Ordinal Embedding with a Latent Factor Model

Mingyuan Zhang, Nora Farouk, and Laura Balzano*
Electrical Engineering and Computer Science, University of Michigan
Email: {mingyuaz, nrfarouk, girasole}@umich.edu

September 13, 2016

Abstract

Constructing low-dimensional embeddings based on ordinal measurements has been a subject of significant recent interest, motivated in part by machine learning applications using human input in a robust way. Recent work has focused on observations of comparisons on distances between objects. We consider a different model where the embedding is formed within a latent space of factors upon which a user may make judgements in the form of a rank order. The user gives an answer based on weighting latent factors as opposed to Euclidean distance in the embedding. Our contribution is an algorithm that learns the embedding reliably and efficiently and can use as much information as the user is willing to provide in the form of a rank-ordered list.

1 Introduction

The problem of ranking a set of n objects given only partial ordering information is relevant in many applications, from recommender systems to web search to resource prioritization. There are also applications wherever people and their work must be judged: hiring at a corporation, graduate school admissions, conference publication acceptance, or science fair judging, to name a few. Recent research in rank learning has shed light on when it is possible to learn the ranking of objects from various kinds of partial information, and several algorithms have been developed for this purpose.

The perspective we will take in this paper is where partial ordering observations arise from a low-dimensional embedding of the objects. Whereas much recent work considers low-dimensional embeddings that respect distances or distance comparisons, we consider a different model where the embedding is part of a latent space of factors upon which a user may make judgements in order to then come to some conclusion in the form of a rank order. So, questions common to recent work like "Do you prefer A or B" are still appropriate, but

*This work was supported by the U.S. Army Research Office under grant number W911NF1410634.

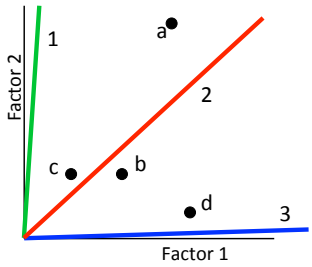


Figure 1: Example embedding where a,b,c,d are objects in 2d and judges 1,2,3 weight factors differently when ranking the objects.

the user is giving an answer based on weighting latent factors as opposed to Euclidean distance in the embedding. This model is appropriate for problems where one might assume the users have latent factors under consideration, and the relative goodness of each object compared along a certain dimension determines its ranking when considering only that factor. Our goal will be to learn an embedding for the objects and weights describing each user's preferences that match the partial rankings observed. Our contribution is both to formulate this model mathematically and develop an algorithm that learns the embedding reliably and efficiently and can use as much information as the user is willing to provide in the form of a rank-ordered list, as opposed to only pairwise or triple-wise comparisons.

2 Problem Formulation

We consider the problem as follows. Suppose we have n objects and m judges who we will ask to rank the objects. We assume that judge j's ranking is obtained as

$$r_j = \text{rank\_order}(Uw_j) \qquad j = 1, \ldots, m$$

for fixed $U \in \mathbb{R}^{n \times d}$ that is common to all judges and $w_j \in \mathbb{R}^d$ that are weights particular to each judge. The operator rank_order(x) takes the column x and outputs the index of a descending sort function; *i.e.,* the output is a vector of $1, \ldots, n$ where the position of the highest number in x will get a 1 and so on. This model implies that all the judges use some shared criteria for the objects but weight those criteria differently when ranking.

Consider the example given in Figure 1. Here we have four objects represented by black circles embedded in $\mathbb{R}^2$. The experts or judges are represented by lines in the plane, and the ranking is the projection of each point onto that line; note that for this example all the weights are positive. Judge 1, the green line, weights factor 2 very heavily and factor 1 almost not at all. So the resulting rank for Judge 1 would be $\{a, b, c, d\}$ (with a tight call between b, c). Judge 3, the blue judge, on the other hand feels the reverse about factor 1 and 2 and has

ranking $\{d, a, b, c\}$. Finally Judge 2 (in red) finds both factors to be important and would rank $\{a, d, b, c\}$.

