DIFFUSION FILTERING OF GRAPH SIGNALS AND ITS USE IN RECOMMENDATION SYSTEMS

Jeremy Ma†  Weiyu Huang‡  Santiago Segarra‡  Alejandro Ribeiro‡

†School of Electrical Engineering and Telecommunications, University of New South Wales
‡Department of Electrical and Systems Engineering, University of Pennsylvania

ABSTRACT
This paper presents diffusion filtering as a method to smooth signals defined on the nodes of a graph or network. Diffusion filtering considers how the given signals as initial temperature distributions in the nodes and diffuses heat through the edges of the graph. The filtered signal is determined by the accumulated temperatures over time at each node. We show multiple other interpretations of diffusion filtering and describe how it can be generalized to encompass a wide class of networks making it suitable for real-world applications. We prove that diffused signals are stable to perturbations in the underlying network. Further, we demonstrate how diffusion filtering can be applied to improve the performance of recommendation systems by considering the problem of predicting ratings from a signal processing perspective.

Index Terms— Graph signal processing, filtering, pattern recognitions, recommendation systems.

1. INTRODUCTION
Networks and graphs can be defined as structures that encode pairwise relations between elements of a set. The simplicity of this definition drives the application of graphs and networks to a wide variety of disciplines such as biology and sociology. The emerging theory of graph signal processing is not specifically concerned with the study of networks but rather with the study of signals supported on the nodes of a graph, where the latter captures an underlying notion of proximity between the nodes. Examples of network-supported signals include gene expression patterns defined on top of gene networks and brain activity signals supported on top of brain connectivity networks.

In this paper we address the problem of smoothing out irregularities of graph signals by incorporating the structure of the underlying network. Since the network encodes similarities between nodes, we assume that the signals supported on the graph also exhibit such similarities and want the filtered signal to be such that signals on nodes with high proximities become more alike. We present the notion of diffusion filtering and argue that it inherits this functionality through its connections to heat diffusion on the graph.

The main contributions of this paper are the definition and interpretation of diffusion filtering for signals supported on networks (Section 2), its generalization to arbitrary graphs (Section 3), the proof that the diffused signals are stable to perturbations in the underlying network (Section 4), and the utility of diffusion filtering to improve the performance of recommendation systems (Section 5).

2. DIFFUSION FILTERING
Consider weighted and symmetric networks formally defined as triplets \( \mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}) \). The set \( \mathcal{V} = \{1, \ldots, n\} \) represents the \( n \) vertices, the set \( \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \) denotes the edges defined as ordered pairs \((i, j)\), and \( \mathcal{W} : \mathcal{E} \rightarrow \mathbb{R}_{++} \) is a map from the set of edges to the strictly positive reals associating a weight \( w_{ij} > 0 \) to each edge \((i, j)\). Since the graph is symmetric we have \((i, j) \in \mathcal{E}\) if and only if \((j, i) \in \mathcal{E}\) and \( w_{ij} = w_{ji} \) for all \((i, j) \in \mathcal{E}\). The edge \((i, j) \in \mathcal{E}\) symbolizes the existence of a relationship between \( i \) and \( j \) and we say that \( i \) and \( j \) are adjacent or neighboring. The set of neighboring nodes of \( i \) is denoted as \( \mathcal{N}_i \). The weight \( w_{ij} = w_{ji} \) stands for the strength of the relation, or, equivalently, the proximity or similarity between nodes \( i \) and \( j \). The networks considered here do not contain self loops, i.e., \((i, i) \notin \mathcal{E}\) for all \( i \in \mathcal{V}\).

