Algorithms for Random Ranking Generation

Liqun Xu

A Major Report
in
The Department
of
Computer Science

Presented in Partial Fulfilment of the Requirements
for the Degree of Master of Computer Science at
Concordia University
Montreal, Quebec, Canada

September, 2000

0-612-54338-2

Abstract

Algorithms for Random Ranking Generation

Liqun Xu

Given a ranking of size n, most of the existing ranking models have relatively small numbers of parameters (around n, or less). Having a small number of parameters do help facilitate the application of a ranking model. But on the other hand, it restricts the capacity for the model to describe the innate structure of a ranking population.

In this report, we suggest a random ranking generator with $(n-1)^2$ parameters. The increased number of parameters enables the generator to simulate ranking populations with greater flexibility. We also suggest the use of a $n \times n$ probability matrix (P-matrix) as a device for specifying the targeted ranking population. In the P-matrix, each cell is the probability of an item being assigned a certain rank. We provide an algorithm that estimates, from a given P-matrix, the parameters for the generator. Numerical examples show that using the P-matrix based parameter estimation algorithm, the proposed generator provides better simulation to the targeted rank data.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Humans seem to be unable to avoid ranking things. Top Ten, Top Twenty, or even Top Hundred lists abound everywhere. We have the Funniest Home Videos, the Best Sellers' lists, the Richest Men, and, of course, the Most Wanted Criminals.

Rankings have very serious uses as well. Companies need to know what products consumers prefer. Social and political leaders need to know what the society values. Prospective students need to be assessed. Data consisting of rankings appear in psychology, educational testing, sociology, economics, and biology. Nonparametric statistical analysis was initially based mainly on ranks.

Ranking has always been an integral part of statistics, both in nonparametric analysis and in the analysis of people's judgement of objects. Our interest here is on the latter. We will discuss how to describe the populations of ranking data. Our work intends to use computing techniques to find a better way for generating ranking data.

A ranking $\pi = (i_1 \ldots i_n)$ of size n is a permutation of the first n positive integers. In applications, π may represent either a rank-sequence or an item-sequence. Accordingly, $i_k(k = 1, \ldots, n)$ may denote either the rank assigned to the k-th item or the item being assigned rank k.

Let us see an example. The Graduate Record Examination Board asked 98 college students to rank five words according to their strength of association with a target word [3]. For the target word 'song' the five choices were (1) 'score', (2) 'instrument', (3) 'solo', (4) 'benediction', and (5) 'suit'. The observed rankings and their frequencies are listed in Table 1.1.

As we all know, in statistics, the word population stands for a set of random data, and a model is a mathematical description of a population. Theoretically, a multinomial model on all permutations can specify any ranking population. However, this enumerative approach provides little insight into the population, and the large number of parameters ($n!$) is difficult to handle. It is necessary to find some way to specify a population with a tractable number of parameters [9].

Table 1.1: Observed Word Ranking Frequencies

Rankings	Frequencies
(3 2 1 4 5)	19
(3 1 2 4 5)	10
(1 3 2 4 5)	9
(3 2 4 1 5)	8
(1 2 3 4 5)	7
(3 2 1 5 4)	6
(2 3 1 4 5)	6
(3 2 4 5 1)	5
(2 1 3 4 5)	4
(3 1 4 2 5)	3
(3 2 5 4 1)	2
(3 4 2 1 5)	2
(2 3 4 1 5)	2
15 rankings	1 each
92 rankings	0 each

Compared with other forms of random data, ranking has probably the most structured format, and this makes the description of random ranking population difficult. Most existing ranking models have relatively small numbers of parameters, around n. These models are one-dimensional in nature. They assume, explicitly or implicitly, a dominant modal ranking and describe some relatively simple patterns in which the rankings of a population distribute around that modal ranking.

We try to go beyond that. We will see that the complexity introduced by our attempt cannot be handled by mathematical analysis tools. Our new description method depends on computer algorithms for its practical usage.

In this report, we briefly review the existing ranking theory in Chapter 2. We then proceed in Chapter 3 to elaborate a new $(n-1)^2$ parameter, multistage ranking model. We suggest that the $n \times n$ item-rank relative frequency matrix (the "P-matrix") be used as a device for summarising a set of rankings. For the proposed model we provide an algorithm that estimates the parameters from the P-matrix. Illustrative numerical examples are given in Chapter 4. In Chapter 5. the P-matrix is further discussed, and we show some special P-matrix patterns possessed by the some well-known ranking models. The proposed random ranking generation method is discussed in Chapter 6.

Chapter 2

Review of Ranking Theory

Given a set of ranking data, the first thing we can do is to measure how consistent are the rankings. Kendall's *tau* [8] and Spearman's *rho* [**14**] are designed as measures of the correlation between two rankings. Their probability distributions have been deliberated under the null assumption that all possible rankings have the same probability of being observed. So, we can test the significance of the observed coincidence between two rankings and determine how likely the observed coincidence occurs purely by chance.

For m rankings, Kendall's W, the coefficient of concordance, is defined as

$$W = \frac{12}{m^2(n^2-n)} \sum_{i=1}^{n} \left[S_i - \frac{m(n+1)}{2} \right]^2, \tag{2.1}$$

where S_i is the sum of all ranks item i gets. W varies between 0 and 1. When there is a perfect concordance, W equals 1; when rankings are "perfectly" random (all rankings have equal chance of being observed), W equals 0. To test the significance, the following approximation, based on Fisher's z-distribution, can be used.

$$z = \frac{1}{2} \log_e \frac{(m-1)W}{1-W}, \tag{2.2}$$

$$v_1 = n - 1 - \frac{2}{m}, \tag{2.3}$$

$$v_2 = (m-1)v_1, \tag{2.4}$$

where v_1, v_2 are the degrees of freedom. When n is larger than 7, we have a Chi-square approximation

$$\chi_r^2 = m(n-1)W, \tag{2.5}$$

where $r = n - 1$ is the degree of freedom. The average of Spearman's *rho* between all possible ranking pairs can also be used to measure the concordance.

Now, let us see some examples of the ranking models. For a brief survey on ranking models, see [**3**]. Probability ranking models so far proposed fall into four categories – Thurstonian models, models induced by paired comparisons, distance-based models, and multistage models.

2.1 Thurstonian models

The Thurstonian models [15][12][4] extend Thurstone's theory of paired comparison to the full ordering of n items. Suppose that the discriminate processes corresponding to n items take the form $Z_1 + u_1, \ldots, Z_n + u_n$, where $Z_1, \ldots, Z_n$ are independent identically distributed continuous random variables, and $u_1, \ldots, u_n$ are some constants. Let $X_i = Z_i + u_i$; then the distribution of X_i has the form $F_i(x) = F(x - u_i)$, where F is some continuous distribution. Define the random ranking π by setting its k-th element equal to i_k, where X_{i_k} is ranked k among $\{X_1, \ldots, X_n\}$. Then, the ranking $\pi = (i_1 \ldots i_n)$ has the probability

$$P(\pi) = P(X_{i_1} < X_{i_2} < \ldots < X_{i_n}). \tag{2.6}$$

Thurstone's paired comparison theory discusses a special case of this model in which $n = 2$ and F is the normal distribution.