This model is in some sense a special case of the landmark model [1, 2, 3], where each judge is a landmark placed at an unknown location infinitely far from the origin, implying that an object that is further away from the origin on a given axis is favored (in the case of positive weight on that axis; disfavored in the case of negative weight). Natural applications for this model are those where judges typically differ not in how much of a quality is good (or bad) but instead how to weight the qualities for an overall ranking. Applications that are more natural for the landmark model using distance-based embeddings are those where a user may actually prefer to have "just enough" of a quality; *e.g.,* in the beer mapper application [4], you may be interested in slightly hoppy beer or beer with a moderate yeast flavor as opposed to all hops or no yeast.

3 Related Work

A popular model used for ordinal embedding problems seeks an embedding from given distances or distance comparisons. Our model, while related, has some fundamental differences resulting in very different properties. However, much of our approach to ordinal embedding is inspired by related work that we now describe.

In an early milestone collection of papers including [5, 6] Shepard and then Kruskal defined non-metric multi-dimensional scaling to be the problem of finding a configuration of points in dimension d such that observed dissimilarities between those points are respected. To quote from [6], "it is supposed that the 'true' dissimilarities result from some unknown monotone distortion of the interpoint distances of some 'true' configuration, and that the observed dissimilarities differ from the true dissimilarities only because of random fluctuation." Since the true dissimilarities are not formally distances but some distortion thereof, the problem is called "non-metric." In Kruskal's seminal work [6] he proposed starting from a random configuration and using gradient methods to improve the stress function, a measure of the difference between a test configuration with distances d_{ij} and given dissimilarities δ_{ij}, which is:

$$S = \sqrt{\frac{\sum \left(d_{ij} - \hat{d}_{ij}\right)^2}{\sum d_{ij}^2}}$$

where

$$\hat{d}_{ij} = \arg\min S \quad \text{subject to} \quad \hat{d}_{ij} \leq \hat{d}_{k\ell} \Leftrightarrow \delta_{ij} \leq \delta_{k\ell} \ .$$

The work in [7] generalizes the approach of Kruskal and Shepherd to the case where dissimilarities are not used but only the ordering of them is exploited; they call this generalized non-metric multi-dimensional scaling (GNMDS) and we will also use this naming. This leads them to the following optimization problem. Let $G = X^T X$ be the Gram matrix for a set of points in a matrix X.

Suppose we are given a set of constraints for some pairs of points such that we know only $\delta_{ij} \leq \delta_{k\ell}$; call this set Ω. The paper proposes solving the following semidefinite program:

$$\begin{aligned} \text{minimize}_{G,\xi_{ijk\ell}} \quad & \sum_{(i,j,k,\ell)\in\Omega} \xi_{ijk\ell} + \lambda \operatorname{trace}(G) && (1) \\ \text{subject to} \quad & G_{kk} - 2G_{k\ell} + G_{\ell\ell} - G_{ii} + 2G_{ij} - G_{jj} \\ & \geq 1 - \xi_{ijk\ell} \quad \forall (i,j,k,\ell) \in \Omega && (2) \\ & \sum_{ab} G_{ab} = 0; \quad G \succeq 0 \ . \end{aligned}$$

The objective (1) minimizes the sum of slack variables and the weighted trace of the matrix variable to regularize for rank of the embedding. The constraints in (2) capture all of the comparison information that we have, requiring the points to satisfy those constraints with some buffer (a constant 1 in this case) and allowing slack using the variables ξ. Finally, $\sum_{ab} G_{ab} = 0$ centers the embedding to remove translational ambiguity and $G \succeq 0$ imposes the PSD constraint. Our work takes a very similar algorithmic approach by minimizing slack variables associated with constraints from the ranking information given. However, we assume the desired embedding dimension is given, which allows for a faster and more scalable algorithm without the nuclear norm regularization. Our constraints are also more structured; since each judge gives ranking information for a set of objects, we can reduce redundancy introduced by the slack variables of related constraints.

