DOKUZ EYLÜL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

ON THE CONSTRUCTION OF STUDENT GROUPS IN A PROBLEM BASED LEARNING SYSTEM THROUGH FUZZY LOGIC CONSIDERING VARIOUS OBJECTIVES

by

Ayşe Övgü KINAY

March, 2008

İZMİR

ON THE CONSTRUCTION OF STUDENT GROUPS IN A PROBLEM BASED LEARNING SYSTEM THROUGH FUZZY LOGIC CONSIDERING VARIOUS OBJECTIVES

A Thesis Submitted to the

Graduate School of Natural and Applied Sciences of Dokuz Eylül University

In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Statistics Program

by

Ayşe Övgü KINAY

March, 2008

İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"ON THE CONSTRUCTION OF STUDENT GROUPS IN A PROBLEM BASED LEARNING SYSTEM THROUGH FUZZY LOGIC CONSIDERING VARIOUS OBJECTIVES"** completed by **AYŞE ÖVGÜ KINAY** under supervision of **PROF. DR. EFENDİ NASİBOĞLU** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Efendi NASİBOĞLU

Supervisor

Thesis Committee Member	Thesis Committee Member
Examining Committee Member	Examining Committee Member

Prof. Dr. Cahit HELVACI

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

Before all, I owe thanks to my very distinguished consultant Professor Dr. Efendi NASİBOĞLU who never stopped extending me his support, time and knowledge during the whole period my dissertation. I would like to thank my dear head of department and instructor Professor Dr. Serdar KURT who facilitated all hard bureaucratic procedures in my name within this period and who always believed in my ability to complete this work. Otherwise, I would like to acknowledge my dissertation committee member, Assoc. Prof. Dr. Kaan YARALIOĞLU for their constructive comments and suggestions.

I thank my dear roommate and friend Burcu ÜÇER for all her support in the completion process of this dissertation, dear Neslihan DEMİREL and Selma GÜRLER for always encouraging me, dear Gözde ULUTAGAY for sharing her knowledge and experience with me with her whole sincerity and all my close friends who again always supported me in this process.

I would like to extend my whole-hearted appreciation to my invaluable husband Rıza KINAY who has always been my greatest supporter with his patience and love and given me unlimited strength. I would also like to thank to my beloved family Canset & N. Ümit TEKİN, from whom I have learned a lot about life and who have been with me whenever I am in need during my professional life and who have always helped me in choosing the right way; my dear brother Dr. Eng. Evren TEKİN, who shared his doctorate experiences, who never stopped helping me whenever I was in need and who always encouraged me. Finally, I would like to thank to my mother-in-law Fatma KINAY and Füsun & Nedim Atilla for their love, care and generous support through the years and for their confidence in me.

Ayşe Övgü KINAY

ON THE CONSTRUCTION OF STUDENT GROUPS IN A PROBLEM BASED LEARNING SYSTEM THROUGH FUZZY LOGIC CONSIDERING VARIOUS OBJECTIVES

ABSTRACT

Fuzziness is a concept that was suggested in 1965 by Zadeh that has improved rapidly until today and that has a number of successful applications in many fields. The reason why it has such successful applications and it can be applied in many fields is that it allows expression and analysis of the problems we encounter in daily life more realistically and, thanks to this, it produces more realistic solutions to problems. Therefore, the concept of fuzziness and the theories suggested and the methods developed on this concept are gaining more and more importance day by day.

The creation of suitable learning conditions for students is of great importance in the method of problem based learning system which has been continuing in the Department of Statistics at Dokuz Eylül University since 2001. The most important of these conditions is the suitable composition of student groups for the purposes of instruction. For instance, level-based student groups can be composed by dividing students according to their success levels or balanced student groups can be constituted by students of each success level taking place in each group in approximate equal numbers. In addition, student groups can also be constituted by choosing students completely randomly. However, it is quite important that student evaluation grades, which are the fundamental elements used in the group constitution strategies mentioned here, should also be determined suitably. Especially while carrying out such performance evaluations, the opinion formed about the student is both quite difficult to turn into numerical expressions and vary according to each instructor. Thus, there exists the requirement of a system in which the student performance evaluations will be carried out verbally in a more suitable way for human structure of thinking and in which numerical results will later be obtained by using this information.

In this dissertation work, a student performance evaluation system and a student group assignment system have been developed by searching for a solution for the above-mentioned problems. Five distinct group assignment strategies have been introduced within the group assignment system. Borland C++ Builder 6.0 Software Development Kit (SDK) was used for the implementation of the mentioned methods with a view to provide a solution.

Keywords: Fuzzy logic, Performance evaluation, Optimization, Assignment problem

AKTİF EĞİTİMDE FARKLI AMAÇLAR DOĞRULTUSUNDA ÖĞRENCİ GRUPLARININ BULANIK MANTIK YARDIMIYLA OLUŞTURULMASI

ÖZ

Bulanıklık kavramı 1965 yıllında Zadeh tarafından önerilen, günümüze kadar hızla gelişme gösteren ve birçok alanda çok miktarda başarılı uygulamaları olan bir kavramdır. Bu kadar başarılı uygulamasının oluşu ve birçok alanda uygulanabilmesinin sebebi ise günlük hayatta karşılaştığımız problemleri daha gerçekçi ifade etmeyi sağlaması, analiz etmesi ve bu sayede sorunlara da daha gerçekçi çözümler üretmesidir. Dolayısıyla bulanıklık kavramı ve bu kavram üzerine önerilen teoriler, geliştirilen yöntemler günden güne daha da önem kazanmaktadır.

Dokuz Eylül Üniversitesi İstatistik Bölümü'nde 2001 yılından beri devam etmekte olan probleme dayalı öğrenim sisteminde, öğrenciler için uygun öğrenme koşullarının yaratılması çok büyük önem taşımaktadır. Bu koşullardan en önemlisi öğrenci gruplarının farklı amaçlar doğrultusunda uygun olarak oluşturulmasıdır. Buradaki "farklı amaçlar" ifadesinden kastedilen, öğretim sürecine yöneliktir. Örneğin, öğrencilerin başarı seviyelerine göre ayrılarak elde edilen seviye temelli öğrenci grupları ya da her başarı seviyesinden öğrencinin her grupta yaklaşık eşit sayılarda olmasıyla oluşacak dengeli öğrenci grupları oluşturulabilir. Bununla birlikte öğrencilerin tamamen rasgele seçilmesiyle de öğrenci grupları oluşturulabilir. Fakat burada bahsedilen grup oluşturma stratejilerinde kullanılan temel unsur olan öğrenci değerlendirme puanlarının da uygun olarak belirlenmiş olması oldukça önemlidir. Özellikle bu tür performans değerlendirmeleri yapılırken öğrenci hakkında oluşan düşüncelerin sayısal ifadelere dönüşmesi hem oldukça güçtür hem de her öğretim elemanına göre değişiklik göstermektedir. Dolayısıyla öğrenci performans değerlendirmelerinin insan düşünce yapısına daha uygun bir şekilde sözel olarak yapılacağı, daha sonra da bu bilgiler kullanılarak sayısal sonuçların elde edileceği bir sistemin gerekliliği söz konusudur.

Bu tez çalışmasında yukarıda anlatılan problemlere çözüm arayışıyla bir öğrenci performans değerlendirme sistemi ve öğrenci grup atama sistemi geliştirilmiştir.

Grup atama sistemi içinde 5 ayrı grup atama stratejisi tanıtılmıştır. Her iki ana yöntem için ise çözüm yapmayı sağlaması açısından iki ayrı Borland C++ Builder 6.0 kodu oluşturulmuştur.

Anahtar sözcükler: Bulanık mantık, Performans değerlendirmesi, Optimizasyon, Atama problemi

CONTENTS

Page

THESIS EXAMINATION RESULT FORM ii
ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZ vi

CHAPTER ONE – INTRODUCTION 1

CHAPTER TWO – FUZZY SETS AND LINGUISTIC VARIABLES 4

2.1 Definitions and Operations on Fuzzy Sets 4
2.1.1 Fundamental Set Operations on Fuzzy Sets 6
2.1.2 Fuzzy Relations 8
2.1.3 The Resolution and Extension Principle 11
2.1.4 Aggregation and Defuzzification Operations 13
2.2 Linguistic Variables and Its Constitution Methods 16
2.2.1 Linguistic Variables 16
2.2.2 Parametric Constitution Methods of Linguistic Variables 19
2.2.3 Fuzzy Clustering Approach to Constitution of Linguistic Variables . 20
2.2.3.1 Fuzzy *c*-Means 21
2.2.3.2 Cluster Validity Indexes 23

CHAPTER THREE – AN OPTIMIZATION APPROACH FOR THE EVALUATION OF STUDENT PERFORMANCES 25

3.1 Introduction 25
3.2 Fuzzy Optimization Approaches to Performance Evaluation 27
3.3 Formulation of The Performance Evaluation Problem with Linguistic Variables 29

3.3.1 Determination of The Evaluation Criteria and Their Values............ 30
3.3.2 An Optimization Formulation of The Performance Evaluation Problem.. 32
3.4 Solution Method and Algorithm of The Performance Evaluation Problem.. 33
3.4.1 An Optimal Solution of The Problem for Fixed β 33
3.4.2 An Optimal Solution of The Problem for Optimal β 35
3.4.3 An Iterative Solution Algorithm of The Problem.............................. 37

CHAPTER FOUR – GROUP CONSTITUTION PROBLEM WITH DIFFERENT STRATEGIES 39

4.1 Introduction 39
4.2 Random Group Constitution Strategies..................................... 40
4.2.1 Balanced Random Assignment.. 40
4.2.2 Simple Random Assignment 43
4.2.3 Level-Based Random Assignment 43
4.3 Deterministic Group Constitution Strategies 45
4.3.1 Balanced Assignment 45
4.3.2 Level-Based Assignment.. 48

CHAPTER FIVE – APPLICATIONS AND EXPERIMENTAL RESULTS 50

5.1 Introduction 50
5.2 Performance Evaluation Tools and Experimental Results......................... 50
5.2.1 Performance Evaluation Tools 50
5.2.1.1 Forms 50
5.2.1.2 Functional Modules 55
5.2.1.3 Informative Components 56
5.2.2 Performance Evaluation Results.. 57
5.3 Group Constitution Tools and Experimental Results.......................... 60

5.3.1 Group Constitution Tools 60

5.3.1.1 Forms 60

5.3.1.2 Functional Modules 64

5.3.1.3 Informative Components 64

5.3.2 Group Constitution Results 64

CHAPTER SIX – CONCLUSIONS 66

REFERENCES 69

APPENDICES 76

Appendix A 76

Appendix B 88

Appendix C 104

CHAPTER ONE
INTRODUCTION

What lays the foundations for many problems in daily life is the unification of two elements, abundant information and abundant uncertainty, which is, in other words, the problem of "complexity". The decision of simplifying complexity by making a satisfactory exchange between the available information and the amount of uncertainty underlies the solution of the problem of complexity. In other words, it means increasing the amount of uncertainty by undervaluing some complete information in favor of uncertainty. However, a stronger summary description occurs in this way. Actually, uncertainty or indefiniteness, the characteristics of the natural language, should not be perceived as the loss or meaninglessness of the accuracy of language.

Independent of a certain issue, one of the methods used in coping with complexity is the theory of fuzzy logic. Briefly, fuzzy logic can be defined as modeling semantic flexibility present in the nature of the linguistic data. This method has almost unlimited application areas. There are countless exist successful applications of fuzzy logic in various fields such as engineering, psychology, artificial intelligence, pharmaceutical technology, medicine, decision theory, pattern recognition, meteorology and sociology.

Suggested first in 1965 by Zadeh, fuzzy sets are the generalized forms of classical sets and there exists a soft transitivity instead of the strict distinction between members and nonmembers in fuzzy sets. In classical sets, an element of the universe is either an element of a set or not. That is to say, their membership degrees of being or not being an element of a set can be stated as 1 and 0 respectively. However, in fuzzy sets, membership function has values in the interval of [0, 1]. Therefore, the membership function of a fuzzy set shows the belongingness degrees of all elements to the set. Generally, as Zadeh also stated, any areas can be fuzzified and, therefore, classical sets can be generalized by the concept of fuzzy set.

Fuzziness is generally confused with the concept of probability. Similar to probabilities, fuzzy membership degrees also have the same values. However, these values are not probability values. Fuzziness is a form of uncertainty. There are uncertainties in defining concepts such as “old car” or “large house” or in the meanings of words. Nevertheless, uncertainty in probability is related to randomness. In other words, an expression’s being probabilistic is only that an expression contains a kind of possibility or that the results of clearly defined but randomly occurred events. Therefore, fuzziness and randomness are different in nature; that is to say, both are different types of uncertainty. Fuzziness indicates uncertainties in “subjective” human thoughts, emotions or spoken language whereas randomness is “objective” statistics in natural sciences. If it is required to model this perspective, fuzzy models and probabilistic models are different kinds of information; fuzzy memberships express similarities between objects while probabilities give information about relative frequencies (Lin & Lee, 1996).

While forming student groups for different purposes, various criteria have been used in problem based learning system in the Department of Statistics, Dokuz Eylül University since 2001. Among the reasons why these criteria are being considered and why they are of crucial importance are;

1. To prepare a suitable learning atmosphere for students
2. To provide adaptation between students with each other in a group (or in other words, encourage team work in any condition)
3. Constituting group dynamics by bringing students with different characteristics together.

Briefly, the purpose of forming the student groups is to affect their learning positively.

There are many criteria that lecturers take into consideration while planning new groupings. Among these elements, the opinion of each lecturer about the student, their numerical assessments and students’ relationships with each other are of crucial importance. In addition, these groups are regularly rebuilt with different students in order to make them get to know each other better and so that they can learn how to behave professionally in an atmosphere which they will hova to work with individuals of different characteristics. This process requires long term commitments with great responsibility of the lectureres which is necessary for considering too many criteria together.

As mentioned above, we use linguistic variables and assessments that we are accustomed to during such tasks. It usually gets difficult to agree on the most suitable view among many other expert views, because of the lecturers' assessment of students with different point of views. In other words, the fact that many lecturers have different points of view while evaluating the performances of students causes the performance of a student likely to be evaluated differently. It is considered appropriate to obtain an agreed decision, that is to say, that a new evaluation system is required which reflects the opinions of all lecturers or an experienced group of lecturers. Therefore, a system was proposed in order to each student to be evaluated by a common performance evaluation system and then form the student groups by using these evaluations. So, the solution of this problem directed us to use fuzzy set, fuzzy clustering and assignment methods.

This thesis contains six chapters. In Chapter 2, brief information about fuzzy sets and basic operations on fuzzy sets is given. Also, extended information on an important clustering tecnique, Fuzzy *c*-means method, which is needed in student clustering for construction of student groups is given in this chapter. In Chapter 3 and 4, we present an optimization approach for the evaluation of student performances and five heuristic assignment approaches for constitution of student groups respectively. Chapter 5 presents some real problem examples and the numerical results of our performance evaluation and heuristic assignment approaches. Also, two Borland C++ Builder 6.0 applications, which are developed for the evaluation of student performances and construction of student groups, are mentioned in Chapter 5. Finally, conclusions will be presented in Chapter 6.

CHAPTER TWO
FUZZY SETS AND LINGUISTIC VARIABLES

2.1 Definitions and Operations on Fuzzy Sets

As mentioned above, fuzzy sets introduce vagueness by eliminating the sharp boundary dividing members of the class from nonmembers in the group. Consequently, the transition between full membership and nonmembership is graded. Hence, fuzzy sets can be denoted as a generalization of the crisp sets. However, some theories are unique for the fuzzy sets.

Zadeh defines fuzzy set in 1965 as below,

Definition 2.1: A fuzzy set is characterized by a membership function mapping the elements of a space, or universe of discourse U to the unit interval $[0,1]$ (Zadeh, 1965). That is, $\widetilde{A} : U \to [0,1]$. Thus, a fuzzy set $\widetilde{A}$ in the universe of discourse U may be represented as a set of ordered pairs of an element $x \in U$ and its grade of membership function which is shown as below,

$$\widetilde{A} = \{(x, \mu_{\widetilde{A}}(x)) \mid x \in U\} \tag{2.1}$$

where $\mu_{\widetilde{A}}(x)$ is the degree of membership of x and it indicates the degree that x belongs to $\widetilde{A}$.

From now on, we will refer to $\widetilde{A}$ as *A* for convenience.

Some of the important features of fuzzy sets are as follows;

1. The *support* of a fuzzy set *A* is the crisp set of all $x \in U$ such that $\mu_A(x) > 0$. That is,

$$\text{Supp}(A) = \{x \in U \mid \mu_A(x) > 0\} \tag{2.2}$$

2. The *core* of a fuzzy set *A* is the crisp set of all $x \in U$, which satisfies a unit level of membership in *A*. More formally,

$$\text{Core}(A) = \{x \in U \mid \mu_A(x) = 1\} \tag{2.3}$$

3. The element $x \in U$ at which $\mu_A(x) = 0.5$ is called the *crossover point.*

4. The *height* of a fuzzy set *A* is the supremum of $\mu_A(x)$ over U. That is,

$$\text{Height}(A) \equiv \sup_x \mu_A(x) \tag{2.4}$$

5. A fuzzy set *A* is *normal* when the height of the fuzzy set is "1", that is $\sup_x \mu(x) = 1$, otherwise it is *subnormal.*

6. A nonempty fuzzy set *A* can always be normalized by dividing $\mu_A(x)$ by the height of *A*.

Convexity of fuzzy sets plays an important role in fuzzy set theory. A fuzzy set is *convex* if and only if each of its α-cuts is a convex set. Equivalently, a fuzzy set *A* is convex if and only if

$$\mu_A(\lambda x_1 + (1-\lambda)x_2) \geq \min(\mu_A(x_1), \mu_A(x_2)), \qquad x_1, x_2 \in U, \lambda \in [0, 1]. \tag{2.5}$$

In addition, the *cardinality* of a fuzzy set can be defined as the summation of the membership grades of all elements of *x* in *A* which is similar to the crisp set theory. That is,

$$|A| = \sum_{x \in U} \mu_A(x). \tag{2.6}$$

For a discrete universe of discourse U, a fuzzy set *A* can be written by using the support of *A* as

$$A = \mu_1/x_1 + \mu_2/x_2 + \ldots + \mu_n/x_n = \sum_{i=1}^{n} \mu_i/x_i \tag{2.7}$$

where "+" indicates the union of the elements, "/" is employed to link the elements of the support with their grades of membership in A and $\mu_i = \mu_A(x_i) > 0$. If U is not discrete, but is an interval of real numbers, below notation can be used,

$$A = \int_U \mu_A(x)/x \tag{2.8}$$

where $\int$ indicates the union of the elements in A (Klir & Folger; 1988; Lin & Lee, 1996; Pedrycz & Gomide, 1998).

In the next section, some important fundamental set operations on fuzzy sets are mentioned.

2.1.1 Fundamental Set Operations on Fuzzy Sets

While in classical clusters an element can only be member of a single cluster, in fuzzy cluster an element can be attached to different clusters with different membership values. Therefore, fuzzy cluster operators are interested in the membership values of each element.

Let A and B be two fuzzy sets in the universe of discourse U.

1. Complement: For $\mu_A(x) \in [0,1]$, the complement of A is defined by its membership function as

   $$\mu_{\overline{A}}(x) = 1 - \mu_A(x), \qquad \forall x \in U \tag{2.9}$$

2. Intersection: The intersection of fuzzy sets A and B is defined as

   $$\mu_{A\cap B}(x) \overset{\Delta}{=} \min[\mu_A(x), \mu_B(x)] = \mu_A(x) \wedge \mu_B(x), \qquad \forall x \in U \tag{2.10}$$

3. Union: The union of fuzzy sets A and B is defined by

   $$\mu_{A\cup B}(x) \overset{\Delta}{=} \max[\mu_A(x), \mu_B(x)] = \mu_A(x) \vee \mu_B(x), \qquad \forall x \in U \tag{2.11}$$

   where $\vee$ indicates the max operation.

