LOCALIZING USERS AND ITEMS FROM PAIRED COMPARISONS

Matthew R. O'Shaughnessy and Mark A. Davenport

Georgia Institute of Technology
School of Electrical and Computer Engineering
{moshaughnessy6, mdav}@gatech.edu

ABSTRACT

Suppose that we wish to determine an embedding of points given only *paired comparisons* of the form "user $\boldsymbol{x}$ prefers item $\boldsymbol{q}_i$ to item $\boldsymbol{q}_j$." Such observations arise in a variety of contexts, including applications such as recommendation systems, targeted advertisement, and psychological studies. In this paper we first present an optimization-based framework for localizing new users and items when an existing embedding is known. We demonstrate that user localization can be formulated as a simple constrained quadratic program, and that although item localization produces a set of non-convex constraints, we can make the problem convex by strategically combining comparisons to produce a set of convex linear constraints. Finally, we show that by iteratively applying this method to every user and item, we can recover an accurate embedding, allowing us to iteratively improve a given embedding or even generate an embedding from scratch.

Index Terms— localization, paired comparisons, non-metric multidimensional scaling, ideal point models, recommendation systems

1. INTRODUCTION

In this paper we consider several problems related to learning an embedding of points from *paired comparisons* of the form "$\boldsymbol{x}$ is closer to $\boldsymbol{q}_i$ than $\boldsymbol{q}_j$," where $\boldsymbol{x}, \boldsymbol{q}_i, \boldsymbol{q}_j \in \mathbb{R}^n$ correspond to points whose locations we would like to estimate. Observations of this form arise in a variety of contexts, but a particularly important class of applications involve recommendation systems, targeted advertisement, and psychological studies where $\boldsymbol{x}$ represents an *ideal point* that models a particular user's preferences and $\boldsymbol{q}_i$ and $\boldsymbol{q}_j$ represent items that the user compares. In this model, which dates back at least to [1], items which are close to $\boldsymbol{x}$ are those most preferred by the user. Paired comparisons arise naturally in this context since precise numerical scores quantifying a user's preference are generally much more difficult to assign than comparative judgments [2]. Moreover, data consisting of paired comparisons is often generated implicitly in contexts where the user has the option to act on two (or more) alternatives; for instance, they may choose to watch a particular movie or click a particular advertisement out of those displayed to them [3]. In such contexts, the "true distances" in the ideal point model are generally inaccessible in any direct way, but it is nevertheless still possible to learn useful information from such data.

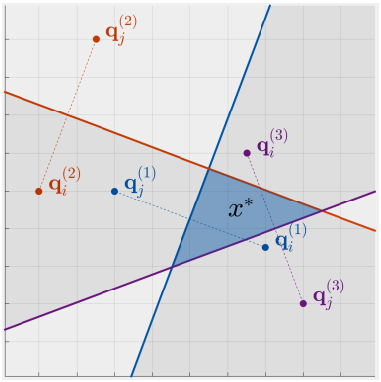


Fig. 1: Set of linear constraints induced by the binary paired comparisons. Shading represents the feasible region resulting from the set of comparisons.

Some of the simplest questions in this model arise in the setting where we are given an existing embedding of users/items (as in a mature recommender system) and consider the on-line problem of localizing a single new user or item based on paired comparisons. (We will return later to the problem of how this initial embedding might be generated.) In this context, there are two main problems of interest: localizing *users* and *items*. In the case of localizing a user, [4] proposes a relatively simple approach (which we review in greater detail in Section 2) which builds on the observation that each comparison provided by a user with ideal point $\boldsymbol{x}$ divides the space $\mathbb{R}^n$ in half and tells us which side of a hyperplane the point $\boldsymbol{x}$ lies in. As we will see in Section 3, however, localizing a new item from the same kind of measurements is a bit more complicated – in particular, the comparisons now give rise to a set of non-convex constraints. However, we will demonstrate how to relax this problem to a

M. O'Shaughnessy was supported by the Georgia Tech President's Undergraduate Research Award program. M. Davenport was supported by grants NRL N00173-14-2-C001, AFOSR FA9550-14-1-0342, NSF CCF-1409406, CCF-1350616, and CMMI-1537261.

978-1-5090-0746-2/16/$31.00

convex one that is extremely similar to the approach we take for localizing users, and which we demonstrate in Section 4 to be highly effective in practice.