This work in turn inspired many other algorithmic developments. As two examples, we describe [1, 8]. The work of [1] proposes an algorithm for *landmark* generalized non-metric MDS where comparisons are made only to points in a landmark set with known locations. Then a convex optimization problem is executed to find the location of each new point. This problem looks very similar to that in (1) except (a) with only constraints on a point's relationship to the landmark points and (b) without the rank regularization, since we know the dimension from the landmark points. The work of [8] seeks an ordinal embedding of relationships in a kNN graph; in other words, for each point x we are given information as to which k other points are closest, and from this information we want to reconstruct the embedding. Their approach is to split the graph into overlapping patches, find an embedding on each patch using Local Ordinal Embedding [9], and then stitch those patches together by finding an orthogonal transform and shift for each individual patch that matches nicely on the overlap.

While all the work described thus far focuses on algorithms, as does our work here, a great deal of recent effort has gone towards understanding theoretically when ordinal embeddings are unique and recoverable. This line of work includes [10, 9, 3], which ask when a N comparisons are enough to uniquely determine an embedding up to a similarity transform. [10] gives uniqueness results assuming the sample size becomes *dense* in the embedding space $\mathbb{R}^d$.

[9] follows up to solve some open questions of [10] and gives results based only on local information (like the kNN mentioned above). [3] further extends these results that were only for quadruple comparisons (of the form $d(ij) < d(k\ell)$) to the case of triple comparisons (of the form $d(ij) < d(jk)$). Both [10, 3] assume that *all comparisons* are available, and that as the number of objects n grows they become dense in $\mathbb{R}^d$. [3] also gives rates that say as the number of objects n grows, an embedding can be found that is within some ϵ_n Hausdorff distance from the true embedding. Again, all comparisons are available. And in these works, they assume the comparisons are consistent with some true low-dimensional underlying embedding.

A more recent paper by Jain, Jamieson, and Nowak [11] addressed a technique similar to [7] and provides even more general guarantees. This paper gives prediction error bounds for an ordinal embedding learned from only a subset of all the distance comparisons, and the comparisons may be noisy where the noise is modeled as a probability p with which the user swaps the comparative distance inequality. Again where G is the gram matrix, given regularization parameters λ, γ they solve

$$\begin{aligned} \text{minimize}_G \quad & \widehat{R}(G) \\ \text{subject to} \quad & G \in \mathbb{S}_+^n, \|G\|_* \leq \lambda, \|G\|_\infty \leq \gamma \,, \end{aligned} \tag{3}$$

where $\widehat{R}(G)$ is the empirical risk over the triple comparisons observed, $\|\cdot\|_*$ is the nuclear norm and $\|\cdot\|_\infty$ is the max absolute value of all entries of the matrix. They analyze this program under arbitrary probabilities of error, but in the case that the probability of error decreases as the distances are further apart according to a known link function (such as logistic), they also guarantee recovery of the original distance matrix (up to invariances) using the maximum likelihood embedding. They propose two algorithms, one alternating a step of a gradient method with a low-rank projection step, and another building off the existing nuclear norm methods by adding a debiasing step to improve the estimation accuracy.

Both the optimization algorithm of [7] in Eq (1) and of [11] in Eq (3) inspired the Aura Algorithm we present in the next section. However, our model is quite different, and so the results we have just described using distance comparisons are not directly applicable. Consider again the example in Figure 1. From the ranking information we observe, we cannot recover distance comparisons. Judge 2, for example, ranks $\{a, d, b, c\}$, but this does not imply that $\text{dist}(a, d) < \text{dist}(a, b)$. The same is true for Judge 3; her rank is $\{d, a, b, c\}$ even though $\text{dist}(d, a) > \text{dist}(d, b)$ and even $\text{dist}(d, a) > \text{dist}(d, c)$.

Our model additionally allows for a PCA-like analysis, where the factors and weights identified in the embedding may have a semantic meaning that will allow a deeper understanding of the judges and objects involved. This would in turn allow one to improve judge diversity and facilitate active selection of which new objects a judge should incorporate into his ranking. The second strength is that this model, when restricted to have positive weights, can be naturally applied to the *rank aggregation* problem, a well-known difficult problem in social

choice theory.

