Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence

Rank Aggregation via Low-Rank and Structured-Sparse Decomposition

Yan Pan[1], **Hanjiang Lai**[1], **Cong Liu**[1], **Yong Tang**[2], **and Shuicheng Yan**[3]
[1]Sun Yat-sen University, Guangzhou, China
[2]Southern Normal University, Guangzhou, China
[3]National University of Singapore, Singapore

Abstract

Rank aggregation, which combines multiple individual rank lists to obtain a better one, is a fundamental technique in various applications such as meta-search and recommendation systems. Most existing rank aggregation methods blindly combine multiple rank lists with possibly considerable noises, which often degrades their performances. In this paper, we propose a new model for *robust rank aggregation (RRA)* via matrix learning, which recovers a latent rank list from the possibly incomplete and noisy input rank lists. In our model, we construct a pairwise comparison matrix to encode the order information in each input rank list. Based on our observations, each comparison matrix can be naturally decomposed into a shared low-rank matrix, combined with a deviation error matrix which is the sum of a column-sparse matrix and a row-sparse one. The latent rank list can be easily extracted from the learned low-rank matrix. The optimization formulation of RRA has an element-wise multiplication operator to handle missing values, a symmetric constraint on the noise structure, and a factorization trick to restrict the maximum rank of the low-rank matrix. To solve this challenging optimization problem, we propose a novel procedure based on the Augmented Lagrangian Multiplier scheme. We conduct extensive experiments on meta-search and collaborative filtering benchmark datasets. The results show that the proposed RRA has superior performance gain over several state-of-the-art algorithms for rank aggregation.

Introduction

Rank aggregation, a task which combines multiple individual rank lists (a rank list is an ordered list of a set of items) to obtain a better rank list, has received considerable attention in recent years. Rank aggregation plays an important role in a wide range of applications, such as meta-search (Dwork *et al*. 2001), collaborative filtering (Gleich and Lim 2011), machine translation (Rosti *et al*. 2007) and object categorization (Ye *et al*. 2012). Many methods for rank aggregation have been proposed in the literature (Borda 1781; Dwork *et al*. 2001; Fagin *et al*. 2003; Liu *et al*. 2007; Klementiev *et al*. 2008; Qin *et al*. 2010a; Gleich and Lim 2011).

In real world applications such as meta-search and recommendation systems, there are two main challenges in rank aggregation: the input rank lists might (1) be incomplete, and (2) be corrupted by considerable noises. The first challenge has been well addressed by existing methods (Gleich and Lim 2011; Qin *et al*. 2010a; Dwork *et al*. 2001). However, most existing rank aggregation methods blindly combine all the individual rank lists with possibly considerable noises, which often degrades their performances.

To explicitly handle the noises, Ye *et al.* (Ye *et al*. 2012) proposed a robust data fusion method for object categorization, which can be regarded as a method for rank aggregation. Given n input rank lists, their method constructs n pairwise comparison matrices, each of which is a shared rank-2 matrix combined with independent-sparse noises, and then recovers the rank-2 matrix and extracts the final rank list from this matrix. However, their method requires the input rank lists to be complete (with no missing values), which is limited in real world scenarios of rank aggregation.

In this paper, we propose the ***R**obust **R**ank **A**ggregation* (RRA), a novel method that simultaneously handles the possible noises as well as missing values in the individual rank lists. As shown in Fig.1, we firstly use a pairwise comparison matrix to encode the (possibly incomplete) order information in each individual rank list. Then, based on our observations, we show that this comparison matrix can be naturally decomposed into (1) a shared latent matrix with low rank, and (2) a deviation error matrix which is the combination of a column-sparse matrix and a row-sparse matrix. The final rank list can be easily extracted from the learned low-rank matrix. Consequently, we propose to formulate rank aggregation as a problem of low-rank and structured-sparse decomposition. Different from the formulation in (Ye *et al*. 2012), our formulation introduces the element-wise multi-

[0]This work was funded in part by National Science Foundation of China (grant No. 61003045, 61033010, 61272067), Natural Science Foundation of Guangdong Province, China (grant No. 10451027501005667, S2012030006242), Educational Commission of Guangdong Province, China, Shenzhen Special Development Fund for Emerging Strategic Industries (grant No. JCYJ20120615151826623), and the Fundamental Research Funds for the Central Universities. Corresponding author: Yong Tang.

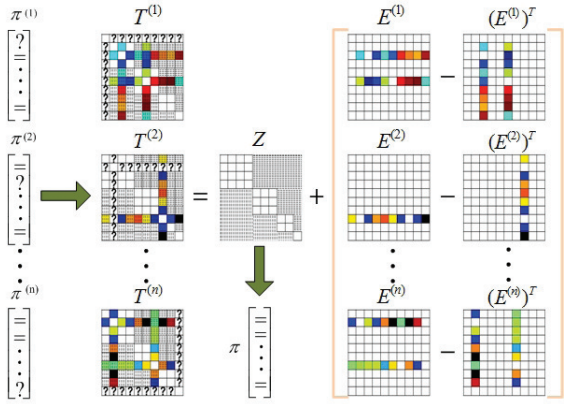

Figure 1: Overview of Robust Rank Aggregation (RRA). Given n *possibly incomplete and noisy* input rank lists $\pi^{(i)}(i = 1, 2, ..., n)$, RRA constructs n matrices $T^{(i)}(i = 1, 2, ..., n)$, where each $T^{(i)}$ encodes the pairwise relations between every two items in $\pi^{(i)}$. Then $T^{(i)}(i = 1, 2, ..., n)$ are used as input of RRA to learn a shared, low-rank matrix Z. Each $T^{(i)}$ can be reconstructed by Z combining with an additive sparse matrix $N^{(i)}$, which can be decomposed into a column-sparse matrix $E^{(i)}$ and its negative transpose $(-(E^{(i)})^T)$. At last, RRA recovers a final and more accurate rank list π from Z.