In Section 5 we return to the problem of generating an embedding of users/items in the first place. One approach would be to generate an item embedding independently using a variety of methods, such as multidimensional scaling applied to a set of item features, and then localize each user using the approach described above. A more robust approach that uses only paired comparisons is described in [5], which proposes a general purpose algorithm for nonmetric multidimensional scaling based on semidefinite programming. Unfortunately, this algorithm scales poorly with the number of total users/items. In this paper we consider an alternative and highly scalable scheme based on the iterative application of the algorithms described in Sections 2 and 3, which we demonstrate to be surprisingly effective.

2. LOCALIZING A USER

We first review the method of localizing a new user (given an existing embedding of items) presented in [4]. Each binary paired comparison of the form "user $\boldsymbol{x}$ prefers item $\boldsymbol{q}_i^{(k)}$ to item $\boldsymbol{q}_j^{(k)}$," which we denote $\|\boldsymbol{x}-\boldsymbol{q}_i^{(k)}\|_2^2 \leq \|\boldsymbol{x}-\boldsymbol{q}_j^{(k)}\|_2^2$, is represented in the optimization problem as a constraint. Here, $\boldsymbol{q}_i$ refers to the item that is preferred (over $\boldsymbol{q}_j$), and the superscript (k) indicates that the two items are represented in the k^{th} comparison. In the objective function of our optimization problem, we minimize the total magnitude of comparison violations (expressed in the vector of slack variables $\boldsymbol{\xi}$) plus a regularization term:

$$\begin{aligned} \underset{\boldsymbol{x}^*,\,\boldsymbol{\xi}}{\text{minimize}} \;\; & \tfrac{1}{2}\|\boldsymbol{x}^*-\boldsymbol{x}_0\|_2^2 + \sum_k c_k\xi_k \\ \text{subject to} \;\; & \|\boldsymbol{x}^*-\boldsymbol{q}_i^{(k)}\|_2^2 \leq \|\boldsymbol{x}^*-\boldsymbol{q}_j^{(k)}\|_2^2 + \xi_k \\ & \xi_k \geq 0 \end{aligned} \tag{1}$$

The values $\{c_k\}$ represent how heavily violations of each comparison are penalized relative to each other and the regularization term; adjusting $\{c_k\}$ is a natural way to represent the relative confidence we have in each the accuracy of each comparison. In the objective function, $\boldsymbol{x}_0$ represents an initial estimate of the ideal point for the user to be localized (in the case where we do not have a previous estimate of $\boldsymbol{x}$, we initialize $\boldsymbol{x}_0$ to the origin or the center of the embedding of items). The regularization term $\|\boldsymbol{x}^*-\boldsymbol{x}_0\|_2^2$ serves three purposes: (*i*) it constrains $\boldsymbol{x}^*$ when the set of comparisons does not result in a completely bounded feasible region; (*ii*) it regularizes the solution to avoid sensitivity to outliers; and (*iii*) it allows us to bias the solution towards a pre-existing estimate, which can be very helpful when such an estimate exists and our comparisons are noisy or small in number.

While perhaps not initially obvious, the optimization problem in (1) is a standard quadratic program with linear constraints. To see this, one needs to simplify the constraint, eliminating the $\|\boldsymbol{x}^*\|_2^2$ term from both sides and rearranging to obtain the linear constraint:

$$2\left(\boldsymbol{q}_j^{(k)}-\boldsymbol{q}_i^{(k)}\right)^T\boldsymbol{x}^* + \|\boldsymbol{q}_i^{(k)}\|_2^2 - \|\boldsymbol{q}_j^{(k)}\|_2^2 - \xi_k \leq 0 \tag{2}$$