4 The Aura Algorithm

We suppose that several judges are queried, each on a subset of objects, and we observe a resulting ranking matrix:

$$R = \text{rank\_order}([UW]_\Omega)$$

where $W = \begin{bmatrix} w_1 & \dots & w_m \end{bmatrix}$ is the matrix stacking each judge's weights as columns, $\Omega \subset \{1, \dots, n\} \times \{1, \dots, m\}$ is the subset of object-judge pairs that were part of the query, and it is understood that rank_order is applied to each column individually and only the observed items therein. From here we wish to

Algorithm 1 Aura Rank Satisfaction

Input: R, d, tolerance ϵ, max iterations
Output: $\hat{U}$, $\hat{W}$
Initialize random $\hat{U} \in \mathbb{R}^{n \times d}$, $\hat{W} \in \mathbb{R}^{d \times m}$
$\hat{R} = \text{rank\_order}([\hat{U}\hat{W}]_\Omega)$;
Set $\tau_j(i) = k$ such that $r_j(k) = i$ for all $(i, j) \in \Omega$ where r_j is the j^{th} column of R.
for $\ell = 1$ **to** max iterations **do**
  if $\|R - \hat{R}\|_F \geq \epsilon$ **then**
    Update $\hat{U}$ by solving

$$\begin{aligned} \underset{\hat{U}, \xi \in \mathbb{R}^M}{\text{minimize}} \quad & \|\xi\|_1 && (4) \\ \text{subject to} \quad & U_{\tau_j(i)} w_j \geq U_{\tau_j(i+1)} w_j + 1 - \xi_{ij} && (5) \\ & \text{and } \xi_{ij} \geq 0 \quad \forall (i, j) \in \Omega \end{aligned}$$

    where w_j are columns of W and U_k are rows of U, and M is the number of constraints.
  end if
  if $\|R - \text{rank\_order}(\hat{U}\hat{W})\|_F \geq \epsilon$ **then**
    Update $\hat{W}$ by solving

$$\begin{aligned} \underset{\hat{W}, \xi \in \mathbb{R}^M}{\text{minimize}} \quad & \|\xi\|_1 && (6) \\ \text{subject to} \quad & U_{\tau_j(i)} w_j \geq U_{\tau_j(i+1)} w_j + 1 - \xi_{ij} && (7) \\ & \text{and } \xi_{ij} \geq 0 \quad \forall (i, j) \in \Omega \end{aligned}$$

  end if
end for
Return $\hat{U}$ and $\hat{W}$

estimate $\hat{U}$ and $\hat{W}$ so that

$$\text{rank\_order}([\hat{U}\hat{W}]_\Omega) =: \hat{R} \approx R \, .$$

The Aura Ranking algorithm begins with a fixed embedding dimension and randomly initializes the $n \times d$ matrix $\hat{U}$ and $d \times m$ matrix $\hat{W}$. We then use alternating minimization to solve for $\hat{U}, \hat{W}$, enforcing the sorted order of every column in $\hat{U}\hat{W}$ with constraints to approximate the information given in R. After convergence, the algorithm outputs $\hat{U}, \hat{W}$ such that $\text{rank\_order}([\hat{U}\hat{W}]_\Omega) = \hat{R} \approx R$. The algorithm is given in Algorithm 1.

We make a few remarks about our proposed algorithm. It is a natural alternating estimation algorithm for the low-rank embedding given by our model. Since each judge ranks several objects in order, we only need to constrain those objects that follow each other in the ranking, eliminating redundancy that may be in the slack variables of Eq (1). We require the embedding dimension as input. We also require as input a maximum number of iterations, but we find that an embedding consistent with the given ranking information is often found in just a few iterations (see *e.g.,* Fig 2(b)) and the decrease in objective (or lack thereof) is a reliable indicator of whether the algorithm will succeed in finding a consistent embedding. Finally, even with random initialization we get excellent results, as we now demonstrate empirically.

