

APPLICATION OF PRINCIPAL COMPONENT
ANALYSIS TO DECISION
SUPPORT SYSTEM

By

SHAOKAI WEN

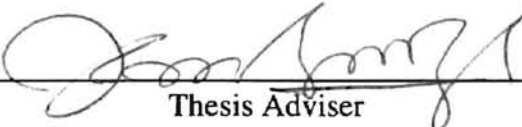
Master of Science
Beijing Agricultural University
Beijing, China
1988

Doctor of Philosophy
Oklahoma State University
Stillwater, Oklahoma
1996

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
July, 1997

APPLICATION OF PRINCIPAL COMPONENT
ANALYSIS TO DECISION
SUPPORT SYSTEM


Thesis Approved:



Thesis Adviser







Dean of the Graduate College

ACKNOWLEDGMENTS

I would like to thank Dr. John P. Chandler for awarding me assistantship and giving me this opportunity to pursue my graduate studies at Computer Science Department, Oklahoma State University. Thanks are also extended to my major advisor, Dr. K. M. George, for his guidance, advice, assistance and encouragement throughout my master program. Appreciation is likewise expressed to Dr. William D. Warde for his encouragement and help throughout my graduate studies and service on my graduate committee. Thanks are also extended to Dr. Dave S. Buchanan, for his encouragement and letting me take many computer courses when studying for my Ph. D program in Animal Science, and Dr. Blayne E. Mayfield for consulting me when I developed the package.

My thanks go to all of my fellow graduate students who broadened the quality of my education at OSU.

My appreciation also extends to Mrs. Anna Ventris for preparing many paperwork for me during my study in Computer Science Department.

To my parents, Xibin Wen and Qinxiu Jiang, I humbly acknowledge their love, understanding and support. Finally, my greatest thanks go to my lovely wife, Rui Zhang, for her love, encouragement, understanding, support and many sacrifices she made so that I could complete my education.

TABLE OF CONTENTS

Chapter		Page
I.	INTRODUCTION	1
II.	REVIEW OF THE LITERATURE	4
	Principal Component Analysis and Eigenvalues of Covariance Matrix	4
	Correlation Matrix vs. Covariance Matrix	10
	Application of Principal Component Analysis.....	13
	Computation of Eigenvalue and Eigenvector.....	15
	Power Method	15
	Hessenberg and Inverse Power Method	19
	Bisection Method for Symmetric Hessenberg.....	21
	QR Algorithm for Non-symmetric Hessenberg	23
	Inverse Power Method.....	25
	Sensitivity Analysis of the Estimates and Decision System.....	26
	Roundoff Error	27
	Influence of Outliers and Influential Observations on Estimates	28
	Sampling Error	29
	Sensitivity of Decision Systems	30
III.	DESIGN AND IMPLEMENTATION	33
	Classes.....	34
	RowClass.....	35
	TableClass	36
	Matrix	38
	SquareMatrix.....	39
	Sensitivity	39
	Software Architecture.....	41
	Abstract Level Algorithm.....	42
	Key Algorithms	43

Chapter		Page
IV.	RESULTS AND DISCUSSION.....	47
V.	CONCLUSION AND FUTURE WORK.....	65
	REFERENCES.....	66

LIST OF TABLES

Table	Page
1. Principal components based on the covariance matrix for five variables.....	10
2. Principal components based on the covariance matrix for five variables after the units of cost changed	11
3. Principal components based on the correlation matrix for five variables.....	11

LIST OF FIGURES

Figure	Page
1. The simple view of PCA.....	5
2. The relationship among the class in the project.....	41
3. Abstract level control flow	42
4. The window before loading data	49
5. The file open window	50
6. Data loading window	51
7. The window before calculating correlation	52
8. Correlation coefficient window	53
9.. The window before estimating eigenvalue and eigenvector.....	54
10. Eigenvalue and eigenvector window	55
11. The window before calculating rank value.....	56
12. The data and rank window.....	57
13. The window before calculating rank interval	58
14. Weight choose window.....	59
15. Data and rank interval window	60
16. Data sort dialog and window	61
17. Sorted data and rank window.....	62
18. The window before calculating weight interval.....	63
19. Data, rank interval and weight interval window.....	64

CHAPTER I

Introduction

Decision support systems are software which are used to develop insight into system behavior and help managers to make effective plans and decisions. Simulation and modeling are the basic weapons which are used to simplify the problem, abstract system behavior, state and explore the relationship among the components of the system, understand system essence and behavior, predict the results and utilize knowledge to help decision maker to make high quality decisions. One type of decision support system addresses the problem to select a choice from many alternatives [George, 1996]. In other words, the problem is to evaluate and rank a finite number of alternatives with respect to a finite number of criteria. Rank computation depends on the values of the criteria variables and their weight values which directly determine the influence of the variables. How to weight each criteria and how the weights influence the preference of the alternatives is a very important part in decision research. Much research has been done in this area, but most of it is subjective. The best weight value should depict the information of the data set and system behavior.

Principal component analysis (PCA) can reduce the dimensionality of the data set and simplify the interrelated variables while retaining most of the information presented

in the data set. Much research has indicated principal component analysis has an intuitively satisfying interpretation and illustrated its application in areas where judgments are not easy to come by [Ahamad, 1967; Bailey, 1956; Cahalan, 1983; Chang, 1988; Cochran and Horne, 1977; Dawkins, 1989; Jolicoeur, 1959; Jolicoeur and Mosimann, 1960; Kloek and Mennes, 1960; Lee and Chang, 1976; Rao, 1964; Sloan, 1983; Wold, 1976]. Dawkins [Dawkins, 1989], using the first principal component of the national track records from principal component analysis, ranked the world track performance. But principal components are influenced by roundoff error, sample data variation and sampling error. How the rank value changes when the weight is changed and what are the intervals of the weights with the restriction that the final ranking of the alternatives does not change? The objective of this research is to explore the application of PCA in decision support systems and investigate the model behavior under small changes in its assumption and its parameters, understand the key variables and their relationships which can most affect the model solutions and corresponding decisions, validate the model and find better and robust solutions for some particular problems. A decision support system is implemented as part of this research. It is implemented using the MS Visual C++ programming language under the MS Windows 95 environment. The system provides a graphical user interface (GUI) to view results.

The remainder of this thesis is organized as follows. Computation of PCA, application of the PCA, sensitivity analysis of the decision systems are studied in Chapter 2. Design and implementation of the system are explained in Chapter 3, also the process, class, architecture and key algorithms were briefly explained in this Chapter. The result

and the interface were shown in Chapter 4. Chapter 5 gives conclusion and some directions for future work.

CHAPTER II

Literature Review

As mentioned in the previous Chapter, decision support systems are software which are used to develop insight into system behavior which in turn help people to make effective plans and decisions. Generally, mathematical models are used to simplify the problem, describe the essence of the problem, state the relationships between decision variables, intermediate variables and outcomes. In large interrelated data information systems, reduction of dimension and simplification of the interrelated variables is the first and also the most important step to interpret the data, and thus to help people to make right decisions. In this literature review, the author will explain the theory of principal component analysis and its application in decision support systems.

Principal Component Analysis and Eigenvalues of Covariance Matrix

The central idea of principal component analysis (PCA) is to reduce the dimension of a data set which consists of a large number of interrelated variables, while retaining as much as possible the information presented in the data set [Jolliffe 1986]. Let us consider Figure 1, there are 15 observations on two highly correlated variables X, and Y. There is considerable variation in both variables, though rather more in the direction of X than Y. If the above observations are expressed in another coordinate

system, or the points are projected onto D1 and D2, then we can find that the variation in D1 is increased and variation in D2 decreased. If the observations are different in X and Y, then they are different in D1, but maybe not in D2. Also if the observations are different in D1, then they are different in X and Y. D1 is also an important direction because if two points are close on D1, then it is likely that they are close before they are projected onto D1. This is not the case in D2. Many points which are close on D2 may be originally quite far apart. So projections of the points on D1 are good representations of the above observations because we can get most of the information represented by the original data set. Why is D1 better than D2 in expressing the data? Because D1 preserves the variation of the data. In general, if the original data are expressed as

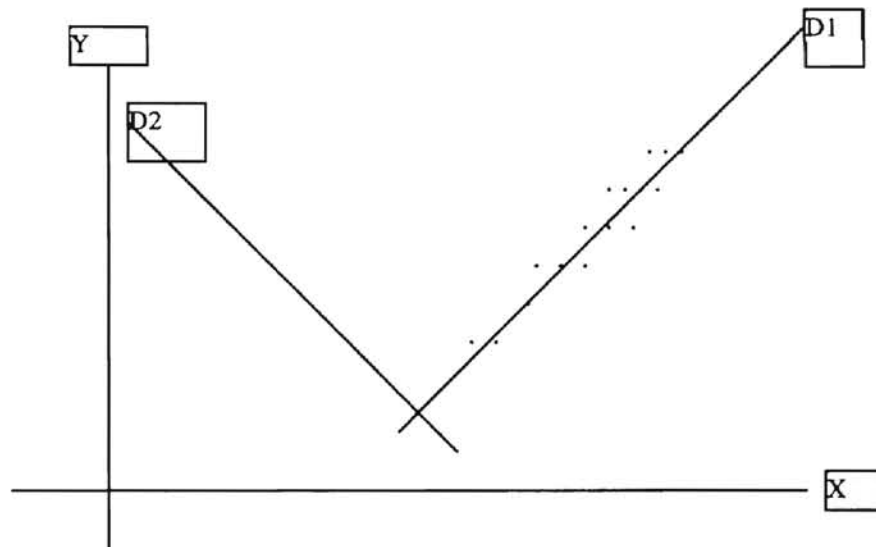


Figure 1. The simple view of PCA

vectors in a P-dimensional space, then the transformed data are vectors in a subspace of the P-dimensional space. If x is a vector of variables, then $d1$ (the projections of the

observations on D1) can be expressed as a linear combination of the components of \mathbf{x} as shown in equation (1) below:

$$d1 = \alpha_1^T \mathbf{x} \quad (1)$$

$$\text{Var}(d1) = \text{Var}(\alpha_1^T \mathbf{x}) = \alpha_1^T \text{Var}(\mathbf{x}) \alpha_1 = \alpha_1^T \Sigma \alpha_1 \quad (2)$$

Where

$d1$ is a scalar (a value of one direction in transferred coordinate system)

α_1^T is a $1 \times p$ vector

Σ is $p \times p$ population covariance matrix of \mathbf{x} .

The objective of PCA is to find the direction such that after the points are projected onto it, the variance of the projected points is maximized. In other words $\alpha_1^T \Sigma \alpha_1$ is maximized. The maximum of $\alpha_1^T \Sigma \alpha_1$ will be achieved for infinite α_1 , so a normalization constraint must be imposed for α_1 . The most convenient constraint here is $\alpha_1^T \alpha_1 = 1$.

To maximize $\alpha_1^T \Sigma \alpha_1$ subject to $\alpha_1^T \alpha_1 = 1$, use the technique of Lagrange multipliers [Jolliffe, 1986] and maximize

$$\alpha_1^T \Sigma \alpha_1 - \lambda (\alpha_1^T \alpha_1 - 1), \quad (3)$$

where λ is a Lagrange multiplier. Differentiation with respect to α_1 gives

$$\begin{aligned}\Sigma \alpha_1 - \lambda \alpha_1 &= \mathbf{0}, \text{ or} \\ (\Sigma - \lambda \mathbf{I}_p) \alpha_1 &= \mathbf{0},\end{aligned}\tag{4}$$

where $\mathbf{0}$ is $p \times 1$ vector with value of 0 for each element, and \mathbf{I}_p is the $p \times p$ identity matrix. So λ is an eigenvalue of Σ and α_1 is the corresponding eigenvector. Note that the quantity to be maximized is

$$\alpha_1^T \Sigma \alpha_1 = \alpha_1^T \lambda \alpha_1 = \lambda \alpha_1^T \alpha_1 = \lambda,\tag{5}$$

so λ must be as large as possible. Thus, α_1 is the eigenvector corresponding to the largest eigenvalue of Σ , and $\text{Var}(\alpha_1^T \mathbf{X}) = \alpha_1^T \mathbf{Var}(\mathbf{X}) \alpha_1 = \alpha_1^T \Sigma \alpha_1 = \lambda_1$, the largest eigenvalue.

A similar theory can apply for samples, for example, if n observations were collected from a population with P -dimensional random variables, let \mathbf{X} represent n observations of P -dimensional random variables, then the projections of n points on the direction of α_1 are

$$\mathbf{D}_1 = \mathbf{X} \alpha_1\tag{6}$$

Where \mathbf{D}_1 is the $n \times 1$ vector of the projections of \mathbf{X} on D_1
 \mathbf{X} is the $n \times p$ matrix which represents the original sample data

α_1 is the projection direction

$$\mathbf{D}_1^T = \alpha_1^T \mathbf{X}^T \quad (7)$$

So the variance of \mathbf{D}_1^T is

$$\mathbf{Var}(\mathbf{D}_1^T) = \mathbf{Var}(\alpha_1^T \mathbf{X}^T) = \alpha_1^T \mathbf{Var}(\mathbf{X}^T) \alpha_1 = \alpha_1^T \Sigma \alpha_1 \quad (8)$$

Where Σ is the $p \times p$ covariance matrix of sample \mathbf{X}^T . The formula (8) is the same as formula (3). Therefore, the procedure for population matrix can be used to derive the PC for the sample covariance matrix. From this we can find that the essence of PCA is actually to estimate the eigenvalues and eigenvectors of the covariance matrix.

Σ is a $p \times p$ symmetric matrix, so there exists a $p \times p$ orthogonal matrix \mathbf{P} such that $\mathbf{P}^T \Sigma \mathbf{P} = \mathbf{D}$ where \mathbf{D} is a diagonal matrix whose diagonal elements are the eigenvalues of Σ and the columns of \mathbf{P} are the normalized eigenvectors of Σ . The i^{th} column of \mathbf{P} corresponds to the i^{th} PC with variance equal to the diagonal element of \mathbf{D} for $i = 1, \dots, p$ [Moser, 1996]. The variance of \mathbf{x} (the original data) is the trace of Σ , and

$$\text{tr}(\Sigma) = \text{tr}(\Sigma \mathbf{I}) = \text{tr}(\Sigma \mathbf{P}^T \mathbf{P}) = \text{tr}(\mathbf{P}^T \Sigma \mathbf{P}) = \text{tr}(\mathbf{D}) = \sum_{i=1}^p \lambda_i$$

so the variance of the original data is equal to the sum of the eigenvalues. For any integer q ($1 \leq q \leq p$), if the eigenvectors of the largest q eigenvalues were used as the linear transformation matrix, then the variance of the transformed variables will be maximized. So the task of PCA is to find the q largest eigenvalues and their corresponding eigenvectors.