plication to handle the missing values, a symmetric constraint on the noise structure, and a factorization trick to explicitly restrict the maximum rank of the shared latent matrix, resulting in a considerably more challenging optimization problem. We propose an algorithm to solve the optimization problem based on the Augmented Lagrangian Multiplier (ALM) scheme (Lin *et al*. 2010). Experimental results on benchmark datasets of meta-search and collaborative filtering show that the proposed RRA outperforms several state-of-the-art rank aggregation methods.

Related Work

Rank aggregation ia a classical problem which can be traced back to the 1780s in social choice research (Borda 1781). Recently, rank aggregation attracts much attention again in modern applications such as meta-search (Dwork *et al*. 2001), recommendation systems (Gleich and Lim 2011), spam webpage detection (Dwork *et al*. 2001) and machine translation (Rosti *et al*. 2007).

The rank aggregation methods can be categorized into two main streams. The first stream is the score-based methods (Manmatha *et al*. 2001; Montague and Aslam 2001; Gleich and Lim 2011), where each item in an individual rank list is assigned a score. The second stream is the order-based methods (Dwork *et al*. 2001; Fagin *et al*. 2003; Liu *et al*. 2007; Qin *et al*. 2010a), where only the order relations of the items in the individual rank list are used. In real world applications, methods in the second stream are more popular because (1) the scores in some input rank lists might not be sufficiently accurate and less reliable than the order information, and (2) in some applications such as meta-search, only the order information of the results from individual search engines is available. Our proposed method belongs to the second stream.

The representative order-based methods for rank aggregation include: BordaCount (Borda 1781), median rank aggregation (Fagin *et al*. 2003), Markov Chain based methods (Dwork *et al*. 2001), probabilistic rank aggregation (Qin *et al*. 2010a) and matrix completion based rank aggregation (Gleich and Lim 2011). In BordaCount (Borda 1781), each item is sorted by counting the number of items ranked before it in all the individual rank lists. Median rank aggregation (Fagin *et al*. 2003) sorts the items by their median ranks in all the individual rank lists. The Markov Chain based methods (Dwork *et al*. 2001) learns a Markov Chain on the items using the order information in the individual rank lists, and then uses the stationary distribution of the learned Markov Chain to sort the items. Probabilistic rank aggregation (Qin *et al*. 2010a) uses a distance-based probabilistic model on permutation to model the generation of a rank list. The matrix completion based method (Gleich and Lim 2011) constructs a possibly incomplete pairwise comparison matrix by averaging the pairwise relation between items over all the individual rank lists, and then recovers a rank-2 matrix from the pairwise comparison matrix, finally the items are sorted by a score list exacted from the recovered matrix. However, these methods blindly combine the individual rank lists that might be corrupted by considerable noises, which often degrades their performances.

Ye *et al.* (Ye *et al*. 2012) proposed a robust data fusion method for object categorization, which explicitly handles the noises in the individual rank lists. However, their method requires all the input rank lists to be complete and cannot be applied to partial rank aggregation in real world scenarios.

Robust Rank Aggregation

General framework

We first present the general framework for robust rank aggregation via matrix learning. Given m items and n (incomplete) individual rank lists $\pi^{(1)}, \pi^{(2)}, ..., \pi^{(n)}$ (each of which is a partial ordered list of the m items), the task of rank aggregation is to obtain a better rank list by combining these individual rank lists. Fig. 1 illustrates the framework of the proposed RRA. We first convert each $\pi^{(i)}$ into an $m \times m$ comparison matrix $T^{(i)}$, in which each $T^{(i)}_{j,k}$ is defined as:

$$T^{(i)}_{j,k} = \begin{cases} sign(\pi^{(i)}_j - \pi^{(i)}_k) & \text{if } \pi^{(i)}_j \text{ and } \pi^{(i)}_k \text{ are observed} \\ unknown & \text{if } \pi^{(i)}_j \text{ or } \pi^{(i)}_k \text{ is missing} \end{cases}$$

where $\pi^{(i)}_j/\pi^{(i)}_k$ denotes the rank position of j-th/k-th item in $\pi^{(i)}$ (for convenience, we assume top items in a rank list have larger rank positions). To handle the possible missing values in the input rank lists, we define an indicator matrix $W^{(i)}$ of the same size as $T^{(i)}$, where $W^{(i)}_{j,k} = 0$ if $T^{(i)}_{j,k}$ is unknown, otherwise $W^{(i)}_{j,k} = 1$. Note that each $W^{(i)}$ is symmetric (i.e., $W_{j,k} = W_{k,j}$).