We utilize the customary definitions of adjacency, degree, and Laplacian matrices for the weighted graph \( \mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}) \) [14, Chapter 1]. The adjacency matrix \( A \in \mathbb{R}^{n \times n} \) is defined such that \( A_{ij} = w_{ij} \) whenever \( i \) and \( j \) are adjacent, i.e., whenever \((i, j) \in \mathcal{E}\), with \( A_{ij} = 0 \) for \((i, j) \notin \mathcal{E}\). The degree matrix \( D \in \mathbb{R}^{n \times n} \) is a diagonal matrix with its \( i \)-th diagonal element \( D_{ii} = \sum_j w_{ij} \). The Laplacian matrix is defined as the difference \( L := D - A \in \mathbb{R}^{n \times n} \); thus, its elements are explicitly given by:

\[
L_{ij} = \begin{cases} 
-A_{ij}, & \text{if } i \neq j, \\
\sum_{k=1}^{n} A_{ik}, & \text{if } i = j. 
\end{cases}
\]  
(1)

Models with asymmetric weights \( w_{ij} \neq w_{ji} \) and signed edges \( w_{ij} < 0 \) are considered in Section 3.

Our interest in this paper is on signals \( x = [x_1, \ldots, x_n]^T \in \mathbb{R}^n \) whose component \( x_i \) is interpreted as a value associated with the \( i \)-th node of \( \mathcal{G} \). For a given graph with Laplacian \( L \) and given signal \( x \), introduce a constant \( \alpha > 0 \) to define the time-varying vector \( x(t) \in \mathbb{R}^n \) as the solution of the linear differential equation

\[
\frac{dx(t)}{dt} = -\alpha L x(t), \quad x(0) = x. 
\]  
(2)

The differential equation in (2) models the diffusion of heat on the graph \( \mathcal{G} \) when the weights represent thermal conductivities across the edges [10]. The given vector \( x = x(0) \) specifies the initial temperature distribution and \( x(t) \) represents the temperature distribution at time \( t \). The constant \( \alpha \) scales the thermal conductivities and controls the heat diffusion rate. The solution of (2) is given by the matrix exponential

\[
x(t) = e^{-\alpha L t} x. 
\]  
(3)

Using (3) we can compute the temperature distribution at any time point \( t \) by diffusing the initial heat profile \( x \) on the structure of the underlying network through its Laplacian \( L \). Notice that as diffusion continues, \( x(t) \) tends to a steady-state distribution, i.e., \( x(t) \rightarrow x^* \) as \( t \rightarrow \infty \), where \( x^* \) is a stationary vector. This paper presents diffusion filtering as a method to smooth signals supported on networks (Section 2), define diffusion filtering as a method to smooth signals supported on networks (Section 2), and argue that it inherits this functionality through its connections to heat diffusion on the graph.
settles to an isothermal equilibrium – temperature becomes identical on all nodes – as long as the graph is fully connected.

It is informative to write a componentwise version of (3). If we focus on the i-th entry $x_i(t)$ of $x(t)$, it follows from (1) and (2) that

$$\frac{dx_i(t)}{dt} = \sum_{j \in N_i} \alpha \ w_{ij} \ (x_j(t) - x_i(t)).$$

Since (4) describes the rate of change in $x_i(t)$, it follows that the flow of heat on an edge increases proportionally with the temperature imbalance $x_j(t) - x_i(t)$ as well as the weight $w_{ij}$. In particular, with other things being equal, nodes with larger proximity tend to equalize their temperatures faster. This means that the time varying vector $x(t)$ defines a series of temperature profiles by smoothing the initial distribution $x$ using the underlying graph structure. This allows the interpretation of $x(t)$ as a smoothed version of the given signal $x$ where variations between nodes that are strongly related are reduced.

