

https://dspace.org/

Implementation Criteria for the Shotgun Stochastic Parameter Estimation Algorithm

Moodley, Devandren

2014-01-09
http://hdl.handle.net10539/13410

Downloaded from a DSpace repository. https://dspace.org/

Implementation Criteria for the Shotgun Stochastic Parameter Estimation Algorithm

Devandren Moodley

A research report submitted to the Faculty of Commerce, Law and Management, University of the Witwatersrand, in partial fulfilment of the requirements for the degree of Master of Business Administration

Johannesburg, 2012

ABSTRACT

The use of rank ordered data is widespread and commonplace in fields of research from marketing to biology and medicine. Complexity and the mathematical rigour required for the analysis of rank ordered data has resulted in the use of parametric methods for data analysis. There have been conflicting views on the validity of the results produced by parametric methods.

Developments in stochastic methods have allowed researchers the opportunity to bypass the complexity and rigour associated with rank ordered data analysis by trading accuracy for simplified solutions. The cost of computing has been the limiting factor for the use of these techniques. With the recent decline in computing cost and increasing accessibility more researchers are now turning their attention back to these alternatives.

Stacey (2006) introduced one such method known as the Shotgun Stochastic Parameter Estimation Algorithm, to determine the mean and variance of rank ordered data. This study determined the reliability of the algorithm by developing guidelines that ensured consistency in the results. This was achieved by observing the output as key variables were changed. Another important contribution of the algorithm was its ability to analyse partially ranked data. This research investigated the behaviour of the *SSPEA* with partially ranked data and quantified the error introduced including possible solutions to reduce the error.

The data used for the study was obtained from Grant (2010) for some test cases and simulated internally for others. Similar to the incorrect application of parametric methods to rank ordered data; this research proved that the algorithms true potential is realised by correct choice of simulation software and efficient implementation. A set of guidelines and key considerations is produced that will assist researchers utilising this algorithm to prevent potential pitfalls in the data analysis. The study has demonstrated that the algorithm can be used with confidence to analyse rank ordered data provided these guidelines are followed and considerations noted.

DECLARATION

I, Devandren Moodley, declare that this research report is my own work except as indicated in the references and acknowledgements. It is submitted in partial fulfilment of the requirements for the degree of Master of Business Administration in the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in this or any other university.

Devandren Moodley

Signed at ……SUNNINGHILL……

On the ……15th…….. day of …August…… 2012

DEDICATION

To my son Kayan, the giant on whose shoulders I stand to attain greatness. You are my greatest teacher and inspiration in all things.

ACKNOWLEDGEMENTS

To my fellow classmates, who endured this journey with me and imparted some of your wisdom along the way.

I am deeply grateful to ESKOM for providing the financial support and granting me this life changing opportunity. I remain forever indebted.

The wise lecturers, who with sleight of hand removed the veil of ignorance and showed me an exciting new world. You have given me enough information to be dangerous for a start, but the curiosity to delve even deeper.

TABLE OF CONTENTS

ABSTRACT ..... **II**

DECLARATION ..... **III**

DEDICATION ..... **IV**

ACKNOWLEDGEMENTS ..... **V**

LIST OF TABLES ..... **IX**

LIST OF FIGURES ..... **IX**

CHAPTER 1: INTRODUCTION **1**

1.1 PURPOSE OF THE STUDY 1
1.2 CONTEXT OF THE STUDY 1
1.3 PROBLEM STATEMENT 3
1.3.1 MAIN PROBLEM 3
1.3.2 SUB-PROBLEMS 3
1.4 SIGNIFICANCE OF THE STUDY 3
1.5 DELIMITATIONS OF THE STUDY 4
1.6 DEFINITION OF TERMS 4
1.7 ASSUMPTIONS 4

CHAPTER 2: LITERATURE REVIEW **6**

2.1 INTRODUCTION 6
2.2 RANK ORDER FORMULATION 6
2.3 SOLVING THE *MVND* INTEGRAL 9
2.3.1 NUMERICAL INTEGRATION TECHNIQUES 9
2.3.2 SIMULATION BASED APPROACHES 10
2.4 APPLICATION OF THE RESULTS FROM THE SOLUTION TO MVND 12
2.5 ALTERNATIVE APPROACHES TO THE ANALYSIS OF RANK ORDERED DATA 14
2.6 CONCLUSION OF LITERATURE REVIEW 17
2.6.1 HYPOTHESIS 1: 17
2.6.2 RESEARCH QUESTION 2: 17
2.6.3 RESEARCH QUESTION 3 17

CHAPTER 3: RESEARCH METHODOLOGY **18**

3.1 RESEARCH METHODOLOGY 18

3.2 RESEARCH DESIGN 19
3.3 POPULATION AND SAMPLE 21
3.3.1 POPULATION 21
3.3.2 SAMPLE AND SAMPLING METHOD 21
3.4 THE RESEARCH INSTRUMENT 21
3.5 PROCEDURE FOR DATA COLLECTION 22
3.6 DATA ANALYSIS AND INTERPRETATION 22
3.7 LIMITATIONS OF THE STUDY 23
3.8 VALIDITY AND RELIABILITY 23
3.8.1 EXTERNAL VALIDITY 23
3.8.2 INTERNAL VALIDITY 23
3.8.3 RELIABILITY 23

CHAPTER 4: PRESENTATION OF RESULTS 24

4.1 INTRODUCTION 24
4.2 BEHAVIOUR OF THE *SSE* AND *SM* 24
4.2.1 VARYING THE NUMBER OF LHS 25
4.2.2 VARYING THE NUMBER OF ITERATIONS 28
4.2.3 INPUT DATA EFFECTS 31
4.2.4 INCONSISTENT DATA SETS 32
4.3 RUN TIME PERFORMANCE OF THE *SSPEA* 34
4.3.1 INCREASING THE NUMBER OF RESPONSE ITEMS 35
4.4 PARTIALLY RANKED DATA 36
4.4.1 PARTIAL RANKS AND LHS SIZE 36
4.4.2 ERROR INTRODUCED WITH PARTIAL RANKS 38
4.5 DETERMINING THE ERROR BEHAVIOUR OF THE *SSPEA* 41
4.6 SUMMARY OF THE RESULTS 44

CHAPTER 5: DISCUSSION OF RESULTS 45

5.1 INTRODUCTION 45
5.2 BEHAVIOUR OF THE *SSE* AND *SM* 45
5.3 PARTIALLY RANKED DATA 47
5.4 ERROR BEHAVIOUR OF THE *SSPEA* 48
5.5 GUIDELINES 50
5.6 CONCLUSION 51

CHAPTER 6: CONCLUSIONS AND RECOMMENDATIONS 52

6.1 INTRODUCTION 52
6.2 CONCLUSIONS OF THE STUDY 52
6.3 RECOMMENDATIONS 53
6.4 SUGGESTIONS FOR FURTHER RESEARCH 53

REFERENCES 54

APPENDIX A.1: MATLAB® SOURCE CODE 59

APPENDIX A.2: GENERATING LATIN HYPERCUBE SAMPLES 67

APPENDIX A.3: SIMULATING A RANDOM POPULATION 71

LIST OF TABLES

Table 2.1: Four levels of Measurement and the statistics appropriate to each level

Table 4.1: Increase ratio of simulation time with increase in number of iterations

Table 4.2: Percentage of voters ranking candidate $i \in \{1, 5\}$ in position $j \in \{1, 5\}$

Table 4.3: Comparison of run times

Table 4.4: Run times with increase in no. of items

Table 4.5: Simulated partial ranking data

Table 4.6: Results of simulation runs to determine the consistency for partially ranked data

Table 4.7: Simulated full rank data

Table 4.8: Simulated full rank data

Table 4.9: Study cases to determine error behaviour

LIST OF FIGURES

Figure 2.1: Algorithmic flow of the SSPEA

Figure 4.1: SSE plotted against log (no. of LHS)

Figure 4.2: log (simulation time) plotted against log (no. of LHS)

Figure 4.3: Plots of the standardized means for LHS = 2000 & 30000

Figure 4.4: SSE plotted against number of iterations

Figure 4.5: log (simulation time) plotted against logarithm no. of iterations

Figure 4.6: Standardized means as function of number of iterations

Figure 4.7: Standardized means as influenced by input data truncation

Figure 4.8: Plot of standardized mean of responses for ‘unadjusted’ and ‘adjusted’ data

Figure 4.9: Plot of SSE for ‘unadjusted’ and ‘adjusted’ data

Figure 4.10: Run time

Figure 4.11: Standardized means over the 30 simulation runs

Figure 4:12: Standardized means for a full and partially ranked data set

Figure 4:13: Standardized means for a full and modified partially ranked data set

Figure 4.14: Error in mean estimate and mean error

Figure 4.15: Estimates of the item means for Case 1 of table 4.8

Figure 5.1: Eight item data adjusted for zero entries

Figure 5.2: Best guesses plotted against all guesses of item 1 for cases in table 4.8

CHAPTER 1: INTRODUCTION

1.1 Purpose of the study

The purpose of this research is to determine guidelines and limitations for the implementation of the Shotgun Stochastic Parameter Estimation Algorithm (*SSPEA*) developed by Stacey (2006).

The *SSPEA* is applicable for use with full and partially rank-ordered data sets. The computational intensity of the *SSPEA* increases as the sample size and number of categories included increases. The *SSPEA* inherently includes methodological error in the formulation which is assumed to be random and centred on a mean error. The guidelines developed will include restrictions placed on sample size and potential solutions, using partial ranking compared to full ranking and confirmation of the behaviour of the methodological error.

The *SSPEA* will be implemented in Matlab® and performance will be compared with the Microsoft Excel / C# results presented by Grant (2010).

1.2 Context of the study

Descriptive statistics are essential tools in the analysis of data accumulated from surveys and other types of research. Two key parameters, the '*mean'* and '*variance'*, are used to determine the average of the sample and the spread (or distribution) around the average.

A vast array of literature exists on the use of parametric methods in statistics, to determine descriptive measures, and its' application is both commonplace and well understood. One downfall of parametric methods is the reliance on the data being at least interval level (Knapp, 1990; Stacey, 2006). Parametric methods therefore cannot be directly applied to typical ordinal level data, (Kaptein, Nass and Markopoulos, 2010), generated from survey's undertaken in marketing research and similar fields.

Spearman (1904) proposed a non-parametric alternative to deal with this type of data to obtain the descriptive measures. The methodology possesses significant mathematical rigour in its application forcing the average researcher to seek alternative methods for analysis.

Vermunt (1999) states that when dealing with ordinal data; truly ordinal models are rarely used and researchers generally use nominal, interval or quasi-ordinal methods in the analysis of such data. Interval analysis relies on assigning scores to the categories of ordinal variables (Vermunt, 1999) making it possible to analyse with parametric methods.

Assigning rank orders to ordinal scales has been the subject of much research, with the primary contention being the definition of the distance between the different scale items (Guttman, 1944; Cook, Kress and Seiford, 1996; Hartwell and Gatti, 2001). Mayer (1971) cautions on the use of interval scales to represent ordinal scales stating that the results can be disastrous even in terms of the correlation coefficient. He suggests that the class of rank order statistics be used on ordinal data which prevents making the assumptions associated with interval data. Labovitz (1970, 1971) counters these arguments and provides guidelines to circumvent these potential hazards. He further indicates that although some small error is introduced in this technique, it is offset by using more powerful, sensitive and interpretable statistics with known sampling error. Weiss (1986) demonstrated that rank ordered data was equally effective in providing meaningful results when compared with the results produced from the raw data. Svensson (2000, 2001) questioned whether different scales of the same variable provided comparable results and found that order consistency is an important consideration for interchangeability of scales.

Yu (2000) suggests that the computational complexity inherent in the analysis of rank ordered data can be overcome by utilising a Bayesian approach and using the Monte Carlo Markov Chain (*MCMC*).

More recently Stacey (2006) has developed the *SSPEA* to determine the *mean* and *variance* of both full and partially rank ordered survey item data. The technique offers a simple yet powerful approach to determine these parameters

without delving into the mathematically complex approaches proposed by others. Advances in computational power make it possible to implement this algorithm albeit subject to limitations to be determined in this research.

1.3 Problem statement

1.3.1 Main problem

The primary objective of this study is to determine a set of guidelines for the implementation of the algorithm. Does the computer language chosen for simulation impact the algorithm's efficiency, speed and ease of implementation? Identify the limitations imposed on its application by sample size, number of permutations and survey items considered. The *SSPEA* is in essence a computational formulation and is subject to methodological error which requires investigation.

1.3.2 Sub-problems

The first sub-problem is to implement the *SSPEA* in Matlab® and compare the results with those obtained by Grant's (2010) implementation in Microsoft Excel / C#.

The second sub-problem will be to analyse the effect on the algorithm's performance when sample size, number of iterations and survey items considered are varied.

The third sub-problem is to determine if the methodological error can be treated as a bias in the model or if it is random and oscillates around the mean error.

1.4 Significance of the study

The study fills a gap in that the guidelines developed will enable researchers to analyse rank-ordered data by exploiting the advances in computational power. It makes available a technique directly suited to treat rank-ordered data without relying on complex mathematical formulations which are unwieldy to use.

The study will also demonstrate the treatment of partially rank-ordered data using the *SSPEA* bridging the gap in the currently available literature on such data.

The outcomes of this research will have direct practical applications for marketing and social science researchers by making it possible to directly determine the *mean* and *variance* of ordinal level survey data.

1.5 Delimitations of the study

The study relies on simulated and freely available survey data to accomplish the objectives and no surveys are conducted to obtain data. The guidelines developed will be general in nature and has not been formulated with a specific discipline as the focal point.

The formulation of the algorithm and the methodology are accepted as presented and this study excludes validation of the algorithm.

1.6 Definition of terms

SSPEA will refer to the algorithm developed by Stacey (2006) unless explicitly stated as having an alternative definition. Rank-order, ranking or ranked data will be used interchangeably but the meaning of all three terms will refer to ordinal data that has been arranged / defined in order of rank. A category "*j*" with a utility greater than the utility of category "*i*" always has a lower assigned rank. *LHS* will be used to define Latin Hypercube Sampling or Samples as dictated by the context with '*k*' being the reference for the number of iterations that the *SSPEA* runs through.

1.7 Assumptions

It is assumed that the simulated hypothetical data sets adequately represent random samples drawn from a large population. A further assumption is that the failure of the simulated data sets to capture the emotional element of survey

respondents will not affect the *SSPEA*'s ability to determine the underlying *mean* and *variance* of the data sets.

The research will limit the number of variations of some variables and it is assumed that no intermediate points will behave abnormally.

It is assumed that the error in the estimates sums linearly or scales the methodological error to produce the total error. The objective is to isolate the methodological error and determine its behaviour.

Where further realistic assumptions are made during the research to enable completion, these will be reported.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

Chapter One outlined the requirement for descriptive measures in statistics on measurement scales other than interval and ratio level data. The chapter presented some insight into the current approach of applying parametric methods to obtain these measures and the shortcomings of their use. The most notable has been the error introduced when applying parametric techniques to categorical and ordinal level data.

This chapter will analyse the existing literature pertaining to the formulation of the rank ordering problem, followed by the mathematical formulation to address the problem of dealing with rank ordered data. The second section will discuss alternate numerical formulations to solve the problem and highlight the complexities inherent in this approach. The third section will present new approaches for solving the original problem by exploiting the advances in computational power. This is followed by a discussion of the algorithm being implemented. The chapter concludes with alternative approaches to the analysis of rank ordered data.

2.2 Rank order formulation

The practice of rank ordering ordinal level data has found widespread use in a multitude of disciplines. It has been successfully employed in studies as diverse as analysing preference change after an election and candidate preference in an election (Böckenholt, 2002; Diaconis, 2004). The concept of rank ordering was introduced by Spearman (1904) to obviate the need to deal with data directly when measuring the non-proportional association between two items.

Thurstone (1927, 1928) expanded on this idea and postulated that to measure a continuous variable one can use a set of values “as landmarks”. This subset is used to effectively divide the continuous variable into regions. He suggested that this subset be chosen in such a manner that they dissect the entire variable

space uniformly and proceeds to explain how rank orders can be used as a basis for measurement. This gives rise to the assumption that rank ordered data arise from measurements that follow a multivariate normal distribution (*MVND*) by design. Subsequent work on this subject has generally used this as an *a priori* choice for the distribution with unknown mean (μ) and variance (σ).

To illustrate the use and ideas of Thurstonian ranking models in practice consider the following example. In a study with *m* distinct options, the resulting problem of determining the probability of a particular rank r_i, requires evaluation of an *(m-1)*-variate normal distribution (Yao and Böckenholt, 1999) of the form:

$$P(r_i|\beta^*.\Sigma^*) = \frac{|\Gamma_i|^{-\frac{1}{2}}}{(2\pi)(m-1)/2}\int_0^\infty \dots \int_0^\infty exp\left\{-\frac{1}{2}\,(\delta_i - \boldsymbol{x})\boldsymbol{\Gamma_i^{-1}}(\delta_i - \boldsymbol{x})\right\}\, d\boldsymbol{x} \qquad (1)$$

In equation one $\delta_i = C_i^*\beta^*$ and $\Gamma_i = C_i^*\Sigma^*(C_i^*)'$ with C_i^* an *(m-1) x m* contrast matrix; β^* is a vector of means; Σ^* is the covariance matrix. As illustrated, equation one is extremely complex and precludes use by all except those adept at mathematics. The complexity inherent in equation one increases as the items to be ranked increases beyond two. This coupled with the difficulty related to estimating the parameters of Thurstonian models prevents its regular use in applied research (Yao and Böckenholt, 1999). Guttman (1956) deviated from Thurstone's approach and opted to determine directly what numerical values can be derived from observations that reproduced the best observations. This approach relied on the method of least squares for the analysis and avoided the conventional problem of using hypothetical variables and unverified hypotheses. The article claims that the method has been successfully employed in investigating morale in the United States Army.

The majority of analyses rely on the data assuming a *MVND* but Schulman (1976) has proposed the existence of alternate distributions. In the article "Ordinal data: An alternate distribution" Schulman acknowledges the difficulty introduced when assuming a distribution other than normal leaving little choice for an alternative.

