Cross-Validation in Row-conditional Nonmetric Unfolding: A Pilot Study

Author: Baobao Yuan

First Supervisor: Prof. dr. Willem J. Heiser
Mathematical Institute
Faculty of Science
Leiden University

Second Supervisor: Dr. Frank M.T.A. Busing
Methodology and Statistics Unit
Institute of Psychology
Leiden University

28102015
# Contents

1 Introduction 1
   1.1 Unfolding 2
      1.1.1 Nonmetric Multidimensional Unfolding Method 4
      1.1.2 Degeneracy Problem 6
      1.1.3 PREFSCAL 7
      1.1.4 PREFSCAL-II 8
   1.2 Cross-Validation 9
      1.2.1 Cross-Validation idea in Unfolding 10
      1.2.2 Why we use Cross-Validation in Unfolding? 11

2 Research Questions 12

3 Method 12
   3.1 Missing Data 12
   3.2 Data Generation 12
   3.3 Prediction in Unfolding 13
   3.4 Simulation Study Design 15
   3.5 Description of the Factors 16
   3.6 Outcome Measures 17
      3.6.1 Prediction Error Measures 17
      3.6.2 Odds Ratio and 95% Confidence Intervals 17
      3.6.3 Measures of Fit 18

4 Results 20
   4.1 Main results 20
   4.2 Other results 25
   4.3 Summary 29

5 Discussion 31
   5.1 Perfect Data 31

6 Conclusion 33

Appendices 37

Appendix A Notations 37

Appendix B R code for Functions 38

Appendix C Tables 43
Abstract

The unfolding method analyses preference data by constructing a joint spatial configuration of individuals and stimuli. In this thesis, we focus on row-conditional nonmetric unfolding. Nonmetric multidimensional unfolding is known as suffering from degeneracy problems. The PREFSCAL method proposed by Busing, Groenen and Heiser can successfully avoid the degenerate solution by penalizing on the coefficient of variation. But it contains two penalty parameters and it is not easy to consider them at the same time. Based on this idea, a new version of PREFSCAL was upgraded by Busing and Heiser. The new loss function has an additive penalty based on the coefficient of variation and uses only one penalty parameter $\lambda$. The new version of PREFSCAL becomes more user friendly. Cross-Validation is a method that is most widely used for selecting models by estimating prediction error. We want to use cross-validation to choose the unknown parameter tuning $\lambda$ and to estimate the prediction error of the final model. So we set up a simulation study as a pilot study of cross-validation for a range of values of the penalty parameter $\lambda$ and test a series of design factors. The results of the simulation study are discussed and suggestions to implement cross-validation are provided for future research.
1 Introduction

The general idea of scaling made by the unfolding method had a great influence on the psychological and social science in the second half of the twentieth century (Busing, 2010). Unfolding attempts to create a low-dimensional configuration of points for both individuals and stimuli based on stimulus preferences given by each individual and keeps the order of the distances reflecting the order of the rankings. It can be widely used in marketing decision and research (Busing & De Rooij, 2009) and voting theory (Heiser, 1981). There are two ways to classify unfolding models. The first way is to take dimensionality as the criterion to categorize it into unidimensional and multidimensional unfolding. The other way is to consider measurement level as the criterion to categorize it into metric and nonmetric unfolding (Cox & Cox, 2000). In this paper, we restrict ourselves to row-conditional multidimensional nonmetric unfolding, which is the most common case.

Cross-Validation is a method that is most widely used for selecting models by estimating prediction error (Hastie, Tibshirani, Friedman, & Franklin, 2005). However, cross-validation was never used before in nonmetric unfolding. We want to use cross-validation to determine the unknown parameters tuning and to estimate the prediction error of the final model. There are several unknown tuning parameters in unfolding we can focus on; such as the dimensionality, the transformations, or the penalty parameter $\lambda$. In our study, we focus on the penalty parameter $\lambda$.

In the following, we will first present unfolding models, then briefly discuss the degeneracy problem and discuss how this problem is currently solved. Then a new version of PREFSCAL (Busing and Heiser, 2015), the steps of cross-validation, and four research questions are presented. Finally, a full factorial design will be used in a simulation study that aims to explore in different circumstances whether we could find an optimal penalty parameter $\lambda$ value by using two prediction methods.
1.1 Unfolding

Guttman (1950) showed for the first time how both subjects and items can be scaled on the same continuum when we have only qualitative data and when no distributional assumptions are considered in his famous Guttman Scale. Building on the work of Guttman and Thurstone (1927), Coombs first presented the unidimensional nonmetric unfolding model. Bennett and Hays (1960) and Hays and Bennett (1961) extended Coombs’ unidimensional unfolding model to multiple dimensions.

Coombs (1950, 1964) introduced the $J$ scale and $I$ scale, where the $J$ scale is a line on which $n$ individuals and $m$ stimuli are placed together and the $I$ scale is each individual’s preference ordering. Figure 1 is a $J$ scale example for four stimuli from Cox and Cox (2000). There are 6 midpoints between all pairs of stimuli in total. These 6 midpoints act as 6 boundaries, splitting the $J$ scale into 7 intervals. Every interval corresponds to a particular preference ordering, as shown in Table 1.

$$
\begin{array}{cccccccc}
\hline
\text{Interval} & I_1 & I_2 & I_3 & I_4 & I_5 & I_6 & I_7 \\
\text{Ordering} & ABCD & BACD & BCAD & CBAD & CBDA & CDBA & DCBA \\
\hline
\end{array}
$$

It is noteworthy that not all $I$ scales can be accommodated. For instance, the preference ordering $ADBC$ is not available in this example. It is more challenging to generate the $J$ scale from a set of $I$ scales than to generate all possible $I$ scales from a given $J$ scale.
The geometric structure becomes more complicated in the multidimensional case. Figure 2 depicts a two dimensional example from Cox and Cox (2000). \( H(A, B) \) is a boundary hyperplane between \( A \) and \( B \) consisting of all points that are equally far from the two stimulus points. On one side of the boundary hyperplane is a region that contains all of the points closer to stimulus \( A \) than stimulus \( B \). On the other side of the boundary hyperplane is a region that contains all of the points closer to stimulus \( B \) than stimulus \( A \). The two dimensional space of four stimuli is divided by boundary hyperplanes into mutually exclusive and exhaustive regions, called isotonic regions. Thus, the intersection of these half spaces produces the isotonic regions (Young, 2013). A particular isotonic region has a particular preference ordering. The preferred order in shaded isotonic region is \( DCAB \). Again, the model predicts only a limited number of orders out of all \( 4! = 24 \) orders that can be formed of 4 stimuli.
The multidimensional unfolding objective is to find a common dimension called the $J$ scale, based on a set of $I$ scales. Distances from an individual’s point in the unfolding space to the stimulus have to match the order of the original preferences.

### 1.1.1 Nonmetric Multidimensional Unfolding Method

The nonmetric multidimensional unfolding model is one of the most common models used in unfolding methods. Nonmetric data is transformed into intermediate ratio data, while maintaining the original order of the data. The intermediate ratio data is then used to construct a metric representation. The unfolding method considers both individuals and stimuli as points in Euclidean space, where the distance between them corresponds to intermediate ratio data. The points for the individuals are usually called ideal points. Small distances indicate the most preferred stimuli and vice versa.