4. Equality: A and B are equal if and only if $\mu_A(x) = \mu_B(x)$ is satisfied for all $x \in U$. If $\mu_A(x) \neq \mu_B(x)$ for some $x \in U$, then $A \neq B$. But this definition of equality is crisp. To check the degree of equality of two fuzzy sets, similarity measure can be used which is defined as shown below. This measure takes values in closed interval $[0,1]$.

$$E(A,B) \equiv \text{degree}(A = B) \overset{\Delta}{=} \frac{|A \cap B|}{|A \cup B|} \tag{2.12}$$

5. Subset: If $\mu_A(x) \leq \mu_B(x)$ for all $x \in U$ then $A \subseteq B$. If $A \subseteq B$ and $A \neq B$, then A is proper subset of B; that is $A \subset B$. Subsethood measure which is used to check the degree that A is a subset of B is shown below.

$$S(A,B) \equiv \text{degree}(A \subseteq B) \overset{\Delta}{=} \frac{|A \cap B|}{|A|} \tag{2.13}$$

6. DeMorgan's laws:

$$\begin{aligned} \overline{A \cup B} &= \overline{A} \cap \overline{B} \\ \overline{A \cap B} &= \overline{A} \cup \overline{B} \end{aligned} \tag{2.14}$$

7. Cartesian product: Let $A_1, A_2, \ldots, A_n$ be fuzzy sets in $U_1, U_2, \ldots, U_n$, respectively. The Cartesian product of $A_1, A_2, \ldots, A_n$ is a fuzzy set in the product space $U_1 \times U_2 \times \ldots \times U_n$ with the membership function as

$$\mu_{A_1 \times A_2 \times \ldots \times A_n}(x_1, x_2, \ldots, x_n) \overset{\Delta}{=} \min\left[\mu_{A_1}(x_1), \mu_{A_2}(x_2), \ldots, \mu_{A_n}(x_n)\right], \tag{2.15}$$

where $x_1 \in U_1, x_2 \in U_2, \ldots, x_n \in U_n$.

8. Algebraic sum: The algebraic sum of two fuzzy sets is defined as

$$\mu_{A+B}(x) \overset{\Delta}{=} \mu_A(x) + \mu_B(x) - \mu_A(x).\mu_B(x) \tag{2.16}$$

9. Algebraic product: The algebraic product of two fuzzy sets is defined as

$$\mu_{A \cdot B}(x) \overset{\Delta}{=} \mu_A(x).\mu_B(x) \tag{2.17}$$

10. Bounded sum: The bounded sum of two fuzzy sets is defined as

$$\mu_{A\oplus B}(x)\overset{\Delta}{=}\min\{1,\ \mu_A(x)+\mu_B(x)\} \tag{2.18}$$

11. Bounded difference: The bounded difference of two fuzzy sets is defined as

$$\mu_{A-B}(x)\overset{\Delta}{=}\max\{0,\ \mu_A(x)-\mu_B(x)\} \tag{2.19}$$

2.1.2 Fuzzy Relations

The notion of relations in science and engineering, essentially donates the discovery of relations between observations and variables. The crisp relation represents the presence or absence of interactions between the elements of two or more sets. However, *fuzzy relation* has been obtained by generalizing this concept to allow for various degrees of interactions between elements. Hence, a fuzzy relation is based on the philosophy that everything is related to each other to some extent or unrelated.

A fuzzy relation is a fuzzy set defined on the Cartesian product of crisp sets $\{X_1,X_2,\ldots,X_n\}$, where tuples $(x_1,x_2,\ldots,x_n)$ may have varying degrees of membership $\mu_R(x_1,x_2,\ldots,x_n)$ within the relation. That is,

$$R(X_1,X_2,\ldots,X_n)=\int_{X_1\times X_2\times\ldots\times X_n}\mu_R(x_1,x_2,\ldots,x_n)/(x_1,x_2,\ldots,x_n), \qquad x_i\in X_i \tag{2.20}$$

In the simplest case, consider two crisp sets $X_1,\ X_2$. Then

$$R(X_1,X_2)=\{((x_1,x_2),\mu_R(x_1,x_2))\mid(x_1,x_2)\in X_1\times X_2\} \tag{2.21}$$

is a fuzzy relation on $X_1\times X_2$. It is clear that a fuzzy relation is a fuzzy set.

A special fuzzy relation called *binary fuzzy relation* plays an important role in fuzzy set theory. This concept is a fuzzy relation between two sets X and Y and it is denoted by $R(X,Y)$.

There are more convenient forms of representation of binary fuzzy relations $R(X,Y)$ in addition to the membership function. Let $X=\{x_1,x_2,\ldots,x_n\}$ and $Y=\{y_1,y_2,\ldots,y_m\}$. First, the fuzzy relation $R(X,Y)$ can be expressed by a $n\times m$ matrix as below.

$$R(X,Y)=\begin{bmatrix} \mu_R(x_1,y_1) & \mu_R(x_1,y_2) & \ldots & \mu_R(x_1,y_m) \\ \mu_R(x_2,y_1) & \mu_R(x_2,y_2) & \ldots & \mu_R(x_2,y_m) \\ \vdots & \vdots & \ddots & \vdots \\ \mu_R(x_n,y_1) & \mu_R(x_n,y_2) & \ldots & \mu_R(x_n,y_m) \end{bmatrix}_{n\times m} \tag{2.22}$$

An important operation on fuzzy relations is the composition of fuzzy relations. Basically, there are two types of composition operators: *max-min composition* and *min-max composition*.

Let $P(X,Y)$ and $Q(Y,Z)$ be two fuzzy relations on $X\times Y$ and $Y\times Z$, respectively. The max-min composition of $P(X,Y)$ and $Q(Y,Z)$, denoted as $P(X,Y)\circ Q(Y,Z)$, is defined as

$$\mu_{P\circ Q}(x,z)\overset{\Delta}{=}\max_{y\in Y}\min[\mu_P(x,y),\ \mu_Q(y,z)], \quad \forall x\in X, \forall z\in Z \tag{2.23}$$

The min-max composition of $P(X,Y)$ and $Q(Y,Z)$, denoted as $P(X,Y)\,\square\, Q(Y,Z)$, is defined as

$$\mu_{P\square Q}(x,z)\overset{\Delta}{=}\min_{y\in Y}\max[\mu_P(x,y),\ \mu_Q(y,z)], \quad \forall x\in X, \forall z\in Z \tag{2.24}$$

The max-min composition is the most commonly used composition operation. These compositions can be generalized to other compositions by replacing the min operator in max-min composition and max operator in min-max composition with any t-norm and t-conorm operators, respectively.

A similar operator on two binary fuzzy relations is called *relational joint.* Let $P(X,Y)$ and $Q(Y,Z)$ be two binary fuzzy relations. Then the relational joint of P and Q can be shown as below for each $x \in X, y \in Y$ and $z \in Z$.

$$\mu_{P \star Q}(x,y,z) \overset{\Delta}{=} \min\left[\mu_P(x,y), \mu_Q(y,z)\right] \tag{2.25}$$

Some basic properties of the relations are as follows:

1. Reflexivity: A fuzzy relation $R(X,X)$ is *reflexive* if and only if $\mu_R(x,x)=1$ for all $x \in X$. This property states that all diagonal elements of the relation are equal to 1. If it is not satisfied for all $x \in X$, then the relation is called *antireflexive*. If it is not the case for some $x \in X$, then $R(X,X)$ is *irreflexive*.

2. Symmetry: A fuzzy relation $R(X,X)$ is *symmetric* if and only if $\mu_R(x,y)=\mu_R(y,x)$ for all $x,y \in X$. If the equality is not satisfied for all members of the support of the relation, then it is called *anti-symmetric*. If it is not satisfied for all $x,y \in X$ then $R(X,X)$ is called *strictly anti-symmetric*. Whenever this equality is not satisfied for some $x,y \in X$, the relation is called *asymmetric*.

3. Transitivity: A fuzzy relation $R(X,X)$ is *transitive* if and only if $\mu_R(x,z) \geq \max_{y \in Y} \min\left[\mu_R(x,y), \mu_R(y,z)\right]$ for all $(x,z) \in X^2$. If this inequality does not hold for all $(x,z) \in X^2$, then $R(X,X)$ is called *anti-transitive*. If it is satisfied for only some members of X but not all, then it is called *nontransitive.*

2.1.3 The Resolution and Extension Principle

Another important property of fuzzy sets, which requires us to understand α-level sets, is called as *resolution principle*. An α-level set of a fuzzy set A is a crisp set A_α that contains all the elements of U having a membership grade in A greater than or equal to α. That is,

$$A_\alpha = \left\{ x \in U \middle| \mu_A(x) \geq \alpha \right\}, \qquad \alpha \in (0,1] \tag{2.26}$$

If $A_\alpha = \left\{ x \in U \middle| \mu_A(x) > \alpha \right\}$, then A_α is called a *strong α-cut.*

Consequently, resolution principle, which is defined as the membership function of A can be expressed in terms of the membership functions of its α-cuts, according to

$$\mu_A(x) = \sup_{\alpha \in [0,1]} \left(\alpha \wedge \mu_{A_\alpha}(x) \right), \qquad \forall x \in U \tag{2.27}$$

where $\wedge$ denotes the min operation and $\mu_{A_\alpha}(x)$ is the membership function of the crisp set A_α,

$$\mu_{A_\alpha}(x) = \begin{cases} 1 & \text{if and only if } x \in A_\alpha \\ 0 & \text{otherwise} \end{cases} \tag{2.28}$$

This leads to the following representation of a fuzzy set A using the resolution principle. Let αA_α denote a fuzzy set with the membership function

$$\mu_{\alpha A_\alpha}(x) = [\alpha \wedge \mu_{A_\alpha}(x)], \qquad \forall x \in U \,. \tag{2.29}$$

Then the resolution principle states that the fuzzy set A can be expressed as given below.

$$A = \bigcup_{\alpha \in \Lambda_A} \alpha \, A_\alpha \qquad \text{or} \qquad A = \int_0^1 \alpha \, A_\alpha \tag{2.30}$$

The resolution principle indicates that a fuzzy set A can be decomposed into αA_α, $\alpha \in (0,1]$. On the other hand, a fuzzy set A can be retrieved as a union of its αA_α, which is called the *representation theorem*.

The *extension principle* is one of the most important tools of fuzzy set theory, which is used for translation of crisp set into their fuzzy set framework and extends point-to-point mappings to mappings for fuzzy sets.

Let X and Y be two crisp sets and $f : X \to Y$. Let A be a fuzzy set in X where $A = \mu_1/x_1 + \mu_2/x_2 + \ldots + \mu_n/x_n$. The extension principle states that,

$$f(A) = f(\mu_1/x_1 + \mu_2/x_2 + \ldots + \mu_n/x_n) = \mu_1/f(x_1) + \mu_2/f(x_2) + \ldots + \mu_n/f(x_n). \qquad (2.31)$$

If more than one element of X is mapped by function f to the same element $y \in Y$, then the maximum of the membership grades of these elements is chosen as the membership grade of y in $f(A)$. If no element x in X is mapped to y, then the membership grade of y is zero.

Often a function f maps ordered tuples of elements of different sets $X_1, X_2, \ldots, X_n$ as $f(x_1, x_2, \ldots, x_n) = y,\ y \in Y$. Let $A_1, A_2, \ldots, A_n$ be n fuzzy sets in $X_1, X_2, \ldots, X_n$, respectively. The extension principle allows the function $f(x_1, x_2, \ldots, x_n)$ to be extended to act on the n fuzzy subsets of X, $A_1, A_2, \ldots, A_n$, such that

$$B = f(A_1, A_2, \ldots, A_n) \qquad (2.32)$$

where B is the fuzzy image of $A_1, A_2, \ldots, A_n$ through function f. The fuzzy set B is defined as

$$B = \{(y, \mu_B(y)) \mid y = f(x_1, x_2, \ldots, x_n),\ (x_1, x_2, \ldots, x_n) \in X\} \qquad (2.33)$$

where

$$\mu_B(y) = \sup_{y=f(x_1,x_2,\ldots,x_n)} \min[\mu_{A_1}(x_1), \mu_{A_2}(x_2), \ldots, \mu_{A_n}(x_n)]. \tag{2.34}$$

2.1.4 Aggregation and Defuzzification Operations

Aggregation operations are used to combine several fuzzy sets to produce a single common fuzzy set. Aggregation operation is defined as below.

$$h : [0,1]^n \rightarrow [0,1], \qquad n \geq 2 \tag{2.35}$$

When applied to n fuzzy sets defined on U, h produces an aggregate fuzzy set A by operating on the membership grades of each $x \in U$ in the aggregated set. Thus,

$$\mu_A(x) = h(\mu_{A_1}(x), \mu_{A_2}(x), \ldots, \mu_{A_n}(x)), \qquad \forall x \in U \tag{2.36}$$

(Klir & Folger, 1988).

An aggregation must satisfy the boundary and the monotonic conditions. In addition to these conditions h is a continuous and a symmetric function in all its arguments. Hence, fuzzy unions and intersections can be viewed as special aggregation operations and they do not produce any aggregates of $\mu_{A_1}(x), \mu_{A_2}(x), \ldots, \mu_{A_n}(x)$ that produce values between $\min(\mu_{A_1}(x), \mu_{A_2}(x), \ldots, \mu_{A_n}(x))$ and $\max(\mu_{A_1}(x), \mu_{A_2}(x), \ldots, \mu_{A_n}(x))$. Aggregates, which are between these values, are usually called as *averaging operations*. Hence, averaging operators are aggregation operations for which

$$\min(\mu_{A_1}(x), \ldots, \mu_{A_n}(x)) \leq h(\mu_{A_1}(x), \ldots, \mu_{A_n}(x)) \leq \max(\mu_{A_1}(x), \ldots, \mu_{A_n}(x)). \tag{2.37}$$

One typical parametric averaging operator is the *generalized means*, which is defined as

$$h_\alpha(\mu_{A_1}(x),\mu_{A_2}(x),\ldots,\mu_{A_n}(x)) \overset{\Delta}{=} \left(\frac{\sum_{i=1}^{n} [\mu_{A_i}(x)]^\alpha}{n} \right)^{1/\alpha} \tag{2.38}$$

where $\alpha \in \Re$ but $\alpha \neq 0$. When α approaches $-\infty$ then h_α becomes $\min(\mu_{A_1}(x),\mu_{A_2}(x),\ldots,\mu_{A_n}(x))$, and when α approaches ∞ then h_α becomes $\max(\mu_{A_1}(x),\mu_{A_2}(x),\ldots,\mu_{A_n}(x))$.

An important extension of the generalized means is the *weighted generalized means* and is defined as

$$h_\alpha(\mu_{A_1}(x),\mu_{A_2}(x),\ldots,\mu_{A_n}(x);w_1,w_2,\ldots,w_n) \overset{\Delta}{=} \left(\sum_{i=1}^{n} w_i[\mu_{A_i}(x)]^\alpha \right)^{1/\alpha} \tag{2.39}$$

where $w_i \geq 0$ and $\sum_{i=1}^{n} w_i = 1$. The weights express the relative importance of the aggregated set. This operation is useful in decision-making problems where different criteria differ in importance (Lin & Lee, 1996).

Ordered weighted averaging operator (OWA) is another important aggregation operator is proposed by Yager (1988). Essentially, this operator is a weighted sum whose arguments are ordered. By using these operators, researchers can obtain aggregation results which lie in between "and" and "or" operators' which means "all the criteria must be satisfied" and "any of the criteria must be satisfied" respectively.

Let $w_i, i = 1,...,n$ and $\sum_{i=1}^{n} w_i = 1$. The sequence of membership values $\mu_A(x_i)$ can be ordered as $\mu_A(x_1) \leq \mu_A(x_2) \leq \ldots \leq \mu_A(x_n)$. Thus, this operator can be shown as below.

$$OWA = \sum_{i=1}^{n} w_i \mu_A(x_i) \tag{2.40}$$

If $w_i, i = 1,...,n$ values are chosen equal to $1/n$ then the result will be the arithmetic mean. If only $w_1 = 1$ and the other weight values are chosen equal to zero then the On the contrary, if $w_n = 1$ and the other weight values chosen equal to zero then the operator will act as an "and" operator.

Yager extends this operator in 2004 as *Generalized OWA Aggregation operators (GOWA)* to provide a new class of operators.

Besides the aggregation operation, defuzzification is another important operation in the theory of fuzzy sets and it is used to transform fuzzy values into crisp values. There are four most often used defuzzification mechanisms in the fuzzy control theory: *the mean of maxima (MOM)*, *the center of area (COA)*, *the center of means*, and the *midpoint of an area* procedures (Klir & Folger, 1988; Roychowdhury & Pedrycz, 2001). In addition to these methods, many approaches were suggested. From these methods, MOM and COA methods and *WABL (Weighted Averaging Based on The Levels)* method which is proposed and investigated by Nasibov(2002, 2003a, 2003b, 2005, 2007e) are used in this thesis.

In MOM method, defuzzified value is the mean of the x_i elements, which have maximum membership values. Mathematical form is as shown below.

$$MOM(A) = \sum_{i=1}^{m} \frac{x_i}{m} \tag{2.41}$$

COA is also known as the *Center of Gravity (COG)* method in the fuzzy literature. The COA method determines the center of area of membership function and is defined as in (2.42).

$$COA(A) = \frac{\int_{-\infty}^{\infty} x\mu_A(x)dx}{\int_{-\infty}^{\infty} \mu_A(x)dx} \tag{2.42}$$

The mathematical form of WABL method is defined by Nasibov (2002, 2003a) as below.

$$\begin{gathered} I(A) = \int_0^1 \left(c_L L_A(\alpha) + c_R R_A(\alpha)\right) \cdot p(\alpha)d\alpha \\ \int_0^1 p(\alpha)d\alpha = 1 \\ c_L \geq 0, \quad c_R \geq 0, \quad c_L + c_R = 1, \quad p:[0,1] \to E_+ \end{gathered} \tag{2.43}$$

WABL parameters c_L and c_R represent the weights of $L_A(\alpha)$ and $R_A(\alpha)$ functions respectively. $L_A(\alpha)$ and $R_A(\alpha)$ functions are the left and right sides of the fuzzy number. $L_A(\alpha)$ is a non-decreasing and $R_A(\alpha)$ is a non-increasing and both are left continuous functions. $p(\alpha)$ is the distribution function of the importance of the level sets. By using the distribution function, WABL adds all level sets into the defuzzification process (Nasibov, 2003b).

2.2 Linguistic Variables and Its Constitution Methods

2.2.1 Linguistic Variables

Linguistic variable is an important concept in many areas, especially in fuzzy logic, approximate reasoning, fuzzy expert systems etc. Fundamentally, a linguistic variable can be defined as a variable whose values are words or sentences in natural languages. For example, “heat” is a linguistic variable and can take a range of the values such as {very cold, cold, mild, hot, very hot,…}. Zadeh introduced the concept of linguistic variables in 1975 to provide a means of approximate characterization of phenomena that are too complex or too hard to define in conventional quantitative terms.

A linguistic variable is characterized by a quintuple denoted by $(x, T(x), U, G, M)$ in which *x* is the name of the variable; $T(x)$ is the term set of *x*, that is the set of names of linguistic values of *x* with each value being a fuzzy set defined on *U*; *G* is a syntactic rule for generating the names of values of *x*; and *M* is a semantic rule for associating each value of *x* with its meaning.

In general, a linguistic variable involves a finite number of primary terms such as “absent”, “few”, “middle”, etc. a finite number of hedges such as “very”, “more”, “less”, etc. and the connectives *and* and *or*, and the negation *not.* These terms are referred to as modifiers. Some important fuzzy set operations, which are used in defining linguistic hedges, are as shown below.