2.2 Multistage models

Multistage models split the ranking process into $n - 1$ stages. Starting with the full set of n items, at the first stage, one item is selected and assigned rank 1; at the second stage, another item is selected from the remaining items and assigned rank 2; and so on. The last remaining item is assigned rank n by default.

Fligner and Verducci [6] proposed a multistage model. At each stage, the probability of an item being chosen is related to a principal ranking π_0. Let $\{p(m, r) : m = 0, \ldots, n-r\}$ denote the fixed set of choice probabilities at stage $r, r = 1, \ldots, n-1$. These probabilities are assigned to the stimuli indexed in the remaining subset B_r by assessing the correctness of the choice made at stage r, with respect to π_0. Specifically, let $s_r = m$ if, at stage r, the $(m+1)$st best of the items in B_r (according to π_0) is selected. In fact, m may be thought of as the number of mistakes made at stage r. For example, if π_0 and π correspond to the orderings (3 1 2 4) and (3 4 2 1), respectively, then $s_1 = 0$ since the best item (item 3) is selected at the first stage; $s_2 = 2$ since the third best (item 4) of the three remaining items is selected at the second stage; and $s_3 = 1$ since the second best (item 2) of the two remaining items is selected at the last stage. Then $p(m, r) = p(s_r = m)$ and the model is a $C(n; 2)$ parameter model given by

$$p(\pi) = \prod_{r=1}^{k-1} p(s_r, r). \tag{2.7}$$

Notice that for any π, the corresponding vector $s = (s_1, \ldots, s_{n-1})$ is related to Kendall's *tau* between π and π_0 by

$$T(\pi, \pi_0) = \sum_{r=1}^{k-1} s_r. \tag{2.8}$$

Another multistage model is based on Luce's choice axiom and rank postulate [10]. Suppose that $T = \{1, \ldots, n\}$ is a set of items, and p_i is the choice probability that item i is selected as the best in T. Luce proved that if the choice probabilities satisfy the choice axiom then the existence theorem holds:

Theorem 1 *For any $B \subset T, i \in B$, the probability that i is selected as the best in B is*

$$p_B(i) = \frac{p_i}{\sum\limits_{j \in B} p_j}. \tag{2.9}$$

Luce's rank postulate supposes ranking is obtained by repeated selections of the best item and that the probability of the ranking $\pi = (i_1 \ldots i_n)$ is

$$P(\pi) = P_{T_1}(i_1)P_{T_2}(i_2)\ldots P_{T_{n-1}}(i_{n-1}), \tag{2.10}$$

where $T_1 = T, T_2 = T_1 - \{i_1\}, \ldots, T_{n-1} = T_{n-2} - \{i_{n-2}\}$. The selection probabilities in (2.10) are derived from (2.9), and they are based on a common set of choice probabilities. It has been found that in a Thurstonian model, if $F(x)$ is a distribution function of the double exponential type, then this model is equivalent to Luce model [16]. For Luce model, the ranking probability can be easily calculated. But for Thurstonian models using other distributions, such as the normal distribution, the calculation of the exact ranking probabilities may be difficult because it involves a multiple integral.

2.3 Models induced from paired comparisons

Babington Smith [1] suggested inducing a ranking model from a set of arbitrary paired comparison probabilities. This model has $n(n-1)/2$ parameters. For each pair of items $i < j$, let p_{ij} be the probability that item i is preferred to item j, that is $i \to j$. Imagine a tournament in which all possible paired comparisons are made independently. If the results contain no circular triads, like $(h \to i \to j \to h)$, then the tournament corresponds to a unique ranking π; otherwise the entire tournament is repeated until a unique ranking is obtained. The probability of resulting in the neat ranking π from the tournament is

$$p(\pi) = C \prod_{\{(i,j):\pi(i)<\pi(j)\}} p_{ij}, \tag{2.11}$$

where C is a constant whose value is chosen such that all $P(\pi)$ sum to 1. These probabilities form a ranking distribution. It is tedious to calculate the sum of $P(\pi)$ which covers all $n!$ rankings. This model has $n(n-1)/2$

parameters and is difficult to use for generating random rankings. To reduce the number of parameters in the Babington Smith model, Bradley and Terry [**2**] introduced the condition that the paired comparison probabilities have the form

$$p_{ij} = \frac{q_{ij}}{q_i + q_j} \tag{2.12}$$

for some nonnegative parameters $q_1, \ldots, q_n$. Substituting Bradley-Terry probabilities into the Babington Smith model [**11**] leads to the well-known Mallows-Bradley-Terry (MBT) model, for which the probability of the ranking $\pi = (i_1, \ldots, i_n)$ is

$$p(\pi) = C(\mathbf{q}) \prod_{r=1}^{n-1} (q_{i_r})^{n-r} \tag{2.13}$$

where $q = (q_1, \ldots, q_n)$ and $C(\mathbf{q})$ is chosen to make the probabilities sum to 1.

2.4 Distance-based models

Distance-based model was first suggested by Mallows [**11**]. This type of models are based on the assumption that there is a modal ranking π_0 in the population, and that rankings which are at the same distance from the modal ranking have the same probability. The two matrices Mallows used for the distance are

$$T(\pi,\pi_0) = \sum_{i<j} \mathbf{I}\{[\pi(i) - \pi(j)][\pi_0(i) - \pi_0(j)] < 0\}, \tag{2.14}$$

where $\pi(i)$ and $\pi_0(i)$ are the ith elements in the rankings, $\mathbf{I}(.)$ is the indicator function: $\mathbf{I}(A)$ is 1, if the event A occurs, 0 otherwise; and

$$R^2(\pi,\pi_0) = \sum_{i=1}^{n} [\pi(i) - \pi_0(i)]^2. \tag{2.15}$$

Metrics T and R^2 are related to the concordance measures, Kendall's *tau* and Spearman's *rho*, respectively. Mallows' model has the form

$$p(\pi|\theta,\phi,\pi_0) = C(\theta,\phi)\theta^{R^2(\pi,\pi_0)}\phi^{T(\pi,\pi_0)}, \tag{2.16}$$

where $\theta, \phi > 0$, and $C(\theta,\phi)$ is a constant chosen to make the probabilities sum to 1. This model has only two parameters. If ϕ is equal to 1, or θ is equal to 1, then we obtain, respectively, the famous θ-model or the ϕ-model. The ϕ-model can be written as

$$p(\pi|\lambda,\pi_0) = C(\lambda)exp[-\lambda T(\pi,\pi_0)]. \tag{2.17}$$

Fligner and Verducci [**6**] found that the ϕ-model is a special case of a multistage model which they called "ϕ-component". Diaconis [**5**] generalised (2.17) into a class of distance-based models

$$p(\pi|\lambda,\pi_0) = C(\lambda)exp[-\lambda d(\pi,\pi_0)], \tag{2.18}$$

where d is an arbitrary distance metric defined on all permutations. In addition to Kendall's *tau* and Spearman's *rho*, Diaconis considered the following metrics as among the most frequently used in applications: Spearman's footrule metric

$$F(\pi, \pi_0) = \sum_i |\pi(i) - \pi_0(i)|, \tag{2.19}$$

generalised Spearman's footrule metric

$$F_h(\pi, \pi_0) = \sum_i |h(\pi(i)) - h(\pi_0(i))|, \tag{2.20}$$

where $h : \{1, \ldots, k\} \to R$ is a strictly increasing function that re-scales the ranks: and Hamming's metric

$$H(\pi, \pi_0) = \sum_i \mathbf{I}\{\pi(i) \neq \pi_0(i)\}. \tag{2.21}$$

In the above, the Luce's model, the Thurstonian models, and the distance-based models either explicitly postulate a modal ranking or imply its existence. All the models that found practical use have relatively small numbers of parameters. Having a small number of parameters help facilitating the application. But on the other hand, it restricts the model's capacity to describe a ranking population's innate structures. It would be difficult to use these models to generate random ranking for a population, if the population does not have a clear modal ranking.