From this form, we can see that each linear constraint resulting from a binary paired comparison defines a half-space in $\mathbb{R}^n$ where the localized user $\boldsymbol{x}$ lives. Figure 1 shows a geometric interpretation of the feasible region created by several comparisons. See [4] for further details. From here we can simplify the optimization problem by assigning the vector $2(\boldsymbol{q}_j^{(k)}-\boldsymbol{q}_i^{(k)})^T$ from each comparison to the rows of matrix $\boldsymbol{A}$ and the scalar entries $\|\boldsymbol{q}_i^{(k)}\|_2^2 - \|\boldsymbol{q}_j^{(k)}\|_2^2$ as entries of the column vector $\boldsymbol{b}$. With this notation, the optimization problem takes its final form as:

$$\begin{aligned} \underset{\boldsymbol{x}^*,\,\boldsymbol{\xi}}{\text{minimize}} \;\; & \tfrac{1}{2}\|\boldsymbol{x}^*-\boldsymbol{x}_0\|_2^2 + \boldsymbol{c}^T\boldsymbol{\xi} \\ \text{subject to} \;\; & \boldsymbol{A}\boldsymbol{x}^* + \boldsymbol{b} - \boldsymbol{\xi} \leq 0 \\ & -\boldsymbol{\xi} \leq 0 \end{aligned} \tag{3}$$

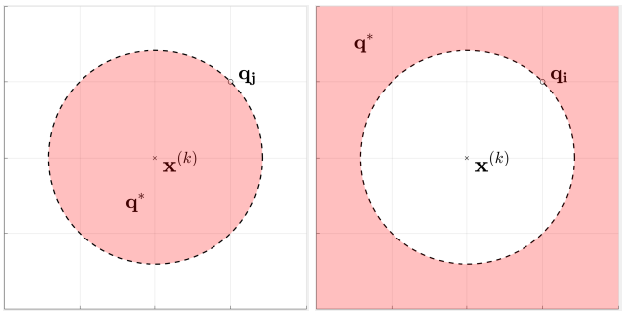
where the inequality sign in both constraints is applied element-wise. This is a standard quadratic program, and is solvable using a convex optimization software package such as CVX [6, 7].

3. LOCALIZING AN ITEM

To localize a new item given an existing embedding of users/items, we begin by constructing an optimization problem similar to (1). However, when localizing a new item, it is useful to divide the constraints into two groups: (*i*) those deriving from comparisons in which the item to be localized, $\boldsymbol{q}^*$, is the item preferred to some alternative $\boldsymbol{q}_j^{(k)}$, and (*ii*) those for which $\boldsymbol{q}^*$ represents the item preferred less than some $\boldsymbol{q}_i^{(k)}$. We will let $\mathcal{T}_+$ and $\mathcal{T}_-$ index these two sets of constraints in our optimization problem as follows:

$$\begin{aligned} \underset{\boldsymbol{q}^*,\,\boldsymbol{\xi}}{\text{minimize}} \;\; & \tfrac{1}{2}\|\boldsymbol{q}^*-\boldsymbol{q}_0\|_2^2 + \sum_k c_k\xi_k \\ \text{subject to} \;\; & \|\boldsymbol{x}^{(k)}-\boldsymbol{q}^*\|_2^2 \leq \|\boldsymbol{x}^{(k)}-\boldsymbol{q}_j^{(k)}\|_2^2 + \xi_k \;\; k\in\mathcal{T}_+ \\ & \|\boldsymbol{x}^{(k)}-\boldsymbol{q}_i^{(k)}\|_2^2 \leq \|\boldsymbol{x}^{(k)}-\boldsymbol{q}^*\|_2^2 + \xi_k \;\; k\in\mathcal{T}_- \\ & \xi_k \geq 0 \end{aligned} \tag{4}$$

As illustrated geometrically in Figure 2, this set of constraints makes the problem nonconvex. On the one hand, each constraint in $\mathcal{T}_+$ (where $\boldsymbol{q}^*$ is the preferred item) defines a hypersphere within which the localized item must lie. Constraints of this type are convex (but quadratic, in contrast to the linear constraints that arise when localizing a user). On the other hand, each constraint in $\mathcal{T}_-$ (where $\boldsymbol{q}^*$ is the less preferred item) defines a hypersphere within which the localized item *cannot* lie. The inclusion of the constraints in $\mathcal{T}_+$ make the optimization problem in (4) nonconvex.



(a) Constraint resulting in convex feasible region (shaded)

(b) Constraint resulting in non-convex feasible region (shaded)

Fig. 2: Feasible region in $\mathbb{R}^2$ for (a) comparisons where the item to be localized is the preferred item, and (b) comparisons where the item to be localized is the less preferred item.

