Collaborative Filtering via Graph Signal Processing

Weiyu Huang, Antonio G. Marques, and Alejandro Ribeiro

***Abstract*—This paper develops new designs for recommender systems inspired by recent advances in graph signal processing. Recommender systems aim to predict unknown ratings by exploiting the information revealed in a subset of user-item observed ratings. Leveraging the notions of graph frequency and graph filters, we demonstrate that a common collaborative filtering method – k-nearest neighbors – can be modeled as a specific *band-stop* graph filter on networks describing similarities between users or items. These new interpretations pave the way to new methods for enhanced rating prediction. For collaborative filtering, we develop more general band stop graph filters. The performance of our algorithms is assessed in the MovieLens-100k dataset, showing that our designs reduce the root mean squared error (up to a 6.20% improvement) compared to one incurred by the benchmark collaborative filtering approach.**

***Index Terms*—Collaborative filtering, recommender systems, graph signal processing, bandlimited graph signals, graph filters.**

I. Introduction

The widespread deployment of the Internet technologies has generated a massive enrollment of online customers in web services, propelling the need for implementation of recommender systems (RS) to assist customers in making decisions. In a succinct way, RS are algorithms that collect information about how users of a particular service rate different items. The collected information is then used, along with additional sources of exogenous information, to provide customers with recommendations for the unrated items [1], [2].

Research on RS includes the so-called content filtering approach, which starts by defining a set of features that characterize users and items and then uses those to perform predictions on the unrated items [1], [2]. It also includes the collaborative filtering (CoFi) approach, which relies mostly on past user behavior and carries out predictions without defining an a priori set of features. Although CoFi comes with certain disadvantages (in particular when rating new products or users), it typically requires less assumptions than content filtering and yields a superior performance in real datasets [2]. As a result, it has emerged as the central approach for RS. A common technique to design CoFi algorithms is nearest neighborhood methods (NNM), which work under the assumption that users who are similar tend to give similar ratings for the same product, proceed into two phases. Firstly, using a pre-specified similarity metric, a similarity score is computed for each pair of users. Secondly, the unknown ratings for a particular user are obtained by combining the ratings that similar users have given to the unrated items. To avoid overfitting and simplify computations, only a subset of the users (the ones who are more similar and have rated the item) is considered. A similar approach can be used to compute a similarity score among items, giving rise to the so-caller item-based collaborative approaches.

The goal in this paper is to reinterpret CoFi algorithms using tools from graph signal processing (SP). In simple words, graph SP addresses the problem of analyzing and extracting information from data defined not in regular domains such as time or space, but on more irregular domains that can be conveniently represented by a graph. The tacit assumption is that the network structure defines a notion of proximity or dependence among the nodes of the graph [3], [4], which must be leveraged when generalizing classical SP algorithms to process signals defined in more irregular graph domains. The theory and applications of graph SP is growing rapidly [5]–[10]. This paper designs new and more general schemes, but equally relevant unveils important connections between CoFi and graph SP. More precisely, we show that NNM can be viewed as algorithms that obtain the ratings by processing the available information with a graph filter. This interpretation not only provides a better understanding on the differences and similarities between both approaches, but it also opens the door to the design of more advanced algorithms leading to a better recommendation accuracy. In short, the contributions of this paper are: (a) To demonstrate how the CoFi approaches based on NNM can be considered from graph SP approach. (b) To exploit this interpretation to design more general algorithms for NNM. (c) To show that the proposed methods produce significant improvement for the MovieLens-100k dataset [11][1].

II. Fundamentals of CoFi and Graph SP

We start by introducing the basic notation and formulating the CoFi problem. We then describe the NNM method and review the graph SP tools used in the following sections.

Consider an RS setup with U users indexed by u, and I items indexed by i. The rating that user u has given to item i is represented as $X_{u,i}$. For mathematical convenience, such ratings can be collected either into the rating matrix $\mathbf{X} \in \mathbb{R}^{U\times I}$, or into the rating vector $\mathbf{x} = \text{vec}(\mathbf{X}) \in \mathbb{R}^{UI}$. Additionally, vectors $\mathbf{x}_u = [X_{u,1}, ..., X_{u,I}]^\top \in \mathbb{R}^I$ represent the ratings by the u-th user. To account for the fact that not

Work in this paper is supported by the Spanish MINECO grants No TEC2013-41604-R and TEC2016-75361-R, and the USA NSF CCF-1217963.
W. Huang and A. Ribeiro are with the Dept. of Electrical and Systems Eng., Univ. of Pennsylvania. A. G. Marques is with the Dept. of Signal Theory and Comms., King Juan Carlos Univ.