5 Numerical Results

We implemented Aura with cvx [12] in Matlab.

5.1 Full observed rankings.

We begin with the case where every judge ranks all the objects and we observe these full rankings. We vary the number of objects and number of judges for two different embedding dimensions, $d = 2, 10$. For a range of number of objects (n) and number of judges (m), we plot several metrics to show that the performance of Aura is very strong, despite it being a simple alternating algorithm solving a highly non-convex and underdetermined problem. First in Figure 2(a) we sweep the number of objects n from 5 to 85 and number of judges m from 5 to 100 to show that from a random initialization, this algorithm succeeds in finding an embedding to match the given ranking matrix at a rate of nearly 100%. When there are too few judges for the number of objects, we have worse performance. Figure 2(b) shows we require a small number of algorithm iterations for convergence. More iterations are needed for larger problem sizes, as expected. In Figure 2(c) we show a metric that demonstrates the accuracy of the underlying embedding itself: we compute all three angles of every triangle formed by the objects in both the true and estimated embedding; we take the difference between the corresponding angles and average over all angles in the embedding. Figure 2(d) we show the proportion of successful trials for higher embedding dimension $d = 10$.

5.2 Partially observed and Noisy rankings.

In Figure 3 we show results for partial rankings. When the sampling density is above or below a certain threshold, the algorithm is able to match the given ranking information. The angle to the true embedding, on the other hand, improves smoothly as we increase both the number of judges and proportion of objects ranked. In Figure 4 we show the noisy results; when we add noise to $X = UW$ and the noise is high enough, our algorithm does not match the given rankings because X is no longer low-dimensional. The rightmost plot however shows error in the embedding, computed simply by $\|R - \hat{R}\|_F^2/nm$. Below a certain noise level, the error is < 0.2, which could correspond to a scenario where roughly a fifth of the rank orders are off by only 1 position.

5.3 Embedding examples.

Finally, we show examples of embeddings in $\mathbb{R}^2$ recovered by Aura in Figure 5. To align them for visualization purposes, we matched the points with label 1 and 2 by translation and scale, and then with reflection we ensured label 3 is on the correct side of the 1-2 line. We show both successful embeddings for full and partial rankings as well as failed embeddings, where the random initialization did not admit a match within 20 iterations. We can see that the successful embeddings are very well aligned with the original, true embeddings. The failed embeddings both seem to have begun to collapse to a line; studying this phenomenon is a subject of future research.

6 Conclusion

We have proposed a new model and associated algorithm for ordinal embedding from ranking information. When full ranking information is given, the Aura algorithm succeeds in finding an embedding to match the given ranking matrix at a rate of nearly 100%, despite being an underdetermined and non-convex problem. Aura's performance degrades gracefully with partial rankings and noise. Our various experiments illustrate a variety of compelling directions for future work.

We require the embedding dimension be specified. One could start with a small dimension and increase it until the results are satisfactory; studying this approach is a subject of future work. Identifying an initialization better than random is another interesting direction for future work.

While the empirical results for Aura are compelling, we lack theoretical guarantees. The various experiments illustrate a variety of potential routes to understand Aura theoretically. In particular, success in matching partial rankings has a clear trend where for a moderately small number of ranked objects the algorithm stalls entirely, despite the fact that this would result in fewer constraints. Finally, when a random initialization does fail it seems to often find an embedding that has collapsed to a line; understanding this

phenomenon could help improve the iteration and the reliability of algorithm convergence.

References

[1] Mark A Davenport, "Lost without a compass: Nonmetric triangulation and landmark multidimensional scaling," in *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2013 IEEE 5th International Workshop on*. IEEE, 2013, pp. 13–16.

[2] Kevin G Jamieson and Robert D Nowak, "Low-dimensional embedding using adaptively selected ordinal data," in *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*. IEEE, 2011, pp. 1077–1084.

[3] Ery Arias-Castro, "Some theory for ordinal embedding," *arXiv preprint arXiv:1501.02861*, 2015.

[4] Kevin Jamieson, "Beer mapper," .

