (or relationships) between the users as new associations are formed, and older ones may fade off. Mathematically, we model an OSN as a dynamic graph $G_\tau(V_\tau, E_\tau)$ where the sub-index τ denotes the time parameter. We now study a widely used model for dynamic, temporally evolving, graphs called *preferential attachment* [3].

Operation	Ω	$\mathbf{L}^+$
First Join	$x,y \in G_1 : \Omega^{G_3}_{xy} = \Omega^{G_1}_{xy}$ $x,y \in G_2 : \Omega^{G_3}_{xy} = \Omega^{G_2}_{xy}$ $x \in G_1, y \in G_2 : \Omega^{G_3}_{xy} = \Omega^{G_1}_{xi} + \omega_{ij} + \Omega^{G_1}_{jy}$	$l^{+(1)}_{xy} - \frac{n_2 n_3 \left(l^{+(1)}_{xi} + l^{+(1)}_{iy}\right) - n_2^2 \left(l^{+(1)}_{ii} + l^{+(2)}_{jj} + \omega_{ij}\right)}{n_3^2}$ $l^{+(2)}_{xy} - \frac{n_1 n_3 \left(l^{+(2)}_{xj} + l^{+(2)}_{jy}\right) - n_1^2 \left(l^{+(1)}_{ii} + l^{+(2)}_{jj} + \omega_{ij}\right)}{n_3^2}$ $\frac{n_3 \left(n_1 l^{+(1)}_{xi} + n_2 l^{+(2)}_{jy}\right) - n_1 n_2 \left(l^{+(1)}_{ii} + l^{+(2)}_{jj} + \omega_{ij}\right)}{n_3^2}$
Edge firing	$\Omega^{G_1}_{xy} - \frac{\left[\left(\Omega^{G_1}_{xj} - \Omega^{G_1}_{xi}\right) - \left(\Omega^{G_1}_{jy} - \Omega^{G_1}_{iy}\right)\right]^2}{4(\omega_{ij} + \Omega^{G_1}_{ij})}$	$l^{+(1)}_{xy} - \frac{\left(l^{+(1)}_{xi} - l^{+(1)}_{xj}\right)\left(l^{+(1)}_{iy} - l^{+(1)}_{jy}\right)}{\omega_{ij} + \Omega^{G_1}_{ij}}$
Non-bridge delete	$\Omega^{G_1}_{xy} + \frac{\left[\left(\Omega^{G_1}_{xj} - \Omega^{G_1}_{xi}\right) - \left(\Omega^{G_1}_{jy} - \Omega^{G_1}_{iy}\right)\right]^2}{4(\omega_{ij} - \Omega^{G_1}_{ij})}$	$l^{+(1)}_{xy} + \frac{\left(l^{+(1)}_{xi} - l^{+(1)}_{xj}\right)\left(l^{+(1)}_{iy} - l^{+(1)}_{jy}\right)}{\omega_{ij} - \Omega^{G_1}_{ij}}$
Bridge delete	$x,y \in G_k : \Omega^{G_k}_{xy} = \Omega^{G_1}_{xy}$	$l^{+(1)}_{xy} - \frac{n_k \sum_{z \in G_k} \left(l^{+(1)}_{xz} + l^{+(1)}_{zy}\right) - \sum_{x \in G_k} \sum_{y \in G_k} l^{+(1)}_{xy}}{n_k^2}$

Table 1. Summary of results: Atomic operations of the divide-and-conquer methodology.

The preferential attachment model is a parametric model for network growth determined by parameters (n, κ) such that n is the desired order of the network and κ is the desired average degree per node. In its simplest form, the model proceeds in discrete time steps whereby at each time instant $1 < \tau + 1 \leq n$, a new node $v_{\tau+1}$ is introduced in the network with κ edges. This incoming node $v_{\tau+1}$, gets attached to a node $v_i : 1 \leq i \leq \tau$, through exactly one of its κ edges, with the following probability:

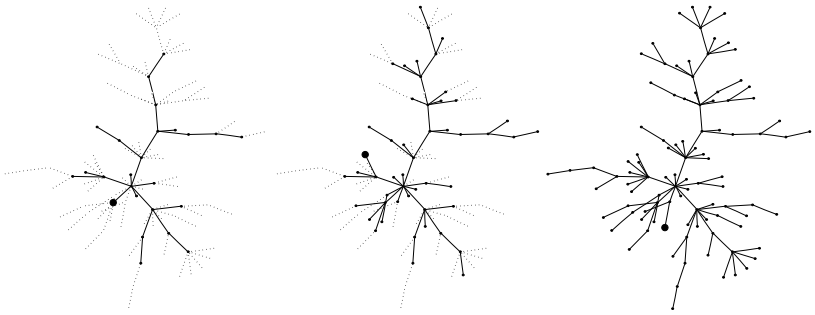
$$P_{\tau+1}(v_i) = \frac{d_\tau(i)}{\sum_{j=1}^{\tau} d_\tau(j)} \tag{16}$$

where $d_\tau(i)$ is the degree of node v_i at time τ. The end-points of all the edges emanating from $v_{\tau+1}$ are selected in a similar fashion. At the end of time step $\tau + 1$, we obtain $G_{\tau+1}(V_{\tau+1}, E_{\tau+1})$, and the process continues until we have a graph $G_n(V_n, E_n)$ of order n.[2]

Simplistic though it may seem, this model has been shown to account for several characteristics observed in real-world networks, including the *power law* degree distributions, the *small-world* characteristics and the logarithmic growth of network diameter with time. We return to these in detail in the next sub-section while dealing with the more general case. It is easy to see that in order to study the structural evolution of dynamic networks, particularly in terms of the sub-structures like spanning trees and forests [16], or centralities of nodes and edges [21,24]; or voltage distributions in growing conducting networks [28], we require not only the final state $G_n(V_n, E_n)$, but all the intermediate states of the network. In other words, we need to compute the pseudo-inverses of the Laplacians for all the graphs in the sequence $(G_1 \to G_2 \to ... \to G_n)$. Clearly, if the standard methods are used, the cost at time step τ is $O(\tau^3)$. The overall asymptotic cost for the entire sequence is then $O\left(\sum_{\tau=1}^{n} \tau^3 = \left[\frac{n(n+1)}{2}\right]^2\right)$.

In contrast, using our incremental approach, we can accomplish this at a much lower computational cost. Note that in the case of growing networks, we do not need an explicit divide operation at all. The two sub-problems at time step $\tau+1$ are given *a priori*. We have, $G_\tau(V_\tau, E_\tau)$ and a singleton vertex graph $\{v_{\tau+1}\}$ as a pair of disjoint sub-graphs. The κ edges emanating from $\{v_{\tau+1}\}$ have end-points in G_τ as determined by (16). The conquer operation is then performed through a first join between the singleton node $\{v_{\tau+1}\}$ and the graph $G_\tau(V_\tau, E_\tau)$. We can assume that $\mathbf{L}^+_{G_\tau}$ is already known at this time step (the induction hypothesis). Also, $\mathbf{L}^+_{\{v_{\tau+1}\}} = [0]$ and $n_2 = 1$ during the first join. Substituting in Theorem 2 we obtain the desired results. The rest of the $\kappa - 1$ edges are accounted for by edge firings (cf. the discussion in §3). Therefore, we need

[2] In practice, for $\kappa > 1$, the process starts with a small connected network as a base to facilitate probabilistic selection of neighbors for an incoming node. For $\kappa = 1$, we may start with a singleton node, and the resulting structure is a tree.



(a) $\tau = 25$ (b) $\tau = 50$ $\tau = 100$

Fig. 5. Growing a tree by preferential attachment ($n = 100,\ \kappa = 1$). The node v_τ, being added to the tree at time step τ, is emphasized (larger circle). Dotted edges at time steps $\tau = \{25,\ 50\}$ are a visual aid representing edges that are yet to be added in the tree until the order-limit ($n = 100$) is attained.

