Imma Sort by two or more attributes with Interpretable Monotonic Multi-Attribute Sorting

Yan Lyu, Fan Gao, I-Shuen Wu, and Brian Y. Lim

Abstract—Many choice problems often involve multiple attributes which are mentally challenging, because only one attribute is neatly sorted while others could be randomly arranged. We hypothesize that perceiving approximately monotonic trends across multiple attributes is key to the overall interpretability of sorted results, because users can easily predict the attribute values of the next items. We extend a ranking principal curve model to tune monotonic trends in attributes and present Imma Sort to sort items by multiple attributes simultaneously by trading-off the monotonicity in the primary sorted attribute to increase the human predictability for other attributes. We characterize how it performs for varying attribute correlations, attribute preferences, list lengths and number of attributes.We further extend Imma Sort with ImmaAnchor and ImmaCenter to improve the learnability and efficiency to search sorted items with conflicting attributes. We demonstrated usage scenarios for two applications and evaluate its learnability, usability, interpretability and user performance in prediction and search tasks. We found that Imma Sort improved the interpretability and satisfaction of sorting by ≥ 2 attributes. We discuss why, when, where, and how to deploy Imma Sort for real-world applications.

Index Terms—Multi-attribute sorting, decision making, interpretability, human predictability, predictive interpretability.

1 Introduction

When choosing an item from a list, it is common to sort the list by some attribute to allow comparison. However, consumer decision making often involves comparing across multiple attributes. For example, when choosing where to stay, we can sort hotel listings by price or distance to a point-of-interest. This can be mentally challenging because while one attribute is neatly sorted, the others could be randomly arranged. Common methods to consider multiple attributes include i) sorting by one attribute at a time, then switching to sort by another, ii) first filtering by one attribute, then sorting by another [1], iii) sorting by a weighted utility preference, e.g., simple additive weighting, that summarizes multiple attributes into a score [2], or iv) analyzing with large interactive spreadsheets [3], [4], [5]. Nevertheless, it remains challenging for users to seamlessly and intuitively sort with more than one attribute without performing context switching and storing the previous attribute values in working memory.

In this paper, we study the intuitiveness and interpretability of multi-attribute sorting and present Imma Sort, a sorting objective that optimizes the human predictability of sort results. Our **first key insight** is that sorting values by one attribute provides an intuitive basis for comparison, as it allows users to anticipate the attribute value of an item at a position in the list. That is, a simple sorted list is *interpretable* by its sorted attribute because it provides a *monotonically* increasing or decreasing trend, which improves the *ease of prediction* for the attribute. In contrast, that list will be less interpretable by other unsorted attributes due to the their lack of clear trend. However, we argue that users can still predict the value of an attribute as long as it has an *approximately monotonic trend*, i.e., roughly increasing or decreasing with *small fluctuations*. Thus, our **second key insight** is to trade off the monotonicity in the first sorted attribute to improve the trend perception of a second or other attributes. This improves the overall ease of prediction for multiple attributes. Fig. 1 and Fig. 2 show how Imma Sort reduces the monotonicity for one attribute to increase it for another, compared to simple one-attribute sort.

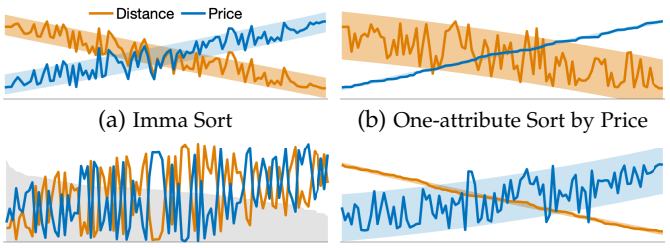
Hence, with Imma Sort, we support Interpretable, Monotonic, Multi-Attribute sorting. While most work on sorting and ranking have focused on the usefulness and accuracy of sorted results [2], [6] or interpretability of sort methods [3], [7], there is a lack of research on the interpretability of sorted results. To address this gap, this is the first work to define and study the *human predictability for multiple attributes* of sorted results. This adds to the fundamental understanding of their usability and opens up research on optimizing sorted results for human interpretability. It is related to but distinct from research in recommendation systems, interactive visualization, and interpretable machine learning.

Imma Sort leverages a ranking principal curve model (RPC) [8] to maintain the monotonicity of each attribute. We further improve it to enable users to tune the monotonic smoothness for different attributes based on their preferences. RPC was proposed to produce an aggregated ranking that has the most agreement with each individual ranking. However, it remains unclear how usable and interpretable this is to support search tasks. Our contributions are:

- The notion of *predictive interpretability* to quantify the usability of multi-attribute sorting as approximate monotonicity to perceive trends in sorted items. We support the tunability of trend smoothness for each attribute.
- Mock-up demonstrations and usage scenarios of Imma Sort for various consumer decision making applications.

- *Yan Lyu, I-Shuen Wu and Brian. Y. Lim are with the School of Computing, National University of Singapore, Singapore. E-mail:dcslyuy@nus.edu.sg, ishuen@comp.nus.edu.sg, brianlim@comp.nus.edu.sg.*
- *Fan Gao is with School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China. E-mail:fangao0802@gmail.com*

Fig. 1: Imma Sort orders items such that they are approximately monotonic in multiple attributes. In the sorted hotel list, the prices (blue line) are mostly increasing while the distances (orange line) to downtown are mostly decreasing.



(a) Imma Sort (b) One-attribute Sort by Price

(c) Simple Additive Weighting (d) One-attribute Sort by Distance

Fig. 2: A list of 100 Airbnb rentals[1] sorted by different techniques. x-axis shows the sort order, and y-axis shows values of two attributes, i.e., price and distance. (a) Imma Sort presents both attributes in monotonic trends with small fluctuation; (b) and (d) One-attribute sort presents one attribute monotonically but the other one randomly; (c) Simple Additive Weighting sorts by a two-attribute utility score (gray) but randomly orders both attributes.

- Characterization of how well Imma Sort could improve the ease of prediction in a simulation study when sorting items from both synthetic and real datasets.
- Evaluation showing that Imma Sort improved user prediction of item multi-attribute values, search performance to find user-preferred multi-attribute items, search satisfaction and helpfulness compared to baselines.

2 Scope and Related Work

In this paper, we focus on defining, improving and evaluating the human interpretability of a sort result, not model interpretability [9]. We investigated the trade-off of tuning approximate monotonic trends to transition from single-attribute to multi-attribute sort, which is separate from studying how to elicit user ranking or utility preferences [10], [11]. We do not propose a recommender system to prioritize items by relevance, but Imma Sort can be used to post-process and sort a list of top recommended items. Imma Sort provides a new arrangement of items in a list, not a new visualization or interactive technique [3], [5]. It can be added to existing list user interfaces, such as e-commerce product search results, by adding a sort mode and displaying items in the original list layout.

Many multi-attribute sorting/ranking and machine learning based ranking techniques have been developed to support decision making. We discuss how these techniques assume that users will trust their method, mathematics or mechanisms, yet produce results that appear unintuitively unsorted by single attributes and this hurts interpretation.

1. We randomly sampled rentals with location and price information from Airbnb dataset, http://insideairbnb.com/get-the-data.html

2.1 Multi-Attribute Decision Making Techniques

Ranking items based on multiple attributes has been well-studied in decision theory literature [2]. Simple additive weighting (SAW) [12], [13] is a popular method which ranks item based on a utility score calculated as a weighted sum of multiple attributes. More sophisticated techniques include multi-attribute utility theory (MAUT) [14] that introduces uncertainty about the consequences, analytic hierarchy process (AHP) [15] that structures the decision making process through pairwise comparisons, TOPSIS [16] that ranks items based on the distances to positive and negative ideal solutions, and outranking methods such as PROMETHEE [17], [18] and ELECTRE [19] that consider the relative ordering of each single attribute. The resulting rank of an item represents a summary of complicated relationships between its attributes and those of the other items. Such results are often hard for users to understand why an item has a lower or a higher rank than others [3], and can stifle decision making.

Intuitiveness and Interpretability. To interpret multi-attribute sorting/ranking, existing techniques provide users visualization interfaces to interact with ranking and attribute values. Users can manually refine attribute combinations [3], [20], [21], track rank changes [3], [22], [23], inspect attribute weights or preferences [4], [24], [25], and compare multiple alternative rankings [3], [26], [27]. Some also focus on eliciting user preference on multiple attributes from user interactions [5], [10]. However, while these visual tools provide deep insight and support rigorous multi-attribute decision making, they are designed for data analysis, i.e., users need to spend a lot of time to understand a ranking through a series of interactions. Also, they focused on evaluating the basic usability of sorting [3], [20], [21], [22], [26] rather than the interpretability of sorted results. With Imma Sort, we focus on the more lightweight interaction of browsing a sorted list, which is also suitable for frequent and limited consumer decision-making [28], such as choosing hotels and foods, instead of intensive analytical interaction. This requires the sorted list itself to be intuitively interpretable. This is the first work to evaluate the intuitive interpretability and ease of prediction of sorted results.

2.2 Machine Learning based Ranking Techniques

More recently, many machine learning methods have been developed to generate the best ranking in the domain of information retrieval and recommendation systems. Learning to rank algorithms [6], [29], [30], [31], [32], [33] apply supervised learning to rank items based on training data that consists of lists of items with ground-truth rank scores or partial orders. While these algorithms are designed to best

satisfy user preferences or information relevance, the ranked attributes are usually latent features, such as factorized user-item ratings. Therefore, the result rankings are even more uninterpretable with respect to the visible attributes. In addition, unsupervised ranking techniques aggregate different rankings generated under multiple criteria [34], [35], [36] by optimizing a "consensus" of the rankings. The consensus has been defined by various distance metrics between the rankings, such as Kendall's τ [34], Spearman ρ [37] and Hausdrodff distance [38]. Li et al. [8] further extended the concept of consensus to an *"order-preserving"* monotonicity requirement to produce an aggregated ranking that is *"closer to ground truth"*, i.e., in more agreement with each individual ranking. However, none of these works evaluated the interpretability of the aggregated ranking result.

Intuitiveness and Interpretability. The increasing sophistication in models have driven the need for explanation techniques, such as LIME [39], which have been adapted to explain black-box rankers [7], [40], [41]. These explanation by relevance [42], association [43], user preferences [9], using auxiliary information such as textual sentences [44] or visual image [45], and includes designing novel explanation interfaces [46]. However, these explanations mostly focus on why the top items were selected from the search or recommendation models, and not about their relative order, i.e., why an item is ranked higher/lower others. In contrast, Imma Sort focuses on human ease of prediction for multiple attributes and optimizes their monotonicity for intuitive interpretability. Specifically, we leverage a ranking principal curve model proposed by [8], and extend its capability with a tunable parameter to trade off monotonicity between attributes based on user preference. Our key contribution is defining and studying human interpretability and ease of prediction for multiple attributes of sorted results.

3 Imma Sort: Intuition and Applications

To help appreciate the value of Imma Sort and understand how to use it, we describe its intuition and requirements, and illustrate its usage with three application examples.

3.1 Intuition and Requirements

Sorting helps users to intuitively compare and search for items. We assert that this is because users can perceive monotonic trends in the sorted attribute, and easily predict where an item with an expected value will be and predict that items far away will have very dissimilar values. We call this perception *predictive interpretability*. For example, sorting products with price in ascending order helps users to anticipate that the next item will be a higher price, and an item with a much higher price will be much farther along. Users can stop considering subsequent items when they anticipate that remaining items will be more expensive than a threshold. Therefore, we hypothesize that monotonicity in the attributes of a sorted list helps users to predict the attribute values of subsequent items and quicken search.

Likewise, we argue that predictive interpretability helps to reduce human memory load when reading and searching sorted lists of items, including simple list interfaces for e-commerce applications, such as finding a hotel, movie, or food dish. When items are not sorted, users have to "jump around" to find comparable items with similar values and store them in working memory as a ranked list with descending or ascending values. Participants in our qualitative user study reported this mental effort. When items are neatly sorted, users can see the clear trend of values across neighboring items and externalize the information in the list interface instead. This clear trend with higher predictive interpretability helps users to predict that there are not likely to be any items with closely comparable attribute values far away. So, users would not need to mentally construct their own ranked list when searching farther down the list, and this reduces the strain on working memory. The decrease in memory demands can improve consumer choice and ease of decision-making [47], [48]. *Therefore, we hypothesize that improving predictive interpretability can improve user satisfaction in search tasks on sorted lists by reducing cognitive load*.