Let $x_{ik}$ represent the coordinates of the individuals and $y_{jk}$ represent the coordinates of the stimuli, with $i = 1, ..., n$, $j = 1, ..., m$ and $k = 1, ..., p$, where $p$ equals the pre-chosen dimensionality of the solution. The ideal points and stimulus points are collected in the $p$-vector $x_i$ and the $p$-vector $y_j$, respectively. The $x_i$'s are collected in the matrix $X$ (size $n \times p$) and the $y_j$'s are collected in the matrix $Y$ (size $m \times p$). The dissimilarity $\delta_{ij}$ indicates the preference between $i$th individual and $j$th stimulus. The nonmetric data (rankings) are converted into metric information through some monotonically nondecreasing transformation function $f(\cdot)$. In the row-conditional case, the observed preferences are replaced by a separate monotonically nondecreasing transformation function for each individual. The reproduced intermediate ratio data are called pseudo-distances and are defined as $\gamma_{ij} = f(\delta_{ij})$. There are various transformation functions $f(\cdot)$. The most commonly used transformations are linear transformations with and without intercept, an arbitrary monotone transformation and a monotone spline transformation.

The Euclidean distance between $x_i$ and $y_j$ is defined as

$$d_{ij}(x_i, y_j) = \sqrt{(x_i - y_j)^\top (x_i - y_j)}.$$

The unfolding problem is to find the pseudo-distances $\gamma_{ij}$ which should match the Euclidean distances $d_{ij}$ as closely as possible.
We now use the dataset coming from Dr. Delbeke and analyze it with PREFSCAL (available in IBM SPSS statistics). In a study of family composition, 82 students of Leuven University ranked 16 combinations of boys (0-3) and girls (0-3) in terms of preference (1-16), where 1 is the most preferred combination and 16 the least preferred family composition.

Figure 3: Unfolding of preferences for family composition in two dimensions

In Figure 3, black dots represent family compositions and blue circles represent students. The closer the distance between a black dot and a blue circle, the more preferred the family composition was by the student. For example, the most preferred family composition of student 15 is $B1G3$ (one boy and three girls), while this student ranked $B2G3$ (two boys and three girls) as his second preference.
1.1.2 Degeneracy Problem

The badness-of-fit between the pseudo-distances and the distances is usually measured by the ratio of a least squares function and some normalization function. The degeneracy problem occurs in almost all circumstances since optimizing the least squares loss function leads to a “perfect” solution which shows perfect badness-of-fit but is not useful for interpretation. The fundamental cause of degeneracy is that optimizing the badness-of-fit measures allows for transformations of the data that equalize all distances to a single intercept term, which can be perfectly reproduced by an equal distance solution.

Degenerate solutions can be recognized by the fact that almost all distances from individual points to stimulus points are equal. Additionally, the degenerate solutions can also be recognized by the transformation plot. The transformation plot of an absolute degenerate solution will show a horizontal line (no slope and an intercept equal to some positive constant) for the unconditional case.

![Figure 4: The Degenerate Solution](image)

For the family composition study, we turned the penalty of PREFSCAL off (\( \omega = 0 \)) and got the expected degenerate solution. Consequently, almost all stimulus points collapsed to a single point in the unfolding configuration (left panel of Figure 4). The fact is also reflected in the transformation plot (right panel of Figure 4), where it shows the horizontal lines that are characteristic of a degenerate solution. Since the family
composition study is a row-conditional case, thus, the transformation plot presents a horizontal line for each row.

The degenerate unfolding solutions occur when both the slope and the intercept are estimated freely. It will identify the model and subsequently avoid degeneracy, if either or both parameters is fixed to a constant (Busing, 2006). The transformation which has the fixed intercept and a free slope, coincides with a ratio transformation and will not lead to a degenerate solution. Alternatively, if the slope of the transformation remains fixed, equally transformed proximities cannot occur so that a degenerate solution will be avoided as well. Consequently, the degenerate problem can be solved by fixing one of the two parameters.

1.1.3 PREFSCAL

Busing, Groenen, and Heiser (2005) avoid degeneracy by penalizing on Karl Pearson’s coefficient of variation $v(.)$ (Pearson, 1896). The squared of PENALIZED STRESS is defined as

$$
\sigma^2_p(\gamma, d) = \sigma^2_n(\gamma, d)\mu(\gamma),
$$

where $\sigma_n$ is NORMALIZED RAW STRESS. The squared of NORMALIZED RAW STRESS is given by:

$$
\sigma^2_n = \frac{||\gamma - d||^2_W}{||\gamma||^2_W},
$$

where $||\gamma - d||^2_W$ is the weighted sum of squared Euclidean norm of the differences between the transformed data $\gamma=vec(\Gamma)$ and the distances $d=vec(D)$ in the metric $W$ and $||\gamma||^2_W$ is the weighted sum-of-squares of $\gamma$ (Busing, 2010). The $\lambda$ parameter constitutes a lack of penalty ($0 < \lambda \leq 1$), and when $\lambda$ increases, the influence of the penalty term decreases. The penalty function is discriminated between the unconditional case and the row-conditional case, as the variation of row-conditional transformations
depends on single rows. In the unconditional case, the penalty function is defined as:

$$\mu(\gamma) = 1 + \frac{\omega}{v^2(\gamma)}.$$  (3)

In the row-conditional case, the penalty function is defined as:

$$\mu_c(\Gamma) = \frac{1}{n} \sum_{i=1}^{n} \left( 1 + \frac{\omega}{v^2(\gamma_i)} \right),$$  (4)

where $v(\gamma_i)$ is the variation of the pseudo-distance in row $i$. The $\omega$ is a range parameter: when $\omega$ is large, the large values of the variation coefficient will cause relatively effective penalties. Penalized stress is minimized by an alternating least squares procedure, in which we alternate finding an update for the configuration and for the transformation; the algorithm has been implemented in PREFSCAL (available in IBM SPSS statistics). Details can be found in Busing (2010).

### 1.1.4 PREFSCAL-II

In this paper, we use the new version of PREFSCAL (Busing and Heiser (2015), unpublished paper). The new version of PREFSCAL uses a new loss function which is defined as the RAW STRESS plus a penalty function:

$$f = \sigma_r + \mu(\gamma).$$  (5)

The RAW STRESS is given by $||\gamma - d||_W$ and the additive penalty function is defined as the inverse of the coefficient of variation of the transformed data $\gamma$: $\lambda \frac{\gamma}{v^2(\gamma)}$. In the unconditional case, the loss function is defined as

$$f = ||\gamma - d||_W + \lambda \frac{||\gamma||_M}{||\gamma||_V},$$  (6)

where $M = w_{++}^{-1} w w', w_{++} = 1' w$, and $V = W - M$. In the row-conditional case, the loss function is defined as:

$$f_c = \sum_{i=1}^{n} \left( ||\gamma_i - d_i||_{W_i} + \lambda \frac{||\gamma_i||_{M_i}}{||\gamma_i||_{V_i}} \right).$$  (7)
1.2 Cross-Validation

Cross-Validation is a method that is the most widely used for estimating prediction error (Hastie, Tibshirani, Friedman, & Franklin, 2005). If we have enough data, we would randomly divide the dataset into three parts: a training set, a validation set, and a test set. They are used to fit the models, estimate prediction error for model selection, and assess performance of the final chosen model, respectively. But it is not always possible, because of the data scarcity problem. K-fold cross-validation is designed to solve the data scarcity problem. In K-fold cross-validation we split data into K roughly equal size parts. We use the K−1 parts of the data to fit the model and calculate the prediction error of the fitted model on the kth part of the data. We do this for k = 1, 2,..., K and combine the K estimates of prediction error. Each part of the data can be used for training and none has to be held back in a separate test set.

But the choice of K still needs discussion. K is usually chosen as 5, 10 or N (N is the total number of samples) corresponding to the well known 5-fold, 10-fold, and leave-one-out cross-validation, respectively. Basically, the more complex the model, the lower the bias but the higher the variance. If K is a large number (say K=N), the size of each part is very small. The cross-validation estimate of prediction error is approximately unbiased for the true prediction error but has high variance. Besides, it requires N applications of the learning method which leads the computational burden problem. On the other hand, if K is a small number (say K=5), the cross-validation estimate of prediction error has lower variance. Whether bias could be a problem, depends on the various training set sizes, for it has influence on the performance of the learning method.