In real world applications such as meta-search and recommendation system, the quality of each item is usually measured by some relevance level. For example, in meta-search, the relevance of the web pages is measured by five levels:

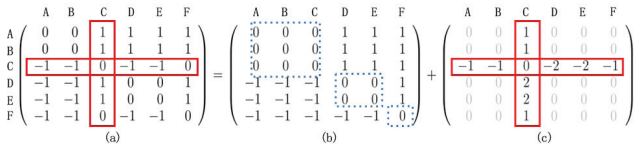

Figure 2: An illustration of the low-rank pattern in the shared latent matrix, and the structure-sparse pattern in the error matrix caused by noises. (a) An example of comparison matrix with an item being in an incorrect position. (b) The shared latent matrix is block-diagonal and of low-rank. (c) The pattern in the error matrix caused by an item being in an incorrect position. Please refer to the text in Section *Low-rank and structured-sparse modeling* for details.

perfect, excellent, good, fair and bad; in movie recommendation systems, the quality of each movie is measured by $1 \sim 5$ stars. Assume that $L(i)$ is the fixed but unknown relevance level of the i-th item, $L(i) \in \{1, 2, ..., k\}$. The larger $L(i)$ is, the more relevant the i-th item is. Ideally, the target pairwise comparison matrix $Z \in \mathcal{R}^{m \times m}$ that encodes the true order relations between items should be defined by:

$$Z_{j,k} = sign(L(j) - L(k)).$$

Each comparison matrix $T^{(i)}$ is an isotonic representation that encodes the relative order relations between the items. Some entries in $T^{(i)}$ correctly reflect the relative order between the items, while other entries may be incorrect due to the wrong predictions in the individual rank list $\pi^{(i)}$. $T^{(i)}$ can be viewed as a combination of two parts: the target pairwise comparison matrix Z that reflects the correct relations between the items, and a deviation error matrix $N^{(i)}$ that encodes the errors made by the incorrect entries in $\pi^{(i)}$. Concretely, for all i, we have

$$W^{(i)} \odot T^{(i)} = W^{(i)} \odot (Z + N^{(i)}). \quad (1)$$

where $\odot$ denote element-wise (Hadamard) multiplication.

If Z is given, we can easily obtain a score list z on the items by $z = \frac{1}{m} Z e^T$ (Gleich and Lim 2011) (where e is a m-dimensional vector with all ones), and then get the final rank list by sorting the items using z.

To sum up, given $T^{(i)}$ $(i = 1, 2, ..., n)$, the central task of robust rank aggregation is to infer the latent matrix Z.

Low-rank and structured-sparse modeling

A key question raised by the framework of RRA is how to model the structures in the shared matrix Z and each deviation error matrix $N^{(i)}$. To answer this question, without loss of generality, we assume each item has a fixed but unknown relevance level from $\{1, 2, ..., k\}$. Then we will show that, by this assumption, (1) the target pairwise comparison matrix Z is naturally of low-rank, and (2) each error matrix $N^{(i)}$ has a structured-sparse pattern.

Firstly, as illustrated in Fig. 2(b), given a list of 6 items whose relevance levels are $3, 3, 3, 2, 2, 1$, the corresponding pairwise comparison matrix has a block-diagonal structure. In this matrix, since the first three columns are identical and the 4th and 5th columns are identical, it is obvious that the rank of this matrix is 3. In general, for any pairwise comparison matrix generated by an item list associated with k relevance levels, we can convert it into a block-diagonal with k blocks (note that this conversion does not change the rank of the matrix because exchanging two columns/rows of any matrix A does not change the rank of A), and the rank of the block-diagonal matrix is not larger than k. This motivates us to assume the underlying pairwise comparison matrix Z in Eq. (1) to be of low-rank. Note that this is different from the rank-2 assumption on T within score-based settings in (Gleich and Lim 2011).

Secondly, in practice, an input rank list $\pi^{(i)}$ might be partially incorrect, resulting in incorrect order information in $T^{(i)}$. Concretely, the corruptions in an individual rank list can be regarded as some items being in incorrect rank positions. For example, Fig. 2 is an illustration of the effect caused by putting item C in an incorrect rank position. More specifically, Assume the true rank list is $(A\ B\ C) \rightarrow (D\ E) \rightarrow F$ (associated with the true comparison matrix in Fig. 2(b)). If we change item C from the 3rd position to the 5th position, then the rank list becomes $(A\ B) \rightarrow (D\ E) \rightarrow (C\ F)$ and the comparison matrix in Fig. 2(b) is changed to the one in Fig. 2(a). The errors caused by this position change are in the matrix of Fig. 2(c), which can be decomposed into a column $[1, 1, 0, 2, 2, 1]^T$ and a row $[-1, -1, 0, -2, -2, -1]$, where the row is exactly the negative transpose of the column. In general, if there are a small fraction of items being in incorrect positions within an individual rank list, the corresponding error matrix ($N^{(i)}$ in Eq. (1)) is the combination of a column-sparse matrix and its negative transpose (a row-sparse matrix). More specifically, for an input pairwise matrix $T^{(i)}$, we have:

$$W^{(i)} \odot T^{(i)} = W^{(i)} \odot (Z + E^{(i)} - (E^{(i)})^T),$$

where $E^{(i)}$ is a column-sparse matrix, and its transpose $(E^{(i)})^T$ is a row-sparse one.