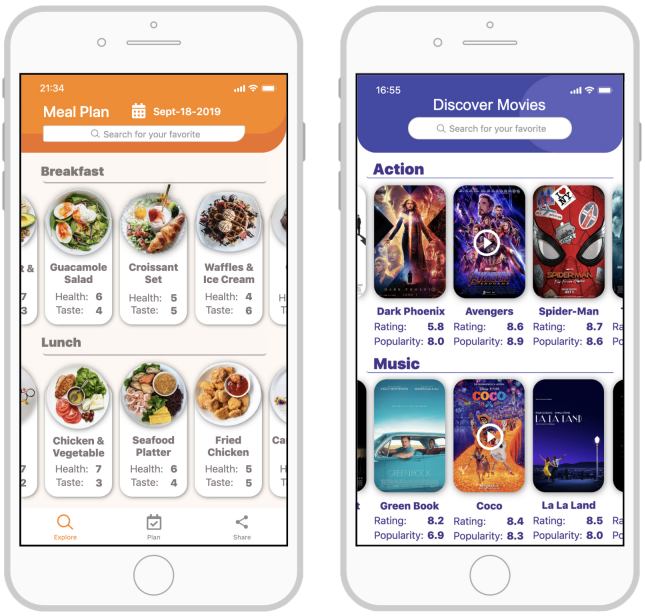
Furthermore, many choice problems often involve multiple attributes and this is more mentally challenging. When one primary attribute is neatly sorted, other secondary attributes could be unsorted and randomly ordered. Users could strain their working memory by searching an unsorted list as aforementioned, or toggle between re-sorting by different attributes or pre-filter by other attributes. Such context switching also puts a strain on working memory and hurts consumer choice and ease of decision making [47], [48]. Therefore, this motivates us to make multiple attributes more predictable by designing Imma Sort to present items sorted simultaneously by two or more attributes.

With Imma Sort, users will be able to perceive approximate monotonic trends for more than one attribute, and more easily predict values of multiple attributes as they navigate down the sorted list. Imma Sort achieves this by trading off the monotonicity in the first attribute to increase the monotonicity in the second or subsequent attributes. It does so in a tunable manner to give the user and application designer more control on the sorting outcome. Therefore, the first requirements for Imma Sort to have:

1) *Balanced monotonic trends* to trade off the monotonicity of the primary sorted attribute to increase the monotonicity (ease of prediction) for other attributes.
2) *Smoothness preference weights* for the monotonic smoothness of each attribute to be tunable based on preference weights, i.e., more weighted attributes can be more monotonic than the less weighted attributes.

The novel sort order provided by Imma Sort also presents a new interaction challenge when the sorted attributes are negatively correlated. Assuming that higher values of each attribute is better, multi-sorting positively correlated attributes can rank the best item at the front, but multi-sorting negatively correlated (conflicting) attributes, will place the best item somewhere in the middle of the list. This issue is not unique to Imma Sort, since sorting with a single attribute may also not place the best item in the first position. However, with Imma Sort, users can more easily identify the best item as a compromise between the conflicting attributes. This manifests as an "X" shape trends pattern for both attributes (Fig. 1 and Fig. 2(a)). Therefore, we propose additional requirements for Imma Sort:

3) *Anchoring* to highlight the best compromise item to help



(a) Food: Tastiness & Healthiness (b) Movies: Ratings & Popularity

Fig. 3: Mock-ups of list user interfaces using Imma Sort to sort (a) food dishes based on conflicting attributes Tastiness and Healthiness (ImmaCenter), and (b) movies based on correlated attributes Ratings and Popularity (regular Imma).

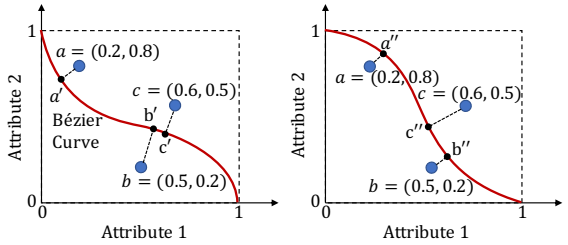
users to quickly locate and compare around that item. Users can conventionally search down the list from the first (left/top) position and note how far down this item is located. We call this Imma Sort variant ImmaAnchor.

4) *Searching from the center* to reduce the initial search time by positioning users to start searching at the best (anchor) item. This is especially useful if the first item is very far from the best item. Note that the best item may not be located in the center, just not the first position. We call this variant of Imma Sort as ImmaCenter. Given the unconventional interaction of ImmaCenter, we studied its learnability in a qualitative study (Section 7.3).

A key requirement of simple interfaces is to not impose a high cognitive load on users. By clarifying trends, Imma Sort improves predictive interpretability and helps to reduce mental load. This facilitates users to perform multi-attribute sorting with simple list interfaces, rather than depending on sophisticated spreadsheets or data analytic visualizations.

3.2 Application Examples

Imma Sort can be integrated into many real-world applications such as searching and recommendation systems to help with choice-making. We demonstrate its usefulness in three real-world illustrations to: select a hotel based on price and distance to point-of-interest (Fig. 1), select a food dish that is both healthy and tasty (Fig. 3(a)), and select a movie that highly rated and popular (Fig. 3(b)). The first example shows a long list to allow longer-span examination of the sort order. The first two cases sort conflicting attributes, while the third sorts correlated attributes. These simple examples demonstrate the usefulness of Imma Sort for two attributes, but Imma Sort can also handle > 2 attributes. We will detail the usage scenarios of Imma Sort in Section 6.



(a) $a' \succ b' \succ c'$ (b) $a'' \succ c'' \succ b''$

Fig. 4: Examples of two monotonic cubic Bézier Curves. (a) shows projections of data points a, b and c on one curve, the sequence of the projections on the curve suggests a sorting order $a \succ b \succ c$. (b) shows projections of a, b and c on another curve, suggesting another order $a \succ c \succ b$.

4 Technical Approach

In the next section, we describe technical approach of how we achieve sorting by multiple attributes for Imma Sort to satisfy the four aforementioned requirements. We describe the intuition of the ranking principal curve (RPC) model [8] and why it is suitable for Imma Sort (Section 4.1), its solution (Section 4.2), and our extension to control the monotonic smoothness of different attributes to help users prioritize attributes based on preference weights (Section 4.3).

4.1 Mathematical Intuition

Sorting by one attribute may cause other attributes to be randomly ordered, depending on the strength of attribute correlations. Two attributes with *perfect rank correlated* (Spearman coefficient $= 1$ or -1) can both be strictly monotonic at the same time, while less correlated attributes cannot. Consider three items as data points where two attributes have rank correlation -0.5: $a = (0.2, 0.8)$, $b = (0.5, 0.2)$ and $c = (0.6, 0.5)$. Sorting by Attribute 1, i.e., $a \succ b \succ c$ will not arrange Attribute 2 strictly monotonically, and vice versa.

Since there is no way to make *partially correlated attributes* strictly monotonic simultaneously, we seek to make the trends approximately monotonic by perturbing data points. Specifically, we look for approximations of the three data points, a', b', and c', such that, they can be sorted monotonically for both attributes. The objective is to find the best approximations that minimize the perturbation, i.e., $||a - a'||^2 + ||b - b'||^2 + ||c - c'||^2$. Principal component analysis (PCA) models [8], [49], [50] can be used to find such approximations by searching for a smooth curve that minimizes the sum of squares distances from the original item data points to their orthogonal projections on the curve.

We used a monotonic cubic Bézier curve for the data fitting just as in [8] to obtain the projections with monotonic points (see Fig. 4(a)). Fig. 4(a) illustrates a cubic Bézier curve projection that has Attribute 2 monotonically decreasing as Attribute 1 increases. If we project the data points a, b and c on this curve, their projections a', b', c' are sorted strictly monotonically in the sequence on the curve, i.e., $a' \succ b' \succ c'$. We can apply this sorting order for the original data points, i.e., $a \succ b \succ c$, but this sort order will only be approximately monotonic, with smoothness depending on how small the perturbations were. There can be other Bézier curve projections that produce different sort orders (e.g.,

Fig. 4(b) sorts with $a \succ c \succ b$), but we seek to obtain one that maximizes smoothness, i.e., minimizes perturbation.

4.2 Ranking Principal Curve (RPC) Model

Here we describe the Ranking Principal Curve model to find the best cubic Bézier curve [8] for multi-attribute sorting.

4.2.1 Cubic Bézier Curve

A cubic Bézier curve, $\boldsymbol{f}(s, P)$, is defined as

$$(1-s)^3\boldsymbol{p_0} + 3s(1-s)^2\boldsymbol{p_1} + 3s^2(1-s)\boldsymbol{p_2} + s^3\boldsymbol{p_3}, \quad (1)$$

where $P = \{\boldsymbol{p_0}, \boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{p_3}\}$ are the four control points that determine the curve's shape, and s specifies a position index on the curve, $0 \leq s \leq 1$ (see Fig. 5). As s increases from 0 to 1, the curve starts at $\boldsymbol{p_0}$ ($s = 0$) going towards $\boldsymbol{p_1}$, then $\boldsymbol{p}_2$ and ends at $\boldsymbol{p_3}$ ($s = 1$). Each point is in $d \geq 2$ dimensions, one for each attribute. We can control the locations of the four controls points to make the curve strictly monotonic [8]. Let $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_d)^T$ denote the *monotonicity index* for d attributes, $\alpha_a = 1$ indicates that the ath attribute should be sorted in ascending order, and $\alpha_a = -1$ indicates descending order. This can be predefined by users, or be data-driven and set to match the correlations between attributes. [8] proved that the curve $\boldsymbol{f}(s, P)$ is strictly monotonic if and only if $\boldsymbol{p_0} = (1 - \boldsymbol{\alpha})/2$, $\boldsymbol{p_3} = (1 + \boldsymbol{\alpha})/2$, and $\boldsymbol{p_1}$ and $\boldsymbol{p_2}$ are bounded such that $\boldsymbol{p_1}, \boldsymbol{p_2} \in [0, 1]^d$. For example, if we set $\boldsymbol{\alpha} = (1, -1)^T$, i.e., Attribute 1 ascending and Attribute 2 descending, then $\boldsymbol{p_0} = (0, 1)^T$, $\boldsymbol{p_3} = (1, 0)^T$ and $\boldsymbol{f}$ monotonic if $\boldsymbol{p_1}$ and $\boldsymbol{p_2}$ are within $[0, 1]^2$ (see Fig. 5).

4.2.2 Problem Definition and Solution

Let $X = \{\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_n}\}$ denote the set of n data points to be sorted, where each data point $\boldsymbol{x_i}$ has d attributes, i.e., $\boldsymbol{x_i} = (x_{i,1}, x_{i,2}, ..., x_{i,d})^T$. For a cubic Bézier curve $\boldsymbol{f}(s, P)$ with control points P, it projects the data point $\boldsymbol{x_i}$ to s_i on the curve with projected value $\boldsymbol{f}(s_i, P)$. For the smoothest approximate monotonicity projection, s_i will be the closest point on the curve to x_i, i.e., $s_i = \text{argmin}_{0 \leq s \leq 1} ||\boldsymbol{x_i} - \boldsymbol{f}(s, P)||$. We seek to find the set of control points P that defines the best monotonic curve $\boldsymbol{f}(s, P)$, such that the sum of squared distance from $\boldsymbol{x_i}$ to its projection $\boldsymbol{f}(s_i, P)$ is minimized, i.e.,

$$\text{minimize} \quad \sum_{i=1}^{n} ||\boldsymbol{x_i} - \boldsymbol{f}(s_i, P)||^2, \quad (2)$$

$$\text{subject to} \quad s_i = \text{argmin}_{0 \leq s \leq 1} ||\boldsymbol{x_i} - \boldsymbol{f}(s, P)||, \quad (3)$$

$$\boldsymbol{p_0} = (1 - \boldsymbol{\alpha})/2, \boldsymbol{p_3} = (1 + \boldsymbol{\alpha})/2, \quad (4)$$

$$\boldsymbol{p_1}, \boldsymbol{p_2} \in [0, 1]^d. \quad (5)$$

Note that this is a non-linear optimization problem, since the intermediate variable s_i is determined by decision variables P (Eq. 3) but cannot be expressed by an explicit function of P. To solve this problem, [8] adopted the classic Hastie-Stuetzle principal curve algorithm [49]. The algorithm starts with a random set of control points P that satisfy Eqs. 4 and 5, iteratively calculates data projections $\boldsymbol{f}(s_i, P)$ on the curve and updates P by minimizing Eq. 2. Li et al. have proved this algorithm converges in limited iteration steps and with a computational complexity of $O(n)$ [8] to search for a local optimal Bézier curve.