Agresti (1999) presented the advances in modelling ordered categorical data and highlighted that these techniques are not widely used due to the complexity

of the newly developing methodologies. Diaconis and Eriksson (2004) have used noncommutative Fourier analysis to deal with the ranked data, from an election of five candidates, and determine the adequacy of the summarised data to capture all the information. The inherent complexity in Thurstone's formulation has not been circumvented by the advances highlighted above and continues to be the definitive standard to the problem of rank ordered data.

Siegel (1957) has confirmed that the tests applicable are dependent on the nature of the variables under investigation. This lends credence to the requirement that non-parametric methods are indeed required to address rank ordered data. These findings are summarised in the table below extracted verbatim from the article "Nonparametric statistics".

Table 2.1: Four levels of Measurement and the statistics appropriate to each level

SCALE	**DEFINING RELATIONS**	**EXAMPLES OF APPROPRIATE STATISTICS**	**APPROPRIATE STATISTICAL TESTS**
NOMINAL	Equivalence	Mode Frequency Contingency coefficient	Nonparametric statistical tests
ORDINAL	Equivalence Order	Median Percentile Spearman r_s Kendall *τ* Kendall W	
INTERVAL	Equivalence Order Ratio of intervals	Mean Standard deviation Pearson product-moment correlation Multiple product-moment correlation	Nonparametric and parametric statistical tests
RATIO	Equivalence Order Ratio of intervals Ratio of values	Geometric mean Coefficient of variation	

The following section discusses possible methodologies applicable for solving equation one in the general sense.

2.3 Solving the *MVND* integral

Finding a closed form solution to equation one has been shown to be extremely difficult and reliant on a thorough understanding of the underlying mathematics. Advances in numerical integration and computer simulation techniques have given rise to new approaches to solve equation one, and determine the unknown μ and σ of the *a priori* distribution chosen to represent the rank ordered data.

2.3.1 Numerical integration techniques

Numerical integration techniques have been used with varying degrees of success to solve problems where closed form solutions to integrals are not readily available. Statistics is no exception and Genz (1992) has proposed an extension of a numerical formulation for the computation of a *MVND*. The technique relies on transforming the original integral into an integral over a hypercube and then employing simple Monte Carlo or sub-region adaptive numerical integration algorithms. The drawback of this and other methods is the simulation time required for a single observation with *m* elements. In the context of any real survey the method would have to be capable of running through *N* such observations adding to the simulation time. Gelman (1992) presented the use of iterative and non-iterative simulation algorithms for approximating the target distribution using samples from a known starting distribution. It has been recognized that the starting distribution is key in this approach and monitoring convergence of these methods can be difficult for practical applications of these algorithms.

It is therefore clear that it is presently impractical to utilise numerical techniques to solve the *MVND* problem with currently available computing power and mathematical formulations.

2.3.2 Simulation based approaches

Statisticians are being drawn to a simpler solution to the problem presented in §2.2. This method relies on the use of Monte Carlo Markov Chains (*MCMC*) to simulate complex, nonstandard multivariate distributions (Chib and Greenberg, 1995). The most general formulation to utilise the *MCMC* approach is the versatile Metropolis-Hastings algorithm (*MHA*). Chib and Greenberg (1995) illustrate the use of the *MHA* to simulate both a Bivariate Normal and Bayesian posterior distribution.

A particularly popular special case of the *MHA* is the *GIBBS* sampler that is capable of generating random variables from a distribution indirectly without the need to calculate the density function (Casella and George, 1992; Smith and Roberts, 1993). The *GIBBS* sampler replaces the integral in equation one with a series of one dimensional random variable generators. The use of the *GIBBS* sampler is not free of important considerations and the extraction and use of information generated in the data sequence is the topic of current research and debate (Casella and George, 1992). Geweke (1991) has pointed out the following three complications that arise with the use of the *GIBBS* sampler: (1) the draws are not independent, complicating the evaluation of accuracy of the approximation when standard methods are used; (2) the distribution from which the draws are made only converge to *MVND* but is never identical with it; (3) the method does not produce an estimate of the probability of the constraint set under the unrestricted *MVND*. Geweke (1991) then proceeds to present the solutions to these three shortcomings and comments that the inclusion of these proposals generates an algorithm that produces probability estimates within 0.01 of analytically computed values. One extremely important parameter is the number of iterations required in a *GIBBS* sampler to yield sufficient accuracy. This question was addressed by Raftery and Lewis (1991), who discovered that for “well behaved” posterior distributions running the sampler for 5000 iterations, and using all iterates achieved an accuracy level of ± 0.005. This finding is not applicable to badly behaved distributions which may require a greater number of iterations. The paper also presents other important considerations for efficient implementation of the *GIBBS* sampler.

Several researchers (Yao and Böckenholt, 1999; Yu, 2000) have however successfully used the *GIBBS* sampler to address the problem of random variable generation when dealing with ranked data.

To address some of the constraints in the sampling methods discussed above some researchers have developed alternate sampling techniques (McKay, Beckman and Conover, 1979; Diaconis and Sturmfels, 1998; Cheng and Druzdel, 2000). The authors in the first and third citations above proposed the use of a stochastic simulation method, known as Latin Hypercube Sampling (*LHS*). The authors assert that *LHS* always performed better than random sampling, with improvements in the convergence rate observed in some cases. This technique trades precision of results for simulation speed but simplifies the generation of random numbers. The *LHS* also suffers from storage problems (Cheng and Druzdel, 2000). Consider two independent, uniformly distributed variables *X* and *Y*, with samples [*i, j*] required from each respectively. These can be determined follows:

$$X = F_X^{-1}((i - 1 + \xi X)/n) \quad (2)$$

$$Y = F_Y^{-1}((j - 1 + \xi Y)/n) \quad (3)$$

In the equations above *n* is the sample size, ξX and ξY are random numbers ($\xi X, \xi Y \in [0,1]$) and F_X and F_Y are the cumulative probability distributions of *X* and *Y* (Cheng and Druzdel, 2000). Stein (1987) outlines some considerations when generating large samples using *LHS* with the most notable being that if the sample size, relative to the number of variables being simulated, is not large enough the joint distribution may contain bias errors. Huntington and Lyrintzis (1998) presented two further extensions to the *LHS* formulation to improve the statistical accuracy of this sampling scheme. The first addresses the generation of the random variable and the second suggested the use of rank numbers in the ordering scheme and not the samples themselves.

Assessing the statistical significance of the parameters generated is equally important to the researcher. Efron and Tibshirani (1986) have developed techniques to assess the statistical significance of a number of parameters

generated from the simulations described above. These non-parametric algorithms rely on computational power rather than stringent theoretical formulations and avoid any parametric assumptions, approximations and analytic difficulties.

2.4 Application of the results from the solution to MVND

Stacey (2006) has encapsulated the developments to date in a novel Bayesian based approach to estimate the underlying parameters, μ and σ, of survey items that have been fully or partially ranked. The most important concept introduced is the definition of the permutation function, which overcomes the problem of over-specification in partial ranking faced by Yu's (2000) formulation. The algorithm deviates from the traditional "goodness of fit", χ^2, statistic and uses instead the sum of squared errors (*SSE*). This is due to some ranked permutations yielding very small expected frequencies which will make the χ^2 statistic unstable. A detailed description of the calculation steps in the *SSPEA* is presented below and is followed by Figure 2.1 which shows a flow diagram illustrating the sequence to be programmed.

1. Start with initial guesses for the item means, $\mu_i = 0$, and item variances, $\sigma_i^2 = 1$ for $i = 1$ to k
2. Standardise the means and variances such that the total variance is equal to 1; i.e $\sigma_{total}^2 = \sigma_{\mu}^2 + \mu_{\sigma^2} = 1$
3. Numerically simulate the population rank ordering responses by taking a very large simulated sample
4. Calculate the sum of squared errors (*SSE*) between the observed sample proportions and the simulated population proportions across all preference permutations
5. Compare the *SSE* to that obtained using previous estimates of item means and variances. If the *SSE* is less than the previously recorded minimum SSE, then the current guesses become the best estimates of the items' mean and variances.

Introduce uniformly distributed random perturbations, which decrease in magnitude with the square root of the number of iterations, to the guesses of the item means and variances and repeat steps 2 to 6 until the *SSE* is no longer reducing with successive iterations.

.

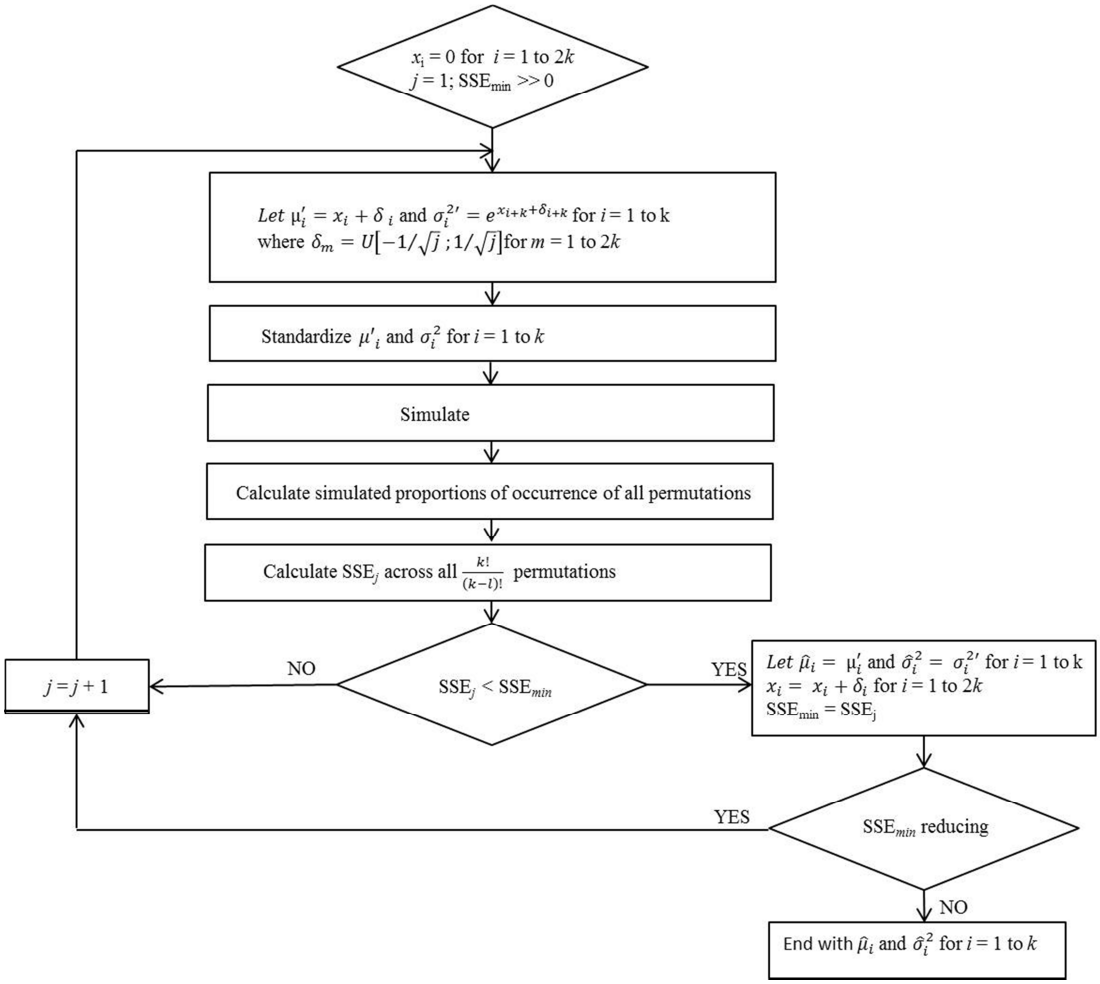


Figure 2.1: Algorithmic flow of the SSPEA

2.5 Alternative approaches to the analysis of rank ordered data

The previous sections have presented the available solutions to the original Thurstonian rank order problem formulation. This section deals with alternatives for the analysis of rank ordered data and is presented in the interests of completeness.

Kidwell, Lebanon and Cleveland (2008) assert that a ranking scale has little metric content and simply provides an ordered data set, typically with ties. If the data can be effectively visualised; important statistical properties can be revealed about the population of rater's of the items and of item-rater interaction. The authors have developed such a framework and defined a dissimilarity score based on *Kendall*'s tau distance on permutations. This dissimilarity score is then used in conjunction with multidimensional scaling to project the data into a low dimensional continuous vector space for easy visualisation.

Lebanon and Mao (2008) presented a technique for non-parametric modelling of partially ranked data that neglects the requirements for parametric assumptions on the underlying generative process. The model relies on a non-parametric *Parzen* window estimator with a *Mallows* kernel on permutations. The resulting model allows for estimation of the underlying structure based on data containing partial ranks of different types.

Francis, Dittrich and Hatzinger (2010) describe the transformation of rank ordered data into paired comparison data and demonstrate that a ranking outcome can be recorded as a paired comparison response vector. The article delves into the shortcomings of direct application of the simple *Bradley-Terry* (*BT*) model, which includes intransitive patterns, to achieve the transformation. The *BT* model is shown to not adequately address the dependence introduced by rankings transformed to paired comparisons. The authors proceed to illustrate that modelling the probabilities of whole response patterns and reducing the number of possible patterns to only those that is transitive removes these dependencies. The model is then extended to include the effect of known

covariates and random effects which accounts for heterogeneity between respondents in the item parameters.

Maydeu-Olivares and Böckenholt (2005) utilised the paired comparison approach and recast the problem with minor modifications within a structural equation modelling framework. This method has been shown to accommodate a much larger number of choice alternatives than is currently feasible with alternative estimation methods without much sacrifice of statistical efficiency.

Typical groups of rank choices can be identified using cluster analysis (Busse, Orbanz and Buhmann; 2007). Busse *et al.* (2007) develop a parametric location-scale model based on the Kendall distance and is thus related to the *Mallows*' model. The approach relies on unsupervised clustering that is "capable of performing an integrated analysis on heterogeneous, real world data, rather than decimating the data to fit the model". This method is purported to have advantages over other available clustering techniques.

Diaconis (1989) introduced the concept of applying spectral analysis to ranked data. The idea is to decompose the data into a simple interpretable structure. This method is applied to a set of election results from the American Psychological Association (*APA*) and the second order analysis gives a clear indication of the propensity of a particular grouping of two candidates to be ranked in specific positions. The technique is easily adaptable to partially ranked data. Spectral analysis has recently been applied in a study to determine the most effective termite control solution preferred by home-owners (Pandit, Paudel and Dunn; 2010).

Common characteristics among a set of variables may be identified using a powerful method termed Factor Analysis. It is commonly applicable to variables that are measured on a continuous scale but Yu *et al.* (Yu, Lam and Lo; 2005) provided an extension to the method to deal with rank ordered data. The method incorporates the Monte-Carlo Expectation-Maximization (*MCEM*) algorithm and the *GIBBS* sampler for parameter estimation and expectation respectively.

Cook and Zhu (2006) developed a general framework for modelling and treating qualitative data and provided a unified structure to embed rank ordered data in Data Envelopment Analysis (*DEA*).

Moors and Vermunt (2007) employed a latent class discrete model to validate the results of a previous study conducted by Inglehart and demonstrated the new techniques ability to uncover previously unobserved heterogeneity in the preferences.

2.6 Conclusion of Literature Review

The literature review has provided insight into the problems faced when dealing with rank ordered data. It has illustrated the progression in approaches to dealing with the original formulation and focused on the most popular, with other approaches receiving cursory attention. The literature review concluded with the latest in a string of computationally based stochastic approaches with the limitations on the implementation of the algorithm being incomplete. Although incomplete in dealing with the questions under investigation the literature review uncovered important considerations for the successful execution of this research.

This study aims to answer the questions related to the *SSPEA* posed in chapter one and further contribute to the area of rank ordered data analysis.

2.6.1 Hypothesis 1:

It is proposed that the efficiency, speed and ease of implementation of the *SSPEA* can be improved by the use of software other than Microsoft Excel / C#, in this case Matlab®

2.6.2 Research question 2:

What are the limitations placed on sample size, number of permutations and survey items considered when dealing with the implementation of the *SSPEA*?

2.6.3 Research question 3

Does the methodological error behave as a bias in the model, or is it random and oscillate around the mean error, and how is it impacted by the factors described in question 2?

CHAPTER 3: RESEARCH METHODOLOGY

This section illustrates the approach adopted to accomplish the research objectives. The assumptions, control variables and outputs will be discussed first, followed by the research design. The section concludes with an outline of the research instrument, procedure for data collection, data analysis and interpretation and comments on reliability and validity of the research design in that order.

3.1 Research methodology

Much of the data classification used in current research relies on the original definition of data types, coined by Stevens (1946), but some authors have questioned the validity of this terminology and accused it of being misleading (Velleman and Wilkinson, 1993). The importance of data classification has been stressed throughout the introductory chapters as it largely determines the appropriate analysis techniques that can applied.

This research does not address these issues and assumes that the data being analysed is correctly classified at the outset of a survey. It applies non-parametric quantitative analysis to qualitative data using the *SSPEA* to determine the *mean* and *variance* of a rank ordered data set.

Estimates of these parameters for a given set of observed data are acceptable if the algorithm produces consistent results for the *SSE* and standardized means (*SM*) over multiple runs. The field of study and its purpose will dictate the acceptable resolution required in the results. For example, in marketing studies an error of 10% in the *mean* approximation might be tolerable, but for those involving brain-function a much finer resolution in the results may be required.

The *SSPEA* is cast as a minimization problem with the *SSE* being the variable to be minimised. For a given number of survey items, the *SSE* can be influenced by the number of *LHS* utilized and the number of iterations that the

algorithm runs through to arrive at a result. With the latter parameters held constant, the assumption is that the *SSPEA* will consistently arrive at approximately the same values for the *SSE* and the associated *SM* over repetitive runs.

In order to test the assertions made by Grant (2010) and to generate the results required to draw the comparisons necessary to satisfy hypothesis one, the case of four ranks will be simulated subject to variation in the number of *LHS* and iterations (*k*) of the *SSPEA*. Each case will be run 30 times to test the optimal combination for consistency in the results. The cases presented by Grant (2010) will be repeated and the simulation run times will be tabulated for comparison and to verify hypothesis one. Different data sets will be used to test the remaining hypotheses and assertions of the research. In particular, data for an election (Diaconis, 1989) will be used to illustrate the importance of using consistent[1] data sets.