Chapter 3

Random Ranking Generation

3.1 The description of ranking population

To generate random rankings we have to describe what kind of ranking population is desired. A natural way to summarise a ranking data set is to use a relative frequency matrix $\mathbf{P} = (p_{ij})$, where p_{ij} is the relative frequency that number i is assigned to the jth position. So, each column of $\mathbf{P}$ stands for a position and each row for a number, and

$$\sum_{i=1}^{n} p_{ij} = \sum_{j=1}^{n} p_{ij} = 1. \tag{3.1}$$

Let us call this matrix the P-matrix. For a finite ranking data set, the P-matrix provides a description of the distribution of the numbers over the positions in the rankings. If we view the ranking set as a population, the element p_{ij} is the probability that number i is assigned to position j in this population. For an infinite ranking data set, we can replace the element of the P-matrix with the probabilities that number i be assigned to position j, and thus use the P-matrix to describe the population.

In general, with probabilities as its elements, P-matrix can describe the ranking population we want. For a ranking data set or a ranking population, the P-matrix provides an overview of the distribution of the numbers over the positions. Given a P-matrix, our proposed random ranking generator will produce random rankings in such a way that the number-position relative frequencies of the generated rankings are as close as possible to the corresponding elements of the given P-matrix.

3.2 The ranking generator

The proposed ranking generator uses a multistage procedure. At each stage, the random selection of the number is controlled by n nonnegative weights. Putting the weights of stage j in a vector we have

$$\mathrm{C_j} = \begin{bmatrix} c_{1j} \\ c_{2j} \\ \vdots \\ c_{nj} \end{bmatrix}, \tag{3.2}$$

where $j = 1, 2, \ldots, n-1$, and $c_{ij} \geq 0$ is the weight of number i at the jth stage. The weights of each stage are scaled to have a sum equals to 1. We call the $n \times (n-1)$ matrix

$$\mathbf{C} = (\mathbf{C_1 C_2 \ldots C_{n-1}}) \tag{3.3}$$

the control matrix of the generator, or the C-matrix,

Algorithm 1 describes the process of random ranking generation. In which, j is the current stage, that is, we are selecting a number for the jth position in the ranking; B is the set of the numbers eligible for the current stage; c_{Bj} is the set $\{c_{ij}\}$, where $i \in B$. At the starting point, $j = 1$ and B contains all n numbers.

Algorithm 1

1. Randomly select a number from B. The probability of number i being chosen is:

   $$\frac{c_{ij}}{\sum\limits_{r \in B} c_{rj}}.$$

   Say, k is selected, assign number k to the jth position, remove k from B, and increase j by 1.

2. IF $j < n$, repeat Step 1; ELSE assign the number in B to the nth position and STOP.

It is easy to see that for any two numbers, say l and k, as long as they are both eligible at the jth stage, the ratio of the probability that number l being selected and the probability that number k being selected is always equal to c_{lj}/c_{kj}, no matter what are the other eligible numbers.

Given the C-matrix, for the ranking population specified by Algorithm 1, we have the follows.

Lemma 1. *For a partial ranking $\pi = (r_1 r_2 \ldots r_j)$, where $j \leq n$, the probability of π is*

$$p(\pi) = \prod_{k=1}^{j} \frac{c_{r_k,k}}{1 - \sum_{m=1}^{k-1} c_{r_m,k}}. \tag{3.4}$$

We know that $p(\pi)$ equals to the probability that r_1 is selected at stage 1 and r_2 is selected at stage 2, ..., and r_j is selected at stage j. From Algorithm 1, it is easy to see that the probability of r_1 being selected at the first stage is $c_{r_1,1}$. Then, at the second stage, the probability of r_2 being selected from the remaining set $\{1, 2, \ldots, n\} - \{r_1\}$ is $c_{r_2,2}/(1 - c_{r_1,2})$, the element $c_{r_1,2}$ is removed because the number r_1 is no longer eligible. Similarly, at the third stage, the probability of r_3 being selected from the remaining set $\{1, 2, \ldots, n\} - \{r_1, r_2\}$ is $c_{r_3,3}/(1 - c_{r_1,3} - c_{r_2,3})$, and so on. Thus, we have

(3.4).

Following similar arguments, it is not difficult to see the probability that number i is assigned to the first position is c_{i1}. That is, $p_{i1} = c_{i1}$. The probability that number i is assigned to the second position is

$$p_{i2} = \sum_{k \neq i} \frac{c_{k1} \times c_{i2}}{1 - c_{k2}}. \tag{3.5}$$

The probability that number i is assigned to the third position is

$$p_{i3} = \sum_{k \neq i, l \neq i, k \neq l} \frac{c_{k1} \times c_{l2} \times c_{i3}}{(1 - c_{k2})(1 - c_{k3} - c_{l3})}. \tag{3.6}$$

In general, we have

Lemma 2. *The probability that number i is assigned to the jth position is*

$$p_{ij} = \sum_{(r_1 \ldots r_j)} \left[\prod_{k=1}^{j} \frac{c_{r_k,k}}{1 - \sum_{m=1}^{k-1} c_{r_m,k}} \right], \tag{3.7}$$

where the summation extends over all possible $(r_1 \ldots r_{j-1} r_j)$, in which $(r_1 \ldots r_{j-1})$ is a permutation of the numbers from $\{1, 2, \ldots, i-1, i+1, \ldots, n\}$, and $r_j = i$.

Now, the problem is to find a C-matrix with which the generator produces a ranking population that has the desired P-matrix, or a similar one. For convenience, if **P** is the P-matrix of the ranking population induced from the C-matrix **C**, then we say "**C** produces **P**".

It is easy to see that there is not always a C-matrix which produces a given P-matrix **P**. For example, consider P-matrix

$$\mathbf{P} = \begin{bmatrix} p_{11} & p_{12} & 0 \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix}, \tag{3.8}$$

where $0 < p_{11} < 1, 0 < p_{22}+p_{23}$ and $p_{12} = 1-p_{11}$. Suppose that the C-matrix

$$\mathbf{C} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \\ c_{31} & c_{32} \end{bmatrix} \tag{3.9}$$

produces **P'** $= (p'_{ij})$. If **P'=P** then, on the one hand, c_{12} must be 1 so that $p'_{12} = p_{21} + p_{31} = 1 - p_{11} = p_{12}$. Therefore, c_{22} and c_{32} must both be 0. On the other hand, since $0 < p_{22} + p_{32}$, the elements c_{22} and c_{32} cannot both be 0. Therefore, there is no C-matrix that produces **P**.