1. Concentration: This operation is used to obtain a membership function, which is more concentrated around the points with higher membership grades. For example, “very” is the one of the frequently used concentration operation.

$$\mu_{\text{CON(A)}}(x) = (\mu_A(x))^2 \tag{2.44}$$

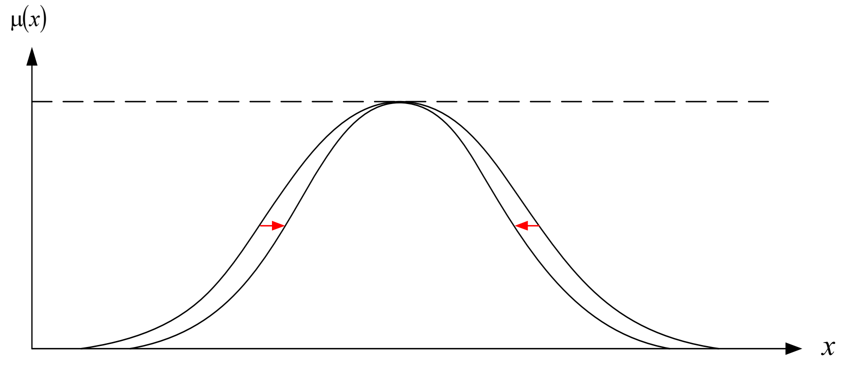


Figure 2.1 Concentration of a membership function

2. Dilation: This operation has the opposite effect of the concentration operation.

$$\mu_{\mathrm{DIL(A)}}(x) = \left(\mu_A(x)\right)^{1/2} \tag{2.45}$$

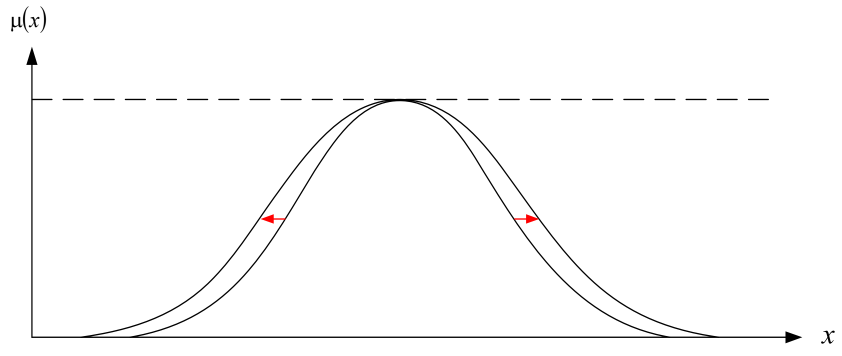


Figure 2.2 Dilation of a membership function

3. Intensification: The membership values in interval $[0, 0.5]$ are diminished while the grades of membership in interval $(0.5, 1]$ are elevated. This operation is shown as in Figure 2.3 and defined as below.

$$\mu_{\mathrm{INT(A)}}(x) = \begin{cases} 2\left(\mu_A(x)\right)^2, & \mu_A(x) \in [0,\ 0.5] \\ 1 - 2\left(1 - \mu_A(x)\right)^2, & \text{otherwise} \end{cases} \tag{2.46}$$

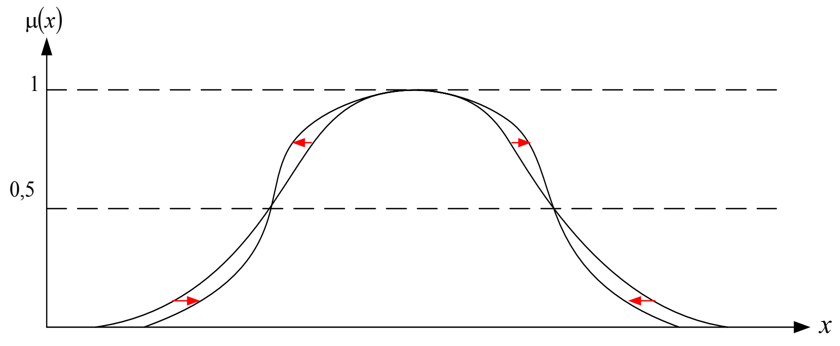


Figure 2.3 Intensification of a membership function

4. Fuzzification: This operation is complementary to that of intensification and it is defined as below.

$$\mu_{\mathrm{FUZZ(A)}}(x) = \begin{cases} \sqrt{\mu_A(x)/2}, & \mu_A(x) \in [0,\ 0.5] \\ 1 - \sqrt{\left(1 - \mu_A(x)\right)/2}, & \text{otherwise} \end{cases} \tag{2.47}$$

2.2.2 Parametric Constitution Methods of Linguistic Variables

Parametric and statistical methods can be used in populating linguistic variables. Parametric methods are mainly based on parametric fuzzy numbers. That is to say, the membership function of a linguistic variable that can be given by parametric fuzzy numbers.

Some of the frequently preferred membership function types which reflect the linguistic variables are as follows;

1. Triangular membership function:

$$\mu_{\tilde{A}}(x)=\begin{cases}0\,, & x<a\\ \dfrac{x-a}{b-a}\,, & x\in\left[a,b\right)\\ \dfrac{c-x}{c-b}\,, & x\in\left[b,c\right]\\ 0\,, & x>c\end{cases} \tag{2.48}$$

2. Trapezoidal membership function:

$$\mu_{\tilde{A}}(x)=\begin{cases}0\,, & x<a\\ \dfrac{x-a}{b-a}\,, & x\in\left[a,b\right)\\ 1, & x\in\left[b,c\right)\\ \dfrac{d-x}{d-c}\,, & x\in\left[c,d\right]\\ 0\,, & x>d\end{cases} \tag{2.49}$$

3. S - membership function:

$$\mu_{\tilde{A}}(x)=\begin{cases}0, & x<a\\ 2\left(\dfrac{x-a}{b-a}\right)^2, & x\in\left[a,\dfrac{a+b}{2}\right)\\ 1-2\left(\dfrac{x-b}{b-a}\right)^2, & x\in\left[\dfrac{a+b}{2},b\right)\\ 1, & x\geq b\end{cases} \tag{2.50}$$

4. Γ- membership function:

$$\mu_{\tilde{A}}(x) = \begin{cases} 0, & x \leq a \\ 1 - e^{-k(x-a)^2}, & x > a \end{cases} \tag{2.51}$$

In the application section of this work, triangular and trapezoidal fuzzy numbers will be used.

2.2.3 Fuzzy Clustering Approach to Constitution of Linguistic Variables

So as to constitute the membership function of a linguistic variable depending on statistics, data mining techniques are used. The most frequently used technique among these techniques is the fuzzy clustering.

Clustering methods are unsupervised learning methods that are used to organize data into groups based on similarities among the individual data items. Most clustering algorithms are useful in situations where little prior knowledge exists.

In general, the clustering methods can be investigated into five main classes; *partitioning methods*, *hierarchical methods*, *density-based methods*, *grid-based methods*, and *model-based methods*. In partitioning methods, the *k*-means algorithm and the *k*-medoids algorithm are the most known and important methods. Based on the hierarchical decomposition form, hierarchical methods can be classified as being agglomerative or divisive. The methods, which have been developed based on the notion of density, are called density-based methods. DBSCAN, and OPTICS, are amongs the examples of such methods. Grid-based methods quantize the object space into a finite number of cells to form a grid structure. Advantage of these methods is the short computational time. STING is a typical example of grid-based methods. Lastly, the model-based methods hypothesize a model for each of the clusters and find the best fit of the data for the given models (Han & Kamber, 2001).

As briefly mentioned above, many clustering algorithms have been discussed in literature. Since clusters can formally be seen as subsets of the data set, one possible classification of clustering methods can be according to whether the subsets are fuzzy or crisp (hard). In hard clustering, an object either does or does not belong to a cluster and this means partitioning the data into a specified number of mutually exclusive subsets. On the other hand in fuzzy clustering, the boundary between clusters may not be precisely defined or in another words, these methods allow the elements to belong to several clusters with different membership grades.

In recent years, many approaches have been investigated by many researchers on fuzzy clustering methods (Bezdek, 1981; Bobrowski & Bezdek, 1991; Dunn, 1973; Gordon, 1981; Hathaway & Bezdek, 1993). One of the most widely used clustering methods is the Fuzzy *c*-means (FCM) algorithm, which was introduced by Dunn (1973) and developed by Bezdek (1981). This algorithm is also a generalization of the *k*-means algorithm.

2.2.3.1 Fuzzy c-Means

In clustering techniques a general form of the objective function is

$$J(\mu_{ij}, v_k) = \sum_{i=1}^{c} \sum_{j=1}^{n} \sum_{k=1}^{c} g[w(x_j), \mu_{ij}]\, d(x_j, v_k), \tag{2.52}$$

where $w(x_j)$ is the priori weight for each x_j, $g[w(x_j), \mu_{ij}]$ is the degree of fuzziness of the partition matrix, and $d(x_j, v_k)$ is the degree of dissimilarity between the data x_j and the supplement element v_k, which can be considered the central vector of the kth cluster. Several distance measures can be used to represent degree of dissimilarity as Minkowski, Euclidean, Mahalanobis, Tchebyschev, Hamming (city block) or maximum distances. Each of these distance measures indicates a different view of the data because of their geometry. Thus, the most appropriate distance measure can be selected by using the pattern of data.

The degree of dissimilarity must satisfy the following axioms.

i. $d(x_j, v_k) \geq 0, \quad \forall j, k$

ii. $d(x_j, x_j) = 0, \quad \forall j$

iii. $d(x_j, v_k) = d(v_k, x_j).$

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a finite set of elements in the p-dimensional Euclidean space $\Re^p$. The aim is to perform a partition of this collection of elements into c fuzzy sets, where c is a given number of clusters and the result of this fuzzy clustering can be expressed by a partition matrix U such that

$$U = [\mu_{ij}]_{i=1,\ldots,c,\ j=1,\ldots,n} \tag{2.53}$$

where μ_{ij} is a numerical value in [0,1] and denotes the degree to which the element x_j belongs to the i^{th} cluster. There are two constraints on the value of μ_{ij}. Firstly, a total membership of the element $x_j \in X$ in all classes must be equal to 1; that is,

$$\sum_{i=1}^{c} \mu_{ij} = 1 \qquad j = 1, 2, \ldots, n. \tag{2.54}$$

Secondly, every constructed cluster must be nonempty and different from the entire set; that is,

$$0 < \sum_{j=1}^{n} \mu_{ij} < n, \qquad i = 1, 2, \ldots, c. \tag{2.55}$$

Using these information, fuzzy clustering optimization problem can be formulated as follows,

$$\text{Minimize } J(\mu_{ij}, v_i) = \sum_{i=1}^{c} \sum_{j=1}^{n} (\mu_{ij})^m \left\| x_j - v_i \right\|^2, \qquad m > 1, \tag{2.56}$$

Subject to Eqs. (2.54) and (2.55)

where *m* is a parameter which is called *exponential weight* and influences the degree of fuzziness of the membership matrix. The minimization of this nonlinear optimization problem can be solved by using different methods as iterative minimization, simulated annealing or genetic algorithms. The most popular method is a simple Picard iteration for stationary points of (2.56), known as Fuzzy *c*-means algorithm. Thus, the nonlinear minimization problem can be solved by using Lagrange multiplier method as below,

$$v_i = \frac{\sum_{j=1}^{n} (\mu_{ij})^m x_j}{\sum_{j=1}^{n} (\mu_{ij})^m}, \quad i = 1,2,\ldots,c, \tag{2.57}$$

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\left\| x_j - v_i \right\|}{\left\| x_j - v_k \right\|} \right)^{2/(m-1)}}, \quad i = 1,2,\ldots,c;\ j = 1,2,\ldots,n. \tag{2.58}$$

This system can be solved iteratively. At first, (2.57) is used to obtain the new center of each cluster and then (2.58) is used to obtain new fuzzy partition. Center values and fuzzy partitions are recalculated by repeating this procedure until (2.56) reaches to minimum.

2.2.3.2 Cluster Validity Indexes

An important issue for the FCM algorithm is the determination of the correct number of clusters, *c*. Some scalar measures of partitioning fuzziness are used as synthetic indices, called *validity indicators*, to point out the most plausible number of clusters in the data set since there is no exact solution of this problem. Some widely used scalar measures are given in Table 2.1 (Bezdek, 1974, 1975; Dunn, 1974; Fukuyamo & Sugeno, 1989; Xie & Beni, 1991, Kwon, 1998; Nasibov & Ulutagay, 2006b).

Table 2.1 Some important cluster validity criteria

Validity criteria	**Functional description**	**Optimal cluster number**
Partition coefficient	$V_{PC} = \frac{1}{n}\sum_{i=1}^{c}\sum_{j=1}^{n}\mu_{ij}^{2}$	$\max(V_{PC},U,c)$
Partition entropy	$V_{PE} = -\frac{1}{n}\sum_{i=1}^{c}\sum_{j=1}^{n}\mu_{ij}\log_a \mu_{ij}$	$\min(V_{PE},U,c)$
Separation index	$V_{SI} = \frac{\min\limits_{i\neq j} d(\mu_i,\mu_j)}{\max\limits_{i}\delta(\mu_i)}$	$\max(V_{SI},U)$
Xie-Beni index	$V_{XB} = \frac{\sum_{i=1}^{c}\sum_{j=1}^{n}\mu_{ij}^{2}\left\Vert x_j - v_i\right\Vert^{2}}{n\left(\min\limits_{i\neq k}\left\Vert v_i - v_k\right\Vert^{2}\right)}$	$\min(V_{XB},U,c)$
Fukuyamo-Sugeno index	$V_{FS_m} = \sum_{i=1}^{c}\sum_{j=1}^{n}\mu_{ij}^{m}[d^2(x_j,v_i) - d^2(m_X,v_i)]$	$\min(V_{FS},U,c)$
Kwon	$V_K = \frac{\sum_{i=1}^{c}\sum_{j=1}^{n}\mu_{ij}^{2}\left\Vert x_j - v_i\right\Vert^{2} + \frac{1}{c}\sum_{i=1}^{c}\left\Vert v_i - \bar{v}\right\Vert}{\left(\min\limits_{i\neq k}\left\Vert v_i - v_k\right\Vert^{2}\right)}$	$\min(V_K,U,c)$
Fuzzy Joint Points criteria	$V_{FJP} = \min\limits_{i\neq j}\, d(X^i,X^j) - \max\limits_{k}\,\{d_{\max}\cdot(1 - \min\limits_{x,y\in X^k}\hat{T}(x,y))\}$	$\max(V_{FJP},U,\alpha)$

CHAPTER THREE
AN OPTIMIZATION APPROACH FOR THE EVALUATION OF STUDENT PERFORMANCES

3.1 Introduction

In group decision analysis, different approaches have been suggested by many researchers for the problem of aggregation of the individual fuzzy opinions to form a group consensus as the basis of group decision. These approaches are used in many different application areas such as evaluation of the workers' performances, selection of the most suitable worker and meauring the students' success.

A multi-criteria personnel selection problem with multi-decision makers was studied by Chen (2000) using TOPSIS (Technique for order performance by similarity to ideal solution) procedure and vertex method with fuzzy information. TOPSIS procedure can briefly be explained as a concept where the chosen alternative should have the shortest distance from the positive ideal solution while having the furthest distance from the negative one. In addition, Saghafian and Hejazi (2005) proposed a modified TOPSIS for the multi-criteria decision-making problem with multi-decision makers. Kuo et al. (2007) proposed a new method of analysis of multi-criteria based on the incorporated efficient model and concepts of TOPSIS to solve decision-making problems with multi-judges and multi-criteria in real-life situations. Other studies have also been carried out by applying AHP (Analytical Hierarchy Process) approach suggested by Saaty (1990) on fuzzy numbers (Bonder, Graan & Lootsma, 1989; Kahraman, Ruan & Doğan, 2003).

Bardossy et al. (1993) suggests five combination techniques and defines seven characteristics of the combination techniques. These five techniques are named as *crisp weighting*, *fuzzy weighting*, *minimal fuzzy extension*, *convex fuzzy extension* and *mixed linear extension*. Hsu and Chen (1996) propose an aggregation method, which is named as similarity aggregation method. In this study, pairwise similarities of experts' opinions are calculated first. Then an average of these pairwise similarities is obtained for each expert. These average values represent their corresponding experts' agreement degrees. Finally, aggregation of experts' opinions is obtained by combining the weighted averages. Lee (2002) proposes an iterative procedure for aggregation of the expert opinions. Wang and Parkan (2006) improved Lee's study by suggesting two methods both based on the weighted distances between experts' opinions. They indicate that one common opinion could be obtained from the decision makers' opinions in various subjects. Ma and Zhou (2000) proposed a group decision support system for assessing students' learning outcomes. Yong and Wen-Kang (2003) obtained the consensus degree coefficient using the relative weight agreement degrees through weighting of the fuzzy opinions of experts.

In student-centered learning system, a student's performance is based on evaluation of a set of criteria where each criterion has different importance for each lecturer. Moreover, points awarded for any level (such as absent, few, middle, good, strong, etc.) in each criterion may vary between each lecturer. In our study, aggregate weight values, which reflect the opinions of lecturers on importance of each different criterion, are obtained from the relationship between linguistic evaluations and grade evaluations. These aggregated weight values are computed through an iterative procedure. Use of final aggregate weight values introduces consistency between different lecturers when assessing student performances. In other words, a method for obtaining aggregate weight values reflecting different points of views of lecturers for the evaluation of student performances in student-centered learning system is suggested. In our iterative procedure, defuzzification parameter and weight values are optimized in the optimization problem. Also the objective function in the optimization problem is based on the least square errors method. Consequently both defuzzified values and the least square method are the differences our study from the Lee's study.

This chapter is organized as follows. In Section 3.2, some approaches on performance evaluation problem are presented. In Section 3.3, our problem definition is introduced and Section 3.4 gives a detailed explanation of our solution to the described problem.

3.2 Fuzzy Optimization Approaches to Performance Evaluation

Let $\widetilde{A} = (a_1, a_2, a_3, a_4)$ and $\widetilde{B} = (b_1, b_2, b_3, b_4)$ be two trapezoidal fuzzy numbers and $S_2(\widetilde{A}, \widetilde{B})$ be the similarity measure between fuzzy numbers $\widetilde{A}$ and $\widetilde{B}$. Different from Hsu's similarity measure, Lee's similarity measure includes a distance metric between fuzzy numbers, which was also used by Tong and Bonissone (1980). From the similarity measure, the dissimilarity measure is defined as $c - S_2(\widetilde{A}, \widetilde{B})$, where $c > 1$. The value of c affects the aggregation of experts opinions.

Lee (2002) tries to minimize the sum of the weighted dissimilarities between aggregated opinion and each expert's opinion. $\widetilde{R}_i (i = 1,2,...,n)$ represents its corresponding expert i's opinion and $\widetilde{R}$ represents the aggregated opinion. To find the $\widetilde{R}$ value, below equation must be solved, where m is an integer > 1, c is a constant > 1 and w_i values are weight degrees.

$$\begin{aligned} & Z(W, \widetilde{R}) = \sum_{i=1}^{n} (w_i)^m \left(c - S_2(\widetilde{R}_i, \widetilde{R})\right) \rightarrow \min \\ & \text{s.t.} \\ & M = \left\{ W \mid W = (w_1, w_2, ..., w_n), w_i \geq 0, \ \sum_{i=1}^{n} w_i = 1 \right\} \end{aligned} \tag{3.1}$$

This optimization problem solution is introduced in Lee's study (2002) as follows without proof,

$$\widetilde{R} = \frac{1}{\sum_{i=1}^{n} (\widetilde{w}_i)^{m_0}} \sum_{i=1}^{n} (\widetilde{w}_i)^{m_0} \widetilde{R}_i \,, \tag{3.2}$$

$$\widetilde{w}_i = \frac{\left[1/\left(c - S_2(\widetilde{R}_i, \widetilde{R})\right)\right]^{1/(m_0 - 1)}}{\sum_{j=1}^{n} \left[1/\left(c - S_2(\widetilde{R}_i, \widetilde{R})\right)\right]^{1/(m_0 - 1)}} \tag{3.3}$$

$\widetilde{R}$ and $\widetilde{w}_i$ values can be obtained only by an iterative procedure.