The performance of the *SSPEA* will also be tested in the presence of partially ranked data using a simulated data set. The data set in its fully ranked form will be used to determine if any error is introduced with partial ranking. The source code to generate the partially ranked data set is included as Appendix A.2.

3.2 Research Design

The research will take the form of an investigative and comparative study. The investigative component is required to yield the desired guidelines and the comparative component will investigate if any improvements in the results are noted from those presented in Grant (2010). The data used to generate any results during the course of this research will be identical to the data sets proposed by Grant (2010), simulated internally or sourced from existing surveys.

[1] This will be described in more detail in §4.2

The primary control variables will be the number of *LHS* and number of iterations with the output variables of interest being the *SSE* and *SM*. The consistency of the output variables will be used as a measure for accuracy of the *SSPEA* in predicting the unknown *mean* and *variance* of an observed data set.

Another output of interest is the simulation time, which is recorded as each of the primary control variables is changed while the number of response items is held constant. Using the optimal combination of the primary control variables the number of response items is then varied to determine its impact on simulation time. The simulation time is considered an indicator of the efficiency of the implementation proposed in this research.

The influence of data integrity has been stressed on numerous occasions during the course of this research. Using the optimally identified combination of the primary control variables as before; the impact of quality[2] and consistency of the observed data on the accuracy of the *SSPEA* will be assessed. In both cases solutions to potential pitfalls are presented.

An important consideration is the limitation placed on the number of response items that may be included for analysis with the *SSPEA*. This is addressed in the course of this research with results presented including comments on data reduction techniques available that could be used prior to analysis.

The previously untested behaviour of the *SSPEA* in the presence of partially ranked data will be presented and the error introduced quantified. The impact of the number of *LHS* and number of iterations on accuracy will be reassessed for partially ranked data to determine the new optimal combination to be used for further assessments. Three alternatives will be presented as potential solutions to deal with the error introduced when analysing partially ranked data.

[2] With reference to truncation of the observed data

The final part of the investigation addresses the behaviour of the methodological error of the *SSPEA* as a function of the primary control variables and the number of response items.

3.3 Population and sample

3.3.1 Population

This research gives no consideration to the concept of population and relies solely on data gathered from real surveys or simulated data for the purpose of testing the behaviour of the *SSPEA*. In the case of simulated samples increasing the number of *LHS* only improves the accuracy at the tails of the distribution.

3.3.2 Sample and sampling method

Where data cannot be used from Grant (2010) it will be generated and reported. A benchmark data set (Diaconis, 1989), from an election survey is included as part of the observed data sets used in this research.

3.4 The research instrument

The *SSPEA* will be implemented in Matlab® which is a powerful simulation software package widely used in complex simulation studies. The programming flexibility, coupled with the built in statistic functionality of the software will allow easier and more efficient implementation. All data required for the research was sourced from Grant (2010), real surveys or internally generated as required.

The study was conducted on a Dell laptop with a Corei7 processor, Windows 64 bit operating system and 6GB of ram. All source code was implemented using functions available in Matlab®.[3]

3.5 Procedure for data collection

As stated before the data required or this research will be sourced from Grant (2010), real surveys and where not available will be internally generated.

3.6 Data analysis and interpretation

The assumption is that the data to be analysed has been manipulated into its final form and is ready to be run through the *SSPEA*. The primary objective of this research is to analyse the change in *SSE*, *SM* and simulation time as each primary control variable is changed. Dealing with the error in the sample mean is relatively straight forward and be calculated as $SE(\bar{X}) = SD/\sqrt{n}$, where *SD* is the standard deviation of the sample and *n* is the sample size. The methodological error (*ME*) will be more difficult to address since there is no equation describing its behaviour. In this instance a graphical representation of the *ME* will be produced in the final part of this research which may shed light on its overall behaviour. The other variables of interest will be output from the simulation and will be tabulated against the results presented by Grant (2010). Graphical representation of a select set of key variables will be presented to aid visualization of the results. The results of this research are intended to be directly interpretable and no manipulation is required, save for the reordering required for presentation.

[3] There is a statistics toolbox available in newer versions of the software that was not available to the author during this study. The functionality was created with standard functions in the software.

3.7 Limitations of the study

This research excludes field data collection and in all cases will rely on simulated and freely available survey data. This does not however preclude the application of the *SSPEA* to actual field data.

3.8 Validity and reliability

3.8.1 External validity

As previously stated the major portion of this work will be carried out on contrived data sets but this in no way precludes application of the *SSPEA* to actual field data. It is assumed that the sample generated or used in the course of this research emanates from a realizable population.

3.8.2 Internal validity

The structure of the *SSPEA* lends itself to manipulation of a few key variables. The internal validity will be guaranteed by the consistent implementation of the *SSPEA* for all simulation runs.

3.8.3 Reliability

Reliability refers to the consistency of results of an experiment in repeated trials (Kalaf, Dan and Dietz, 2008b). The reliability in this research will be assessed by running a single data set through multiple independent runs of the *SSPEA* and comparing the resulting output.

CHAPTER 4: PRESENTATION OF RESULTS

4.1 Introduction

In order to facilitate ease of reference, the results will be tabulated or presented graphically as each assertion or hypothesis is addressed in turn. Some results may not directly address any of the hypotheses in this research but are essential inputs when developing the final guidelines which are the objectives of this research. The Matlab® source code inclusive of the observed data sets used to generate all results is included as Appendix A.

4.2 Behaviour of the *SSE* and *SM*

The primary objective of the *SSPEA* is minimization of the *SSE* over a predetermined number of iterations. At first glance the behaviour of the *SSE* can be attributed to the number of *LHS*, and the number of iterations. Closer inspection demonstrates the importance of two further components affecting the behaviour of the *SSE*. These are the deviations introduced by poorly truncated and inconsistent data sets.

This section considers the behaviour of the minimum, mean and maximum *SSE* over multiple runs of the *SSPEA* for a given data set. The range and final value gives an indication of the accuracy of the *SSE*. Also of importance is consistency of the resulting *SM* which is closely related to the behaviour of the *SSE*.

The investigation begins by assessing the impact of varying the number of *LHS* on the *SSE* and *SM* and proceeds to investigate the impact of varying the number of iterations. In order to draw conclusions relating to hypothesis one, the simulation time for both investigations are recorded and presented. The third part of this section investigates the impact of poorly truncated data and includes the improvement when the truncation is ‘adjusted’. The last part of this section investigates the behaviour of the *SSE* and *SM* in the presence of inconsistent data sets and concludes with the improvement achieved by

'adjusting' the data set to be consistent. The assumption is that the behaviour of the *SM* will mirror the behaviour of *SSE* i.e as the range between SSE_{min} and SSE_{max} reduces, so will range in the *SM*. The analysis in this section is conducted with a four item data set obtained from Grant (2010).

4.2.1 Varying the number of LHS

The number of *LHS* was varied as $LHS \in \{2000, 10000, 18000, 20000, 30000, 100000, 200000, 1000000\}$ to determine the impact on the *SSE*, consistency of the *SM* and simulation time calculated by the *SSPEA*. The number of iterations is held constant at 1000 and each simulation was run 30 times. Figure 4.1 illustrates the behaviour of the *SSE*[4] plotted against the logarithm of number of *LHS*. To determine the impact of changing the number of *LHS* on the simulation run time, the results for the previous test cases are potted in figure 4.2.

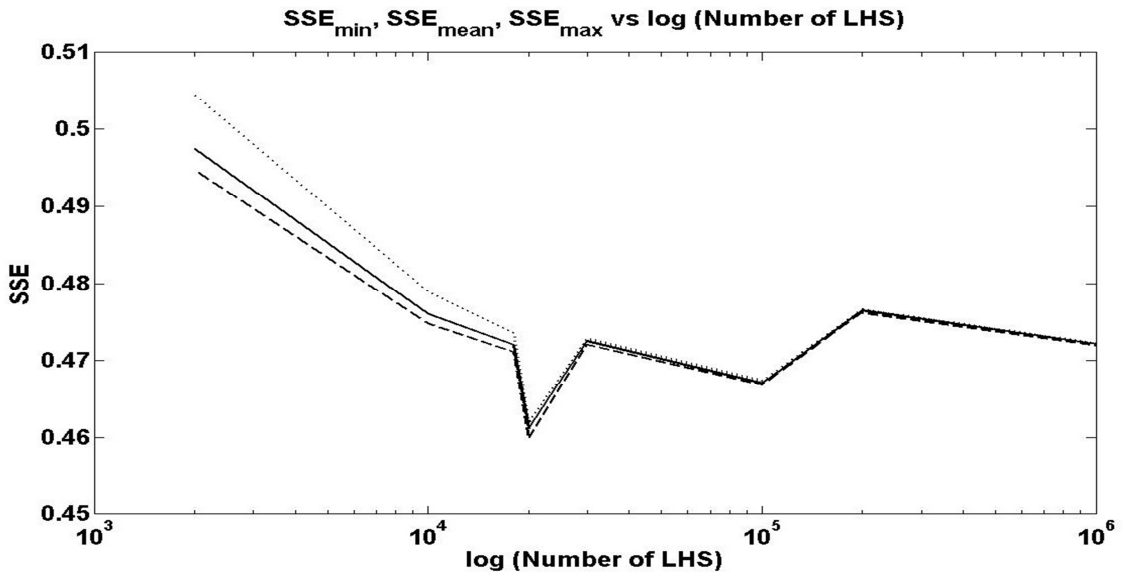


Figure 4.1: SSE plotted against log (no. of LHS)

[4] The minimum, maximum and mean of the *SSE* over the 30 runs are the 3 points plotted per run

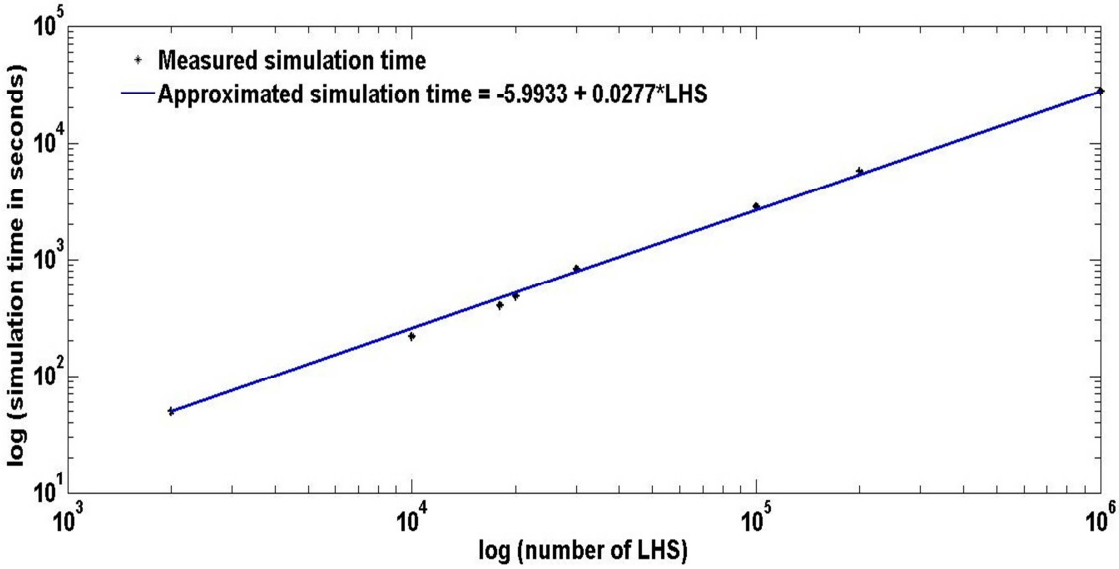
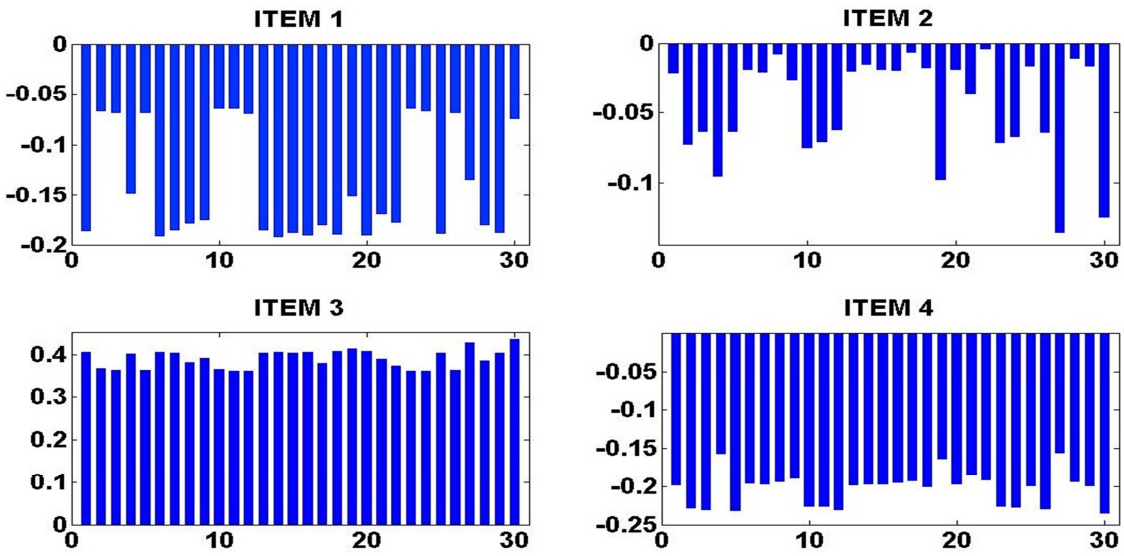


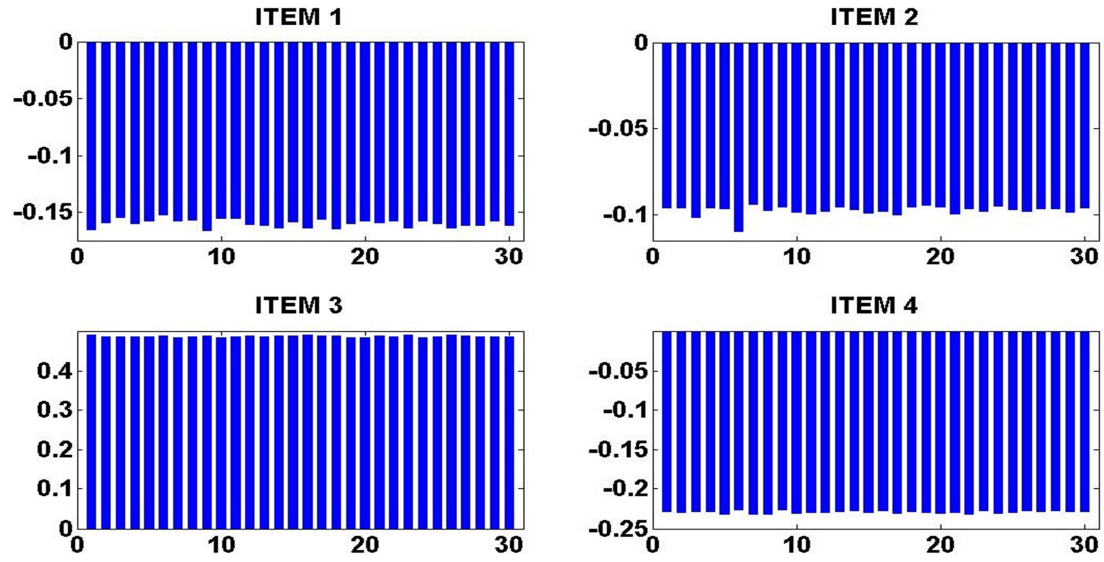
Figure 4.2: log (simulation time) plotted against log (no. of LHS)

The gradient of the line approximating the simulation times in figure 4.2 suggests that for every 10000 samples added to the *LHS* the simulation time increases by approximately a factor of 2.

Figure 4.3 demonstrates the correlation between the range {SSE_{min}, SSE_{max}} and the associated standardized item means and confirms the assumption in the introduction of this section. An increase in the number of *LHS* is accompanied by a reduction in the variation of both the range {SSE_{min}, SSE_{max}} and the estimates for the *SM*.



(a) LHS = 2000



(b) LHS = 30000

Figure 4.3: Plots of the standardized means for LHS = 2000 & 30000

From figures 4.1 and 4.3 it is clear that the *SSE*, consistency of the *SM* and simulation time is closely related to the *LHS* size.

4.2.2 Varying the number of iterations

To determine the impact of the number of iterations on the *SSE*, consistency of the *SM* and simulation time, the previous analysis is repeated with the exception that the *LHS* size is fixed at 10000 while the number of iterations is allowed to vary as, $k \in \{1000, 2000, 5000, 10000\}$. The results for the range, $\{SSE_{min}, SSE_{max}\}$ and simulation run time as functions of '***k***' are shown in figures 4.4 and 4.5 respectively. Both figures display similar behaviour to those of §4.1.1. The major difference is that the convergence between SSE_{min}, SSE_{mean} and SSE_{max} is slower with increases in the number of iterations compared with the case of increasing the number of *LHS*. Increasing the number of iterations also has a greater impact on the simulation time when compared to increasing the *LHS* size.