Problem formulation

With the low-rank and column-sparse assumptions, RRA can be formulated as

$$\begin{aligned} &\min_{Z, E^{(i)}} ||Z||_* + \lambda \sum_{i=1}^{n} ||E^{(i)}||_{2,1} \\ &s.t.\ W^{(i)} \odot T^{(i)} = W^{(i)} \odot (Z + E^{(i)} - (E^{(i)})^T) \\ &\quad (i = 1, 2, ..., n), \end{aligned} \quad (2)$$

where the trace norm regularization term $||Z||_*$ induces the desirable low-rank structure in the matrix Z, the $\ell_{2,1}$-norm regularization term $||E^{(i)}||_{2,1} = \sum_{k=1}^{m} \sqrt{\sum_{j=1}^{m} (E^{(i)}_{j,k})^2}$ which encourages the column-sparsity in $E^{(i)}$ (correspondingly, it also encourages the row-sparsity in $(E^{(i)})^T$), and λ is a non-negative trade-off parameter.

Note that the trace norm in a matrix Z is a well-known convex surrogate of the rank in Z. Hence, minimizing the trace norm in Z does not guarantee the rank of Z not exceeding a pre-specified value (although the trace norm usually encourages a relative low-rank structure). However, in rank aggregation, it is desirable to restrict the learned matrix Z to have a pre-specified rank. For example, assuming there

are k relevance levels in meta-search, it is ideally to restrict the rank of Z to be k[1](see the explanation in the previous subsection).

To address this issue, we propose to factorize the pairwise comparison matrix $Z \in \mathbb{R}^{m\times m}$ into the product of two size-reduced matrices (i.e.,$m \times k$ and $k \times m$). Similar methods (Liu *et al*. 2012; Liu and Yan 2012) take advantage of a fixed rank for a different purpose, i.e., to accelerate the optimization for low-rank matrix learning problems such as RPCA (Candes *et al*. 2011) and LRR (Liu *et al*. 2010).

More specifically, we assume the factorization be $Z = QJ$ where $Q \in \mathbb{R}^{m\times k}$ is column-orthogonal and $J \in \mathbb{R}^{k\times m}$. Then the problem in Eq. (2) can be rewritten as:

$$\begin{aligned} &\min_{Z,Q,J,E^{(i)}} ||J||_* + \lambda \sum_{i=1}^{n} ||E||_{2,1} \\ &s.t.\ W^{(i)} \odot T^{(i)} = W^{(i)} \odot (Z + E^{(i)} - (E^{(i)})^T), \\ &\quad Z = QJ,\ Q^TQ = I_r (i = 1, 2, ..., n). \end{aligned} \tag{3}$$

where I_r is the $r \times r$ identity matrix, and we use the fact that $||Z||_* = ||QJ||_* = ||J||_*$ because Q is column-orthogonal.

Optimization Procedure

In the section, we propose a novel procedure to solve the optimization problem in Eq. (3) based on the Augmented Lagrangian Multiplier (ALM) scheme (Lin *et al*. 2010). ALM has shown a good balance between efficiency and accuracy in many matrix learning problems.

Algorithm based on ALM

We first convert Eq. (3) into the following equivalent problem:

$$\begin{aligned} &\min_{Z,Q,J,E^{(i)},F^{(i)}} ||J||_* + \lambda \sum_{i=1}^{n} ||E||_{2,1} \\ &s.t.\ W^{(i)} \odot T^{(i)} = W^{(i)} \odot (Z + F^{(i)} - (F^{(i)})^T), Z = QJ, \\ &\quad Q^TQ = I_r,\ F^{(i)} = E^{(i)}\ (i = 1, 2, ..., n), \end{aligned} \tag{4}$$

where we introduce the auxiliary variables $F^{(i)}$ to make it easier to solve the corresponding subproblems.

The corresponding Lagrangian function of Eq. (4) is:

$$\begin{aligned} &\mathcal{L}(Q, J, Z, E^{(i)}, F^{(i)}) = ||J||_* + \lambda \sum_{i=1}^{n} ||E^{(i)}||_{2,1} \\ &+ \frac{\mu}{2} ||W^{(i)} \odot (T^{(i)} - Z - F^{(i)} + (F^{(i)})^T)||_F^2) \\ &+ \sum_{i=1}^{n} (\langle X^{(i)}, W^{(i)} \odot (T^{(i)} - Z - F^{(i)} + (F^{(i)})^T)\rangle \\ &+ \frac{\mu}{2} \sum_{i=1}^{n} ||F^{(i)} - E^{(i)}||_F^2) + \sum_{i=1}^{n} (\langle Y_i, F^{(i)} - E^{(i)}\rangle \\ &+ \frac{\mu}{2} ||Z - QJ||_F^2 + \langle L, Z - QJ\rangle\ \ s.t.\ \ Q^TQ = I_r, \end{aligned}$$

where $X^{(i)}$, $Y^{(i)}$ and L are Lagrangian multipliers.