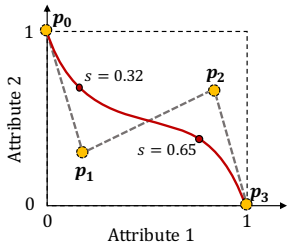


Fig. 5: A cubic Bézier curve with four control points $\boldsymbol{p_0}, \boldsymbol{p_1}, \boldsymbol{p_2}$, and $\boldsymbol{p_3}$. The points on the curve can be identified by a position parameter s, e.g., $s = 0.32$ and $s = 0.65$.

4.3 Adding Monotonic Smoothness Control

To control which attributes should be more smoothly monotonic, we introduce *smoothness preference weights* on different attributes to the RPC model. Specifically, we insert the weights into the distance measurement of $||\cdot||$ in Eqs. 2 and 3. Let $\boldsymbol{w} = (w_1, w_2, ..., w_d)^T$ denote the weights on d attributes. Then, the total distance between data points and their projections on the curve can be measured by a weighted Euclidean distance, i.e., Eq. 2 can be written as

$$\varepsilon = \sum_{i=1}^{n} \sum_{a=1}^{d} w_a (x_{i,a} - \boldsymbol{f}(s_i, P)_a)^2, \quad (6)$$

where $\boldsymbol{f}(s_i, P)_a$ denotes the ath dimension of Bézier curve and refers to the projection of the ath attribute. Similarly, the projection of the data points should be re-defined by minimizing the weighted Euclidean distance, i.e.,

$$s_i = \text{argmin}_{0 \leq s \leq 1} \sqrt{\sum_{a=1}^{d} w_a (x_{i,a} - \boldsymbol{f}(s, P)_a)^2}, \quad (7)$$

where $0 \leq w_a \leq 1$ and $\sum_{1 \leq a \leq d} w_a = 1$. The weights help users to specify their preference on attributes. The higher weight on an attribute, the more monotonic the attribute will be. At one extreme, if $w_a = 1$ and $w_{\neq a} = 0$, then the curve will generate single-attribute sorting by the ath attribute. We adapted the Hastie-Stuetzle principal curve algorithm [49] with Eqs. 6 and 7 (details in Appendix).

5 Characterization Simulation Studies

Having defined the technical approach for Imma Sort, we next evaluated its performance on synthetic and real datasets. We hypothesized that improving approximate monotonicity should improve ease of prediction. We conducted simulation experiments to characterize when and how much Imma Sort improves the ease of prediction on multiple attributes. We first describe the estimation of ease of prediction and then compare Imma Sort with baseline methods on both synthetic and real datasets.

5.1 Prediction Interval to estimate Ease of Prediction

We use *prediction interval* to quantify and simulate users' ease of prediction on attribute values in a sorted list. A prediction interval (PI) is an estimate of an interval within which a future observation will fall, with a specified degree of confidence, given what has already been observed [51]. While a confidence interval describes uncertainty for a single stochastic variable, a PI describes uncertainty for a stochastic variable function. A clearer trend with smoother

monotonicity will have smaller prediction interval width ($|PI|$) and be more human predictable. In our prediction user study (Section 7, Table 1), we found that $|PI|$ is consistent with users' uncertainty about their prediction. Specifically, for each attribute, we fit a cubic regression model on the sorted items with position as the factor and attribute value as the response, and calculated the 95% $|PI|$.

5.2 Baseline Techniques

We compare Imma Sort against three common baseline sort techniques: i) Simple Additive Weighting (SAW) that sorts by the weighted sum of normalized attribute values; ii) Single Sort by the primary Attribute (ByA1) that sorts based on the attribute on which a user has the highest preference weight; and iii) Single Sort by the secondary Attribute (ByA2) that sorts items by the attribute on which a user has the second highest preference weight.

5.3 Simulation Experiments with Synthetic Data

To characterize how Imma Sort supports better ease of prediction, we synthesized different datasets with varying characteristics for search with varying requirements.

5.3.1 Factors for varying datasets and search parameters

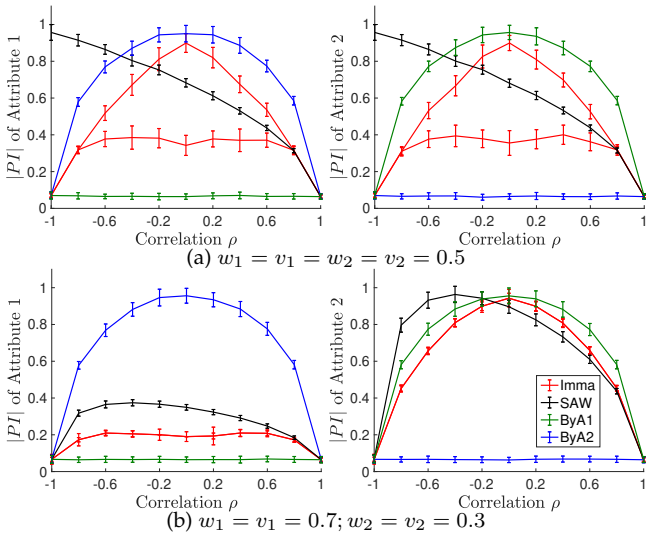
We identified four factors that may affect the performance of multi-attribute sorting techniques. 1) *Correlation* between attributes, ρ, as calculated by the Spearman's rank coefficient to depict the extent to which, as one attribute increases, the other variable tends to increase or decrease [52]. 2) *Attribute preference weight* to indicate user preference on different attributes. For Imma Sort, attribute preference, denoted by $\boldsymbol{w}$, refers to how monotonic of a trend in the attribute the user would want; an attribute a with higher preference w_a will be smoother than another with lower preference. For SAW, attribute preference, denoted by $\boldsymbol{v}$, refers to the relative importance in utility of the attribute, i.e., its importance to the weighted sum of partial utilities. 3) *List length*, n, as the number of items to be sorted. We hypothesize that longer lists would have smaller $|PI|$, because more items can be rearranged to produce smoother and more monotonic trends. 4) *Number of attributes*, d, by which to sort items simultaneously. Sorting by more attributes would increase $|PI|$, since it will be more difficult to maintain and balance the monotonic trends of multiple attributes in the sort result.

We randomly generated synthetic datasets with correlation ρ varying from -1 to 1, list length n varying from 10 to 100 and attribute number d varying from 2 to 6. Each simulation was repeated 100 times with different datasets to account for variability in slight differences in item attribute values in the same simulation settings.

5.3.2 Results

We report simulation results to show the impact of the four factors, and conclude when and how much Imma Sort improves ease of prediction compared to baseline techniques.

Increasing attribute correlation magnitude for both positive and negative correlation decreased the prediction interval width $|PI|$ and improved predictive interpretability of attributes for most sort techniques (see Fig. 6). This



(a) $w_1 = v_1 = w_2 = v_2 = 0.5$

(b) $w_1 = v_1 = 0.7; w_2 = v_2 = 0.3$

Fig. 6: Prediction interval width $|PI|$ of both Attributes 1 and 2 decreased (better predictive interpretability) with increasing correlation magnitude for balanced (a) and imbalanced (b) weights preferences. Imma Sort has two lines in (a) to indicate its bimodal distribution for weak correlations ($-0.6 < \rho < 0.6$). Error bar is the standard deviation.

general trend is expected, but Fig. 6 shows how $|PI|$ increases sharply as correlation magnitude decreased to 0. For simplicity in our illustration, we limited our analysis to number of attributes[1] $d = 2$, and list length[2] $n = 100$. Imma Sort decreased the $|PI|$ for the second attribute (Attribute 2) by increasing the $|PI|$ for the first attribute (Attribute 1). Although the $|PI|$ for Imma Sort was symmetrical for positive or negative correlation, the $|PI|$ of SAW was asymmetrical. For more negative attribute correlations , SAW sorted items were harder to predict (higher $|PI|$), while Imma Sort were easier to predict (lower $|PI|$). Across the range of attribute correlations, Imma Sort had lower $|PI|$ values for negative correlations and similar $|PI|$ for positive correlations compared to SAW (see Fig. 6(b)), indicating that it achieved better predictive interpretability overall. Single sorting, ByA1 or ByA2, perfectly sorted Attribute 1 or 2, respectively; so the sorted attribute had the smaller $|PI|$ but the unsorted attribute had the larger $|PI|$.

Increasing attribute preference weight of Attribute 1 decreased the prediction interval width $|PI|$ and improved predictive interpretability of Attribute 1 for Imma Sort and SAW, but increased the $|PI|$ for Attribute 2 (see Fig. 6(b)). Imma Sort perturbed Attribute 1 values less than Attribute 2 values, leading to smoother monotonicity for Attribute 1 than Attribute 2. SAW similarly had smaller $|PI|$ for Attribute 1 than Attribute 2, because it sorted items based on the weighted sum of normalized attributes, which was more influenced by Attribute 1 than Attribute 2. However, with its focus to improve monotonicity, Imma Sort had lower $|PI|$ than SAW for Attribute 1 for all attribute correlations, and lower $|PI|$ for Attribute 2 for negative attribute correlations. Note that Single Sort was not influenced by imbalanced attribute preference weights. We also found that for balanced

1. For simplicity to limit the number of additional parameters that needed to be set for 3 or more attributes.

2. $|PI|$ is mostly stable by 100 items, as shown in Fig. 7.

attribute preference ($w_1 = w_2 = 0.5$), the $|PI|$ for Imma Sort had a bimodal distribution when attributes were not strongly correlated, i.e., $-0.6 < \rho < 0.6$ (see Appendix). This indicates that slight differences in items led to either a higher or lower $|PI|$, but not a $|PI|$ value in between. To make these trends clear, we applied k-means clustering on $|PI|$ values for each ρ, and chose $k = 1$ or 2 based on the silhouette criterion. We plotted lines through the k mean values in Fig. 6(a). This bimodal distribution on $|PI|$ did not occur for unequal weights (Fig. 6(b)).

Increasing list length slightly decreased the attribute prediction interval width $|PI|$ and slightly improved predictive interpretability for all sort techniques (see Fig. 7). This was because more items could be re-arranged to produce smoother and more monotonic trends. For simplicity, we limitted the analysis to two attributes[1], which are conflicting ($\rho = -0.4$)[3], with Attribute 1 preferred over Attribute 2 ($w_1 = v_1 = 0.7, w_2 = v_2 = 0.3$)[4] The $|PI|$ of both attributes with Imma Sort was consistently lower than with SAW, and consistently between that of the sorted (A1) and unsorted (A2) attributes with Single Sorts.

Smaller number of attributes led to lower prediction interval width $|PI|$ for all sort techniques (see Fig. 8(b)). Fig. 8(a) shows that, with Imma Sort, the preferred attribute was less predictable (higher $|PI|$) with more attributes, while all other less preferred attributes were slightly more predictable (slightly lower $|PI|$). For simplicity, in this analysis, we set a pairwise correlation coefficient between attributes of 0.4[5]. We define the attribute preference weights as an arithmetic series, $w_{a+1} = w_a - \Delta w$, where $w_d = \Delta w$ and the weights sum to 1; e.g., when $d = 4$, the weights of four attributes are $0.4, 0.3, 0.2$ and 0.1, respectively. Fig. 8(b) shows how less preferred attributes had higher $|PI|$ for all sort techniques. Overall, Imma Sort had a slightly lower $|PI|$ than SAW for all the attributes, and lower $|PI|$ than Single Sort for all non-primary attributes. The difference between them was small because at moderate pairwise correlation ($\rho = 0.4$), as we had shown that Imma Sort and SAW had similar $|PI|$ (see Fig. 6). The difference should be more pronounced for negative pairwise correlations.