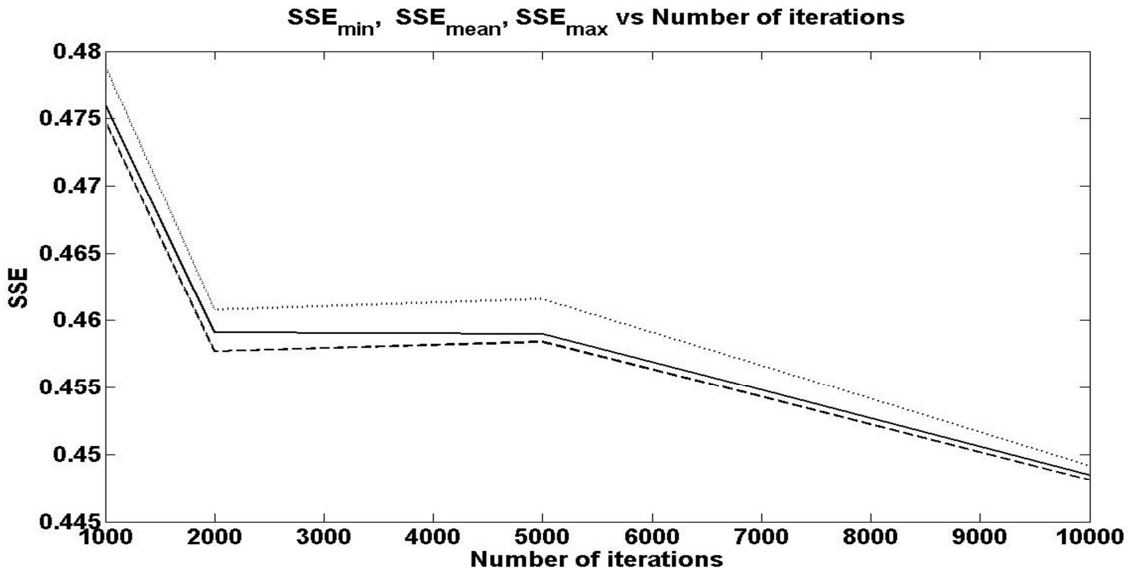


Figure 4.4: SSE plotted against number of iterations

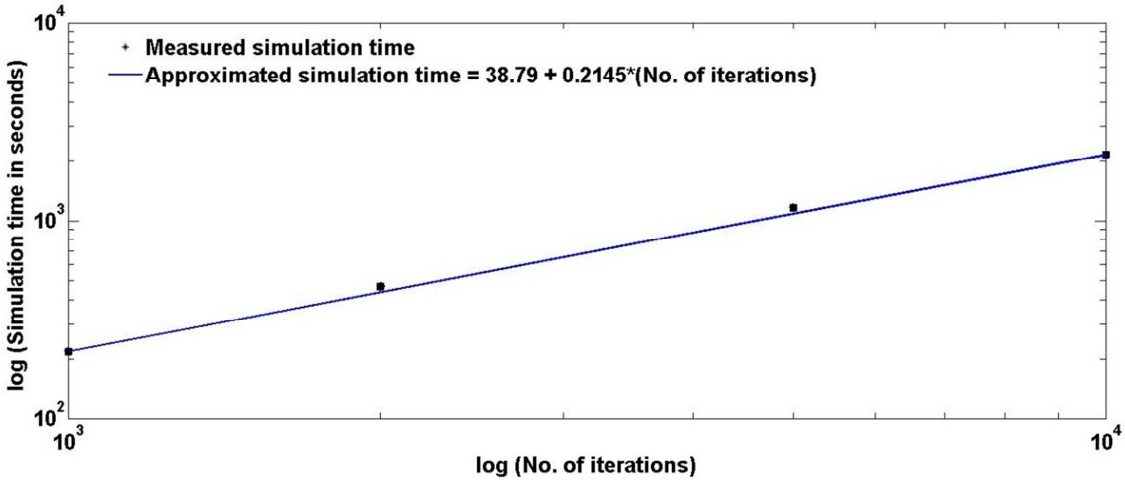


Figure 4.5: log (simulation time) plotted against logarithm no. of iterations

From figure 4.5 it is clear that the simulation time increases by approximately the same factor as the increase in iteration number. For ease of reference, the results are restated in table 4.1 for the range of iterations considered.

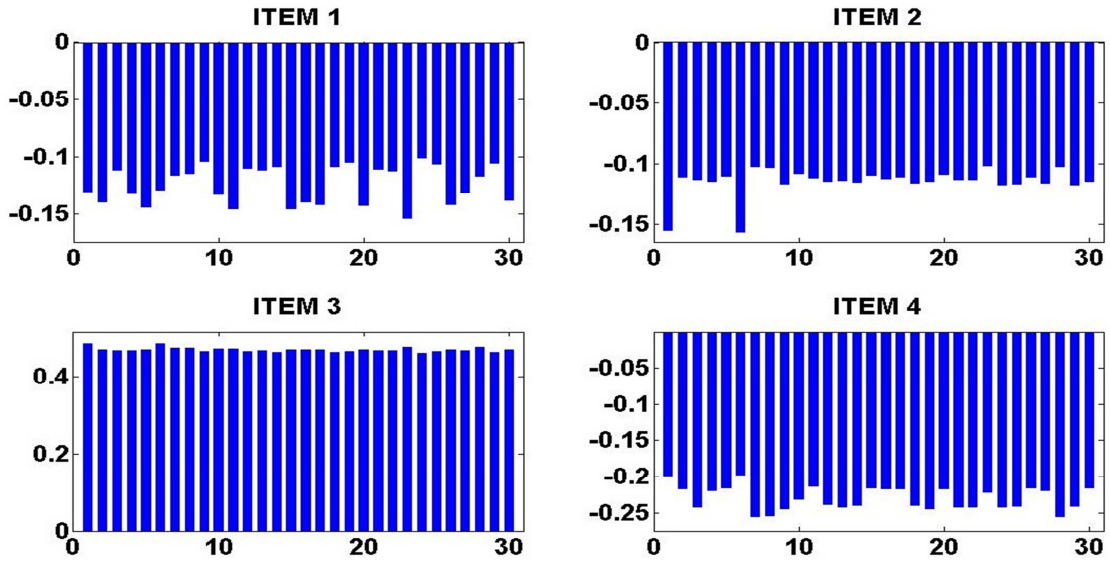
Table 4.1: Increase ratio of simulation time with increase in number of iterations

	No. of iterations			
	1000	2000	5000	10000
Simulation time	219	466	1167	2156
Factor*		2.13	5.33	9.84

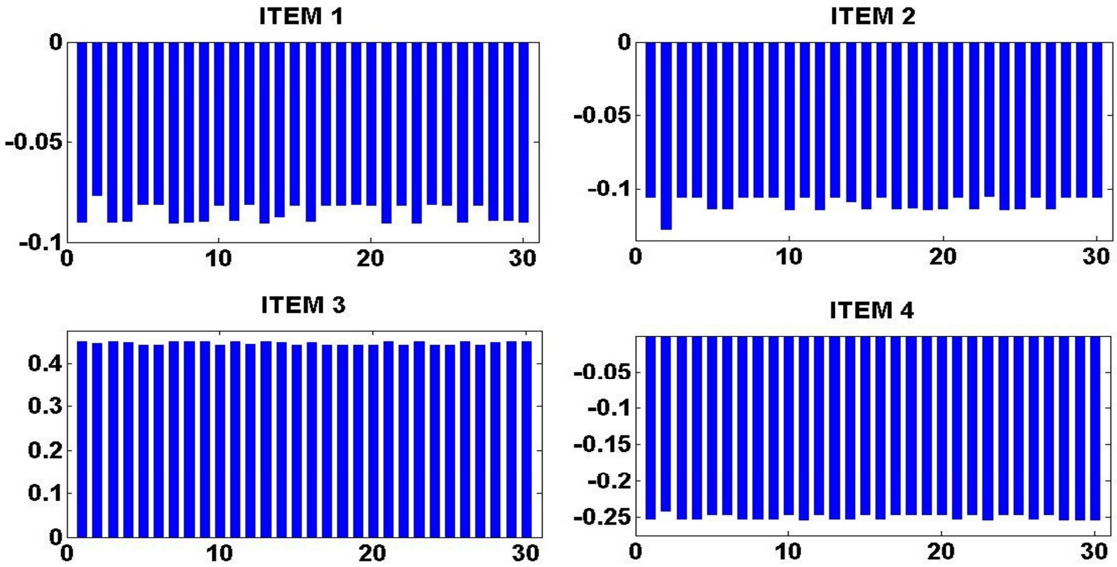
* *factors calculated relative to time for k=1000*

It is now prudent to investigate if there has been any improvement in the consistency of the *SM* as the number of iterations is changed. Figure 4.6[5] illustrates the *SM* plotted for 1000 and 10000 iterations.

[5] *No of LHS = 10000, 4 items, full rank*



(a) Number of iterations = 1000



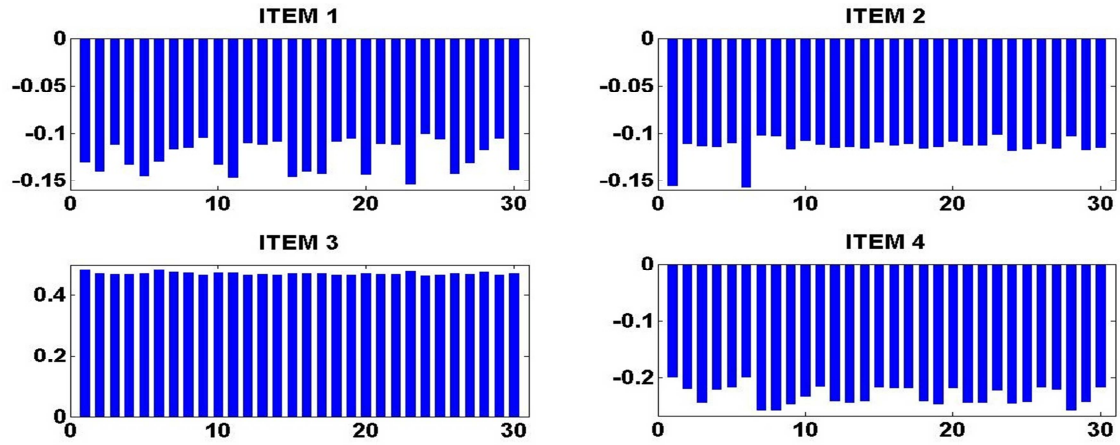
(b) Number of iterations = 10000

Figure 4.6: Standardized means as function of number of iterations

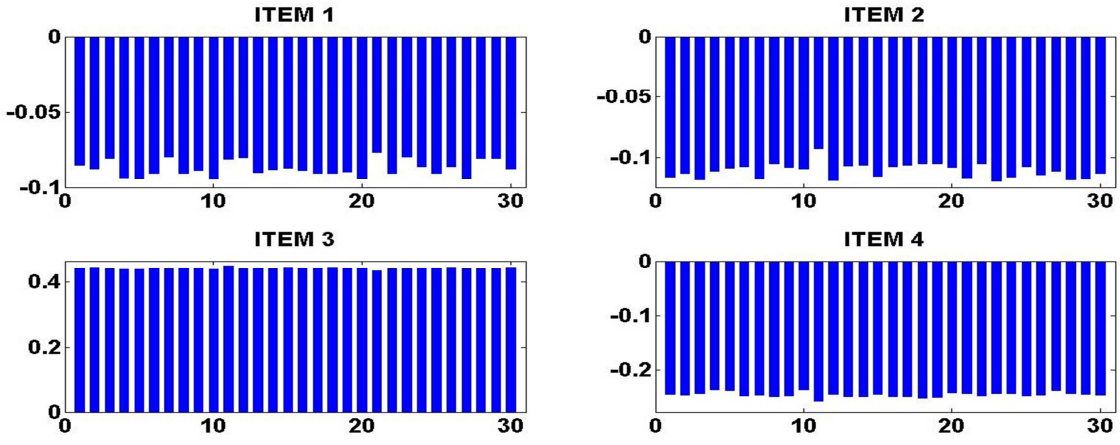
From figure 4.6 it is clear that increasing the number of iterations improves the consistency of the results and figure 4.4 demonstrates a corresponding reduction in the range $\{SSE_{min}, SSE_{max}\}$. This comes at the cost of a much higher increase in simulation time.

4.2.3 Input data effects

An interesting development emerged during the course of this research regarding input data to the *SSPEA*. It seems intuitive that the input data 'quality' will affect the outcome of the results. To illustrate this effect the original simulated four item data set from Grant (2010), truncated to six decimal places will be used to create a second data set truncated to three decimal places. Both data sets will be used as input to the *SSPEA* with *LHS* = 10000, '***k***' = 1000 and run 30 times. The consistency of the *SM* for both cases is then assessed to determine the improvement gained by reducing the number of decimal places in the input data. The results of the simulations are presented in the figure 4.7.



(a) original data



(b) 'modified' data

Figure 4.7: Standardized means as influenced by input data truncation

Figure 4.7 illustrates that when reasonably truncated data is supplied to the *SSPEA* the results are more consistent even with a relatively low number of *LHS*. In practice this problem should rarely arise but in the event that simulated data is required the researcher must take cognisance of this phenomenon and account for the discrepancy created in the results.

4.2.4 Inconsistent data sets

With fully ranked, 'consistent' data sets the sum of the frequencies across each row and column should equal to one in the observed frequency matrix. This indicates that if there was missing data in one item of an observation, the entire observation was removed from the original data set reducing the total number of observations thereby preserving 'consistency'. Another interesting property of the *SSPEA* that emerged during the course of this research is its inherent ability to detect problematic fully ranked data sets that do not fulfil this criterion. Table 4.2 illustrates such a data set obtained from Diaconis (1989). It must be noted that in this example the inconsistency results from truncation and not from errors in the design of the experiment.

*Table 4.2: Percentage of voters ranking candidate **i** ∈ {1, 5} in position **j** ∈ {1, 5}*

Candidate	Rank					
	1	**2**	**3**	**4**	**5**	Σ_{row}
1	0.18	0.26	0.23	0.17	0.15	0.99
2	0.14	0.19	0.25	0.24	0.18	1
3	0.28	0.17	0.14	0.18	0.23	1
4	0.20	0.17	0.19	0.20	0.23	0.99
5	0.20	0.21	0.20	0.19	0.20	1
Σ_{col}	1	1	1.01	0.98	0.99	

This data set was then 'adjusted' to have the row and column sums equal to one. This adjusted data set was the analysed with the *SSPEA*, and the results compared to those obtained for the original data set. Before any analysis was performed the original data from table 4.2 was transposed to have the candidates on the columns and the ranks as rows. This was repeated for the

adjusted data set. The simulation result[6] for the *SM* calculated for both data sets is shown in Figure 4.8

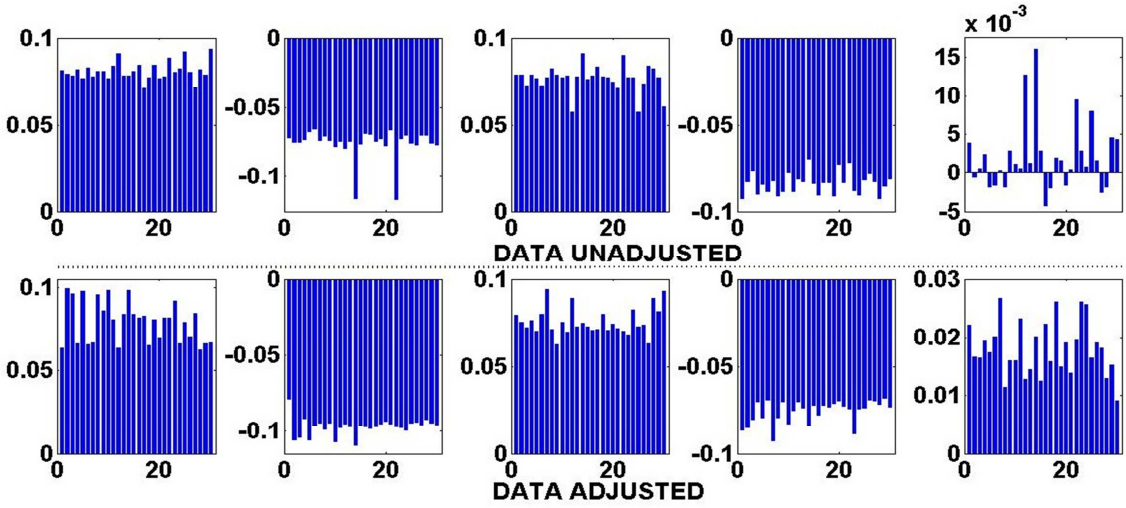


Figure 4.8: Plot of standardized mean of responses for 'unadjusted' and 'adjusted' data

It is clear from the previous figure that when inconsistent data sets are analysed using the *SSPEA*, the signs of the resulting *SM* cannot be guaranteed to be correct. The *SSE* for both cases was also plotted and evaluated to determine the impact the 'adjustment' had on the outcome of the analysis of the data. The plot of the *SSE* for both cases is shown in Figure 4.9.

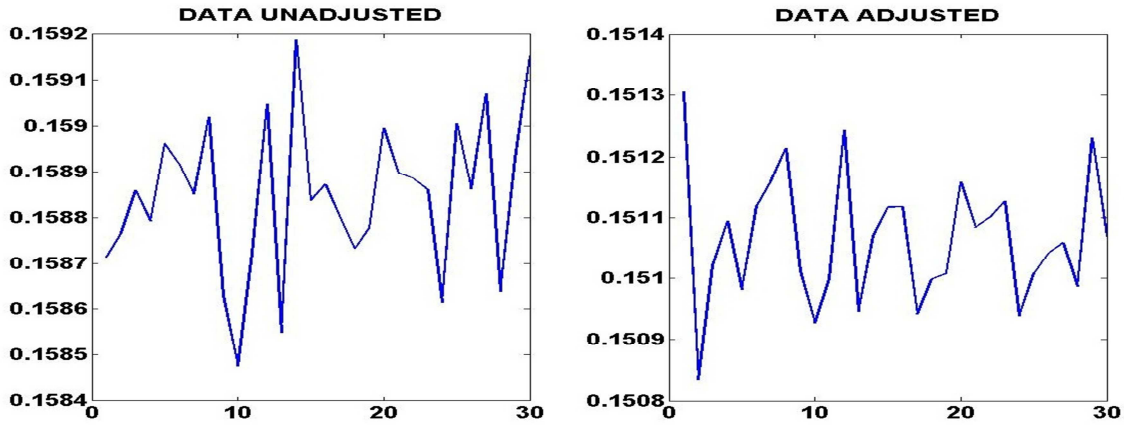


Figure 4.9: Plot of SSE for 'unadjusted' and 'adjusted' data

[6] LHS = 30000; '**k**' = 1000; no. of runs = 30

Figure 4.9 reveals that the *SSE* converges to a smaller value combined with a reduced range for the 'adjusted' data set suggesting that satisfying the consistency criteria results in a reduction in the overall error.

4.3 Run time performance of the *SSPEA*

In order to draw conclusions in support of hypothesis one, the results for the simulation time from the current implementation is compared with that of Grant (2010). For a given combination of *LHS*, '*k*', number of ranked items being considered and number of runs of the simulation the run times are compared to assess any improvement or deterioration.

With the '*k*' held constant at 1000, a comparison of the run times for the case of four ranks with $LHS \in \{10000, 30000, 50000, 70000, 90000\}$, is presented in Table 4.3.