[1] **Notation:** Generically, the entries of a matrix $\mathbf{X}$ and a vector $\mathbf{x}$ will be denoted as X_{ij} and x_i; however, when contributing to avoid confusion, the alternative notation $[\mathbf{X}]_{ij}$ and $[\mathbf{x}]_i$ will be used. In general, $\hat{\mathbf{x}}$ denotes the frequency coefficients of $\mathbf{x}$, while $\hat{\mathbf{x}}$ is the estimate of $\mathbf{x}$. The notation $^\top$ stands for transpose; $\text{diag}(\mathbf{X})$ is a diagonal matrix satisfying $[\text{diag}(\mathbf{X})]_{ii} = [\mathbf{X}]_{ii}$ and zero for other entries; $\lambda_{\max}(\mathbf{X})$ is the largest eigenvalue of the symmetric matrix $\mathbf{X}$;$\|\mathbf{x}\|_p$ is the p-norm of $\mathbf{x}$; and $|\mathcal{X}|$ is the cardinality of the set $\mathcal{X}$.

ISBN 978-0-9928626-7-1 © EURASIP 2017

all ratings are available, let $\mathcal{S}$ denote the set of indexes that identify user-item pairs whose rating is known. Similarly, $\mathcal{S}_u$ denotes a set containing the indexes of the items that user u has rated. We can then use $\mathbf{x}_{\mathcal{S}} \in \mathbb{R}^{|\mathcal{S}|}$ to denote a vector containing the *known* ratings. The problem of interest is as follows: Given the ratings $\mathbf{x}_{\mathcal{S}}$ for the item-user pairs in $\mathcal{S}$, estimate the full rating vector $\mathbf{x}$ (matrix $\mathbf{X}$).

A. CoFi via NNM

As explained in the introduction, NNM builds on the assumption that if a pair of users u and v possess similar taste, then their ratings X_{iu} and X_{iv} for a particular item i, are going to be similar as well. To formulate this rigorously, we start with user-based NNM, and let $\mathbf{B} \in \mathbb{R}^{U \times U}$ be a matrix whose entry B_{uv} denotes the similarity between the pair of users u and v. Given the particularities of a CoFi setup, $B_{u,v}$ has to be computed using a metric that takes into account only the ratings available in $\mathbf{x}_{\mathcal{S}}$. Define the set $\mathcal{S}_{uv}$ as the intersection of $\mathcal{S}_u$ and $\mathcal{S}_v$, i.e., the set of items that have been rated by both u and v, a common choice to compute the similarity score is finding first the sample correlations as

$$\Sigma^{\mathcal{U}}_{uv} := \frac{1}{|\mathcal{S}_{uv}|} \sum_{i \in \mathcal{S}_{uv}} (X_{ui} - \mu_{uv})(X_{vi} - \mu_{uv}), \qquad (1)$$

with $\mu_{uv} := \sum_{i \in \mathcal{S}_{uv}} X_{ui}/|\mathcal{S}_{uv}|$. Note that the previous covariances and means are found using only the items that were commonly rated by u and v. The similarity score would then be found by simply setting $B_{uv} = \Sigma^{\mathcal{U}}_{uv}$. In the context of RS, a more common approach is to use Pearson correlations,

$$B_{uv} = \left[(\text{diag}(\mathbf{\Sigma}^{\mathcal{U}}))^{-1/2} \, \mathbf{\Sigma}^{\mathcal{U}} \, (\text{diag}(\mathbf{\Sigma}^{\mathcal{U}}))^{-1/2} \right]_{uv}, u \neq v, \quad (2)$$

and $B_{uu} = 0$. The main idea behind NNM is that when predicting the rating X_{ui}, only the ratings X_{vi} from users v that are very *similar* to u must be used. To do so, denote $\mathcal{K}_{ui}$ as the set of k users who are the most similar to u (largest values of B_{uv}) and have rated the item i. Leveraging these definitions, the unknown ratings are finally predicted as

$$\hat{X}_{ui} = \frac{\sum_{v \in \mathcal{K}_{ui}} B_{uv} \, (X_{vi} - \mu_v)}{\sum_{v \in \mathcal{K}_{ui}} B_{uv}} + \mu_u, \qquad (3)$$

where $\mu_u = \sum_{i \in \mathcal{S}_u} X_{ui}/|\mathcal{S}_u|$. At an intuitive level, the subtraction and addition of μ_v and μ_u account for the fact that different users may be more generous than others.