[1]However, if there is large variation in the input rank lists, Z might have an unknown rank larger than k. Hence, one can also set the rank of Z to be $k + p$ where p is a small integer.

Algorithm 1 shows the sketch of the proposed algorithm. Next, we will present the update rule for each of Z, Q, J, $E^{(i)}$, $F^{(i)}$ when other variables being fixed. Please refer to Algorithm 1 for the details.

Algorithm 1 Robust Rank Aggregation via ALM

Input: $\lambda, T^{(i)} \in \mathbb{R}^{m\times m} (i = 1, \ldots, n)$.
Initialize:All zeros for $Q, J, Z, L, E^{(i)}, F^{(i)}, X^{(i)}, Y^{(i)}$ (i=1,...,n). $\mu = 10^{-6}, \rho = 1.9, \epsilon = 10^{-8}$.
Repeat
Updating Q via Eq.(6).
Updating J via Eq.(8).
Updating Z via Eq.(10).
Updating $E^{(i)}$ via Eq.(11).
Updating $F^{(i)}$ via Eq.(15).
$L \leftarrow L + \mu(Z - QJ)$
$X^{(i)} \leftarrow X^{(i)} + \mu W^{(i)} \odot (T^{(i)} - Z - F^{(i)} + (F^{(i)})^T)$
$Y^{(i)} \leftarrow Y^{(i)} + \mu(F^{(i)} - E^{(i)})$
$\mu \leftarrow \min(\rho\mu, 10^{10})$.
Until $||Z - QJ||_\infty \le \epsilon, ||F^{(i)} - E^{(i)}||_\infty \le \epsilon$ and $||W^{(i)} \odot (T^{(i)} - Z - F^{(i)} + (F^{(i)})^T)||_\infty) \le \epsilon$.
Output: $Z, E^{(i)} (i = 1, \ldots, n)$.

Solving Q

When other variables are fixed, the subproblem w.r.t. Q is:

$$\min_{Q} \frac{\mu}{2} ||Z - QJ + L/\mu||_F^2\ \ s.t.\ \ Q^TQ = I_r. \tag{5}$$

This is the well-known orthogonal procrustes problem, whose solution can be derived from the Singular Value Decomposition (SVD) of $(Z+L/\mu)J^T$. Concretely, if $U_1\Sigma_1V_1$ is the SVD form of $(Z + L/\mu)J^T$, then the solution to Eq. (5) is given by:

$$Q = U_1V_1^T. \tag{6}$$

Solving J

The subproblem w.r.t. J is simplified as:

$$\min_{J} ||J||_* + \frac{\mu}{2} ||Z - QJ + L/\mu||_F^2. \tag{7}$$

This can be solved by the Singular Value Threshold method (Cai *et al*. 2010). Specifically, by assuming $U_2\Sigma_2V_2$ be the SVD form of $Q^T(Z + L/\mu)$, the solution to E.q. (7) is given by:

$$J = U_2\mathcal{S}_{1/\mu}(\Sigma_2)V_2^T, \tag{8}$$

where $\mathcal{S}_{1/\mu}(X) = \max(0, X + 1/\mu) + \min(0, X - 1/\mu)$ is the shrinkage operator (Lin *et al*. 2010).

Solving Z

When other variables are known, we update Z by solving:

$$\begin{aligned} &\min_{Z} \sum_{i=1}^{n} ||W^{(i)} \odot (T^{(i)} - Z - F^{(i)} + (F^{(i)})^T + X^{(i)}/\mu)||_F^2 \\ &+ ||Z - QJ + L/\mu||_F^2 \end{aligned} \tag{9}$$

By setting the derivative of Eq. (9) w.r.t. Z to be zero, we have $\sum_{i=1}^{n}(W^{(i)} \odot (T^{(i)} - Z - F^{(i)} + (F^{(i)})^T + X^{(i)}/\mu) - (Z - QJ + L/\mu) = 0$. Simple algebra operations yield:

$$Z = (QJ - \frac{L}{\mu} + \sum_{i=1}^{n} W^{(i)} \odot (T^{(i)} - F^{(i)} + (F^{(i)})^T + \frac{X^{(i)}}{\mu})) \oslash (\sum_{i=1}^{n} W^{(i)} + \mathbf{1_{m \times m}}), \tag{10}$$

where $\oslash$ denotes element-wise division (i.e., if $C = A \oslash B$ for matrices A, B and C, then we have $\forall i,\ j,\ C_{ij} = A_{ij}/B_{ij}$) and $\mathbf{1_{m \times m}}$ denotes an $m \times m$ matrix with all ones.

Solving $E^{(i)}$

With other variables fixed, we update each of $E^{(i)}$ $(i = 1, 2, ..., n)$ by solving

$$\min_{E^{(i)}} \lambda ||E^{(i)}||_{2,1} + \frac{\mu}{2} ||F^{(i)} - E^{(i)} + Y^{(i)}/\mu||_F^2.$$

This problem has the following closed-form solution (see e.g., Lemma 3.2 in (Liu *et al*. 2010)):

$$E^{(i)}(:, j) = \begin{cases} \frac{||K(:,j)||_2 - \lambda/\mu}{||K(:,j)||_2} K(:, j) & \text{if } \lambda/\mu < ||K(:, j)||_2, \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

where $K = F^{(i)} + Y^{(i)}/\mu$, and $K(:, j)$ $(E(:, j))$ is the j-th column of K (E), respectively.