Table 4.3: Comparison of run times

	No. of LHS				
	10000	30000	50000	70000	90000
Grant	1278	3840	6402	9600	12300
Moodley	219	831	1357	1861	2534

k = 1000. No. of runs = 30. time(sec)

To visualize the difference in the rate of increase in simulation time as the number of *LHS* increases the data from table 4.3 is plotted in figure 4.8 and includes the straight line approximations for each data set.

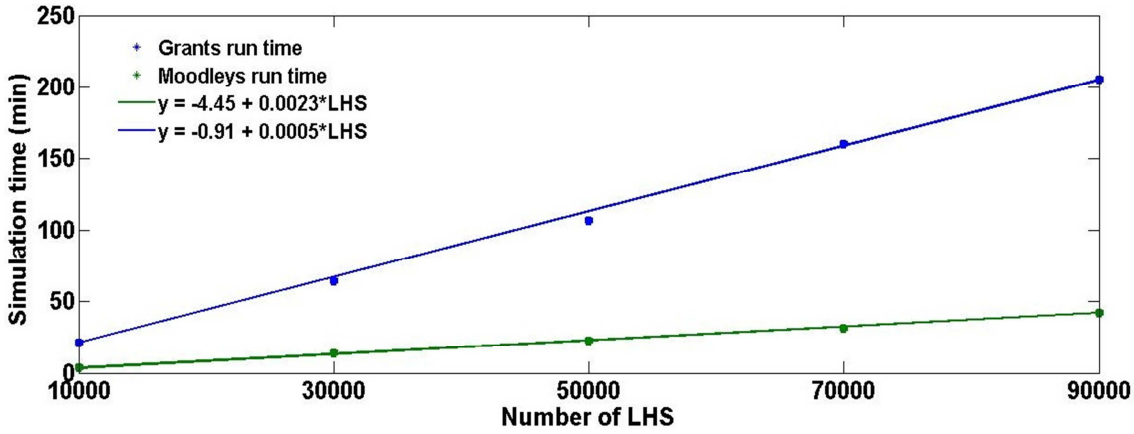


Figure 4.10: Run time vs Number of LHS

Figure 4.8 demonstrates that the implementation used in this research study has a smaller rate of increase in simulation time for increases in the number of *LHS*. The proposed implementation also has a shorter overall run time per combination of variables. This is an indication that using the proposed implementation the number of variables to be simulated can be increased without a drastic impact on the simulation time confirming hypothesis one.

4.3.1 Increasing the number of response items

Expanding on the last paragraph of the previous section, this section assesses the impact on simulation time as the number of fully ranked items increases. The amount of data processing increases significantly as the number of survey items included increases. This has been a major limitation in past attempts at approaches of this nature. The results for the simulation time as a function of the number of ranked items is given in table 4.4. Included are the results available for similar cases reported by Grant (2010).

Table 4.4: Run times with increase in no. of items

	No. of items					
	2	4	5	6	8	10
Moodley	12	25	35	39	56	68
Grant		128		233	383	

LHS = 30000, k = 1000. No. of runs = 1. time (sec)

Table 4.4 confirms that the efficient implementation proposed makes it possible to analyse larger data sets without being subjected to prohibitive simulation times. The implementation is self-contained and all elements of the simulation can be accomplished locally in contrast to Grant's (2010) Microsoft Excel / C# implementation. The simulation times are also substantially shorter than those reported by Grant. These improvements make it possible to analyse larger data sets as the 'bottleneck' is no longer the software implementation. The data suggests that an additional item adds approximately 7 seconds to the run time subject to the *LHS* size and '*k*' quoted in the footnote of table 4.4.

4.4 Partially ranked data

Stacey (2006) highlights a key advantage of the *SSPEA* as its ability to deal with partially ranked data. This first part of this section assesses the impact of the number of *LHS* on the range of the *SSE* and the consistency of the *SM* when analysing partially ranked data. The error introduced when dealing with partially ranked data sets, together with possible solutions is addressed in the second part of this section.

4.4.1 Partial ranks and LHS size

The observed, partially ranked data set in Table 4.5 was simulated as a population of 50 respondents, each ranking the top three of five candidates. The Matlab® computer code to generate the data set is included in Appendix A.3

Table 4.5: Simulated partial ranking data

Rank	Candidate				
	1	2	3	4	5
1	0.26	0.14	0.18	0.22	0.20
2	0.28	0.22	0.28	0.10	0.12
3	0.10	0.16	0.24	0.26	0.24

The algorithm was run 30 times with the number of *LHS* $\in$ {30000, 100000, 200000} and $\boldsymbol{k}$ = 1000 for each run. As demonstrated in table 4.6, the *SSE* is

consistently minimized to within a very small range with the *SM* displaying reasonably consistent results. The table also shows the *mean* and *variance* of the *SSE* and the *SM* for each item taken across all the runs as the *LHS* size increases.

Table 4.6: Results of simulation runs to determine the consistency for partially ranked data

	[min, max]	$\Delta_{\text{max-min}}$	μ	σ
SSE	0.2650, 0.2671*	0.0020	0.2660	$2.475e^{-07}$
	0.2760,0.2767**	0.0007	0.2673	$1.72e^{-08}$
	0.2724,0.2729***	$5.66e^{-04}$	0.2726	$1.55e^{-08}$
Candidate 1	0.2902, 0.3179	0.0277	0.3063	$3.744e^{-05}$
	0.2749,0.2878	0.0129	0.2784	$1.029e^{-05}$
	0.2612,0.2991	0.0378	0.2787	$7.942e^{-05}$
Candidate 2	-0.2232, -0.1691	0.0542	-0.2085	$3.044e^{-04}$
	-0.1797,-0.1586	0.0211	-0.1704	$2.196e^{-05}$
	-0.1898,-0.1619	0.0279	-0.1807	$5.746e^{-05}$
Candidate 3	0.0697, 0.0994	0.0297	0.0833	$5.43e^{-5}$
	0.0123,0.0538	0.0415	0.0425	$1.145e^{-04}$
	0.0307,0.0594	0.0288	0.0384	$5.185e^{-05}$
Candidate 4	-0.0961, -0.0367	0.0595	-0.0558	$3.59e^{-04}$
	-0.0522,-0.0347	0.0175	-0.0462	$1.722e^{-05}$
	-0.074, -0.0392	0.0347	-0.0479	$6.743e^{-05}$
Candidate 5	-0.1379, 0.0808	0.0571	-0.1253	$9.94e^{-5}$
	-0.1129,-0.0911	0.0218	-0.0144	$2.56e^{-05}$
	-0.1144, -0.0742	0.0402	-0,0886	$1.091e^{-04}$

** - 30000LHS , ** - 100000LHS, *** - 200000LHS*

To fully appreciate and visualize the variation of the *SM* over the range of the 30 runs, they have been plotted in Figure 4.11.

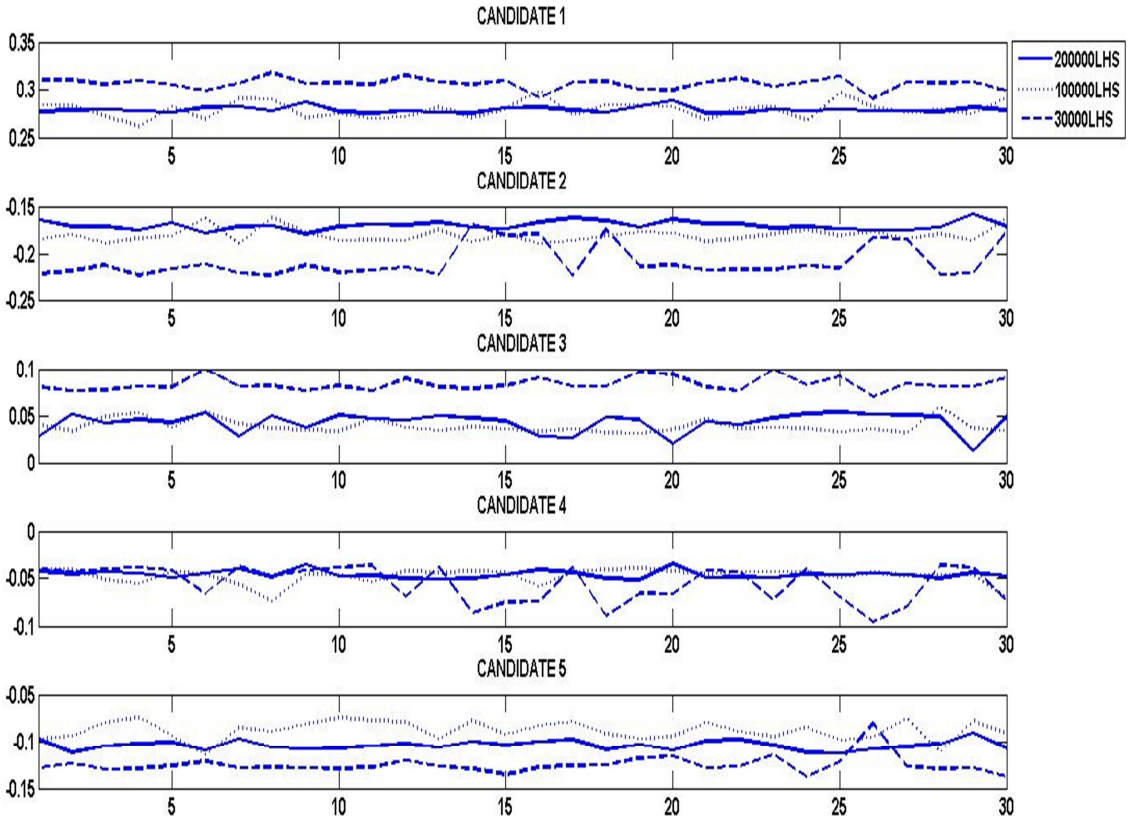


Figure 4.11: Standardized means over the 30 simulation runs

Table 4.6 and figure 4.11 show that variations in the *SSE* and *SM* reduce as the number of *LHS* increases for partially ranked data analysed using the *SSPEA*.

4.4.2 Error introduced with partial ranks

It is important to quantify if any error is introduced during the analysis of partially ranked data and if any measures can be implemented to reduce this error. To eliminate the possibility that the error emanates from variations caused by the *LHS* size the partially ranked data simulation was repeated with a *LHS* = 400000 which gave very good results with very little variation in the *SSE* and *SM*. The number of iterations was retained at 1000.

Combining the use of random numbers and the ‘consistency’ criterion, the data set from table 4.5 was expanded to create a fully ranked data set and is shown in table 4.7.

Table 4.7: Simulated full rank data

Rank	Candidate				
	1	2	3	4	5
1	0.26	0.14	0.18	0.22	0.20
2	0.28	0.22	0.28	0.10	0.12
3	0.10	0.16	0.24	0.26	0.24
4	0.16	0.30	0.10	0.15	0.29
5	0.20	0.18	0.20	0.27	0.15

This fully ranked data set is now supplied as input to the *SSPEA* and the results for the *SM* will be plotted against those obtained for the case of partial ranking. A comparison of the results is shown in figure 4.12 with the error being the difference between the two solutions.

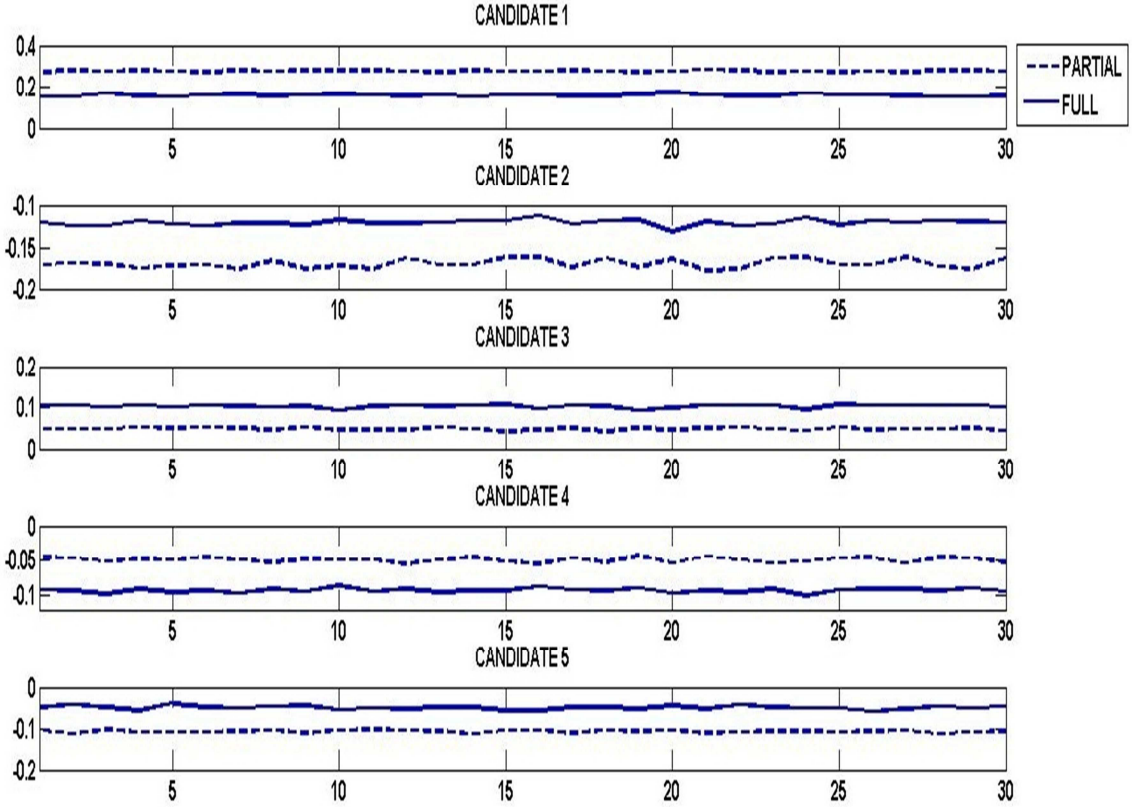


Figure 4:12: Standardized means for a full and partially ranked data set

The results in figure 4.12 indicate a significant difference between full and partially ranked data for the *SM* when analysed using the *SSPEA*. Three approaches are proposed in an attempt to reduce or eliminate the error

altogether. The first relies on the trivial approach of replacing the unranked rows in table 4.5 with zero's, the second approach uses an equal division of the difference $(1 - \sum_{row\ i=1}^{m} Freq_i)$ for a column j, to replace the rows that are not ranked. In the previous description *m* is the number of measured ranks. The final approach was to randomly divide the difference described in approach two among the rows that did not contain a rank.

All three approaches provided some improvement in the error but the latter approach provided the best overall improvement and the results for the *SM* are depicted in figure 4.13.

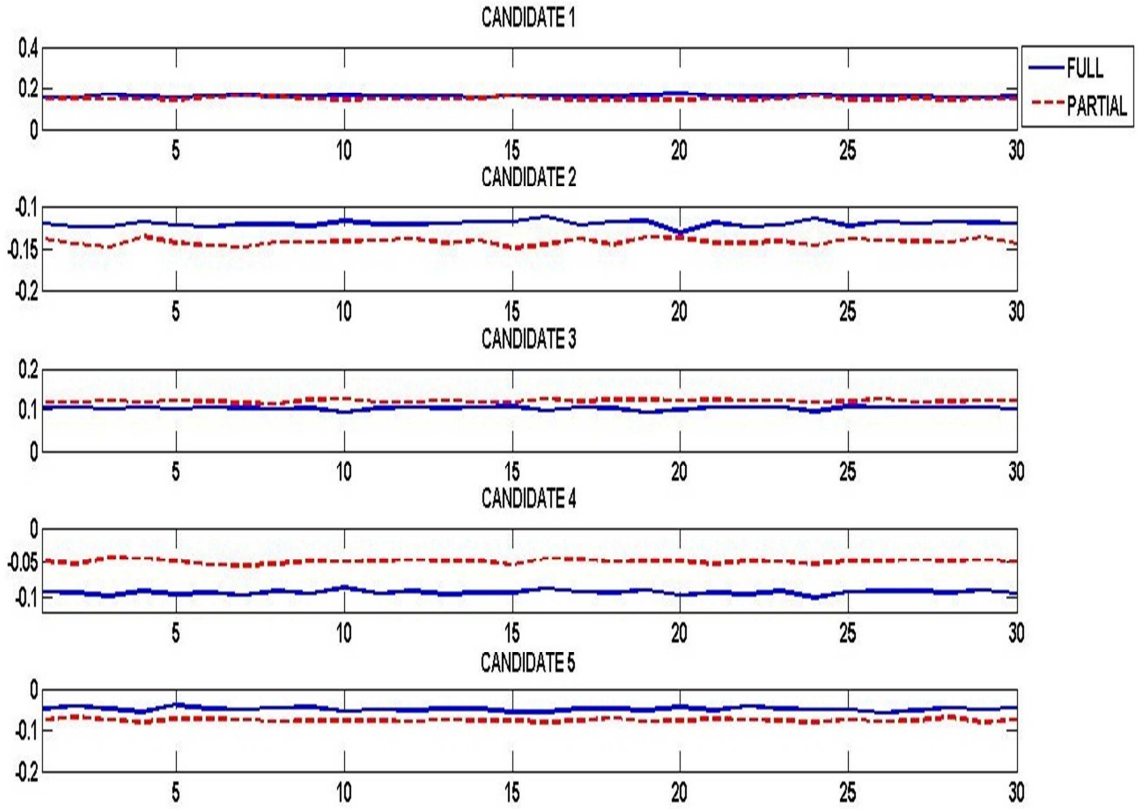


Figure 4:13: Standardized means for a full and modified partially ranked data set

The previous analysis demonstrates that the error cannot be completely eliminated but it is possible to reduce it, although it must be noted that this discussion is limited to the case of five items considering the top three ranks.

4.5 Determining the error behaviour of the *SSPEA*

As discussed previously, the *SSPEA* is not an analytic technique which makes an accurate expression for the error in the item mean estimates impossible. The previous section dealt with the error introduced when analysing partially ranked data. This section will attempt to quantify the behaviour of the methodological error in the formulation of the *SSPEA.* Section 4.2 indicated that for a fully ranked data set, the *SSPEA* with a *LHS* size greater than 30000 and run through 1000 iterations gave relatively consistent results in the *SM*. The data set for this exercise is shown in table 4.8.