B. Graph SP

Consider a directed graph $\mathcal{G}$ with a set of N nodes or vertices $\mathcal{N}$ and a set of links $\mathcal{E}$, such that if node n is connected to m, then $(n, m) \in \mathcal{E}$. For any given graph we define the adjacency matrix $\mathbf{A}$ as a sparse $N \times N$ matrix with non-zero elements $A_{m,n}$ if and only if $(n, m) \in \mathcal{E}$. The value of $A_{m,n}$ captures the strength of the connection from n to m.

The focus of graph SP is on graph signals defined on the set of nodes $\mathcal{N}$. Formally, each of these signals can be represented as a vector $\mathbf{z} \in \mathbb{R}^N$ where the n-th element represents the value of the signal at node n. To facilitate the connections with NNM, in this work we chose as shift the adjacency matrix $\mathbf{A}$;

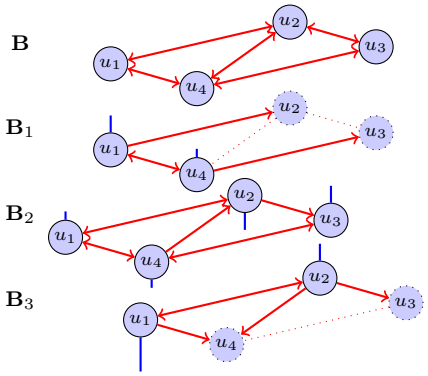

Fig. 1: CoFi as graph filters. The ratings for each item can be considered as graph signals on a network that depends on the item. For each specific item, edges starting from users who have not rate them are removed. Then, given a specific user u, for all the edges coming into u, only the ones with the k-highest edge weights are kept. Proper normalization are then applied to make each $\mathbf{B}_i$ right stochastic.

however, our results can be easily generalized for other choices such as Laplacians [3]. We assume $\mathbf{S}$ is diagonalizable, so that $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$ with $\mathbf{\Lambda} = \text{diag}(\boldsymbol{\lambda}) \in \mathbb{C}^{N \times N}$ being diagonal and $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N]$. When $\mathbf{S}$ is symmetric we have that $\mathbf{V}$ is real and unitary, which implies $\mathbf{V}^{-1} = \mathbf{V}^{\top}$.

Graph filters are a particular class of *linear* graph-signal operators able to be represented as matrix polynomials of $\mathbf{S}$ [4]

$$\mathbf{H} := \sum_{l=0}^{L-1} h_l \mathbf{S}^l. \qquad (4)$$

For a given input $\mathbf{z}$, the output of the filter is simply $\mathbf{y} = \mathbf{H}\mathbf{z}$. The filter coefficients are collected into $\mathbf{h} := [h_0, \ldots, h_{L-1}]^{\top}$, with $L-1$ denoting the filter degree. The eigendecomposition of $\mathbf{S}$ is used to define the frequency representation of graph signals and filters. For a signal $\mathbf{z} \in \mathbb{R}^N$ and a graph shift operator $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1} \in \mathbb{R}$. The vectors

$$\tilde{\mathbf{z}} = \mathbf{V}^{-1}\mathbf{z} \quad \text{and} \quad \mathbf{z} = \mathbf{V}\tilde{\mathbf{z}} \qquad (5)$$

form a Graph Fourier Transform (GFT) pair [3], [4].

The GFT encodes a notion of variability for graph signals akin to one that the Fourier transform encodes for temporal signals [4], [12]. Specifically, the smaller the distance between λ_p and $|\lambda_{\max}|$ in the complex spectrum, the lower the frequency it represents. This idea is based on defining the total variation of a graph signal $\mathbf{z}$ as $\text{TV}(\mathbf{z}) = \|\mathbf{z} - \mathbf{S}\mathbf{z}/\lambda_{\max}(\mathbf{S})\|_1$, with smoothness being associated to small values of TV. Then, given a $(\lambda_p, \mathbf{v}_p)$ pair, one has that $\text{TV}(\mathbf{v}_p)$, which provides an intuitive way to order the different frequencies.