However, it is worth considering whether the following is true.

Conjecture 1. *If all the elements of a P-matrix **P** are larger than zero, then there must be a C-matrix **C** such that **C** produces **P**.*

We shall see the significance of this Conjecture later. On the other hand, we can prove

Theorem 2 *Different C-matrices will produce different P-matrices.*

Proof

In fact, suppose **C** $= (c_{ij})$ and **C'** $= (c'_{ij})$ are two C-matrices and **P** $=$ (p_{ij}) and **P'** $= (p'_{ij})$ are the two corresponding P-matrices. If the first columns of the two C-matrices are not equal then the first columns of the two P-matrices are not equal. Assume that the first $j-1$ columns of the two C-matrices are the same, and that their jth columns are not equal, that is

$$\begin{bmatrix} c_{1j} \\ c_{2j} \\ \vdots \\ c_{nj} \end{bmatrix} \neq \begin{bmatrix} c'_{1j} \\ c'_{2j} \\ \vdots \\ c'_{nj} \end{bmatrix}, \tag{3.10}$$

then, we can always find an i such that for all $k \neq i$,

$$\frac{c'_{ij}}{c_{ij}} \geq \frac{c'_{kj}}{c_{kj}}. \tag{3.11}$$

Let $c'_{ij} = (1+\theta_i)c_{ij}$ and $c'_{kj} = (1+\theta_k)c_{kj}$ then

$$\frac{1+\theta_k}{1+\theta_i} \leq 1. \tag{3.12}$$

Now, let $(r_1 \dots r_{j-1})$ be any permutation of numbers from$\{1, 2, \dots, i-1, i+1, \dots, n\}$; let $r_j = i$ and

$$\{r_j, r_{j+1}, \dots, r_n\} = \{1, \dots, n\} - \{r_1, \dots, r_{j-1}\}. \tag{3.13}$$

Let $P\{(r_1 \dots r_{j-1})\}$ denote the probability that $r_1, \dots, r_{j-1}$ are assigned to the first $j-1$ positions, that is

$$P((r_1, r_2, \ldots, r_{j-1})) = \prod_{k=1}^{j-1} \frac{c_{r_k,k}}{1 - \sum_{m=1}^{k-1} c_{r_m,k}}. \tag{3.14}$$

Then, the (ij) elements in the corresponding P-matrices are

$$p_{ij} = \sum_{(r_1,\ldots,r_{j-1})} P((r_1, r_2, \ldots, r_{j-1})) \frac{c_{r_j,j}}{c_{r_j,j} + c_{r_{j+1},j} + \ldots + c_{r_n,j}}, \tag{3.15}$$

and

$$\begin{aligned} p'_{ij} &= \sum_{(r_1,\ldots,r_{j-1})} P((r_1, r_2, \ldots, r_{j-1})) \frac{c'_{r_j,j}}{c'_{r_j,j} + c'_{r_{j+1},j} + \ldots + c'_{r_n,j}} \qquad (3.16) \\ &= \sum_{(r_1,\ldots,r_{j-1})} P((r_1, r_2, \ldots, r_{j-1})) \frac{c_{r_j,j}}{\sum_{k=j}^{n} c_{r_k,j} \left(\frac{1+\theta_{r_k}}{1+\theta_{r_j}} \right)}. \end{aligned}$$

If the jth columns of the two C-matrices are different, as shown in (3.10), then there must exist some k such that the strict inequalities in (3.11) and (3.12) become true. Therefore, from (3.15) and (3.16), we have

$$p'_{ij} > p_{ij}. \tag{3.17}$$

Thus, we have proved that there is at most one C-matrix that can produce a given P-matrix.

3.3 The algorithm for parameter estimation

Given a P-matrix, the corresponding C-matrix is determined by the equation system represented by (3.7). This is a multivariate high-degree system. Mathematical theories currently available cannot solve this type of system. The C-matrix has to be estimated through an iterative algorithm. The goal is to find the C-matrix that produces a P-matrix as similar as possible to the target P-matrix.

In the following algorithm, **C** is the C-matrix being estimated; **T** is the target P-matrix; **P** is the P-matrix produced by **C**; j is the current column number; $s \geq 0$ is the precision criterion; *column_error* is the sum of squares of the differences between the corresponding elements of the jth columns of **T** and **P**; *prec_obtain* is a variable keeping the previously obtained *column_error*.

Algorithm 2

1. Copy **T** to **C**; let $j = 2$

2. Calculate the jth column of **P** from the first j columns of **C**

3. Calculate the *column_error* between the jth columns of **T** and **P**

```
4. IF ($column\_error < s$ OR $column\_error > prec\_obtain$) THEN
        $j = j + 1$
        IF $j = n$ THEN calculate the $n$th column of $\mathbf{P}$ and STOP
        ELSE GOTO 2
5. ELSE let $prec\_obtain = column\_error$ adjust the elements in the
        $j$th column of $\mathbf{C}$, GOTO 2
```

In Step 1, we use **T** as the initial value of **C**. Step 2 calculates the elements of **P**. This can be implemented by a recursive procedure.

In Step 5, the strategy used for adjusting the elements of **C**'s jth column is rather simple. If the element p_{ij} of **P** is larger than the corresponding element t_{ij} of **T**, then we decrease the c_{ij} of **C**; if p_{ij} is smaller than t_{ij}, then we increase c_{ij}. Specifically,

$$c'_{ij} = c_{ij} + \alpha(t_{ij} - p_{ij}), \tag{3.18}$$

where $0 < \alpha < 1$ is a tuning ratio. This ratio should be small enough so that $0 < c'_{ij}$. Apply (3.18) to (3.16), we have

$$p'_{ij} = \sum_{(r_1,\dots,r_{j-1})} P((r_1, r_2, \dots, r_{j-1})) \frac{c'_{r_j,j}}{c'_{r_j,j} + c'_{r_{j+1},j} + \dots + c'_{r_n,j}} \tag{3.19}$$

$$= \sum_{(r_1,\ldots,r_{j-1})} P((r_1,r_2,\ldots,r_{j-1}))\frac{c_{r_j,j}+\alpha(t_{r_j,j}-p_{r_j,j})}{\sum_{k=j}^{n}[c_{r_k,j}+\alpha(t_{r_k,j}-p_{r_k,j})]}.$$

The adjusting process stops whenever the current adjustment leads to an increase of the column-error, or when the required precision is reached.

This algorithm has been implemented in the C programming language. A pseudo code program showing the implementation details can be found in the Appendix.

Chapter 4

Examples

Now, let us demonstrate the application of our algorithms in two cases.

4.1 A Thurstonian ranking model

Define a Thurstonian ranking model: Let X_t be normal variates with distributions $N_t = N(0.3t, 1)$, where $t = 1, \ldots, 10$. A random ranking π is defined by setting the kth element $\pi(k)$ equal to i_k, where X_k is ranked i_k among $\{X_1, \ldots, X_{10}\}$. Here, the probability $P(\pi) = P\{X_{i_1} < \ldots < X_{i_{10}}\}$ and its exact value is difficult to obtain.