$G(V,E)$	$n = \|V(G)\|$	$m = \|E(G)\|$	Leaves	Cut-off	# Comp.	$\|V(GCC)\|$	$\|E(GCC)\|$	# Cut-Edges
Epinions	75,888	405,740	35,763	4,429	30,376	37,924	61,482	102,452
SlashDot	82,168	504, 230	28,499	7,012	36,311	41,084	62,225	164,719

Table 2. Basic properties: Epinions and SlashDot networks.

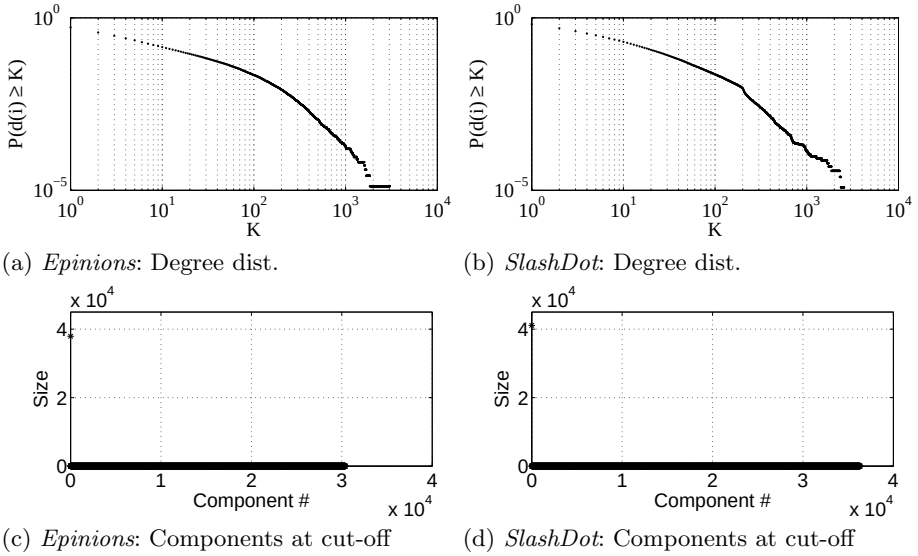
only $O(\kappa \cdot \tau^2)$ computations at time step τ, and hence $O\left(\kappa \cdot \sum_{\tau=1}^{n} \tau^2 = \kappa \cdot \frac{n(n+1)(2n+1)}{6}\right)$, overall. As $\kappa << n$ in most practical cases, we have an order of magnitude lower average cost than that incurred by the standard methods. Further improvements follow from the parallelizability of our approach. Although we have not discussed it explicitly, it is evident that node and edge deletions can all be handled within this framework in the same way and at the same $O(n^2)$ cost per operation (cf. the discussion in §4).

5.2 Large Real-World Networks: A Divide-And-Conquer Approach

In order to compute $\mathbf{L}^+$ for an arbitrary graph $G(V, E)$, in a divide-and-conquer fashion, we need to first determine independent sub-graphs of G in an efficient manner. Theoretically, an optimal divide step entails determining a *balanced* connected bi-partition $P(G_1, G_2)$ of the graph G such that $|V(G_1)| \approx |V(G_2)|$ and $|E(G_1, G_2)|$, the number of edges violating the partition, is minimized. Such balanced bi-partitioning of the graph, if feasible, can then be repeated recursively until we obtain sub-graphs of relatively small orders. The solutions to these sub-problems can then be computed and the recursion unwinds to yield the final result, using our two-stage methodology in the respective conquer steps. Alas, computing such balanced bi-partitions, along with the condition of minimality of $|E(G_1, G_2)|$, belongs to the class of *NP-Complete* problems [27], and hence a polynomial time solution does not exist. We therefore need an efficient alternative to accomplish the task at hand that works well on large *real-world* networks.