Figure 5 depicts a hypothetical learning curve plot of 1-Err versus the size of the training set N where Err=E[L(Y, ˆf(X))] from Hastie, T. et al (2005). The performance of the classifier dramatically rises from training set of size 0 to around 50 and then the speed of increase becomes slight until training set of size 150, where is a leveling off. For 5-fold cross-validation, if the size of the training set is 200, the classifier estimate over 160 observations is roughly similar to that of the full training set in Figure 5. However,
if the size of the training set is 50, the performance of the classifier estimate over 40 observations is much worse than that of the full training set. The second example suffers from much bias and shows when $K$ is small, bias becomes a problem in a small size of the training set condition. Overall, according to Breiman and Spector (1992) and Kohavi (1995), 5-fold and 10-fold cross-validation are suggested as a good compromise.

![Figure 5: Hypothetical Learning Curve For a Classifier On a Given Task: a plot of $1-\text{Err}$ ($\text{Err}=\mathbb{E}[L(Y, \hat{f}(X))]$) versus the size of the training set $N$.](image)

1.2.1 Cross-Validation idea in Unfolding

In the ordinary cross-validation, the first step is that we roughly split the whole data into $K$ parts at random. In the cross-validation of row-conditional unfolding, the data are randomly divided into $K$ parts for each row separately. For each row, the number of missing cells is equal to the size of test set. Two scenarios of 5-fold cross-validation are schematically depicted like this:
Figure 6: Scenarios of 5-fold Cross-Validation

The left panel is an example of the ordinary cross-validation and the right panel is an example of the unfolding cross-validation. In the above 5-fold cross-validation case, we roughly split the whole data into 5 parts at random for the ordinary cross-validation. In the cross-validation of row-conditional unfolding, the data are randomly divided into 5 parts for each row separately. The remaining steps of cross-validation are the same in the ordinary one and the row-conditional unfolding.

1.2.2 Why we use Cross-Validation in Unfolding?

The reason why we introduce the cross-validation to unfolding is because we want to use cross-validation to choose the unknown tuning parameters and to estimate the prediction error of the final model. There are several unknown tuning parameters in unfolding we can focus on, such as the dimensionality of the solution, the transformations, and the penalty parameter $\lambda$. In our row-conditional nonmetric unfolding study, we focus on the penalty parameter $\lambda$, because $\lambda$ is unknown and a technical issue. When users use the procedure, the value of $\lambda$ should be found automatically.

We want to choose the value of penalty parameter $\lambda$ by using cross-validation. Before we can start using the cross-validation for this purpose we need to decide what prediction method and what prediction error measure we should used. We set up a simulation study for a range of values for the penalty parameter $\lambda$ and make several design factors. The simulation study can be seen as a kind of exploratory analysis in which factors are manipulated that may make a difference. Before we can start the actual cross-validation, the simulation is a pilot study in which we look whether the penalty parameter $\lambda$ has any influence on the prediction methods and the prediction measures.
2 Research Questions

According to the aim of our study and the steps of cross-validation, we formulate the following four research questions:

1. Can we find the optimal value of penalty parameter $\lambda$ by using prediction methods under different circumstances?

2. What kind of prediction method should be used in the cross-validation of row-conditional unfolding?

3. What kind of prediction error measure can be used for quality in the cross-validation of row-conditional unfolding?

4. What other factors besides the penalty parameter $\lambda$ contribute to the quality of prediction in the cross-validation of row-conditional unfolding?

We used two prediction methods, two prediction error measures, and 5 independent factors which might have influence on the accuracy of prediction. They are explained in the next section.

3 Method

3.1 Missing Data

Most unfolding procedures can handle missing data by disregarding the missing entries in the data. According to Busing and De Rooij (2009), even if more than half data is missing, the solutions could still be reliable. Therefore, it presents no serious problem for finding solutions, when we leave out part of the data to estimate the prediction error and to fit the unfolding model on the nonmissing part.

3.2 Data Generation

We generated data following Busing, Groenen, and Heiser (2005). Both row coordinates $X$ and column coordinates $Y$ were drawn from a standard normal distribution.
The Euclidean distances between these coordinates were calculated. Error was added by multiplying the distances with the exponential of a standard normal distribution, corresponding to imposing log-normal error on the distances (Wagenaar and Padmos, 1971). To create rankings with a limited number of tie-blocks, we used binning. Statistical data binning was the way that a number of values were grouped into a small number of tie-blocks. We defined the tie-blocks in such a way that observations within each tie-block were considered equal. Then with only one observation in each tie-block, rankings were called full ranking. In our study, we chose 5 tie-blocks for partial rankings. The number of observations in each tie-block equals the number of columns divided by 5. Data were transformed for each row separately. To create incomplete data, we set a zero-one weight matrix $W$ to handle missing cells. The entry in row $i$ and column $j$ was 0 if the $i$th individual’s preference of the $j$th stimulus was missing and 1 if it was not. Since row-conditional nonmetric unfolding was considered here, each respondent had an equal number of missing in random positions.

3.3 Prediction in Unfolding

We used two prediction methods: the multiple prediction method and the single prediction method. The idea of the multiple prediction method is to use the other individuals’ preferences to help predicting the ranking of stimulus for a certain individual. The idea of the single prediction method is to predict the ranking only relying on the individual itself.

Multiple Prediction

In the multiple prediction method, each column of the pseudo-distance matrix $\hat{D}$ is considered as a unit. And each unit has its own boundaries to predict rankings, that is, each column has its boundaries which are decided by its multiple respondents. For our family composition study, for example, the raw preference matrix $\Delta$ is a 82 rows by 16 columns matrix. In the 25% missing condition, each row has 4 (16×0.25) random missing cells for our row-conditional case. We use non missing cells to predict the missing cells column by column.
We would like to use Figure 7 to explain how to determine the boundaries in each column. Figure 7 is the plot of raw preference matrix $\Delta$ with missing cells against its corresponding pseudo-distance matrix $\hat{D}$ for the fifth column. Here two auxiliary lines are added. The red line is the upper bound of the pseudo-distance of ranking=2, and the green line is the lower bound of the pseudo-distance of ranking=3. We take the average of the upper bound of group ranking=2 and lower bound of group ranking=3 and regard the resulting value as the boundary between the groups. We use the same methods to determine the boundaries for the groups with rankings 4 to 16 and finally we get 15 boundaries for the 16 rankings. In our family composition study, the raw preference matrix $\Delta$ has 16 columns which means we can make 16 figures like Figure 7 and apply the same procedure to determine their boundaries. The prediction of missing cell values in each column belonging to each rank will be done based on these boundaries, column by column.

Figure 7: Raw preference matrix $\Delta$ with missing cells against its corresponding pseudo-distance matrix $\hat{D}$ for the fifth column plot

The distance matrix $D$ is not necessary monotonically related to the raw preference matrix $\Delta$ since it contains the fit error while the pseudo-distance matrix $\hat{D}$ is restricted to be monotone. The boundaries could be less overlapped in the pseudo-distance matrix
\( \hat{D} \) case than in the distance matrix \( D \) case. That is the reason why the pseudo-distance matrix \( \hat{D} \) is used instead of the distance matrix \( D \).

**Single Prediction**

For the single prediction method, we directly rank the distance matrix for each row separately in the full ranking condition. The smallest distance gets rank 1 and the second smallest distance gets rank 2, etc. For the partial ranking type, since the generated rankings are designed to have 5 tie-blocks, it means the maximum ranking is 5. When we predict the partial ranking, we use the same idea as for the data generation. We bin the distances into 5 intervals in ascending order. Each interval represents a ranking. Observations in each interval are considered equal.