In summary, Imma Sort provided a predictive interpretability that was upper-bounded by Single Sort (better lower $|PI|$ than the unpreferred attributed and $|PI|$ capped at as good as the preferred attribute) and is mostly better than SAW for varying attribute correlation, attribute preference weight, list length, and number of attributes.

5.4 Simulation Experiments with Real Data

We used two real-world datasets — Airbnb[6] and USDA Food Nutrition[7] — to evaluate Imma Sort performance in the real-world applications. Specifically, we studied the correlations of combinations of attributes for typical sort use cases to evaluate how predictable the sort results would be.

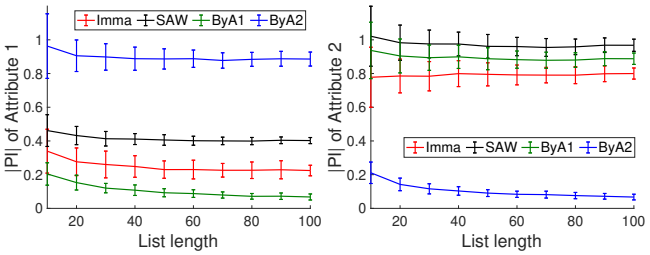
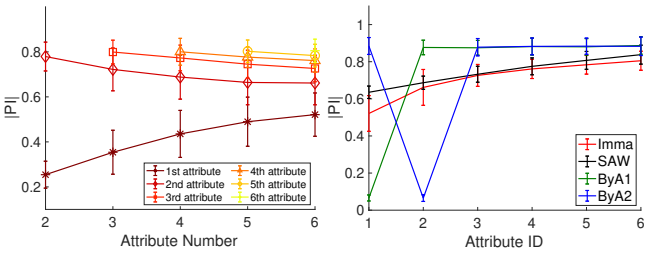


Fig. 7: Prediction interval width $|PI|$ of Attributes 1 and 2 decreased (better interpretability) with increasing list length.



(a) $|PI|$ for Imma Sort only (b) $|PI|$ for all sort techniques

Fig. 8: Prediction interval width $|PI|$ converges with (a) increasing number of attributes for Imma Sort and (b) for less prioritized attributes for all sort techniques.

5.4.1 *Airbnb data*

We collected the Airbnb dataset of New York City (NYC), which contains 48,800 Airbnb rooms with location and price. Rather than sort over all items, we simulated the use case of retrieving subsets of 100 relevant items that are nearest to one of the top 100 tourist attractions[8] in NYC. For simplicity, we computed relevance using an equally-weighted (i.e., $v_1 = v_2 = 0.5$) SAW based on utility of min-max normalized short distance and low prices. Since price and distance are conflicting attributes, as expected, most of the item subsets had negative attribute correlations ($\rho = -0.9$ to -0.5, see Fig. 9(a)). We sorted each subset of items using Imma Sort and three baselines, SAW, ByA1 (distance) and ByA2 (price). For simplicity, we set equal attribute preference weights for Imma Sort and SAW. Fig. 10(a) shows the prediction interval width ($|PI|$) of the four techniques, showing that Imma Sort improved ease of prediction for both attributes — distance and price — compared to baseline techniques for sorting rental items from the Airbnb dataset. Thus, Imma Sort is well-suited for the use case of sorting accommodations based on the conflicting attributes of distance and price.

5.4.2 *USDA Food Nutrition Data*

We collected the Food Nutrition dataset from the USDA National Nutrient Database containing 320K food products in 223 categories (cereal, yogurt, etc.). We focused on the top 20 categories, and 3 to 11 most common nutrition facts (carbohydrate, protein, etc.)[9] as the attributes to sort by in each category. We simulated the use case of retrieving subsets of 50 relevant items (highest utility) in the same category.

3. Since our analysis of datasets (Fig. 9) found that attributes tended to be negatively correlated, and we chose a moderate correlation strength; strong correlations would only need single sort (Fig. 6), and no correlation provided the least information for multi-attribute sorting.

4. Moderately imbalanced between (0.5, 0.5) and (1, 0).

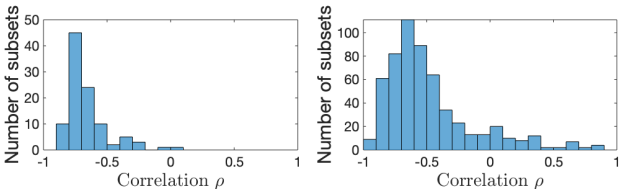
5. It was difficult to simulate data having more than two attributes with pairwise negative correlation, since not all pairwise correlations can be negative. So we simulated with positive moderate correlation.

6. http://insideairbnb.com/get-the-data.html

7. https://fdc.nal.usda.gov/

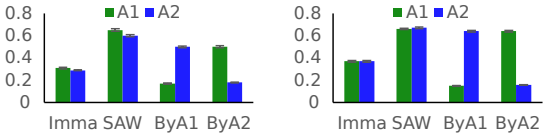
8. https://www.google.com/travel/

9. We define a common nutrient by checking whether more than half of the food products in that category have the nutrient recorded.



(a) Airbnb Rentals (b) USDA Food Nutrition

Fig. 9: Distribution of correlation between pairwise attributes of item subsets real-world datasets. Most lists had negative correlations for which Imma Sort is well-suited.



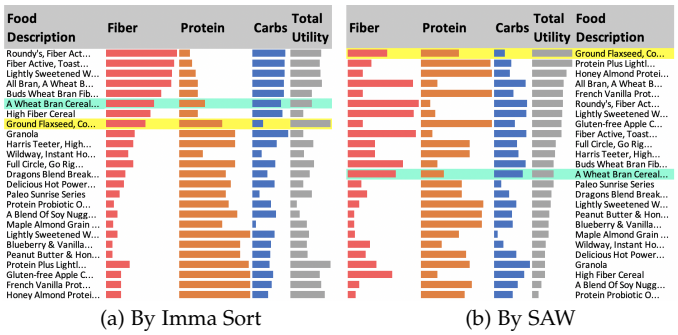
(a) Airbnb Rentals (b) USDA Food Nutrition

Fig. 10: Prediction interval widths $|PI|$ for different sort techniques on real-world datasets. Imma Sort had smaller $|PI|$ than SAW, and good compromise compared to ByA1 and ByA2. Error bar indicates standard error.

We computed a healthiness utility as the equally-weighted sum of normalized attributes (e.g., low carbohydrates, high proteins, high fiber, low fats). For evaluation, we sorted items by combinations of two attributes. This resulted in 590 sorted subsets. We found that most attribute pairs were negatively correlated (Fig. 9(b)). Fig. 10(b) shows that Imma Sort improved ease of prediction (lower $|PI|$) compared to SAW for both attributes, and compared to ByA1 for Attribute 2 and ByA2 for Attribute 1. Therefore, Imma Sort is well-suited to present users with the most predictively interpretable multi-attribute sorted results.

5.5 Implications from simulation results

Our simulation experiments with synthetic datasets found that, for all the sort techniques, ease of prediction is highly influenced by attribute correlations, but marginally influenced by list length and number of attributes. Consequently, in our user studies (Section 7), we chose attribute correlation as one of the key independent variables, but fixed list length and number of attributes throughout the user studies. Comparing between sort techniques, we found that Imma Sort had the lowest prediction interval width $|PI|$ among sort techniques (naturally, except ByA1 for Attribute 1 or ByA2 for Attribute 2) across the factors of attribute correlation, list length, and number of attributes. It retains low $|PI|$ for long lists of items with few attributes that were moderately correlated. Our simulation experiments with two real datasets found that typical lists have attributes that are negatively correlated for which Imma Sort also had a balanced and lowest $|PI|$ compared to the other baseline techniques. Although promising, these results are based on simulated human ability, so we validate them in user studies of prediction and search tasks, which we describe later in Section 7. Next, we discuss how using Imma Sort can help make searching among items more intuitive.



(a) By Imma Sort (b) By SAW

Fig. 11: Example cereal food items sorted by (a) Imma Sort and (b) SAW. The width of each nutrient column represents smoothness preference weight (Imma Sort) or importance (SAW). The length of each bar within each column represents the relative nutrient amount. A healthier item has higher fiber and protein, and lower carbohydrates. Yellow highlight indicates item with best utility (anchor) and blue-green highlight indicates plausible final user-selected item.

6 Usage Scenarios for Search Task

With two usage scenarios with different comparisons, we now discuss how the improved ease of prediction when using Imma Sort, as evidenced from our simulation studies, can help to improve user navigation intuition, search experience and performance. This drives our hypotheses for our quantitative user study (described in the next section).

6.1 Choosing Airbnb rooms: Imma Sort vs. Single Sort

Consider the user task to find a rental from the Airbnb database that simultaneously has the lowest price and shortest distance to a tourist attraction. After an initial query, the retrieved items are typically sorted by decreasing relevance (e.g., by some utility score). However, despite this relevance sort, users still regularly sort by a single attribute, e.g., price (See Fig. 2(b)). Then, the user will start from the item with lowest price (left to right), but continue to look at items with higher price, because she desires a shorter distance too. However, for the single-sorted list, the distance attribute is randomly sorted, so she may have to keep looking farther until the price gets too high.

In contrast, using an Imma Sorted list (Fig. 2(a)), the user can see that as price approximately increases, distances approximately decreases. Although neither attribute is perfectly monotonic by the sort order, it is still easy for the user to perceive the increasing and decreasing trends in price and distance, respectively. While scanning from lowest price (left to right), unlike single sort, the user may *conclude her search earlier* by perceiving both distance and price trends. She can stop quickly if she feels that the distance will not decrease further to her desired level as she sees the price increasing.

6.2 Choosing healthy foods: Imma Sort vs. SAW

Consider the user task to find a breakfast cereal food item that is healthy (higher fiber and protein but lower carbohydrates) from a list of 25 items. We discuss two ways to find a satisfying item using Imma Sort and SAW. For a more complex demonstration, we sort with three attributes, and define healthiness as a weighted importance of 0.4, 0.4, and

0.2 on min-max normalized values for fiber, protein, and carbohydrate, respectively. We computed the utility score as this weighted sum, which is also used for SAW. While Imma Sort does not model relative importance for utility, it models the relative preference for monotonicity for interpretability. For comparison, we use the same weights for importance as for monotonicity preference. Since this scenario is more complex, we visualize the items and attribute values in a spreadsheet similar to LineUp [3] (Fig. 11). This demonstrates that while LineUp used SAW for sorting, Imma Sort can be substituted to improve interpretability.

Using a SAW sorted list (Fig. 11(b)), the user will scan from the top few items that are the healthiest based on total utility scores. Suppose she desires higher fiber than the top recommendation (Ground Flaxseed). She will search downward for an alternative item at the cost of lower utility score. However, the three attributes, especially fiber, are in a messy order. Searching downward, she identifies that the 4th, 6th and 7th items have much higher fiber, but much lower protein. The spikes and dives of nutrient levels are confusing, making it hard to compare alternatives. She may have to haphazardly search upward and downward. This lowers the search satisfaction, compared to Imma Sort, as we measured in our user study described later.

In contrast, using an Imma Sorted list (Fig. 11(a)) the user can see approximately that as fiber decreases, protein increases and carbohydrates decreases. These trends are determined by inherent correlations between the attributes in the data instead of the user's utility of each attribute. Although healthier items may not be near the top, each nutrient would be more intuitively sorted and predictable. To enable the user to search from the highest utility item (Ground Flaxseed), the list highlights the best item anchor with yellow color. From this starting location, the user looks upward and sees increasing fiber and carbohydrates, but decreasing protein, and vice versa. She can easily compare adjacent or nearby items to decide whether she is willing to compromise with higher/lower fiber, carbohydrate or protein. She can intuitively see that as fiber incrementally increases, this is slightly traded-off by clear decrease in protein and increase in carbohydrates. Without searching too far, she settles for the item two positions above (Wheat Bran Cereal highlighted in blue-green in Fig. 11(a)). While this item is only ranked 13 by SAW, note that SAW does not fully model the user's preference, since weighted-sum utility is just an estimate, and thus can be somewhat inaccurate. In contrast, Imma Sort can support more flexible and intuitive search, thus improving the sorted list user experience.

7 Evaluation: User Studies

We conducted two quantitative user studies to evaluate user performance in prediction and search tasks using Imma Sort compared to three baseline techniques. We further conducted a qualitative think aloud study to deeply understand how users perceived and used various sort techniques and why they found them helpful or confusing.