In general, let \mathbf{X} be a set of points, Σ be the covariance matrix of \mathbf{X} , the rank of Σ be p , $\mathbf{D}_1, \mathbf{D}_2, \dots, \mathbf{D}_p$ be the eigenvectors of Σ corresponding to eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_p$ where $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_p$. Let $\|\mathbf{D}_i\| = 1$ for all i . The properties of \mathbf{D}_i 's are now summarized as follows:

- 1) All \mathbf{D}_i^T s are mutually orthonormal. That is, $\mathbf{D}_i^T \mathbf{D}_j = 0$ for $i \neq j$ and $\mathbf{D}_i^T \mathbf{D}_i = 1$.
- 2) If the set \mathbf{X} of points are projected on to \mathbf{D}_i , then the variance of the projected points on \mathbf{D}_i is λ_i .
- 3) Among all possible directions, \mathbf{D}_1 is the direction which will produce the largest variance by projecting points onto it. \mathbf{D}_2 is the direction in the space perpendicular to \mathbf{D}_1 which will produce the second largest variance by projects onto it. In general, \mathbf{D}_i $1 \leq i \leq p$ is the direction in the space perpendicular to $\mathbf{D}_1, \dots, \mathbf{D}_{i-1}$ which will produce the i^{th} largest variance by projecting points onto it. Because $q \leq p$ and \mathbf{D}_i and \mathbf{D}_j , $i \neq j$, are perpendicular to each other, the PCA objective, reducing the dimension and simplifying the data, are attained.

Correlation Matrix vs. Covariance Matrix

In practice, it often occurs that different elements of the original data set are measured in completely different types of units which in turn results in widely different variance among variables. For example in the NSN data [George, 1996], the standard deviation for MAN-HR, FAILURES, COST, CANN and MIC_HRS are 1315.3927, 70.0615, 47539.789, 17.3814 and 7347.9464, respectively. The principal components based on the covariance matrix are given in Table 1.

The first component is a slight perturbation of the single variable COST which has the largest standard deviation, the second component is almost the same as the variable MIC_HRS with the second highest standard deviation, the third component is

Table 1. Principal Components Based On the Covariance Matrix for Five Variables

Component Number	1	2	3	4	5
MAN-HR (X1)	0.0275	-0.0007	0.9976	-0.0570	-0.0274
FAILURES (X2)	0.0012	0.0001	0.0539	0.9934	-0.1009
COST (X3)	0.9995	0.0159	-0.0275	0.0003	0.0006
CANN (X4)	0.0003	-0.0002	0.0329	0.0993	0.9945
MIC_HRS (X5)	-0.0159	0.9999	0.0011	-0.0002	0.0002
Eigenvalue	2.26E9	5.47E7	1.69E4	1614.75	71.7292

also almost the same as the variable MAN_HRS with the third highest standard deviation, and so on. Also the eigenvalues for components almost equal the variances of the corresponding variables. The variance for COST is 2.26E9, the first eigenvalue is also

2.26E9. Thus the first five components for the covariance matrix tell us almost nothing apart from the order of sizes of the variances of the original variables. Also even in the same data set, for example, the above data set, if the units of cost were changed to thousand dollars, then the variance for it will change to 2.26E3. The PCs also changed proportional to the change in the variance (Table 2). So the drawback of PCA based on the covariance matrices is the sensitivity of the PCs to the units of measurement used for each element [Jolliffe, 1986]. Also another drawback for covariance matrices is that due to the widely different variance, the covariance among the variables are relatively small which cause the loss of information for the covariance due to roundoff errors because of the inherently inaccurate computation of computer. So weighted covariance matrices are used to eliminate this shortcoming. Most of the time standardized variables (the original data divided by standard deviation of the variable) are used. Then the covariance matrix for standard variables changes to correlation matrix of the original variables. The principal components for the NSN data set using correlation matrix are listed in Table 3. The first component has moderate-sized coefficients for four of the five variables. The other components except for the second also have moderate-sized coefficients for several variables. The eigenvalue of the first PC for the correlation matrix shows that certain non-trivial linear functions of the standardized variables account for 71%, although less than proportionate, 94%, of the first PC for the covariance matrix in the original variables, proportion of the total variation in the standardized variables.

All the properties for the covariance are still valid for the correlation matrices, except that we are now considering PCs of the standardized variable, instead of the original variable [Jolliffe, 1988]. Although the PCs for the correlation matrix are from

Table 2. Principal Components Based On the Covariance Matrix for Five Variables after the Units of Cost Changed

Component Number	1	2	3	4	5
MAN-HR (X1)	-0.0191	0.9981	-0.0445	0.0015	-0.0365
FAILURES (X2)	-0.0006	0.0438	0.9937	-0.1010	-0.0189
COST (X3)	-0.0007	0.0359	-0.0043	-0.2097	0.9771
CANN (X4)	-0.0004	0.0107	0.1023	0.9725	0.2088
MIC_HRS (X5)	0.9982	0.0192	-0.0002	0.0002	0.0000
Eigenvalue	5.53E7	1.72E6	1617.15	82.04	19.29

Table 3. Principal Components Based On the Correlation Matrix for Five Variables

Component Number	1	2	3	4	5
MAN-HR (X1)	0.5142	0.0546	-0.4582	0.0438	-0.7217
FAILURES (X2)	0.4831	0.0880	0.4989	-0.7141	-0.0092
COST (X3)	0.5106	0.0609	-0.5085	-0.0113	0.6906
CANN (X4)	0.4841	-0.0357	0.5306	0.6932	0.04732
MIC_HRS (X5)	-0.0851	0.9921	0.0313	0.0866	-0.0002
Eigenvalue	3.5511	0.9881	0.2652	0.1912	0.0042

the standardized variable, the eigenvalues and eigenvectors of the correlation matrix have no simple relationship with the corresponding covariance matrix. The PCs for covariance and correlation do not give equivalent information, nor can they be derived directly from

each other [Jolliffe, 1986]. If the units for all the variables are the same, covariance are preferred for PCs.

Application of Principal Component Analysis

The beginnings of principal component analysis are probably to be found in the works of Karl Pearson in 1901 [Johnson and Wichern, 1982]. The statistical properties of principal components were investigated in detail by Hotelling in 1933 [Jolliffe, 1986]. Many researchers [Anderson, 1984; Jolliffe, 1986] have given comprehensive expositions. Since then, PCA has been applied in agriculture, biology, chemistry, climatology, demography, ecology, economics, food research, geology, psychology and quality control and other areas [Ahamad, 1967; Bailey, 1956; Cahalan, 1983; Chang, 1988; Cochran and Horne, 1977; Dawkins, 1989; Jolicoeur, 1959; Jolicoeur and Mosimann, 1966; Kloek and Mennes, 1960; Lee and Chang, 1976; Rao, 1964; Sloan, 1983; Wold, 1976]. In the following paragraphs, the author reviews some typical applications of PCA.

Principal Component Analysis combined with factor analysis was used to interpret the data in biology and economics and other areas [Johnson and Wichern, 1982; Jolliffe, 1986]. In biology, the growth of animals are determined by two different unobservable factors: genetic and environmental. Bailey [Bailey, 1956] using principal components analysis combined with factors successfully explained the morphogenetic changes of mice according the observable characters (size and weight). Principal components were also used as the intermediate step in discriminant analysis, cluster analysis and canonical correlation analysis [Duchene and Leclercq, 1988; Jeffers, 1967;

Jolliffe, 1986; Johnson and Wichern, 1982; Lachenburch, 1975; Sloan, 1983]. In these applications, the principal component analysis is used to reduce the data. Dawkins [Dawkins, 1989] used principal component analysis to find the first principal component which was used to rank the world track performance based on the national track records. The analysis has an intuitively satisfying interpretation and illustrated well the application of the principal component analysis in areas where judgments are not easy to come by.

Principal component analysis has been used in allocating multi-attribute records on several disks so as to achieve high degree of concurrency of disk accessing when responding to partial match queries [Chang, 1988]. The first principal component of a record-query incidence matrix was used to rank the records and then similar records were allocated to different disks. It was found that the average response time of retrieval was less than that for random allocation. This method is very good for parallel searching.

Principal component analysis was also used in multikey searching [Lee and Chang, 1976]. When the records are in the form of vectors and each key is in numerical form, principal component analysis can be used to create new keys from a set of old keys. These new keys were useful in narrowing down the search domain. The first principal component could be viewed as hashing addresses for the best-match searching problem. Instead of having to read in all the prototypes, one only had to read a few samples, resulting in a tremendous saving of the secondary storage device access time.

Computation of Eigenvalue and Eigenvector

Due to the differences in the properties of the matrix (symmetric and asymmetric, sparing and unsparing), requirement of calculation, flexibility of calculation, and

hardware and software availability, many methods (Power Iteration, Hessenberg and QR methods, QZ algorithm, Jacobi Method, Divide and Conquer Methods and Lanczos Methods) were invented to compute the eigenvalues and eigenvectors [Golub and Van Loan, 1993; Johnson and Arnold, 1989]. But here the author explains three methods: the Power method, the QR method and the Bisection method with Hessenberg form.

1. Power Method

The easiest method for calculating the largest eigenvalue and corresponding eigenvector is power method which uses an iterative procedure to estimate the dominant eigenvalue of a matrix. Suppose that \mathbf{A} is an $(n \times n)$ matrix and \mathbf{A} has eigenvalues $\lambda_1, \lambda_2, \lambda_3, \dots, \lambda_n$, with corresponding eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_n$; so

$$\mathbf{A}\mathbf{u}_j = \lambda_j \mathbf{u}_j \quad 1 \leq j \leq n$$

We assume that $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_n\}$ is a set of linearly independent vectors. Let us choose some initial vector \mathbf{v}_0 , where $\mathbf{v}_0 \neq \theta$, and let $\mathbf{v}_j = \mathbf{A}\mathbf{v}_{j-1}$. By our linear independence assumption we know that \mathbf{v}_0 can be expressed in the form

$$\mathbf{v}_0 = a_1 \mathbf{u}_1 + a_2 \mathbf{u}_2 + a_3 \mathbf{u}_3 + \dots + a_n \mathbf{u}_n$$

$$\mathbf{v}_1 = \mathbf{A}\mathbf{v}_0, \text{ then}$$

$$\mathbf{v}_1 = a_1 \mathbf{A}\mathbf{u}_1 + a_2 \mathbf{A}\mathbf{u}_2 + a_3 \mathbf{A}\mathbf{u}_3 + \dots + a_n \mathbf{A}\mathbf{u}_n$$

$$= a_1 \lambda_1 \mathbf{u}_1 + a_2 \lambda_2 \mathbf{u}_2 + a_3 \lambda_3 \mathbf{u}_3 + \dots + a_n \lambda_n \mathbf{u}_n$$

$$\mathbf{v}_2 = \mathbf{A}\mathbf{v}_1 = a_1 \lambda_1 \mathbf{A}\mathbf{u}_1 + a_2 \lambda_2 \mathbf{A}\mathbf{u}_2 + a_3 \lambda_3 \mathbf{A}\mathbf{u}_3 + \dots + a_n \lambda_n \mathbf{A}\mathbf{u}_n$$

$$= a_1 \lambda_1^2 \mathbf{u}_1 + a_2 \lambda_2^2 \mathbf{u}_2 + a_3 \lambda_3^2 \mathbf{u}_3 + \dots + a_n \lambda_n^2 \mathbf{u}_n$$

$$\mathbf{v}_k = \mathbf{A} \mathbf{v}_{k-1} = a_1 \lambda_1^k \mathbf{u}_1 + a_2 \lambda_2^k \mathbf{u}_2 + a_3 \lambda_3^k \mathbf{u}_3 + \dots + a_n \lambda_n^k \mathbf{u}_n$$

Now suppose the eigenvalues are ordered so that $|\lambda_1| \geq |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n|$,

$$\mathbf{v}_k = \lambda_1^k (a_1 \mathbf{u}_1 + a_2 (\lambda_2/\lambda_1)^k \mathbf{u}_2 + a_3 (\lambda_3/\lambda_1)^k \mathbf{u}_3 + \dots + a_n (\lambda_n/\lambda_1)^k \mathbf{u}_n)$$

if $|\lambda_1| > |\lambda_2|$, then the terms $(\lambda_i/\lambda_1)^k$ are small for large k , where $2 \leq i \leq n$.

If $a_1 \neq 0$, then

$$\mathbf{v}_k \approx a_1 \lambda_1^k \mathbf{u}_1$$

To obtain an estimate to λ_1 , we utilize two vectors \mathbf{v}_k and \mathbf{v}_{k+1} calculated iteratively,

where we expect that

$$\mathbf{v}_{k+1} \approx a_1 \lambda_1^{k+1} \mathbf{u}_1$$

Now if we form the quotient

$$\beta_k = \mathbf{w}^T \mathbf{v}_{k+1} / \mathbf{w}^T \mathbf{v}_k, \quad \text{where } \mathbf{w} \text{ is any vector such that } \mathbf{w}^T \mathbf{u}_1 \neq 0,$$

$$\beta_k = \mathbf{w}^T \mathbf{v}_{k+1} / \mathbf{w}^T \mathbf{v}_k \approx \lambda_1^{k+1} a_1 \mathbf{w}^T \mathbf{u}_1 / \lambda_1^k a_1 \mathbf{w}^T \mathbf{u}_1 = \lambda_1.$$

The approximation in the above formula is the essence of the power method.

With respect to the above formula, we note that the reasonable choice for \mathbf{w} is the vector \mathbf{v}_k itself. This choice leads to the approximation

$$\beta_k = \mathbf{v}_k^T \mathbf{v}_{k+1} / \mathbf{v}_k^T \mathbf{v}_k \approx \lambda_1.$$

It can be shown that if $a_1 \neq 0$ and $|\lambda_1| > |\lambda_2|$, then limit value of β_k , is,

$$\lim_{k \rightarrow \infty} \beta_k = \lambda_1.$$

Generally, a scaling method is used to find the eigenvector and avoid the overflow. The scaling is shown below:

$$\|\mathbf{v}_k\| = 1.$$

$$\mathbf{z}_{k+1} = \mathbf{A} \mathbf{v}_k$$

$$\beta_k = \mathbf{v}_k^T \mathbf{z}_{k+1} \approx \lambda_1$$

$$\mathbf{v}_{k+1} = \mathbf{z}_{k+1} / \|\mathbf{z}_{k+1}\|$$

In summary, the power method proceeds as follows:

1. Guess the initial vector \mathbf{z}_0 , $\mathbf{v}_0 = \mathbf{z}_0 / \|\mathbf{z}_0\|$.
2. Form the sequence $\mathbf{z}_k = \mathbf{A} \mathbf{v}_{k-1}$, $k=1, 2, \dots$
3. For each k calculate the $\beta_{k-1} = \mathbf{v}_{k-1}^T \mathbf{z}_k$, $\mathbf{v}_k = \mathbf{z}_k / \|\mathbf{z}_k\|$

Then β_k converges to the dominant eigenvalue, that is the largest in absolute value, of the matrix \mathbf{A} .

The power method has several severe restrictions and shortcomings. The choice of an initial vector \mathbf{v}_0 must make sure that $a_1 \neq 0$. This method is only suitable when \mathbf{A}

has a single dominant eigenvalue, $|\lambda_1| > |\lambda_2|$. Also the method can find just the dominant eigenvalue and the corresponding eigenvector. In practice, the usefulness of the power method depends on the ratio $|\lambda_2|/|\lambda_1|$, since it dictates the rate of convergence. Moreover, it is typically the case in applications where the dominant eigenvalue and eigenvector are desired. Note that the only thing required to implement the power method is a subroutine capable of computing matrix-vector products. It is not necessary to store \mathbf{A} in an n -by- n array. For this reason, the algorithm can be of interest when \mathbf{A} is large and sparse and when there is a sufficient gap between $|\lambda_1|$ and $|\lambda_2|$ [Golub and Van Loan, 1993].