Using 1000 random rankings generated by the Thurstonian model we obtained the following 10×10 target P-matrix. From it, we estimated the C-matrix, with $s = 10^{-4}$, and we calculated the corresponding P-matrix from this C-matrix. The target P-matrix is

$$\begin{bmatrix}
0.355 & 0.200 & 0.185 & 0.130 & 0.035 & 0.045 & 0.045 & 0.000 & 0.005 & 0.000 \\
0.295 & 0.210 & 0.195 & 0.105 & 0.060 & 0.050 & 0.035 & 0.030 & 0.010 & 0.010 \\
0.095 & 0.250 & 0.185 & 0.150 & 0.095 & 0.075 & 0.070 & 0.045 & 0.025 & 0.010 \\
0.140 & 0.110 & 0.180 & 0.105 & 0.100 & 0.180 & 0.095 & 0.050 & 0.015 & 0.025 \\
0.065 & 0.110 & 0.105 & 0.135 & 0.155 & 0.150 & 0.105 & 0.100 & 0.035 & 0.040 \\
0.030 & 0.040 & 0.045 & 0.120 & 0.180 & 0.115 & 0.145 & 0.145 & 0.130 & 0.050 \\
0.015 & 0.035 & 0.060 & 0.080 & 0.145 & 0.095 & 0.155 & 0.180 & 0.140 & 0.095 \\
0.005 & 0.025 & 0.025 & 0.085 & 0.075 & 0.130 & 0.145 & 0.140 & 0.190 & 0.180 \\
0.000 & 0.010 & 0.020 & 0.045 & 0.075 & 0.100 & 0.135 & 0.190 & 0.225 & 0.200
\end{bmatrix}.$$

The estimated C-matrix is

$$\begin{bmatrix}
0.355 & 0.251 & 0.271 & 0.303 & 0.117 & 0.163 & 0.301 & 0.000 & 0.026 & 0.000 \\
0.295 & 0.240 & 0.255 & 0.183 & 0.160 & 0.137 & 0.100 & 0.150 & 0.057 & 0.010 \\
0.095 & 0.220 & 0.167 & 0.159 & 0.160 & 0.125 & 0.135 & 0.177 & 0.164 & 0.010 \\
0.140 & 0.100 & 0.140 & 0.082 & 0.105 & 0.220 & 0.157 & 0.171 & 0.060 & 0.025 \\
0.065 & 0.093 & 0.073 & 0.085 & 0.133 & 0.127 & 0.087 & 0.173 & 0.088 & 0.040 \\
0.030 & 0.033 & 0.028 & 0.063 & 0.116 & 0.064 & 0.065 & 0.098 & 0.232 & 0.050 \\
0.015 & 0.028 & 0.037 & 0.041 & 0.087 & 0.046 & 0.056 & 0.092 & 0.128 & 0.095 \\
0.005 & 0.020 & 0.015 & 0.042 & 0.042 & 0.055 & 0.045 & 0.050 & 0.090 & 0.180 \\
0.000 & 0.008 & 0.012 & 0.022 & 0.039 & 0.039 & 0.037 & 0.059 & 0.095 & 0.200 \\
0.000 & 0.008 & 0.000 & 0.021 & 0.041 & 0.023 & 0.018 & 0.031 & 0.061 & 0.390
\end{bmatrix}.$$

The obtained P-matrix from the above C-matrix is

$$\begin{bmatrix}
0.355 & 0.198 & 0.182 & 0.127 & 0.033 & 0.042 & 0.042 & 0.000 & 0.005 & 0.015 \\
0.295 & 0.210 & 0.195 & 0.106 & 0.059 & 0.050 & 0.035 & 0.028 & 0.009 & 0.014 \\
0.095 & 0.251 & 0.186 & 0.151 & 0.096 & 0.076 & 0.071 & 0.044 & 0.020 & 0.011 \\
0.140 & 0.110 & 0.181 & 0.105 & 0.101 & 0.181 & 0.096 & 0.050 & 0.015 & 0.021 \\
0.065 & 0.110 & 0.105 & 0.135 & 0.156 & 0.150 & 0.105 & 0.102 & 0.036 & 0.035 \\
0.030 & 0.040 & 0.045 & 0.120 & 0.181 & 0.115 & 0.145 & 0.146 & 0.132 & 0.046 \\
0.015 & 0.035 & 0.060 & 0.080 & 0.145 & 0.095 & 0.155 & 0.181 & 0.141 & 0.092 \\
0.005 & 0.025 & 0.025 & 0.085 & 0.075 & 0.130 & 0.145 & 0.140 & 0.191 & 0.178 \\
0.000 & 0.010 & 0.020 & 0.045 & 0.075 & 0.100 & 0.135 & 0.190 & 0.226 & 0.198 \\
0.000 & 0.010 & 0.000 & 0.045 & 0.080 & 0.060 & 0.070 & 0.120 & 0.226 & 0.389
\end{bmatrix}.$$

Using this C-matrix with Algorithm 1 to simulate the Thurstonian rankings, we reproduced the exact P-matrix and the probabilities of individual rankings are easy to calculate.

4.2 The GRE word rankings

Consider again the 98 word rankings example from The Graduate Record Examination Board. As mentioned in Chapter 1, in this case, 98 college students were asked to rank five words according to their strength of association with the target word 'song'. The five given words are (1) 'score', (2) 'instrument', (3) 'solo', (4) 'benediction', and (5) 'suit'. The observed rankings, their frequencies, and the fitted frequencies given by various models are listed in Table 4.1. The fitted frequencies given by our new model are in the last column of Table 4.1. The P-matrix of this ranking sample (a relative frequency matrix) is

$$\mathbf{F} = \begin{bmatrix} .204 & .204 & .357 & .133 & .102 \\ .163 & .510 & .245 & .061 & .020 \\ .602 & .224 & .143 & .031 & .000 \\ .031 & .051 & .214 & .582 & .122 \\ .000 & .010 & .041 & .194 & .755 \end{bmatrix}. \qquad (4.1)$$

Table 4.1: Observed and Fitted Frequencies

				Six Models				
	Obs.			Fitted Freq.				Fitted
Rankings	Freq.	Luce	MBT	ϕ	F	F_h	ϕ-Comp	Freq.
(3 2 1 4 5)	19	16.1	14.4	21.6	25.2	24.6	22.7	21.6
(3 1 2 4 5)	10	9.6	10.5	7.5	7.1	9.8	7.31	0.4
(1 3 2 4 5)	9	4.9	4.6	2.6	2.0	3.6	4.0	6.5
(3 2 4 1 5)	8	7.3	7.5	7.5	7.1	4.9	6.4	7.4
(1 2 3 4 5)	7	2.6	2.7	0.9	2.0	3.6	1.3	6.6
(3 2 1 5 4)	6	4.5	5.2	7.5	7.1	5.0	6.2	4.1
(2 3 1 4 5)	6	9.7	8.5	7.5	7.1	8.9	9.6	5.9
(3 2 4 5 1)	5	0.8	1.4	2.6	2.0	1.0	1.8	4.3
(2 1 3 4 5)	4	3.0	3.7	2.6	2.0	3.6	3.1	2.9
(3 1 4 2 5)	3	2.3	4.1	2.6	2.0	1.9	2.1	2.2
(3 2 5 4 1)	2	0.6	0.5	0.9	2.0	1.0	0.5	1.6
(3 4 2 1 5)	2	3.5	2.9	2.6	2.0	1.9	2.3	1.4
(2 3 4 1 5)	2	4.4	4.4	2.6	2.0	1.8	2.7	2.0
15 rankings	1 each	14.3	17.8	14.3	12.7	11.9	12.8	11.3
92 rankings	0 each	14.4	9.8	14.7	15.7	14.5	15.2	9.8