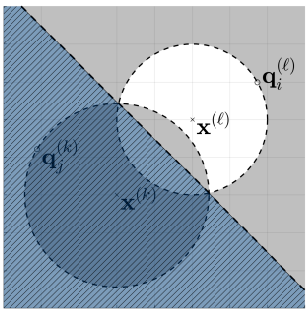


Fig. 3: Linear constraint generated by the combination of two quadratic constraints. Gray shading represents the feasible regions resulting from each of the two paired comparisons; blue hatched shading represents the feasible region defined by the hyperplane resulting from the combination of two comparisons. Note that the linear relaxation results in a much larger feasible region.

To make the optimization problem convex, we consider one possible convex relaxation; we combine pairs of comparisons by simply adding them, creating a single linear constraint from two quadratic constraints. While other convex relaxations are possible, we will see that this simple method results in linear constraints, leads to a computationally efficient algorithm, and produces good results in practice.

To describe our procedure, we will suppose that $\boldsymbol{q}^*$ denotes the item that we wish to update. We can combine constraints by adding them to produce useful linear constraints when the pair is of the form

$$\begin{aligned} \|\boldsymbol{x}^{(k)} - \boldsymbol{q}^*\|_2^2 &\leq \|\boldsymbol{x}^{(k)} - \boldsymbol{q}_j^{(k)}\|_2^2 \\ \|\boldsymbol{x}^{(\ell)} - \boldsymbol{q}_i^{(\ell)}\|_2^2 &\leq \|\boldsymbol{x}^{(\ell)} - \boldsymbol{q}^*\|_2^2 \end{aligned} \tag{5}$$

and $\boldsymbol{x}^{(k)} \neq \boldsymbol{x}^{(\ell)}$. The requirement that $\boldsymbol{q}^*$ appear on opposite sides of the two inequalities ensures the quadratic terms involving $\boldsymbol{q}^*$ cancel when adding, and $\boldsymbol{x}^{(k)} \neq \boldsymbol{x}^{(\ell)}$ is necessary to avoid making the linear term in the resulting constraint zero and the constraint vacuous.

For each inequality, the sides not containing the item that we wish to localize ($\boldsymbol{q}^*$) are constants, which we denote c and d. We then combine the inequalities as

$$\begin{aligned} \|\boldsymbol{x}^{(k)} - \boldsymbol{q}^*\|_2^2 &\leq \overbrace{\|\boldsymbol{x}^{(k)} - \boldsymbol{q}_j^{(k)}\|_2^2}^{c} \\ + \underbrace{\|\boldsymbol{x}^{(\ell)} - \boldsymbol{q}_i^{(\ell)}\|_2^2}_{d} &\leq \|\boldsymbol{x}^{(\ell)} - \boldsymbol{q}^*\|_2^2. \end{aligned} \tag{6}$$

Expanding the norm in each inequality, we note that combining the comparisons results in the cancellation of the quadratic term $\|\boldsymbol{q}^*\|_2^2$, simplifying the pair of more computationally expensive quadratic constraints into a single linear one. Adding and simplifying yields

$$2\left(\boldsymbol{x}^{(\ell)} - \boldsymbol{x}^{(k)}\right)^T \boldsymbol{q}^* \leq c - d + \|\boldsymbol{x}^{(\ell)}\|_2^2 - \|\boldsymbol{x}^{(k)}\|_2^2. \tag{7}$$

We can substitute these constraints into (4) to obtain a linearly-constrained quadratic problem that has the same form as the user localization optimization problem in (3), with each $2(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(\ell)})^T$ assigned to a row of $\boldsymbol{A}$ and each $c - d + \|\boldsymbol{x}^{(\ell)}\|_2^2 - \|\boldsymbol{x}^{(k)}\|_2^2$ assigned to an entry of $\boldsymbol{b}$.

Figure 3 shows a geometric interpretation of the linear constraint induced by a combination of the two quadratic constraints. It is important to note that while the linear relaxation obtained by this procedure results in a much larger feasible region (in particular, one that is unbounded), the combination of many such pairs of constraints may still be an effective simplification of the original set of nonconvex constraints.