2. Hessenberg and Inverse Power Method

Another basic method is to reduce \mathbf{A} to Hessenberg form, \mathbf{H} , to find the estimates of the eigenvalues of \mathbf{A} from \mathbf{H} and then apply the inverse power method to the original matrix \mathbf{A} to refine these estimates by iteration.

We know that eigenvalues are calculated using

$$p(t) = \det(\mathbf{A} - t\mathbf{I})$$

The roots of $p(t) = 0$ are the eigenvalues of \mathbf{A} . The polynomial $p(t)$ may be difficult to calculate. If \mathbf{S} is a nonsingular $n \times n$ matrix and $\mathbf{B} = \mathbf{S}^{-1}\mathbf{A}\mathbf{S}$, then the eigenvalues for \mathbf{B} are given by:

$$\begin{aligned} p(t) &= \det(\mathbf{B} - t\mathbf{I}) = \det(\mathbf{S}^{-1}\mathbf{A}\mathbf{S} - t\mathbf{I}) = \det(\mathbf{S}^{-1}\mathbf{A}\mathbf{S} - t\mathbf{S}^{-1}\mathbf{S}) \\ &= \det[\mathbf{S}^{-1}(\mathbf{A} - t\mathbf{I})\mathbf{S}] = \det(\mathbf{S}^{-1}) \det(\mathbf{A} - t\mathbf{I}) \det(\mathbf{S}) \\ &= \det(\mathbf{S}^{-1}) \det(\mathbf{S}) \det(\mathbf{A} - t\mathbf{I}) = \det(\mathbf{A} - t\mathbf{I}). \end{aligned}$$

which are the same as that for \mathbf{A} . So, \mathbf{A} is generally reduced to Hessenberg form, a simpler form to compute the eigenvalues of \mathbf{A} . Householder transformations are used to transform \mathbf{A} into Hessenberg form. The Householder matrix is

$$\mathbf{Q} = \mathbf{I} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u}$$

where \mathbf{u} is a nonzero vector in \mathbb{R}^n and \mathbf{I} is an $(n \times n)$ identity matrix.

$$\begin{aligned} \mathbf{Q}\mathbf{Q} &= (\mathbf{I} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u})(\mathbf{I} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u}) \\ &= \mathbf{I} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u} + (2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u})(2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u}) \end{aligned}$$

$$\begin{aligned}
&= \mathbf{I} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u} + 4(\mathbf{u}\mathbf{u}^T\mathbf{u}\mathbf{u}^T)/(\mathbf{u}^T\mathbf{u}\mathbf{u}^T\mathbf{u}) \\
&= \mathbf{I} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u} + 4(\mathbf{u}\mathbf{u}^T)(\mathbf{u}^T\mathbf{u})/(\mathbf{u}^T\mathbf{u}\mathbf{u}^T\mathbf{u}) \\
&= \mathbf{I} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u} + 4(\mathbf{u}\mathbf{u}^T)/(\mathbf{u}^T\mathbf{u}) = \mathbf{I}
\end{aligned}$$

So $\mathbf{Q}^{-1} = \mathbf{Q}$

$$\begin{aligned}
\mathbf{Q}\mathbf{x} &= (\mathbf{I} - 2\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u})\mathbf{x} = \mathbf{x} - 2\mathbf{u}\mathbf{u}^T\mathbf{x}/\mathbf{u}^T\mathbf{u} = \mathbf{x} - 2(\mathbf{u}^T\mathbf{x}/\mathbf{u}^T\mathbf{u})\mathbf{u} \\
&= \mathbf{x} - \gamma\mathbf{u}
\end{aligned}$$

where $\gamma = 2\mathbf{u}^T\mathbf{x}/\mathbf{u}^T\mathbf{u}$ and \mathbf{x} is a nonzero vector.

$$\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_n]$$

$$\mathbf{Q}\mathbf{A} = [\mathbf{Q}\mathbf{A}_1, \mathbf{Q}\mathbf{A}_2, \dots, \mathbf{Q}\mathbf{A}_n]$$

$$\mathbf{Q}\mathbf{A}_k = \mathbf{A}_k - \gamma_k\mathbf{u} \quad \gamma_k = 2\mathbf{u}^T\mathbf{A}_k/\mathbf{u}^T\mathbf{u}$$

$$\mathbf{u} = [u_1, u_2, u_3, \dots, u_n]^T$$

For any non-zero vector $\mathbf{v} = [v_1, v_2, v_3, \dots, v_n]^T$,

choose

$$u_1 = u_2 = u_3 = \dots = u_{k-1} = 0 \text{ and}$$

$$u_k = v_k - s$$

where

$$s = \pm (v_k^2 + v_{k+1}^2 + v_{k+2}^2 + \dots + v_n^2)^{0.5}$$

the sign of s to be chosen so that $v_k s \leq 0$

and

$$u_i = v_i \quad \text{for } i = k+1, k+2, \dots, n.$$

then

$$Qv = v - u = [v_1, v_2, \dots, v_{k-1}, s, 0, 0, \dots, 0]^T$$

so using this method the matrix A can be transformed to Hessenberg form.

$$\begin{aligned} & Q_{n-2}Q_{n-1}\dots Q_2Q_1AQ_1^{-1}Q_2^{-1}\dots Q_{n-1}^{-1}Q_{n-2}^{-1} \\ &= Q_{n-2}Q_{n-1}\dots Q_2Q_1AQ_1Q_2\dots Q_{n-1}Q_{n-2} = H \end{aligned}$$

The method to estimate the eigenvalues for a symmetric H is easier than that for a non-symmetric H . Generally bisection method is used to estimate the eigenvalues of symmetric H and QR method for non-symmetric H . Bisection and QR algorithms are explained below:

I) Bisection Method

Suppose H is an $(n \times n)$ symmetric Hessenberg matrix. Then H is tridiagonal and has the form

$$H_n = \begin{bmatrix} d_1 & b_1 & 0 & 0 & \cdot & \cdot & \cdot & 0 & 0 \\ b_1 & d_2 & b_2 & 0 & \cdot & \cdot & \cdot & 0 & 0 \\ 0 & b_2 & d_3 & b_3 & \cdot & \cdot & \cdot & 0 & 0 \\ 0 & 0 & b_3 & d_4 & b_4 & \cdot & \cdot & 0 & 0 \\ 0 & 0 & 0 & b_4 & d_5 & \cdot & \cdot & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & 0 & 0 & b_{n-2} & d_{n-1} & b_{n-1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{n-1} & d_n \end{bmatrix}$$

Suppose we define the sequence of polynomials [Jacob, 1995]

$$p_0(t) = 1$$

$$p_1(t) = d_1 - t = \det (H_1 - t\mathbf{I})$$

$$p_2(t) = (d_2 - t) p_1(t) - b_1^2 p_0(t) = \det (H_2 - t\mathbf{I})$$

$$p_i(t) = (d_i - t) p_{i-1}(t) - b_{i-1}^2 p_{i-2}(t) = \det (H_i - t\mathbf{I})$$

$$p_n(t) = (d_n - t) p_{n-1}(t) - b_{n-1}^2 p_{n-2}(t) = \det (H_n - t\mathbf{I})$$

$p_n(t)$ is the characteristic polynomial for \mathbf{H} .

If the subdiagonal entries b_1, b_2, \dots, b_{n-1} are all nonzero, then the algorithm is as follows:

1. Let c be some real number.
2. Calculate the values of $p_0(c), p_1(c), p_2(c), \dots, p_n(c)$.
3. Let $N(c)$ be the number of agreements in sign in the sequence $p_0(c), p_1(c), p_2(c), \dots, p_n(c)$.
4. $N(c)$ is equal to the number of roots of $p_n(t) = 0$ that are in the interval $[c, \infty)$.

In the event that $p_k(c) = 0$ for some k , we take the sign of $p_k(c)$ to be that of $p_{k-1}(c)$.

To use the above algorithm for computational purposes, we would first determine an interval $[a, b]$ that contains all the roots of $p_n(t) = 0$; generally

$$|\lambda| \leq \left(\sum_{i=1}^n d_i^2 + 2 \sum_{i=1}^{n-1} b_i^2 \right)^{0.5}$$

where $|\lambda|$ is the maximum range between a and b .

Next, let c be the midpoint of $[a, b]$. If $N(c) > N(b)$, then there is at least one eigenvalue in $[c, b]$. Let d be the midpoint of $[c, b]$. If $N(d) > N(b)$, then there is at least one eigenvalue in $[d, b]$; on the other hand, if $N(d) = N(b)$, then any eigenvalue in $[c, b]$ must in $[c, d]$. In this fashion, by repeatedly halving and testing subintervals we can determine a small subinterval $[r, s]$ that contains $N(r) - N(s)$ eigenvalues of \mathbf{H} . This process can be terminated when we have determined k small subintervals, $I_1, I_2, I_3, \dots, I_k$, whose union contains all the eigenvalues of \mathbf{H} . The midpoint of subintervals are the estimates of the eigenvalues.

II). QR algorithm for non-symmetric Hessenberg matrix

For a given $(n \times n)$ Hessenberg matrix \mathbf{H} , let $\mathbf{H}^{(1)} = \mathbf{H}$. For each positive integer k , using the same algorithm that transforms a $(n \times n)$ matrix to Hessenberg form transform matrix $\mathbf{H}^{(k)}$ into an upper triangular matrix $\mathbf{R}^{(k)}$. Then the matrix $\mathbf{H}^{(k)}$ can be written as:

$$\mathbf{H}^{(k)} = \mathbf{Q}^{(k)} \mathbf{R}^{(k)}, \quad (1)$$

where $\mathbf{Q}^{(k)}$ is an orthogonal matrix [Jacob, 1995] and,

$$\mathbf{Q}^{(k)} = \mathbf{Q}_{(1)}^{(k)} \mathbf{Q}_{(2)}^{(k)} \mathbf{Q}_{(3)}^{(k)} \dots \mathbf{Q}_{(n-1)}^{(k)} \quad (2)$$

Then set

$$\mathbf{H}^{(k+1)} = \mathbf{R}^{(k)} \mathbf{Q}^{(k)} \quad (3)$$

From (1), we can get

$$\mathbf{R}^{(k)} = \mathbf{Q}_{(n-1)}^{(k)} \mathbf{Q}_{(n-2)}^{(k)} \dots \mathbf{Q}_{(2)}^{(k)} \mathbf{Q}_{(1)}^{(k)} \mathbf{H}^{(k)} = (\mathbf{Q}^{(k)})^{-1} \mathbf{H}^{(k)}$$

and

$$\mathbf{H}^{(k+1)} = \mathbf{R}^{(k)} \mathbf{Q}^{(k)} = (\mathbf{Q}^{(k)})^{-1} \mathbf{H}^{(k)} \mathbf{Q}^{(k)}$$

so $\mathbf{H}^{(k+1)}$ is similar to $\mathbf{H}^{(k)}$. When the above procedure is repeated, $\mathbf{H}^{(k)}$ will converge to an upper-triangular matrix with the eigenvalues of \mathbf{H} on the diagonal.

Unfortunately, the above approach (Hessenberg Form) may lead to severe errors due to roundoff during the process of reducing the matrix to Hessenberg form. To overcome this difficulty, inverse power method is applied to the original matrix to refine the estimates by iteration. The inverse power method is explained below:

3) Inverse Power Method for the Eigenvalue Problem

The inverse power method is nothing more than power method applied to the matrix $(\mathbf{A} - \alpha\mathbf{I})^{-1}$ [Johnson et. al., 1989]. If α , estimated by \mathbf{H} , is a reasonably good estimate to an eigenvalue λ of \mathbf{A} , then several steps of the inverse power method will give a very accurate estimate to λ and a corresponding eigenvector. If λ is an eigenvalue of \mathbf{A} , then

$$\mathbf{A}\mathbf{u} = \lambda\mathbf{u}$$

$$\mathbf{A}\mathbf{u} - \alpha\mathbf{u} = \lambda\mathbf{u} - \alpha\mathbf{u}$$

$$(\mathbf{A} - \alpha\mathbf{I})\mathbf{u} = (\lambda - \alpha)\mathbf{u}$$

Since α is not an eigenvalue of \mathbf{A} , $(\mathbf{A} - \alpha\mathbf{I})$ is nonsingular; and we can write

$$(\mathbf{A} - \alpha\mathbf{I})^{-1}\mathbf{u} = (\lambda - \alpha)^{-1}\mathbf{u}$$

so $1/(\lambda - \alpha)$ is an eigenvalue of $(\mathbf{A} - \alpha\mathbf{I})^{-1}$ and \mathbf{u} is a corresponding eigenvector. Suppose \mathbf{A} has eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_n$ and α_i is a good estimates to λ_i ($1 \leq i \leq n$), the eigenvalues of $(\mathbf{A} - \alpha\mathbf{I})^{-1}$ are $\mu_1, \mu_2, \dots, \mu_n$, where

$$\mu_i = 1/(\lambda_i - \alpha_i) \text{ for } 1 \leq i \leq n$$

if $\alpha_1 \equiv \lambda_1$, then μ_1 is the dominant eigenvalue of $(A - \alpha_1 I)^{-1}$, the power method can be used to compute μ_1 ,

$$\lambda_1 = \alpha_1 + 1/\mu_1 .$$

The same procedure can be used to compute the other eigenvalues and their corresponding eigenvectors.

Sensitivity Analysis of the Estimates and Decision Systems

Sensitivity analysis consists of identifying the relatively sensitive parameters (i.e., those which can not be changed without changing the outcome), try to estimate those parameters more closely, and then select a solution which remains a good one over the range of likely values of the sensitive parameters [Hillier and Lieberman, 1986]. Due to roundoff errors and finite steps of iteration during the process of estimating parameters, the eigenvalues and their corresponding vectors will not be accurate. Also the outliers and influential observation, sampling error, even man-made error during the collection of data will also make the estimates more questionable. Then the application of these estimates in decision support systems (here in ranking decision system) will result in changing the final ranking of the alternatives. In the following sections the author will briefly explain the influence of roundoff error, outliers and influential data to the estimates and their influence on the decision system.