III. CoFi from a Graph SP Perspective

In this section, we show that if the ratings $\mathbf{x}$ are viewed as graph signals defined on user-to-user networks, then NNM predict signals $\hat{\mathbf{x}}$ that are *bandlimited* in the frequency domain of those networks. That is, signals that can be expressed as a combination of a few eigenvectors of the graph shift operator. This viewpoint allows us to develop more general algorithms

ISBN 978-0-9928626-7-1 © EURASIP 2017

with better performance. To that end, let us focus on the generation of $\hat{\mathbf{x}}^i$, i.e., the predicted ratings for item i, using the ratings from other *users* and the similarities among them.

The first step is to define the input graph signal denoted as $\check{\mathbf{x}}^i \in \mathbb{R}^U$. This requires setting $[\check{\mathbf{x}}^i]_v = X_{vi} - \mu_v$ for every user v who has rated item i, and $[\check{\mathbf{x}}^i]_{v'} = 0$ for $v' \notin \mathcal{S}^i$. Since the bias has been removed, setting the unknown ratings to zero assigns a neutral preference for the item. The second step is to construct the user-similarity network, which will serve as graph shift operator. To this end, we start with the matrix $\mathbf{B}$ whose entries are given in (2). Then, in order to account for the fact that ratings from users who do not rate i should not be considered when predicting i, we remove any edges starting from v if X_{vi} is unknown. This implies that the similarity network, which will be denoted as $\mathbf{B}_i$, will depend on the particular item i. The final steps are to keep only the edges corresponding to the k most similar users and normalize each row so that the resultant matrix is left stochastic [cf. the denominator in (3)]. Mathematically, this implies that the matrix $\mathbf{B}_i \in \mathbb{R}^{U\times U}$ is defined as

$$[\mathbf{B}_i]_{uv} = \begin{cases} B_{uv}/\sum_{v'\in\mathcal{K}_{u,i}} B_{uv'} & \text{if} \quad v \in \mathcal{K}_{u,i} \\ 0 & \text{if} \quad v \notin \mathcal{K}_{u,i} \end{cases}, \qquad (6)$$

where we recall that $\mathcal{K}_{u,i}$ contains the k users that are most similar to u and have rated item i. An example of this procedure using the MovieLenss-100k dataset is illustrated in Figure 1, where the top network represents the original $\mathbf{B}$ and the subsequent plots represent $\mathbf{B}_i$ for several items.

Once the graph signal $\check{\mathbf{x}}^i$ and the graph shift-operator $\mathbf{B}_i$ are defined, the predicted ratings are simply given by

$$\hat{\mathbf{x}}^i = \mathbf{B}_i\check{\mathbf{x}}^i, \qquad (7)$$

cf. (3). In words, the estimated ratings are obtained after applying the graph filter $\mathbf{H} = \mathbf{B}_i$ to the input signal $\check{\mathbf{x}}^i$.

We now analyze the behavior of (7) in the frequency domain, to conclude that $\mathbf{H} = \mathbf{B}_i$ acts as a band-stop graph filter. Given an item i, consider the eigen-decomposition for the user-similarity network $\mathbf{S} = \mathbf{B}_i = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$. Denote the GFT of the known input signal as $\tilde{\check{\mathbf{x}}}^i = \mathbf{V}^{-1}\check{\mathbf{x}}^i$, and the GFT of the predicted rating as $\tilde{\hat{\mathbf{x}}}^i = \mathbf{V}^{-1}\hat{\mathbf{x}}^i$. The two GFTs are related via

$$\tilde{\hat{\mathbf{x}}}^i = \mathbf{V}^{-1}\hat{\mathbf{x}}^i = \mathbf{V}^{-1}\mathbf{B}_i\check{\mathbf{x}}^i = \mathbf{V}^{-1}\mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}(\mathbf{V}\tilde{\check{\mathbf{x}}}^i) = \mathbf{\Lambda}\tilde{\check{\mathbf{x}}}^i. \qquad (8)$$

Therefore, the frequency response of the filter implementing NNM is $\text{diag}(\tilde{\mathbf{h}}) = \text{diag}(\tilde{\mathbf{b}}_i) = \mathbf{\Lambda}$ and the p-th frequency coefficient of the predicted output is $[\tilde{\hat{\mathbf{x}}}^i]_p = \lambda_p[\tilde{\check{\mathbf{x}}}^i]_p$. Since $\mathbf{B}_i$ is likely to be non-symmetric, λ_p is expected to be a complex number. Remember that $\lambda_{\max}(\mathbf{B}_i)$ is always 1 because of right stochasticity and that eigenvectors can be ordered according to $\text{TV}(\mathbf{v}_p) = |\lambda_p - 1|$; see [12] and the related discussion after Definition 1. As a result, smooth (low-frequency) eigenvectors are signals where $\|\mathbf{v}_q - \mathbf{B}_i\mathbf{v}_q\| \approx 0$; i.e., full rating signals where users that are similar tend to agree.