Note that if we have a large number of comparisons, the number of valid combinations – $|\mathcal{T}_+||\mathcal{T}_-|$ – could be extremely large. In this case, we simply choose a subset of m combinations at random, although more intelligent choices are likely possible (particularly in real-world applications, where the distribution of users and items present in comparisons will be highly uneven). It is also important to note that only pairs of comparisons that produce a hyperplane like the one shown in Figure 3, where the hyperplane splits $\boldsymbol{x}^{(k)}$ and $\boldsymbol{x}^{(\ell)}$, are combined in this way. To only select combinations with this behavior, we also require that either $\boldsymbol{a}^T\boldsymbol{x}^{(k)} < b < \boldsymbol{a}^T\boldsymbol{x}^{(\ell)}$ or $\boldsymbol{a}^T\boldsymbol{x}^{(\ell)} < b < \boldsymbol{a}^T\boldsymbol{x}^{(k)}$ for each combination.

4. SIMULATIONS

In this section, we demonstrate the performance of these approaches to localizing a new user or item. For now we assume *a priori* knowledge of an existing embedding of users/items, and use our list of paired comparisons to localize a new user or item in the embedding.

In each simulation, a ground truth embedding of many users and items is generated uniformly on the unit square in $\mathbb{R}^n$. Comparisons are generated by randomly selecting a user $\boldsymbol{x}^{(k)}$ and two items $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$, $\boldsymbol{q}_1 \neq \boldsymbol{q}_2$. For each comparison, $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ are assigned to $\boldsymbol{q}_i^{(k)}$ and $\boldsymbol{q}_j^{(k)}$ to make them

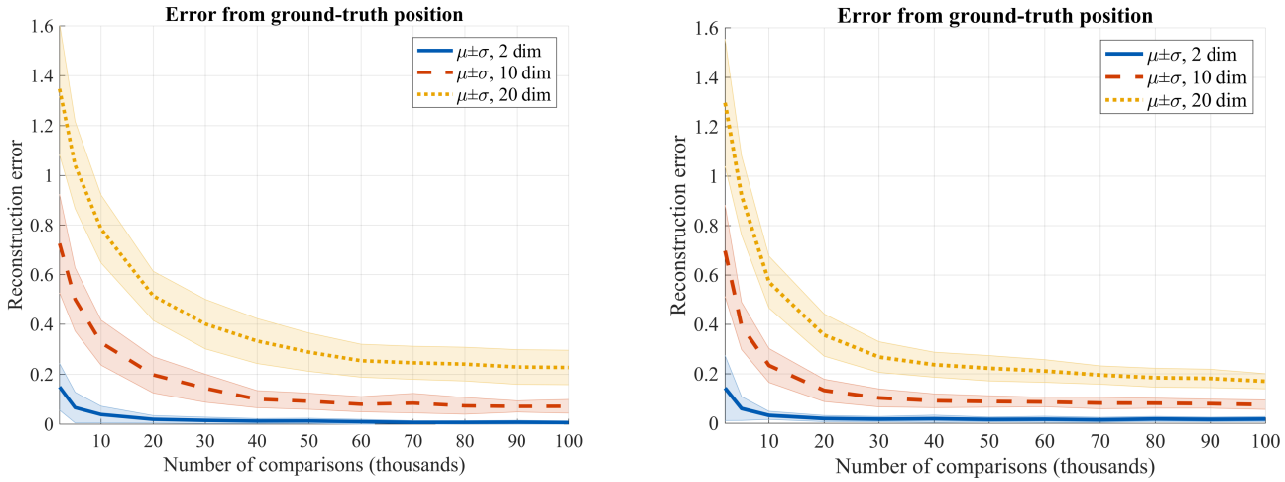


Fig. 4: Localization error measured in terms of ℓ_2 distance from ground truth as a function of the *total* number of comparisons available to the system, which contains 100 users and 100 items. (a) User localization error. (b) Item localization error.

consistent with $\|\boldsymbol{x}^{(k)} - \boldsymbol{q}_i^{(k)}\|_2 \leq \|\boldsymbol{x}^{(k)} - \boldsymbol{q}_j^{(k)}\|_2$.

In each of the following simulations, our "ground truth" embedding consists of 100 users and 100 items. Comparisons are generated uniformly at random from the complete set of users and items, and performance is measured by computing the ℓ_2 distance between the estimated point and the ground truth. We repeat each simulation 30 times and report the mean and standard deviation.