1). Roundoff Error

The advent of the computer has greatly increased the range of problems in which matrix theory and linear algebra are applicable to find solutions. However, every computer has computational limitations which result in a potential source of error for every arithmetic operation in the computer [Johnson et. al., 1989]. In particular, when a matrix is reduced to Hessenberg form, roundoff error will occur, and the Hessenberg matrix found by the machine will not be quite what it would be (if exact arithmetic were used). So the eigenvalues which are estimated from Hessenberg form of \mathbf{A} are not the same as those of the original matrix \mathbf{A} (and may differ substantially from the eigenvalues of \mathbf{A}). For example:

$$H = \begin{bmatrix} 1 & 0 & 0 & . & . & . & 0 & 0 \\ 1 & 1 & 0 & . & . & . & 0 & 0 \\ 0 & 1 & 1 & . & . & . & 0 & 0 \\ . & & & & & & & . \\ . & & & & & & & . \\ . & & & & & & & . \\ 0 & 0 & 0 & . & . & . & 1 & 0 \\ 0 & 0 & 0 & . & . & . & 1 & 1 \end{bmatrix},$$

$$H + E = \begin{bmatrix} 1 & 0 & 0 & . & . & . & 0 & \epsilon \\ 1 & 1 & 0 & . & . & . & 0 & 0 \\ 0 & 1 & 1 & . & . & . & 0 & 0 \\ . & & & & & & & . \\ . & & & & & & & . \\ . & & & & & & & . \\ 0 & 0 & 0 & . & . & . & 1 & 0 \\ 0 & 0 & 0 & . & . & . & 1 & 1 \end{bmatrix}$$

Then

$$\det(\mathbf{H} - t\mathbf{I}) = (1-t)^n$$

and
$$\det(\mathbf{H} + \mathbf{E} - t\mathbf{I}) = (1-t)^n + (-1)^{n+1}\epsilon$$

Suppose $n=10$ and $\epsilon = 2^{-10}$, then the eigenvalue for \mathbf{H} is equal 1 with multiplicity of 10, but the eigenvalues for $\mathbf{H} + \mathbf{E}$ are 1.5 or 0.5. A change in \mathbf{H} of amount 2^{-10} produces a 50% change in eigenvalue. But not every perturbation of entries in \mathbf{H} will lead to such a large change in the eigenvalues. Golub and Van Loan [Golub and Van Loan, 1993] has done comprehensive analysis of perturbation theory for eigenvalues and eigenvectors.

2). Influence of Outliers and Influential Observations on Estimates

During the process of the data collection, some atypical factors (systematic and random errors) may influence the values of the data set which can, but need not, have a disproportionate effect on PCs. If PCA is used blindly, then the results can be largely determined by a few influential observations [Jolliffe, 1986].

Outliers are generally viewed as observations which are a long way from, or inconsistent with, the remainder of the data [Jolliffe, 1986]. There are two kinds of outliers: the extreme data on the original variable and the data which does not conform with the correlation structure of the remainder of the data. It is impossible to detect the second outlier by looking solely at the original variables one at a time. Numerous procedures have been suggested for detecting outliers with respect to a single variables [Jolliffe, 1986]. Generally, the PCs themselves were used to detect potential outliers. Gnanadesikan and Kettering [Gnanadesikan and Kettering, 1972] found that the outliers which inflate variance and covariance can be detected from a plot of the first few PCs.

By contrast, the last few PCs may detect observations which violate the correlation structure imposed by the bulk of data, which are not apparent with respect to the original variables [Jolliffe, 1986]. But in a small sample data set, the best way to detect outliers is to compute PCs leaving out one (or more) observation(s) [Jolliffe, 1986]. The other possible methods that can be used to detect outliers are test statistics [Gnanadesikan and Kettenring, 1972; Hawkins, 1974; Jolliffe, 1986].

Outliers whose removal has a large effect are called influential observations. Whether or not an observation is influential depends on the analysis being done on the data set; observations which are influential for one type of analysis or parameter of interest may not be so for a different analysis or parameter. There are two methods which can be used to detect influence of the observations. One is removal of the observations; the other is to use influence function [Jolliffe, 1986]. The two methods match each other very well [Jolliffe, 1986]. Jolliffe [Jolliffe, 1986] also found that observations which were most influential for a particular eigenvalue need not be so for the corresponding eigenvector, and vice versa. Observations may be influential for PC in the covariance matrix, but may not be in the correlation matrix. An observation may be influential for one PC only in covariance matrix, but more than one value in correlation matrix is likely to be affected because the sum of the eigenvalues remains the same.

3. Sampling Error:

Due to the sampling variation, the eigenvalues and eigenvectors from the sample covariance matrix will differ from their underlying population counterparts. Some research on the sampling distribution of the eigenvalues and eigenvectors has been done

[Anderson, 1984]. Anderson [1963] has developed “large sample distribution theory” for the eigenvalues and eigenvectors.

4. Sensitivity of Decision Systems

Principal component analysis was applied in one type of decision system which evaluates and ranks a finite number of alternatives with respect to a finite number of criteria [George, 1996]. Rank computation depends only on the values of the critical variables. Therefore the computed weights of the critical variables directly determine the influence of the variables and the contribution of the variables to the rank computation. How to weight each criterion and how the weight influences the preference of the alternatives is a very important aspect in decision support system research. The best weight value should depict the information of the data. Principal component is a good way to evaluate objectively each criterion [Dawkins,1989]. But principal components are influenced by roundoff error, sample data variation and sampling error. How does the rank value change when the weight changes, and what are the intervals of the weights in which the final ranking of the alternatives does not change? For example, consider n alternatives with m criteria. Let the $(m \times 1)$ column vector \mathbf{A}_i denote the values for each record and let $(m \times 1)$ column vector \mathbf{W} represent the weight value of the criteria. Then the ranking value for each alternative is

$$R_i = (\mathbf{A}_i^T \mathbf{W}) / (\mathbf{1}^T \mathbf{W})$$

where $\mathbf{1}$ is the $(m \times 1)$ column vector with value of 1 for each element. The relationship of weight value and ranking can be formulated as below:

When the weight value for criteria i is $w_i \in [w_u, w_l]$, the interval of the ranking value for each alternative is $[A_i^l, A_i^u]$ where

$$A_i^u = \max ((A_i^T \mathbf{W}) / (\mathbf{1}^T \mathbf{W})) \quad \text{and}$$

$$A_i^l = \min ((A_i^T \mathbf{W}) / (\mathbf{1}^T \mathbf{W}))$$

In fact the above two formulae are linear fractional programming problems. The A_i^u and A_i^l are not necessarily upper bound or lower bound values for \mathbf{W} . Much research has been done for solving this problem [Ben-Israel, 1968; Ben-Isreal and Robers, 1970; Charnes and Cooper, 1962, 1973; Zionts, 1968]. The detailed derivation and proof will be omitted in this thesis. Ben-Israel and Charnes [Ben-Israel and Charnes, 1968] has proved that the maximum and minimum values are located at the vertices of the convex volume (denominator).

Sometimes, we may be interested in determining the intervals with the restriction that the final ranking of the alternatives does not change. Another option is following: With the restriction that final top 100, 50, 20 or 10 items in the ranking do not change, what percentage (d) of the weight \mathbf{W} can be changed?

Let

$$w_i \in [(1-d)w_i, (1+d)w_i]$$

where d is the percentage value of weight value that can be changed when the rank does not change.

Then, the largest value of d satisfies the expression:

$$\min_d \left\{ \max_d \left(\frac{A_i^T W}{\mathbf{1}^T W} - \frac{A_{i+1}^T W}{\mathbf{1}^T W} \right) > 0 \right\}$$

where i is the i th alternative in the ranking.

Also, considering a subset of the alternatives in which the change of the final ranking values is allowed, in what intervals are the weight allowed to vary, and how will these modifications effect the final ranking values in the entire set of the alternatives? A similar linear fractional programming problem can be used to solve the above problems.

Up to this point, the author briefly explained the computation and application of PCA and sensitivity analysis of the decision support systems from the theoretical point of view. In the next Chapter, the design and implementation of the software will be given.

CHAPTER III

Design and Implementation

In this thesis, the author has implemented a software package to compute ranking based on PCA. The software also implemented sensitivity analysis.

As mentioned earlier, there are many methods to estimate eigenvalues and eigenvectors and to solve linear fractional programming problem. All the methods are problem-dependent. So, the author has selected algorithms and data structures based on previous experience. For the data set representation, an observation is viewed as a class (RowClass). Intuitively, an observation is a row in a matrix, and so, the matrix can be treated as a collection of instances of RowClass.

As to the computation of eigenvalues and eigenvectors, the bisection method and the inverse power method were used due to the accuracy of these methods. The correlation matrix was transformed to a Hessenberg matrix by using Householder transformations and then the estimates of the eigenvalues were calculated using the bisection method. The estimates of the eigenvalues and their corresponding eigenvectors were refined by using inverse power method.

The computation for linear fractional programming varies depending on the conditions of the denominator and numerator. Meszaros and Rapcsak [Meszaros and Rapcsak, 1996] provided a simplex iteration method to do sensitivity analysis. This algorithm requires $O(n \log n)$ arithmetic operations. Ben-Israel and Charnes [Ben-Israel and Charnes, 1968] has proved that the maximum and minimum values of the linear fractional programming problem is located at the vertices of the convex volume. An algorithm based on the above fact is implemented in this software. Several test data were used to verify the correctness of this implementation.

The remainder of this Chapter gives the design and implementation. The software is implemented as a “project” in MS Visual C++. Software design is described in terms of C++ classes. Their relationship also is shown as a graph. Key algorithms are also described.

1. Classes

The project implements the following classes: application class, document class, main frame class, view class, some dialog classes, row class, table class, matrix class, square matrix class and sensitivity class. The main framework of the first five classes are generated by using AppWizard and ClassWizard provided by MS Visual C++ environment. The row class is designed to represent the data for each record and table class represents the whole data set. The matrix class is used to manipulate and manage the data set, for example, multiplication of the data set. The square matrix class is used to estimate the eigenvalues and eigenvectors for correlation matrix. The attributes and methods for the last five classes are listed below:

a. RowClass

Instance Variables:

```
Array (double)           // Address of an array.  
Length (Unsigned)       // Length of the array.
```

Methods:

```
RowClass(void);  
RowClass(unsigned N);           // Initialize array length to N.  
RowClass (const RowClass& OrigRow);  
/* This is copy constructor. The value of length is set to OrigRow.Length, and the  
Array member contains the address of an array that is a copy of OrigRow's array  
(or the NULL address). */  
~RowClass(void);                 // Destructor method.  
double& operator[](unsigned i);  
/* This function performs the subscript operation on a RowClass object. It returns  
the element of the array pointed to by the instance variable Array whose index is i.  
*/  
RowClass& operator=(const RowClass& RowObj);  
/* This function assigns to an instance of RowClass, a distinct copy of RowObj. */  
friend ostream& operator<<(ostream&, const RowClass& RowObj);  
/* Prints the RowObj to the output stream. */  
friend istream& operator>>(istream&, RowClass& RowObj);
```

```
/* Reads a row from the input stream into the RowObj. */
```

b. TableClass

protected:

Instance variables:

```
RowNum (unsigned)           // Number of rows in the table
ColNum (unsigned)           // Number of columns in the table
ChangeRate (double)         // Change in weight
weight (double)             // Weight value
Grid (RowClass *)           // Address of a table
```

Public:

Methods:

```
TableClass (unsigned NumRows, unsigned NumCols, Double InitVal);
```

```
TableClass (void);
```

```
/* This two constructors set RowNum to NumRows, ColNum to NumCols (their
default values are zero). */
```

```
TableClass (const TableClass& Original);
```

```
/* This copy constructor returns a copy of original object. */
```

```
~ TableClass(void);           // Destructor method, reallocation storage
```

```
RowClass& operator[] (unsigned i);
```

```
/* This method returns the ith row of an instance of TableClass. */
```

```
TableClass& operator=(const Table* Tan);
```

```

/* The RowNum and ColNum of the target object are set to Tan.RowNum and
Tan.ColNum, respectively. Its data members contain the copies of those of Tan.
*/

double RowSum( unsigned r) const;

/* This method computes the sum of the elements of the rth row */

double ColSum( unsigned r) const;

/* The function computes the sum of the elements of the rth column. */

/* Other "get" and "set" methods for the data members are also included in
TableClass. */

Boolean Load(const strings& FileName);

/* This function loads the data in the file referred by FileName into the instance
variables of the receiver. If load is successful, the function returns a value of true,
otherwise it returns the value false. */

Boolean Write(const strings& FileName) const;

/* This function writes the data from the TableClass to the file specified by
FileName. */

void WeightValue(double* weight);

/* This function sets the weight value for each criteria. */

void CalculateRankVal (void);

/* This function calculates the rank value for each record. */

void QuickSort (unsigned i);

/* This function sorts the table using ith column as key. */

```

```
friend istream& operator>>(istream& In, TableClass& InTab);
```

```
/* The table InTab is initialized appropriately with the number of rows and  
columns needed to store the input value. */
```

```
friend ostream& operator <<(ostream& out, const TableClass& T);
```

```
/* This function overloads the output operator for the TableClass object. */
```

c. class matrix:TableClass

```
friend matrix& operator+ (const matrix& mat1, const matrix& mat2);
```

```
/* This function adds two matrices. */
```

```
friend matrix& operator- (const matrix& mat1, const matrix& mat2);
```

```
/* This function subtracts the matrix named mat2 from the matrix named mat1. */
```

```
friend matrix& operator* (const matrix& mat1, const matrix& mat2);
```

```
/* This function multiplies two matrices. */
```

```
matrix& operator**(double k) const;
```

```
/* This function transforms a matrix into another matrix whose elements are kth  
powers of the original elements. */
```

```
matrix& operator/(double k) const;
```

```
/* This function returns a matrix whose elements are kth roots of the  
corresponding elements of the matrix it receives. */
```

d. SquareMatrix: public matrix

Row& DomEigenVect(void) const;

/ This function computes the eigenvector corresponding to the dominant eigenvalue. */*

double DomEigVal(void) const;

/ This function computes the dominant eigenvalue. */*

Long double Det(void) const;

/ This functions computes the determinant of a matrix */*

SquareMatrix& Diag(void) const;

/ This function transforms a square matrix into a new square matrix whose diagonal elements are the same as the original matrix and whose off diagonal elements are all zeros. */*

e. class Sensitivity

Instance Varibales:

RowClass* Weight; // The weight for attributes.

RowClass* RankValue, *HighRank, *LowRank;

// The rank, high rank and low rank values for a record.

TableClass *data; //Pointer to the start address of the data set.

SquarMat *correlation;

//Pointer to the correlation matrix of the data set.

TableElement range;

// Change in weight with the restriction that the final rank will not be changed.

Methods:

Sensitivity(void);

/ This function constructs the sensitivity class with the default values zero for scalar instance variables and NULL for address instance variables.*/*

~Sensitivity(); *// Destructor method, deallocates storage.*

/ Two functions “set” and “get” are also defined to set and get instance variables of the receiver. */*

void CalculatePC ();

// This function is used to calculate the rank for each record.

void CalculatePCInterval();

/ This function is used to estimate the maximum and minimum rank values for each record with the restriction that the weights can be changed within a range. */*

void CalculateMaxWeight();

/ This function is used to estimate the maximum range with the restriction that the final rank will not be changed. */*

2. Software architecture

The relationship among classes and information exchanges is shown in figure 2.

Figure 2 is based on the classes described in the previous section.