### 3.4 Simulation Study Design

The simulation study is conducted to explore, for two different prediction methods, if there is a minimum prediction error for different penalty values of the parameter \( \lambda \) under different circumstances. The data in this simulation study are generated as described in Section 3.2. Values for the penalty parameter \( \lambda \) range from 0.1 to 2 with increments of 0.1. The larger the value of \( \lambda \), the stronger the penalty (see Equation 7). The influence of data size, levels of error, levels of missing, and ranking types are taken into consideration for the study as well. Two types of rational initial configurations are used. The full factorial design is summarized in Table 2.

<table>
<thead>
<tr>
<th>Factor</th>
<th>No. of Levels</th>
<th>Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sizes of Data</td>
<td>2</td>
<td>40 Individuals, 10 Stimuli(400)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>80 Individuals, 20 Stimuli(1600)</td>
</tr>
<tr>
<td>Rational Starts</td>
<td>2</td>
<td>Torgerson Triangle Inequality Start</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Centroid Start with First Choice</td>
</tr>
<tr>
<td>( \lambda )</td>
<td>21</td>
<td>0, 0.1, 0.2, \ldots, 1.9, 2.0</td>
</tr>
<tr>
<td>Ranking Types</td>
<td>3</td>
<td>Full rank order</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Partial Ranking 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Partial Ranking 2</td>
</tr>
<tr>
<td>Percentage of Error</td>
<td>3</td>
<td>None, 15%, 30%</td>
</tr>
<tr>
<td>Percentage of Missing</td>
<td>2</td>
<td>25%, 50%</td>
</tr>
</tbody>
</table>
3.5 Description of the Factors

Six independent factors are considered that can have influence on the prediction accuracy. The factors of data sizes, penalty parameter $\lambda$, percentages of error and percentages of missing are easy to understand as their literal meanings so we do not explain them in details. The other two factors, rational starts and ranking types, are described in detail below.

Initial Configuration

The methods to compute the initial configuration can be categorized into rational starts, user-provided starts, and random starts. The random starts are most likely not ideal for finding the optimal solution, because it does not pay any attention to the data. The idea of rational starts is that a start is calculated by an other procedure to solve the same objective. We use two rational starts in our study: the centroid start and the triangle start. Details on these two types of rational starts can be found in Heiser and De Leeuw (1979) and Busing (2010).

Ranking types

There are two types of rankings considering in our study, as well as two types of ties approaches. The full ranking is the ranking which has no tied objects and its maximum ranking is equal to the number of columns in our case. The partial ranking is the ranking which has two or more objects are tied in a so-called tie-block and obtain the same rank number so that the maximum rank is less than the number of columns. In our study, the number of tie-blocks is 5 which means the maximum partial ranking is 5. In case of ties, there are two approaches implemented in PREFSCAL. The first approach to ties allows ties to be untied. The second approach to ties requires ties to be kept tied. Since full ranking does not have ties, there is no difference between using the first and the second approach for full ranking. The second approach was used in full ranking type and both first and second approaches were used in partial ranking type: partial ranking type 1 and partial ranking type 2.
3.6 Outcome Measures

3.6.1 Prediction Error Measures

Two prediction error measures will be used. The loss functions for measuring errors between original ranking and prediction ranking are denoted by $\sigma_x$ and $\sigma_s$ for absolute error and squared error, respectively. Let $r_j$ and $s_j$ be coordinate vectors for original ranking $j$ and predicted ranking $j$, respectively. $M$ is the set of indexes for missing elements. $N_M$ is the total number of missing elements. The prediction error $\sigma_x$ and $\sigma_s$ between $r_j$ and $s_j$ are defined as

$$
\sigma_x = \frac{1}{N_M (\max(r) - 1)} \sum_{j \in M} |r_j - s_j| \text{ (absolute error), and}
$$

$$
\sigma_s = \frac{1}{N_M (\max(r) - 1)^2} \sum_{j \in M} (r_j - s_j)^2 \text{ (squared error),}
$$

where $r$ is the set of original ranking and $\max(r)$ is the largest rank in the set $r$.

3.6.2 Odds Ratio and 95% Confidence Intervals

Odds Ratio

An odds ratio ($OR$) is used to measure the association between a factor level and an outcome and it is frequently used in case-control studies. The $OR$ represents the odds that an outcome will occur given a particular factor level, instead of another level of the factor. We use a two by two frequency table (see Table 3) to illustrate the calculation of an $OR$.

<table>
<thead>
<tr>
<th>Factor Levels</th>
<th>Outcome Types</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>$a$</td>
</tr>
<tr>
<td>2</td>
<td>$c$</td>
</tr>
</tbody>
</table>

Now the odds ratio is defined

$$
OR = \frac{a/b}{c/d},
$$

where the value of odds ratio can be interpreted as follows: if $OR = 1$, it means that factor level 1 has same distribution of outcome types as factor level 2. If $OR > 1$,
it means that factor level 1 has higher odds of outcome types than factor level 2. If $OR < 1$, it means that factor level 1 has lower odds of outcome types than factor level 2.

95% Confidence Intervals

The 95% Confidence Intervals (CI) is used to estimate the precision of the $OR$. The 95% is used as a proxy for presence of statistical significance if it does not overlap the null value ($OR = 1$).

\[
\text{Upper 95\% CI} = e^{\ln(OR) + 1.96\sqrt{\left(\frac{1}{a}\right) + \left(\frac{1}{b}\right) + \left(\frac{1}{c}\right) + \left(\frac{1}{d}\right)}}
\]

\[
\text{Lower 95\% CI} = e^{\ln(OR) - 1.96\sqrt{\left(\frac{1}{a}\right) + \left(\frac{1}{b}\right) + \left(\frac{1}{c}\right) + \left(\frac{1}{d}\right)}}
\]

3.6.3 Measures of Fit

Kendall’s $\tau_x$

Kendall’s $\tau$ is a ranking correlation coefficient and it performs well when ties are not permitted. Kendall (1948) extended Kendall’s $\tau$ to Kendall’s $\tau_b$ to deal with the case of weak orderings. However, Kendall’s $\tau_b$ has problems resulting from the way it handles ties. For instance, Kendall’s $\tau_b$ is flawed to consider the distance metric associated with it since Kendall’s $\tau_b$ violates the triangle inequality. Emond and Mason (2002) noted problems with $\tau_b$ in details.

Emond and Mason (2002) present a new ranking correlation coefficient called Kendall’s $\tau_x$. Kendall’s $\tau_x$ is closely related to Kendall’s $\tau$ but differs from it in the way ties are handled. And $\tau_x$ differs from Kendall’s $\tau_b$ by using a value of 1 in the ranking matrix to represent ties instead of the value of 0. They have chosen the name $\tau_x$ (‘x’ for extended) for the new rank correlation coefficient. A weak ordering $A$ of $n$ objects may
be represented using the $n \times n$ score matrix $\{a_{ij}\}$ as follows:

$$a(ij) = \begin{cases} 
1 & \text{if object } i \text{ is ranked ahead of or tied with object } j, \\
-1 & \text{if object } i \text{ is ranked behind object } j, \\
0 & \text{if } i = j.
\end{cases}$$

The $\tau_x$ between two weak orderings $A$ and $B$ is given by the dot product of their score matrices, i.e.,

$$\tau_x (A, B) = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{ij}}{n(n-1)}.$$  

Kendall’s $\tau_x$ is computed to compare the rankings of the raw preferences $\Delta$ and the estimated preferences $\hat{\Delta}$.

**STRESS-2**

The square of Kruskal’s STRESS-2 (Kruskal & Carroll, 1969; Kruskal et al., 1978) is a fit measure comparing the transformed preferences $\gamma$ and the distances $d$. It uses the variance of the distances as normalization factor. It is defined as the sum of squared differences between $\gamma$ and $d$, divided by the variance of the distances, i.e.,

$$\sigma_2^2 = \frac{||\gamma - d||^2_W}{||d - \overline{d}||^2_W},$$  

(8)

where $\overline{d} = (nm)^{-1} \sum_i \sum_j d_{ij}$ and $\overline{d} = 1 \overline{d}$. We used the square of Kruskal’s STRESS-2 as a fit measure of the pseudo distances $\hat{D}$ and the distances $D$ in our study.
4 Results

The result section contains three parts: the main results, the other results, and a summary.