As mentioned before, Wang and Parkan (2006) suggests two methods which are called LSDM (Least squares distance method) and DLSM (Defuzzification based least squares method). These methods are based on the solution of the optimization problem, which minimizes the sum of squared distances between all pairs of weighted opinions. In LSDM, fuzzy opinions are used whereas in DLSM, defuzzified values are used in calculations.

Let $\tilde{R}_i = (r_{i1},...,r_{im})$ and $\tilde{R}_j = (r_{j1},...,r_{jm})$ be two fuzzy numbers. m defines the shape of fuzzy number. For instance if m is 3 then the fuzzy number will be a triangular or if it is 4 then fuzzy the number will be a trapezoidal fuzzy number. w_i and w_j values represent weight values.

The minimization problem for the LSDM can be shown as in (3.4).

$$\begin{aligned} & J = \sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n}\left[\sum_{k=1}^{m}\left(w_i r_{ik} - w_j r_{jk}\right)^2\right] \rightarrow \min \\ & \text{s.t.} \\ & \sum_{i=1}^{n} w_i = 1, \quad w_i \geq 0, \, i = 1,...,n \end{aligned} \tag{3.4}$$

The solution of this optimization problem is as shown below,

Theorem 3.1 (Wang & Parkan, 2006)**:** Let $\mathbf{W} = (w_1,...,w_n)^T$ be the optimum solution of the problem (3.4). Then,

$$\mathbf{W} = \frac{\mathbf{G}^{-1}\mathbf{e}}{\mathbf{e}^T\mathbf{G}^{-1}\mathbf{e}} \geq 0 \tag{3.5}$$

where $\mathbf{e} = (1,...,1)$ is the transpose of $\mathbf{e}^T$ and $\mathbf{G}^{-1}$ is the inverse of $\mathbf{G}$, elements of which are defined as

$$g_{ij} = \begin{cases} (n-1)\sum_{k=1}^{m} r_{ik}^2, & i = j = 1,...,n \\ -\sum_{k=1}^{m} r_{ik} r_{jk}, & i, j = 1,...,n;\ i \neq j \end{cases} \tag{3.6}$$

The minimization problem for the DLSM can be shown as below,

$$J = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \left(w_i z_i - w_j z_j \right)^2 \rightarrow \min$$

$$\text{s.t.} \qquad (3.7)$$

$$\sum_{i=1}^{n} w_i = 1, \quad w_i \geq 0, \; i = 1,..., n$$

where z_i represents defuzzification values and are defined as below.

$$z_i = \frac{1}{m} \sum_{k=1}^{m} r_{ik} \qquad (3.8)$$

The solution of this optimization problem is given in the next theorem.

Theorem 3.2 (Wang & Parkan, 2006)**:** Let $\mathbf{W} = \left(w_1,..., w_n \right)^T$ be the optimum solution for the problem (3.7). The optimum solution can be given as below.

$$w_i = \frac{1/z_i}{\sum_{k=1}^{n} \left(1/z_k \right)}, \qquad i = 1,..., n \qquad (3.9)$$

3.3 Formulation of The Performance Evaluation Problem with Linguistic Variables

In student-centered learning system, after each problem based learning session, each student's performance is assessed by using a predefined set of evaluation criteria. These evaluation criteria are specified as leadership, research skill, responsibility, discussion skill and creativity, and are defined by all lecturers in our department (Table 3.1). The importance, hence the weight of each evaluation criterion can be different for each lecturer. Consequently, even when the fuzzy answer can be the same for any evaluation criterion its reflection as a defuzzified value is highly likely to be different for each lecturer because of the different point of views. Because of this, WABL method is used for defuzzification. The fundamental reason for use of WABL method is that this method can be adjusted for the defined task to produce more accurate results when compared to the other known methods (Nasibov, 2003a, 2003b).

Table 3.1 The evaluation form for each problem-based learning session

STUDENT NAME-SURNAME		GRADE
		p
EVALUATION CRITERIA		
1. LEADERSHIP	w_1	□ Absent □ Middle □ Strong
2. CREATIVITY	w_2	□ Absent □ Middle □ Strong
3. RESEARCH SKILL	w_3	□ Absent □ Few □ Middle □ Good □ Strong
4. RESPONSIBILITY	w_4	□ Absent □ Few □ Middle □ Good □ Strong
5. DISCUSSION SKILL	w_5	□ Absent □ Few □ Middle □ Good □ Strong

3.3.1 Determination of The Evaluation Criteria and Their Values

The fuzzy numbers of the evaluation criteria are determined as triangular and trapezoidal fuzzy numbers as shown in Figure 3.1 and Figure 3.2.

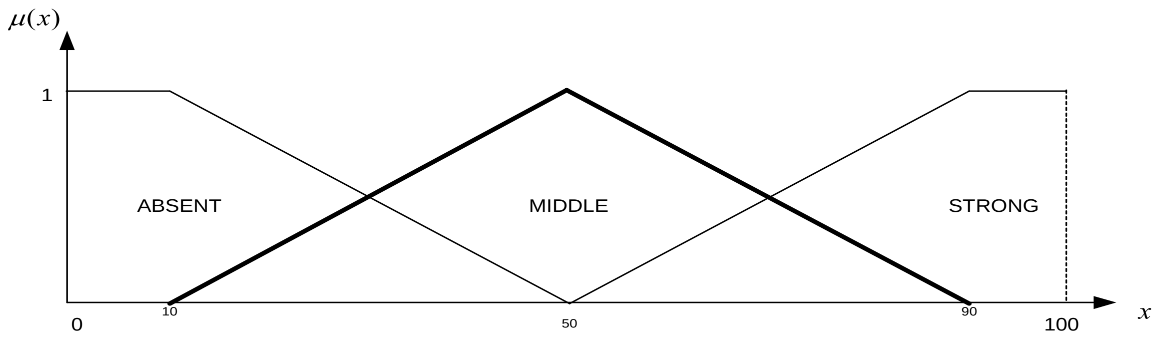


Figure 3.1 Membership function for the criteria Leadership and Creativity.

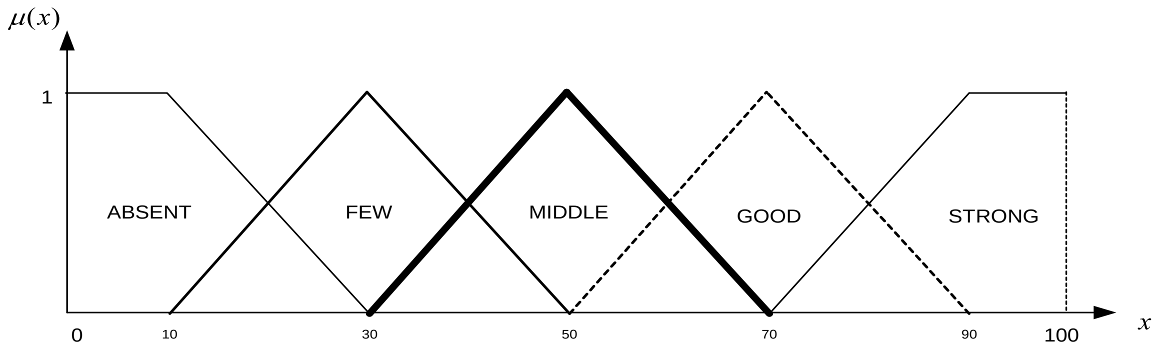


Figure 3.2 Membership function for the criteria Research Skill, Responsibility and Discussion Skill.

In general, membership function of the trapezoidal fuzzy number $A=(a,b,c,d)$ is as follows.

$$\mu_{\tilde{A}}(x)=\begin{cases}0, & x<a\\ \dfrac{x-a}{b-a}, & x\in[a,b)\\ 1, & x\in[b,c)\\ \dfrac{d-x}{d-c}, & x\in[c,d]\\ 0, & x>d\end{cases} \tag{3.10}$$

In this equation, a trapezoidal fuzzy number transforms into a triangular fuzzy number when $b=c$. Similarly, a triangular fuzzy number can be denoted as a trapezoidal fuzzy number.

The defuzzification values of $\tilde{R}_{ij}$ fuzzy numbers can be obtained from any of defuzzification methods such as explained in Lin and Lee (1996), Mendel (2001), Pedrycz and Gomide (1998). However, for the reason indicated in the beginning of this section, we have concentrated on the use of WABL method in our study.

WABL values that are used in our study for the defuzzification of the triangular and trapezoidal fuzzy numbers can be calculated as below.

Theorem 3.3 (Nasibov & Mert, 2005; 2007e)**:** Let $A=(a,b,c)$ be a triangular fuzzy number. $p(\alpha)=(q+1)\alpha^q$, $q\geq 0$ and c_L, c_R can be any values that satisfy normality and positivity conditions. The WABL value of the fuzzy number A can be calculated as follows.

$$I(A)=c_R\left[c-\frac{q+1}{q+2}(c-b)\right]+c_L\left[a+\frac{q+1}{q+2}(b-a)\right] \tag{3.11}$$

Theorem 3.4 (Nasibov & Mert, 2005; 2007e)**:** Let $A=(a,b,c,d)$ be a trapezoidal fuzzy number with membership function (3.10). $p(\alpha)=(q+1)\alpha^q$, $q \geq 0$ and c_L, c_R can be any values that satisfy normality and positivity conditions. The WABL value of the fuzzy number A can be calculated as follows.

$$I(A) = c_R\left[d - \frac{q+1}{q+2}(d-c)\right] + c_L\left[a + \frac{q+1}{q+2}(b-a)\right] \tag{3.12}$$

3.3.2 An Optimization Formulation of The Performance Evaluation Problem

Assume that n student performances will be evaluated by using m evaluation criteria. The evaluation results are indicated as $\widetilde{R}_{ij}$ for each student i as to criterion j where $i=1,\ldots,n$ and $j=1,\ldots,m$. $p_1,p_2,\ldots,p_n$ are the numerical values of students' grades after evaluation of each respective student. Also $\mathbf{W}=(w_1,w_2,\ldots,w_m)^T$ represents the unknown weights of the m evaluation criteria. R_{ij} values are the defuzzified values of the $\widetilde{R}_{ij}$ fuzzy values. Our objective function is to minimize the sum of squared distances between all grade evaluations and the defuzzified values of linguistic evaluations of each criterion. Therefore, our nonlinear optimization problem can be constituted as follows.

$$\begin{aligned} &L(\mathbf{W}) = \sum_{i=1}^{n}\left(\sum_{j=1}^{m} w_j R_{ij} - p_i\right)^2 \quad \rightarrow \min \\ &\text{s. t.} \\ &\sum_{j=1}^{m} w_j = 1, \quad j = 1,\ldots, m \end{aligned} \tag{3.13}$$

Problem (3.13) is investigated in the next section.

3.4 Solution Method and Algorithm of The Performance Evaluation Problem

3.4.1 An Optimal Solution of The Problem for Fixed β

We want to determine the $w_j, j=1,...,m$ values that minimize the expression $L(\mathbf{W})$. There is no sign restriction and no upper limit for the value of $w_j, j=1,...,m$. Increasing values of $w_j, j=1,...,m$ show their effectiveness and the signs show their positive or negative effect which will be found as below.

Theorem 3.5: Vector $\mathbf{W}^* = \left(w_1^*, w_2^*, \ldots, w_m^*\right)^T$, which represents the optimum solution for the problem (3.13), is as below,

$$\mathbf{W}^* = \frac{\mathbf{G}^{-1}\mathbf{e}\left(1-\mathbf{e}^T\mathbf{G}^{-1}\mathbf{R}^T\mathbf{P}\right)}{\mathbf{e}^T\mathbf{G}^{-1}\mathbf{e}} + \mathbf{G}^{-1}\mathbf{R}^T\mathbf{P}, \tag{3.14}$$

where $\mathbf{e} = (1,1,\ldots,1)$ is the transpose of $\mathbf{e}^T$ and $\mathbf{G}^{-1}$ is the inverse of $\mathbf{G}$. $\mathbf{G}$ and $\mathbf{R}^T$ matrices and $\mathbf{W}$ and $\mathbf{P}$ vectors are respectively as follows.

$$\mathbf{G} = \begin{bmatrix} \sum_{i=1}^{n} R_{i1}^2 & \sum_{i=1}^{n} R_{i1}R_{i2} & \cdots & \sum_{i=1}^{n} R_{i1}R_{im} \\ \sum_{i=1}^{n} R_{i1}R_{i2} & \sum_{i=1}^{n} R_{i2}^2 & \cdots & \sum_{i=1}^{n} R_{i2}R_{im} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{n} R_{i1}R_{im} & \sum_{i=1}^{n} R_{i2}R_{im} & \cdots & \sum_{i=1}^{n} R_{im}^2 \end{bmatrix}_{m\times m} \tag{3.15}$$

$$\mathbf{R}^T = \begin{bmatrix} R_{11} & R_{21} & \cdots & R_{n1} \\ R_{12} & R_{22} & \cdots & R_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ R_{1m} & R_{2m} & \cdots & R_{nm} \end{bmatrix}_{m\times n} \tag{3.16}$$

$$\mathbf{W} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}_{m\times 1} \qquad \mathbf{P} = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix}_{n\times 1} \tag{3.17}$$

Proof: Lagrange multiplier method can be applied to the problem (3.13) as below.

$$L(\mathbf{W}) = \sum_{i=1}^{n}\left(\sum_{j=1}^{m} w_j R_{ij} - p_i\right)^2 - 2\lambda\left(\sum_{j=1}^{m} w_j - 1\right) \tag{3.18}$$

Letting $\dfrac{\partial L(\mathbf{W})}{\partial w_z} = 0$ for each $z = 1,...,m$ we obtain

$$\frac{\partial L(\mathbf{W})}{\partial w_z} = 2\sum_{i=1}^{n}\left(\sum_{j=1}^{m} w_j R_{ij} - p_i\right)\cdot R_{iz} - 2\lambda = 0, \quad z = 1,...,m \tag{3.19}$$

that can be simplified as

$$\sum_{i=1}^{n}\left(\sum_{j=1}^{m} w_j R_{ij} R_{iz} - p_i R_{iz}\right) - \lambda = 0, \quad z = 1,...,m \tag{3.20}$$

Expression (3.20) can be rewritten in matrix form as

$$\mathbf{GW} - \mathbf{R}^T\mathbf{P} - \lambda\mathbf{e} = 0 \tag{3.21}$$

where $\mathbf{G}$ and $\mathbf{R}^T$ matrices are defined in (3.15) and (3.16) respectively. As defined in problem (3.13), sum of all weight values must be equal to “1” and this restriction can be rewritten as below.

$$\mathbf{e}^T\mathbf{W} = 1 \tag{3.22}$$

Consequently, (3.21) and (3.22) can be solved together as

$$\lambda^* = \frac{1 - \mathbf{e}^T\mathbf{G}^{-1}\mathbf{R}^T\mathbf{P}}{\mathbf{e}^T\mathbf{G}^{-1}\mathbf{e}} \tag{3.23}$$

and

$$\mathbf{W}^* = \frac{\mathbf{G}^{-1}\mathbf{e}\left(1-\mathbf{e}^T\mathbf{G}^{-1}\mathbf{R}^T\mathbf{P}\right)}{\mathbf{e}^T\mathbf{G}^{-1}\mathbf{e}} + \mathbf{G}^{-1}\mathbf{R}^T\mathbf{P} \tag{3.24}$$

As a result, weight values can be calculated for each evaluation criterion owing to the equation (3.24).

3.4.2 An Optimal Solution of The Problem for Optimal β

The calculated weight values that in previous section are valid for the optimism degree of $\beta = c_R$ which is predefined before the calculations explained above. Therefore, the weight values, which are calculated for the optimal optimism degree, will also be the optimal solution. With this point of view, we can expand problem (3.13) by using equation (3.12) as shown in (3.25). In other words, when we rearrange defuzzification operation of fuzzy values $\widetilde{R}_{ij}$ with WABL method, the solution below will provide us both the optimal optimism degree β^* and optimal weight values w_j^* for the obtained β^*.

$$\begin{aligned}
&L(\beta,\mathbf{W}) = \sum_{i=1}^{n}\left(\sum_{j=1}^{m} w_j\left[\beta A_{ij} + (1-\beta)B_{ij}\right] - p_i\right)^2 \quad \rightarrow \min \\
&\text{s.t}: \\
&\sum_{j=1}^{m} w_j = 1, \quad j = 1,\dots,m \\
&0 \le \beta \le 1
\end{aligned} \tag{3.25}$$

$(a_{ij}, b_{ij}, c_{ij}, d_{ij})$ values are the fuzzy number characteristics for each evaluation criterion value $\widetilde{R}_{ij}$. By substituting c_R with β, c_L with $1-\beta$ and using definitions (3.26) and (3.27), defuzzified value R_{ij} can be written as (3.28) by using the Theorem 3.4.

$$A_{ij} = d_{ij} - \frac{q+1}{q+2}(d_{ij} - c_{ij}) \tag{3.26}$$

$$B_{ij} = a_{ij} - \frac{q+1}{q+2}(b_{ij} - a_{ij}) \tag{3.27}$$

$$R_{ij} = \beta A_{ij} + (1-\beta)B_{ij} \tag{3.28}$$

By derivation the goal function of problem (3.25) with respect to β and equating to zero we obtain,

$$\frac{\partial L(\beta, \mathbf{W})}{\partial \beta} = 2\sum_{i=1}^{n}\left(\sum_{j=1}^{m}\left[\beta\left(w_j A_{ij} - w_j B_{ij}\right) + w_j B_{ij} - p_i\right]\sum_{j=1}^{m}\left[w_j\left(A_{ij} - B_{ij}\right)\right]\right) = 0 \tag{3.29}$$

Thus we have,

$$\beta = -\frac{\sum_{i=1}^{n}\left(\sum_{j=1}^{m} w_j (A_{ij} - B_{ij})(w_j B_{ij} - p_i)\right)}{\sum_{i=1}^{n}\left(\sum_{j=1}^{m} w_j{}^2 (A_{ij} - B_{ij})^2\right)} \tag{3.30}$$

or in the matrix form,

$$\beta = -\frac{(\mathbf{B}\mathbf{W}^* - \mathbf{P})^T (\mathbf{A} - \mathbf{B})\,\mathbf{W}^*}{\mathbf{W}^{*T} (\mathbf{A} - \mathbf{B})^T (\mathbf{A} - \mathbf{B})\,\mathbf{W}^*} \tag{3.31}$$

Lemma 3.1: $L(\beta, \mathbf{W})$ is a convex function with respect to β.

Proof:

$$\begin{aligned}\frac{\partial^2 L(\beta, \mathbf{W})}{\partial \beta^2} &= \left[w_1(A_{11} - B_{11}) + \ldots + w_m(A_{1m} - B_{1m})\right]^2 + \ldots + \\ &\ldots + \left[w_1(A_{n1} - B_{n1}) + \ldots + w_m(A_{nm} - B_{nm})\right]^2 \geq 0\end{aligned} \tag{3.32}$$

In the matrix form it could be shown as below.

$$[(\mathbf{A}-\mathbf{B})\mathbf{W}]^T[(\mathbf{A}-\mathbf{B})\mathbf{W}] \geq 0 \tag{3.33}$$

Where $\mathbf{A}-\mathbf{B} = \begin{bmatrix} A_{11}-B_{11} & A_{12}-B_{12} & \cdots & A_{1m}-B_{1m} \\ A_{21}-B_{21} & A_{22}-B_{22} & \cdots & A_{2m}-B_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1}-B_{n1} & A_{n2}-B_{n2} & \cdots & A_{nm}-B_{nm} \end{bmatrix}_{n\times m}$, $\mathbf{W} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}_{m\times 1}$

In Figure 3.3, SSE (Sum of squared error) versus β for different data sets is shown. It can be seen that all these functions are convex functions.