While it is possible to focus on a particular $x(t)$ as a filtered version of $x$ it is more convenient to aggregate the whole diffusion process. A possible way of constructing a vector $y$ condensing the time varying vector $x(t)$ is to consider the discounted integral

$$y = \int_0^\infty e^{-t} x(t) \ dt = \int_0^\infty e^{-t} e^{-\alpha \ L t} x \ dt.$$ (5)

We say that the vector $y$ in (5) is the diffused or diffusion filtered signal obtained from $x$ associated with $L$. The motivation for (5) is that it considers the heat diffusion process generated by $x$ on $L$ while giving more importance to initial times. This is necessary because in the asymptotic thermal equilibrium the variability of the original signal is completely smoothed out. Further note that the primitive of the matrix exponential $e^{-t e^{-\alpha L t}} = e^{-(I + \alpha L)t}$ is the matrix $-(I + \alpha L)^{-1} e^{-(I + \alpha L)t}$ with $I$ being the identity matrix of corresponding size. This can be used to solve (5) in closed-form as we summarize in the following definition.

**Definition 1** Given a network $\mathcal{G} = (V, E, W)$ with Laplacian $L$, the diffusion rate $\alpha \geq 0$, and a graph signal $x \in \mathbb{R}^n$ defined in the node space $V$, the diffusion filtered signal $y$ obtained from $x$ is defined as

$$y = (I + \alpha \ L)^{-1} x.$$ (6)

Observe that the diffused signal $y$ is another graph signal defined in the same node space $V$ and that $y$ always exists because $I + \alpha L$ is strictly diagonally dominant [15, Thm. 6.2.27]. Besides its construction from diffusion dynamics, diffusion filtering can also be seen to follow as the solution of the optimization problem [16,17].

$$\min_{y \in \mathbb{R}^n} \|x - y\|^2 + \alpha \ y^T Ly.$$ (7)

The diffused signal is then close to $x$ but with small total variation on the graph, as indicated by the first and second summands in (7), respectively. The balance of these conflicting requirements is mediated by $\alpha$. Alternatively, we can also think of $y$ as the result of processing $x$ with a low-pass graph filter [18,20]. To explain this interpretation, consider the eigendecomposition of the Laplacian $L = \mathcal{V}\Lambda\mathcal{V}^H$ and define the graph Fourier transforms of $x$ and $y$ as $\hat{x} = \mathcal{V}^H x$ and $\hat{y} = \mathcal{V}^H y$, respectively. The form of (6) is such that

$$\hat{y}_i = \frac{1}{1 + \alpha \lambda_i} \hat{x}_i.$$ (8)

Therefore, higher frequency components of $x$, which are associated with larger eigenvalues $\lambda_i$, are attenuated by a larger coefficient. Conversely, lower frequencies are attenuated by a smaller coefficient as is expected of a (graph) low-pass filter. Moreover, the larger the diffusion rate $\alpha$ the steeper the decay of the graph filter response for high frequencies. These three interpretations – diffusion dynamics, a signal close to the original structure of spas, and becoming equal, nodes with larger proximity tend to equalize their temperature faster. This means that the time varying vector $x(t)$ defines a series of temperature profiles by smoothing the initial distribution $x$ using the underlying graph structure. This allows the interpretation of $x(t)$ as a smoothed version of the given signal $x$ where variations between nodes that are strongly related are reduced.

While it is possible to focus on a particular $x(t)$ as a filtered version of $x$ it is more convenient to aggregate the whole diffusion process. A possible way of constructing a vector $y$ condensing the time varying vector $x(t)$ is to consider the discounted integral

$$y = \int_0^\infty e^{-t} x(t) \ dt = \int_0^\infty e^{-t} e^{-\alpha L t} x \ dt.$$ (5)

We say that the vector $y$ in (5) is the diffused or diffusion filtered signal obtained from $x$ associated with $L$. The motivation for (5) is that it considers the heat diffusion process generated by $x$ on $L$ while giving more importance to initial times. This is necessary because in the asymptotic thermal equilibrium the variability of the original signal is completely smoothed out. Further note that the primitive of the matrix exponential $e^{-t e^{-\alpha L t}} = e^{-(I + \alpha L)t}$ is the matrix $-(I + \alpha L)^{-1} e^{-(I + \alpha L)t}$ with $I$ being the identity matrix of corresponding size. This can be used to solve (5) in closed-form as we summarize in the following definition.