To gain further intuition on the spectral behavior of (8), we examine the frequency response of $\mathbf{B}_i$ for the MovieLenss-100k dataset. Specifically, for each $\mathbf{B}_i$, we order its eigenvalues according to $|\lambda_p - 1|$, and record the frequency response $\tilde{\mathbf{b}}_i$ for low, middle, and high frequencies. The I frequency responses obtained using this procedure are then averaged across i, giving rise to the single transfer function depicted in Figure 2 (a). To help visualization, the scale in the horizontal axis is not homogeneous and only the real part of the eigenvalues is shown (the imaginary part is very small). The main finding is that the frequency response is zero for more than 90% of the frequencies, implying that the predicted signal will be graph bandlimited. Another observation of interest is that the frequencies not rejected by the filter and that are present in the predicted output are the ones associated with the first eigenvectors (low values of p) and the last eigenvectors (high values of p). The first eigenvectors represent signals of small total variation, while the last ones are associated with signals of high variation. Since the diagonal elements of each matrix $\mathbf{B}_i$ are zero, the sum of the eigenvalues is zero, with the eigenvalues associated with low frequencies being positive, and those associated with signals of large total variance associated being negative. The low-pass components represent signals where similar users tend to have similar ratings, providing the *the big picture* for the predicted rating. Differently, the high pass component focuses on the *differences between users with similar taste for the particular item*. With this interpretation one can see (8) as a filter that eliminates the irrelevant features (middle frequencies), smoothes out the similar components (low frequencies) and preserves the discriminative features (high frequencies). This band-stop behavior where both high and low graph frequencies are preserved is not uncommon in image processing (image de-noising and image sharpening, respectively) [13], and was also observed in brain signal analytics [8], [14].

IV. Enhancing CoFi via Graph SP

Using definitions and tools from graph SP, the previous section demonstrated that the rating predictions generated by NNM can be understood as signals that are sparse in a graph frequency domain. In this section, we illustrate how these interpretations can be leveraged to design novel graph-SP-based CoFi methods with enhanced prediction performance.

As shown in Section III, the user-based NNM predict the rating for item i via $\hat{\mathbf{x}}^i = \mathbf{B}_i\mathbf{x}_i$, which can be modeled as the implementation of a band-stop graph filter of order one. Our proposal here is, using $\mathbf{S} = \mathbf{B}_i$ as shift, to design other types of *band-stop* graph filters $\mathbf{H}(\mathbf{S})$ to perform rating predictions. Consider first $\mathbf{H} = \mathbf{B}_i^2$, whose frequency response is $\text{diag}(\tilde{\mathbf{h}}) = \text{diag}(\tilde{\mathbf{b}}_i)^2 = \mathbf{\Lambda}^2$. The fact of $\mathbf{B}_i$ being a band-stop filter implies that many of the entries of its frequency response $\tilde{\mathbf{b}}_i$ are zero. As a result, $\mathbf{B}_i^2$ has a band-stop behavior too and the same holds true for any positive power of $\mathbf{B}_i$. Since all powers of $\mathbf{B}_i$ are band-stop operators, the unknown ratings predicted with graph filters of the form

$$\hat{\mathbf{x}}_i = \mathbf{H}\check{\mathbf{x}}_i \;\text{ with }\; \mathbf{H} = \sum_{l=0}^{L} h_l\mathbf{B}_i^l, \qquad (9)$$