Real-world networks, and particularly online social networks, have been shown to have several interesting structural properties: edge sparsity, power-law scale-free degree distributions, existence of the so called *rich club connectivity*, small-world characteristics [30] with relatively small diameters ($O(log\ n)$). Collectively, these properties amount to a simple fact: the overall connectivity between arbitrary node pairs is dependent on higher degree nodes in the network. Based on these insights, we now study two real-world online social networks — the *Epinions* and *SlashDot* networks [1] — to attain our objective of a quick and easy divide step. Table 2 gives some of the basic statistics about the two networks.[3] It is easy to see that the networks

[3] Although the networks originally have uni-directional and bi-directional links, we symmetrize the uni-directional edges to make the graphs undirected.



(a) *Epinions*: Degree dist. (b) *SlashDot*: Degree dist.

(c) *Epinions*: Components at cut-off (d) *SlashDot*: Components at cut-off

Fig. 6. Structural regress: Epinions and SlashDot Networks.

are sparse as $m = O(n) << O(n^2)$ in both cases. Moreover, note that a significant fraction of nodes in the graphs are leaf/pendant nodes, i.e. nodes of degree 1 ($\approx 47\%$ for Epinions and $\approx 34\%$ for *SlashDot*). From Fig. 6 (a-b), it is also evident that the node degrees indeed follow a heavy tail distribution in both cases. Thus, there are many nodes of very small degree (e.g. leaves) and relatively fewer nodes of very high degrees in these networks. Therefore, in order to break the graph into smaller sub-graphs, we adopt an incremental regress methodology of deleting high degree nodes. Ordering the nodes in decreasing order by degree, we remove them one at a time. This process divides the set of nodes into three parts at each stage:

a. **The Rich Club**: High degree nodes that have been deleted until that stage.
b. **The Giant Connected Component** (GCC): The largest connected component at that stage.
c. **Others**: All nodes that are neither in the rich club nor the GCC.

We repeat the regress, one node at a time, until the size of the GCC is less than half the size of the original graph. We call this the cut-off point. We then retain the GCC as one of our sub-graphs (one independent sub-problem) and re-combine all the non-GCC nodes together with the rich club to obtain (possibly) multiple sub-graphs (other sub-problems). This concludes the divide step.

Table 2 shows the relevant statistics at the cut-off point for the two networks. Note that the cut-off point is attained at the expense of a relatively small number of high degree nodes ($\approx 5\%$ for *Epinions* and $\approx 8\%$ for *SlashDot*). Moreover, the number of nodes in the GCC is indeed roughly half of the overall order, albeit the GCC is surely sparser in terms of edge density than the overall network ($|E(GCC)|/|V(GCC)| = 1.63$ vs. $|E(G)|/|V(G)| = 5.35$ for *Epinions* and $|E(GCC)|/|V(GCC)| = 1.51$ vs. $|E(G)|/|V(G)| = 6.13$ for *Slashdot*). Fig. 6 (c-d) shows the sizes (in terms of nodes) of all the connected components for the respective graphs at the cut-off point. It is easy to see that other than the GCC, the remaining components are of negligibly small orders. Recombining the non-GCC components together (including the rich club) yields an interesting result. For the *Epinions* network, we obtain two sub-graphs of orders $37,933$ and 31 respectively while for the *Slashdot* network we obtain exactly one sub-graph of order $41,084$. This clearly demonstrates that our simple divide method, yields a roughly equal partitioning of the network — and thus comparable sub-problems — in terms of nodes. The pseudo-inverses of these sub-problems can now be computed in parallel. Albeit, as in the case of all tradeoffs, this equitable split comes at a price of roughly $\kappa = O(n)$ edges that violate the cut (cf. Table 2). This yields an $O(n^3)$ average cost for the *two-stage* process (cf. §3). However, given the element-wise parallelizability of our method, we obtain the pseudo-inverses in acceptable times of roughly 15 minutes for the *Epinions* and 18 minutes for the *SlashDot* networks.