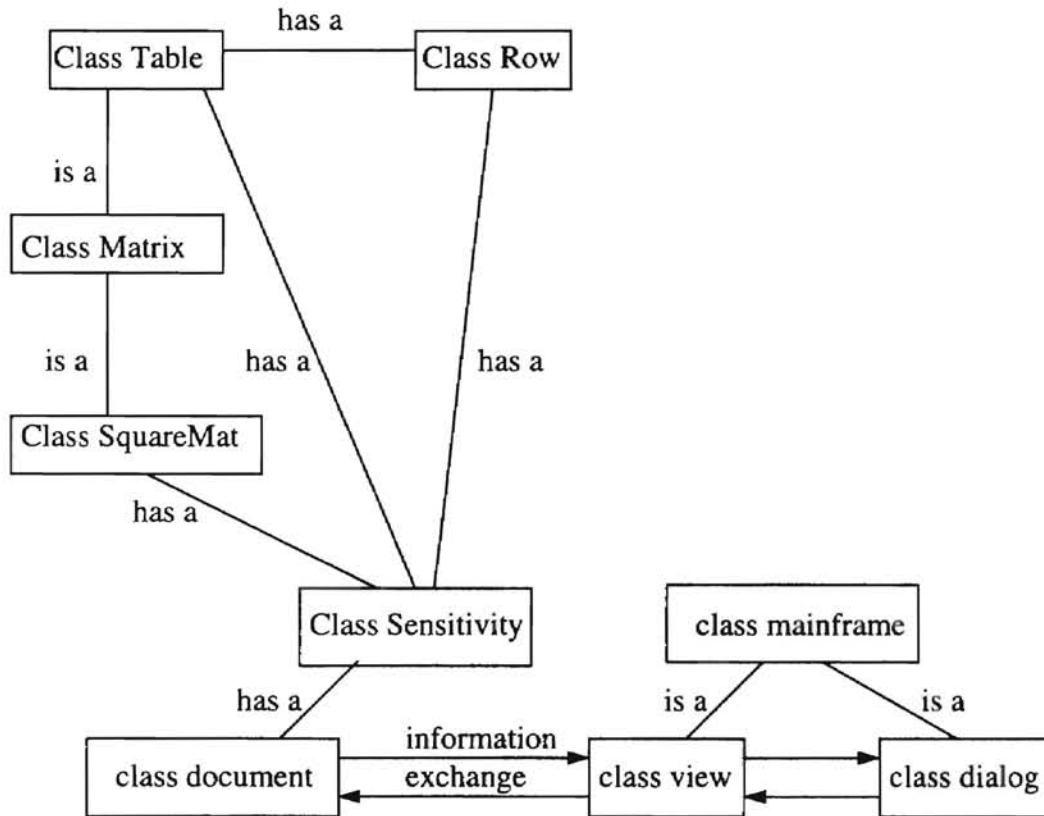


Figure 2. The relationship among the classes in the project

Each observation is stored in a row and the data set has many observations which are stored in a table. The estimated correlation matrix is stored in a square matrix represented by the class SquareMatrix. Weight values and rank values are calculated and stored in row.

3. Abstract Level Algorithm

An abstract view of the software control flow is shown in Figure 3.

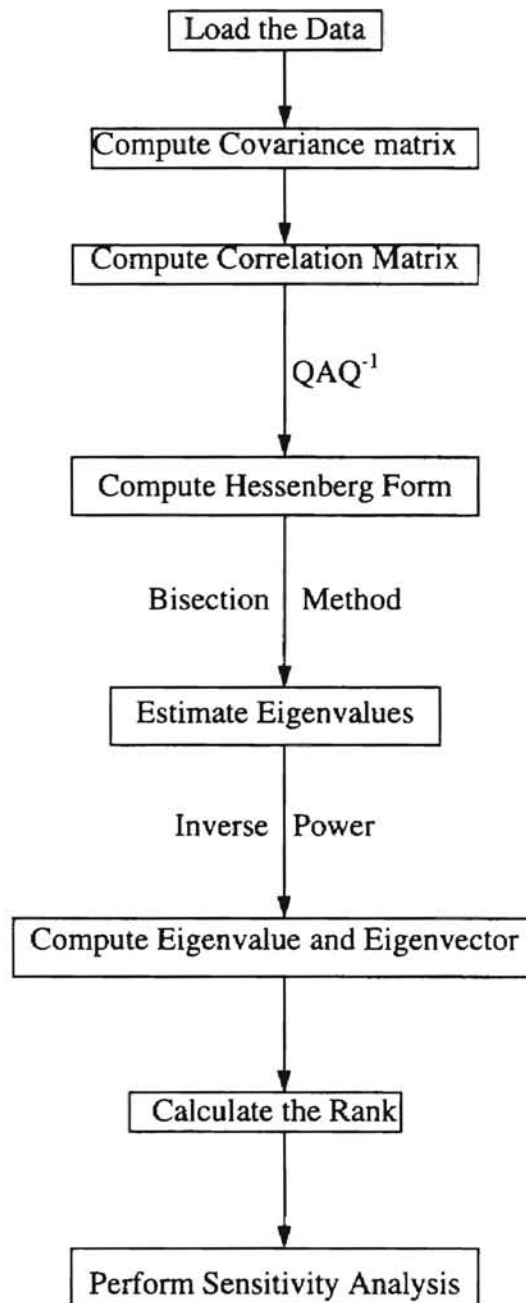


Figure 3. Abstract level control flow

A TableClass variable is declared to store the data in the table. The covariance matrix and correlation matrix are calculated for the data set. Then the eigenvector and eigenvalues were estimated for the correlation matrix which were used to calculate the rank values. Two kinds of sensitivity analyses described in Chapter 2 were done using the algorithms given in the next section.

4. Key algorithms

a. Rank value interval calculation

(adopted from Meszaro and Rapcsak [Meszaro and Rapcsak, 1996] and modified):

Input: the data set and weight value and their range for each critical variable.

Output: the data set with rank value, high rank value, low rank value.

```

For i ← 1 to n {           // n alternatives

    W ← Vl                // Vl is the low bound weight value

    G ← AiTW              // Ai is the standard data for alternative i and G is a
                           // scalar.

    H ← 1TW                // H is a scalar, the sum of weight value

    Sorting the components of Ai, determine a permutation p of (1, 2, ..., m)
    such that the sequence {Ai(p)} is monotone nonincreasing (m is the number
    attributes).

    for j ← 1 to m {      // Evaluate each criteria for each alternative

        set φ ← G/H;
    }
}

```

```

    k ← p(j);           // The jth largest component of Ai
    if Ai(k) ≤ φ then break;   // φ is the maximum value Ai
    else {
        W(k) = Vu(k) // Vu is the kth upper bound weight value.
        G ← G + Ai(k) * (Vu(k) - Vl(k));
        H ← H + (Vu(k) - Vl(k));
    }
}
}
}

```

This algorithm can calculate the $\max ((\mathbf{A}_i^T \mathbf{W}) / (\mathbf{1}^T \mathbf{W}))$, and the $\min ((\mathbf{A}_i^T \mathbf{W}) / (\mathbf{1}^T \mathbf{W}))$ is estimated by changing the sign of \mathbf{A}_i .

b. Interval weight value estimation

(adopted from Meszaro and Rapcsak [Meszaro and Rapcsak, 1996] and modified):

Input: the data set with rank value.

Output: the data set with rank value and degree of tolerable weight change.

Sort the rank value in monotone nonincreasing order. $\text{Rank}(i) \geq \text{Rank}(i+1)$

$\lambda_{\min} \leftarrow 100;$

for $i \leftarrow 1$ to $n-1$ {

```

    Di ← Ai - Ai+1 ;           // Ai the standard data of alternatives with
                                   // rank i.

```

```

    Gi ← ABS(Di) ;           // Each component in Gi is greater than or

```

```

// equal to zero.


$$\lambda \leftarrow \frac{D_i^T V}{G_i^T V} \times 100;$$

// V the weight value.

if  $\lambda \leq \lambda_{min}$  then  $\lambda_{min} \leftarrow \lambda$ ;

}

```

c. Matrix inverse computation

Input: Square matrix A of dimension n

Output: Inverse of A if it exists

```

Check the matrix's dimension;

D = det (A) ;

if (D == 0 ) return error ;

temp = A || I // A || B means the matrix [A B]

// I is the identity matrix of dimension n.

for (unsigned i=0; i<dimension; i++) {

    find the partial pivot value;

    if ( pivot row != i ) then swap the rows;

    pivot = temp[i][i];

    for (unsigned j=i; j<2*RowNum; j++)

        temp[i][j]=temp[i][j]/pivot;

    for (j=0; j<RowNum; j++) {

        if (j==i) continue;

        pivot = temp[j][i]*(-1);

```

```
        for (unsigned k=0; k<2*RowNum; k++)
            temp[j][k] +=temp[i][k]*pivot;
    }
}

// The right half of the matrix temp is the inverse of the original matrix.
```

CHAPTER IV

Results and Discussion

The data published by George [1996] were used as a sample to test the program. The first PC accounted for $3.54/5=70.8\%$ of the variation in the standard variables. The weight values of the standard variables are 0.513, 0.484, 0.513, 0.482 and 0.084 for MAN_HR(), FAILURES(), COST(), CANN() and MIC_HRS() respectively. In fact the first PC is the average of the first 4 standardized variables (the original value of the variable divided by their standard deviation). The importance, (i.e. the correlation between the variable and the first PC [Sarkar, handout in STAT5063, 1997]), for each standard variables are 0.965, 0.911, 0.965, 0.907 and 0.158 respectively. Also for the sample data set when the weight varied within 0.4%, the final rank will not be changed.

Figures 4 through 19 illustrate the interface provided by the software. They also show the results obtained using the test data. Figures 4 and 5 illustrate how to load the input data. Figure 6 shows the input data. Figures 7 through 11 show intermediate steps that can be viewed if the user prefers to view them (In the current implementation, the user is required to go through all steps.). Figure 12 shows the ranked data. Figure 13 illustrates the pull down window interface for sensitivity analysis. Figure 14

illustrates the window for specifying weight range for sensitivity analysis. Figure 15 shows the minimum and maximum rank values for the change specified for weight values in Figure 14. Figures 16 and 17 show the sorting facility. Figures 18 and 19 show for each item the percent of weight value for which the ranking will not change.

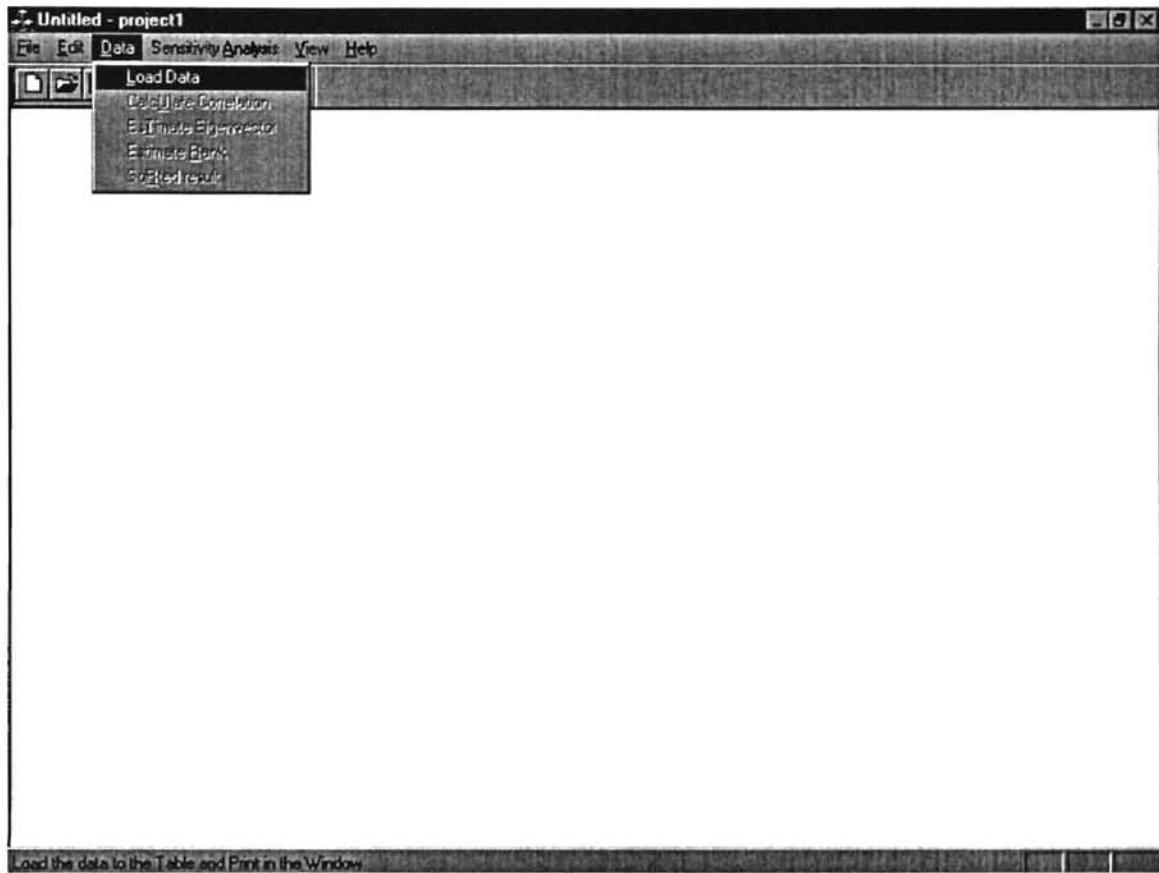


Figure 4. The window before loading data



Figure 5. The File open window

Untitled - project1

File Edit Data Sensitivity Analysis View Help

col: 5, row: 39

MAN_HR	FAILURE	COST	CANN	MIC_HRS
6435.00	317.00	232764.00	68.00	7538.00
3960.00	174.00	143244.10	47.00	1166.00
0.00	0.00	0.00	0.00	43530.00
2877.00	186.00	104050.20	55.00	936.00
1051.00	196.00	38000.20	11.00	9581.00
3327.00	91.00	120330.40	32.00	171.00
1861.00	144.00	67323.20	49.00	0.00
1009.00	181.00	36495.50	6.00	3464.00
2520.00	48.00	91137.50	9.00	4318.00
1957.00	100.00	70799.20	30.00	26.00
1511.00	104.00	54638.40	38.00	1156.00
1680.00	73.00	60765.60	21.00	1530.00
844.00	109.00	30520.20	27.00	3415.00
1560.00	79.00	56407.10	21.00	377.00
1560.00	79.00	56407.10	21.00	0.00
1024.00	84.00	37048.90	24.00	0.00
758.00	51.00	27409.60	13.00	4650.00
0.00	0.00	0.00	0.00	14643.00
662.00	84.00	23962.60	38.00	333.00
827.00	58.00	29905.40	12.00	60.00
1368.00	20.00	49473.30	0.00	240.00
875.00	51.00	31634.30	11.00	25.00
0.00	0.00	0.00	0.00	10716.00
317.00	38.00	11473.10	21.00	3119.00
266.00	31.00	9628.50	18.00	2890.00
266.00	33.00	9639.30	11.00	2593.00
649.00	18.00	23401.60	10.00	663.00
0.00	0.00	0.00	0.00	5542.00
430.00	18.00	15567.60	10.00	0.00
222.00	35.00	8037.00	8.00	0.00
669.00	2.00	24205.00	0.00	0.00
223.00	33.00	8062.30	1.00	123.00