**Definition 1** Given a network $\mathcal{G} = (V, E, W)$ with Laplacian $L$, the diffusion rate $\alpha \geq 0$, and a graph signal $x \in \mathbb{R}^n$ defined in the node space $V$, the diffusion filtered signal $y$ obtained from $x$ is defined as

$$y = (I + \alpha \ L)^{-1} x.$$ (6)

Observe that the diffused signal $y$ is another graph signal defined in the same node space $V$ and that $y$ always exists because $I + \alpha L$ is strictly diagonally dominant [15, Thm. 6.2.27]. Besides its construction from diffusion dynamics, diffusion filtering can also be seen to follow as the solution of the optimization problem [16,17].

$$\min_{y \in \mathbb{R}^n} \|x - y\|^2 + \alpha \ y^T Ly.$$ (7)

The diffused signal is then close to $x$ but with small total variation on the graph, as indicated by the first and second summands in (7), respectively. The balance of these conflicting requirements is mediated by $\alpha$. Alternatively, we can also think of $y$ as the result of processing $x$ with a low-pass graph filter [18,20]. To explain this interpretation, consider the eigendecomposition of the Laplacian $L = \mathcal{V}\Lambda\mathcal{V}^H$ and define the graph Fourier transforms of $x$ and $y$ as $\hat{x} = \mathcal{V}^H x$ and $\hat{y} = \mathcal{V}^H y$, respectively. The form of (6) is such that

$$\hat{y}_i = \frac{1}{1 + \alpha \lambda_i} \hat{x}_i.$$ (8)

Therefore, higher frequency components of $x$, which are associated with larger eigenvalues $\lambda_i$, are attenuated by a larger coefficient. Conversely, lower frequencies are attenuated by a smaller coefficient as is expected of a (graph) low-pass filter. Moreover, the larger the diffusion rate $\alpha$ the steeper the decay of the graph filter response for high frequencies. These three interpretations – diffusion dynamics, a signal close to the original

3. GENERALIZED DIFFUSION FILTERING

The diffusion filter in Section 2 is derived for symmetric networks with positive weights and is defined in terms of the graph Laplacian. In some applications, recommendation systems being an example, we have to deal with asymmetric networks with weights that can be negative and the structure of the problem justifies the use of normalized Laplacians. We discuss these extensions in this section.

**Normalized Laplacian.** The random walk normalized Laplacian is defined as $\tilde{L} := D^{-1}L$; see, e.g., [21]. In terms of an explicit representation akin to the Laplacian in (1), this definition means that values in the diagonals are $\tilde{L}_{ii} = 1$ and that the off-diagonal values are $\tilde{L}_{ij} = -w_{ij} / \sum_{j \in N_i} w_{ij}$. To define the normalized diffusion filter we just replace $L$ with $\tilde{L}$ in Definition 1. This results in the filtering relationship

$$y = (I + \alpha \tilde{L})^{-1} x = (I + \alpha D^{-1}L)^{-1} x.$$ (9)

This expression corresponds to diffusion dynamics where the conductances out of a node are normalized to sum up to one. In particular, the componentwise rate of change in node $i$ induced by its neighbors becomes [cf. (10)]

$$\frac{dx_i(t)}{dt} = \left( \sum_{j \in N_i} w_{ij} \right)^{-1} \sum_{j \in N_i} \alpha \ w_{ij} \ (x_j(t) - x_i(t)).$$ (10)

In (4), nodes with higher degree change at a faster rate. In (10), the rate of change of $x_i$ is driven by the difference with its neighbors and their relative proximity but is independent of the node’s degree. This property is of interest when the underlying graph has a heterogeneous degree distribution. In practice, we find that normalized diffusion filtering is indeed beneficial in these situations; see Section 5.