6 Related Work

We now provide a brief review of the related work, highlighting in particular a few instances of the applications of the Moore-Penrose pseudo-inverse and the sub-matrix inverses of the Laplacian for a graph. As alluded to earlier, $\mathbf{L}^+$ is used to compute effective resistance distances between the nodes of a graph [17] as well as the one way hitting and commute times in random walks between node pairs in a graph. All these distances serve as measures of *multi-hop* dissimilarity between nodes and find applications in several graph mining contexts [9,14]. Moreover, for every connected undirected graph there is an analogous *reversible* Markov chain, $\mathbf{L}^+$ finds use in the computation of relevant metrics (such as hitting time, cover time and mixing rates). $\mathbf{L}^+$ is a gram-matrix. Its eigen decomposition yields an $n-dimensional$ Euclidean embedding of the graph whereby each node in the graph is represented as a point in that space. The general term l^+_{xy} represents the inner product of the respective position vectors for the nodes x and y and thus $\mathbf{L}^+$ is a valid kernel for a graph. This geometric interpretation has been used in collaborative recommendation systems [14].

In [11,32] the elements of $\mathbf{L}^+$ have been given an elegant topological interpretation in terms of the *dense* spanning rooted forests and *connected bi-partitions* of the graph. In [24] the authors provide multiple interpretations of the diagonal elements of $\mathbf{L}^+$ and show that they reflect both the overall positions and connectivity of nodes in a complex network. They therefore refer to L^+_{ii} as the *topological centrality*, which provides a measure of the role of a node in the overall connectivity of a network and its robustness to random multiple failures in the network that breaks it down into two connected parts [32]. By extension, $Tr(\mathbf{L}^+)$, also called the Kirchhoff index of a graph [17,31], is a global structural descriptor for the graph on a whole. This index is quite popular in the field of mathematical chemistry and is used to measure overall molecular strength (see, e.g., [22]). The elements of sub-matrix inverses have analogous interpretations in terms of *unrooted* spanning forests of the graph [16]. In [21], the sub-matrix inverses have been used to compute the random-walk betweenness centrality, another useful index to characterize roles of nodes in a network.

7 Conclusion and Future Work

In this work, we presented a divide-and-conquer based approach for computing the Moore-Penrose pseudo-inverse of the Laplacian ($\mathbf{L}^+$) for a simple, connected, undirected graph. Our method relies on an elegant interplay between the elements of $\mathbf{L}^+$ and the pairwise effective resistance distances in the graph. Exploiting this relationship, we derived closed form solutions that enable us to compute $\mathbf{L}^+$ in an incremental fashion. We also extended these results to analogous cases for structural regress. Using dynamic networks and online social networks as examples, we demonstrated the efficacy of our method for computing the pseudo-inverse relatively faster than the standard methods. The insights from our work open up several interesting questions for future research. First and foremost, similar explorations can be done for the case of directed graphs (asymmetric relationships), where analogous distance functions — such as the expected commute time in random walks — are defined, albeit the Laplacians (more than one kind in literature) are no longer symmetric [6]. Secondly, matrix-distance interplays of the kind exploited in this work, also exist for a general case of the so called *forest matrix* and its distance counterpart the *forest distance* [2,10], both for undirected and directed graphs. The results presented here should find natural extensions to the forest matrix and the forest metric, at least for the undirected case. Finally, our closed forms can be used in conjunction with several interesting approaches for sparse inverse computations [8], to further expedite the pseudo-inverse computation for large generalized graphs. All these motivate ample scope for future work.

8 Acknowledgment

This research was supported in part by DTRA grant HDTRA1-09-1-0050, DoD ARO MURI Award W911NF-12-1-0385, and NSF grants IIS-0916750, CNS-10171647, CNS-1017092, CNS-1117536, IIS-1319749 and CRI-1305237.