[5] Roger N Shepard, "The analysis of proximities: Multidimensional scaling with an unknown distance function. i.," *Psychometrika*, vol. 27, no. 2, pp. 125–140, 1962.

[6] Joseph B Kruskal, "Nonmetric multidimensional scaling: a numerical method," *Psychometrika*, vol. 29, no. 2, pp. 115–129, 1964.

[7] Sameer Agarwal, Josh Wills, Lawrence Cayton, Gert RG Lanckriet, David J Kriegman, and Serge Belongie, "Generalized non-metric multidimensional scaling.," in *AISTATS*, 2007, pp. 11–18.

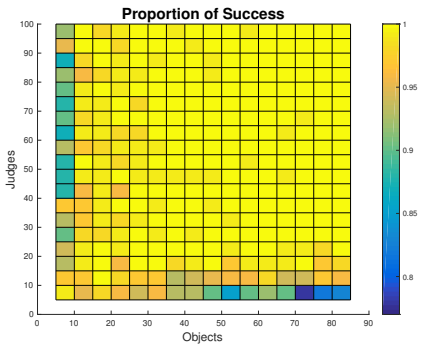
[8] Mihai Cucuringu and Joseph Woodworth, "Ordinal embedding of unweighted knn graphs via synchronization," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2015, pp. 1–6.

[9] Yoshikazu Terada and Ulrike V Luxburg, "Local ordinal embedding," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 847–855.

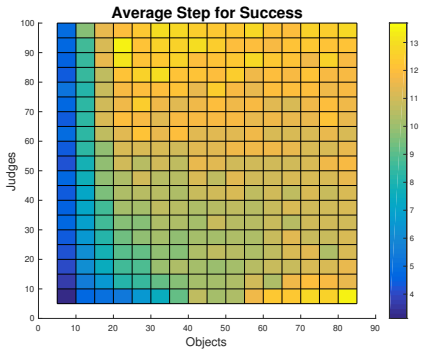
[10] Matthäus Kleindessner and Ulrike von Luxburg, "Uniqueness of ordinal embedding.," in *COLT*, 2014, pp. 40–67.

[11] Lalit Jain, Kevin Jamieson, and Robert Nowak, "Finite sample prediction and recovery bounds for ordinal embedding," *arXiv preprint arXiv:1606.07081*, 2016.

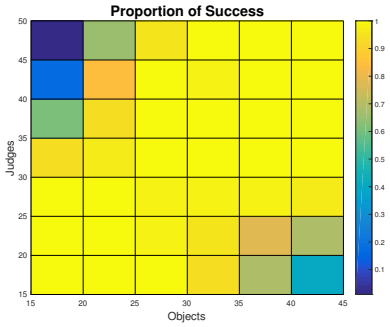
[12] Michael Grant and Stephen Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," `http://cvxr.com/cvx`, Mar. 2014.



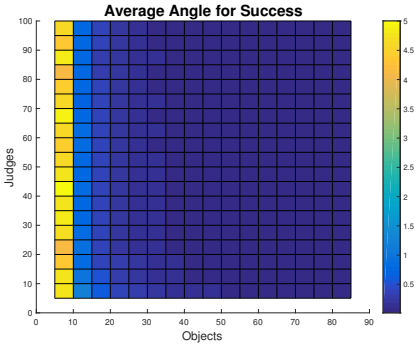
(a) $d = 2$ Proportion successful runs



(b) $d = 2$ Algorithm steps required



(c) $d = 10$ Proportion successful runs



(d) $d = 2$ Angle difference true embedding

Figure 2: This figure shows recovery results for the Aura algorithm for $d = 2$, maximum allowed 40 steps, and full rankings observed. (a) Proportion of 100 trials that succeed in finding an embedding to match the full given rankings. (b) Number of algorithm steps (two per iteration) required to converge to a matching ranking. (c) The average difference between all angles (computed from every triple of points) in the original and estimated embedding. (d) We increase the embedding dimension to $d = 10$ and show the proportion of 100 trials that succeed. Here we see a stronger trend that the proportion of objects to judges must not be too large or small.