**Directed network.** In asymmetric networks we can define different Laplacians [22,23] depending on whether we focus, e.g., in incoming connections or outgoing connections. The definition that preserves the diffusion interpretation is the outgoing directed Laplacian $L_{out} := D_{out}^{-1} A$, where the diagonals of $D_{out}$ contain the out-degree of each node. The diffusion filtered signal in asymmetric graphs can then be written as

$$y = (I + \alpha L_{out})^{-1} x.$$ (11)

The componentwise relationship induced by the corresponding diffusion process is identical to (4). The difference is that the effect of node $j$ in the rate of change of the value at node $i$ is not necessarily equal to the effect of $i$ in the rate of change of the value at $j$. As in the case of symmetric networks we can defined a normalized diffusion filter by replacing $L_{out}$ with the normalized Laplacian $\tilde{L}_{out} := D_{out}^{-1} L_{out}$ in (11).

**Network with negative weights.** In a network with negative weights, the positive weights encode affinity, null weights encode indifference, and negative weights dissimilarity; see e.g., [24]. Negative values in the Laplacian cannot be allowed in Definition 1 because the associated dynamical system in (2) is unstable. To resolve our problem we define the absolute degree matrix $D$ with diagonal elements $D_{ii} = \sum_{j \in N_i} |A_{ij}|$. [25]. The corresponding absolute Laplacian is defined as $\tilde{L} := D - A$ and the diffusion filtered signal in networks with negative weights is defined as

$$y = (I + \alpha \tilde{L})^{-1} x.$$ (12)
The generative diffusion process follows from substituting \( L \) by \( \hat{L} \) in (2).

The resulting componentwise dynamics can be written as

\[
\frac{dx_i(t)}{dt} = \sum_{j \in \mathcal{N}_i} \alpha \left| w_{ij} \right| (\text{sign}(w_{ij}) x_j(t) - x_i(t)).
\]

The rate of change on node \( i \) induced by neighbor \( j \) is governed by the magnitude of the weight between them. Edges with positive weights drive neighbor signals towards each other – as in regular diffusion – whereas the value at a node is driven to the additive inverse of the values at adjacent nodes when the shared edge has a negative weight.

4. STABILITY

The diffusion filtered signal depends on the underlying network structure through the associated Laplacian. It is therefore crucial to analyze how a perturbation of the underlying network affects the filtered signal. In this section we prove that all the generalized diffusion filters defined through the associated Laplacian. It is therefore crucial to analyze how a perturbation as the matrix \( L \) for different network structures. (d) \( \| y' - y \|_2 / \| \mathbf{E} \|_2 \| x \|_2 \) for networks with negative weights. The ratio is considerably lower than the theoretical upper bound of 1.

**Proposition 1**

Given the unnormalized (normalized) Laplacian matrix \( L (\hat{L}) \) associated with any graph \( G \), potentially directed and with negative edge weights, and a diffusion rate \( \alpha \), for a bounded signal \( x \) on the network with \( \| x \|_{\infty} \leq \gamma \), if we perturb the network resulting in the Laplacian \( \hat{L} = L + E (\hat{L}' = \hat{L} + E) \) with the difference \( E \) such that \( \| E \|_{\infty} \leq \epsilon < 1 \), the difference between the diffusion signal \( y' \) on the perturbed Laplacian \( \hat{L} \) and the diffusion signal \( y \) on the original Laplacian \( L \) is bounded by

\[
\| y' - y \|_{\infty} \leq \gamma \epsilon + o(\epsilon) < \gamma + o(1).
\]

Moreover, if the original Laplacian \( L (\hat{L}) \) is symmetric, then the above statement is also true when \( \| \cdot \|_{\infty} \) is replaced by \( \| \cdot \|_1 \) or \( \| \cdot \|_2 \).