In the first simulation, we demonstrate the accuracy with which we can localize a new user based on the number of comparisons we have available. As shown in Figure 4(a), a very small number of comparisons involving the localized user results in a large error, but this error drops off rapidly with an increasing number of comparisons. Note that in this paper we consider comparisons chosen randomly; similar performance could be obtained with a much smaller number of comparisons if the user and items to compare were adaptively selected (see, for example, the approaches in [8, 9, 10]). Figure 4(a) also shows the large impact of dimensionality on the ℓ_2 recovery error. This is expected; on the unit square in $\mathbb{R}^n$, the maximum possible error grows as $\sqrt{n}$. These results are consistent with recent theoretical results for this problem established in [10, 11].

Next, we apply the item localization procedure described in Section 3 to localize a new item given an existing embedding of users/items. As noted in the previous section, we select a subset of comparison combinations at random from the (possibly large) set of valid combinations; here, we pick $m = 20000$ combinations. In Figure 4(b), we show that our convex relaxation of adding comparisons allows us to localize a new item with accuracy comparable to that of the user localization problem. This may be somewhat surprising since our procedure for generating a convex set of constraints seems to discard a significant amount of information.

5. GENERATING A COMPLETE EMBEDDING

Using the procedures described in Sections 2 and 3, we can localize a new user or item into an embedding of users and items given only a list of paired comparisons. However, our localization procedures can also be used to improve an existing embedding using a list of paired comparisons. When presented with an existing embedding, we can iterate through each point and use our localization procedure to improve its accuracy. Because the accuracy with which we can localize a new point depends on the accuracy of every other point in the embedding, each update can improve the current point as well as the results of future iterations.

The iterative localization procedure is as follows: *(i)* Create an initial embedding (if no initial embedding is available, initialize each point at random); *(ii)* Iterate through each user and item in the embedding, updating the user/item's location at each step by solving the optimization problem from Sections 2 or 3 as appropriate. Repeat this iteration until convergence.

Using this procedure, we can use a list of paired comparisons to improve the accuracy of a noisy embedding of users and items. Further, even if we do not have an initial estimate of the embedding, we can use this procedure to generate one from scratch with surprising accuracy.

Improving an existing embedding. To evaluate the potential of this approach, we begin by generating comparisons between users and items as in the experiments in Section 4, but after generating the comparisons we perturb every user and item in the embedding with Gaussian noise. Figure 5 shows the reduction of error our procedure can achieve for three different perturbation noise levels (σ_{emb}). We show performance both in terms of the ℓ_2 distance to the ground truth (across all points in the embedding, after applying an affine transformation, also known as the Procrustes distance) but also in

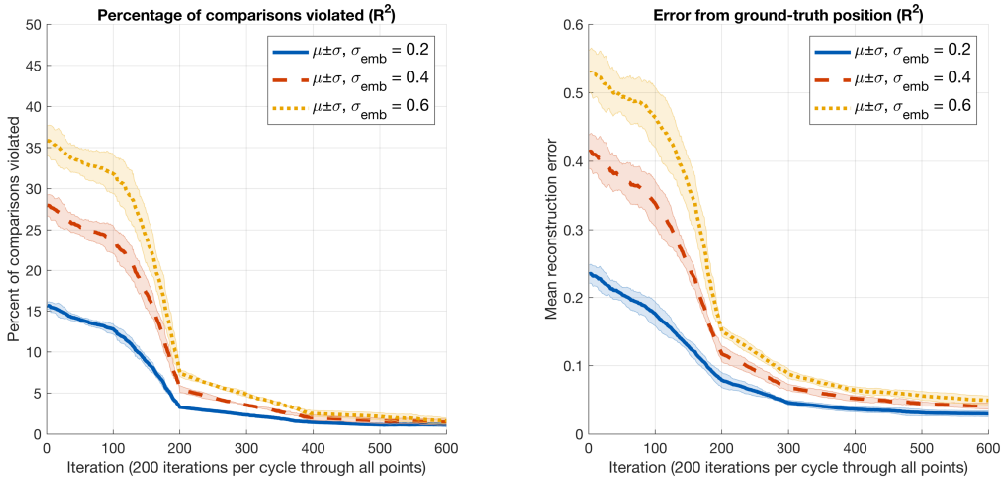


Fig. 5: Reduction in noise achieved for a noisy embedding in $\mathbb{R}^2$, measured in terms of percent of all comparisons violated and mean ℓ_2 recovery error calculated using the Procrustes distance.

terms of the percentage of observed comparisons which are violated by the learned embedding. Given a suitable number of iterations, we observe that we can reduce the error in the embedding to approximately the same level, regardless of the perturbation noise level.