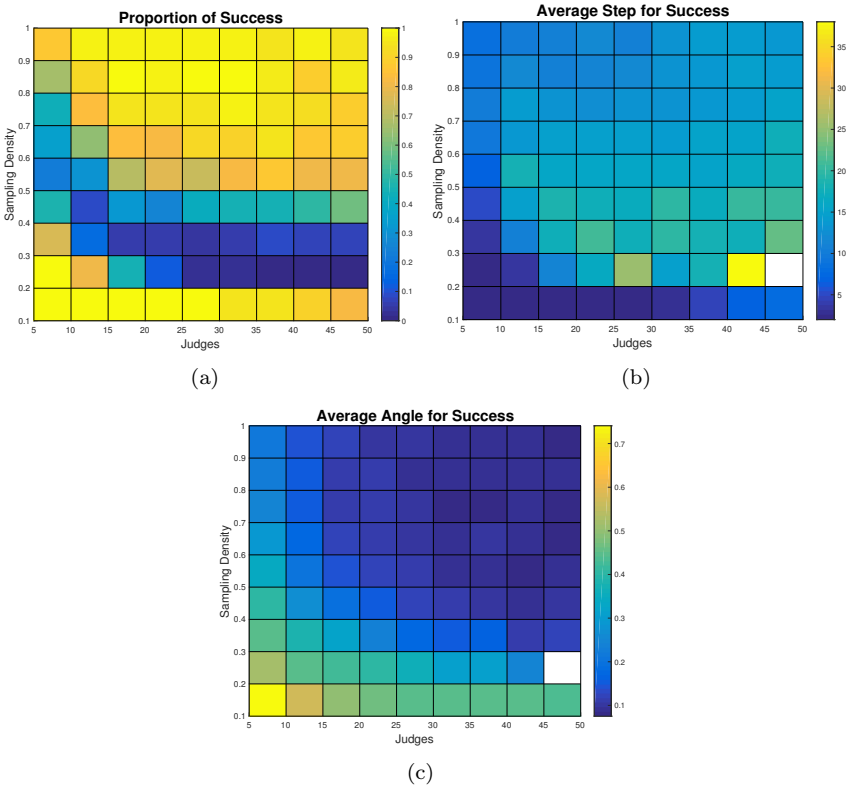


Figure 3: This figure shows recovery results for the Aura algorithm given partial rankings. $d = 2$, $n = 30$, varying sampling density and m. (a) Proportion of 100 trials that succeed in finding an embedding to match partial rankings. (b) Number of alg steps required to converge. (c) Average angle difference to original embedding.