ISBN 978-0-9928626-7-1 © EURASIP 2017

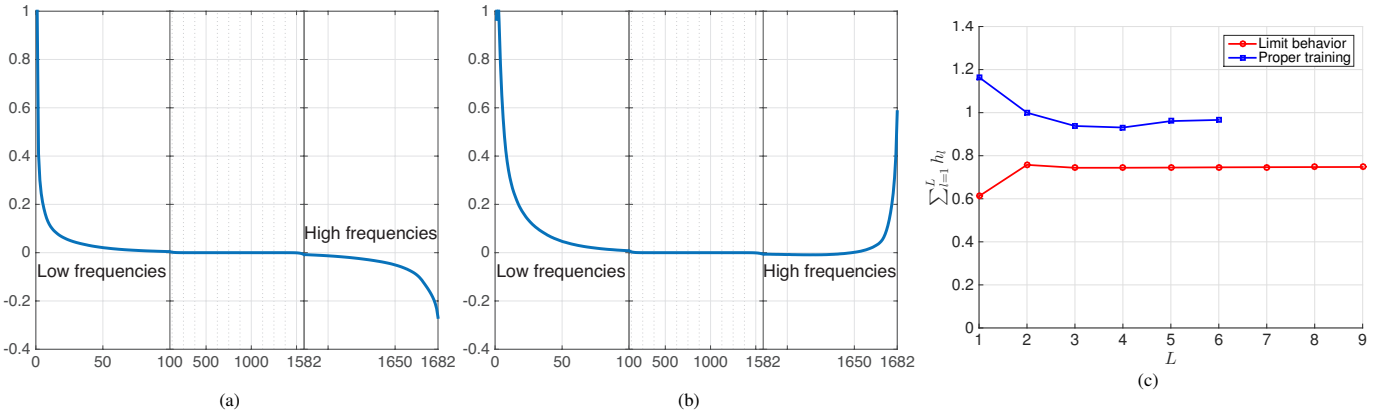

Fig. 2: (a) and (b): Frequency response for the graph filters. For each $\mathbf{B}_i$, we order its eigenvalues according to $|\lambda_l - \lambda_{\max}|$ to denote high and low frequencies, and record the corresponding frequency response for low, middle, and high frequencies. The average behavior of such frequency response across all $\mathbf{B}_i$ is visualized. (a) Response for user-based CoFi; (b) Response for the best user-based filter trained on training set using networks $\{\mathbf{B}_i^l\}_{l=1}^6$. (c) Sum of filter coefficients $\sum_{l=1}^{L} h_l$ for different tap of filters. In both the Limit behavior test (Red) and Proper training (Blue), the sum of coefficients is close to 1.

will also give rise to bandlimited signals. Hence, predictions in (9) are generalizations of the traditional NNM in (7), which estimate $\hat{\mathbf{x}}_i$ using a filter $\mathbf{H} = \mathbf{B}_i$ of order one. This graph-frequency interpretation can be complemented by understanding the effect of $\mathbf{B}_i$ on the graph vertex domain. To do so, note that $\mathbf{B}_i^0 \check{\mathbf{x}}^i = \check{\mathbf{x}}^i$ coincides with the original signal, $\mathbf{B}_i \check{\mathbf{x}}^i$ is an average of the ratings given by one-hop neighbors, $\mathbf{B}_i^2 \check{\mathbf{x}}^i$ is an average of elements in nodes that interact via intermediate common neighbors, and, in general, $\mathbf{B}_i^l \check{\mathbf{x}}^i$ describes interactions between l-hop neighbors. Therefore, on top of using the ratings of very similar users to make predictions, the powers of the matrix $\mathbf{B}_i^l$ in the graph filter in (9) also account for chains of users with similar taste, exploiting them to generate enhanced predictions.

Compared to classical NNM, the filter coefficients $\mathbf{h}$ are not known a priori, and therefore need to be learned from a training set. Besides, h_0 is irrelevant since $\mathbf{B}_i^0 \mathbf{x}^i = \mathbf{x}^i$ and therefore would not be helpful in predictions. Then, the filter coefficients are found by solving

$$\min_{\mathbf{h}} \sum_{(u,i)\in\mathcal{S}} \left| \left[\left(\sum_{l=1}^{L} h_l \mathbf{B}_i^l \right) \check{\mathbf{x}}^i \right]_u - X_{u,i} \right|^2 + r\|\mathbf{h}\|_2^2, \quad (10)$$

where r is a regularization parameter that can be tuned by cross-validation on the training set to avoid overfitting. Note that formulations in (10) are least square problems, which using the Moore–Penrose pseudo inverse, admit "closed form" solutions. If the value of L is too large and a sparse vector of filter coefficients is desired, the regularizer $\|\mathbf{h}\|_2^2$ can be either replaced or augmented with $\|\mathbf{h}\|_1$.