7.1 User Study 1: Prediction Task

To quickly search and compare items on a sorted list, users should be able to predict attribute values of items to reduce confusion when finding unexpected values. Our first user study evaluates how well Imma Sort (Imma) supports users to predict the next item after seeing a partially sorted list, compared to baseline sort techniques: SAW, and Single Sort ByA1 or ByA2. We describe the experiment procedure, variables, hypotheses, statistical analysis, and results.

7.1.1 Experiment Apparatus and Procedure

We tasked participants to view items in a sorted list of 12 items and predict attribute values for its last item, i.e., the prediction task. Items were randomly sampled from the Airbnb dataset, and each hotel was represented by a generic hotel icon and the two attributes: distance to downtown and price per night. We used the same icon instead of hotel images to represent all the hotels in order to avoid confounds due to in branding and imagery. The list is laid out horizontally due to the compactness of each item, such that users navigate left and right. To test user prediction, we concealed the distance and price of the last hotel in the list and asked participants to predict their values. Instead of asking for point estimates of each attribute value, we used the "balls and bins" question [53] to more richly measure the participants' uncertainty about their predicted value. This equally divides the possible range of values into 12 bins, and asks participants to assign 100 points across these bins to indicate the likelihood they think the value would be in each bin. The experiment apparatus was implemented in Qualtrics and deployed online.

Participants followed the procedure: 1) After consenting to the study, the participant is randomly assigned to one of four sort techniques. 2) She completes a pre-study questionnaire about her weighted attribute preference, study a tutorial about the experiment task and the sort technique, and take a screening quiz that she needs to pass to continue. 3) Next, she proceeds to the main survey with 5 trials of different lists to predict values for 2 attributes. For each trial, she sees a list of 12 hotels and is asked to predict the distance and price of the last hotel. 4) She concludes with a post-study questionnaire asking about the prediction ease with the sorted list, how she made her prediction in an open-ended text description, and demographic questions.

7.1.2 Experiment Design and Variables

We conducted a 4×5 mixed design study with Technique and attribute Correlation as independent variables (IV). **IV: Technique** (Imma, SAW, ByA1, ByA2) was arranged between-subjects to avoid learning effects and minimize fatigue. **IV: Correlation** (Spearman's correlation coefficient $\rho = -0.7, -0.4, 0, 0.4, 0.7$) was arranged within-subjects to reduce variance due to individuals, and randomly ordered so participants could not anticipate the predictability of lists between trials. We held constant as control variables (CV): number of attributes $d = 2$, list length $n = 12$, and attribute preference weight $w_1 = w_2 = v_1 = v_2 = 0.5$[10]. With two attributes, we had the random variable **RV: Attribute** as A1 or A2. A1 refers to each participant's preferred attribute, and A2 the less preferred one; ByA1 and ByA2 refers to sorting by each attribute, respectively. We selected lists of items

10. This was similar to participant preferences which were slightly imbalanced ($w_{\text{price}} = 0.54$ and $w_{\text{distance}} = 0.46$, see Appendix).

from the simulation experiment with Airbnb data, with three lists per Correlation level (15 lists). We used the same set of items across Techniques for controlled comparison.

For each Attribute, we measured dependent variables (DV) regarding the participant's prediction of the attribute values of the final item. **DV: Prediction Error** measures the inaccuracy of the participant's prediction based on her "balls and bins" response, i.e., $|\bar{x} - x|$, where x is the actual attribute value, $\bar{x} = \sum_{i=1}^{n} p_i b_i$ is the participant's mean estimated value, p_i is the likelihood (number of points out of 100) assigned to the ith bin, and b_i is the middle value of the bin. All values are normalized by the minimum and maximum values in the list. **DV: Prediction Standard Deviation (SD)** measures how uncertain the participant was about her prediction based on her "balls and bins" response, i.e., $sd = \sqrt{\sum_{i=1}^{n} p_i (b_i - \bar{x})^2}$. **DV: Prediction Time** measures time taken to predict both attribute values. We analyzed its logarithmic transform, i.e., Log(Prediction Time), since Prediction Time had a skewed distribution, as is typical. **DV: Prediction Ease** measures perceived ease to use the sorted item to predict the final item as a rating on a 7-point Likert scale (-3=Strongly Disagree, 0=Neither, 3=Strongly Agree).

7.1.3 Hypotheses

We hypothesized that Imma can help human predictions be more accurate (H1), more certain (H2), faster (H3), and easier (H4) than SAW and ByA2. We expected ByA1 to always outperform the other three techniques in terms of all DVs because the primary attribute A1 is perfectly sorted.

7.1.4 Results and Statistical Analysis

We recruited 158 participants from Amazon Mechanical Turk (MTurk) with high qualification (at least 5000 completed HITs with $\geq$ 97% approval rate). Participants were 45.6% female, between 20 and 77 years old (M=39). We excluded 71/229 participants who failed the screening questions. Participants were randomly assigned into one of four Technique conditions: Imma, n=39; SAW, n=37; ByA1, n=34; ByA2, n=48. Participants completed the study in 6 to 57 minutes (Mean=26) and were compensated USD $2.00.

To validate prediction interval width ($|PI|$) as an estimation for perceived predictive interpretability, we calculated the Spearman correlation[11] between them (see Table 1). Participants perceived that attributes with lower $|PI|$ were easier to predict (moderate correlation), and could predict with prediction error (moderate correlation), more certainty (weak correlation with Prediction SD) and somewhat more quickly (weak correlation with Prediction Time).

We fit a multivariate linear mixed effects model on multiple dependent variables with fixed effects Technique and Correlation, interaction effect Correlation $\times$ Technique, and random effects Participant and Trial Sequence nested in Correlation. We performed Tukey HSD tests at $\alpha = .005$ on fixed effects that were notable. Fig. 12 summarizes the results and we report significant effects at $p \leq .005$. *Prediction Error* and *Prediction SD* were both lower for participants using Imma than SAW and ByA2, and similar to ByA1 (both DVs, Tukey HSD: 2 groups). This is consistent with our simulation results regarding prediction interval width $|PI|$ (Fig. 6), although here ByA1 was not better than Imma. *Prediction Time* was similar for Imma and SAW ($p = n.s.$) and not statistically distinguishable from ByA1 or ByA2, but participants were faster with ByA1 than ByA2 ($p = .0006$). *Prediction Ease* was rated positively and highest for ByA1, followed by Imma, but rated equally negatively for SAW and ByA2 (Tukey HSD: 3 groups). Moderately negative correlated ($\rho = -0.4$) lists had lowest ease, but only significantly lower than strongly positive correlated ($\rho = 0.7$) lists (Tukey HSD: 2 groups).

From their written descriptions, participants reported that they perceived strong increasing or decreasing trends when using Imma for lists with strong attribute correlations ($\rho = -0.7, 0.7$). P25 wrote "*For the most part the prices were decreasing so I put a price that was one the low side of the listed prices.*" P71 felt that :"*the distance doesn't fluctuate that much and it is quite steady in the range before going up a bit.*" In contrast, participants struggled to find trends using SAW. For example, P157 "*looked for a trend. I didn't find a trend so I spread out my prediction.*"; P83 randomly guessed "*since all the numbers are all over the place.*" When reading lists of items with weak and no correlations ($-0.4 \leq \rho \leq 0.4$), participants could still perceive a trend using Imma. For example, even for a list with no correlation ($\rho = 0$), P71

11. We used Spearman correlation to handle nonlinear relationships between numeric and ordinal variables.

TABLE 1: Correlation between prediction interval width $|PI|$ and measures of predictive interpretability.

	Prediction Error	Prediction SD	Prediction Time	Prediction Ease
Spearman ρ_{PI}	0.453	0.285	0.121	-0.561
Prob $> |\rho_{PI}|$	$< .0001$	$< .0001$	$< .0001$	$< .0001$

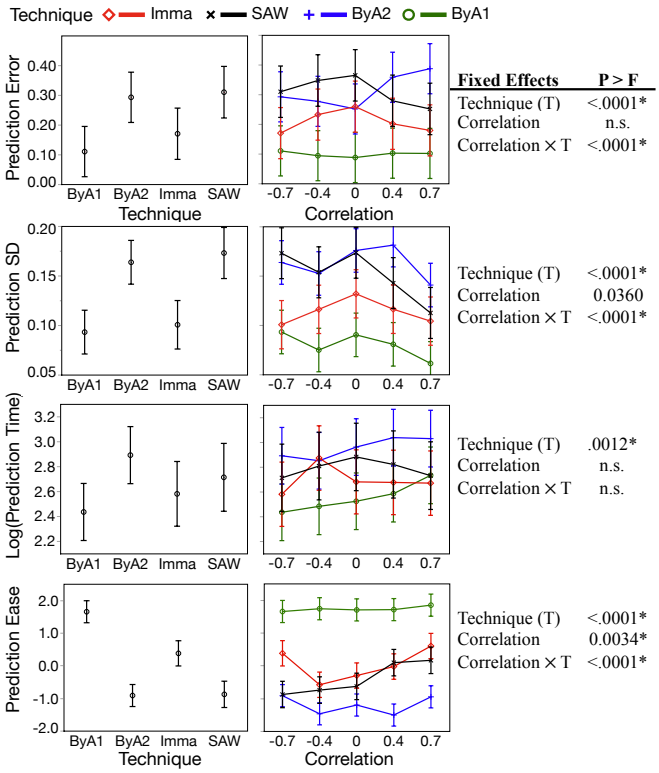


Fig. 12: Results from User Study 2 on sorted list prediction task. Error bars indicate 95% confidence interval. * indicates statistically significant result at $p \leq .005$.

mentioned that "*it seemed in line with the upward trend in the list.*" Conversely, participants using SAW could not report any trend, e.g., P215 wrote "*I picked a middle number because there is no pattern.*" Participants using ByA1 reported that they found a clear trend, unlike those using ByA2.

7.1.5 Summary and Implications

In summary, we found moderate correlations which validate that $|PI|$ can be used to estimate predictive interpretability based on several measures. Participants found Imma Sort more predictable than SAW and ByA2, providing improved prediction ease, confidence, and accuracy. ByA1 was easier to use than Imma Sort, but both had similar predictability. These results are consistent with our hypotheses H1, H2 and H4. Hypothesis H3 was only partially satisfied since predicting with Imma was not faster than with SAW.

7.2 User Study 2: Searching Task

Having validated that Imma Sort improves ease of prediction when browsing the lists, our second user study further evaluated whether the improvement in human predictability could help users make a better selections. We describe the second study's experiment procedure, variables, hypotheses, statistical analysis, and results.

7.2.1 Experiment Apparatus and Procedure

We introduced participants about a scenario to find a low-priced hotel that is close to the downtown of a city. We tasked them to select a hotel from a sorted list of 21 hotels that best satisfies her weighted preference between price and distance, i.e., the search task. We used the same list interface as User Study 1. Each participant followed the same procedure as in User Study 1, except for the main survey, where she performed 5 search task trials. In the post-study questionnaire, she rates her satisfaction and helpfulness of using the sorted list for search. To encourage conscientious search, we incentivised participants with a US$0.50 bonus to be given to the top 10% of participants whose selections were closest to the best item with the highest utility score based on her stated attribute weighted preferences.

7.2.2 Experiment Design and Variables

We conducted a 5×5 mixed design study with the same independent variables (IVs), Technique and attribute Correlation, but with slight additions. **IV: Technique** (Imma, ImmaCenter, SAW, ByA1, ByA2) adds a variant of Imma Sort, ImmaCenter. Imma shows the list starting from the front, but ImmaCenter starts from the best item position. It is more suitable for lists with negative attribute correlation, since the best item is positioned somewhere in the center. ImmaCenter can be more efficient for search, but may be confusing due to its unconventional positioning. The best item has highest utility score based on preference weights. For SAW, it is at the front; for all other sort techniques, it can be anywhere after the front. **IV: Correlation**: (same as User Study 1). We held constant as control variables (CV): number of attributes $d = 2$, list length[12] $n = 21$, and preference weight as ($w_1 = v_1 = 0.46$, $w_2 = v_2 = 0.54$)[13].