Using **F** as the target matrix, the C-matrix is estimated, with $s = 10^{-3}$, to be

$$\mathbf{C} = \begin{bmatrix} .204 & .155 & .154 & .091 \\ .163 & .396 & .331 & .272 \\ .602 & .408 & .433 & .302 \\ .031 & .033 & .070 & .282 \\ .000 & .006 & .013 & .054 \end{bmatrix}, \tag{4.2}$$

and the corresponding P-matrix is

$$\mathbf{P} = \begin{bmatrix} .204 & .204 & .357 & .133 & .101 \\ .163 & .512 & .247 & .061 & .017 \\ .602 & .222 & .141 & .028 & .007 \\ .031 & .051 & .214 & .584 & .129 \\ .000 & .010 & .041 & .194 & .755 \end{bmatrix}. \tag{4.3}$$

It is noted that rankings (1 3 2 4 5) and (1 2 3 4 5) are most adequately accounted for by the proposed model. Other models leave unexplained the pattern that item 1 tends to be ranked first when items 4 and 5 are ranked fourth and fifth. That is what we mean by saying that the new model goes beyond the one-dimensional model category.

The given likelihood estimates by Luce model [3] are as follows: $p_1 = .127, p_2 = .257, p_3 = .552, p_4 = .050$, and $p_5 = .014$. The corresponding P-matrix is

$$\mathbf{P} = \begin{bmatrix} .127 & .209 & .381 & .239 & .044 \\ .257 & .371 & .269 & .093 & .010 \\ .552 & .308 & .117 & .022 & .001 \\ .050 & .087 & .180 & .489 & .194 \\ .014 & .025 & .055 & .157 & .751 \end{bmatrix}. \tag{4.4}$$

Obviously, (4.3) looks more similar to (4.1) than (4.4) does.

To measure the fit between the expected and the observed ranking frequencies for the models listed in Table 4.1, we calculate Pearson's metric

$$X^2 = \sum_{i=1}^{k} \frac{(n_i - o_i)^2}{n_i}, \tag{4.5}$$

where n_i is the expected frequency in category i, o_i is the observed frequency, and k is the number of categories. We use the 15 ranking categories given in Table 4.1. The results are listed in Table 4.2.

With 16 parameters, the proposed model seemingly improved the overall

Table 4.2: Measures of Fitting

Model	X^2
Luce	55.644
MBT	39.347
ϕ	78.264
F	63.281
F_h	48.712
ϕ -Com	61.235
new	14.436

fit between the expected and the observed frequencies in this case.

Chapter 5

More Discussion on P-matrix

Two features of the P-matrix are worth further discussion.

5.1 P-matrix and ranking indices

First, the P-matrix summarises information in the given rankings, and important statistics of a ranking data set can be calculated from it. For example, the rank concordance measures, like Kendall's W, can be calculated from the P-matrix. In fact,

$$W = \frac{12}{m^2(n^3 - n)} \sum_{i=1}^{n} \left[S_i - \frac{1}{2} mn(n+1) \right]^2, \qquad (5.1)$$

where m is the number of rankings and S_i is the sum of all ranks assigned to item i. Given the P-matrix, we have

$$W = \frac{12}{m^2(n^3 - n)} \sum_{i=1}^{n} \left[\sum_{j=1}^{n} jmp_{ij} - \frac{1}{2} mn(n+1) \right]^2 \qquad (5.2)$$

$$= \frac{12}{(n^3 - n)} \sum_{i=1} n \left[\sum_{j=1}^{n} jp_{ij} - \frac{1}{2} n(n+1) \right]^2 .$$

The other two ranking concordance measures are the average of Spearman's ρ_{av} [14] and Friedman's χ_r^2 . They can be written as

$$\rho_{av} = \frac{mW - 1}{m - 1} \tag{5.3}$$

and

$$\chi_r^2 = Wm(n-1). \tag{5.4}$$

From (5.1), these measures can also be calculated from the P-matrix.

To measure the agreement of two ranking data sets S and T, we may use Schucany's [13]

$$\pounds = \sum_{i=i}^{n} S_i T_i, \tag{5.5}$$

where S_i and T_i are the sums of ranks for item i in data set S and data set T, respectively. In terms of the P-matrix,

$$\pounds = \sum_{i=1}^{n} S_i T_i = \sum_{i=1}^{n} \left[\left(\sum_{j=1}^{n} jmp_{ij}^{s} \right) \left(\sum_{j=1}^{n} jmp_{ij}^{t} \right) \right] . \tag{5.6}$$

5.2 P-matrix patterns

Secondly, the P-matrix of a ranking data set may reveal the pattern of the distributions of ranks over items. Let us consider the P-matrices of the distance-based models and the Thurstonian models. Many distance-based models use distance metrics which have a symmetric property. For them, we have

Theorem 3 *Given a distance-based ranking model, if for every ranking $\pi = (\pi_1\pi_2 \dots \pi_n)$*

$$d(\pi, \pi_0) = d(r, \pi_0), \tag{5.7}$$

where $r = (r_1r_2 \dots r_n) = (\theta_{\pi_n}\theta_{\pi_{n-1}} \dots \theta_{\pi_1})$ and $\theta_i = n+1-i$, then in the corresponding P-matrix of this model

$$p_{ij} = p_{n+1-i,\ n+1-j}. \tag{5.8}$$

Proof

Consider any ranking $\pi = (\pi_1\pi_2 \dots \pi_n)$ that contributes to the p_{ij} element of the P-matrix, that is the probability of "item i being assigned rank j". When the ith element in π is $\pi_i = j$, in the corresponding ranking $r = (\theta_{\pi_n}\theta_{\pi_{n-1}} \dots \theta_{\pi_1})$, the $(n+1-i)$th rank is

$$\theta_\pi = n + 1 - \pi_i = n + 1 - j. \tag{5.9}$$

Hence ranking r contributes to the $p_{n+1-i,\ n+1-j}$ element of the P-matrix. Since $d(\pi, \pi_0) = d(r, \pi_0)$, by (2.18), ranking π and r have the same probability in this model. Thus, π's contribution to p_{ij} and r's contribution to $p_{n+1-i,\ n+1-j}$ are equal. Since (π, r) is an one-to-one mapping (e.g., for $\pi = (1\ 3\ 2\ 4\ 5)$ the corresponding $r = (1\ 2\ 4\ 3\ 5)$), we have $p_{ij} = p_{n+1-i,\ n+1-j}$.