Table 4.8: Simulated full rank data

	Item			
Rank	**1**	**2**	**3**	**4**
1	0.26	0.19	0.41	0.14
2	0.22	0.23	0.39	0.16
3	0.17	0.35	0.07	0.41
4	0.35	0.23	0.13	0.29

No. of LHS = 30000, k = 1000

In order to characterize the error behaviour, an assumption must be made for the 'correct' answer of the item means. This is assumed to be those associated with the minimum value of the *SSE*. The difference between this value and each 'estimate' is then considered to be the error. The error in the item means can therefore be expressed as

$$Mean\ estimate_{ierr\ ,k} = Mean\ estimate_{i\ ,k} - Mean\ estimate_{i,SSEmin} \quad (4)$$

$Mean\ estimate_{ierr\ ,k}$ - is the error in the mean for item '*i*' at iteration *k*

$Mean\ estimate_{i\ ,k}$ – is the estimate of the mean for item '*i*' at iteration *k*

$Mean\ estimate_{i,SSEmin}$ – mean of item '*i*' corresponding to the SSE_{min} during *k* iterations

An important consideration is that definition of the error in this manner, explicitly ties it to the behaviour of the *SSE* and the number of iterations of the *SSPEA*. This is compounded by the effect of the *LHS* size on the *SSE*. With these

factors identified, it is apparent that both the number of iterations of the *SSPEA* and the number of *LHS* must be varied in turn to quantify the overall error behaviour. The study cases for this exercise are therefore chosen as indicated in table 4.9:

Table 4.9: Study cases to determine error behaviour

Variable	Case study		
	1*	2	3
LHS	30000	100000	30000
No. of iterations	1000	1000	2000

** - repeated with 5 items to determine the impact of number of response items on error (Case 4)*

A fourth case, described in the footnote to table 4.9, is included to determine if the number of items being analysed has any influence on the error behaviour. In all study cases, the error is assumed to behave in the same manner for each of the item means, and the results will only be displayed for item 1 of each case from table 4.9. Figure 4.14 shows the error and the mean error for each of the cases from table 4.9

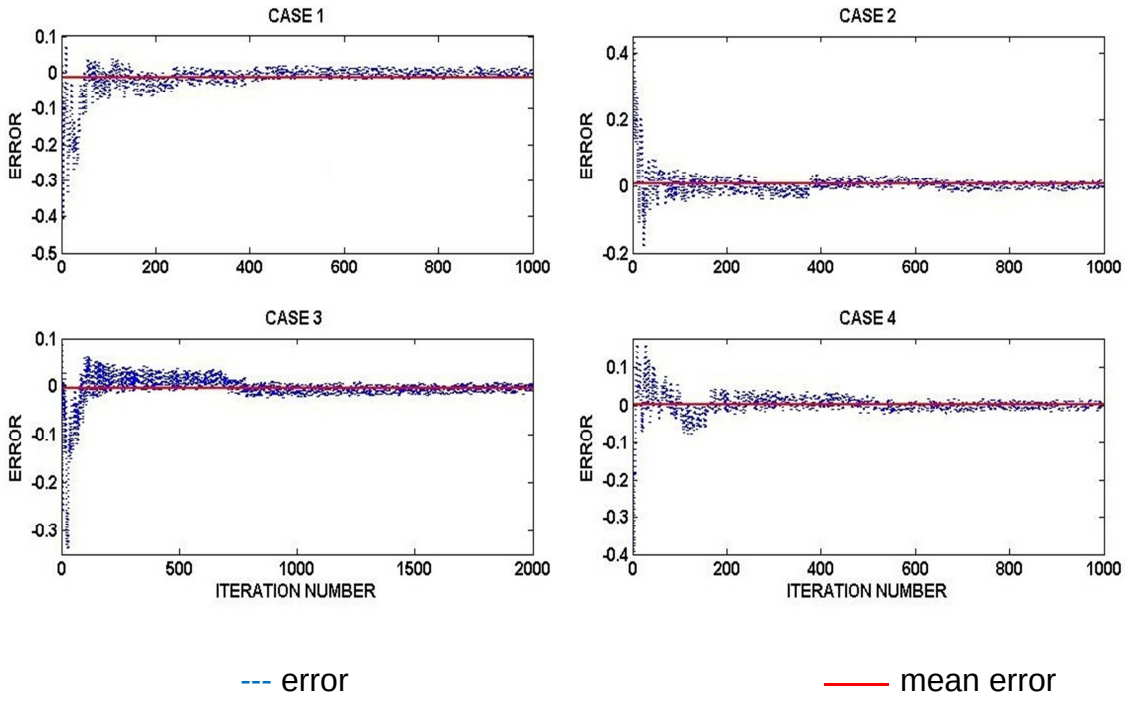


Figure 4.14: Error in mean estimate and mean error

To fully understand the error behaviour displayed in figure 4.14, it is important to consider the convergence of the item means to its final value which is assumed to be the 'correct' answer. Figure 4.15 illustrates the estimates for the item means of table 4.8 for Case 1 as they converge to their respective final values (corresponding to the minimum *SSE*). An interesting to observation in figure 4.15 is the rapid convergence of the item means to its final value. This directly impacts the results observed in figure 4.14.

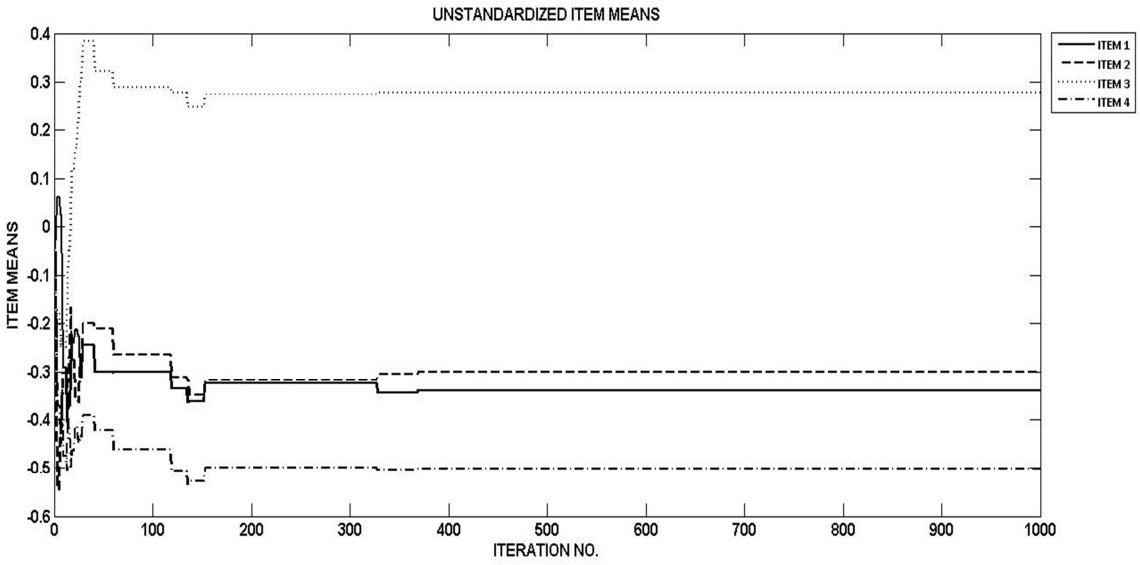


Figure 4.15: Estimates of the item means for Case 1 of table 4.8

4.6 Summary of the results

This chapter presented the results of an in depth analysis of the *SSPEA* as a precursor to satisfying the requirements to prove the hypotheses and answer the research questions posed.

The results show that for a fully ranked data set a *LHS* size of 30000 is sufficient for reasonable consistency in both the *SSE* and the *SM*. This does not hold for partially ranked data which requires a much larger *LHS* size, in the order of 200000, for the same levels of consistency. For both full and partially ranked data, 1000 iterations suffice and increasing this number has a smaller effect than increasing the *LHS*. With partially ranked data, error unrelated to the *LHS* size and number of iterations is introduced in the *SM* due to the lack of completeness in the data set. This research has proposed three possibilities to minimize this error and presented the results of the best alternative.

In the presence of 'inconsistent' data sets and observed data sets truncated to a large number of decimal places, the *SSPEA* produces questionable results for the *SSE* and *SM* if these issues are not dealt with prior to analysis. Manipulating the data into a more representative state is shown to eliminate some of the inconsistency in the *SSE* and *SM*.

A key variable of interest was the simulation time of the *SSPEA* as the number of *LHS*; '***k***' and response items were varied. The results showed that the simulation time doubled for every 10000 *LHS* added but increased by a factor equal to the increase in the number of iterations. Increasing the number of response items, with the *LHS* and '***k***' held constant, increased the simulation time by approximately 7 seconds for each item added. The results also show the improvement in simulation time and efficiency due to the implementation in specialised software.

The final part of this chapter confirmed that the methodological error of the *SSPEA* oscillates around the mean error randomly and its behaviour is not affected by the number of *LHS*, '***k***' or number of response items.

CHAPTER 5: DISCUSSION OF RESULTS

5.1 Introduction

This chapter begins with an analysis of the results presented in the previous chapter. It addresses each item in turn highlighting the major findings or outcomes. This is followed by conclusions relating to each proposition in the research and where possible the title of each section in this chapter will mirror those of chapter 4. The chapter concludes with the guidelines for implementation of the *SSPEA*. These will be informed by the prior analyses conducted in this chapter.

5.2 Behaviour of the *SSE* and *SM*

It is clear from the results that the consistency of the outcomes produced by the *SSPEA* is greatly influenced by the number of *LHS* included and the number of iterations the algorithm runs through. For fully ranked 'consistent' data sets *LHS* = 30000 will produce acceptable results with 1000 iterations. It is also clear that the larger the *LHS* size the more accurate the result, confirming Cheng and Druzdel's (2000) assertions that the results of stochastic methods can be expected to be more consistent with an increase in the number of *LHS*.

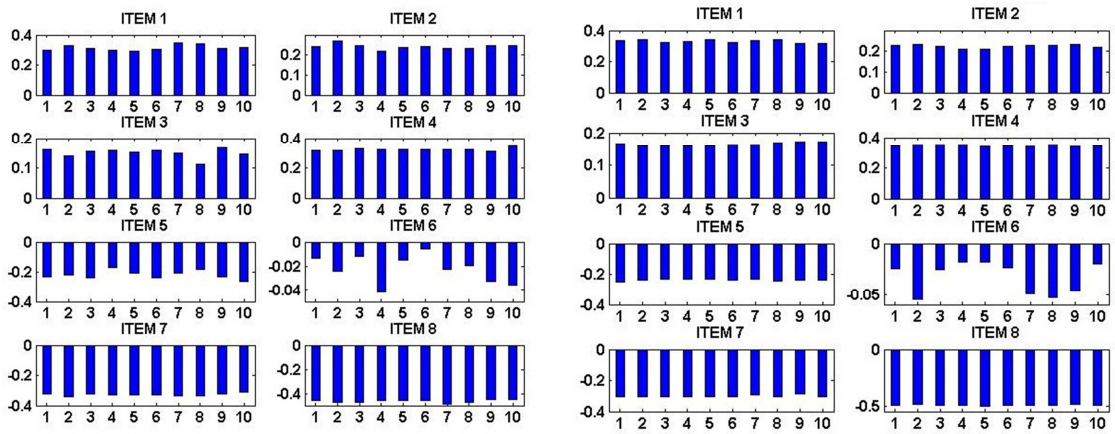
Due to the efficient implementation, a single run of the *SSPEA* with four fully ranked items, *LHS* = 1000000 and 1000 iterations will take approximately 15 minutes. For studies requiring high resolution in the results, this is no longer a prohibitive simulation time and guarantees the researcher trustworthy results.

Increases in the number of *LHS* when compared to increases in the number of iterations, is observed to be more effective in generating greater consistency in the results. The increase in simulation time is also observed to be smaller for the former case. For example, to observe similar improvements in consistency results in a 10 fold increase in simulation time for increased iterations compared with an increase of 3.6 times for increasing the number of *LHS*. This combined with the results from figure 4.13, illustrating that the results converge

to the final value after about 400 iterations, demonstrates that it is more beneficial to increase the number of *LHS* than the number of iterations.

Section 4.2 presents the impact of increasing the number of analysed items on simulation time. The results indicate that there is not a significant impact on simulation time as the number of items increase. The increase for including an extra item is shown to add only 7 seconds to the simulation time, a minimal increase considering that moving from '*n*' to '*m*' items adds (2**n* + 1) extra calculation steps in the error matrix and increases the simulated response by the number of *LHS*.

As the number of item's increase beyond 8 there may be a tendency that the observed frequency for an item, rank combination will be zero. If this arises the *SSPEA* will not generate consistent results. Replacing the zero entries with 0.001 improves the consistency but does not eliminate the variation completely even in the presence of an increased *LHS* size. Figure 5.1 illustrates this for the case of 8 items that contains zeros in rank, item combinations (2, 7) and (7, 3) analysed with *LHS* = 100000, *k* = 1000. The results are shown for the original data and using the replacement approach described.



(a) Unadjusted *(b) Adjusted*

Figure 5.1: Eight item data adjusted for zero entries

The drawback of utilising this approach is that the consistency condition is violated hence the choice of 0.001 as replacement for the zero entries to

minimize the overall impact. Unfortunately as the number of ranked items increase the case of zero entries will become more realistic.

The previous paragraph highlighted the sensitivity of the *SSPEA* to input data quality. In the presence of poor data the *SSPEA* is unable to provide good results even with increases in the *LHS* size and number of iterations. This is confirmed by the results of §4.1.3 and 4.1.4. It is therefore imperative that the researcher examines the input data prior to analysis.

It is not necessary to place a restriction on the allowable number of items considered for analysis based on any limitations of the *SSPEA*. The observed frequency will likely be skewed in one direction or the other and the distribution may even resemble a partially ranked data set as the number of ranked items grow. For even larger item sets, techniques such as cluster or factor analysis or the use latent variables, may be applied to reduce the data set and eliminate the least important items prior to analysis with the *SSPEA*. Diaconis (1989) for example has successfully applied spectral analysis to achieve a first order decomposition of data enabling simpler analysis of the resulting data pairs.

The simulation run times in §4.2 shows that using specialised simulation software and programming techniques improves the speed and efficiency of the algorithm. The functionality and implementation is self-contained improving the efficiency unlike the split between Microsoft Excel / C# employed by Grant (2010).

5.3 Partially ranked data

Analysing partially ranked data is possible using the *SSPEA*; however the results lack consistency when the data is used in its original form. The *SSPEA* has been shown to be sensitive to ‘inconsistent’ data and analysis of partially ranked data may be considered a special case of ‘inconsistent’ data’. The *SSPEA* relies on the *SSE* to update the best estimates for the item means, but with partially ranked data sets the unranked elements are excluded from contributing to the *SSE*.

Minor modifications to the original data by essentially guessing the unknown rank frequencies have been shown to improve the results. This together with an increase in the number of *LHS* to 400000 reduced, but did not completely eliminate, the error between a partially and fully ranked data set. This increases the simulation time, albeit marginally, and a more prudent approach would be to design the experiment to be as close to fully ranked as possible or decompose the data to only include variables of interest prior to analysis. The study only focused on the case of five items to investigate the error introduced in partial ranking and the assumption is that the error will be greater with a greater number of items excluded from the data set.

5.4 Error behaviour of the *SSPEA*

The behaviour of the error is seen to be consistent under variations in the number of *LHS*, iterations and a change in the number of items. In all cases the error converges rapidly to the mean error and appears to be a bias. Figure 4.13 illustrates the convergence of the item means to its final value for case 1 in table 4.8. The results indicate that due to the rapid convergence in the item means a large part of the data has a constant value subtracted from the estimates resulting in the apparent bias in the error after convergence is achieved.

Figure 5.2 below illustrates the behaviour of each estimate for item 1 in all cases from table 4.8. The estimates are seen to closely resemble the best estimates during the simulation and oscillate around the best guesses. In each case it appears that the final value is reached after approximately 400 iterations. This is a result of the formulation in the algorithm that relies on two constraints in finding new best guesses for the item means.

The first is that new estimates for the item means are based on the last best guess that collectively with other item means minimized the *SSE*. The second is that the random variation in each new estimate is reduced by the square root of the iteration number, reducing its impact as the simulation progresses. This guarantees that the error is random and oscillates around the best guesses for

the item means and appears as an apparent bias on the mean error after convergence is reached.

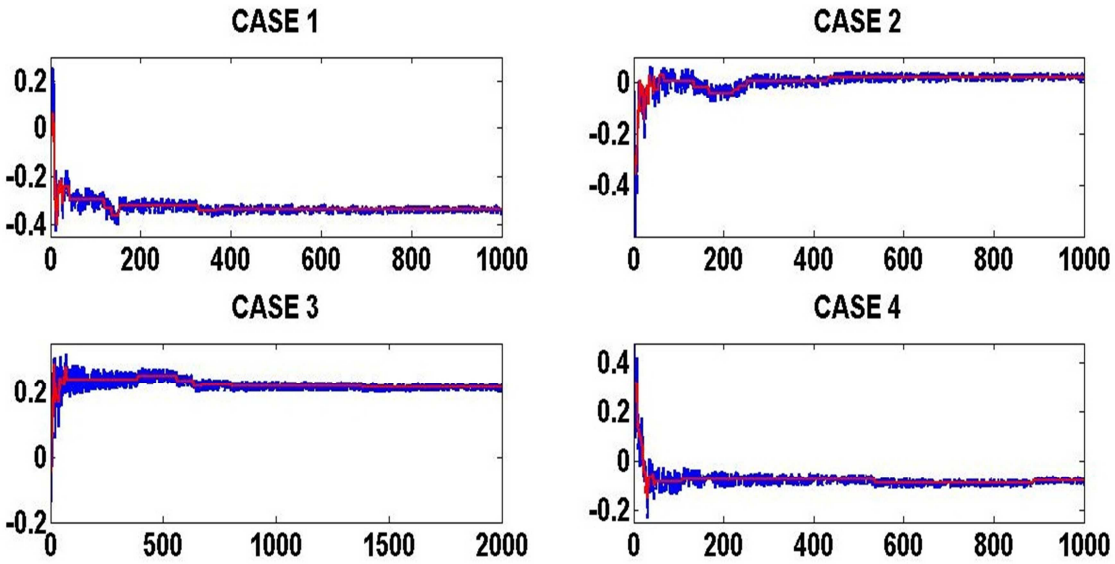


Figure 5.2: Best guesses plotted against all guesses of item 1 for cases in table 4.8

The results confirm that the error does exhibit random oscillatory behaviour around the best guesses but appears as a bias when compared with the final answer.