Ready

Figure 6. Data loading window

Untitled - project1

File Edit Data Sensitivity Analysis View Help

Load Data
 Calculate Correlation
 Estimate Eigenvalues
 Estimate Rank
 Sorted result

col		COST	CANN	MIC_HRS
M		232764.80	68.00	7538.00
64		143244.10	47.00	1166.00
39		0.00	0.00	43530.00
	0.00	186.00	104050.20	55.00
	2877.00	196.00	38000.20	11.00
	1051.00	91.00	120330.40	32.00
	3327.00	144.00	67323.20	49.00
	1861.00	181.00	36495.50	6.00
	1009.00	48.00	91137.50	9.00
	2520.00	100.00	70799.20	30.00
	1957.00	104.00	54638.40	38.00
	1511.00	73.00	60765.60	21.00
	1680.00	109.00	30520.20	27.00
	844.00	79.00	56407.10	21.00
	1560.00	79.00	56407.10	21.00
	1560.00	84.00	37048.90	24.00
	1024.00	51.00	27409.60	13.00
	758.00	0.00	0.00	0.00
	0.00	84.00	23962.60	38.00
	662.00	58.00	29905.40	12.00
	827.00	20.00	49473.30	0.00
	1368.00	51.00	31634.30	11.00
	875.00	0.00	0.00	0.00
	0.00	38.00	11473.10	21.00
	317.00	31.00	9628.50	18.00
	266.00	33.00	9639.30	11.00
	266.00	18.00	23481.60	10.00
	649.00	0.00	0.00	0.00
	0.00	18.00	15567.60	10.00
	430.00	35.00	8037.00	0.00
	222.00	2.00	24205.00	0.00
	669.00	33.00	8062.30	1.00
	223.00			123.00

Calculate the correlation matrix for the table

Figure 7. The window before calculating correlation

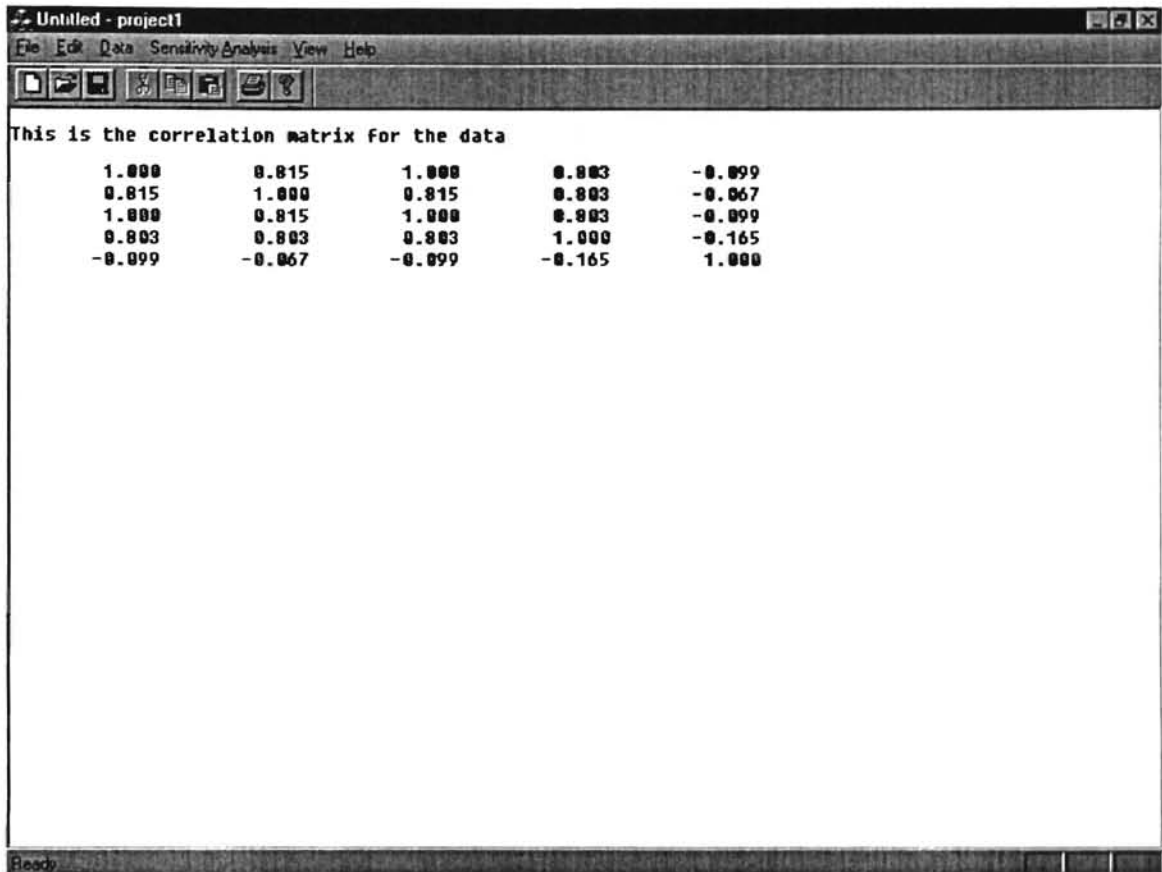


Figure 8. Correlation coefficient window

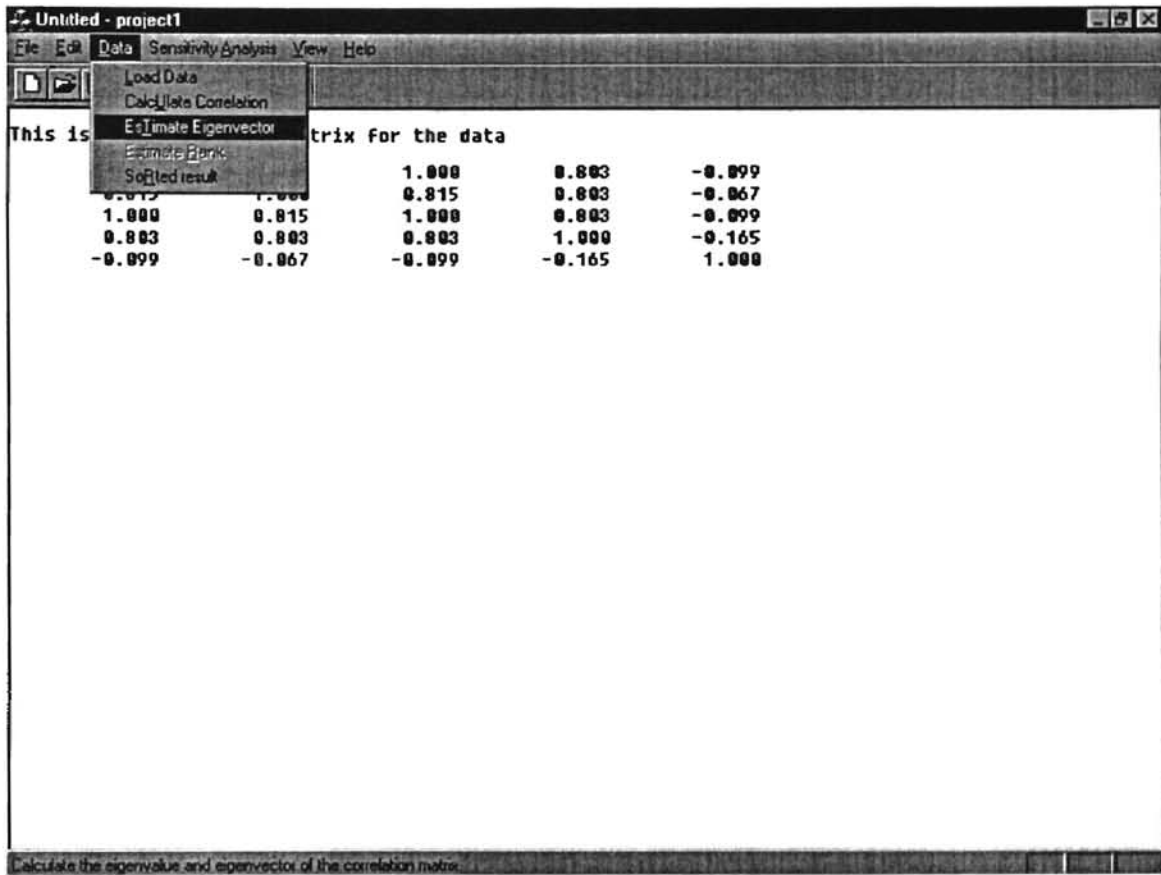


Figure 9. The window before estimating eigenvalue and eigenvector

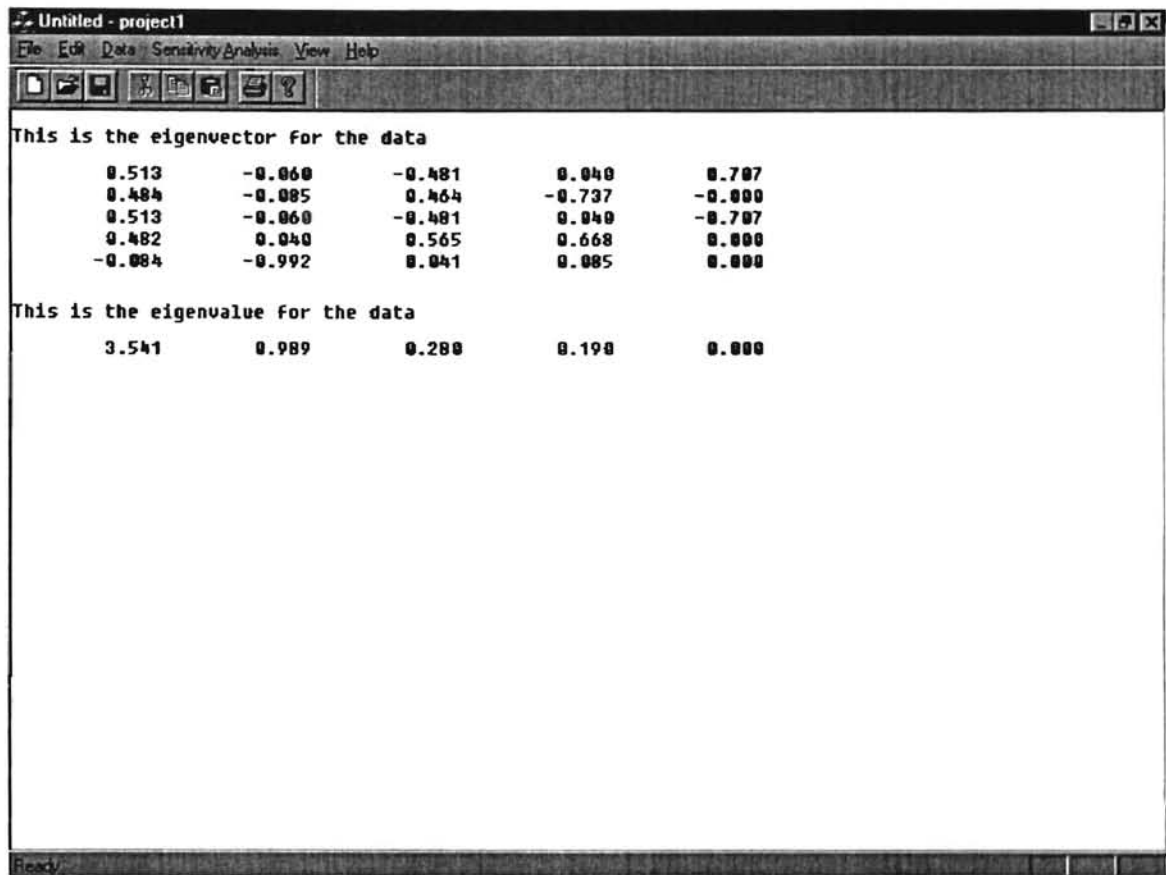


Figure 10. Eigenvalue and eigenvector window

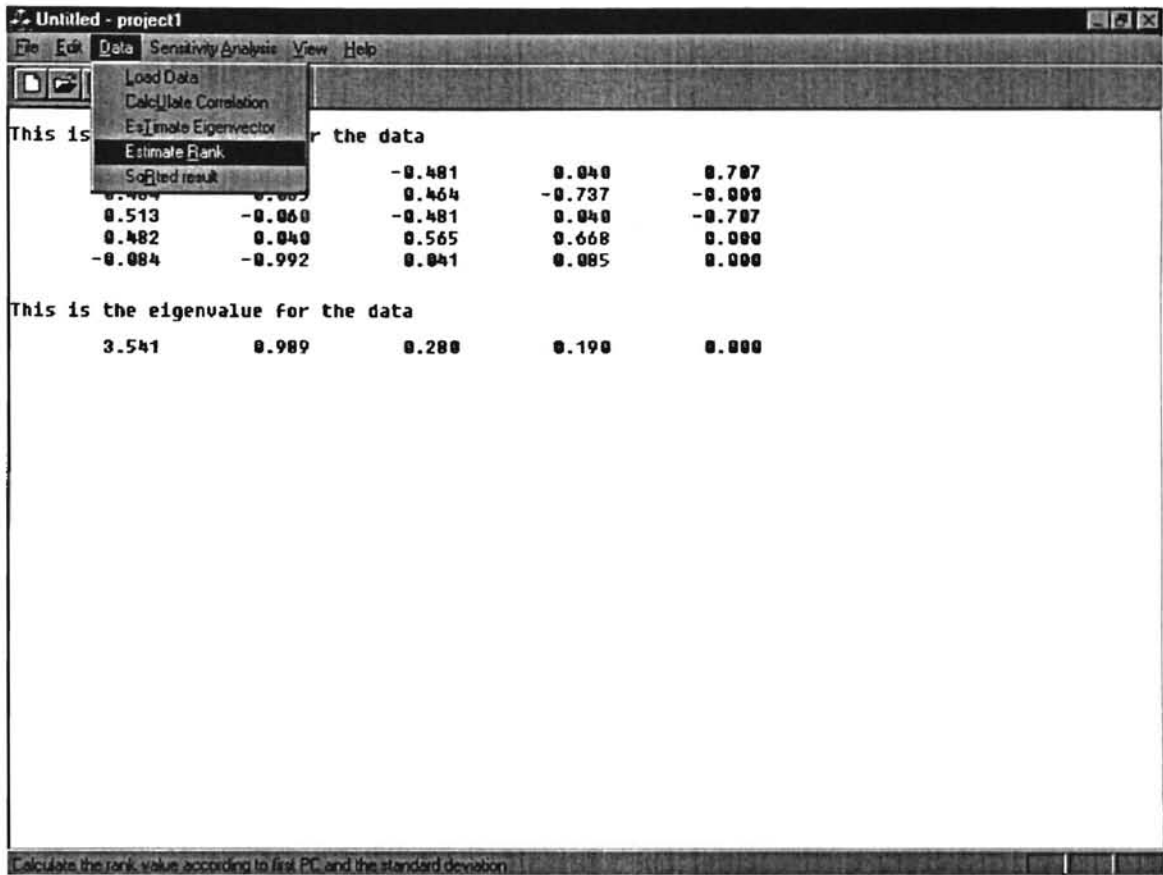


Figure 11. The window before calculating rank value

Untitled - project1

File Edit Data Sensitivity Analysis View Help

MAN_HR	FAILURE	COST	CANN	MIC_HRS	Rank
6435.00	317.00	232764.80	68.00	7538.00	1.49
3960.00	174.00	143244.10	47.00	1166.00	0.91
0.00	0.00	0.00	0.00	43530.00	0.00
2877.00	186.00	104050.20	55.00	936.00	0.82
1051.00	196.00	38000.20	11.00	9581.00	0.42
3327.00	91.00	120330.40	32.00	171.00	0.67
1861.00	144.00	67323.20	49.00	0.00	0.62
1009.00	181.00	36495.50	6.00	3464.00	0.36
2520.00	48.00	91137.50	9.00	4318.00	0.42
1957.00	100.00	70799.20	30.00	26.00	0.49
1511.00	104.00	54638.40	38.00	1156.00	0.48
1680.00	73.00	60765.60	21.00	1530.00	0.39
844.00	109.00	30520.20	27.00	3415.00	0.36
1560.00	79.00	56407.10	21.00	377.00	0.38
1560.00	79.00	56407.10	21.00	0.00	0.38
1024.00	84.00	37048.90	24.00	0.00	0.33
758.00	51.00	27409.60	13.00	4650.00	0.22
0.00	0.00	0.00	0.00	14643.00	0.03
662.00	84.00	23962.60	38.00	333.00	0.35
827.00	58.00	29905.40	12.00	60.00	0.22
1368.00	20.00	49473.30	0.00	240.00	0.20
875.00	51.00	31634.30	11.00	25.00	0.22
0.00	0.00	0.00	0.00	10716.00	0.02
317.00	38.00	11473.10	21.00	3119.00	0.18
266.00	31.00	9628.50	18.00	2890.00	0.15
266.00	33.00	9639.30	11.00	2593.00	0.12
649.00	18.00	23481.60	10.00	663.00	0.15
0.00	0.00	0.00	0.00	5542.00	0.01
430.00	18.00	15567.60	10.00	0.00	0.12
222.00	35.00	8037.00	0.00	0.00	0.10
669.00	2.00	24205.00	0.00	0.00	0.09
223.00	33.00	8062.30	1.00	123.00	0.07
154.00	6.00	5562.90	10.00	1684.00	0.07