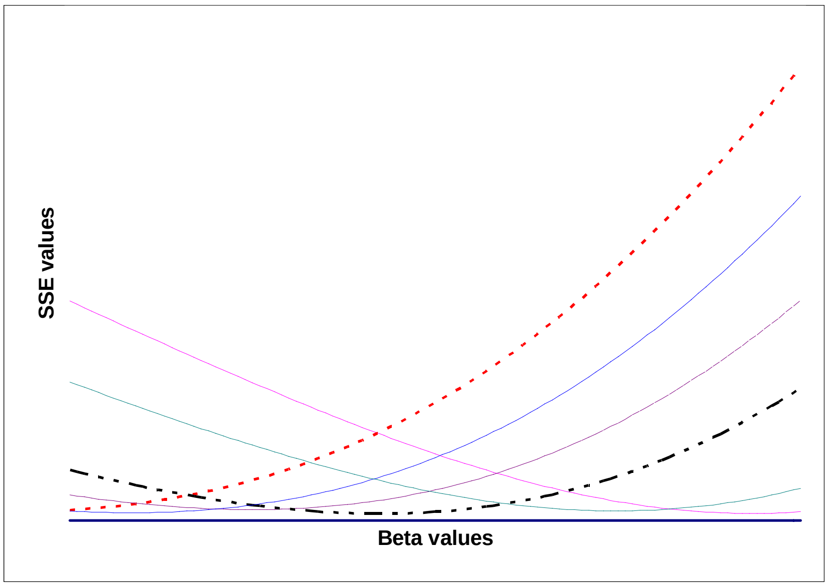


Figure 3.3 Beta values versus SSE values for six data sets.

Any standard optimization procedure (Golden Section, Binary Section, etc.) can be used for the calculation of β value because of convexity of $L(\beta, \mathbf{W})$ function with respect to β values.

3.4.3 An Iterative Solution Algorithm of The Problem

In this section, a more effective iterative algorithm than classical optimization methods (for example Golden Section method), which is explained below, is suggested.

Initially, $\mathbf{R}$ and $\mathbf{W}_0 = \dfrac{\mathbf{G}^{-1}\mathbf{e}\left(1-\mathbf{e}^T\mathbf{G}^{-1}\mathbf{R}^T\mathbf{P}\right)}{\mathbf{e}^T\mathbf{G}^{-1}\mathbf{e}} + \mathbf{G}^{-1}\mathbf{R}^T\mathbf{P}$ are calculated as the solution of problem (3.25) by using an initial β_0. Then, new $\beta = -\dfrac{(\mathbf{BW}-\mathbf{P})^T(\mathbf{A}-\mathbf{B})\mathbf{W}}{\mathbf{W}^T(\mathbf{A}-\mathbf{B})^T(\mathbf{A}-\mathbf{B})\mathbf{W}}$ is calculated by using $\mathbf{W}_0$ and iterations are repeated. In other words, the optimal solution of problem (3.25) could be obtained as related to the iteration $\beta_0 \rightarrow \mathbf{W}_0 \rightarrow \beta_1 \rightarrow \mathbf{W}_1 \rightarrow \beta_2 \rightarrow \cdots \rightarrow \beta^* \rightarrow \mathbf{W}^*$. We can illustrate this procedure as an algorithm as shown below.

Algorithm 3.1:

Step 0. Linguistic evaluations like leadership, creativity, etc. are entered for each student,

Step 1. Numerical grade values, which coincide with the linguistic evaluations, are entered. ($\mathbf{P}$ vector)

Step 2. $\varepsilon > 0$ certainty is defined, β_0 and k are initialized as 0.5 and 0 respectively.

Step 3. For the β_k value, R_{ij} for $i = 1,...,n$ and $j = 1,...,m$ are calculated from formula (3.28).

Step 4. $\mathbf{G}$ is calculated from formula (3.15) and then $\mathbf{G}^{-1}$ is obtained.

Step 5. w_j^* for $j = 1,...,m$ are calculated from formula (3.24).

Step 6. $\mathbf{A}$ and $\mathbf{B}$ matrices are calculated from formulas (3.26) and (3.27).

Step 7. β^* value is obtained from formula (3.31).

Step 8. If $\left|\beta_k - \beta^*\right| < \varepsilon$ then go to Step 10.

Step 9. Update k as $k+1$ and β_k as β^*, then go to Step 3.

Step 10. w_j^* for $j = 1,...,m$ and β^* are determined as optimal parameters.

Step 11. Stop.

When compared with classical optimization algorithms, this algorithm gives more effective results. The results of this comparison are given in Chapter 5.

CHAPTER FOUR
GROUP CONSTITUTION PROBLEM WITH DIFFERENT STRATEGIES

4.1 Introduction

Although the name "assignment problem" seems to have first discussed in 1952 by Votaw and Orden, in fact, it was firstly recognized in the beginning of the development of practical solution methods and variations on the assignment problem in Kuhn's study (1955). Afterwards, many variations of the problem have been studied by many researchers (Caron, Hansen, & Jaumard, 1999; Cattrysse & Van Wassenhove, 1992; Daskalai, Birbas, & Housos, 2004; Dell'Amico & Martello, 1997; Duin & Volgenant, 1991; Ford & Fulkerson, 1966; Gross, 1959; Gupta & Punnen, 1988; Martello, Pulleyblank, Toth, & Werra, 1984; Punnen & Aneja, 1993; Werra, 1985). Workers' placement problems, bin packing problems, and task allocations problems can be given as some examples of the variations of the assignment problems.

In real life, assignment problems are very usual. These problems contain optimally matching the elements of two or more sets. When there are two sets, they may be referred to as "tasks" and "agents". For example, "tasks" may be jobs that need to be done and "agents" the people or machines that can do them. In general, assignment problems involves assignment each of task to individual agent. However, these problems may have different structures according to the matching that needs to be performed between tasks and agents such as assignment of multiple tasks to the same agent or multiple agents to a single task. In our group constitution problem, students must be assigned to individual groups similar to the assignment of multiple tasks to a single agent.

In this chapter, the heuristic assignment approaches proposed in this dissertation are examined as the objective, mathematical model, algorithm and assignment type. In this study, for the assignment process that is taken into consideration, some improvements also can be made with respect to the definite properties after the suggested application of assignment process to the groups in fact. For example, while assigning students to different groups, improvements to ensure approximately equal distribution of genders within each group or to group students who understand each other well enough etc. can be used. Similar studies of such kind of assignment process with use of such improvements are presented in Nasibov (2004), Nasibov and Nasibova (2003c) and Nasibov and Kınay (2006c, 2006d). However, in this study, such improvements are not employed for the assignment process..

Besides, so as to be able to use the heuristic assignment approaches mentioned in this chapter, first of all students have to be divided into clusters according to their success status. Therefore, the FCM method mentioned in Chapter 2 was used as the clustering method. The clusters that are constituted are named as *success clusters*. For success clusters are identified: “very good”, “good”, “middle” and “bad”.

4.2 Random Group Constitution Strategies

Three heuristic methods are proposed so as to be used in the assignment of students into different groups and the mathematical models of two of these methods are proposed. These methods are named as *balanced random assignment, simple random assignment,* and *level-based random assignment*. The working mechanisms of these methods are detailed as follows.

4.2.1 Balanced Random Assignment

The purpose of using this assignment method is to form small groups that reflect the features of the whole class. That is to say, regardless the group sizes, students from all success clusters will be distributed equally between each group. The mathematical model of this method is as follows.

Suppose we want to assign n students to k groups as balanced random. Let the success grade of each student be shown as $p_i, i = 1,...,n$. In addition, suppose it x_{ij} takes value of “1” if student i is assigned to group j, otherwise it takes “0”. That is,

$$x_{ij} = \begin{cases} 1, & \text{if student } i \text{ is assigned to group } j \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

There exist the following two constraints in the assignment of n students to m groups as balanced random:

a. A student can be assigned to only one group. That is to say,

$$\sum_{j=1}^{k} x_{ij} = 1, \qquad i = 1,\ldots,n . \tag{4.2}$$

b. Group sizes have to be approximately equal. Therefore, the student numbers in each group have to be as much as the upper limit of n/k at the most and the lower limit of n/k at the least.

$$\left\lfloor \frac{n}{k} \right\rfloor \le \sum_{i=1}^{n} x_{ij} \le \left\lceil \frac{n}{k} \right\rceil, \qquad j = 1,\ldots,k \tag{4.3}$$

So as to carry out balanced random assignment or, in other words, to assign students to small groups to reflect the structure of the class, the solution which minimizes the sum of squares of the difference between group averages and the general average of the class would be the optimum solution. That is to say, mathematically, the optimization of the problem (4.4) is required under the constraints (4.1)-(4.3). The problem (4.4) is shown with the constraints as follows.

$$\sum_{j=1}^{k} \left(\frac{\sum_{i=1}^{n} x_{ij} p_i}{n_j} - \bar{x} \right)^2 \rightarrow \min \tag{4.4}$$

s. t.

$$\sum_{j=1}^{k} x_{ij} = 1, \qquad i = 1,\ldots,n$$

$$\left\lfloor \frac{n}{k} \right\rfloor \le \sum_{i=1}^{n} x_{ij} \le \left\lceil \frac{n}{k} \right\rceil, \qquad j = 1,\ldots,k$$

$$x_{ij} \in \{0,1\} \quad i = 1,\ldots,n \text{ , } j = 1,\ldots,k$$

In this kind of assignment problems, since it is mostly not possible even to describe the problem in any available software programmes, heuristic solution algorithms are developed for this and other proposed assignment methods. Thus, the algorithm of the heuristic approach developed for the solution of this mathematical model is as follows.

Algorithm 4.1:

Step 0. The numerical grade evaluations of the students are calculated.

Step 1. Success clusters are constituted for students by using clustering algorithm.

Step 2. Students are selected randomly from the success cluster “Very Good” and they are assigned to groups one by one. When the number of groups to be constituted is completed, assignment is carried on to the first group again and this continues.

Step 3. Students are selected one by one randomly from the success cluster “Good” and assignment is carried on where the assignment remained. The same procedure is carried out also for the success clusters “Middle” and “Bad” and the assignment procedure is terminated.

The demonstration of the method is as in Figure 4.1.

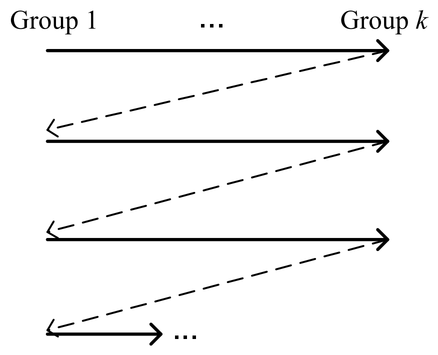


Figure 4.1 Working way of Balanced Random Assignment method

For example, suppose that 19 students will be assigned to 4 groups. Let 5 of these students belong to the cluster “Very Good”, 3 to “Good”, 8 to “Middle” and 3 to “Bad”. The assignment of the students according to this assignment principle would be as in Table 4.1.

Table 4.1 Assignment results according to Balanced Random Assignment method

Group 1	Group 2	Group 3	Group 4
VG	VG	VG	VG
VG	G	G	G
M	M	M	M
M	M	M	M
B	B	B	

4.2.2 Simple Random Assignment

In this assignment method, the assignment of students to the groups is completely random. That is to say, without paying attention in which cluster the students are, students are assigned to groups completely in a random manner. The only constraint in this procedure is again that the number of students in each group to be approximately equal. Also according to the simple random assignment method, the assignment type is identical to the previous method. The algorithm of this method is as follows.

Algorithm 4.2:

Step 0. The numerical grade evaluations of the students are calculated.

Step 1. Without using any clustering procedures, students are chosen randomly one by one and assigned to groups respectively.

4.2.3 Level-Based Random Assignment

In the level-based random assignment method, provided that students are again selected randomly from within each success cluster as in balanced random assignment method, assignment procedure is carried out in a way that students with similar grades will gather in one group. Therefore, the aim of this method is to form groups from students with similar grades.

The mathematical model of this assignment method is given in problem (4.5). The objective function of this method is based on the principle of maximization of the squares of the differences between the total grades in each group. The constraints are identical to those of problem (4.4).

$$\sum_{j=1}^{k}\left(\frac{\sum_{i=1}^{n} x_{ij} p_i}{n_j} - \overline{x}\right)^2 \to \max \tag{4.5}$$

s. t.

$$\sum_{j=1}^{k} x_{ij} = 1, \qquad i = 1,\dots,n$$

$$\left\lfloor \frac{n}{k} \right\rfloor \le \sum_{i=1}^{n} x_{ij} \le \left\lceil \frac{n}{k} \right\rceil, \qquad j = 1,\dots,k$$

$$x_{ij} \in \{0,1\} \quad i = 1,\dots,n,\ j = 1,\dots,k$$

The algorithm of this heuristic approach proposed for the solution of this model is as follows.

Algorithm 4.3:

Step 0. The numerical grade evaluations of the students are calculated.

Step 1. Success clusters are formed for students by using clustering algorithm.

Step 2. Students are selected randomly from the success cluster “Very Good” and assigned one by one to the same group. When the number of students to be assigned to the group is attained, the following students are assigned to the following group.

Step 3. Students are selected randomly one by one from the success cluster “Good” and the assignment procedure is carried on from where the assignment remained. The same procedure is carried out for the success clusters “Middle” and “Bad” as well and the assignment procedure is terminated.

In this method, the assignment type of students to groups is a little bit different from the first two methods and as in Figure 4.2. Furthermore, the results over the same example given at the end of section 4.2.1 according to this assignment principle would be as in Table 4.2.

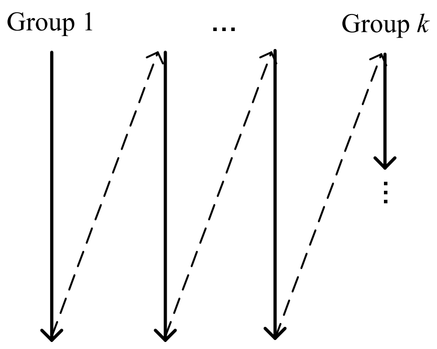


Figure 4.2 Working way of Level-Based Random Assignment

Table 4.2 Assignment results according to Level-Based Random Assignment

Group 1	Group 2	Group 3	Group 4
VG	G	M	M
VG	G	M	B
VG	G	M	B
VG	M	M	B
VG	M	M	

4.3 Deterministic Group Constitution Strategies

Two heuristic methods are proposed so as to be used in the assignment of students in groups and their respective mathematical models are formulated. These methods are named *balanced assignment* and *level-based assignment*. The working mechanisms of these methods are summarized as below.

4.3.1 Balanced Assignment

The aim of using this assignment method is similar to the aim of using the balanced random assignment method. In this method, it is intended to form student groups which would represent the characteristics of the whole class much better than random balanced assignment. Briefly, it is aimed at reaching a closer result to optimum.

The mathematical model is given in (4.6). The objective function of this method is based on the principle of minimization of the squares of the differences between the total grades in the groups constituted.

$$\sum_{k_1=2}^{k} \sum_{k_2=1}^{k_1-1} \left(\sum_{i=1}^{n} x_{ik_1} p_i - \sum_{i=1}^{n} x_{ik_2} p_i \right)^2 \rightarrow \min \tag{4.6}$$

s. t.

$$\sum_{j=1}^{k} x_{ij} = 1, \qquad i = 1,\dots,n$$

$$\left\lfloor \frac{n}{k} \right\rfloor \le \sum_{i=1}^{n} x_{ij} \le \left\lceil \frac{n}{k} \right\rceil, \qquad j = 1,\dots,k$$

$$x_{ij} \in \{0,1\} \quad i = 1,\dots,n \;,\; j = 1,\dots,k$$

Actually, the objective functions of problem (4.4) and (4.6) are the same. The algorithm of the heuristic approach developed for the solution of this mathematical model is as follows.

Algorithm 4.4:

Step 0. The numerical grade evaluations of the students are calculated.

Step 1. All students are arranged in descending order according to their grades. ($p_{(1)} \le p_{(2)} \le \dots \le p_{(n)}$)

Step 2. The first student is assigned to the first group, the second one is assigned to the second group etc. until student *k* is assigned to the last group *k*. After this, the next student is assigned to the last group *k* and the following students are continued to be assigned towards the first group. The assignment procedure is carried on in this way and all students are assigned to groups.

To understand this algorithm better, the assignment type is demonstrated in Figure 4.3.

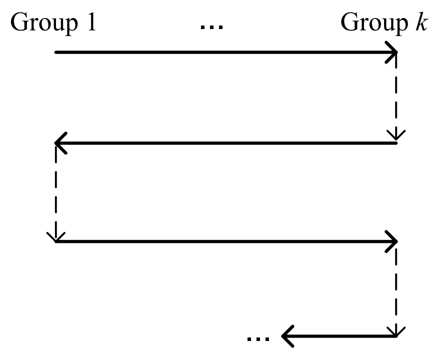


Figure 4.3 Working way of Balanced Assignment

Therefore, the groups formed according to this assignment principle will be as in Table 4.3. Since all students are arranged in descending order, the solution of this assignment method is unique.

Table 4.3 Assignment results according to Balanced Assignment

Group 1	Group 2	Group 3	Group 4
$p_{(1)}$	$p_{(2)}$	$p_{(3)}$	$p_{(4)}$
$p_{(8)}$	$p_{(7)}$	$p_{(6)}$	$p_{(5)}$
$p_{(9)}$	$p_{(10)}$	$p_{(11)}$	$p_{(12)}$
$p_{(16)}$	$p_{(15)}$	$p_{(14)}$	$p_{(13)}$
$p_{(17)}$	$p_{(18)}$	$p_{(19)}$	

To compare the validity of the grouping by using this method an error ratio calculated. This is obtained by the ratio of the difference between the highest and lowest of the group averages to the highest and lowest of the student grades and the results are given under the title of "Group Constitution Results" in Chapter 5.

$$\text{Error Ratio} = \frac{\max\limits_{i,j}\left|\bar{p}^{i} - \bar{p}^{j}\right|}{\max\limits_{i,j}\left|p_{i} - p_{j}\right|} \tag{4.7}$$

In this equation, $\bar{p}^{i}$ and $\bar{p}^{j}$ values indicate the average grade of the groups constituted as a result of assignment when $i, j = 1,...,k$ and p_i and p_j values indicate the grades of the students when $i, j = 1,...,k$.

4.3.2 Level-Based Assignment

In the final method, the aim is identical with the level-based random assignment method. However, since students are arranged according to the their grades here, like in the previous method, students with much similar grades will gather.

The objective function in this method is based on the maximization of the sum of squares of the differences between the total grades of the students in each group. Therefore, the mathematical model of the level-based assignment approach is as in problem (4.8).

$$\sum_{k_1=2}^{k}\sum_{k_2=1}^{k_1-1}\left(\sum_{i=1}^{n} x_{ik_1}p_i - \sum_{i=1}^{n} x_{ik_2}p_i\right)^2 \rightarrow \max \tag{4.8}$$

s. t.

$$\sum_{j=1}^{k} x_{ij} = 1, \qquad i = 1,\dots,n$$

$$\left\lfloor \frac{n}{k} \right\rfloor \le \sum_{i=1}^{n} x_{ij} \le \left\lceil \frac{n}{k} \right\rceil, \qquad j = 1,\dots,k$$

$$x_{ij} \in \{0,1\} \quad i = 1,\dots,n,\ j = 1,\dots,k$$

Algorithm 4.5:

Step 0. The numerical grade evaluations of the students are calculated.