V. Numerical Experiments

In this section, we illustrate how our methods improve the rating accuracy in real data. For that purpose we use the MovieLens-100k dataset [11], which contains ratings from 943 users on 1,682 movies. The number of available ratings is 100,000, i.e., the 6.3% of the total number of user-item pairs. We randomly select 100 ratings as the testing set $\mathcal{X}_{\text{ts}}$, and use the rest as training set $\mathcal{X}_{\text{tr}}$. The set containing the indexes of elements in $\mathcal{X}_{\text{ts}}$ and $\mathcal{X}_{\text{tr}}$ is denoted as $\mathcal{S}_{\text{ts}}$ and $\mathcal{S}_{\text{tr}}$, respectively. The networks and filter coefficients are only trained on the training set. As a performance metric we use the global root mean squared error (RMSE). User-based NNM is used as benchmark algorithms. To get an estimate for the regularization constant r used in (10), we perform cross-validation by breaking the ratings in the training set into three equally sized subsets.

Before we start to compare different approaches, the first task is to assess the best performance that one can achieve

TABLE I: Generalized filtering – limit behavior: RMSE for different taps with coefficients learned on the testing set $\mathcal{X}_{\text{ts}}$

Number of taps	RMSE ($r = 0$)	RMSE ($r = 0.5$)
$L = 1$	0.9036	0.9036
$L = 2$	0.8921	0.8921
$L = 3$	0.8226	0.8735
$L = 4$	0.8218	0.8643
$L = 5$	0.8128	0.8593
$L = 6$	0.8073	0.8572
$L = 7$	0.8068	0.8560
$L = 8$	0.7922	0.8554
$L = 9$	0.8026	**0.8550**

TABLE II: Generalized filtering – proper training: RMSE for different taps with coefficients learned on the training set $\mathcal{X}_{\text{tr}}$

Number of taps	RMSE ($r = 0$)	RMSE (r learned from cross-validation)
benchmark	0.9116	
$L = 1$	0.9175	0.9175
$L = 2$	0.8875	0.8875
$L = 3$	0.8647	0.8647
$L = 4$	0.8661	0.8661
$L = 5$	0.8557	0.8554
$L = 6$	0.8609	**0.8551 (6.20% improvement)**

with the setup at hand. To this end, we use the networks $\mathbf{B}_i$ learned on the training set $\mathcal{X}_{\text{tr}}$ and learn the filter coefficients by solving the problem in (10) using the *testing* set $\mathcal{X}_{\text{ts}}$

$$\min_{\mathbf{h}} \sum_{(u,i)\in\mathcal{S}_{\text{ts}}} \left| \left[\left(\sum_{l=1}^{L} h_l \mathbf{B}_i^l \right) \check{\mathbf{x}}^i \right]_u - \check{X}_{u,i} \right|^2 + r\|\mathbf{h}\|_2^2. \quad (11)$$

Since the coefficients above are biased towards the data in $\mathcal{X}_{\text{ts}}$ and all other schemes will be trained using $\mathcal{X}_{\text{ts}}$, the performance achieved by (11) on $\mathcal{X}_{\text{ts}}$ will serve as a benchmark for all other schemes. The RMSE across ratings in the testing set $\mathcal{X}_{\text{ts}}$ using the $\mathbf{h}$ trained in (11) for different values of L and r is presented in Table I. There are several interesting observations. Firstly, the RMSE for both $r = 0$ and $r = 0.5$ decreases as the number of filter taps L increases from 1 to 6, remaining flat for $L > 6$. This seems to imply that considering chains of more than 6 users does not improve prediction performance (recall $U = 943$ and $k = 40$). Secondly, the RMSE with large L and $r = 0$ is around 0.80, which will be the value considered as the benchmark for algorithm that learn $\mathbf{h}$ in $\mathcal{X}_{\text{tr}}$ and test their performance in $\mathcal{X}_{\text{ts}}$. Finally, the difference between RMSE for $r = 0$ and $r = 0.5$ is around 0.05, which represents an increase of approximately 6%.

When we solve the actual problem in (10) with coefficients learned on the training set $\mathcal{X}_{\text{tr}}$, we rely on the results in Table I to limit the maximum number of taps to 6. The RMSE on the testing set $\mathcal{X}_{\text{ts}}$ for different values of L and r is presented in Table II. The main observations are: i) higher order filters perform better than the traditional order-one NNM (user-based NNM attain an RMSE of 0.9116, while for $L = 6$ and $r = 0.5$ our method attains an error of 0.8551, which is 6.20% smaller); and ii) the prediction performance, especially that for the case where $r = 0.5$, is not much worse than that shown in Table I, and the trends are also similar to those shown in Table I. Moreover, when proper regularization is applied, the optimal coefficients learned from the training set are also close to the coefficients learned from the testing set.