For each Attribute, we measured dependent variables (DV) regarding the participant's search performance and opinions. **DV: Closeness to Best Selection** measures the relative nearness between the positions l_s and $\hat{l}$ of participant's selected item and her *personal* best item, respectively. Her personal best item is the item with highest utility score based on her stated preference weights. We calculated Closeness as $1 - |l_s - \hat{l}|/n$, where n is the list length to normalize the metric. **DV: Selection Satisfaction** and **DV: Helpfulness** measure perceived satisfaction with the selected item and helpfulness of the sort order for the search task, respectively; both measured on a 7-point Likert scale.

7.2.3 Hypotheses

We hypothesized that participants would: (H5) select items closer to their personal best with ImmaCenter than Imma and ByA2, but similarly close as SAW because ImmaCenter and SAW initially show the best item in view; (H6) be more satisfied with ImmaCenter than SAW due to its improved intuitiveness; (H7) find Imma and ImmaCenter more helpful than SAW due to clearer trends in attributes; and (H8) report higher satisfaction and helpfulness for lists with positively correlated attributes than negatively correlated ones.

7.2.4 Results and Statistical Analysis

We recruited 279 participants from Amazon Mechanical Turk with the high qualification as User Study 1. Participants were 44.8% female, between 20 and 77 years old (Mean=40). We excluded 132 participants (out of 411) who failed the screening questions at the pre-survey. Participants were randomly assigned into one of five Technique conditions: Imma, n=56; ImmaCenter, n=56; SAW, n=56; ByA1, n=62; ByA2, n=49. Participants completed the study in 4 to 32 minutes (Mean=17) and were compensated USD $1.50.

While we found moderate evidence that prediction interval width $|PI|$ influenced predictive interpretability, its effect on search performance was weaker. We calculated the Spearman correlation between $|PI|$ and dependent variables (see Table 2). We found that $|PI|$ had a weak and negative correlation with Helpfulness but no correlation with Closeness to Best Selection and Satisfaction. This lack of relation could be due to additional, unmeasured factors that drove Satisfaction and imprecise or evolving user preference of attributes than what was elicited. Nevertheless, hypothesis testing on sort techniques found some effects.

We fit a multivariate linear mixed effects model on multiple dependent variables with fixed effects Technique and Correlation, interaction effect Correlation $\times$ Technique, and random effects Participant and Trial Sequence nested in Correlation. We performed Tukey HSD tests at $\alpha = .005$

12. To provide more challenge for the search task, since $n = 12$ of the prediction task would be too easy.

13. Preference weights selected from average preferences measured from participants in User Study 1 (see Appendix). Using average preference is common practice in many real-world ranking applications [10] due to the impracticality of eliciting preference for every new user. Consequently, the average "best" item may not be the personal best for the participant, causing some variability in where the personal highest utility item is placed and necessitating the user to search around.

TABLE 2: Correlation between prediction interval width $|PI|$, and search performance and user experience.

	Closeness to Best Selection	Satisfaction	Helpfulness
Spearman ρ_{PI}	-0.034	-0.040	-0.131
Prob $> |\rho_{PI}|$	n.s.	n.s.	$< .0001$

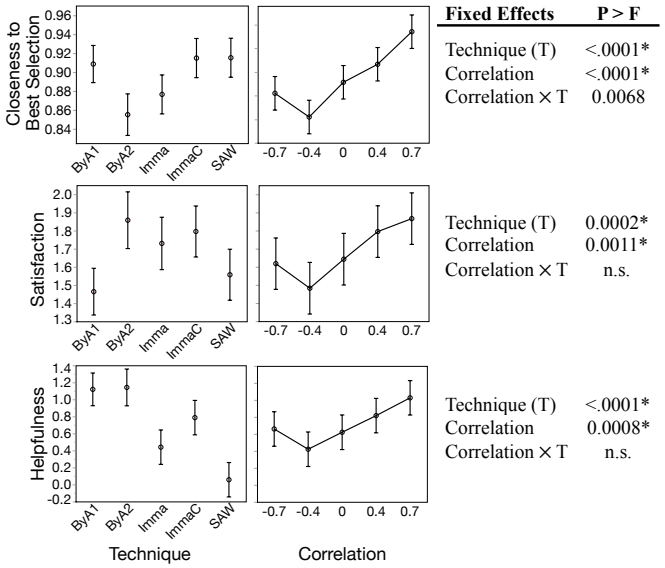


Fig. 13: Results from User Study 2 on sorted list search task. Error bars indicate 95% confidence interval. * indicates statistically significant result at $p \leq .005$.

on fixed effects that were notable. Fig. 13 summarizes the results and we report significant effects at $p \leq .005$. *Closeness to Best Selection* was higher when using ImmaCenter (ImmaC), ByA1 and SAW than using Imma or ByA2 (Tukey HSD: 2 groups). Participants made better selections from the lists with strong positive correlation ($\rho = 0.7$), than lists with moderately positive or no correlations ($\rho = 0.4$ or $\rho = 0$), and made the worst selections using lists with negative correlations (Tukey HSD: 3 groups). *Selection Satisfaction* was higher with ImmaCenter, Imma and ByA2 than SAW and ByA1 (Tukey HSD: 2 groups), and higher for lists with positive attribute correlations than lists with moderate negatively correlation (contrast test: $p < .0001$). *Helpfulness* was highest for ByA1 and ByA2, followed by ImmaCenter and Imma, and SAW was least helpful (Tukey HSD: 3 groups). All sorting was most helpful for lists with strong positive attribute correlation, but least helpful for lists with moderate negative correlation (contrast test, $p < .0001$).

7.2.5 Summary and Implications

In summary, ImmaCenter, ByA1 and SAW helped users make better selections (closer to best item) than ByA2 and Imma. Participants rated ImmaCenter as more helpful than SAW, and were more satisfied in their selections with ImmaCenter than ByA1. As expected, participants performed better on lists with positive attributes correlations than with negative correlations, especially moderate negative correlations. Thus, these results are consistent with our hypotheses H5 to H8. We found that ImmaCenter supports the best search performance and satisfaction and is rated helpful.

7.3 User Study 3: Qualitative Think Aloud Study

Since Imma Sort and its variants (ImmaAnchor, ImmaCenter) are new interactions, we conducted qualitative user studies to understand its user experience and learnability. We investigated whether users navigated quickly from starting positions to the best item, and around the best item.

7.3.1 Procedure and Method

We asked participants to perform the same hotel search task as in the previous user study, each time for two of five different sort techniques: SAW, single sort (SingleSort), single sort with highlighted best item (SingleAnchor), ImmaAnchor, and ImmaCenter. We excluded Imma Sort without anchor to focus on studying the usage of the anchor. We tested on two lists of 21 hotels with strong and moderate negative correlations ($\rho = -0.7$ and -0.4, respectively). We asked participants to think aloud while searching and ended with a structured interview about their experience and rationale.

We recruited 8 participants (4 male, 4 female) between 24 and 30 years old. All were graduate students with varied backgrounds in social science, medicine, geography, computer science, and electronic engineering. All had experience with on-line shopping and were comfortable with list user interfaces. Unlike the between-subjects arrangement in the quantitative study, in the qualitative study, participants used 3-4 sort techniques (within-subjects), so that we could understand their relative opinions. We divided the 8 participants into two groups: 4 participants used and compared SingleSort, SingleAnchor, ImmaAnchor and ImmaCenter, while the others used and compared SAW, ImmaAnchor and ImmaCenter. This enabled us to compare Imma Sort variants with Single Sort and SAW, respectively. We used the same average preference weights for SAW and Imma Sort variants as in the quantitative search task study, thus the participant's personal best item may not be at the front in SAW or be the anchor item in ImmaAnchor/ImmaCenter.

7.3.2 Findings

We organized our findings into key insights that participants: 1) mostly did not trust or prefer SAW, 2) quickly navigated to the best item (anchor) with SingleAnchor or ImmaAnchor then compared around the anchor, 3) were comfortable starting their search from the center of a list at the anchor, and 4) preferred ImmaCenter to ImmaAnchor.

(1) Distrust of SAW. With SAW, although items were sorted by average preference-based utility, participants mostly scanned the list to search for hotels with short distances and low prices, and had difficulty comparing items. P5 *"[could] not tell the logic behind the sorting [SAW]"*, because she did not like the first item, and wanted to compare its price and distance with other items. With SAW, she found it *"difficult as I need to compare a lot of times"* and felt it was necessary to *"re-sort the list again by price"*. Preferring to use ImmaAnchor instead of SAW, P6 could *"easily find the hotels within some distance"*, and could *"easily compare prices of these hotels because price also has some trend"*.

(2) Quick navigation to anchor then compare nearby. Participants used the additional information in SingleAnchor to focus their comparison around the best item anchor. Having navigated to the anchor, P1 *"liked that [the anchor]*

provided a baseline for comparison" and would *"look around the hotels with slightly different price and distance"*. The anchor helped P2 to easily filter many hotels that were *"obviously not better than the highlighted hotel"*. Using ImmaAnchor, participants quickly navigated to the anchor and examined items around it without searching farther down, since they could tell that distance would keep increasing as price decreased. P8 was *"attracted to the highlighted item"* and perceived it as a *"meaningful compromise between distance and price"*. Although P8 *"usually likes to pick the top ranked items"* from online recommendations, after a few trials using ImmaAnchor, he realized *"this sorting convinced me to choose [the anchor] or similar ones around it, because now I know why it is better than others"*. P3 felt the clear trends in ImmaAnchor helped him to *"quickly locate candidates and compare between them"*. P4 preferred ImmaAnchor to SingleAnchor, because both prices and distances were closely comparable with ImmaAnchor, while only price was easily comparable with SingleAchor and she had to *"jump around to compare"* distance.

(3) Focused navigation around anchor when starting from center. With ImmaAchor, the anchor item was at a middle position (around positions 8-13 in a list of 21 items) which was not always in visible range (9-item span) from the front (left) position, thus participants would have had to navigate far from the front to notice it. Using ImmaCenter, starting from the center, participants would quickly scan left to the front, return to the center anchor, and quickly scan to the near right (not to the end). This suggests some limited trust in starting from the center, but participants were mostly verifying the trend between the front item and anchor and did so quickly. Eventually, participants felt little need to examine items far from the anchor: P5 *"realized that there was a trend, so I think the left most and right most are not what I liked, so I just focus on the hotels in the middle of the list"* and P2 could *"tell that to the left [of the anchor], the price is increasing, [and] to the right, the distance is also increasing, so no need to see the left-end or right-end"*. Thus, ImmaCenter helped participants to skip searching items near the front and they appreciated starting near the anchor in the center.

(4) Preference to start searching from the middle anchor. Participants generally preferred ImmaCenter over ImmaAnchor. Only P7 felt that not starting from the front item *"looks weird"*. P1, P2, P4, and P6 felt that seeing a *"baseline [anchor] in the beginning"* could help them to *"quickly filter the hotels"* that were obviously not as good as the anchor to *"save time"*. Furthermore, P2 and P3 did not like the anchor *"to be placed far at the back of the list"* (as with ImmaAnchor), out of visible range. Indicating his preference, P8 mentioned that *"after a few trials [of using ImmaAnchor], I found that the hotels I liked are always around the highlighted one, so I jumped to [it] directly"* and that ImmaCenter *"saves this extra jump and helps me to see the highlighted hotel directly."* Thus, participants felt comfortable to start searching from the anchor placed in the center of the list to jumpstart their item comparisons.

In summary, participants appreciated the clear trends in Imma Sort to allow for close comparisons around the best item anchor. This makes Imma Sort more predictively interpretable than SAW that does not show clear trends. Participants became accustomed to using ImmaCenter to start from the center after realizing that searching among front items was less useful than searching around the anchor.

8 Discussion and Design Implications

We have demonstrated the usage, usability, and usefulness of Imma Sort for sorting multiple attributes with mock up use cases, simulation experiments and user studies to improve predictive interpretability, search satisfaction and search performance. To promote its adoption, we discuss why, when, where, and how to deploy Imma Sort for real-world applications, and the limitations.