4.1 Main results

We made error bars (see Figure 8) of absolute prediction error measure $\sigma_x$ for each condition and tried to find the optimal values of the penalty parameter $\lambda$. There are three main circumstances in total among all error bars. Each panel corresponds to one specific circumstance including two examples which are different but have the same characteristics. The three main circumstances get the lowest prediction error, at the beginning of the range of the $\lambda$’s, in the middle of the range of the $\lambda$’s, and at the end of the range of the $\lambda$’s, respectively.

The panels in the first row of Figure 8 show the lowest prediction errors at the beginning of the $\lambda$’s, then the value of $\sigma_x$ goes up and reaches a peak, after which it decreases and finally levels out. The situation was likely caused by a local minimum at the beginning. The procedure got stuck in a local minimum since it started from a really good initial configuration.

The panels in the second row exhibit the lowest prediction error at the lower-range values of $\lambda$. $\sigma_x$ is larger at the smallest values of $\lambda$ when the penalty is turned off (i.e. $\lambda=0$), but then the value of $\sigma_x$ declines due to the penalty effect. After the value of $\sigma_x$ reaches its lowest value, it rises and then follows roughly the same pattern as the first circumstance depicts in the first row panels. This was the situation we were looking for, because we could find an optimal $\lambda$ value by using our prediction methods and prediction error measure.

The panels in the third row display the lowest $\sigma_x$ at the end of the $\lambda$ values. The pattern in the left panel is distinct from that in the right panel. The value of $\sigma_x$ in the left panel declines at first and then it goes up until it reaches its peak and then goes down before
finally flattening out and has the lowest value at the end of λ’s. σ_x in the left panel has
an extremely large value at the beginning, after it has a dramatal decrease and gets
the lowest prediction error at the end. This circumstance was not good as well because
it got the lowest prediction error at the end of penalty parameter λ values. When the
penalty parameter λ increases it causes the transformation to become linear and not
ordinal any more. We were seeking an optimal value of λ for an ordinal transformation,
but instead we found an optimal value for λ when the data was linearly transformed.

A one-way analysis of variance was performed to test whether the λ values were signi-
icantly different as we can see in the error bars of the prediction error measure σ_x. Table
4 provides the tests of the between-subject effects, including effect sizes, expressed as η^2_p.
According to Cohen (1988), a η^2_p of .010 indicates a small effect, .059 a medium effect,
and .138 a large effect. The differences between different λ values are significant and
η^2_p indicates that the effect of λ values on σ_x is medium to large in size. The ANOVA
result demonstrates the penalty function works for the prediction error and changing
the penalty parameter causes a significant change in prediction error.

Table 4: ANOVA Tests of Between-subjects Effects (Comparing Prediction Error of
Unfolding Solutions using different λ)

<table>
<thead>
<tr>
<th>Source</th>
<th>Type III Sum of Squares</th>
<th>df</th>
<th>F</th>
<th>Sig.</th>
<th>η^2_p</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corrected Model</td>
<td>51.985</td>
<td>20</td>
<td>1087.919</td>
<td>.000</td>
<td>.067</td>
</tr>
<tr>
<td>Intercept</td>
<td>5872.908</td>
<td>1</td>
<td>5872.908</td>
<td>2458121.039</td>
<td>.000</td>
</tr>
<tr>
<td>λ</td>
<td>51.985</td>
<td>20</td>
<td>1087.919</td>
<td>.000</td>
<td>.067</td>
</tr>
<tr>
<td>Error</td>
<td>722.440</td>
<td>302379</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>6647.442</td>
<td>302400</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure 8: Six panels of three main circumstances (rows) among all error bars of the prediction error measures $\sigma_x$: For each circumstance, we presented two different cases (columns) that had the same characteristics.

Table 5 was utilized to count the situations where the lowest error occurs for the absolute prediction error measure $\sigma_x$. The beginning, the cross, and the end indicate the lowest prediction error appears at the beginning of the $\lambda$ values, at the middle of the $\lambda$ values, and at the end of the $\lambda$ values, respectively.
We got the following obvious conclusions from the large cross table. Considering the error free data (Error 1) and full ranking situation, we can make the following observations. If the data are error free (Error 1) and have large size (Data size 2), the full ranking case always yields the optimal values of penalty parameter λ. If the data are error free and have small size (Data size 1), the full ranking case cannot handle a large percentage of missing (see left upper part of the Table 5). But the circumstances in partial ranking are not as good as that of the full ranking. Partial ranking types cannot find the optimal value of the penalty parameter λ under any kind of situation (see right part of the Table 5). It is also a remarkable fact that large values of the penalty parameter λ are always needed in the large percentage of error (Error 3) and large data size (Data size 2) condition (see bottom part of the Table 5).

In order to get detailed information from the cross table, we split the table so we could observe each factor individually. Since we would like to focus on the usage of the penalty parameter λ and the Beginning situation did not use the penalty function, we converted...
the $X$ and the End situations into a Non Beginning situation for the following analyses.

<table>
<thead>
<tr>
<th>Table 6: Frequency table of Ranking Types</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Beginning</strong></td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>Full Ranking</td>
</tr>
<tr>
<td>Partial Ranking 1 and 2</td>
</tr>
<tr>
<td><strong>Total</strong></td>
</tr>
</tbody>
</table>

According to the $OR$ formula in Section 3.6.2, the odds ratio and its 95% confidence interval for Table 6 are: $OR = 0.912 \ [0.324, 2.565]$. The odds ratio is less than 1 which means that the effect of the ranking types is associated with lower odds of the outcome of Beginning but the odds ratio is close to 1 which indicates no big difference between two odds. The 95% confidence interval confirms the deduction as well. The 95% confidence interval spans the value 1.0 so that we can conclude that the association between the outcome of Beginning and ranking types is not statistically significant at $\alpha = 0.5$.

<table>
<thead>
<tr>
<th>Table 7: Frequency table of Percentages of Missing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Beginning</strong></td>
</tr>
<tr>
<td>Missing 25%</td>
</tr>
<tr>
<td>Missing 50%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
</tr>
</tbody>
</table>

The odds ratio and the corresponding 95% confidence interval for Table 7 are as: $OR = 0.058 \ [0.015, 0.225]$. For 25% missing, the odds of the Beginning outcome is 0.091, and for 50% missing, the odds of the Beginning outcome becomes 1.571. The large difference between the two odds is reflected in the small odds ratio of .058 which is much lower than 1. It illustrates that percentages of missing associated with lower odds of the outcome of Beginning. The 95% confidence interval shows that the $OR$ is significantly different from 1. Therefore, there appears to be a dependence of the Beginning outcome and percentages of missing.
Table 8: Frequency table of Data Sizes

<table>
<thead>
<tr>
<th></th>
<th>Beginning</th>
<th>Non Beginning</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Size 1</td>
<td>21</td>
<td>15</td>
<td>36</td>
</tr>
<tr>
<td>Data Size 2</td>
<td>4</td>
<td>32</td>
<td>36</td>
</tr>
<tr>
<td>Total</td>
<td>25</td>
<td>47</td>
<td>72</td>
</tr>
</tbody>
</table>

The odds ratio and its 95% confidence interval were calculated based on Table 8. If the factor level is data size 1, odds of the Beginning outcome is 1.4. If the factor level is data size 2, odds of the Beginning outcome becomes 0.125. The probability of the Beginning outcome is much higher in data size 1 situation than in the data size 2 situation. It also shows on its odds ratio: $OR = 11.200$ which is well over 1. The 95% confidence interval $[3.265, 38.420]$ does not contains 1 which confirms that $OR$ is significantly different from null value ($OR=1$) at $\alpha = 0.5$ level. Thus, we could conclude from the above analysis, changing the data sizes had a significant influence on the outcome of Beginning.