Solving $F^{(i)}$

Given other variables, each of $F^{(i)}$ is updated by solving

$$\min_{F^{(i)}} ||W^{(i)} \odot (T^{(i)} - Z - F^{(i)} + (F^{(i)})^T + X^{(i)}/\mu)||_F^2 + ||F^{(i)} - E^{(i)} + Y^{(i)}/\mu||_F^2. \tag{12}$$

By setting the derivative of Eq. (12) w.r.t. $F^{(i)}$ to be zero and using the fact that $W^{(i)}$ is symmetric, we have $2W^{(i)} \odot (F^{(i)} - (F^{(i)})^T) + F^{(i)} = E^{(i)} - Y^{(i)}/\mu + W^{(i)} \odot [(T^{(i)} - Z + X^{(i)}/\mu) - (T^{(i)} - Z + X^{(i)}/\mu)^T]$. Let $C = E^{(i)} - Y^{(i)}/\mu + W^{(i)} \odot [(T^{(i)} - Z + X^{(i)}/\mu) - (T^{(i)} - Z + X^{(i)}/\mu)^T]$. Then we have

$$6W^{(i)} \odot (F^{(i)} - (F^{(i)})^T) + 3F^{(i)} = 3C, \tag{13}$$

and

$$4W^{(i)} \odot ((F^{(i)})^T - F^{(i)}) + 2(F^{(i)})^T = 2C^T. \tag{14}$$

By summing up Eq. (13) and Eq. (14), we have:

$W^{(i)} \odot (2F^{(i)} - 2(F^{(i)})^T) + 3F^{(i)} + 2(F^{(i)})^T = 3C + 2C^T$.

Finally, simple algebra operations yields the solution to $F^{(i)}$:

$$F^{(i)} = \frac{1}{5} W^{(i)} \odot (3C + 2C^T) + (\mathbf{1_{m \times m}} - W^{(i)}) \odot C, \tag{15}$$

where $\mathbf{1_{m \times m}}$ denotes an $m \times m$ matrix with all ones.

Remarks. Please note that the objective in Eq. (4) is non-convex due to the bilinear form QJ. To the best of our knowledge, there is no convergence proof of inexact ALM for a non-convex objective. However, we empirically found that the proposed RRA (based on ALM) converges with less than 200 iterations in all our experiments, regardless of the initialization.

Experiments

In this section, we evaluate the accuracy of the proposed RRA on meta-search and collaborative filtering benchmarks. We compare RRA with five rank aggregation baselines, including the BordaCount algorithm, two Markov Chain based algorithms (MC_2 and MC_4) in (Dwork *et al*. 2001), a probabilistic rank aggregation algorithm (CPS) in (Qin *et al*. 2010a), and an algorithm based on matrix completion (SVP) in (Gleich and Lim 2011). Note that the method in (Ye *et al*. 2012) is not used as a baseline because it cannot handle the input rank lists with missing values. In addition, we conduct experiments to investigate the parameter sensitivity of RRA.

Results on meta-search

We evaluate and compare the algorithms on MQ2007-agg and MQ2008-agg datasets, which are from the LETOR 4.0 benchmark (Qin *et al*. 2010b) for meta-search.

There are 1700 (800) queries and 21 (25) input rank lists in MQ2007-agg (MQ2008-agg). Each query has $6 \sim 147$ documents. There are three relevance levels for the documents. There exist missing values in the individual rank lists. To compare the performances, we use the standard Mean Average Precision (MAP) (Baeza-Yates and Ribeiro-Neto 1999) and Normalized Discounted Cumulative Gain (NDCG) (Jarvelin and Kekalainen 2002) as the evaluation measures.

The results of BordaCount and CPS are cited from the LETOR web site [2]. The results of SVP are obtained by directly running its open source code [3]. The results of MC_2 and MC_4 are obtained by our careful implementations. We set $r = 3$ and $\lambda = 10^{-2}$ in RRA.

As shown in Table 1, compared to the baseline algorithms, RRA achieves substantially better performances on both datasets. Here are some statistics. On MQ2007-agg, the results of RRA indicate a relative increase of **24.5**%**, 12.6**%**, 11.1**%**, 6.4**% w.r.t. NDCG@1,@3,@5,@10, respectively, compared to the second best algorithm. On MQ2008-agg, RRA has a relative increase of **37.2**%**, 15.0**%**, 4.3**%**, 8.3**% w.r.t. NDCG@1,@3,@5,@10, respectively, compared to the second best algorithm.

Table 2: Comparison results on Movielen100K.

	d_K	d_S
BordaCount	0.3010	0.4581
MC_2	0.3015	0.4234
MC_4	0.2941	0.4102
SVP	0.2912	0.4065
RRA	**0.2798**	**0.3877**

Results on collaborative filtering

We also test the performance of RRA to find out the top movies from the MovieLens100K dataset, which is a col-

[2] http://research.microsoft.com/en-us/um/beijing/projects /letor/letor4baseline.aspx. There are three variants of CPS. Here we use CPS-SpearmanRankCorrelation which has the best results.