Step 1. All students are arranged in descending order according to their grades. ($p_{(1)} \le p_{(2)} \le \dots \le p_{(n)}$)

Step 2. The assignment starts by filling the first group with students until this group is full. When the number of students to be assigned to the group is reached, the following students continue to be assigned to the following group.

The group results according to this assignment principle for the same example will be as in Table 4.4. Since all students are arranged in descending order, the solution of this assignment method is also unique, like the solution of the previous method.

Table 4.4 Assignment results according to Level-Based Assignment

Group 1	**Group 2**	**Group 3**	**Group 4**
$p_{(1)}$	$p_{(6)}$	$p_{(11)}$	$p_{(16)}$
$p_{(2)}$	$p_{(7)}$	$p_{(12)}$	$p_{(17)}$
$p_{(3)}$	$p_{(8)}$	$p_{(13)}$	$p_{(18)}$
$p_{(4)}$	$p_{(9)}$	$p_{(14)}$	$p_{(19)}$
$p_{(5)}$	$p_{(10)}$	$p_{(15)}$	

CHAPTER FIVE
APPLICATIONS AND EXPERIMENTAL RESULTS

5.1 Introduction

In this chapter, the analysis results concerning the efficiency of the iterative method proposed for the student performance evaluation are given. The efficiency results for the balanced assignment method, one of the assignment strategies proposed for the constitution of student groups, have been obtained and examined by using data in different sample sizes and according to different numbers of group assignments. Furthermore, besides numerical results, Optimal Weights Evaluation and Group Constitution programs, implemented in Borland C++ Builder 6.0 SDK, are explained in detail.

5.2 Performance Evaluation Tools and Experimental Results

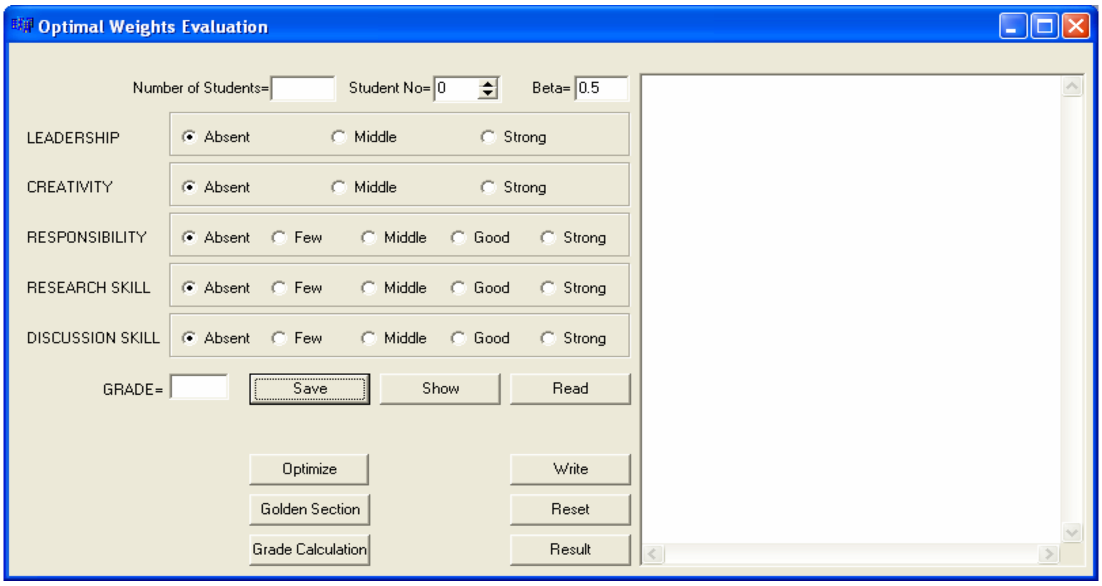
5.2.1 Performance Evaluation Tools

In this section, detailed information is given concerning the forms, functional modules and informative components constituted in the Optimal Weights Evaluation program.

5.2.1.1 Forms

When the program starts to run, the window in Figure 5.1 appearson the screen. Student data is entered in this window. This data can be entered either one by one for each student or form a file consisting this information for each student. Provided that data is entered one by one, "Number of Students" field has to be populated first. For every "Student No" entered, all the linguistic evaluation results and the equivalent graduations are entered and "Save" button is clicked. Owing to this procedure, data will be saved as lines into c:\RRR.txt, for each student. The "Student No" field has to be populated with increments of one for each student to be entered later on. "Show" button can be used at any time to display the contents of the flat file in the text box on the right handside of the form. The information to be displayed here will show the numerical results of the students' linguistic evaluations and their grade values which were calculated on the basis of the "Beta" value determined previously.

In order to calculate the parameters representing the information entered, it is required to push "Optimize" button, which performs the Algorithm 3.1. As a result of this procedure, parameters will have been determined for the student evaluations that have been entered, the calculated value will have been transferred to the "Beta" value on the window and at the same time, these results will have been transferred to "Grade Calculation" part in order to be used in the required calculations. The image of the program, where the parameters have been estimated by pushing "Optimize" button after the data are entered one by one, is as in Figure 5.2.


Optimal Weights Evaluation

Number of Students= | Student No= 0 | Beta= 0.5

LEADERSHIP: [✓ Absent] Middle Strong

CREATIVITY: [✓ Absent] Middle Strong

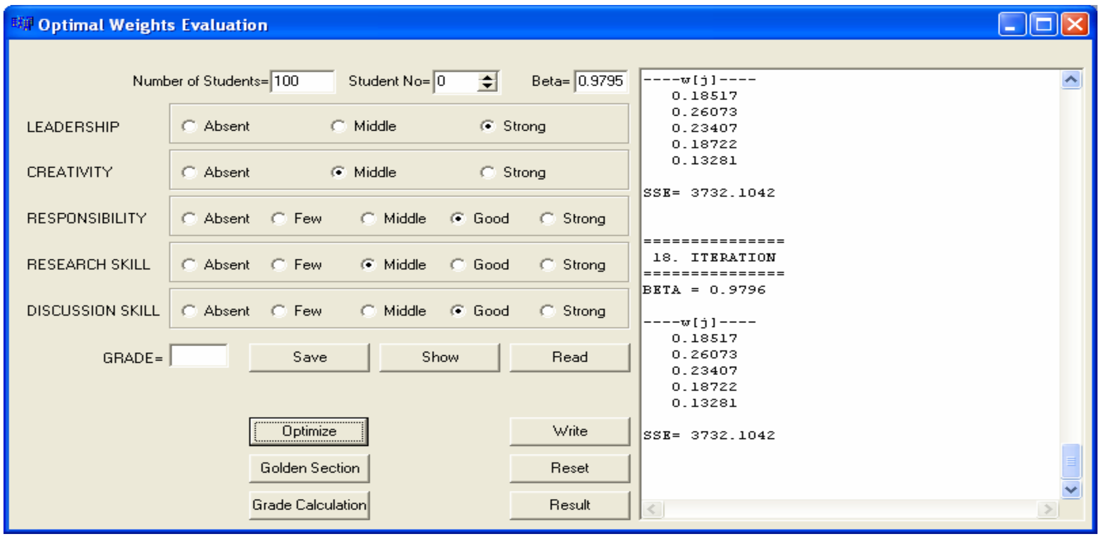
RESPONSIBILITY: [✓ Absent] Few Middle Good Strong

RESEARCH SKILL: [✓ Absent] Few Middle Good Strong

DISCUSSION SKILL: [✓ Absent] Few Middle Good Strong

GRADE=

Figure 5.1 Opening window of the Optimal Weights Evaluation program.



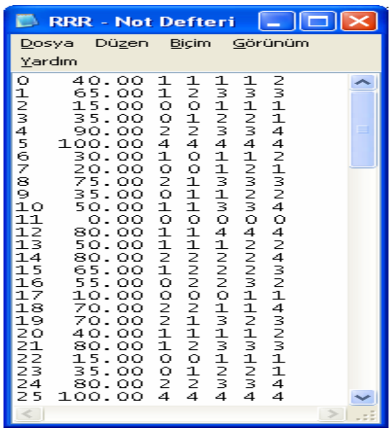
Number of Students= 100 Student No= 0 Beta= 0.9795

LEADERSHIP: Absent Middle [✓ Strong]
CREATIVITY: Absent [✓ Middle] Strong
RESPONSIBILITY: Absent Few Middle [✓ Good] Strong
RESEARCH SKILL: Absent Few [✓ Middle] Good Strong
DISCUSSION SKILL: Absent Few Middle [✓ Good] Strong
GRADE=

```
----w[j]----
   0.18517
   0.26073
   0.23407
   0.18722
   0.13281

SSE= 3732.1042


===============
 18. ITERATION
===============
BETA = 0.9796

----w[j]----
   0.18517
   0.26073
   0.23407
   0.18722
   0.13281

SSE= 3732.1042
```

Figure 5.2 Image after the information is read from the file and the “Optimize” button is pushed.

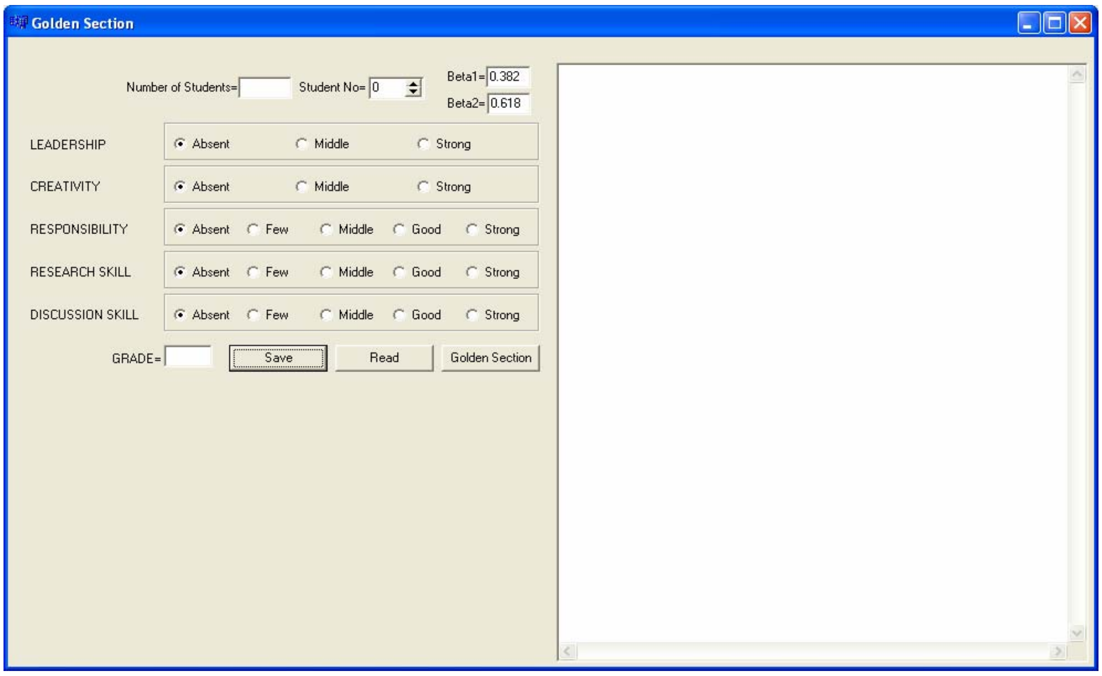
If the data is to be read from the flat file, first of all, the data has to be encoded as shown in Figure 5.3. This file must be saved as c:\RRR.txt. Each line in this file is composed of students order number, the grade evaluation result of the student, followed by five columns of encoded results of performance evaluation criteria. The encoding procedure is in the form of Absent=0, Middle=1 and Strong=2 for the first two criteria while it is in the form of Absent=0, Few=1, Middle=2, Good=3 and Strong=4 for the other three criteria. After the file is created in this way, data matrices are populated by clicking the “Read” button. Later on, as in the previous, “Optimize” button is pushed and optimum optimism degree, β, and w_j, $j = \overline{1,m}$ optimal weight values are determined owing to the iterative method.



```
0     40.00 1 1 1 1 2
1     65.00 1 2 3 3 3
2     15.00 0 0 1 1 1
3     35.00 0 1 2 2 1
4     90.00 2 2 3 3 4
5    100.00 4 4 4 4 4
6     30.00 1 0 1 1 2
7     20.00 0 0 1 2 1
8     75.00 2 1 3 3 3
9     35.00 0 1 1 2 2
10    50.00 1 1 3 3 4
11     0.00 0 0 0 0 0
12    80.00 1 1 4 4 4
13    50.00 1 1 1 2 2
14    80.00 2 2 2 2 4
15    65.00 1 2 2 2 3
16    55.00 0 2 2 3 2
17    10.00 0 0 0 1 1
18    70.00 2 2 1 1 4
19    70.00 2 1 3 2 3
20    40.00 1 1 1 1 2
21    80.00 1 2 3 3 3
22    15.00 0 0 1 1 1
23    35.00 0 1 2 2 1
24    80.00 2 2 3 3 4
25   100.00 4 4 4 4 4
```

Figure 5.3 Image of the file where encoded student information exists.

After "Golden Section" activating the button, the new screen in Figure 5.4 will appear. Data to be used for obtaining the optimum optimism degree will be entered in this screen regardless the way the student data is entered. When the "Golden Section" button on this form is activated, the result will be as in Figure 5.5. This window has been implemented so as to debug the iterative method proposed in this dissertation work according to the well-known Golden Section method.

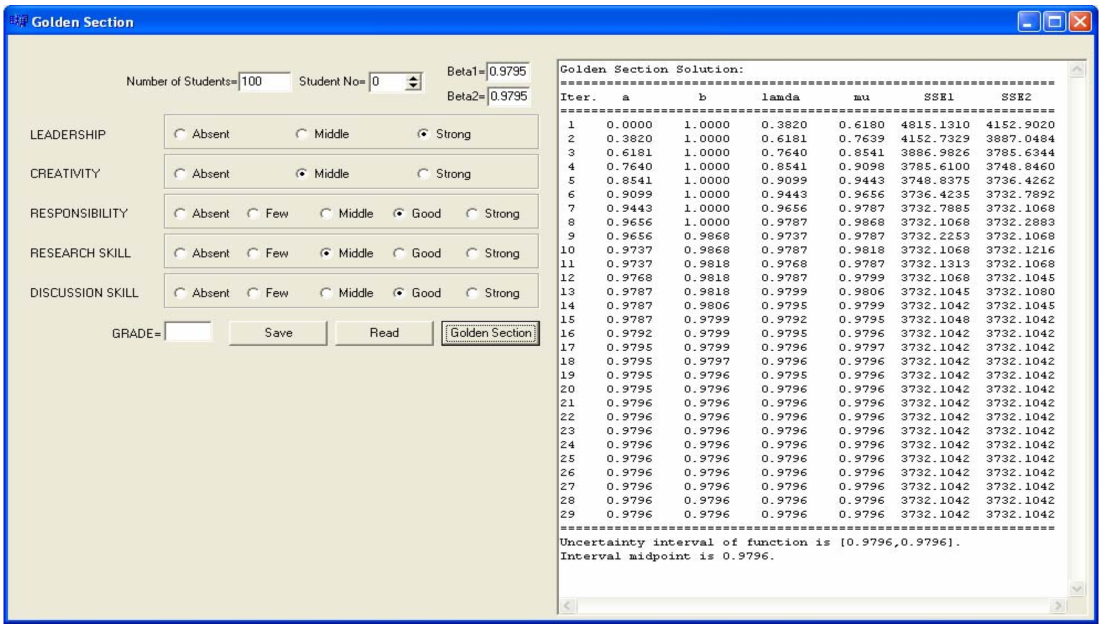


Number of Students= Student No= 0 Beta1= 0.382 Beta2= 0.618

LEADERSHIP	[✓ Absent]		Middle		Strong
CREATIVITY	[✓ Absent]		Middle		Strong
RESPONSIBILITY	[✓ Absent]	Few	Middle	Good	Strong
RESEARCH SKILL	[✓ Absent]	Few	Middle	Good	Strong
DISCUSSION SKILL	[✓ Absent]	Few	Middle	Good	Strong

GRADE=

Figure 5.4 Golden Section window.



Number of Students= 100 Student No= 0 Beta1= 0.9795 Beta2= 0.9795

LEADERSHIP	Absent		Middle		[✓ Strong]
CREATIVITY	Absent		[✓ Middle]		Strong
RESPONSIBILITY	Absent	Few	Middle	[✓ Good]	Strong
RESEARCH SKILL	Absent	Few	[✓ Middle]	Good	Strong
DISCUSSION SKILL	Absent	Few	Middle	[✓ Good]	Strong

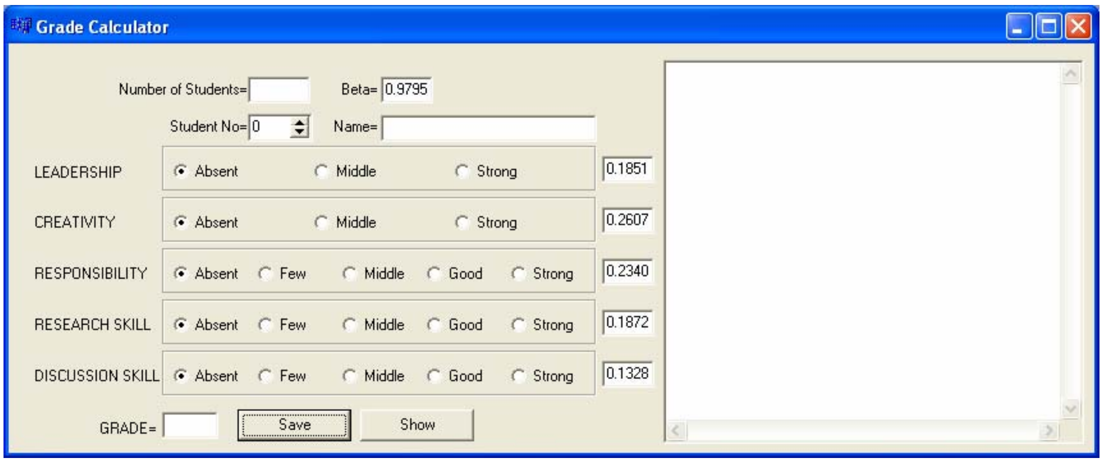
GRADE=

```
Golden Section Solution:
=====================================================================
Iter.   a          b          lamda        mu        SSE1       SSE2
=====================================================================
 1     0.0000     1.0000     0.3820     0.6180   4815.1310  4152.9020
 2     0.3820     1.0000     0.6181     0.7639   4152.7329  3887.0484
 3     0.6181     1.0000     0.7640     0.8541   3886.9826  3785.6344
 4     0.7640     1.0000     0.8541     0.9098   3785.6100  3748.8460
 5     0.8541     1.0000     0.9099     0.9443   3748.8375  3736.4262
 6     0.9099     1.0000     0.9443     0.9656   3736.4235  3732.7892
 7     0.9443     1.0000     0.9656     0.9787   3732.7885  3732.1068
 8     0.9656     1.0000     0.9787     0.9868   3732.1068  3732.2883
 9     0.9656     0.9868     0.9737     0.9787   3732.2253  3732.1068
10     0.9737     0.9868     0.9787     0.9818   3732.1068  3732.1216
11     0.9737     0.9818     0.9768     0.9787   3732.1313  3732.1068
12     0.9768     0.9818     0.9787     0.9799   3732.1068  3732.1045
13     0.9787     0.9818     0.9799     0.9806   3732.1045  3732.1080
14     0.9787     0.9806     0.9795     0.9799   3732.1042  3732.1045
15     0.9787     0.9799     0.9792     0.9795   3732.1048  3732.1042
16     0.9792     0.9799     0.9795     0.9796   3732.1042  3732.1042
17     0.9795     0.9799     0.9796     0.9797   3732.1042  3732.1042
18     0.9795     0.9797     0.9796     0.9796   3732.1042  3732.1042
19     0.9795     0.9796     0.9795     0.9796   3732.1042  3732.1042
20     0.9795     0.9796     0.9796     0.9796   3732.1042  3732.1042
21     0.9796     0.9796     0.9796     0.9796   3732.1042  3732.1042
22     0.9796     0.9796     0.9796     0.9796   3732.1042  3732.1042
23     0.9796     0.9796     0.9796     0.9796   3732.1042  3732.1042
24     0.9796     0.9796     0.9796     0.9796   3732.1042  3732.1042
25     0.9796     0.9796     0.9796     0.9796   3732.1042  3732.1042
26     0.9796     0.9796     0.9796     0.9796   3732.1042  3732.1042
27     0.9796     0.9796     0.9796     0.9796   3732.1042  3732.1042
28     0.9796     0.9796     0.9796     0.9796   3732.1042  3732.1042
29     0.9796     0.9796     0.9796     0.9796   3732.1042  3732.1042
=====================================================================
Uncertainty interval of function is [0.9796,0.9796].
Interval midpoint is 0.9796.
```

Figure 5.5 Image after the information is read from the file and the "Golden Section" button is pushed.