Generating an embedding from scratch. Our iterative procedure is surprisingly effective at "denoising" an embedding, even when it is corrupted with enough noise to violate up to 35% of the given comparisons. This suggest that it may be possible to obtain similar results when we have no *a priori* knowledge of the embedding at all, generating an embedding from scratch.

In this simulation, we again create a ground truth embedding of users and items distributed uniformly on the unit square in $\mathbb{R}^n$. We use this ground truth embedding to generate a list of binary paired comparisons, then initialize the reconstructed embedding to a random set of points and use only the list of comparisons to recreate the embedding. As in the previous experiments, we measure the accuracy of our reconstruction using the Procrustes distance between the reconstruction and original embedding.

Figure 6 shows that even with a random initial embedding (approximately 50% of comparisons violated), we can generate an embedding with only a modest level of error in only a few passes over the entire dataset. While the mean ℓ_2 recovery error in Figure 6 shows error increasing with dimensionality, the error does not grow as quickly as the scaling of distances with dimension (which increases as $\sqrt{n}$).

6. DISCUSSION

In this work, we have extended the optimization-based method in [4] for localizing a new user into an embedding of users/items using a set of binary paired comparisons. This paper has two main contributions: first, we derive a convex relaxation which allows us to use a similar procedure for localizing items into an embedding. Next, we develop a method for iteratively improving a noisy embedding or generating an embedding from scratch using only a set of paired comparisons. Simulations on synthetically generated data show that given a suitable number of comparisons, we can localize new users and items extremely accurately and can generate an embedding with good accuracy, even in high dimensions. The algorithm is computationally efficient and easily parallelizable: localizing a single point only requires solving a simple linearly-constrained quadratic program, and generating an approximation of the entire embedding only requires a few iterations through each user/item in the embedding (which can be performed in parallel).

This efficiency is key, because in real-world recommendation systems the number of users and items may be extremely large. This algorithm scales far better in terms of both computation and memory requirements than existing procedures based on semidefinite programming such as generalized non-metric multidimensional scaling [5]. However, the performance of our algorithm might be significantly improved by an accurate knowledge (or estimate) of several "landmark points," which could be generated by such a procedure.

Finally, we briefly note that the ideal point model described in this paper, while closely related, is distinct from the low-rank model used in *matrix completion* approaches which have recently gained much attention, e.g., [12, 13]. Although both models suppose preferences are guided by a small number of factors, the ideal point model leads to preferences that are *non-monotonic* functions of those attributes. There is empirical evidence that the ideal point model captures user behavior more accurately than factorization based approaches do [14]. Nevertheless, the results in [15, 16, 17, 18], which consider binary observations, paired comparisons, and other more general ordinal measurements in the low-rank context, share a similar inspiration as this work.