Equation (5.8) indicates a symmetric structure in the P-matrix. Many distance-based models have this property. Without loss of generality, let $\pi_0 = (1\ 2 \ldots n)$. We have the follows:

Kendall's *tau* metric

$$\begin{aligned}
T(\pi, \pi_0) &= \sum_{i<j} \mathbf{I}\{(\pi_i - \pi_j)(i-j) < 0\} \\
&= \sum_{i<j} \mathbf{I}\{[(n+1-\pi_j) - (n+1-\pi_i)](i-j) < 0\} \\
&= \sum_{i<j} \mathbf{I}\{(\theta_{\pi_j} - \theta_{\pi_i})(i-j) < 0\} \\
&= \sum_{i<j} \mathbf{I}\{(r_{n+1-j} - r_{n+1-i})[(n+1-j) - (n+1-i)] < 0\} \\
&= \sum_{i<j} \mathbf{I}\{(r_i - r_j)(i-j) < 0\} \\
&= T(r, \pi_0).
\end{aligned}$$

Spearman's *rho* metric

$$\begin{aligned}
R(\pi, \pi_0) &= \left(\sum_i (\pi_i - i)^2\right)^2 \\
&= \left(\sum_i (r_{n+1-i} - (n+1-i))^2\right)^2 \\
&= R(r, \pi_0).
\end{aligned}$$

Spearman's footrule metric

$$\begin{aligned} F(\pi, \pi_0) &= \sum_i |\pi_i - i| \\ &= \sum_i |r_{n+1-i} - (n+1-i)| \\ &= F(r, \pi_0). \end{aligned}$$

It is easy to see that the Hamming metric also meets condition (5.7), although the generalised Spearman's footrule metric does not.

Now, let us consider the Thurstonian models. Without loss of generality, we assume that the shifts in the model definition are in the order $u_1 < \ldots < u_n$. Let us use the Luce model example in Section 4, but with the items reordered. For the model with choice probabilities (.552 .257 .127 .050 .014), the P-matrix is found to be

$$P = \begin{bmatrix} .552 & .308 & .117 & .022 & .001 \\ .257 & .371 & .269 & .093 & .010 \\ .127 & .209 & .381 & .239 & .044 \\ .050 & .087 & .180 & .489 & .194 \\ .014 & .025 & .053 & .157 & .751 \end{bmatrix}. \tag{5.10}$$

In this P-matrix, a diagonal element is always the largest element in the corresponding row and column. The farther an element's position is from the diagonal, either along a row or column, the smaller is its magnitude. This pattern has a straightforward explanation.

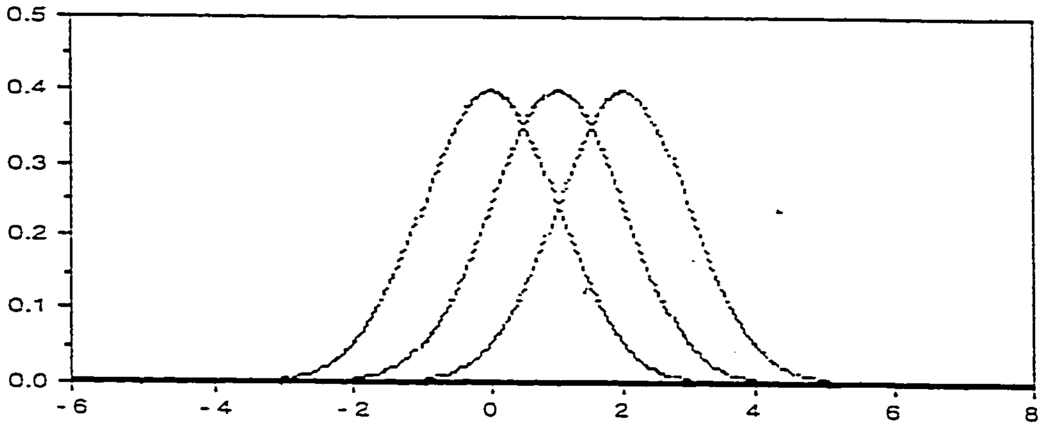

Figure 5.1: The Probability Distributions in a Thurstonian Model

Figure 5.1 shows the probability distributions of three random variables in a Thurstonian model. The density functions for these random variables are $F_i(x) = F(x - u_i) = N(u_i, 1)$, where $u_1 = 0, u_2 = 1, u_3 = 2$, and $N(u, 1)$ is the normal distribution.

From Figure 5.1, it is clear that the random variable X_1 is most likely to be ranked 1, less likely to be ranked 2, and even less likely to be ranked 3. Variable X_2 has the best chance to be ranked 2, and less chance to be ranked 1 or 3. Variable X_3 has the best chance to be ranked 3, less chance to be ranked 2 and even less chance to be ranked 1. The general rule is: variable X_i are more likely to be assigned ranks closer to i, and less likely to be assigned ranks larger or smaller than i. The smaller the difference between ranks j and i, the greater the chance that j is assigned to X_i. Rank i has the largest chance being assigned to X_i. This regularity is to be expected in all the Thurstonian models. The reflection of this regularity is the pattern seen in the P-matrix (5.10).

Chapter 6

Concluding remarks

Given the P-matrix, we can calculate important numerical indices for the ranking population, such as Kendall's rank correlation coefficient [8], the average of Spearman's χ [14], and Schucany's coefficient of concordance between two groups of rankings [13]. Therefore, it is meaningful to use a P-matrix to simulate a ranking population.

As shown in the first example, the proposed generator can be used to generate simulating rankings for another ranking model, so that the exact probabilities of the rankings can be calculated. As shown in the second example, though the parameter estimation for the generator is only aimed to reproduce the P-matrix, the resulting goodness of fit, in the sense of ranking frequencies, can compete with that of many well-known ranking models.

In the format, the proposed ranking generator is a generalisation of the

Luce's model [**10**], in which, the same set of weights is used at all stages.

To estimate a C-matrix for a case where the ranking size is 10 requires approximately 15 minutes on a personal computer. The computing time increases exponentially with the ranking size. More efficient algorithms is needed if this approach is to be used for cases of larger ranking size.

In general, there is not always a C-matrix producing a given P-matrix **P**. However, we speculate that Conjecture 1 is true. That is, when **P** has no zero element, there is a C-matrix **C** such that **C** produces **P**. If Conjecture 1 is true, then the proposed generator appears more likely to fit a wider range of ranking data sets than the Thurstonian models and distance-based models. Because, as we have shown in last section, ranking data described by these models possess special patterns in their P-matrices. It would be difficult for these models to describe a ranking data set whose P-matrix does not have those patterns. Further investigation of this issue is required.