Another interesting observation is that the optimal coefficients learned from either training set in (10) or testing set in (11) tend to satisfy that $\sum_{l=1}^{L} h_l \simeq 1$, as illustrated in Figure 2 (c). Such a property does not seem to depend on the number of taps used in the filter. Recall that traditional user-based NNM can be considered as a specific graph band-pass filter with coefficients $h_1 = 1$ and $h_l = 0$ for any $l \neq 1$. Therefore, this supports the idea that traditional user-based NNM is the optimal design for a band-stop graph filter, if only filters with one tap are considered.

To gain further insights, the frequency response for user-based filter with $L = 6$ and $\mathbf{h}$ learned in $\mathcal{X}_{\text{tr}}$ is illustrated in Figure 2 (b). The frequency response is highly similar to the one of user-based NNM in Figure 2 (a), since both of them are band-stop filters; however, there are two major differences. Firstly, both the amplitude and the range for low and high frequencies with high response in absolute value increases. Secondly, the frequency response for high frequency in Figure 2 (b) becomes positive, whereas the response for high frequency in NNM is negative. The second point is potentially the reason that designing filters by training filter coefficients can improve RMSE by 6.10%. The two differences can be considered as the advantages of designing filters to have a more flexible form compared to NNM.

VI. Conclusions

This paper exploited results from graph SP to propose new interpretations for RS methods. Our first contribution was to show that CoFi can be considered as a specific band-stop graph filter operating on the network describing similarities between users or items. Leveraging this, we then proposed a new method for RS using other types of graph band stop filters. We also proposed a computationally efficient scheme to design the parameters that define our methods and assessed their performance in the MovieLens-100k dataset. The results obtained showed that, compared to the benchmark approaches, we reduced the RMSE by a rate of 6.20%. Relevant observations regarding how the networks are formed as well as on filter coefficients and the corresponding frequency response were also discussed. Future work would be to consider other types of graph filters and to investigate matrix completion from graph SP perspectives.

References

[1] F. Ricci, L. Rokach, and B. Shapira, *Introduction To Recommender Systems Handbook*. Springer, 2011.

[2] Y. Koren *et al.*, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[3] D. Shuman, S. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, Mar. 2013.

[4] A. Sandryhaila and J. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.

[5] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Sampling of graph signals with successive local aggregations," *IEEE Trans. Signal Process.*, vol. 64, no. 7, pp. 1832–1843, Apr. 2016.

[6] N. Perraudin, A. Loukas, F. Grassi, and P. Vandergheynst, "Towards stationary time-vertex signal processing," *arXiv preprint arXiv:1606.06962*, June 2016.

[7] F. Gama, A. G. Marques, G. Mateos, and A. Ribeiro, "Rethinking sketching as sampling: A graph signal processing approach," *arXiv preprint arXiv:1611.00119*, Nov. 2016.

[8] W. Huang *et al.*, "Graph frequency analysis of brain signals," *J. Sel. Topics Signal Process.*, vol. 10, no. 7, pp. 1189–1203, Oct. 2016.

[9] R. Liu, H. Nejati, and N.-M. Cheung, "Simultaneous low-rank component and graph estimation for high-dimensional graph signals: Application to brain imaging," *arXiv preprint arXiv:1609.08221*, Sep. 2016.

[10] V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst, "Matrix completion on graphs," *arXiv preprint arXiv:1408.1717*, Aug. 2014.

[11] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: an open architecture for collaborative filtering of netnews," in *Proc. of the 1994 ACM Conf. on Computer Supported Cooperative Network*. ACM, Oct. 1994, pp. 175–186.

[12] A. Sandryhaila and J. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3042–3054, June 2014.

[13] A. Kheradmand and P. Milanfar, "A general framework for regularized, similarity-based image restoration," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5136–5151, Dec. 2014.

[14] J. D. Medaglia *et al.*, "Functional alignment with anatomical networks is associated with cognitive flexibility," *arXiv preprint arXiv:1611.08751*, Nov. 2016.

ISBN 978-0-9928626-7-1 © EURASIP 2017