After the parameter calculation procedure by the iterative method, in order to automatically carry out the new student performance evaluations by using the obtained values, it is required to click the "Grade Calculation" button on the Optimal Weights Evaluation form. When this button is clicked, the form in Figure 5.6 appears on the screen. As it is observed, the result of the optimum optimism degree on this screen appears next to the "Beta" field and the weight values determined for each criterion appear at the end of the each linguistic evaluation of its respective criteria. Later, this optimism degree and weight values will be used in the new grade calculations.

Firstly, the number of students, for which grade evaluation will be carried out, is entered into the "Number of Students" field. Then, name and surname is entered into the "Name" field for the student and marking is carried out concerning the linguistic evaluation criteria. By clicking the "Save" button, the student data will be saved and the student will be graded using this entered data which will also be displayed on the screen. Evaluation results can be obtained in the same way for all students until all students are entered. "Show" button can be used at any time to display all the evaluation results as a list as shown in Figure 5.7.


Grade Calculator

Number of Students=

Beta= 0.9795

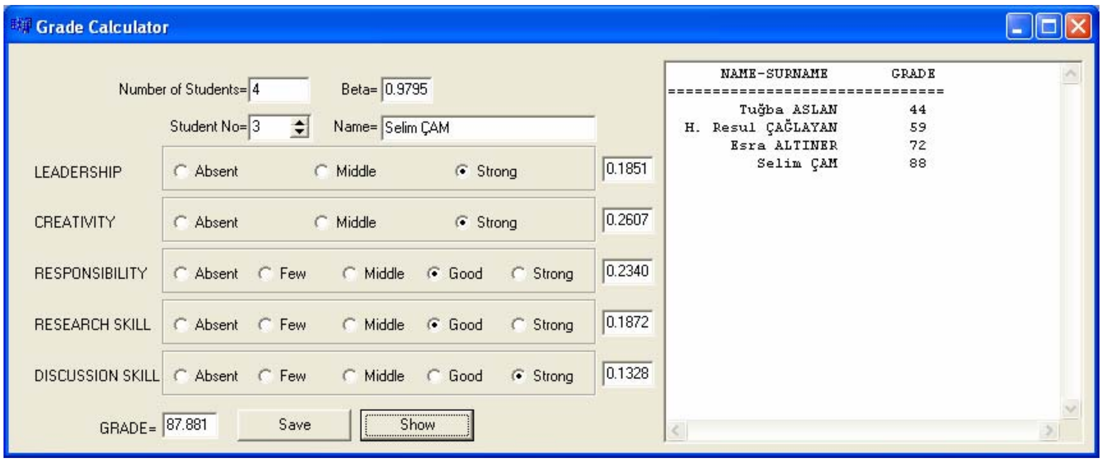
Student No= 0

Name=

LEADERSHIP	[✓ Absent]		Middle		Strong	0.1851
CREATIVITY	[✓ Absent]		Middle		Strong	0.2607
RESPONSIBILITY	[✓ Absent]	Few	Middle	Good	Strong	0.2340
RESEARCH SKILL	[✓ Absent]	Few	Middle	Good	Strong	0.1872
DISCUSSION SKILL	[✓ Absent]	Few	Middle	Good	Strong	0.1328

GRADE=

Figure 5.6 Grade Calculator window.



Number of Students= 4 Beta= 0.9795

Student No= 3 Name= Selim ÇAM

LEADERSHIP	Absent	Middle	[✓ Strong]			0.1851
CREATIVITY	Absent	Middle	[✓ Strong]			0.2607
RESPONSIBILITY	Absent	Few	Middle	[✓ Good]	Strong	0.2340
RESEARCH SKILL	Absent	Few	Middle	[✓ Good]	Strong	0.1872
DISCUSSION SKILL	Absent	Few	Middle	Good	[✓ Strong]	0.1328

GRADE= 87.881

```
        NAME-SURNAME        GRADE
===============================
         Tuğba ASLAN          44
   H. Resul ÇAĞLAYAN          59
        Esra ALTINER          72
           Selim ÇAM          88
```

Figure 5.7 Image after information is saved and "Show" button is pushed.

5.2.1.2 *Functional Modules*

Following are the functional modules assigned to the corresponding buttons on Optimal Weights Evaluation, Golden Section and Grade Calculator forms and their functions:

***Save*:** It saves the results of linguistic performance evaluation for students and the numerical grade values given in return for them in c:\ RRR.txt file.

***Show*:** Writes the defuzzified values of the results of the linguistic performance evaluation by using "Beta" value, determined previously, from the data saved in the flat file and the single numerical grade value given in return for these values on the right screen.

Read: Reads the previously created flat file which contains the students' data and populates the concerned matrices with this data for future calculations.

Optimize: In the light of the available information, it calculates the values of the parameters by using the iterative method.

Golden Section: "Golden Section" button on the Optimal Weights Evaluation form enables transition to the Golden Section program. The "Golden Section" button after transitioning to this program enables the estimation of parameters by using the Golden Section method.

Grade Calculation: Similar to the "Golden Section" button, this button enables transition from Optimal Weights Evaluation window to Grade Calculator window.

Write: Saves all results displayed on the right hand side of the screen into the c:\RRRMemolines.txt file.

Reset: Clears the contest of the flat file c:\RRR.txt.

Result: Saves the obtained optimum optimism degree and weight values into the c:\RRRresult.txt file.

5.2.1.3 Informative Components

Number of Students: Represents the size of the dataset, in other words, the class size.

Student No: Indicates the order of the student whose data have been entered.

Beta: The value of optimism degree used during the defuzzification procedure of the linguistic performance evaluations is entered. Since this value can have values at the interval of $[0,1]$, midpoint value has been entered as default.

Grade: While data are entered separately, the information about the numerical grade value given as a result of the linguistic performance evaluations of each student is written.

Beta1 and ***Beta2:*** Initially, they include 0.382 and 0.618 values respectively. Then, they are changed automatically by using the interval determining formula in Golden Section method.

Name: It enables entering the name of the student whose numerical grade value will be determined in return for linguistic performance evaluations.

5.2.2 Performance Evaluation Results

Our suggested method has been developed as a software application in C++ algorithmic language and 6 data sets have been used in our experiments. 5 lecturers have been asked to assign a numerical grade for 20 students' linguistic evaluations on a set of criteria and these results are presented in Table 5.1.

Table 5.1 Five data sets obtained from 5 lecturers.

		Evaluation Criteria					**Lecturers' Grade Evaluations**				
		L	**C**	**R.S**	**R**	**D.S**	**Lec-1**	**Lec-2**	**Lec-3**	**Lec-4**	**Lec-5**
Student No	**1**	M	M	F	F	M	40	45	40	40	52
	2	M	S	G	G	G	80	75	80	65	80
	3	A	A	F	F	F	30	15	15	15	32
	4	A	M	M	M	F	45	45	35	35	48
	5	S	S	G	G	S	85	90	80	90	92
	6	S	S	S	S	S	100	100	100	100	100
	7	M	A	F	F	M	30	25	30	30	44
	8	A	A	F	M	F	30	25	20	20	36
	9	S	M	G	G	G	75	75	75	75	80
	10	A	M	F	M	M	40	40	35	35	48
	11	M	M	G	G	S	75	70	70	50	76
	12	A	A	A	A	A	0	10	0	0	20
	13	M	M	S	S	S	80	75	85	80	84
	14	M	M	M	M	M	50	50	40	50	60
	15	S	S	M	M	S	80	75	80	80	84
	16	M	S	M	M	G	75	70	70	65	72
	17	A	S	M	G	M	70	65	60	55	64
	18	A	A	A	F	F	20	30	10	10	28
	19	S	S	F	F	S	60	75	65	70	76
	20	S	M	G	M	G	75	70	70	70	76

L=Leadership, C=Creativity, R.S=Research Skill, R=Responsibility, D.S=Discussion Skill.
A=Absent, F=Few, M=Middle, G=Good, S=Strong.

A common optimism degree and a weight value for each criterion are obtained by analyzing the 5 lecturers' relevant data sets. The following methods; WABL, COA and MOM, have been applied to each data set individually and to the aggregate data set, SET-ALL. The details of process are presented in Table 5.2. In this table, the methods which give the minimum least square errors are highlighted in bold. As illustrated in the table, the value of q can be modified and therefore WABL can produce better results than MOM and COA methods.

Table 5.2 The results of WABL, COA and MOM methods for six data sets.

		w_1^*	w_2^*	w_3^*	w_4^*	w_5^*	β^*	SSE
SET-1	**WABL** $(q=0)$	0.0096	0.3058	0.4862	-0.0186	0.2171	0.6957	632.7768
	WABL $(q=10)$	0.0740	0.2735	0.3326	0.1589	0.1610	1.0000	**509.2870**
	WABL $(q=20)$	0.0671	0.2662	0.3024	0.1790	0.1852	1.0000	548.9734
	COA	-0.1367	0.2676	0.4070	-0.0292	0.4912	-	898.5181
	MOM	0.0083	0.2375	0.2607	0.1736	0.3199	-	619.8824
SET-2	**WABL** $(q=0)$	0.1615	0.3684	0.1995	0.1079	0.1627	0.6395	545.0104
	WABL $(q=10)$	0.2065	0.3086	0.0799	0.3064	0.0987	0.9536	**326.3041**
	WABL $(q=20)$	0.2007	0.3001	0.0604	0.3242	0.1146	1.0000	334.6471
	COA	0.0432	0.3430	0.1474	0.0736	0.3928	-	788.8690
	MOM	0.1541	0.2847	-0.0325	0.3598	0.2341	-	450.6970
SET-3	**WABL** $(q=0)$	-0.0084	0.2797	0.6198	-0.2660	0.3748	0.5020	742.3373
	WABL $(q=10)$	0.0899	0.2622	0.5105	-0.0913	0.2288	0.5975	374.5684
	WABL $(q=20)$	0.0925	0.2588	0.4994	-0.0764	0.2257	0.6077	351.8783
	WABL $(q \geq 160)$	0.0936	0.2549	0.4859	-0.0609	0.2265	0.6050	$\leq$ **331.0163**
	COA	-0.0521	0.2788	0.6524	-0.3088	0.4297	-	840.3809
	MOM	0.0884	0.2516	0.4725	-0.0566	0.2441	-	331.0723
SET-4	**WABL** $(q=0)$	0.2959	0.2992	0.2798	0.0136	0.1114	0.5230	948.9585
	WABL $(q=10)$	0.3388	0.2714	0.1660	0.2257	-0.0018	0.6473	402.4287
	WABL $(q=20)$	0.3374	0.2668	0.1572	0.2415	-0.0030	0.6817	370.6944
	WABL $(q \geq 35)$	0.3362	0.2644	0.1525	0.2490	-0.0021	0.7014	$\leq$ **356.3858**
	COA	0.2478	0.2929	0.3116	-0.0525	0.2002	-	1155.5135
	MOM	0.3193	0.2595	0.1196	0.2654	0.0362	-	357.4269
SET-5	**WABL** $(q=0)$	0.2265	0.2313	0.1783	0.1768	0.1871	0.8922	**32.4472**
	WABL $(q=10)$	0.1514	0.1714	-0.0867	0.3935	0.3704	1.0000	534.5341
	WABL $(q=20)$	0.1401	0.1636	-0.1117	0.4118	0.3963	1.0000	672.0539
	COA	-0.0268	0.1455	-0.0329	0.1909	0.7233	-	1038.6621
	MOM	0.0600	0.1345	-0.1061	0.3507	0.5610	-	1055.4155
SET-ALL	**WABL** $(q=0)$	0.1372	0.2974	0.3530	0.0024	0.2099	0.6512	4959.6541
	WABL $(q=10)$	0.1852	0.2607	0.2341	0.1872	0.1328	0.9796	**3732.1042**
	WABL $(q=20)$	0.1776	0.2538	0.2078	0.2072	0.1536	1.0000	3790.1983
	COA	0.0151	0.2656	0.2971	-0.0252	0.4474	-	6145.2656
	MOM	0.1260	0.2336	0.1428	0.2185	0.2791	-	4332.4927

Optimum solutions by using both our iterative method and the well-known Golden Section method are obtained for different certainty levels for comparing the effectiveness of two methods. Comparative table is shown below (Table 5.3).

Table 5.3 Number of iterations of the suggested iterative $(q=0)$ and Golden Section methods.

Certainty level	Method	Iterations					
		SET1	SET2	SET3	SET4	SET5	SET-ALL
$\varepsilon = 0.001$	Suggested Iterative	9	9	2	6	9	9
	Golden Section	15	15	15	15	15	15
$\varepsilon = 0.0001$	Suggested Iterative	12	12	6	10	11	12
	Golden Section	20	20	20	20	20	20
$\varepsilon = 0.00001$	Suggested Iterative	15	15	10	14	14	15
	Golden Section	24	24	24	24	24	24
$\varepsilon = 0.000001$	Suggested Iterative	18	18	14	17	17	18
	Golden Section	29	29	29	29	29	29

The comparison of our iterative method and Golden Section method is performed with sign test instead of Wilcoxon sign rank test because of the asymmetrical distribution of data. As shown in Table 5.3, when the results are investigated separately for each certainty level, the number of iterations in Golden Section method remains constant. On the other hand, the number of iterations changes in our iterative method. Although, the number of iterations increases as certainty level decrease for both methods, our iterative method needs less iteration in comparison to Golden Section. When significance of the difference between the numbers of iterations in both methods is investigated by using the sign test, the result is significant with $p = 0.0313$ at $\alpha = 0.05$ level. As a result, it can be said that our proposed method can produce more optimal solutions in less number of iterations when compared to Golden Section method.

Table 5.4 The results of iterative method for SET-ALL data set with $q = 10$.

Iteration numbers	β	$w_j, j=\overline{1,5}$					SSE
1	0.5000	0.11917	0.23974	0.15960	0.18827	0.29322	4451.0432
2	0.7369	0.15512	0.25160	0.18947	0.19484	0.20898	3927.0836
3	0.8637	0.17175	0.25678	0.21105	0.19263	0.16779	3777.8210
4	0.9260	0.17918	0.25900	0.22304	0.19012	0.14867	3742.0216
5	0.9552	0.18248	0.25997	0.22896	0.18862	0.13996	3734.1757
6	0.9685	0.18396	0.26039	0.23174	0.18787	0.13603	3732.5293
7	0.9746	0.18463	0.26058	0.23302	0.18752	0.13426	3732.1907
8	0.9773	0.18492	0.26067	0.23359	0.18736	0.13346	3732.1217
9	0.9786	0.18506	0.26070	0.23386	0.18728	0.13310	3732.1077
10	0.9791	0.18512	0.26072	0.23397	0.18725	0.13294	3732.1049
11	0.9794	0.18514	0.26073	0.23403	0.18723	0.13287	3732.1043
12	0.9795	0.18516	0.26073	0.23405	0.18723	0.13283	3732.1042
13	0.9796	0.18516	0.26073	0.23406	0.18723	0.13282	3732.1042

Demonstration of proposed iterative method applied on SET-ALL dataset for $\varepsilon = 0.0001$ can be seen in Table 5.4. Starting from $\beta = 0.5$, the suggested iterative procedure obtains the β^* and $w_j^*,\ j=\overline{1,5}$ values as a target-driven process.

5.3 Group Constitution Tools and Experimental Results

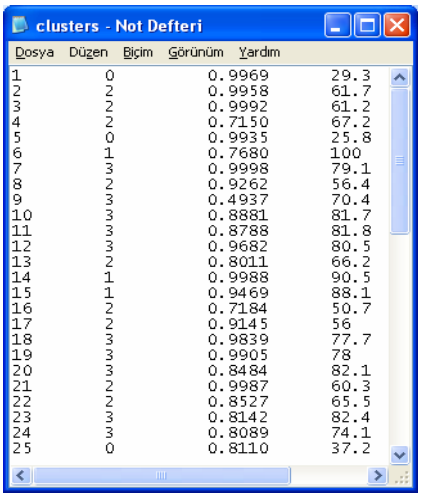
5.3.1 Group Constitution Tools

In this section, detailed information is given concerning the forms, functional modules, and informative components constituted in Group Constitution program.

5.3.1.1 Forms

So as to constitute student groups under different strategies, firstly, all students have to be divided into clusters according to their success status. In this section of the work, the program entitled Visual Clustering 2.0, constituted in the work of Ulutagay (2004) and Nasibov and Ulutagay (2006a), is used in order to determine which student belongs to which group. Therefore, the completion of clustering procedure will be the first step towards constituting the student groups.

After the completion of the procedure of clustering students according to their success status, this information is transferred to a file entitled c:\\clusters.txt as in Figure 5.8. The information taking place on the columns of the files is the student order number, information about which student belongs to which cluster, the degree of belongingness of the student to the cluster and finally information about the grade value the student receives from that module respectively.



```
1         0          0.9969       29.3
2         2          0.9958       61.7
3         2          0.9992       61.2
4         2          0.7150       67.2
5         0          0.9935       25.8
6         1          0.7680       100
7         3          0.9998       79.1
8         2          0.9262       56.4
9         3          0.4937       70.4
10        3          0.8881       81.7
11        3          0.8788       81.8
12        3          0.9682       80.5
13        2          0.8011       66.2
14        1          0.9988       90.5
15        1          0.9469       88.1
16        2          0.7184       50.7
17        2          0.9145       56
18        3          0.9839       77.7
19        3          0.9905       78
20        3          0.8484       82.1
21        2          0.9987       60.3
22        2          0.8527       65.5
23        3          0.8142       82.4
24        3          0.8089       74.1
25        0          0.8110       37.2
```

Figure 5.8 Image of file where encoded student information exists after the clustering procedure.