Appendix

The estimation algorithm

In the follows, we present the details of the estimation algorithm with pseudo code. Procedure 'Main' shows the main body of Algorithm 2. Procedure 'Calc' calculates the expected probability for number i being assigned to position j.

```
Main {

//---input---
int     rnk_sz;          // size of ranking
double  precision;       // desired precision of estimation
double  tm[ ][ ];        // target P-matrix

//---output---
double  cm[ ][ ];        // estimated C-matrix
double  pm[ ][ ];        // P-matrix induced from 'cm'
double  prec_obtain;     // precision reached

//--- variables  ---
int     i;               // row index
int     j;               // column index
int     k;               // loop index
int     r_rnk[ ];        // list of eligible numbers
int     u_rnk[ ];        // list of selected numbers
double  cm_keeper[ ];    // C-matrix column before latest adjustion
double  expct;           // sum of probabilities of partial rankings
double  error[ ];        // error btwn col. elements of pm and tm
double  err_sum;        // sum of positive errors for a column
double  adj_tmp[ ];     // temperary storage of adjusted cm element
double  t_ratio;        // tuning ratio

 begin

   cm[*][*] = tm[*][*];   // initialise the C-matrix
   pm[*][1] = tm[*][1];   // fill up the 1st column of the P-matrix

   for(j=2  to  rnk_sz-1) // estimation starts from the 2nd column
      prec_obtain=1;      // initialise the precision reached

try:;

      for(i=1  to  rnk_sz)    // every element of jth column of pm[ ]
         expct=0;             // initialising

         if(cm[i][j]>0)       // if tm[i][j] = 0, then cm[i][j] = 0

            for(k=1  to  i-1) // initialise eligible number list
               r_rnk[k]=k;
            endfor;

            for(k=i to rnk_sz-1)  // i has been placed at pos j,
               r_rnk[k]=k+1;      // no longer eligible for other pos
            endfor;

            Calc(i, j, 1, r_rnk, u_rnk, 1);  // calculate 'expct'
         endif;

         pm[i][j] = expct;   // probability of i placed at pos j
      endfor;

      // determine the error between the columns
      // of pm[*][j] and the tm[*][j]

      err_sum=0;

      for(i=1  to  rnk_sz)
```

```
        error[i]=pm[i][j]-tm[i][j];
        err_sum=err_sum+error[i]*error[i];
      endfor;

      // decide whether further adjustment is needed

      if(err_sum<precision)     // if precision requirement met
          goto next_col;
      endif;

      if(err_sum>prec_obtain)   // if error enlarged
          goto next_col;
      endif;

      prec_obtain=err_sum;

      // adjusting the current column of cm[ ]

      t_ratio=0.5;             // tuning ratio starts from 0.5

adj_try:;

      for(i=1  to  rnk_sz)

         if((adj_tmp[i]=cm[i][j]-error[i]*t_ratio)<0) // over adjusted
            t_ratio=t_ratio*0.4;          //  reduce the tuning ratio
            goto adj_try;
         endif;

      endfor;

      cm[*][j] = adj_tmp[*];    // newly adjusted C-matrix column
      goto try;                 // re-estimate the pm column

next_col:                       // go to the next pm column

  endfor;
 end
}
```

```
Calc {

//---input---
int    i;      // row number of the P-matrix element
int    j;      // column number of the P-matrix element
int    ck;     // the current position, ck<=j
int    rm[ ];  // list of eligible numbers
int    us[ ];  // list of selected numbers
double ppr;    // probability of a partial ranking

//---output---
//  Procedure Calc update variable 'expct' -
//  the sum of probabilities of partial rankings

//---local variables---
int     k, l;            // loop index
int     rmd[ ], usd[ ];  // lists of eligible and selected numbers
double  c_col_usd;       // weight sum of the selected numbers
double  pr;              // probability of a partial ranking

 begin

  if (ck < j)
     for(k=1  to  rnk_sz-ck)  // this loop and the recursive
                              // call below will result in all
               // size ck partial rankings (permutations made from
               // numbers of {1, 2, ..., i-1, i+1, ..., n})

      if(cm[rm[k]][ck]>0)      // weight > 0
        c_col_usd=0;

        for(l=1  to  <ck-1)   // for all previously selected numbers
           c_col_usd=c_col_usd+cm[us[l]][ck];
        endfor;

        pr=ppr*cm[rm[k]][ck]/(1-c_col_usd);
                            // calculate probability of current
```

```
                                // partial ranking (size ck).

        for(l=1 to k-1)         // delete rm[k] from eligible list rmd[]
           rmd[l]=rm[l];
        endfor;

        for(l=k  to  rnk_sz-1)
           rmd[l]=rm[l+1];
        endfor;

        for(l=1  to  ck-1)    // put rm[k] into selected list usd[]
           usd[l]=us[l];
        endfor;

        usd[ck]=rm[k];

        Calc(i, j, ck+1, rmd, usd, pr); // recursive calling
      endif;
    endfor;
  endif
  else                                          // ck equal to j
      c_col_usd=0;

      for(l=1  to  ck-1)
         c_col_usd=c_col_usd+cm[us[l]][ck];
      endfor;

      expct=expct+ppr*cm[i][ck]/(1-c_col_usd);
                // add the probability of a size j partial
                // ranking to (i,j) element of P
  endelse;
 end
}
```

Bibliography

[1] Babington-Smith, B. (1950). Discussion of Professor Ross' paper. *Journal of the Royal Statistical Society, Series B,* 12, 153–162.

[2] Bradley, R. A. and Terry, M. E. (1952). Rank analysis of incomplete block designs. *Biometrika* 39, 324–335.

[3] Critchlow, D. E., Fligner, M. A. and Verducci, J. S. (1991). Probability models on Rankings. *Journal of Mathematical Psychology* 35, 294–318.

[4] Daniels, H. E. (1950). Rank correlation and population models. *Journal of the Royal Statistical Society, series B,* 12, 171.

[5] Diaconis, P. (1988). *Group representations in probability and statistics.* Hayward: Institute of Mathematical Statistics.

[6] Fligner, M. A. and Verducci, J. S. (1986). Distance based ranking models. *Journal of the Royal Statistical Society, series B,* 48, No. 3, 359–369.

[7] Friedman, M (1940). A comparison of alternative tests of significance for the problem of m rankings.*Annuals of Mathematical Statistics,* 11, 86.

[8] Kendall, M. G. (1948). *Rank Correlation Methods.* London: Charles Griffin and Co.

[9] Kendall, M. G. (1950). Discussion on symposium on ranking methods. *Journal of the Royal Statistical Society, series B,* 12, 189.

[10] Luce, R. D. (1959). *Individual Choice Behavior.* New York: John Wiley.

[11] Mallows, C. L. (1957). Non-null ranking models. I. *Biometrika,* 44, 114–130.

[12] Mosteller, F. (1951). Remarks on the method of paired comparisons, I: The least squares solution assuming equal standard deviations and equal correlations. *Psychometrika,* 16, 3–9.

[13] Schucany, W. R. and Frawley, W. H. (1973). A rank test for two group concordance. *Psychometrika,* 38, No. 2, 249–258.

[14] Spearman, C. (1904). The proof and measurement of association between two things. *American Journal of Psychology,* 15, 88.

[15] Thurstone, L. L. (1927). A law of comparative judgement. *Psychological Review*, 34, 273–286.

[16] Yellott, J.I., Jr.(1977). The relationship between Luce's choice axiom, Thurstone's theory of comparative judgement, and the double exponential distribution. *Journal of Mathematical Psychology*, 15, 109–144.