From figure 4.15 it is clear that the number of iterations may be reduced without loss of accuracy in the results. From this and other results generated but not presented it was determined that the number of iterations could be reduced to 500 with negligible loss of accuracy especially as the number of *LHS* increased. This will further reduce the simulation run time by a factor of approximately two based on the results presented in §4.1.2.

5.5 Guidelines

The results from chapter 4 and the prior analyses inform the guidelines for the implementation of the *SSPEA* presented below.

Stochastic methods, such as the *SSPEA*, rely on the manipulation of large data sets which increase with a requirement for higher consistency and accuracy. To maximize the benefit offered by these techniques, these algorithms should be implemented using specialised simulation software programs. This research has illustrated the benefit of using one such program, Matlab®.

To support the software high performance hardware should be used. With the rapid decline in the cost of computing power this is no longer out of reach. This research was conducted on a Dell laptop with a Core i7 processor, 6GB of ram and running on a Windows 64 bit operating system.

When generating the *LHS*, the method of Stein (1987) should be used. As the sample size increases this method approximates the tails of a normal distribution more closely.

For fully ranked data sets the observed frequency table should be truncated to two decimal places and consistency should be ensured by setting row and column sums equal to one. Using a *LHS* = 30000 and *k* = 1000 will generate good results.

When analysing partially ranked data the truncation requirement will still hold but the data table must be modified with the missing ranks set to a random proportion of difference $(1 - \sum_{row\ i=1}^{m} Freq_i)$ in a column *j* forcing the row and column sum equal to one. Using a *LHS* = 200000 and *k* = 1000 ensures reasonably consistent results will be generated.

Where possible a *LHS* = 1000000 should be used to ensure the most accurate results. As indicated the time for four items is fifteen minutes. For sizes larger than this the benefit is negligible and outweighed by the increase in simulation time.

5.6 Conclusion

The correct choice of software was demonstrated to improve the speed and efficiency of the *SSPEA*. The proposed Matlab® implementation demonstrated superior performance when compared with the simulation times presented by Grant (2010). The results show a significant reduction in the rate of increase for simulation time as the number of *LHS* or number of items to be analysed is increased. This improvement translated into the algorithm's ability to handle larger data sets without prohibitive simulation times. Increasing the number of items to be analysed was shown to be limited to a greater degree by data distribution constraints and not by the algorithm's formulation. These results proved the assertion of hypothesis one and answered research question one.

The behaviour of the error was shown to actually be random but not around the mean error prior to convergence of the *SSPEA* to a solution. Once convergence was reached the error appeared to be a bias on the mean error. This was shown to result from the properties inherent in the formulation of the algorithm. The error was shown to actually oscillate around the best estimate for the item mean at each iteration of the algorithm and be a function of the best guess plus a random number that reduced with the square root of the iteration number.

Apart from proving the hypothesis and answering the research questions posed in chapter three, this research uncovered other interesting characteristics of the algorithm. It demonstrated the algorithms ability to detect 'inconsistent' and poorly truncated data. The *SSPEA*'s ability to deal with partially ranked data was demonstrated together with the potential error and approaches to minimise this error. The impact of the number of *LHS* on the consistency of the results for partially ranked data was included as a preamble before the error could be determined. This exercise disclosed important considerations for the number of *LHS* when dealing with partially ranked data.

CHAPTER 6: CONCLUSIONS AND RECOMMENDATIONS

6.1 Introduction

This research investigated the improvements possible in the speed and efficiency of the *SSPEA* related to choice of simulation software. This chapter presents concluding remarks on the research, makes recommendations and suggest possible areas for further work

6.2 Conclusions of the study

This objective of this research was to prove that the speed and efficiency of the *SSPEA* could be improved by utilising specialised simulation software. This in itself would not be valuable except that these improvements allow for analysis of larger data sets without the prohibitive simulation times that often befall stochastic methods. This has prevented widespread use of non-parametric stochastic techniques for analysis of ordinal data. With improvements in hardware, software and the diminishing cost of computing these techniques are poised to make headway and should not be stymied by poor implementation. The results of this research proved conclusively that the simulation time for all scenarios could be reduced by a considerable amount by the correct choice of these controllable variables.

Many interesting characteristics of the *SSPEA* were uncovered and investigated. Wherever possible, potential solutions to any shortcomings were proposed with results presented to support the proposal.

The research achieved the outcomes of the hypothesis and questions posed in Chapter Two. In addition the topic of analysing partially ranked data sets using the *SSPEA* was included together with the error introduced when compared to the results for a fully ranked data set. The analysis of the results culminated in a set of guidelines for the implementation of the *SSPEA*.

6.3 Recommendations

Analysis of rank ordered data was previously limited to the use of parametric methods due to the absence of appropriate non-parametric analysis methods. Stochastic methods are becoming more viable and the *SSPEA* offers a non-parametric alternative to determine the mean and variance of rank ordered data. Researchers in fields such as marketing or biology can now use the *SSPEA* with the guidelines provided and remain confident that the results produced are valid.

6.4 Suggestions for further research

The most immediate task would be validation of the results produced by the *SSPEA* by comparison with the analytic solution for the case of simple order. This would require an analytic solution to the two rank case which is currently not available.

Another area of concern is the concept of random number generation. Every software implementation of random number generators generates at best pseudo-random numbers. This is due to the fact that the first random number generated is always the same for every first instantiation of the random number generator. It would be interesting to determine if this has a significant bearing on the results produced by the *SSPEA* and other stochastic methods.

Further work is required on quantifying the differences in the results produced by the *SSPEA* and parametric methods. Many researchers are familiar with parametric methods and argue that the error introduced is negligible and does not warrant the use of specialised non-parametric techniques. Quantifying this error for the two rank case should end this debate if it is indeed substantial.

An important omission from the formulation of the *SSPEA* is the concept of interaction or interdependence of the ranked items in the observed frequency matrix and its impact on the results produced. It would be valuable to include this effect into the formulation. The impact of other external biases in rank choice could also be included.

REFERENCES

Abelson, R. P. and Tukey, J. W. (1963) Efficient utilization of non-numerical information in quantitative analysis general theory and the case of simple order, *The Annals of Mathematical Statistics,* 34(4), pp. 1347-1369.

Agresti, A. (1999) Modelling ordered categorical data: Recent advances and future challenges, *Statistics in Medicine,* 18, pp. 2191-2207.

Böckenholt, U. (2002) A Thurstonian analysis of preference change, *Journal of Mathematical Psychology,* 46, pp. 300-314.

Busse, L. M., Orbanz, P and Buhmann, J. M. (2007) Cluster analysis of heterogeneous rank data, *Proceedings of the 24th International Conference on machine learning,* Corvalis, Oregon, pp. 113-120.

Casella, G., George, E. (1992) Explaining the Gibbs Sampler, *The American Statistician,* 46, pp. 167- 174.

Cheng, J. and Druzdzel, M. J. (2000) Latin Hypercube Sampling in Bayesian networks, *AAAI Press*, pp. 287-292.

Chib, S. and Greenberg, E. (1995) Understanding the Metropolis-Hastings Algorithm, *The American Statistician,* 49(4), pp. 327-225.

Cook, W. D., Kress, M. and Seiford, L. M. (1996) A general framework for distance-based consensus in ordinal ranking models, *European Journal of Operational Research,* 96, pp. 392-397.

Cook, W. D. and Zhu, J. (2006) Rank order data in DEA: A general framework, *European Journal of Operational Research,* 174, pp. 1021-1038.

Diaconis, P. (1989) A generalization of spectral analysis with application to ranked data, *The Annals of Statistics,* 17(3), pp. 949-979.

Diaconis, P. and Eriksson, N. (2006) Markov bases for noncommutative Fourier analysis, *Journal of Symbolic Computation,* 41(2), pp. 182-195.

Diaconis, P. and Sturmfels, B. (1998) Algebraic algorithms for sampling from conditional distributions, *The Annals of Statistics,* 26(1), pp. 363-397.

Efron, B. and Tibshirani, R. (1986) Bootstrap methods for standard errors, confidence intervals and other measures of statistical accuracy, *Statistical Science,* 1(1), pp. 54-75.

Francis, B., Dittrich, R. and Hatzinger, R. (2010) Modelling heterogeneity in ranked responses by nonparametric maximum likelihood: How do Europeans get their scientific knowledge? *The Annals of Applied Statistics,* 4(4), pp. 2181-2202

Gelman, A. (1992) Iterative and non-iterative simulation algorithms, *Computational Science and Statistics (Interface Proceedings),* 24, pp. 433-438

Genz, A. (1992) Computation of multivariate normal probabilities, *Computational and Graphical Statistics,* 1(2), pp. 141-149.

Geweke, J. (1992) Efficient simulation from the multivariate normal and student-t distributions subject to linear constraints and the evaluation of constraint problems, *www.psu.edu accessed 18 August 2011*

Grant, T. J. (2010) *The reliability of the Shotgun Stochastic Parameter Estimation Algorithm for analysing rank ordered data.* Unpublished Master of Business Administration Thesis, University of the Witwatersrand, Gauteng.

Guttman, L. (1944) A basis for scaling qualitative data, *American Sociological Review,* 9(2), pp. 139-150

Guttman, L. (1946) An approach for quantifying paired comparisons and rank order, *The Annals of Mathematical Statistics,* 17(2), pp. 144-163.

Hartwell, M.R. and Gatti, G.G. (2001) Rescaling ordinal data to interval data in educational research, *Review of educational research,* 71(1), pp. 105-131.

Huntington, D.E. and Lyrintzis, C.S. (1998) Improvements to and limitations of Latin hypercube sampling, *Probabilistic Engineering Mechanics,* 13(4), pp. 245-253

Kaptein, M., Nass, C. and Markopoulos, P. (2010) 'Powerful and consistent analysis of Likert type rating scales'*, A paper presented at Proceedings of the 28th International conference on Human factors*, Atlanta, Georgia USA, ACM press, pp. 2391-2394.

Kidwell, P., Lebanon, G. and Cleveland, W.S. (2008) Visualizing incomplete and partially ranked data, *IEEE Transactions on visualization and computer graphics,* 14(6), pp. 1356-1363.

Knapp, T. R. (1990) Treating ordinal scales as interval scales: An attempt to resolve the controversy, *Nursing Research,* 39(2), pp. 121-123.

Labovitz, S. (1970) The assignment of numbers to rank order categories, *American Sociological Review,* 35(3), pp. 515-524.

Labovitz, S. (1971) In defense of assigning numbers to ranks, *American Sociological Review*, pp. 521-522.

Lebanon, G. and Mao, Y. (2008) Non-parametric modelling of partially ranked data, *Journal of Machine Learning Research* 9, pp. 2401-2429

Maydeu-Olivares, A. and Böckenholt, U. (2005) Structural equation modelling of paired-comparison and ranking data, *Psychological Methods*, 10(3), pp. 285-304

Mayer, L. S. (1971) A note on treating ordinal data as interval data, *American Sociological Review,* 36(3), pp. 519-520.

McKay, M. D., Beckman, R.J and Conover, W.T. (1979) A comparison of three methods for selecting values of input variables in the analysis of output from computer code, *Technometrics,* 21(2), pp. 239-245.

Moors, G. and Vermunt, J. (2007) Heterogeneity in post-materialist value priorities; Evidence from a latent class discrete choice approach, *European Sociological Review,* 23(5), pp. 631-648.

Pandit, M, Paudel, K.P and Dunn, M.A. (2010) Analysis of ranked ordered data when there are complete and partial rankings, In *Southern Agricultural Economics Association Annual Meeting,* Orlando FL, *www.umn.edu,* pp. 1-25.

Raftery, A. E. and Lewis, S. (1991) How many iterations in the GIBB's sampler?, *www.psu.edu accessed 18 August 2011*

Savage, R. I. (1956) Contributions to the theory of rank order statistics - the two sample case, *The Annals of Mathematical Statistics,* 27(2), pp. 590-615.

Schulman, R. S. (1979) Ordinal data: An alternative distribution, *Psychometrika,* 44(1), pp. 3-20.

Siegel, S. (1957) Nonparametric statistics, *The American Statistician,* 11(3), pp. 13-19.

Smith, A. F.M. and Roberts, G. O. (1993) Bayesian computation via the GIBBS sampler and related Markov Chain Monte Carlo methods, *Journal of the Royal Statistical Society: Series B (Methodological),* 55(1), pp. 3-23.

Spearman, C. (1904) The proof and measurement of association between two things, *The American Journal of Psychology,* 15(1), pp. 72-101.

Stacey, A. G. (2006) Estimating the means and standard deviations of rank ordered survey items, *Management Dynamics: Journal of the South African Institute for Management Scientists,* 15(3), pp. 26-35.

Stein, C. M. (1981) Estimation of the mean of a multivariate normal distribution, *The Annals of Statistics,* 9(6), pp. 1135-1151

Stein, C.M. (1987) Large sample properties of simulations using Latin Hypercube Sampling, *Technometrics,* 29(2), pp. 143-151.

Stevens, S. S. (1946) On the theory of scales of measurement, *Science: New series,* 103(2684), pp. 677-680.

Svensson, E. (2000) Comparison of the quality of assessments using continuous and discrete ordinal rating scales, *Biometrical Journal,* 42(4), pp. 417-434.

Svensson, E. (2001) Guidelines to statistical evaluation of data from rating scales and questionnaires, *Journal of Rehabilitative Medicine,* 33, pp. 47-48

Thurstone, L.L. (1927) A law of comparative judgement, *Psychological Review,* 34, pp. 273-286

Thurstone, L.L. (1928) Attitudes can be measured, *American Journal of Sociology,* 33(4), pp. 529-554.

Velleman, P. F. and Wilkinson, L. (1993) Nominal, ordinal, interval and ratio typologies are misleading, *The American Statistician,* 47(1), pp. 65-72.

Vermunt, J. K. (1999) A general class of non-parametric models for ordinal categorical data, *Sociological Methodology,* 29(1), pp. 187-223.

Weiss, D. J. (1986) The discriminating power of ordinal data, *Journal of Social Behaviour and Personality,* 1(3), pp. 381-389.

Yao, G. and Böckenholt, U. (1999) Bayesian estimation of Thurstonian ranking models based on the Gibbs sampler, *British Journal of Mathematical and Statistical Psychology,* 52(1), pp. 79-92.

Yu, P.L.H. (2000) Bayesian analysis of order-statistics models for ranking data, *Psychometrika,* 65(3), pp. 281-299.

Yu, P.L.H, Lam, K.F and Lo, S.M (2005) Factor analysis for ranked data with application to a job selection attitude survey, *Journal of the Royal Statistical Society,* 168(3), pp. 583-597.