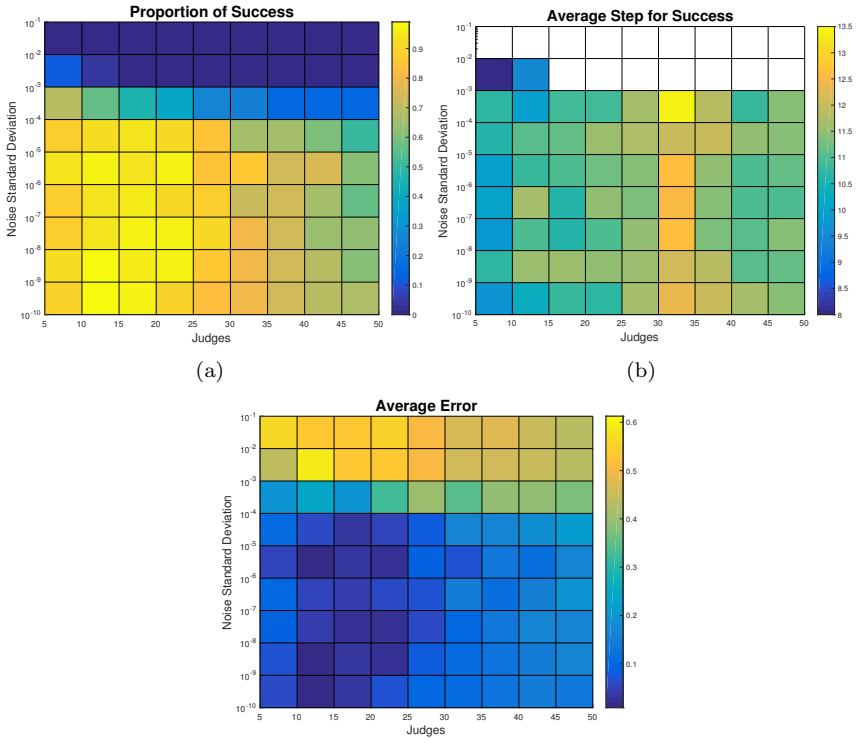
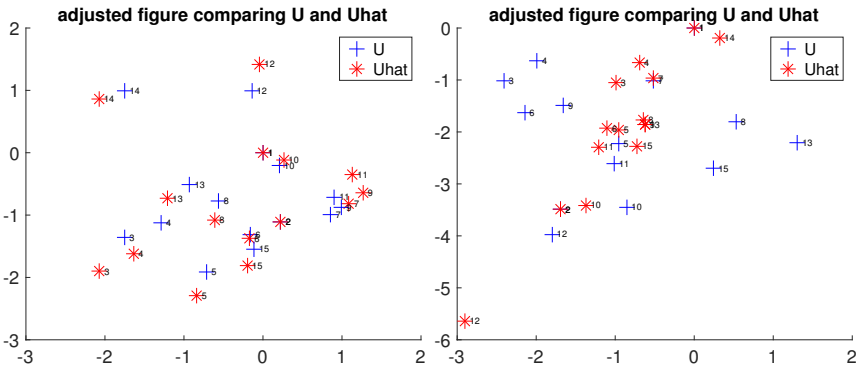
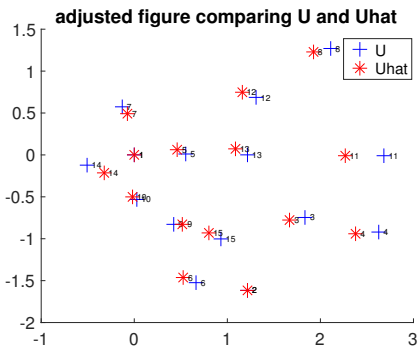


Figure 4: This figure shows recovery results for the Aura algorithm given noisy rankings. $d = 2$, $n = 50$, varying noise variance and m. Noise is added to each entry of the matrix $X = UW$, so it becomes impossible to match the rankings because the true data are not low-dimensional. (a) Proportion of 100 trials that succeed in finding an embedding to match noisy rankings. (b) Number of alg steps required to converge. (c) Error to true rankings (before noise) averaged over 100 trials.

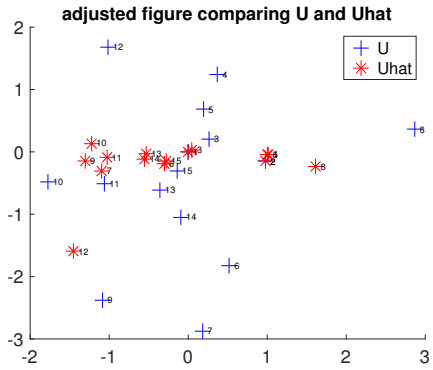


(a) Full rankings, successful embedding

(b) Full rankings, failed embedding



(c) Partial rankings, successful embedding



(d) Partial rankings, failed embedding

Figure 5: This figure shows example embeddings for successful runs and failed runs of the Aura algorithm for $d = 2$, $n = 15$, $m = 20$. We call them 'adjusted' because we have translated, scaled, rotated, and reflected our estimated embedding to match the original by first matching points 1 and 2, and then reflecting to make sure point 3 is on the same side of the 1-2 line. (a) Successful embedding with full data. (b) Failed embedding with full rankings. We see the embedding begins to collapse. (c) Successful embedding with 60% items ranked. (d) Failed embedding with 60% ranked.