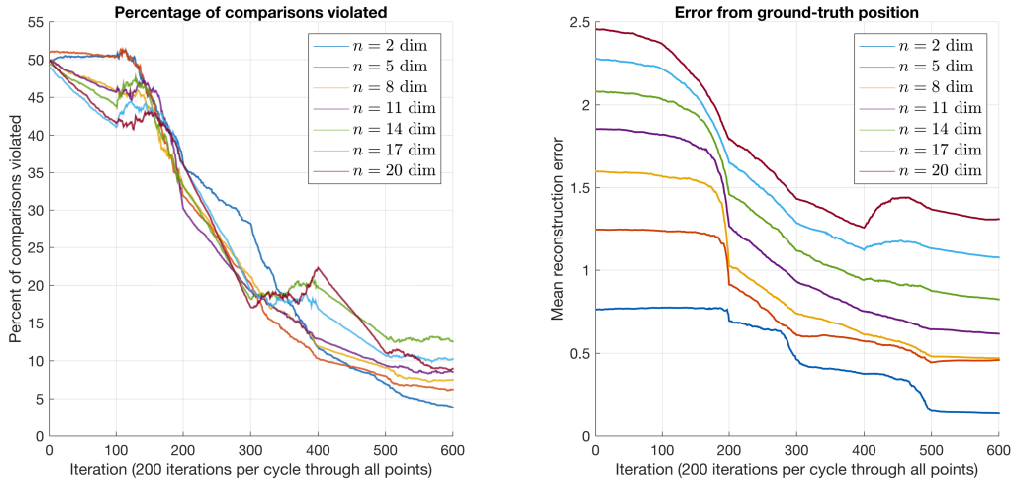


Fig. 6: Recovery error for a generated embedding measured in terms of percent of all comparisons violated and mean ℓ_2 recovery error calculated using the Procrustes distance.

7. REFERENCES

[1] C. Coombs, "Psychological scaling without a unit of measurement," *Psych. Rev.*, vol. 57, no. 3, pp. 145–158, 1950.

[2] H. David, *The method of paired comparisons*, London, UK: Charles Griffin & Company Limited, 1963.

[3] F. Radlinski and T. Joachims, "Active exploration for learning rankings from clickthrough data," in *Proc. ACM SIGKDD Int. Conf. on Knowledge, Discovery, and Data Mining (KDD)*, San Jose, CA, August 2007, ACM.

[4] M. Davenport, "Lost without a compass: Nonmetric triangulation and landmark multidimensional scaling," in *Proc. IEEE Int. Work. Comput. Adv. Multi-Sensor Adaptive Processing (CAMSAP)*, Saint Martin, Dec. 2013.

[5] S. Agarwal, J. Wills, L. Cayton, G. Lanckriet, D. Kriegman, and S. Belongie, "Generalized non-metric multidimensional scaling," in *Proc. Int. Conf. Art. Intell. Stat. (AISTATS)*, San Juan, Puerto Rico, Mar. 2007.

[6] Michael Grant and Stephen Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," `http://cvxr.com/cvx`, Mar. 2014.

[7] Michael Grant and Stephen Boyd, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, V. Blondel, S. Boyd, and H. Kimura, Eds., Lecture Notes in Control and Information Sciences, pp. 95–110. Springer-Verlag Limited, 2008, `http://stanford.edu/~boyd/graph_dcp.html`.

[8] K. Jamieson and R. Nowak, "Active ranking using pairwise comparisons," in *Proc. Adv. in Neural Processing Systems (NIPS)*, Granada, Spain, Dec. 2011.

[9] K. Jamieson and R. Nowak, "Low-dimensional embedding using adaptively selected ordinal data," in *Proc. Allerton Conf. Communication, Control, and Computing*, Monticello, IL, 2011.

[10] A. Massimino and M. Davenport, "As you like it: Localization via paired comparisons," *Preprint*, 2016.

[11] A. Massimino and M. Davenport, "Binary stable embedding via paired comparisons," in *Proc. IEEE Work. Stat. Signal Processing*, Palma de Mallorca, Spain, June 2016.

[12] J. Rennie and N. Srebro, "Fast maximum margin matrix factorization for collaborative prediction," in *Proc. Int. Conf. Machine Learning*, Bonn, Germany, Aug. 2005.

[13] E. Candès and B. Recht, "Exact matrix completion via convex optimization," *Found. Comput. Math.*, vol. 9, no. 6, pp. 717–772, 2009.

[14] A. Maydeu-Olivares and U. Böckenholt, "Modeling preference data," *The SAGE handbook of quantitative methods in psychology*, pp. 264–282, 2009.

[15] M. Davenport, Y. Plan, E. van den Berg, and M. Wootters, "1-bit matrix completion," *Inf. Inference*, vol. 3, no. 3, pp. 189–223, 2014.

[16] Y. Lu and S. Negahban, "Individualized rank aggregation using nuclear norm regularization," *arxiv:1410.0860*, 2014.

[17] D. Park, J. Neeman, J. Zhang, S. Sanghavi, and I. Dhillon, "Preference completion: Large-scale collaborative ranking form pairwise comparisons," in *Proc. Int. Conf. Machine Learning*, Lille, France, July 2015.

[18] S. Oh, K. Thekumparampil, and J. Xu, "Collaboratively learning preferences from ordinal data," in *Proc. Adv. in Neural Processing Systems (NIPS)*, Montréal, Québec, Dec. 2014.