**Proof:** See [26].

**Proposition 1** presents a bound for the difference between the diffusion filtered signals computed based on an original graph and its perturbed version. This bound depends on the original energy of the signals \( \| x \|_{\infty} \) as well as on the magnitude of the perturbation \( \| E \|_{\infty} \). Since the bound in (14) contains high order terms, we may divide (14) by \( \epsilon \) and compute the limit as the perturbation vanishes

\[
\lim_{\epsilon \to 0} \frac{\| y' - y \|_{\infty}}{\epsilon} \leq \gamma,
\]

entailing that, for small perturbations, the difference in the diffused signals grows linearly. When establishing the underlying network in real-world applications, edge weights contain inherent errors. **Proposition 1** ensures that diffused signals are robust to these minor perturbations.

To illustrate stability, we construct directed networks with edge weights uniformly picked either from \([0, 1]\) or \([-1, 1]\), the latter used to generate signed networks. We perturb these networks by adding to each edge a random weight uniformly picked from \([-0.05, 0.05]\), and then consider the diffused versions of a graph signal \( x \) and also randomly generated – with respect to the perturbed graph and the original graph, for both Laplacians \( L \) and \( \hat{L} \). To verify **Proposition 1** for the 2-norm \( \| \cdot \|_2 \), we also examine diffusion on symmetric networks. In Figure 1 we plot the histogram for 1000 network perturbations of the norm of the difference in the diffused signals \( y \) and \( y' \), normalized by both the norm of the perturbation \( \epsilon \) and the norm of the graph signal \( \gamma \), for every setting analyzed. It is immediate that all perturbations are below the theoretical upper bound of 1 [cf. (15)] by a considerable margin. This stability property is of practical relevance, as we present in the next section.

5. RECOMMENDATION SYSTEMS

Rating prediction, i.e., estimating a given user rating for a given item, is at the core of every recommendation system. Novel approaches to solve rating prediction problems, traditionally solved with machine learning tools, have been proposed recently [27, 28]. In this section, we illustrate a signal processing perspective for rating prediction and show how diffusion filtering can be used to improve the performance of recommendation systems.

Usually, ratings are considered in the form of a two dimensional matrix \( \mathbf{R} = \mathbb{R}^{|\mathcal{U}| \times |\mathcal{I}|} \) for the set of users \( \mathcal{U} \) and the set of items \( \mathcal{I} \). Not every entry of \( \mathbf{R} \) is known and the objective of the problem is to predict missing entries of interest, i.e. to obtain predictions \( \hat{R}_{u,i} \), for user \( u \) and item \( i \), from known entries \( (v, j) \in \mathcal{K} \) with \( \mathcal{K} \) denoting the set of all known entries. Two main techniques have been used to solve the prediction problem: matrix completion [29] and collaborative filtering (CF) [30]. Matrix completion looks for a low rank matrix \( \hat{\mathbf{R}} \) such that \( \hat{R}_{u,i} = \hat{R}_{u,i} \) for all known entries \((u, i) \in \mathcal{K}\). In user-based CF, given a certain item \( i \), the recommendation system computes prediction \( \hat{R}_{u,i} \) as the weighted sum of the ratings \( R_{v,i} \) evaluated by different users \( v \) for the same item \( i \), i.e.

\[
\hat{R}_{u,i} = \frac{\sum_{v \in \mathcal{U}(u)} s(u, v) R_{v,i}}{\sum_{v \in \mathcal{U}(u)} s(u, v)},
\]

where \( s(u, v) \) denotes a similarity or proximity between users \( u \) and \( v \). In this paper, we consider cosine similarities

\[
s_{\text{cos}}(u, v) = \frac{\sum_{i \in \mathcal{I}} R_{u,i} R_{v,i}}{\sqrt{\sum_{i \in \mathcal{I}} R_{u,i}^2} \sqrt{\sum_{i \in \mathcal{I}} R_{v,i}^2}} = \frac{\mathbf{R}_u \mathbf{R}_v}{\| \mathbf{R}_u \|_2 \| \mathbf{R}_v \|_2}.
\]