APPENDIX A.1: MATLAB® SOURCE CODE

```
% This code implements the SSPEA. To use change the following
% MeasuredFreq   -> contains the observed data to be analysed
% NumLHSSamples  -> number of latin hypercube samples to be used
%                   to simulate the population of responses
% NumIterations  -> number of iterations the SSPEA should run
%                   through as it attempts to minimize the SSE
% All the data used in the research is included in this file but
% should be deleted if not required. These appear commented out
% by the '%' sign
% The function latin_hs is sourced from the Mathworks file exchange
% area and is written by Budiman Manisny. It is included as an Appendix.
% Author  -> D Moodley
% Date    -> 28 December 2011
% Version -> 0

clear all
clc
t0               = cputime ;
BigLoopMin = 1  ;           % set the start for the number of iterations of the simulation
BigLoopMax = 10  ;          % set the max number of runs in the simulation
time = zeros(BigLoopMax,1) ; % create a vector that stores the time per run of the SSPEA
% ****************************************************************
% Grant 2010 - 4 rank observed data - Section 4.1
%     MeasuredFreq    = [0.256985,0.187693,0.413873,0.141448 ;
%                        0.21679,0.226571,0.386585,0.170054  ;
%                        0.165781,0.345218,0.065758,0.423244 ;
%                        0.360444,0.240518,0.133784,0.265254 ] ;

% data truncated to three decimal places
%     MeasuredFreq    = [0.257,0.188,0.414,0.141 ;
%                        0.217,0.227,0.386,0.170  ;
%                        0.166,0.345,0.066,0.423 ;
```

```
%                       0.360,0.240,0.134,0.266 ] ;

% 6 ranks
%     MeasuredFreq    = [0.21,0.19,0.20,0.14,0.19,0.07 ;
%                        0.22,0.23,0.11,0.17,0.14,0.13 ;
%                        0.17,0.27,0.15,0.27,0.06,0.08 ;
%                        0.14,0.16,0.14,0.20,0.18,0.18 ;
%                        0.16,0.10,0.27,0.17,0.13,0.17 ;
%                        0.10,0.05,0.13,0.05,0.30,0.37 ] ;

% 8 ranks
%   MeasuredFreq    = [0.20,0.19,0.18,0.14,0.05,0.07,0.07,0.10
%                      0.17,0.20,0.11,0.16,0.13,0.13,0.00,0.10
%                      0.14,0.05,0.12,0.22,0.06,0.08,0.26,0.07
%                      0.11,0.16,0.12,0.18,0.11,0.18,0.05,0.09
%                      0.14,0.11,0.21,0.14,0.13,0.17,0.02,0.08
%                      0.10,0.04,0.15,0.05,0.22,0.26,0.08,0.10
%                      0.05,0.16,0.00,0.09,0.23,0.06,0.29,0.12
%                      0.09,0.09,0.11,0.02,0.07,0.05,0.23,0.34  ];


% 10 ranks
MeasuredFreq    = [0.15,0.14,0.14,0.11,0.03,0.05,0.05,0.08,0.15,0.10 ;
                   0.13,0.14,0.08,0.12,0.10,0.10,0.00,0.20,0.05,0.08 ;
                   0.11,0.04,0.09,0.16,0.16,0.06,0.19,0.05,0.05,0.09 ;
                   0.09,0.12,0.09,0.13,0.08,0.13,0.04,0.07,0.12,0.13 ;
                   0.10,0.09,0.16,0.10,0.09,0.12,0.03,0.06,0.13,0.12 ;
                   0.12,0.03,0.11,0.05,0.17,0.19,0.07,0.07,0.10,0.09 ;
                   0.03,0.12,0.00,0.07,0.18,0.04,0.22,0.09,0.12,0.13 ;
                   0.06,0.07,0.08,0.02,0.05,0.13,0.17,0.26,0.03,0.13 ;
                   0.15,0.13,0.11,0.09,0.10,0.08,0.03,0.06,0.19,0.06 ;
                   0.06,0.12,0.14,0.15,0.04,0.10,0.20,0.06,0.06,0.07 ] ;

%     Diaconis APA election results -> results are unadjusted - Section 4.2
%     MeasuredFreq    = [0.18,0.14,0.28,0.20,0.20   ;
%                        0.26,0.19,0.17,0.17,0.21   ;
%                        0.23,0.25,0.14,0.19,0.20   ;
```

```
%                       0.17,0.24,0.18,0.20,0.19   ;
%                       0.15,0.18,0.23,0.23,0.20 ] ;


% Diaconis APA election results -> results are adjusted to force sum down each col and across each row equal to one
% col -> ranks; rows -> candidates - Section 4.2
% MeasuredFreq = [0.18,0.14,0.28,0.20,0.20 ;
%                 0.26,0.19,0.17,0.17,0.21 ;
%                 0.23,0.25,0.14,0.18,0.20 ; % changed (3,4) from 0.19 to 0.18
%                 0.17,0.24,0.18,0.22,0.19 ; % changed (4,4) from 0.2 to 0.22
%                 0.16,0.18,0.23,0.23,0.20 ];% changed (5,1) from 0.15 to 0.16

    % ********************************************************************
    % Partial ranking - simulated population of 50 respondents - Section 4.3
%     MeasuredFreq = [ 0.26, 0.14, 0.18, 0.22, 0.20 ;
%                      0.28, 0.22, 0.28, 0.10, 0.12 ;
%                      0.10, 0.16, 0.24, 0.26, 0.24 ;
%                      0.14, 0.20, 0.12, 0.17, 0.18 ;
%                      0.22, 0.28, 0.18, 0.25, 0.22 ] ;

    % Expanded partial rank to test error
%     MeasuredFreq = [ 0.26, 0.14, 0.18, 0.22, 0.20 ;
%                      0.28, 0.22, 0.28, 0.10, 0.12 ;
%                      0.10, 0.16, 0.24, 0.26, 0.24 ;
%                      0.16, 0.30, 0.10, 0.15, 0.29 ;
%                      0.20, 0.18, 0.20, 0.27, 0.15];

    % ********************************************************************
    % The two rank case
%     MeasuredFreq = [ 0.5, 0.5 ;
%                      0.5, 0.5 ] ;
    % ********************************************************************
NumItems = size(MeasuredFreq,2) ;          % get the number of items from the observed frequency matrix
NumberOfRanks = size(MeasuredFreq,1) ;     % number of ranks is equal to number of rows in observed frequency

% define the number of Latin Hypercube samples to be used
```

```
NumLHSSamples    = 30000 ;
% generate a latin hypercube sample
LHSSample = latin_hs(zeros(1,NumItems),ones(1,NumItems),NumLHSSamples,NumItems);  %lhsdesign(NumLHSSamples,NumItems)
;

%set the number the of iterations in the SSPEA
NumIterations               = 1000 ;   % number of iterations of SSPEA
SumSquaredErrors1           = zeros(1,NumItems) ;   % stores the sum across columns of the error matrix
SumSquaredErrors            = 0;   % stores the final result which is the sum across the col of SumSquaredErrors1
StandardizedMean            = zeros(1,NumItems) ;
StandardizedVariance        = zeros(1,NumItems) ;
TempVariance                = zeros(BigLoopMax,NumItems) ;
TempSimulatedVariance       = zeros(1,NumItems) ;
StandardizedStdDev          = zeros(1,NumItems) ;
% initialize variables to zero
MeanVector                  = zeros(1,NumItems) ;
SimulatedMean               = 0 ;
SimulatedVariance           = zeros(1,NumItems) ;
SimulatedMeanOfVariance     = 0 ;
SimulatedVarianceOfMean     = 0 ;
SimulatedOverallVariance    = 0 ;
CandidateResponse           = zeros(NumLHSSamples, NumItems) ;
SimulatedFreq               = zeros(NumItems ,NumItems);
ErrorMatrix                 = zeros(NumberOfRanks,NumItems) ;
% TempStoreMatris stores SSE, best estimate of mean as SSE reduces and
% corresponding STD means
TempStoreMatrix             = zeros(BigLoopMax,((2*NumItems)+1)) ;
% initialize a storage vector with zeros -> init values for both mean and std mean
TempStoreMatrix(:,1)        = 100 ;
% store the best estimates of the means for each iteration of each run
StoreBestItemMeans          = zeros(BigLoopMax,NumIterations,NumItems) ;
%************************************************************
% create storage for the mean vector estimate at each iteration
ErrorVector                 = zeros(BigLoopMax,NumIterations,NumItems) ;
ErrorVectorMean             = zeros(BigLoopMax,NumItems) ;
%************************************************************
% dummy variables to store intermediate values
```

```
A = zeros(NumLHSSamples,NumItems) ;
B = zeros(NumLHSSamples,NumItems) ;
tinit = cputime - t0 ;
% start the number of runs
while BigLoopMin <= BigLoopMax
   t = cputime ;
   % start the main loop here
   k = 1 ;
   while k <= NumIterations
       % initialise the mean vector
       % first element of TempStoreMatrix was set to 100 for all cases at the start of the simulation
       MeanVector = TempStoreMatrix(BigLoopMin,2:(1+NumItems))+ ((rand(1,NumItems) - 0.5)/sqrt(k)) ;
       SimulatedMean             = mean(MeanVector) ;         % find the mean
       SimulatedVarianceOfMean   = var(MeanVector,1) ;        % find the variance of the means
       % next two statements replace SimulatedVariance = ones(1,NumItems)
       %TempSimulatedVariance     = TempVariance(BigLoopMax,:) + ((rand(1,NumItems) - 0.5)/sqrt(k));  % find the
variance
       %SimulatedVariance         = exp(TempSimulatedVariance) ;
       SimulatedVariance = ones(1,NumItems) ;
       SimulatedMeanOfVariance   = mean(SimulatedVariance) ;               % find the mean of the variance
       SimulatedOverallVariance  = SimulatedVarianceOfMean + SimulatedMeanOfVariance ;   % find the overall variance
       StandardizedMean(1,:)     = (MeanVector - SimulatedMean ) / sqrt(SimulatedOverallVariance) ;       % find the
standardized mean
       StandardizedVariance(1,:) = SimulatedVariance(1,:) / SimulatedOverallVariance ;  % find the standardized
variance
       StandardizedStdDev(1,:)   = sqrt(StandardizedVariance) ;      % find the standardized standard deviation
       for i = 1:NumItems
           A(:,i) =  StandardizedMean(i)*(ones(NumLHSSamples,1)) ;   % create the StandardizedMean 'NumLHSSamples'
times as a column vector each time and store in A
           B(:,i) =  StandardizedStdDev(i)*(ones(NumLHSSamples,1)) ; % create the StandardizedStdDev 'NumLHSSamples'
times as a column vector each time and store in B
       end
       % generate candidate responses and rank order them
       CandidateResponse = A + B.*LHSSample ;
       [vb,vr] = sort(CandidateResponse,2) ;
       [J I]   = ndgrid(1:size(vr,1),1:size(vr,2));
       vr(sub2ind(size(vr),J,vr))=I;
```

```
        [MaxVr,MaxIndex] = max(vr,[],2) ;
        vr = MaxVr(1,1) - vr + 1 ;
        % ************************************************************************
        % simulated candidate response and simulated frequency table generation
        for i=1:NumLHSSamples
            for ii = 1:NumItems   % go across the columns
                SimulatedFreq(vr(i,ii),ii) = SimulatedFreq(vr(i,ii),ii) + 1;
            end
        end
        % ************************************************************************
        % calculate the frequency as a percentage of the total samples
        SimulatedFreq = SimulatedFreq / NumLHSSamples ;

        % ************************************************************************
        % calculate the errors between measured and simulated frequency table
        ErrorMatrix = ((MeasuredFreq - SimulatedFreq(1:NumberOfRanks,:)).^2 ) ./ ...
                      ((SimulatedFreq(1:NumberOfRanks,:) + 0.001).*(1 - SimulatedFreq(1:NumberOfRanks,:) + 0.001)) ;

        SumSquaredErrors1 = sum(ErrorMatrix,1) ; % sum across each col
        SumSquaredErrors  = sum(SumSquaredErrors1,2) ;

        if (SumSquaredErrors < TempStoreMatrix(BigLoopMin,1)) % Stores the best estimate of the mean and associated
standardized values per run
            TempStoreMatrix(BigLoopMin,:) = [SumSquaredErrors,MeanVector,StandardizedMean] ;
                %TempVariance(BigLoopMax,:) = TempSimulatedVariance ;
            StoreBestItemMeans(BigLoopMin,k,:) = MeanVector ;
        else
            StoreBestItemMeans(BigLoopMin,k,:) = StoreBestItemMeans(BigLoopMin,(k-1),:) ;
        end
        %reset the simulated frequency table
        SimulatedFreq = 0*SimulatedFreq;

        %************************************************************
        % store the estimate of the mean for the error calculation
         ErrorVector(BigLoopMin,k,:) = MeanVector ;
        %************************************************************
```

```
        % increment the counter
        k = k + 1 ;
    end
    %*************************************************************
    % find the mean of each error vector
    for i = 1:NumItems
        ErrorVector(BigLoopMin,:,i) =  ErrorVector(BigLoopMin,:,i) - TempStoreMatrix(BigLoopMin,(i+1));
        ErrorVectorMean(BigLoopMin,i) = mean(ErrorVector(BigLoopMin,:,i)) ;
        end
    %*************************************************************
    SimulatedVariance = 0*SimulatedVariance + 1 ;
    BigLoopMin = BigLoopMin + 1 ;
    time((BigLoopMin-1)) = cputime - t ;
end
TotalTime = sum(time) + tinit ;

% plot the standardized means for the best estimate per simulation run
% first set up the number of columns
if (mod(NumItems,2)~= 0)
     j = 1 ;
     RowsInSubPlot = NumItems ;
else
     j = 2 ;
     RowsInSubPlot =  NumItems / 2 ;
end
figure
for i = 1:NumItems
     StrName = strcat({'ITEM '} , {num2str(i)}) ;
     subplot(RowsInSubPlot,j,i),bar(TempStoreMatrix(:,(NumItems +1+i)),'b')
     hold
     title(StrName)
end
% plot the time per simulation run
figure, plot(time)
% plot the sum of squared errors per simulation run
figure,plot(TempStoreMatrix(:,1))
% plot the best means per run
```

```
figure, plot(StoreBestItemMeans(1,:,1),)
% find the mean of the standardized means over the runs
ItemMeansOverRuns    = mean(TempStoreMatrix(:,(NumItems+2):end),1)  ;

ItemVarianceOverRuns = var(TempStoreMatrix(:,(NumItems+2):end),0,1) ;
```

APPENDIX A.2: GENERATING LATIN HYPERCUBE SAMPLES

```
function s=latin_hs(xmean,xsd,nsample,nvar)
% s=latin_hs(xmean,xsd,nsample,nvar)
% LHS from normal distribution, no correlation
% method of Stein
% Stein, M. 1987. Large Sample Properties of Simulations Using Latin Hypercube Sampling.
%                 Technometrics 29:143-151
% Input:
%   xmean   :  mean of data (1,nvar)
%   xsd     : std.dev of data (1,nvar)
%   nsample : no. of samples
%   nvar    : no. of variables
% Output:
%   s       : random sample (nsample,nvar)
%
% Uses Peter Acklam inverse normal CDF
%
%   Budiman (2003)
% References:
% Iman, R. L., and W. J. Conover. 1980. Small Sample Sensitivity Analysis Techniques for Computer Models,
% with an Application to Risk Assessment.Communications in Statistics: Theory and Methods A9: 1749-1874
% McKay, M. D., W. J. Conover and R. J. Beckman. 1979.A Comparison of Three Methods for Selecting Values
% of Input Variables in the Analysis of Output from a Computer Code. Technometrics 21: 239-245
% downloaded from
% http://www.mathworks.com/matlabcentral/fileexchange/4352-latin-hypercube-sampling/content/latin_hs.m
%
ran=rand(nsample,nvar);
s=zeros(nsample,nvar);
% method of Stein
for j=1: nvar
   idx=randperm(nsample);
   P=(idx'-ran(:,j))/nsample;       % probability of the cdf
   s(:,j) = xmean(j) + ltqnorm(P).* xsd(j); % this can be replaced by any inverse distribution function
end
```

```
function z = ltqnorm(p)
%LTQNORM Lower tail quantile for standard normal distribution.
%
%   Z = LTQNORM(P) returns the lower tail quantile for the standard normal
%   distribution function.  I.e., it returns the Z satisfying Pr{X < Z} = P,
%   where X has a standard normal distribution.
%
%   LTQNORM(P) is the same as SQRT(2) * ERFINV(2*P-1), but the former returns a
%   more accurate value when P is close to zero.

%   The algorithm uses a minimax approximation by rational functions and the
%   result has a relative error less than 1.15e-9.  A last refinement by
%   Halley's rational method is applied to achieve full machine precision.

%   Author:      Peter J. Acklam
%   Time-stamp:  2003-04-23 08:26:51 +0200
%   E-mail:      pjacklam@online.no
%   URL:         http://home.online.no/~pjacklam

   % Coefficients in rational approximations.
   a = [ -3.969683028665376e+01  2.209460984245205e+02 ...
         -2.759285104469687e+02  1.383577518672690e+02 ...
         -3.066479806614716e+01  2.506628277459239e+00 ];
   b = [ -5.447609879822406e+01  1.615858368580409e+02 ...
         -1.556989798598866e+02  6.680131188771972e+01 ...
         -1.328068155288572e+01 ];
   c = [ -7.784894002430293e-03 -3.223964580411365e-01 ...
         -2.400758277161838e+00 -2.549732539343734e+00 ...
          4.374664141464968e+00  2.938163982698783e+00 ];
   d = [  7.784695709041462e-03  3.224671290700398e-01 ...
          2.445134137142996e+00  3.754408661907416e+00 ];

   % Define break-points.
   plow  = 0.02425;
   phigh = 1 - plow;
```

```
% Initialize output array.
z = zeros(size(p));

% Rational approximation for central region:
k = plow <= p & p <= phigh;
if any(k(:))
   q = p(k) - 0.5;
   r = q.*q;
   z(k) = (((((a(1)*r+a(2)).*r+a(3)).*r+a(4)).*r+a(5)).*r+a(6)).*q ./ ...
          (((((b(1)*r+b(2)).*r+b(3)).*r+b(4)).*r+b(5)).*r+1);
end

% Rational approximation for lower region:
k = 0 < p & p < plow;
if any(k(:))
   q  = sqrt(-2*log(p(k)));
   z(k) = (((((c(1)*q+c(2)).*q+c(3)).*q+c(4)).*q+c(5)).*q+c(6)) ./ ...
          ((((d(1)*q+d(2)).*q+d(3)).*q+d(4)).*q+1);
end

% Rational approximation for upper region:
k = phigh < p & p < 1;
if any(k(:))
   q  = sqrt(-2*log(1-p(k)));
   z(k) = -(((((c(1)*q+c(2)).*q+c(3)).*q+c(4)).*q+c(5)).*q+c(6)) ./ ...
          ((((d(1)*q+d(2)).*q+d(3)).*q+d(4)).*q+1);
end

% Case when P = 0:
z(p == 0) = -Inf;

% Case when P = 1:
z(p == 1) = Inf;

% Cases when output will be NaN:
k = p < 0 | p > 1 | isnan(p);
```

```
if any(k(:))
   z(k) = NaN;
end

% The relative error of the approximation has absolute value less
% than 1.15e-9.  One iteration of Halley's rational method (third
% order) gives full machine precision.
k = 0 < p & p < 1;
if any(k(:))
   e = 0.5*erfc(-z(k)/sqrt(2)) - p(k);          % error
   u = e * sqrt(2*pi) .* exp(z(k).^2/2);        % f(z)/df(z)
   %z(k) = z(k) - u;                             % Newton's method
   z(k) = z(k) - u./( 1 + z(k).*u/2 );          % Halley's method
end
```

APPENDIX A.3: SIMULATING A RANDOM POPULATION

```
% This code is used to simulate the population used for the partial rank
% analysis
% Each response was generated as a random response for five items in the range {0, 1}.
% A random permutation of the index {1, 50} was generated and a new sample created by combining the random choices
% and the permuted indices.
% SimulatedPopulation -> Specify the size of the population to be simulated
% NumLHSSamples  -> number of latin hypercube samples to be used
% NumberOfItems       -> number of items
% NumberOfRanks       -> number of ranks considered
% Author  -> D Moodley
% Date    -> 28 December 2011
% Version -> 0

clear all
clc
SimulatedPopulation = 50 ;
NumberOfItems       = 5 ;
NumberOfRanks       = 3;
a=rand(SimulatedPopulation,NumberOfItems) ;
b=randperm(SimulatedPopulation) ;
% randomly permute the population while preserving the order
c = zeros(SimulatedPopulation,NumberOfItems);
for i=1:SimulatedPopulation
  c(i,:) = a(b(i),:) ;
end
[vb,vr]=sort(c,2) ;
[J I]   = ndgrid(1:size(vr,1),1:size(vr,2));
vr(sub2ind(size(vr),J,vr))=I;
[MaxVr,MaxIndex] = max(vr,[],2) ;
vr = MaxVr(1,1) - vr + 1 ;
PartialRankingFreq = zeros(NumberOfRanks,NumberOfItems) ;
for i=1:SimulatedPopulation
    for ii=1:NumberOfItems
        TempVar = vr(i,ii) ;
```

```
        switch TempVar
            case 1
                PartialRankingFreq(1,ii) = PartialRankingFreq(1,ii) + 1 ;
            case 2
                PartialRankingFreq(2,ii) = PartialRankingFreq(2,ii) + 1 ;
            case 3
                PartialRankingFreq(3,ii) = PartialRankingFreq(3,ii) + 1 ;
            otherwise
                %PartialRankingFreq(i,ii) = 0;
        end
    end
end
PartialRankingFreq = PartialRankingFreq / SimulatedPopulation ;
```