Table 9: Frequency table of Rational Starts

<table>
<thead>
<tr>
<th></th>
<th>Beginning</th>
<th>Non Beginning</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rational Start 1</td>
<td>12</td>
<td>24</td>
<td>36</td>
</tr>
<tr>
<td>Rational Start 2</td>
<td>13</td>
<td>23</td>
<td>36</td>
</tr>
<tr>
<td>Total</td>
<td>25</td>
<td>47</td>
<td>72</td>
</tr>
</tbody>
</table>

We calculated the odds ratio and its 95% confidence interval for Table 9 as: $OR = 0.885 [0.335, 2.336]$. The 95% confidence spans the value 1.0 so that we can conclude that the association between the outcome of Beginning and rational starts does not reach statistically significant at $\alpha = 0.5$. There is no significant difference between using rational start 1 and rational start 2 in yielding the outcome of Beginning.

4.2 Other results

We first picked up the best $\lambda$ values based on the lowest prediction error it produced and removed the other 20 $\lambda$s and performed the following analyses to test whether all factors we considered perform as expected for our best $\lambda$ values. In the prediction methods part we test the difference between the two prediction methods and then select the better one to do the ANOVA. Following the results of the ANOVA, we pick up significant main
factors to look at them individually.

**Prediction Methods**


<table>
<thead>
<tr>
<th>Methods</th>
<th>( \sigma_x )</th>
<th>( \sigma_s )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Prediction</td>
<td>.104 (.042)</td>
<td>.031 (.018)</td>
</tr>
<tr>
<td>Multiple Prediction</td>
<td>.125 (.038)</td>
<td>.036 (.017)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Between-Subjects Effects</th>
<th>( F )</th>
<th>( Sig )</th>
<th>( \eta_p^2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma_x )</td>
<td>994.041</td>
<td>.000</td>
<td>.065</td>
</tr>
<tr>
<td>( \sigma_s )</td>
<td>277.609</td>
<td>.000</td>
<td>.019</td>
</tr>
</tbody>
</table>

The multivariate analysis of variance test was performed to check whether a significant difference exists on the prediction error measures between the two prediction methods we used in the simulation study. Table 10 provides descriptive statistics and the tests of the between-subject effects. All differences are significant and the descriptive statistics and the effect sizes indicate that the differences in absolute prediction error measure \( \sigma_x \) are of medium size and the differences in squared prediction error measure \( \sigma_s \) are of small size. It means that the prediction error can be better distinguished by the absolute prediction error measure \( \sigma_x \). The single prediction method was slightly better than the multiple prediction method for both prediction error measures. In the following exploration of factors, we will use the results of the single prediction method and the absolute prediction error measure \( \sigma_x \).
The ANOVA tests of the between-subject effects for the single prediction method are presented in Table 11. All main effects are significant \((p < 0.05)\), but vary in effect size. The percentage of error, the percentage of missing, and ranking types have the largest, second largest, and third largest effect sizes respectively (Cohen, 1988), namely very large, large, and medium. The effect size of the rational starts is extremely small comparing to the effect sizes of the other 4 main factors. All two-way interactions except those including rational starts and the interaction between ranking types and the percentage of missing are significant. Although, according to the ANOVA, the rational starts factor is significant, its effect size is too small to have any meaningful effect on the value of \(\sigma_x\). Besides, all its 2-way interactions are non-significant. It is advisable to get rid of the factor rational starts in future analyses. We chose the three largest significant main factors which have at least a medium effect size to look at them individually.
Percentages of Error

Table 12: Descriptive Statistics (Upper Part) and Univariate Tests of Between-Subjects Effects (Lower Part) Comparing Prediction Error of Unfolding Solutions using Different Percentages of Error

<table>
<thead>
<tr>
<th>Percentages of Error</th>
<th>0%</th>
<th>15%</th>
<th>30%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>.058</td>
<td>.102</td>
<td>.151</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>.017</td>
<td>.016</td>
<td>.018</td>
</tr>
<tr>
<td>Univariate Tests</td>
<td>$F$</td>
<td>$Sig$</td>
<td>$\eta^2_p$</td>
</tr>
<tr>
<td></td>
<td>32145.633</td>
<td>.000</td>
<td>.900</td>
</tr>
</tbody>
</table>

Table 12 provides the descriptive statistics for each percentage of error and the univariate test result. The univariate test shows a significant overall difference between the prediction error of the three percentages of error and a large effect size. The descriptive statistics shows that when the percentage of error increases, the prediction is getting worse.

Percentages of Missing

Table 13: Descriptive Statistics (Upper Part) and Univariate Tests of Between-Subjects Effects (Lower Part) Comparing Prediction Error of Unfolding Solutions using Different Percentages of Missing

<table>
<thead>
<tr>
<th>Percentages of Missing</th>
<th>25%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>.095</td>
<td>.112</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>.041</td>
<td>.040</td>
</tr>
<tr>
<td>Univariate Tests</td>
<td>$F$</td>
<td>$Sig$</td>
</tr>
<tr>
<td></td>
<td>3333.009</td>
<td>.000</td>
</tr>
</tbody>
</table>

The descriptive statistics for each percentage of missing and the univariate test result are reported in Table 13. As expected, the percentage of missing follows the same trend as the percentage of error. The lower the percentage of missing, the better the prediction. The univariate test displays a significant overall difference between the prediction error of the two percentages of missing with a large effect size as well. Changing the percentage of missing led to a large alteration in prediction.
Table 14: Descriptive Statistics (Upper Part) and Univariate Tests of Between-Subjects Effects (Lower Part) Comparing Prediction Error of Unfolding Solutions using Different ranking types

<table>
<thead>
<tr>
<th>Ranking Types</th>
<th>Full Ranking</th>
<th>Partial Ranking 1</th>
<th>Partial Ranking 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>.097</td>
<td>.107</td>
<td>.108</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>.041</td>
<td>.041</td>
<td>.042</td>
</tr>
<tr>
<td>Univariate Tests</td>
<td>$F$</td>
<td>$\text{Sig}$</td>
<td>$\eta^2$</td>
</tr>
<tr>
<td></td>
<td>552.264</td>
<td>.000</td>
<td>.133</td>
</tr>
</tbody>
</table>

Table 14 exhibits the descriptive statistics for each ranking type and the univariate test results as well. Full ranking provides a better prediction than partial rankings, according to the descriptive statistics part. For the two partial rankings, partial ranking 1 was slightly better than partial ranking 2, possibly because partial ranking 1 (untie tied ties) had more freedom than partial ranking 2 (keep ties tied) for the transformation to give a better prediction quality. The univariate test presents a significant overall difference between the prediction error of the three ranking types with a medium effect size.

### 4.3 Summary

We got answers for our four research questions from the analyses of the main and subsequent results.

- **Research Question 1**: Can we find the optimal value of penalty parameter $\lambda$ by using prediction methods under different circumstances?

  In the error free and large data size situation, the full ranking case was capable of finding the optimal $\lambda$ value while for the error free and small data size situation, the full ranking case could not handle the large percentage of missing. With increasing of error, in the full ranking situation, it was hard to find the optimal $\lambda$ value. The results in the partial ranking cases were even worse because we could not find the optimal $\lambda$ value in any situation. Thus, the main conclusion was that the full ranking case could find the optimal $\lambda$ value in certain situations but the partial ranking cases was never able to find the optimal $\lambda$ value.
• **Research Question 2 and 3**: What kind of prediction method and prediction error measure should be used in the cross-validation of row-conditional unfolding?