[3] http://www.cs.purdue.edu/homes/dgleich/. Arithmetic mean of score differences is used to average the input matrices.

Table 1: Comparison results on the meta search datasets from LETOR 4.0. We denote NDCG@i as N@i for short.

	N@1	N@2	N@3	N@4	N@5	N@6	N@7	N@8	N@9	N@10	MAP
MQ2007-agg											
BordaCount	0.1902	0.2014	0.2081	0.2128	0.2188	0.2247	0.2312	0.2377	0.2444	0.2507	0.3252
MC_2	0.3263	0.3493	0.3631	0.3696	0.3759	0.3832	0.3895	0.3968	0.4027	0.4085	0.4285
MC_4	0.3198	0.3173	0.3049	0.3117	0.3211	0.3271	0.3340	0.3416	0.3492	0.3568	0.3910
CPS	0.3196	0.3318	0.3386	0.3409	0.3476	0.3519	0.3573	0.3616	0.3668	0.3718	0.4069
SVP	0.2794	0.2937	0.3060	0.3146	0.3223	0.3324	0.3413	0.3488	0.3576	0.3630	0.3982
RRA	**0.4063**	**0.4044**	**0.4090**	**0.4141**	**0.4177**	**0.4213**	**0.4237**	**0.4274**	**0.4310**	**0.4348**	**0.4367**
MQ2008-agg											
BordaCount	0.2368	0.2806	0.3080	0.3432	0.3713	0.3888	0.3992	0.3724	0.1643	0.1694	0.3945
MC_2	0.2351	0.2616	0.2942	0.3258	0.3553	0.3768	0.3923	0.3674	0.1581	0.1658	0.3804
MC_4	0.2478	0.2169	0.2439	0.2767	0.3077	0.3298	0.3460	0.3255	0.1318	0.1383	0.3481
CPS	0.2652	0.3138	0.3459	0.3763	0.4004	0.4192	0.4280	0.3975	0.1828	0.1864	0.4102
SVP	0.2670	0.3267	0.3520	0.3720	0.3914	0.4112	0.4237	0.3964	0.1754	0.1795	0.4079
RRA	**0.3664**	**0.3989**	**0.4051**	**0.4103**	**0.4180**	**0.4314**	**0.4451**	**0.4191**	**0.1973**	**0.2019**	**0.4227**

laborative filtering benchmark collected through the MovieLens web site[4]. MovieLens100K consists of 100,000 ratings from 943 users on 1682 movies. All the ratings are integers of $1 \sim 5$. Each user is an incomplete input rank list for rank aggregation. Existing theoretical results show that it is difficult to recover a low-rank matrix from an input matrix with too many missing values (Candes *et al*. 2011). Hence, we only use a subset of input rank lists, each of which has at least 80 ratings. The results of SVP are obtained by running its open source code. Other results are obtained by our careful implementations. For the parameters of RRA, we empirically set $\lambda = 100$ and $r = 5$.

Since there is no "ground truth" for the rank list of the movies, we follow (Dwork *et al*. 2001) to use *induced Kendall's tau distance* (d_K) and *induced Spearman's footrule distance* (d_S) for partial rank lists as the evaluation metrics. Given π as the output rank list of a rank aggregation algorithm, d_K and d_S measure the average distance of π to every input rank list $\pi^{(1)}, \ldots, \pi^{(n)}$[5].

As shown in Table 2, RRA has a **3.9**% (**4.6**%) improvement w.r.t. d_k (d_S) over the second best SVP algorithm.

Parameter sensitivity

There are two parameters (λ and r) in RRA. A natural question arising here is whether the performance of RRA is sensitive to these parameters. To answer this question, we conduct experiments on MQ2007-agg and MQ2008-agg to investigate the effects with different values of λ and r.

Firstly, we compare the performances of RRA with different values of λ. By fixing $r = 3$, Table 3 lists the MAP and NDCG@10 with λ being chosen from $\{10^{-4}, 10^{-3}, ..., 10^{3}\}$.

As can be seen, on MQ2007-agg, the values of MAP/NDCG@10 are close when λ is in the range $[10^{-4} \sim 10^{-1}]$. Similarly, on MQ2008-agg, the values of MAP/NDCG@10 are close when λ is chosen from $[10^{-4} \sim 10^{0}]$. The results indicate that we need to set a relatively small λ in this task, and RRA is insensitive to λ as long as λ is chosen from a suitable range (i.e., $[10^{-4} \sim 10^{-1}]$).

Table 3: Results of RRA with different values of λ and r=3. We denote NDCG@i as N@i for short.