where \( \mathbf{R}_u \in \mathbb{R}^{|\mathcal{I}|} \) is the vector of all ratings for user \( u \) and similarly for \( v \). Since \( s_{\text{cos}}(u, v) \) computes the cosine of the angle between \( \mathbf{R}_u \) and \( \mathbf{R}_v \), it is contained between –1 and 1, where larger positive values
indicate stronger similarities between $u$ and $v$ and the opposite is true for negative values. In (17), to overcome the problem of having missing values in $\mathbf{R}$, we set all missing values to 0 and adjust the values for known ratings in $\mathbf{R}_{u,i} := \hat{R}_{u,i} - (\bar{R} + \mu_{u})$, $\forall (u, i) \in \mathcal{K}$ where $\bar{R}$ is the average over all known ratings $\bar{R} = \sum_{u,i,(u,i) \in \mathcal{K}} R_{u,i}/|\mathcal{K}|$ and $\mu_{u}$ is the user bias defined as

$$
\mu_{u} = \frac{\sum_{i \in \mathcal{I} \setminus \{u,i\} \in \mathcal{K}} (R_{u,i} - \bar{R})}{|\{i \in \mathcal{I} : (u, i) \in \mathcal{K}\}|}.
$$

In the above normalization, missing values in $\mathbf{R}$ are set to the average taste of each user and known ratings are adjusted to compensate for the fact that some users are more generous than others. Besides resolving missing values, the mean centering also favors the performance of recommendation systems [20,21]. By contrast, in item-based CF, given a user $u$, the system predicts rating $\hat{R}_{u,i}$ as the weighted sum of the ratings $R_{u,j}$ evaluated by the same user $u$ for different items $j$ [cf. (16)]

$$
\hat{R}_{u,i} = \frac{\sum_{j \in \mathcal{I} \setminus \{i\} \in \mathcal{K}} s(i,j)R_{u,j}}{\sum_{j \in \mathcal{I} \setminus \{i\} \in \mathcal{K}} s(i,j)}.
$$

Again, in this paper we use cosine similarity as measure $s(i,j)$, i.e., the analogous to (17) but between items instead of users. One usually restricts attention to the $k$ users – in user-based CF – or items – in item-based CF – that are the most similar to the user or item of interest in the weighted sum. Such restrictions yield user- or item-based $k$-Nearest Neighbor (k-NN) CF.

In contrast with the conventional formulation, we give an interpretation of the problem from a signal processing perspective. For each user $u$, the rating vector $\mathbf{R}_{u}$ can be viewed as a graph signal defined on top of a common network $\mathcal{G}_{\text{item}}$ describing proximities between items. Under such setup, users $u$ and $v$ should be similar if they assign similar ratings to similar movies, even when the items rated by them do not coincide. This interpretation motivates the use of diffusion filtering to smooth rating vectors $\mathbf{R}_{u}$ before evaluating the cosine similarities. We refer to this formulation as item-based-diffusion user-based $k$-NN.

We consider 4 different ways to construct proximity networks between movies to encode similarities between them. First, we can compute the edge weight between two movies $i$ and $j$ as their cosine similarity $s_{\text{cos}}(i,j)$ [cf. (17)]. Second, we can find edge weights through adjusted cosine similarity [32] by normalizing average ratings for each movie. Third, we may utilize Pearson correlation [33] by restricting attention to users who rate both movies $i$ and $j$. All these three constructions – cosine, adjusted cosine, and Pearson – are based on comparing the rating profiles of two items. The intuition is that movies are similar if their reviews by the same users are similar. The resulting networks would be symmetric and, potentially, have edges with negative weights. Another way to construct the network is to consider the proportion of users who rate both movies relative to the users who rate only one of them

$$
s_{\text{usage}}(i,j) = \frac{|\{u \in \mathcal{U} : (u,i) \in \mathcal{K}, (u,j) \in \mathcal{K}\}|}{|\{u \in \mathcal{U} : (u,i) \in \mathcal{K}\}|}.
$$

The above formulation is based on the idea that two movies are similar if the set of users that rate them are similar, thus, yields directed networks since different movies enjoy different levels of popularity represented by the denominator in (20). In all constructions, edges are removed if their weights are smaller than a threshold in absolute values.