There was not much difference between the two prediction error measures. The effect size $\eta^2_p$ of $\sigma_x$ was larger than that of $\sigma_s$ (see Table 10) indicating that the absolute prediction error measure $\sigma_x$ had higher proportion of the total variability. The two prediction methods were significant different on the prediction error measure $\sigma_x$. The single prediction method was slightly better than the multiple prediction method for both prediction error measures. In the following row-conditional unfolding cross-validation, we could just use the single prediction method and the $\sigma_x$ prediction error measure.

• **Research Question 4**: What other factors besides the penalty parameter $\lambda$ contribute to the quality of the prediction in the cross-validation of row-conditional unfolding?

We considered 5 other factors (except the penalty parameter $\lambda$) which could contribute to the quality of the prediction: size of data, rational starts, ranking types, percentage of error, and percentage of missing. All 5 main effects were significant ($p < 0.05$), but varied in effect sizes. The percentage of error, the percentage of missing, and ranking types had the three largest effect sizes. Although, according to the ANOVA, the rational starts factor was significant, its effect size was too small to have any meaningful effect on the value of $\sigma_x$. Also, in our odds ratio and 95% CI analysis (see Table 9), its 95% CI shows that the $OR$ is not significantly different from 1 and we can conclude that the association between the outcome of beginning and rational starts did not reach statistically significant at $\alpha = 0.5$. These results make it feasible to remove this factor for future analyses. Thus, the answer to question 4 was that all factors showed a significant influence on the quality of the prediction, except for rational starts.
5 Discussion

As the main results showed, we could not find an optimal $\lambda$ value in the different situations for the partial ranking cases but we could find an optimal $\lambda$ value in the some situations for the full ranking by using our prediction methods. In this section, we tried to explore the reason why we could not find an optimal $\lambda$ in all situations. Considering the prediction results, we got the following deduction to focus on: the reason we could not find an optimal $\lambda$ in all situations could be in the new PREFSCAL procedure which did not provide us with an optimal solution. We used prefect data to test it in the following section.

5.1 Perfect Data

We wanted to know whether we were able to get the perfect solution under the perfect data condition (no error and no missing). If we got a perfect solution under the no error and no missing data condition, according to Busing and De Rooij (2009), we should still get a perfect solution after we randomly generated missings to the data. If we could not find the perfect solution for perfect data, it meant that our algorithm might not be able to find the perfect solution and prediction error might be present before starting the prediction process anyway. We did a small simulation study to test the algorithm. Full ranking, partial ranking 1, and partial ranking 2 in the perfect data condition were considered, where each case was replicated 50 times. The recovery of the unfolding solutions were quantified using STRESS-2 and Kendall’s $\tau_x$ (STRESS-2 is metric measure and Kendall’s $\tau_x$ is nonmetric measure).

The boxplots in Figure 9 display the distribution of the recovery measures. The results of the simulation study were not as we expected. Kendall’s $\tau_x$ measure did not get 1 and STRESS-2 measure did not get 0 either. In other words, we did not find a perfect solution in the perfect data situation regardless of using either the Kendall’s $\tau_x$ measure or the STRESS-2 measure. Why couldn’t we find the perfect solution for perfect data? At first, we had to consider the statistical measures for the solution. STRESS-2 is a
metric measure which gives different values moving an ideal point around in its isotonic region, the region with the same rank orders towards the items. The rank orders can be perfect, while STRESS-2 is still non-zero. This problem can be circumvented with the use of a rank order measure, like Kendall’s $\tau_x$.

![Figure 9: Six panels: Boxplots of the Kendall’s $\tau_x$ measure (left panel) and the STRESS-2 (Right panel) measure for the three different designs for specifying ranking types: the full ranking (the first row), the partial ranking 1 (the second row) and the partial ranking 2 in the perfect data condition (the third row).]

However, from our boxplots, the Kendall’s $\tau_x$ measure did not provide us with the perfect solution as well. The new loss function seemed to be the reason:

$$f_c = \sum_{i=1}^{n} \left( \| \gamma_i - d_i \|_W + \lambda \frac{\| \gamma_i \|_M}{\| \gamma_i \|_V} \right).$$

If there was no penalty term, we were trying to get the estimated coordinates exactly
spot but we had to make sure there was the variation in our case because of the penalty term in the loss function. The penalty was responsible for the deviation from the actual points and the deviation from the actual isotonic region in some cases. So it created the deviation by deviating from the actual point. It was possible that we found the more variation and less stress case. This was the reason why we had perfect data but we still failed to find the perfect solution. The following example explained this further. The two cases used the same penalty parameter $\lambda$ value and we got two loss function values:

$$F_1 = 0 + 1.0 \times 1.0 = 1.0$$
$$F_2 = 0.2 + 1.0 \times 0.5 = 0.7.$$  

Considering the function value, $F_2$ was better than $F_1$. If we wanted a perfect solution in $F_1$, the STRESS-2 was equal to 0 and the whole function value was 1. In $F_2$, the STRESS-2 was 0.2 but its variation led the whole function value to equal 0.7. The function value became better but it still had a worse stress value. Although it was a better solution in our case, it had error. That caused by the penalty as in $F_2$ and made STRESS-2 $> 0$ and Kendall’s $\tau_x < 1$. So we found the perfect solution but we could not measure the fact that the perfect solution.

6 Conclusion

Cross-validation can be used to determine the unknown parameters tuning and to estimate the prediction error of the final model. We used a simulation study to find the suitable prediction method and prediction measure which can be used in our nonmetric unfolding cross-validation. Four research questions were answered from the results of our analyses. In our study, two prediction methods and two prediction error measures were used to find the optimal penalty parameter $\lambda$ value under different circumstances. We could find the optimal $\lambda$ value in some certain situations for the full ranking case but we could not find the optimal $\lambda$ value for the partial ranking case in any situations, unfortunately. The single prediction method is slightly better than the multiple one in the row-conditional case by considering the prediction error measure $\sigma_x$. The effect size of rational starts factor is too small to have any meaningful effect on the prediction. We recommended using the single prediction method and using absolute prediction error
measure $\sigma_x$ as the prediction method and prediction error measure in the nonmetric unfolding cross-validation. Besides, there is no need to keep rational starts as a factor contributing to the quality of prediction.
References


Guttman, L(1950). The basis for scalagram analysis. In Stouffer, S. A. et al, Measure-


Appendix A  Notations

Symbols

\[ n \] number of row objects
\[ m \] number of column objects
\[ \Delta \] raw preferences \((n \times m\) matrix\)
\[ \Gamma \] transformed preferences \(\Gamma = f(\Delta)\) \((n \times m\) matrix\)
\[ \Gamma^2 \] element-wise squared preferences \(\Gamma\) \((n \times m\) matrix\)
\[ \lambda \] penalty parameter
\[ \phi \] Tuckers’ congruence coefficient
\[ X \] coordinate matrix for row objects
\[ Y \] coordinate matrix for column objects
\[ X_0 \] initial coordinate matrix for row objects
\[ Y_0 \] initial coordinate matrix for column objects
\[ W \] preference weights \((n \times m\) matrix\)
\[ D \] Euclidean distances between rows of columns of \(X\) and \(Y\) \((n \times m\) matrix\)
\[ \hat{D} \] pseudo-distances matrix \((n \times m\) matrix\)
\[ \hat{\Delta} \] estimated preferences
\[ I \] \(I\) scale
\[ J \] \(J\) scale
\[ \tau_x \] Kendall’s extended rank order correlation