Why to use. Despite the intuitiveness of score-based recommendation ranked lists and attribute-weighted ranking (e.g., Simple Additive Weighting), users still often re-sort the pre-ranked items to perform careful comparisons to search for their personal best item. This demonstrates a lack of intuitiveness in utility-based ranking. For example, SAW appears simple because its algorithm to sort is simple, but its sorted list result is typically not simple. Our qualitative study found that users may not trust such sorted results because of the unclear relationship between attributes and the ranking. In contrast, Imma Sort prioritizes presenting items in a simple sort result which users experience, albeit with a more complex sorting technique which is transparent (hidden) to users, and which can be provided to programmers via an application programming interface (API). Our quantitative user studies showed that Imma Sort had better usability (Satisfaction and Helpfulness) than SAW. Our qualitative user study showed that participants appreciated the clear trends due to the predictive interpretability provided by Imma Sort. Therefore, compared to SAW, Imma Sort improves the user experience (UX) for searching a sorted list without compromising search performance.

When to use. As expected, Imma Sort improved the users' predictive interpretability on multiple attributes and helps to improve user search satisfaction, which is useful for cases, such as helping e-commerce consumers to find products from a list of alternatives. Imma Sort will be useful 1) to enhance one-attribute sorting to multiple attributes, for lightweight comparisons (vs. intensive spreadsheet analyses), 2) for items which could have conflicting attributes, and 3) when users have personalized preferences regarding the importance or interpretability of different attributes.

Where to use. Imma Sort can be integrated into many real-world applications, such as recommendation systems and search engines that deal with multi-attribute items. It can help by rearranging the recommended items or search results to be more intuitive and predictively interpretable to lower the cognitive load of multi-attribute decision making. Imma Sort can be especially useful for mobile apps because it takes less space by showing only one list, compared to large spreadsheets [3], filtering menus [1], or other interactive graphical user interface (GUI) elements [21].

How to use. Imma Sort can be added to existing sort user interfaces, such as recommendation list and search results, by adding a new sort mode control interface and displaying items in the original list layout. For correlated attributes, it can be used similarly to one-attribute sorted lists. For conflicting attributes, we recommend highlighting the best item anchor (ImmaAnchor) to encourage users to quickly navigate to the anchor to compare. Once users get more experience, ImmaCenter can be used to "jumpstart" users to the best item anchor and avoid the initial navigation

from front to best item. ImmaCenter may be more intuitively laid out in a horizontal list with small item cards, which is presently popular in mobile and web interfaces (e.g., Netflix video browsing and Google local search). The horizontal list supports unbiased left-right navigation from the center, unlike vertical lists that drive navigation downwards.

Limitations. Imma Sort provides smoother trends for lists with conflicting attributes ($\rho < 0$), but not significantly for lists with uncorrelated attributes ($\rho \approx 0$). Nevertheless, we found from our analysis of real-world datasets that most lists have negative attribute correlations (see Fig. 9). Imma Sort requires attribute preferences weights to be specified. This can be elicited from users, or a default choice can be provided from average preferences as we have done in the search task user study. Nonetheless, specifying attribute weights is a common requirement for multi-attribute sort, such as SAW. Imma Sort implementation is somewhat sophisticated, unlike single sort or SAW, but not necessarily more than recommendation models based on machine learning. These are typically implemented in back-end servers, and transparent to front-end developers. We open source[14] the Imma Sort code as a simple API library to help promote its adoption. Hence, programming to use Imma Sort is not necessarily more complex than using SAW-like utility ranked recommender systems.

9 Conclusion and Future Work

We have introduced the notion of predictive interpretability to identify intuitiveness issues in sorted list results. With this insight, we identified requirements for multi-attribute sorting. We presented Imma Sort to sort items by multiple attributes simultaneously by trading-off the monotonicity in the primary sorted attribute to increase the approximate monotonicity for other attributes. We extended the Ranking Principle Curve model to tune approximate monotonic trends for multiple attributes and prioritize smoother, smaller fluctuations for specific attributes based on preference weights. We demonstrated the usefulness and usability of Imma Sort with application use cases, and a qualitative think aloud study. We validated its improvement in overall prediction interval in simulation experiments with synthetic and real-world data, and its benefit to human predictiveness and search performance in two user studies.

Our work is the first to rigorously study the usability of multi-attribute sort and identified the importance to support predictive interpretability. While Imma Sort presents a novel capability to sort items for interpretability, it raises several potential challenges, such as perceiving some uncertainty in the sort order and not having the estimated best item at the front of the list. While we found early evidence for the user acceptance of Imma Sort despite these concerns, future work could investigate the usage of Imma Sort in the wild and in longitudinal studies with deployed mobile and web apps. Having proposed a nascent, novel interaction, we expect future research to provide further refinements to Imma Sort to ameliorate these concerns [54], and drive more commonplace and more interpretable use of multi-attribute sorting.

14. https://github.com/nus-ubicomplab/imma-sort

10 Acknowledgments

We thank Roland Yap and Shengdong Zhao for helpful discussions. This work was carried out and supported by the NUS Institute for Health Innovation and Technology (iHealthtech) and NUS ODPRT Grant.

References

[1] D. Tunkelang, "Faceted search," *Synthesis lectures on information concepts, retrieval, and services*, vol. 1, no. 1, pp. 1–80, 2009.
[2] G.-H. Tzeng and J.-J. Huang, *Multiple attribute decision making: methods and applications*. Chapman and Hall/CRC, 2011.
[3] S. Gratzl, A. Lex, N. Gehlenborg, H. Pfister, and M. Streit, "Lineup: Visual analysis of multi-attribute rankings," *IEEE transactions on visualization and computer graphics*, vol. 19, no. 12, pp. 2277–2286, 2013.
[4] S. Pajer, M. Streit, T. Torsney-Weir, F. Spechtenhauser, T. Möller, and H. Piringer, "Weightlifter: Visual weight space exploration for multi-criteria decision making," *IEEE transactions on visualization and computer graphics*, vol. 23, no. 1, pp. 611–620, 2016.
[5] E. Wall, S. Das, R. Chawla, B. Kalidindi, E. T. Brown, and A. Endert, "Podium: Ranking data using mixed-initiative visual analytics," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 1, pp. 288–297, 2017.
[6] H. Li, "A short introduction to learning to rank," *IEICE TRANSACTIONS on Information and Systems*, vol. 94, no. 10, pp. 1854–1862, 2011.
[7] J. Singh and A. Anand, "Interpreting search result rankings through intent modeling," *arXiv preprint arXiv:1809.05190*, 2018.
[8] C.-G. Li, X. Mei, and B.-G. Hu, "Unsupervised ranking of multi-attribute objects based on principal curves," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 12, pp. 3404–3416, 2015.
[9] Y. Hou, N. Yang, Y. Wu, and S. Y. Philip, "Explainable recommendation with fusion of aspect information," *World Wide Web*, vol. 22, no. 1, pp. 221–240, 2019.
[10] C. Kuhlman, D. Doherty, M. Nurbekova, G. Deva, Z. Phyo, P.-H. Schoenhagen, M. VanValkenburg, E. Rundensteiner, and L. Harrison, "Evaluating preference collection methods for interactive ranking analytics," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, 2019, p. 512.
[11] J. O. Talton, III, K. Dusad, K. Koiliaris, and R. S. Kumar, "How do people sort by ratings?" in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, ser. CHI '19. ACM, 2019, pp. 305:1–305:10.
[12] K. R. MacCrimmon, "Decisionmaking among multiple-attribute alternatives: a survey and consolidated approach," RAND CORP SANTA MONICA CA, Tech. Rep., 1968.
[13] S. H. Zanakis, A. Solomon, N. Wishart, and S. Dublish, "Multi-attribute decision making: A simulation comparison of select methods," *European journal of operational research*, vol. 107, no. 3, pp. 507–529, 1998.
[14] J. S. Dyer, *Maut — Multiattribute Utility Theory*. New York, NY: Springer New York, 2005, pp. 265–292.
[15] T. L. Saaty, "Analytic hierarchy process," *Encyclopedia of Biostatistics*, vol. 1, 2005.
[16] C.-L. Hwang, Y.-J. Lai, and T.-Y. Liu, "A new approach for multiple objective decision making," *Computers & operations research*, vol. 20, no. 8, pp. 889–899, 1993.
[17] Y. Lyu, C.-Y. Chow, R. Wang, and V. C. Lee, "imcrec: A multi-criteria framework for personalized point-of-interest recommendations," *Information Sciences*, vol. 483, pp. 294–312, 2019.
[18] J. Vincke and P. Brans, "A preference ranking organization method. the promethee method for mcdm," *Management Science*, vol. 31, no. 6, pp. 647–656, 1985.
[19] J. Figueira, V. Mousseau, and B. Roy, "Electre methods," in *Multiple criteria decision analysis: State of the art surveys*. Springer, 2005, pp. 133–153.
[20] C. di Sciascio, V. Sabol, and E. E. Veas, "Rank as you go: User-driven exploration of search results," in *Proceedings of the 21st International Conference on Intelligent User Interfaces*. ACM, 2016, pp. 118–129.
[21] K. Klouche, T. Ruotsalo, L. Micallef, S. Andolina, and G. Jacucci, "Visual re-ranking for multi-aspect information retrieval," in *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*. ACM, 2017, pp. 57–66.

[22] I. Hur and J. S. Yi, "Simulsort: multivariate data exploration through an enhanced sorting technique," in *International Conference on Human-Computer Interaction*. Springer, 2009, pp. 684–693.

[23] C. Perin, R. Vuillemot, and J.-D. Fekete, "A table!: Improving temporal navigation in soccer ranking tables," in *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*. ACM, 2014, pp. 887–896.

[24] G. Carenini and J. Loyd, "Valuecharts: analyzing linear models expressing preferences and evaluations," in *Proceedings of the working conference on Advanced visual interfaces*. ACM, 2004, pp. 150–157.

[25] K. Yang, J. Stoyanovich, A. Asudeh, B. Howe, H. Jagadish, and G. Miklau, "A nutritional label for rankings," in *Proceedings of the 2018 International Conference on Management of Data*. ACM, 2018, pp. 1773–1776.

[26] C. Shi, W. Cui, S. Liu, P. Xu, W. Chen, and H. Qu, "Rankexplorer: Visualization of ranking changes in large time series data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2669–2678, 2012.

[27] P. Kidwell, G. Lebanon, and W. Cleveland, "Visualizing incomplete and partially ranked data," *IEEE Transactions on visualization and computer graphics*, vol. 14, no. 6, pp. 1356–1363, 2008.

[28] J. P. Peter, J. C. Olson, and K. G. Grunert, *Consumer behaviour and marketing strategy*. McGraw-Hill London, 1999.

[29] T. Joachims, "Optimizing search engines using clickthrough data," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2002, pp. 133–142.

[30] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. N. Hullender, "Learning to rank using gradient descent," in *Proceedings of the 22nd International Conference on Machine learning (ICML-05)*, 2005, pp. 89–96.

[31] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences," *Journal of machine learning research*, vol. 4, no. Nov, pp. 933–969, 2003.

[32] K. Crammer and Y. Singer, "Pranking with ranking," in *Advances in neural information processing systems*, 2002, pp. 641–647.

[33] F. Xia, T.-Y. Liu, J. Wang, W. Zhang, and H. Li, "Listwise approach to learning to rank: theory and algorithm," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1192–1199.

[34] 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*. ACM, 2001, pp. 613–622.

[35] S. Lin, "Rank aggregation methods," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 5, pp. 555–570, 2010.

[36] A. Tavanaei, R. Gottumukkalay, A. S. Maida, and V. V. Raghavan, "Unsupervised learning to rank aggregation using parameterized function optimization," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.

[37] J. Bedő and C. S. Ong, "Multivariate spearman's ρ for aggregating ranks using copulas," *Journal of Machine Learning Research*, vol. 17, no. 201, pp. 1–30, 2016.

[38] R. Fagin, R. Kumar, M. Mahdian, D. Sivakumar, and E. Vee, "Comparing and aggregating rankings with ties," in *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2004, pp. 47–58.

[39] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 2016, pp. 1135–1144.

[40] J. Singh and A. Anand, "Posthoc interpretability of learning to rank models using secondary training data," *arXiv preprint arXiv:1806.11330*, 2018.

[41] M. Verma and D. Ganguly, "Lirme: Locally interpretable ranking model explanation," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2019, pp. 1281–1284.

[42] J. L. Herlocker, J. A. Konstan, and J. Riedl, "Explaining collaborative filtering recommendations," in *Proceedings of the 2000 ACM conference on Computer supported cooperative work*. ACM, 2000, pp. 241–250.