Ready

Figure 12. The data and rank window

Untitled - project1

File Edit Data Sensitivity Analysis View Help

Interval for Rank
Interval for Weight

MAN HR	FRT CORE	COST	CANH	MIC HRS	Rank
6435.00	317.00	232764.80	68.00	7538.00	1.49
3960.00	174.00	143244.10	47.00	1166.00	0.91
0.00	0.00	0.00	0.00	43530.00	0.00
2877.00	186.00	104050.20	55.00	936.00	0.82
1051.00	196.00	38000.20	11.00	9581.00	0.42
3327.00	91.00	120330.40	32.00	171.00	0.67
1861.00	144.00	67323.20	49.00	0.00	0.62
1009.00	181.00	36495.50	6.00	3464.00	0.36
2520.00	48.00	91137.50	9.00	4318.00	0.42
1957.00	180.00	70799.20	30.00	26.00	0.49
1511.00	104.00	54638.40	38.00	1156.00	0.48
1680.00	73.00	60765.60	21.00	1530.00	0.39
844.00	109.00	30520.20	27.00	3415.00	0.36
1560.00	79.00	56407.10	21.00	377.00	0.38
1560.00	79.00	56407.10	21.00	0.00	0.38
1024.00	84.00	37040.90	24.00	0.00	0.33
758.00	51.00	27409.60	13.00	4650.00	0.22
0.00	0.00	0.00	0.00	14643.00	0.03
662.00	84.00	23962.60	38.00	333.00	0.35
827.00	58.00	29905.40	12.00	60.00	0.22
1368.00	20.00	49473.30	0.00	240.00	0.20
875.00	51.00	31634.30	11.00	25.00	0.22
0.00	0.00	0.00	0.00	10716.00	0.02
317.00	38.00	11473.10	21.00	3119.00	0.18
266.00	31.00	9628.50	18.00	2890.00	0.15
266.00	33.00	9639.30	11.00	2593.00	0.12
649.00	18.00	23481.60	10.00	663.00	0.15
0.00	0.00	0.00	0.00	5542.00	0.01
430.00	18.00	15567.60	10.00	0.00	0.12
222.00	35.00	8037.00	0.00	0.00	0.10
669.00	2.00	24205.00	0.00	0.00	0.09
223.00	33.00	8062.30	1.00	123.00	0.07
154.00	6.00	5562.90	18.00	1684.00	0.07

Using Fraction Linear Program to calculate the maximum and minimum rank value when the weight value vary with a specific range

Figure 13. The window before calculating rank interval

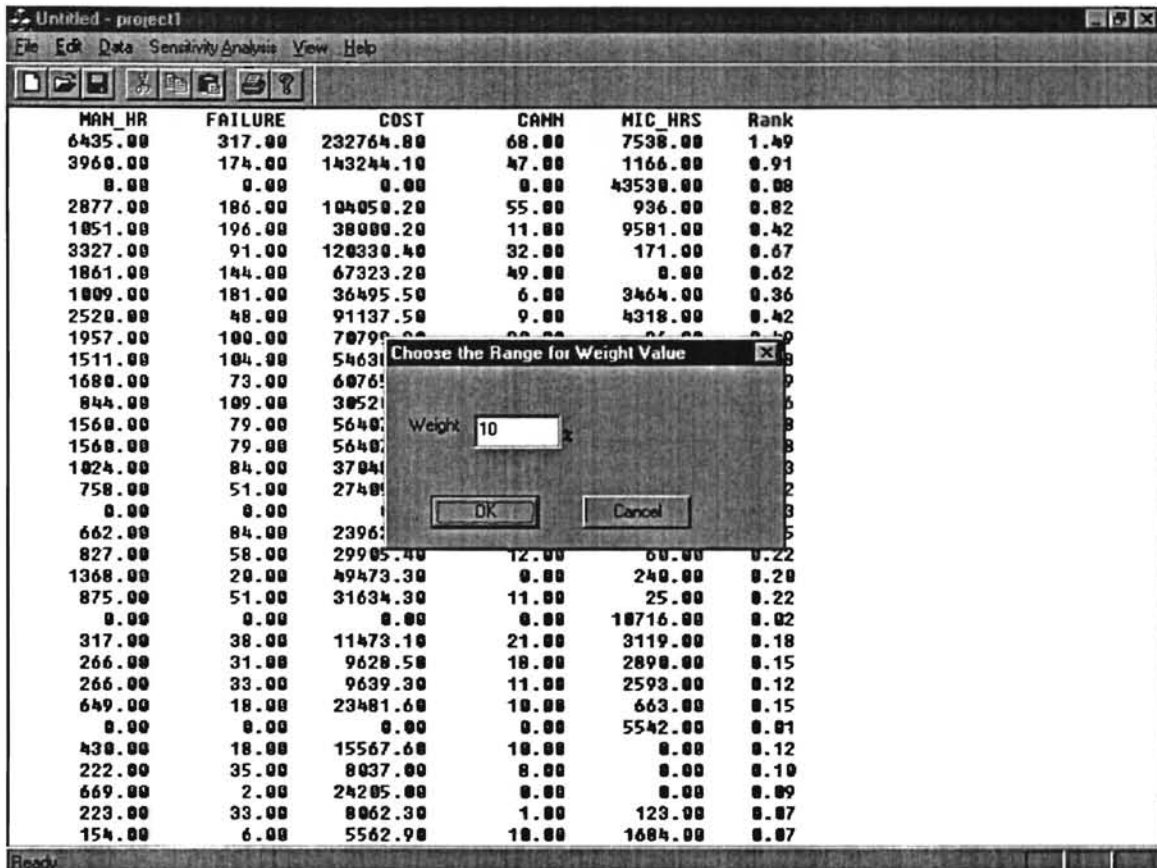


Figure 14. Weight choose window

MAN_HR	FAILURE	COST	CANH	MIC_HRS	Rank	H_Rank	L_Rank
6435.00	317.00	232764.00	68.00	7538.00	1.49	1.50	1.22
3960.00	174.00	143244.10	47.00	1166.00	0.91	0.92	0.74
0.00	0.00	0.00	0.00	43530.00	0.00	0.10	0.07
2877.00	186.00	104050.20	55.00	936.00	0.82	0.83	0.67
1051.00	196.00	38000.20	11.00	9581.00	0.42	0.45	0.38
3327.00	91.00	120330.40	32.00	171.00	0.67	0.68	0.56
1861.00	144.00	67323.20	49.00	0.00	0.62	0.64	0.55
1009.00	181.00	36495.50	6.00	3464.00	0.36	0.39	0.33
2520.00	48.00	91137.50	9.00	4318.00	0.42	0.44	0.36
1957.00	100.00	70799.20	30.00	26.00	0.49	0.50	0.40
1511.00	104.00	54638.40	38.00	1156.00	0.48	0.49	0.43
1680.00	73.00	60765.60	21.00	1530.00	0.39	0.39	0.32
844.00	109.00	30520.20	27.00	3415.00	0.36	0.37	0.31
1560.00	79.00	56407.10	21.00	377.00	0.38	0.38	0.31
1560.00	79.00	56407.10	21.00	0.00	0.38	0.38	0.31
1024.00	84.00	37048.90	24.00	0.00	0.33	0.34	0.30
758.00	51.00	27409.60	13.00	4650.00	0.22	0.22	0.20
0.00	0.00	0.00	0.00	14643.00	0.03	0.03	0.02
662.00	84.00	23962.60	38.00	333.00	0.35	0.37	0.30
827.00	58.00	29905.40	12.00	60.00	0.22	0.23	0.18
1368.00	20.00	49473.30	0.00	240.00	0.20	0.21	0.16
875.00	51.00	31634.30	11.00	25.00	0.22	0.22	0.18
0.00	0.00	0.00	0.00	10716.00	0.02	0.02	0.02
317.00	38.00	11473.10	21.00	3119.00	0.18	0.19	0.16
266.00	31.00	9628.50	18.00	2890.00	0.15	0.16	0.14
266.00	33.00	9639.30	11.00	2593.00	0.12	0.13	0.11
649.00	18.00	23481.60	10.00	663.00	0.15	0.15	0.13
0.00	0.00	0.00	0.00	5542.00	0.01	0.01	0.01
438.00	18.00	15567.60	10.00	0.00	0.12	0.12	0.10
222.00	35.00	8037.00	0.00	0.00	0.10	0.11	0.09
669.00	2.00	24205.00	0.00	0.00	0.09	0.10	0.07
223.00	33.00	8062.30	1.00	123.00	0.07	0.07	0.06
154.00	6.00	5562.90	10.00	1604.00	0.07	0.08	0.07

Figure 15. Data and rank interval window

Untitled - project1

File Edit Data Sensitivity Analysis View Help

MAN_HR	FAILURE	COST	CANH	MIC_HRS	Rank	H_Rank	L_Rank
6435.00	317.00	232764.80	68.00	7538.00	1.49	1.50	1.22
3960.00	174.00	143244.10	47.00	1166.00	0.91	0.92	0.74
0.00	0.00	0.00	0.00	43530.00	0.08	0.10	0.07
2877.00	186.00	104050.20	55.00	936.00	0.82	0.83	0.67
1851.00	196.00	38000.20	11.00	9581.00	0.42	0.45	0.38
3327.00	91.00	120330.40	32.00	171.00	0.67	0.68	0.56
1861.00	144.00	67323.20	49.00	0.00	0.62	0.64	0.55
1809.00	181.00					0.39	0.33
2520.00	48.00					0.44	0.36
1957.00	100.00					0.50	0.40
1511.00	104.00					0.49	0.43
1680.00	73.00					0.39	0.32
844.00	109.00					0.37	0.31
1560.00	79.00					0.38	0.31
1560.00	79.00					0.38	0.31
1024.00	84.00					0.34	0.30
758.00	51.00					0.22	0.20
0.00	0.00					0.03	0.02
662.00	84.00					0.37	0.30
827.00	58.00					0.23	0.18
1368.00	20.00					0.21	0.16
875.00	51.00					0.22	0.18
0.00	0.00	0.00	0.00	10716.00	0.02	0.02	0.02
317.00	38.00	11473.10	21.00	3119.00	0.18	0.19	0.16
266.00	31.00	9628.50	18.00	2890.00	0.15	0.16	0.14
266.00	33.00	9639.30	11.00	2593.00	0.12	0.13	0.11
649.00	18.00	23481.60	10.00	663.00	0.15	0.15	0.13
0.00	0.00	0.00	0.00	5542.00	0.01	0.01	0.01
430.00	18.00	15567.60	10.00	0.00	0.12	0.12	0.10
222.00	35.00	8037.00	8.00	0.00	0.10	0.11	0.09
669.00	2.00	24205.00	0.00	0.00	0.09	0.10	0.07
223.00	33.00	8062.30	1.00	123.00	0.07	0.07	0.06
154.00	6.00	5562.90	10.00	1684.00	0.07	0.08	0.07

Choose the Order Column and Order Rule

Column: 6

Order Rule:

ASCEND

DSCEND

OK

Cancel

Ready

Figure 16. Data sort dialog and window

Untitled - project1

File Edit Data Sensitivity Analysis View Help

MAN_HR	FAILURE	COST	CANH	MIC_HRS	Rank	H_Rank	L_Rank
6435.00	317.00	232764.80	68.00	7538.00	1.49	1.50	1.22
3960.00	174.00	143244.10	47.00	1166.00	0.91	0.92	0.74
2877.00	186.00	104050.20	55.00	936.00	0.82	0.83	0.67
3327.00	91.00	120330.40	32.00	171.00	0.67	0.68	0.56
1861.00	144.00	67323.20	49.00	0.00	0.62	0.64	0.55
1511.00	104.00	54638.40	38.00	1156.00	0.48	0.49	0.43
1957.00	100.00	70799.20	30.00	26.00	0.49	0.50	0.40
2520.00	48.00	91137.50	9.00	4310.00	0.42	0.44	0.36
1051.00	196.00	30000.20	11.00	9501.00	0.42	0.45	0.38
1680.00	73.00	60765.60	21.00	1530.00	0.39	0.39	0.32
1560.00	79.00	56407.10	21.00	377.00	0.38	0.38	0.31
1560.00	79.00	56407.10	21.00	0.00	0.38	0.38	0.31
1009.00	101.00	36495.50	6.00	3464.00	0.36	0.39	0.33
844.00	109.00	30520.20	27.00	3415.00	0.36	0.37	0.31
662.00	84.00	23962.60	38.00	333.00	0.35	0.37	0.30
1024.00	84.00	37048.90	24.00	0.00	0.33	0.34	0.30
827.00	58.00	29905.40	12.00	60.00	0.22	0.23	0.18
758.00	51.00	27409.60	13.00	4650.00	0.22	0.22	0.20
875.00	51.00	31634.30	11.00	25.00	0.22	0.22	0.18
1360.00	20.00	49473.30	0.00	240.00	0.20	0.21	0.16
317.00	30.00	11473.10	21.00	3119.00	0.18	0.19	0.16
266.00	31.00	9628.50	18.00	2890.00	0.15	0.16	0.14
649.00	18.00	23481.60	10.00	663.00	0.15	0.15	0.13
266.00	33.00	9639.30	11.00	2593.00	0.12	0.13	0.11
430.00	18.00	15567.60	10.00	0.00	0.12	0.12	0.10
222.00	35.00	8037.00	8.00	0.00	0.10	0.11	0.09
669.00	2.00	24205.00	0.00	0.00	0.09	0.10	0.07
0.00	0.00	0.00	0.00	43530.00	0.08	0.10	0.07
154.00	6.00	5562.90	10.00	1604.00	0.07	0.08	0.07
154.00	6.00	5562.90	10.00	23.00	0.07	0.08	0.06
223.00	33.00	8062.30	1.00	123.00	0.07	0.07	0.06
165.00	28.00	5978.90	0.00	72.00	0.05	0.06	0.05
126.00	17.00	4564.70	0.00	170.00	0.04	0.04	0.03