Functions

\[ d(.) \] Euclidean distance function
\[ \tau_x(.) \] \(\tau_x\) rank correlations coefficient
\[ \sigma_x \] absolute prediction error measure
\[ \sigma_s \] squared prediction error measure
\[ f(.) \] transformation function
Appendix B  R code for Functions

```
### Thesis R code ###

# Functions Defined for the Analysis#

# data generation function#

generate.row.ordinal <- function(n, m, p, type, error)
{
  # n: the number of rows
  # m: the number of columns
  # p: the number of dimensions
  # error: the percentage of error
  # type: ranking type
  x <- matrix(rnorm(n*p), n, p) # create n row coordinates in p dimensions
  y <- matrix(rnorm(m*p), m, p) # create m column coordinates in p dimensions
  er <- matrix(exp(rnorm(n*m, 0, sd=error)), n, m) # create error
  D <- distxy(x, y) # create Euclidean distance

  # check the input ranking type and do row conditional transformation
  if(type == "full.rank"){
    delta <- t(apply(D, 1, rank))
  }
  else if(type == "partial.rank"){
    # Partial rank order
    r <- round((dim(D)[2])/5)*(c(1, 2, 3, 4))
    boundary <- t(apply(D, 1, function(x)sort(x)[r]))
    temp <- function(x){
      res <- numeric(ncol(D))
      for(i in 1:(ncol(D))){
        for(j in (ncol(boundary) : 1)){
          if(D[x,i]<boundary[x, j] || D[x, i]==boundary[x, j]){res[i]=j
        }else{if(D[x, i]>boundary[x, ncol(boundary)]){res[i]= ncol(boundary)+1}
        }else stop
      }
    }
    delta <- t(sapply(1:nrow(D), function(x)temp(x)))
  }else stop("insufficient input type")

  # finish transformation
  # output
  result <- list(D=D, delta=delta)
  return(result)
}
```
# centroid start

c.fun <- function(delta, p)
{
  # ref: Willem J. Heiser and Jan de Leeuw (May, 1979).
  # How to use smacof–III: A program for metric
  # multidimensional unfolding.
  # n is the number of rows
  # m is the number of columns
  # delta is the distance matrix
  # p is the dimensionality
  # produce the indicator matrix E (first choice) [page2, equation(4a
  #   , 4b)]
  n <- nrow(delta)
  m <- ncol(delta)
  E <- matrix(NA, n, m)
  # compute column minimum
  mi <- apply(delta, 2, min)
  for (j in 1:m)
  {
    for (i in 1:n)
    {
      if (delta[i, j] > mi[j])
        E[i, j] <- 0
      else
        E[i, j] <- 1
    }
  }
  one <- matrix(1, 1, n)
  # produce the diagonal matrix M
  M <- diag(as.numeric(one %*% E))
  # compute ME'DE*2 [page7, equation(24)]
  D2 <- solve(M %*% t(E) %*% (delta^2)
  # centering operator [page5, equation(13)]
  J <- diag(m) - 1/m
  # double centering [page7, equation(26)]
  DD <- -(1/2)*%*% (D2) %*% J
  # eigenvalue decomposition
  De <- 0.5*(DD + t(DD))
  eig <- eigen(De)
  # calculate initial column objects configuration
  Y <- ((eig[[2]])[, 1:p])%*%(sqrt((eig[[1]])[1:p])))
  # calculate initial row objects configuration
  alpha <- matrix(rowSums(Y*Y), length(rowSums(Y*Y)), 1)
  One <- matrix(1, n, 1)
  # X
  X <- -0.5*(delta^2 - One %*% t(alpha))%*%%solve(t(Y)%*%Y))
  # output
  initial <- list(X=X, Y=Y)
  return(initial)
}

# Triangle start
Tri.fun <- function(delta, p)
{
  # ref: author : Frank M.T.A. Busing, Leiden University
```r
# package : dscaling
# function: multidimensional unfolding classical scaling triangle inequality
# n is the number of rows
# m is the number of columns
# delta is the distance matrix
# p is the dimensionality
r <- mdu.classical( delta, p)
X <- r@X
Y <- r@Y
# output
initial <- list(X=X, Y=Y)
return(initial)

boundary <- function(w, delta, dis){
  # w is the weight matrix
  # delta is ranked distance matrix generated by generate.data function
  # dis is the distance matrix generated by mdu
  delta <- delta*w
  dis <- dis*w
  temp.b <- function(x){
    res <- numeric(ncol(delta)-1)
    for (i in 1:(ncol(delta)-1)){
      res[i] <- sum(max(dis[,x][delta[,x]<i]), min(c(dis[,x][delta[,x]>i])))/2
    }
    res
  }
  boundary.crude <- sapply(1:ncol(delta), function(x)temp.b(x))
  boundary.crude
}

missing <- function(n, m, mis){
  if(round(mis*m)==1){
    ind <- t(t(replicate(n, sample(1:m, size=round(mis*m), replace=FALSE))))
  } else{
    ind <- t(replicate(n, sample(1:m, size=round(mis*m), replace=FALSE)))
  }
  w <- matrix(1, n, m)
  for(d in 1:nrow(w)){
    w[d, ind[d, ]] <- 0
  }
  result <- list(w=w, ind=ind)
  result
}
```
# prediction function

## multiple prediction function

`pred.multi <- function(boundary, D, ind){
  # boundary is a matrix made by boundary function
  # D is the distance matrix produced by MDU
  # ind is the index of missing data in distance matrix
  # P is the matrix that only contains missing data part and NA elsewhere
  temp <- function(x){
    res <- numeric(ncol(D))
    for(i in 1:(nrow(D))){
      for(j in (nrow(boundary):1)){
        if(D[i, x]<boundary[j, x]){
          res[i] = j
        } else {
          if(D[i, x]>boundary[nrow(boundary), x]){
            res[i] = nrow(boundary)+1
          } else stop
        }
      }
    }
    res
  }
  DR <- sapply(1:ncol(D), function(x)temp(x))
  P <- matrix(NA, nrow(DR), ncol(DR))
  for(i in 1:nrow(DR)){
    P[i, ind[i, ]] <- DR[i, ind[i, ]]
  }
  P
}

## prediction single

`pred.single <- function(ind, D, type){
  if(type == "full.rank"){
    DR <- t(apply(D, 1, rank))
    P <- matrix(NA, nrow(DR), ncol(DR))
    for(i in 1:nrow(DR)){
      P[i, ind[i, ]] <- DR[i, ind[i, ]]
    }
    P
  } else{
    r <- round((dim(D)[2])/5)*(c(1, 2, 3, 4))
    boundary <- t(apply(D, 1, function(x)sort(x)[r]))
    temp <- function(x){
      res <- numeric(ncol(D))
      for(i in 1:(ncol(D))){
        for(j in (ncol(boundary):1)){
          if(D[x, i]<boundary[x, j]||D[x, i]==boundary[x, j]){res[i] = j}
        } else{
          if(D[x, i]>boundary[x, ncol(boundary)]){res[i] = ncol(boundary)+1}
        } else stop
      }
    }
  }
}

```
DR <- t(sapply(1:nrow(D), function(x)temp(x)))
P <- matrix(NA, nrow(DR), ncol(DR))
for (i in 1:nrow(DR)) {
P[i, ind[i, ]] <- DR[i, ind[i, ]]
}
P
)`
Appendix C  Tables

Table 15: The Cross Table of $\sigma_s$

<table>
<thead>
<tr>
<th></th>
<th>Data size 1</th>
<th></th>
<th>Data size 2</th>
<th></th>
<th>Data size 1</th>
<th></th>
<th>Data size 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Error 1</td>
<td>Full Ranking</td>
<td>Partial Ranking 1</td>
<td>Partial Ranking 2</td>
<td>Error 2</td>
<td>Full Ranking</td>
<td>Partial Ranking 1</td>
<td>Partial Ranking 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Start 1</td>
<td>Missing 1</td>
<td>X</td>
<td>End</td>
<td>Start 1</td>
<td>Missing 1</td>
<td>End</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Start 2</td>
<td>Missing 1</td>
<td>X</td>
<td>Beginning</td>
<td>Start 2</td>
<td>Missing 1</td>
<td>X</td>
</tr>
</tbody>
</table>