[43] B. Abdollahi and O. Nasraoui, "Using explainability for constrained matrix factorization," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017, pp. 79–83.

[44] Y. Zhang, "Incorporating phrase-level sentiment analysis on textual reviews for personalized recommendation," in *Proceedings of the eighth ACM international conference on web search and data mining*. ACM, 2015, pp. 435–440.

[45] X. Chen, H. Chen, H. Xu, Y. Zhang, Y. Cao, Z. Qin, and H. Zha, "Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2019, pp. 765–774.

[46] P. Pu and L. Chen, "Trust building with explanation interfaces," in *Proceedings of the 11th international conference on Intelligent user interfaces*. ACM, 2006, pp. 93–100.

[47] J. R. Bettman, "Memory factors in consumer choice: A review," *Journal of Marketing*, vol. 43, no. 2, pp. 37–53, 1979.

[48] S. Shiloh, S. Koren, and D. Zakay, "Individual differences in compensatory decision-making style and need for closure as correlates of subjective decision complexity and difficulty," *Personality and individual differences*, vol. 30, no. 4, pp. 699–710, 2001.

[49] T. Hastie and W. Stuetzle, "Principal curves," *Journal of the American Statistical Association*, vol. 84, no. 406, pp. 502–516, 1989.

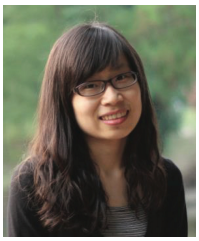
[50] I. Jolliffe, *Principal component analysis*. Springer, 2011.

[51] W. Q. Meeker, G. J. Hahn, and L. A. Escobar, *Statistical intervals: A guide for practitioners and researchers*. John Wiley & Sons, 2017, vol. 541.

[52] T. D. Gauthier, "Detecting trends using spearman's rank correlation coefficient," *Environmental forensics*, vol. 2, no. 4, pp. 359–362, 2001.

[53] A. Delavande and S. Rohwedder, "Eliciting subjective probabilities in internet surveys," *Public Opinion Quarterly*, vol. 72, no. 5, pp. 866–891, 2008.

[54] S. Greenberg and B. Buxton, "Usability evaluation considered harmful (some of the time)," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, 2008, pp. 111–120.



Yan Lyu received the Ph.D. degree in computer science from City University of Hong Kong, Hong Kong, in 2016, and the M.S. degree in pattern recognition and intelligent systems from University of Science and Technology of China, China, in 2013. She was a Postdoctoral Research Fellow with Hong Kong Baptist University, Hong Kong, in 2017, and with National University of Singapore, Singapore, from 2017 to 2020. She is an Associate Professor in the School of Computer Science and Engineering at Southeast University, China. Her research interests include data analytics and visualization, intelligent transportation systems, and location-based services.

Fan Gao is a undergraduate student in Information Engineering at South China University of Technology, Guangzhou, China. Her research interests include data visualization, interactive learning and design, and conversational agents.

I-Shen Wu received the M.Comp. degree in computer science from National University of Singapore (NUS) in 2019 and the B.B.A. degree in information management from National Taiwan University (NTU) in 2016. She is a full-stack software developer working on eCommerce platforms. Her research interests include data visualization and applications for smart healthcare.

Brian Y. Lim received the B.S. degree in engineering physics from Cornell University, Ithaca, New York, U.S.A., in 2006, and the Ph.D. degree in human-computer interaction from Carnegie Mellon University, Pittsburgh, U.S.A., in 2012. He is an Assistant Professor in the Department of Computer Science at the National University of Singapore (NUS). His research interests include ubiquitous computing, explainable artificial intelligence, interfaces, and applications for urban data analytics and smart healthcare.

APPENDIX A

A.1 Principal Curve Algorithm

Algorithm 1 Principal Curve Algorithm for I'mma Sort

Input: A set of data points X, monotonicity index $\boldsymbol{\alpha}$ and weight $\boldsymbol{w}$
Output: Four control points P, and a sorting of X

1: Normalize each $\boldsymbol{x_i} \in [0,1]^d$, $1 \leq i \leq n$
2: Initialize $P^{(0)}$: $\boldsymbol{p_0^{(0)}} = \frac{1}{2}(1-\boldsymbol{\alpha})$, $\boldsymbol{p_3^{(0)}} = \frac{1}{2}(1+\boldsymbol{\alpha})$
3: Randomly generate $\boldsymbol{p_1^{(0)}}, \boldsymbol{p_2^{(0)}} \in [0,1]^d$
4: Initialize $\Delta = \infty$, $t = 1$
5: **while** $\Delta > \delta$ **do**
6: $\quad s_i^{(t-1)} = \underset{0 \leq s \leq 1}{\operatorname{argmin}} \sqrt{\sum_{a=1}^{d} w_a (x_{i,a} - \boldsymbol{f}(s, P^{(t-1)})_a)^2}$
7: $\quad P^{(t)} = \underset{(\boldsymbol{p_1}, \boldsymbol{p_2})}{\operatorname{argmin}} \sum_{i=1}^{n} \sum_{a=1}^{d} w_a (x_{i,a} - \boldsymbol{f}(s_i^{(t-1)}, P)_a)^2$.
8: $\quad \varepsilon^{(t)} = \sum_{i=1}^{n} ||\boldsymbol{x_i} - \boldsymbol{f}(s_i^{(t-1)}, P^{(t)})||^2$
9: $\quad \Delta = |\varepsilon^{(t)} - \varepsilon^{(t-1)}|$
10: $\quad t = t + 1$
11: **end while**
12: $P = P^{(t)}$, $s_i = \operatorname{argmin}_{0 \leq s \leq 1} \sum_{i=1}^{n} ||\boldsymbol{x_i} - \boldsymbol{f}(s, P)||$
13: Sort X based on their projection positions s_i, $1 \leq i \leq n$

Algorithm 1 depicts the principal curve algorithm [6][49] to find the best Bézier curve $\boldsymbol{f}(s, P)$ for data points $X = \{\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_n}\}$ with control points $P = \{\boldsymbol{p_0}, \boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{p_3}\}$ and position index s. The algorithm considers the monotonicity index $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_d)^T$ to specify increasing ($\alpha_a = 1$) or decreasing ($\alpha_a = -1$) trends, and preference weights $\boldsymbol{w} = (w_1, w_2, ..., w_d)^T$ on d attributes as input to allow users to indicate smoothness preference for each attribute $a \in [1, d]$. It firstly normalizes attribute values of each data point into the range $[0, 1]$, and initializes the four control points with $\boldsymbol{\alpha}$ and randomization: $\boldsymbol{p_0} = (1 - \boldsymbol{\alpha})/2$ and $\boldsymbol{p_3} = (1 + \boldsymbol{\alpha})/2$ are determined by $\boldsymbol{\alpha}$ to constrain the increasing or decreasing trend of each attribute; $\boldsymbol{p_1}$ and $\boldsymbol{p_2}$ are randomly selected from $[0, 1]^d$ (Algorithm 1 Lines 2-3).

The algorithm iteratively adjusts control points $\boldsymbol{p_1}$ and $\boldsymbol{p_2}$ to fit the curve that minimizes the total distance of all data points to their projections on the curve (Eq. 6), i.e.,

$$\varepsilon = \sum_{i=1}^{n} \sum_{a=1}^{d} w_a (x_{i,a} - \boldsymbol{f}(s_i, P)_a)^2,$$

where $x_{i,a}$ is the ath attribute of the ith data point, and $\boldsymbol{f}(s_i, P)_a$ denotes the ath dimension of Bézier curve and refers to the projection of the ath attribute. For each iteration t, two steps are performed:

Step 1. For each data point $\boldsymbol{x_i}$, calculate projection position s_i on the curve from the last iteration (Eq. 7, Algorithm 1 Line 6), i.e.,

$$s_i^{(t-1)} = \underset{0 \leq s \leq 1}{\operatorname{argmin}} \sqrt{\sum_{a=1}^{d} w_a (x_{i,a} - \boldsymbol{f}(s, P^{(t-1)})_a)^2}.$$

Step 2. Adjust the control points $\boldsymbol{p_1}$ and $\boldsymbol{p_2}$ by minimizing the sum of distance between each data point $\boldsymbol{x_i}$ and its the projection $\boldsymbol{f}(s_i^{(t-1)}, P)$ (Algorithm 1 Line 7). , i.e.,

$$P^{(t)} = \underset{(\boldsymbol{p_1^{(t)}}, \boldsymbol{p_2^{(t)}})}{\operatorname{argmin}} \sum_{i=1}^{n} \sum_{a=1}^{d} w_a (x_{i,a} - \boldsymbol{f}(s_i^{(t-1)}, P)_a)^2.$$

In each iteration, $P^{(t)}$ should satisfy the monotonicity constraints, i.e., Eqs. 4 and 5. The algorithm terminates when the distance difference between iterations $\Delta = |\varepsilon^{(t)} - \varepsilon^{(t-1)}|$ is less than a threshold δ (Algorithm 1 Line 9). The final sum of distances $\varepsilon^{(t)}$ is a local minimum. After projecting data X to the final Bézier curve, the sorting arrangement of X can be determined based on the sequence of each of their projections s_i on the final curve.

A.2 Distribution of $|PI|$ for Imma Sort

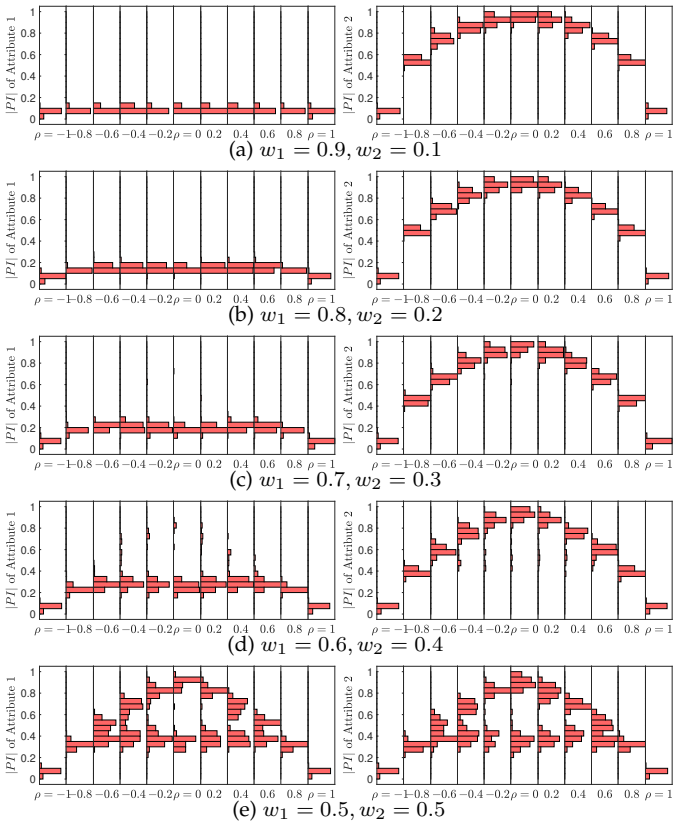


Fig. A1: Distribution of attribute $|PI|$ for different correlations ρ and different preference weights for Imma Sort. As the preference weight for Attribute 1 decreased, the $|PI|$ of Attribute 1 increased while that of Attribute 2 decreased. $|PI|$ becomes binomial when $w_1 = w_2 = 0.5$ and $-0.6 \leq \rho \leq 0.6$.

A.3 Distribution of preference weights

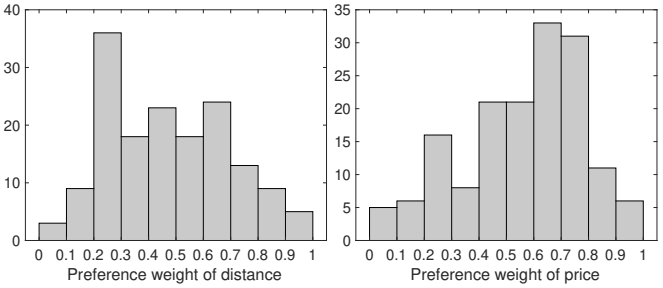


Fig. A2: Distributions of user preference weights collected from User Study 1 for distance weight (Mean=0.54, SD=0.21) and price weight (Mean=0.46, SD=0.21).