The rating prediction problem can also be considered from the dual domain by switching the roles of items and users. For each item $i$, its ratings $\mathbf{R}_{i} \in \mathbb{R}^{|\mathcal{U}|}$ evaluated by different users can be considered as a graph signal defined on top of a common network $\mathcal{G}_{\text{user}}$ describing proximities between users. Under such setup, items $i$ and $j$ should be similar if they are given similar ratings by users with alike tastes, even if there is no user that rated both of them. We therefore consider smoothing the ratings $\mathbf{R}_{i}$ utilizing a similarity network between users before computing the cosine similarity between items in collaborative filtering, and call this method user-based-diffusion item-based $k$-NN.

To illustrate how diffusion filtering can facilitate the similarity computation and improve the performance of recommendation systems, we consider the MovieLen-100k dataset [34] containing 100,000 ratings from 943 users on 1,682 movies. We perform 5-fold cross validation by partitioning the users into 5 random sets. For each cross validation round, 10 randomly selected ratings from each of the users in a given fold are withheld and kept as the test set, with all remaining ratings forming the training set. As a performance metric we use the global root mean squared error (RMSE), averaged across the 5 different testing sets. Matrix completion and $k$-NN CFs without diffusion filtering are utilized as benchmark algorithms. In diffusion filtering, we construct different underlying networks based only on the available training set.

The global RMSEs for different methods averaged across 3 different 5-fold partitions of the data are summarized in Table 1. Collaborative filtering with 8 different diffusion filters – 4 ways to construct networks and 2 choices of Laplacians – all improve the RMSE values except for the user-based CFs with usage networks. Also, user-based CFs outperform matrix completion methods when preprocessed with normalized diffusion filtering. The almost 1% improvement in RMSE – from 0.9014 to 0.8992 and from 0.8881 to 0.8874 – is substantial in rating prediction problems, considering that the benchmark algorithms are highly optimized [30,35]. The improvements increase to 1.5%–4% when networks are constructed from all ratings, as opposed to only those in the training set. The results presented are robust for a wide range of neighbors $k$, diffusion rates $\alpha$, and thresholding parameters used in constructing the networks. The only diffusion filter that increases RMSE is user-based CF with usage networks, partly due to the fact that (20) is based only on the number of ratings and not on the content of these. This problem can be solved by considering positive ratings separately from negative ratings. It is also observed that normalized Laplacian works better than unnormalized Laplacian. Part of the reason is because some movies or users tend to be isolated – few weak connections – in the constructed networks after edge weight thresholding. Therefore, diffusion with un-normalized Laplacian cannot effectively act on these isolated nodes and the normalized Laplacian achieves a better performance (cf. Section 5).

### 6. Conclusion

We defined diffusion filtering as a smoothing method for signals supported on networks. The method relies on the temporal heat map induced by the diffusion of signals across the network and evaluates the accumulated temperatures across time. We showed that diffused signals are stable with respect to perturbations in the underlying network and demonstrated that diffusion filtering can improve the performance of recommendation systems.

<table>
<thead>
<tr>
<th>unnormalized Laplacian</th>
<th>normalized Laplacian</th>
<th>$k$-NN matrix completion</th>
</tr>
</thead>
<tbody>
<tr>
<td>cosine</td>
<td>adj. cosine</td>
<td>Pearson</td>
</tr>
<tr>
<td>user-based CF</td>
<td>0.8995</td>
<td>0.8997</td>
</tr>
<tr>
<td>item-based CF</td>
<td>0.8851</td>
<td>0.8856</td>
</tr>
</tbody>
</table>
7. REFERENCES