Ready

Figure 17. Sorted data and rank window

Untitled - project1

File Edit Data Sensitivity Analysis View Help

Interval for Rank
Interval for Weight

MAN_HR	PRICE	COST	CANN	MIC_HRS	Rank	H_Rank	L_Rank
6435.00	317.00	232764.80	68.00	7538.00	1.49	1.50	1.22
3960.00	174.00	143244.10	47.00	1166.00	0.91	0.92	0.74
2877.00	186.00	184050.20	55.00	936.00	0.82	0.83	0.67
3327.00	91.00	120330.40	32.00	171.00	0.67	0.68	0.56
1861.00	144.00	67323.20	49.00	0.00	0.62	0.64	0.55
1511.00	104.00	54638.40	38.00	1156.00	0.48	0.49	0.43
1957.00	100.00	70799.20	30.00	26.00	0.49	0.50	0.40
2520.00	48.00	91137.50	9.00	4318.00	0.42	0.44	0.36
1051.00	196.00	38008.20	11.00	9581.00	0.42	0.45	0.38
1689.00	73.00	60765.60	21.00	1530.00	0.39	0.39	0.32
1560.00	79.00	56407.10	21.00	377.00	0.38	0.38	0.31
1560.00	79.00	56407.10	21.00	0.00	0.38	0.38	0.31
1009.00	181.00	36495.50	6.00	3464.00	0.36	0.39	0.33
844.00	109.00	30520.20	27.00	3415.00	0.36	0.37	0.31
662.00	84.00	23962.60	38.00	333.00	0.35	0.37	0.30
1024.00	84.00	37048.90	24.00	0.00	0.33	0.34	0.30
827.00	58.00	29905.40	12.00	60.00	0.22	0.23	0.18
758.00	51.00	27409.60	13.00	4650.00	0.22	0.22	0.20
875.00	51.00	31634.30	11.00	25.00	0.22	0.22	0.18
1368.00	20.00	49473.30	0.00	240.00	0.20	0.21	0.16
317.00	38.00	11473.10	21.00	3119.00	0.18	0.19	0.16
266.00	31.00	9628.50	18.00	2890.00	0.15	0.16	0.14
649.00	18.00	23481.60	10.00	663.00	0.15	0.15	0.13
266.00	33.00	9639.30	11.00	2593.00	0.12	0.13	0.11
430.00	18.00	15567.60	10.00	0.00	0.12	0.12	0.10
222.00	35.00	8037.00	8.00	0.00	0.10	0.11	0.09
669.00	2.00	24205.00	0.00	0.00	0.09	0.10	0.07
0.00	0.00	0.00	0.00	43530.00	0.08	0.10	0.07
154.00	6.00	5562.90	10.00	1684.00	0.07	0.08	0.07
154.00	6.00	5562.90	10.00	23.00	0.07	0.08	0.06
223.00	33.00	8062.30	1.00	123.00	0.07	0.07	0.06
165.00	28.00	5978.90	0.00	72.00	0.05	0.06	0.05
126.00	17.00	4564.70	0.00	178.00	0.04	0.04	0.03

Calculate the range of the weight value variation with restriction that the final rank will not changed

Figure 18. The window before calculating weight interval

Untitled - project1

File Edit Data Sensitivity Analysis View Help

This is the maximum weight value: 0.40, that the final rank will not changed.

MAN HR	FAILURE	COST	CANN	MIC_HRS	Rank	H_Rank	L_Rank	WT %
6435.00	317.00	232764.80	68.00	7538.00	1.49	1.50	1.22	100.00
3960.00	174.00	143244.10	47.00	1166.00	0.91	0.92	0.74	47.10
2877.00	186.00	104050.20	55.00	936.00	0.82	0.83	0.67	57.54
3327.00	91.00	120330.40	32.00	171.00	0.67	0.68	0.56	15.52
1861.00	144.00	67323.20	49.00	0.00	0.62	0.64	0.55	83.40
1957.00	100.00	70799.20	30.00	26.00	0.49	0.50	0.40	14.06
1511.00	104.00	54638.40	38.00	1156.00	0.48	0.49	0.43	18.29
2520.00	48.00	91137.50	9.00	4318.00	0.42	0.44	0.36	0.40
1051.00	196.00	38000.20	11.00	9581.00	0.42	0.45	0.38	10.07
1680.00	73.00	60765.60	21.00	1538.00	0.39	0.39	0.32	44.12
1560.00	79.00	56407.10	21.00	377.00	0.38	0.38	0.31	100.00
1560.00	79.00	56407.10	21.00	0.00	0.38	0.38	0.31	6.44
1009.00	181.00	36495.50	6.00	3464.00	0.36	0.39	0.33	3.62
844.00	109.00	30520.20	27.00	3415.00	0.36	0.37	0.31	6.72
662.00	84.00	23962.60	38.00	333.00	0.35	0.37	0.30	16.29
1024.00	84.00	37048.90	24.00	0.00	0.33	0.34	0.30	99.80
827.00	58.00	29905.40	12.00	60.00	0.22	0.23	0.18	12.37
758.00	51.00	27409.60	13.00	4650.00	0.22	0.22	0.20	0.35
875.00	51.00	31634.30	11.00	25.00	0.22	0.22	0.18	14.58
1368.00	20.00	49473.30	0.00	240.00	0.20	0.21	0.16	5.15
317.00	38.00	11473.10	21.00	3119.00	0.18	0.19	0.16	100.00
266.00	31.00	9628.50	18.00	2890.00	0.15	0.16	0.14	6.80
649.00	18.00	23481.60	10.00	663.00	0.15	0.15	0.13	32.21
266.00	33.00	9639.30	11.00	2593.00	0.12	0.13	0.11	11.35
430.00	18.00	15567.60	10.00	0.00	0.12	0.12	0.10	29.97
222.00	35.00	8037.00	8.00	0.00	0.10	0.11	0.09	12.66
669.00	2.00	24205.00	0.00	0.00	0.09	0.10	0.07	4.22
0.00	0.00	0.00	0.00	43530.00	0.08	0.10	0.07	3.77
154.00	6.00	5562.90	10.00	1684.00	0.07	0.08	0.07	100.00
154.00	6.00	5562.90	10.00	23.00	0.07	0.08	0.06	1.69
223.00	33.00	8062.30	1.00	123.00	0.07	0.07	0.06	100.00

Ready

Figure 19. Data, rank interval and weight interval window

CHAPTER V

Conclusion and Future Work

In this thesis, a decision support system based on PCA is developed. The system provides a GUI to view results. The software is implemented using MS Visual C++. The software loads data from a file and ranks them. It also provides methods for performing sensitivity analysis on the ranking.

From the sample data we can find this method is a good way to objectively evaluate and interpret the data to generate accurate and correct information for a manager to make effective decisions. Due to the limitations on accessing actual data sets, the author could not perform extensive tests of the model. Further tests and enhancements are suggested as future work.

References

- Ahamad, B. An Analysis of Crimes by the Method of Principal Components. *Appl. Statist.*, 16, 17-35, 1967.
- Anderson, T. W. Asymptotic Theory for Principal Component Analysis. *Annals of Mathematical Statistics*, 34, 122-148, 1963.
- Anderson, T. W. *An Introduction to Multivariate Statistical Analysis*, New York: John Wiley, 1984.
- Bailey, D. W. A Comparison of Genetic and Environmental Principal Components of Morphogenesis in Mice. *Growth*, 20, 63-74, 1956.
- Beliveau, J. G., S. Cogan, G. Lallement and F. Ayer. Iterative Least-Squares Calculation for Modal Eigenvector Sensitivity. *AIAA Journal*, Vol. 34 (2), 385-391, 1996.
- Ben-Israel, A. and P. D. Robers. A Decomposition Method For Interval Linear Programming. *Management Science*, Vol. 16, 374-387, 1970.
- Ben-Israel, A. and A. Charnes. An Explicit Solution of A Special Class of Linear Programming Problems. *Operations Research* 16, 1166-1175, 1968.
- Bru, M. F. Diffusions of Perturbed Principal Component Analysis. *J. of Multivariate Analysis* 29, 127-136, 1989.
- Cahalan, R. F. EOF Spectral Estimation in Climate Analysis. *Second International Meeting on Statistical Climatology, Preprints*, 4.5.1-4.5.7, 1983.
- Caussinus, H. and L. Ferre. Comparing the Parameters of a Model for Several Units by Means of Principal Component Analysis. *Computational Statistics & Data Analysis*. 13, 269-280, 1992.
- Chang, C. C. Application of Principal Component Analysis to Multi-Disk Concurrent Accessing. *BIT* 28 205-214, 1988.
- Charnes, A. and W. W. Cooper. An explicit General Solution in Linear Fractional Programming. *Naval Research Logistics Quarterly* 20, 449-467, 1973.

- Charnes, A. and W. W. Cooper. Programming with Linear Functional Functionals. *Nav. Res. Log. Quart.* 9, 181-16, 1962.
- Cochran, R. N. and Horne, F. H. Statistically weighted principal component analysis of rapid scanning wavelength kinetics experiments. *Anal. Chem.*, 49, 846-853, 1977.
- Dauxois, J., A. Pousse and Y. Romain. Asymptotic Theory for the Principal Component Analysis of a Random Function: Some Applications to Statistical Inference. *J. of Multivariate Analysis* 12, 136-154, 1982.
- Dawkins, B. Multivariate Analysis of National Track Records. *The American Statistician*, 43, 110-115, 1989.
- Duchene, J. and S. Leclercq. An Optimal Transformation for Discriminant and Principal Component Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 10, 978-983, 1988.
- George, K. M. Computer Method for Sustainability Ranking. AFOSR report, 1996.
- Gnanadesikan, R and J. R. Kettenring. Robust Estimates, residuals, and outlier detection with multiresponse data. *Biometrics*, 28, 81-124, 1972.
- Golub, G. H. and C. F. Van Loan. *Matrix Computations*. Second edition. The Johns Hopkins University Press, Baltimore, 1993.
- Green, B. F. Parameter sensitivity in multivariate methods. *J. Multiv. Behav. Res.*, 12, 263-287, 1977.
- Hawkins, D. M. The detection of errors in multivariate data using principal component analysis. *Appl. Statist.*, 69, 340-344, 1974.
- Hillier, F. S. and G. J. Lieberman. *Introduction to Operations Research*. Holden-Day, Inc., Oakland, CA, 1986.
- Hotelling, H. Analysis of a Complex of Statistical Variables into Principal Components. *Journal of Educational Psychology*, 24, 417-441, 1933.
- Jacob, B. *Linear Functions and Matrix Theory*. Springer-Verlag, 1995.
- Jeffers, J. N. R. Two case studies in the application of principal component analysis. *Appl. Statist.*, 16, 225-236, 1967.
- Johnson, L. W., R. D. Riess and J. T. Arnold. *Introduction to Linear Algebra*. Addison-Wesley Pub., 1989.

- Johnson, R. A. and D. W. Wichern. *Applied Multivariate Statistical Analysis*. Prentice-Hall, Inc., Englewood Cliffs, 1982.
- Jolicoeur, P. Multivariate Geographical Variation in the Wolf *Canis Lupus L.* *Evolution*, 13, 283-299, 1959.
- Jolicoeur, P. and J. E. Mosimann. Size and Shape Variation in the Painted Turtle: A Principal Component Analysis. *Growth*, 24, 339-354, 1966.
- Jolliffe, L. T. *Principal Component Analysis*. Springer-Verlag, New York, 1986.
- King, B. Market and Industry Factors in Stock Price Behavior. *Journal of Business*, 39, 139-190, 1966.
- Kloek, T. and L. B. M. Mennes. Simultaneous equations estimations based on principal components of predetermined variables. *Econometrica*, 28, 45-61, 1960.
- Korhonen, P. J. Subjective Principal Component Analysis. *Computational Statistics & Data Analysis* 2, 243-255, 1984.
- Krzanowski, W. J. Sensitivity of principal components. *J. Roy. Statist. Soc. B*, 46, 558-563, 1984.
- Krzanowski, W. J. Cross-Validatory Choice in Principal Component Analysis; Some Sampling Results. *J. Statist. Comput. Simul.*, Vol. 18, 299-314, 1983.
- Lachenburch, P. A. *Discriminant Analysis*. New York: Hafners Press. 1975.
- Lee, R. C. T., Y. H. Chin and S. C. Chang. Application of Principal Component Analysis to Multikey Searching. *IEEE Transactions on Software Engineering*. 2, 185-193, 1976.
- Maxwell, A. E. *Multivariate Analysis in Behavioural Research*. London: Chapman and Hall, 1977.
- Meszaros, G. and T. Rapcsak. On Sensitivity Analysis for a Class of Decision Systems. *Decision Support Systems* 16, 231-240, 1996.
- Moser, K. B. *Linear Models: A Mean Model Approach*. Academic Press, 1996.
- Rao, C. R. The Use and Interpretation of Principal Component Analysis in Applied Research. *Sankhya A*, 26, 329-358, 1964.
- Rao, C. R. *Linear Statistical Inference and Its Applications*, New York: John Wiley, 1973.

Sloan, M. A. G. Using Principal Component Analysis Prior to Agglomerative Hierarchical Clustering Methods. Ph.D. thesis. Oklahoma State University, 1983.

Steiger, D. M. Beyond What-If: Enhancing Model Analysis In a Decision Support System. Ph.D. thesis, Oklahoma State University, 1993.

Wold, S. Pattern Recognition by Mean of Disjoint Principal Components Models. *Patt. Recog.*, 8, 127-139, 1976.

Zionts, S. Programming with Linear Fractional Functionals. *Nav. Res. Log. Quart.* 15, 449-452, 1968.

VITA

Shaokai Wen

Candidate for the Degree of

Master of Science

Thesis: APPLICATION OF PRINCIPAL COMPONENT ANALYSIS TO DECISION SUPPORT SYSTEM

Major Field: Computer Science

Biographical: Born in Ningxing, Hunan Province, China, March 2, 1964, the eldest son of Wen Xibin and Jiang Qinxiu. Married to Zhang Rui, August 1, 1990.

Education: Graduated in July, 1981 from Ningxing 7th high school in Hunan, China. Received a Bachelor of Agriculture Science degree in Animal Science from Hunan Agricultural University in May, 1985, Hunan, China; received a Master of Animal Science degree from Beijing Agricultural University, Beijing, China in May, 1988; received the Doctor of Philosophy degree in Animal Breeding and Reproduction at Oklahoma State University, Stillwater in July, 1996; completed the requirements of the Master of Science at Oklahoma State University, Stillwater in July, 1997.

Professional Experience: Teaching assistant from September 1988 - January 1987 in Beijing Agricultural University, Beijing, China; research assistant from July 1985 - June 1988 in Beijing Agricultural University, Beijing, China; animal scientist from July 1988 - July 1990 in the Chinese Academy of Science, Beijing, China; assistant research professor from June 1988 - December 1992 in the Chinese Academy of Science, Beijing, China. research assistant from January 1993 - June 1996 in Oklahoma State University, Stillwater, Oklahoma. teaching assistant from August 1996 - May 1997 in Oklahoma State University, Stillwater, Oklahoma.

Professional Organization: Chinese Society of Animal Science; American Society of Animal Science, Gamma Sigma Delta