	MQ2007-agg		**MQ2008-agg**	
	MAP	N@10	MAP	N@10
$\lambda = 10^{-4}$	**0.4338**	**0.4390**	**0.4162**	**0.1953**
$\lambda = 10^{-3}$	**0.4348**	**0.4379**	**0.4150**	**0.1958**
$\lambda = 10^{-2}$	**0.4367**	**0.4348**	**0.4227**	**0.2019**
$\lambda = 10^{-1}$	**0.4363**	**0.4286**	**0.4248**	**0.2017**
$\lambda = 10^{0}$	0.4283	0.4129	**0.4190**	**0.1978**
$\lambda = 10^{1}$	0.4220	0.4016	0.4110	0.1905
$\lambda = 10^{2}$	0.4175	0.3930	0.4025	0.1841
$\lambda = 10^{3}$	0.4137	0.3874	0.3985	0.1802

Secondly, we explore the performance of RRA with different values of r. Note that there are three relevance levels in both MQ2007-agg and MQ2008-agg. Hence, the value of r should be close to 3. In this experiment, we fix $\lambda = 10^{-2}$ and choose r from $\{2, 3, 5, 10, 15\}$.

As shown in Table 4, on both MQ2007-agg and MQ2008-agg, the values of MAP/NDCG@10 are close when r is in $[3 \sim 15]$. This indicates that RRA is insensitive to r as long as r is in an appropriate range.

In summary, RRA is insensitive to its parameters as long as each parameter is chosen from a suitable range. This is beneficial for the unsupervised RRA because we do not need much effort for parameter tuning.

Table 4: Comparison results with different values of r and $\lambda = 10^{-2}$. We denote NDCG@i as N@i for short.

	MQ2007-agg		**MQ2008-agg**	
	MAP	N@10	MAP	N@10
$r = 2$	0.3824	0.3636	0.4080	0.1857
$r = 3$	**0.4367**	**0.4348**	**0.4227**	**0.2019**
$r = 5$	**0.4393**	**0.4369**	**0.4253**	**0.1996**
$r = 10$	**0.4392**	**0.4363**	**0.4249**	**0.1991**
$r = 15$	**0.4392**	**0.4364**	**0.4242**	**0.1979**

[4]http://movielens.umn.edu

[5]Please refer to Section 2.1.1 in (Dwork *et al*. 2001) for the detailed description of these metrics

Conclusions

In this paper, we developed RRA, a rank aggregation algorithm that explicitly handles the possible noises and missing values in the input rank lists via low-rank and structured-sparse decomposition. To solve the optimization problem of RRA, we proposed a procedure based on the ALM scheme. Extensive experiments in meta-search and collaborative filtering show that RRA has encouraging performance gain over the state-of-the-arts, and RRA is insensitive to its parameters as long as the parameters are in suitable ranges, which makes the algorithm easy to use without much effort for parameter tuning.

References

R. Baeza-Yates, and B. Ribeiro-Neto. *Modern information retrieval*, ACM press New York, 1999.

J.C. Borda. Mémoire sur les élections au scrutin. *Histoire de l'Académie Royale des Sciences*, 1781.

E. Candes, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM,*, 58(1):1-37, 2011.

J. Cai, E. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956-1982, 2010.

C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613-622, 2001.

R. Fagin, R. Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation. In *Proceedings of the ACM SIGMOD International Conference on Management of data*, pages 301-312, 2003.

D. F. Gleich, and L. H. Lim. Rank aggregation via nuclear norm minimization. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 60-68, 2011.

K. Jarvelin, and J. Kekalainen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422-446, 2002.

A. Klementiev, D. Roth, and K. Small. Unsupervised Rank Aggregation with Distance-Based Models. In *Proceedings of the 25th International Conference on Machine Learning*, pages 472-479, 2008.

Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv preprint arXiv:1009.5055*, 2010.

Y. Liu, T. Liu, T. Qin, Z. Ma, and H. Li. Supervised rank aggregation. In *Proceedings of the 16th international conference on World Wide Web*, pages 481-490, 2007.

G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *Proceedings of the 27th International Conference on Machine Learning*, pages 663-670, 2010.

R. Liu, Z. Lin, F. Torre, and Z. Su. Fixed-Rank Representation for Unsupervised Visual Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 598-605, 2012.

G. Liu, and S. Yan. Active Subspace: Towards Scalable Low-Rank Learning. *Neural Computation*, 24(12):3371-3394, 2012.

R. Manmatha, T. Rath, and F. Feng. Modeling score distributions for combining the outputs of search engines. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and Development in Information Retrieval*, pages 267-275, 2001.

M. Montague, and J. A. Aslam. Relevance score normalization for metasearch. In *Proceedings of the tenth International Conference on Information and Knowledge Management*, pages 427-433, 2001.

T. Qin, X. Geng, and T. Liu. A new probabilistic model for rank aggregation. In *Proceedings of the Neural Information Processing Systems*, pages 681-689, 2010.

T. Qin, T. Liu, J. Xu, and H. Li. Letor: A benchmark collection for research on learning to rank for information retrieval. *Information Retrieval*, 13(4):346-374, 2010.

A. Rosti, N. Ayan, B. Xiang, S. Matsoukas, R. Schwartz, and B. Dorr. Combining outputs from multiple machine translation systems. In *Proceedings of the Human Language Technologies and the North American Chapter of the Association for Computational Linguistics*, pages 228-235, 2007.

G. N. Ye, D. Liu, I. H. Jhuo, and S.F. Chang. Robust Late Fusion with Rank Minimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3021-3028, 2012.