References

1. http://snap.stanford.edu/data/.
2. R. Agaev and P. Chebotarev. The matrix of maximum out forests of a digraph and its applications. *Automation and Remote Control*, 61(9):1424–1450, 2000.
3. A. L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.
4. A. Ben-Israel and T. Greville. *Generalized Inverses: Theory and Applications,* 2^{nd} *edition*. Springer-Verlag, 2003.
5. N. Biggs. *Algebraic Graph Theory*. Cambridge University Press, 1993.
6. D. Boley, G. Ranjan, and Z.-L. Zhang. Commute times for a directed graph using an asymmetric laplacian. *Linear Algebra and its Applications*, 435(2):224–242, 2011.
7. M. Brand. A random walks perspective on maximizing satisfaction and profit. In *Proc. 2005 SIAM Int'l Conf. Data Mining*, 2005.
8. Y. E. Campbell and T. A. Davis. Computing the sparse inverse subset: An inverse multifrontal approach. *Tech. report TR-95-021, Univ. of Florida, Gainesville*, 1995.
9. A. K. Chandra, P. Raghavan, W. L. Ruzzo, R. Smolensky, and P. Tiwari. The electrical resistance of a graph captures its commute and cover times. In *Proc. of Annual ACM Symposium on Theory of Computing*, pages 574–586, 1989.
10. P. Chebotarev and E. Shamis. The matrix-forest theorem and measuring relations in small social groups. *Automation and Remote Control*, 58(9):1505–1514, 1997.
11. P. Chebotarev and E. Shamis. On proximity measures for graph vertices. *Automation and Remote Control*, 59(10):1443–1459, 1998.
12. P. G. Doyle and J. L. Snell. *Random Walks and Electric Networks*. The Math. Assoc. of America, 1984.
13. M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Math. J.*, 23:298–305, 1973.
14. F. Fouss, A. Pirotte, J. M. Renders, and M. Saerens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19, 2007.
15. D. Isaacson and R. Madsen. *Markov Chains Theory and Applications*. John Wiley and Sons, 1976.
16. S. J. Kirkland, M. Neumann, and B. L. Shader. Distances in weighted trees and group inverse of laplacian matrices. *SIAM J. Matrix Anal. Appl.*, 18:827–841, 1997.
17. D. J. Klein and M. Randić. Resistance distance. *J. Math. Chemistry*, 12:81–95, 1993.
18. U. V. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, December 2007. Max Planck Institute for Biological Cybernetics. Technical Report No. TR-149.
19. U. V. Luxburg, A. Radl, and M. Hein. Getting lost in space: Large sample analysis of the commute distance. *NIPS*, 2010.
20. C. D. Meyer. Generalized inversion of modified matrices. *SIAM Journal on Applied Mathematics*, 24(3):315–323, 1973.
21. M. E. J. Newman. A measure of betweenness centrality based on random walks. *Social Networks*, 27(1):39–54, 2005.
22. J. L. Palacios and J. M. Renom. Bounds for the kirchhoff index of regular graphs via the spectra of their random walks. *Intl. Journal of Quantum Chemistry*, 110:1637–1641, 2001.
23. G. Ranjan and Z.-L. Zhang. How to glue a robust smart-grid: A finite network theory of inter-dependent networks (extended abstract). In *Proc. of the 7th (2011) Cyber Security and Information Intelligence Research Workshop: THEME – Energy Infrastructure Cyber Protection (CSIIRW7)*, 2011.
24. G. Ranjan and Z.-L. Zhang. Geometry of complex networks and topological centrality. *Physica A: Statistical Mechanics and its Applications*, 392(17), 2013.
25. G. Ranjan, Z.-L. Zhang, and D. Boley. Incremental computation of pseudo-inverse of laplacian. Available at `http://arxiv.org/abs/1304.2300`, 2013.
26. B. Sarwar, G. Karypis, J. Konstan, , and J. Riedl. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proc. Fifth Int'l Conf. Computer and Information Technology*, 2002.
27. A. Sen, P. Ghosh, B. Yang, and V. Vittal. A new min-cut problem with application to electric power network partitioning. *European Transactions on Electrical Power*, 2008.
28. B. Tadić and V. Priezzhev. Voltage distribution in growing conduction networks. *European Physical Journal B*, 30:143–146, 2002.
29. P. Tetali. Random walks and effective resistance of networks. *Journal of Theoretical Probability*, pages 101–109, 1991.
30. D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.
31. W. Xiao and I. Gutman. Resistance distance and laplacian spectrum. *Theoretical Chemistry Accounts*, 110:284–289, 2003.
32. Z.-L. Zhang and G. Ranjan. On connected bi-partitions of a graph. manuscript under preparation, 2014.