The next step is the running of Group Constitution program. When the program is run, the first thing that appears on the screen is the window in Figure 5.9

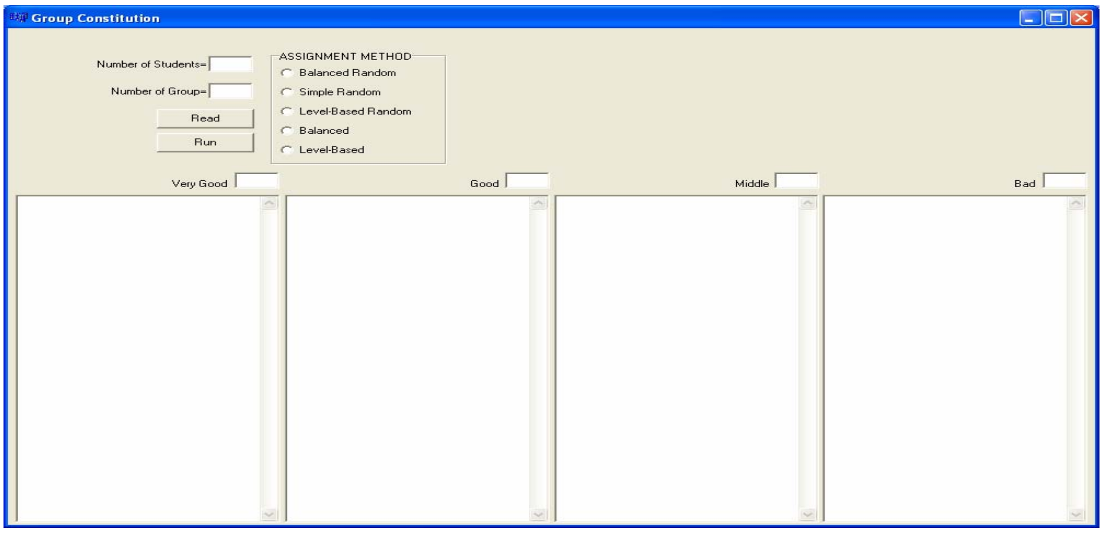
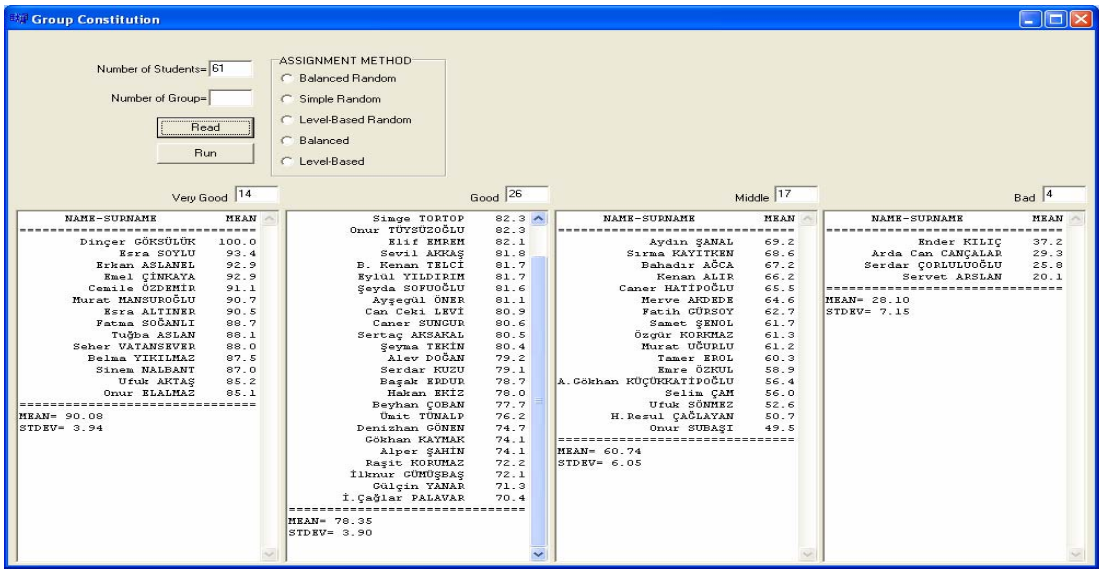


Figure 5.9 Opening window of the Group Constitution program.

Since the data entry in the program is carried out only through reading from the file, it is required first to push "Read" button. The program image after pushing the button is as in Figure 5.10. In conclusion, we can see from "Number of Students" part that there are 61 students in the file. Furthermore, it lists the students, who have been divided into 4 clusters according to the clustering procedure, in descending order in the form of "Very Good", "Good", "Middle" and "Bad" for each cluster according to their average success. Information about how many students there are in each group is given next to each cluster (Figure 5.10).



Number of Students= 61

Number of Group=

Very Good 14

```
     NAME-SURNAME          MEAN
================================
     Dinçer GÖKSÜLÜK      100.0
          Esra SOYLU       93.4
       Erkan ASLANEL       92.9
        Emel ÇİNKAYA       92.9
      Cemile ÖZDEMİR       91.1
   Murat MANSUROĞLU        90.7
        Esra ALTINER       90.5
      Fatma SOĞANLI        88.7
         Tuğba ASLAN       88.1
    Seher VATANSEVER       88.0
      Belma YIKILMAZ       87.5
       Sinem NALBANT       87.0
          Ufuk AKTAŞ       85.2
        Onur ELALMAZ       85.1
================================
MEAN= 90.08
STDEV= 3.94
```

Good 26

```
       Simge TORTOP        82.3
   Onur TÜYSÜZOĞLU         82.3
         Elif EMREM        82.1
        Sevil AKKAŞ        81.8
     B. Kenan TELCİ        81.7
    Eylül YILDIRIM         81.7
    Şeyda SOFUOĞLU         81.6
      Ayşegül ÖNER         81.1
     Can Ceki LEVİ         80.9
      Caner SUNGUR         80.6
    Sertaç AKSAKAL         80.5
       Şeyma TEKİN         80.4
        Alev DOĞAN         79.2
       Serdar KUZU         79.1
      Başak ERDUR          78.7
        Hakan EKİZ         78.0
      Beyhan ÇOBAN         77.7
       Ümit TÜNALP         76.2
    Denizhan GÖNEN         74.7
     Gökhan KAYMAK         74.1
       Alper ŞAHİN         74.1
     Raşit KORUMAZ         72.2
  İlknur GÜMÜŞBAŞ          72.1
      Gülçin YANAR         71.3
  İ.Çağlar PALAVAR         70.4
================================
MEAN= 78.35
STDEV= 3.90
```

Middle 17

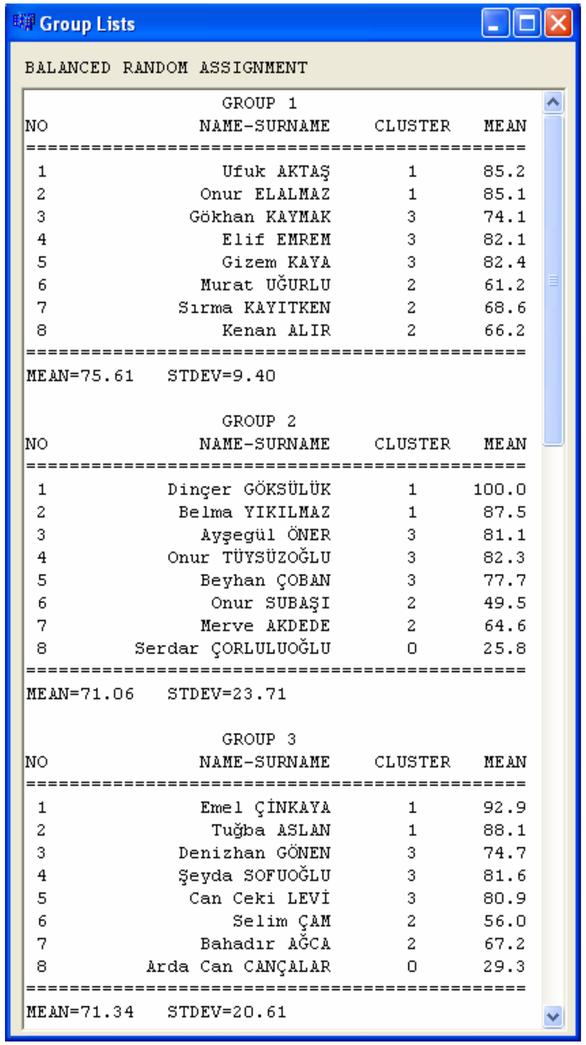
```
     NAME-SURNAME          MEAN
================================
          Aydın ŞANAL      69.2
       Sırma KAYITKEN      68.6
         Bahadır AĞCA      67.2
           Kenan ALIR      66.2
      Caner HATİPOĞLU      65.5
         Merve AKDEDE      64.6
        Fatih GÜRSOY       62.7
         Samet ŞENOL       61.7
      Özgür KORKMAZ        61.3
        Murat UĞURLU       61.2
          Tamer EROL       60.3
          Emre ÖZKUL       58.9
A.Gökhan KÜÇÜKKATİPOĞLU    56.4
           Selim ÇAM       56.0
        Ufuk SÖNMEZ        52.6
   H.Resul ÇAĞLAYAN        50.7
        Onur SUBAŞI        49.5
================================
MEAN= 60.74
STDEV= 6.05
```

Bad 4

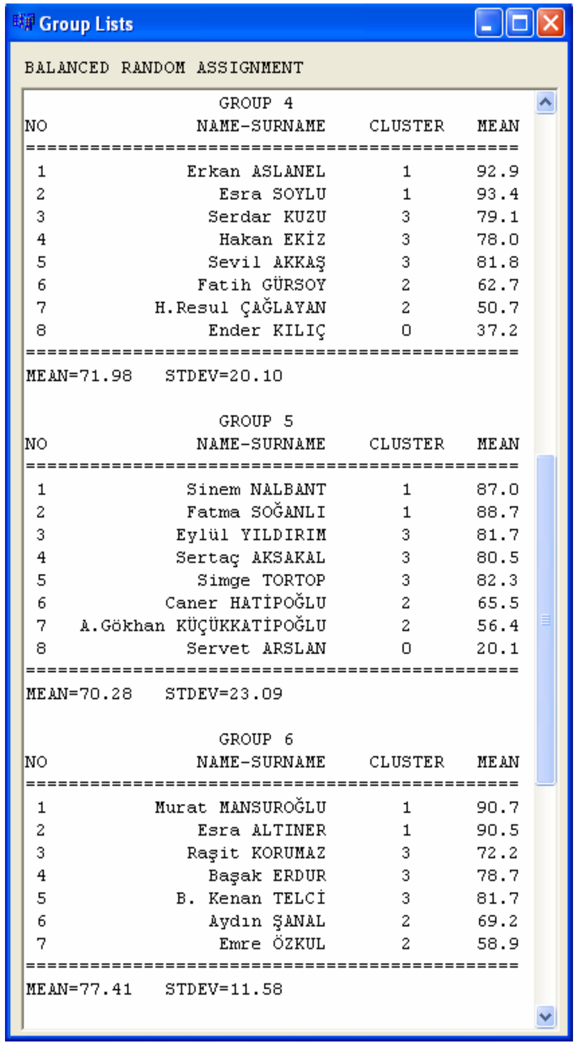
```
     NAME-SURNAME          MEAN
================================
         Ender KILIÇ       37.2
   Arda Can CANÇALAR       29.3
 Serdar ÇORLULUOĞLU        25.8
       Servet ARSLAN       20.1
================================
MEAN= 28.10
STDEV= 7.15
```

Figure 5.10 Image after the information is read from the file by "Read" button.

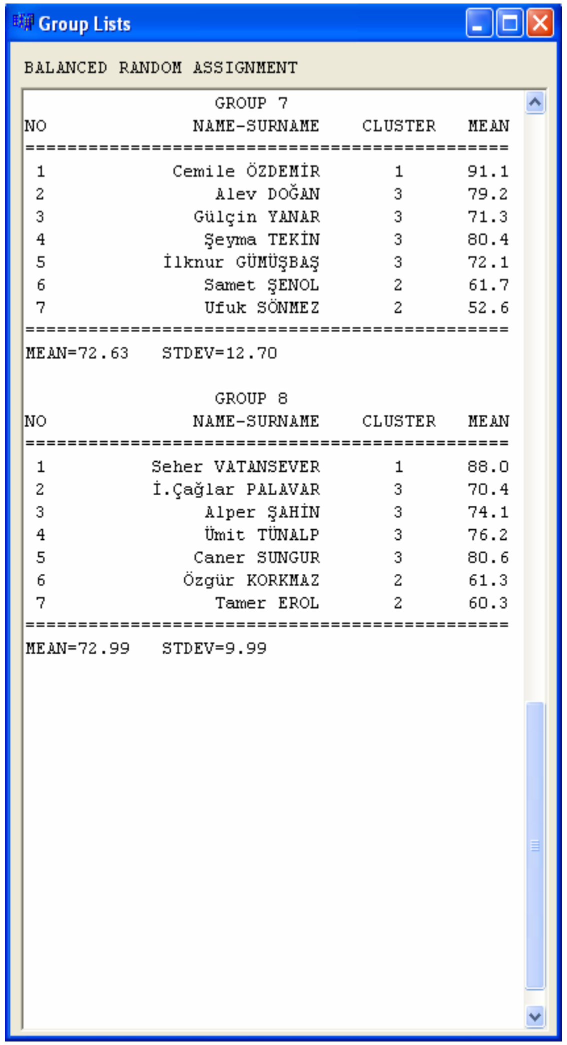
After the information about the students is read, student groups can now be constituted in the desired number and by using the desired strategy. Firstly, the number of groups to be constituted on “Number of Group” part must be determined. Then, group constitution strategy, which is appropriate for the objective, is determined from “Assignment Method” part, which performs one of the appropriate algorithm of Algorithms 4.1-4.5. As it is also mentioned above, the number of students in each student group will be approximately equal. For instance, when we want to constitute 8 student groups by the Balanced Random method, the resulting table would be as in Figure 5.11. Students are here selected randomly from each cluster according to the “Balanced Random” assignment principle and assigned to groups on the condition that they are approximate equal in number. Therefore, no matter what the number of groups constituted in this assignment method is, approximately identical number of students will take place in each group from each cluster.



```
BALANCED RANDOM ASSIGNMENT

                     GROUP 1
NO                NAME-SURNAME    CLUSTER    MEAN
==============================================
 1                  Ufuk AKTAŞ       1       85.2
 2                 Onur ELALMAZ      1       85.1
 3                Gökhan KAYMAK      3       74.1
 4                  Elif EMREM       3       82.1
 5                  Gizem KAYA       3       82.4
 6                Murat UĞURLU       2       61.2
 7               Sırma KAYITKEN      2       68.6
 8                  Kenan ALIR       2       66.2
==============================================
MEAN=75.61   STDEV=9.40

                     GROUP 2
NO                NAME-SURNAME    CLUSTER    MEAN
==============================================
 1              Dinçer GÖKSÜLÜK      1      100.0
 2               Belma YIKILMAZ      1       87.5
 3                Ayşegül ÖNER       3       81.1
 4             Onur TÜYSÜZOĞLU       3       82.3
 5                Beyhan ÇOBAN       3       77.7
 6                 Onur SUBAŞI       2       49.5
 7                Merve AKDEDE       2       64.6
 8          Serdar ÇORLULUOĞLU       0       25.8
==============================================
MEAN=71.06   STDEV=23.71

                     GROUP 3
NO                NAME-SURNAME    CLUSTER    MEAN
==============================================
 1                Emel ÇİNKAYA       1       92.9
 2                 Tuğba ASLAN       1       88.1
 3              Denizhan GÖNEN       3       74.7
 4              Şeyda SOFUOĞLU       3       81.6
 5               Can Ceki LEVİ       3       80.9
 6                  Selim ÇAM        2       56.0
 7                Bahadır AĞCA       2       67.2
 8           Arda Can CANÇALAR       0       29.3
==============================================
MEAN=71.34   STDEV=20.61
```



```
BALANCED RANDOM ASSIGNMENT

                     GROUP 4
NO                NAME-SURNAME    CLUSTER    MEAN
==============================================
 1               Erkan ASLANEL       1       92.9
 2                  Esra SOYLU       1       93.4
 3                 Serdar KUZU       3       79.1
 4                 Hakan EKİZ        3       78.0
 5                Sevil AKKAŞ        3       81.8
 6                Fatih GÜRSOY       2       62.7
 7            H.Resul ÇAĞLAYAN       2       50.7
 8                Ender KILIÇ        0       37.2
==============================================
MEAN=71.98   STDEV=20.10

                     GROUP 5
NO                NAME-SURNAME    CLUSTER    MEAN
==============================================
 1               Sinem NALBANT       1       87.0
 2              Fatma SOĞANLI        1       88.7
 3             Eylül YILDIRIM        3       81.7
 4              Sertaç AKSAKAL       3       80.5
 5                Simge TORTOP       3       82.3
 6             Caner HATİPOĞLU       2       65.5
 7     A.Gökhan KÜÇÜKKATİPOĞLU       2       56.4
 8                Servet ARSLAN      0       20.1
==============================================
MEAN=70.28   STDEV=23.09

                     GROUP 6
NO                NAME-SURNAME    CLUSTER    MEAN
==============================================
 1           Murat MANSUROĞLU        1       90.7
 2               Esra ALTINER        1       90.5
 3              Raşit KORUMAZ        3       72.2
 4               Başak ERDUR         3       78.7
 5            B. Kenan TELCİ         3       81.7
 6                Aydın ŞANAL        2       69.2
 7                 Emre ÖZKUL        2       58.9
==============================================
MEAN=77.41   STDEV=11.58
```



```
BALANCED RANDOM ASSIGNMENT

                     GROUP 7
NO                NAME-SURNAME    CLUSTER    MEAN
==============================================
 1               Cemile ÖZDEMİR      1       91.1
 2                   Alev DOĞAN      3       79.2
 3                Gülçin YANAR       3       71.3
 4                 Şeyma TEKİN       3       80.4
 5             İlknur GÜMÜŞBAŞ       3       72.1
 6                Samet ŞENOL        2       61.7
 7                Ufuk SÖNMEZ        2       52.6
==============================================
MEAN=72.63   STDEV=12.70

                     GROUP 8
NO                NAME-SURNAME    CLUSTER    MEAN
==============================================
 1             Seher VATANSEVER      1       88.0
 2            İ.Çağlar PALAVAR       3       70.4
 3                Alper ŞAHİN        3       74.1
 4                 Ümit TÜNALP       3       76.2
 5                Caner SUNGUR       3       80.6
 6              Özgür KORKMAZ        2       61.3
 7                  Tamer EROL       2       60.3
==============================================
MEAN=72.99   STDEV=9.99
```

Figure 5.11 Result of assignment after the number of groups and the assignment method are determined.

5.3.1.2 Functional Modules

Followings are the functional modules assigned to the corresponding buttons in the Group Constitution program and their functions:

Read: Reads the previously created flat file c:\clusters.txt, which contains the students' data and populates to the concerned matrices with this data for future calculations.

Run: It carries out the assignment procedure of students to groups according to the selected assignment method.

5.3.1.3 Informative Components

Number of Students: Represents the size of the dataset, in other words, the class size.

Number of Group: How many groups the students will be divided into is determined on this information box.

5.3.2 Group Constitution Results

In order to be used in balanced assignment method, 1000 data have been created from each $N\left(50,\ 16^2\right)$, $N\left(70,\ 10^2\right)$ and $\text{Gamma}(14, 3)$ distribution with a view to reflecting grade values between 0-100 and samples have been constituted from these populations in different sample sizes by selecting randomly without replacement of data. Sample sizes have been determined as 30, 50, 80 and 100 and for each sample size, twenty data sets have been selected from the population. Then the group averages have been obtained by carrying out different numbers of group assignments from each data set constituted. Summary tables have been constituted for the sample sizes 30, 50, 80 and 100 for each distribution, which have been constituted according to different group numbers (Appendix A.1–A.12). In these tables, $\min \overline{p}^i$, $\max \overline{p}^j$, $(\max p_i - \min p_j)$ and $\dfrac{\max\limits_{i,j}\left|\overline{p}^i - \overline{p}^j\right|}{\max\limits_{i,j}\left|p_i - p_j\right|}$ values take place. So as to compare the validity of these obtained results, the comparison approach in Equation (4.7) has been used. The results obtained as a result of this approach are as in Table 5.5.

The reason why different group numbers are used for each sample size in Appendix A.1–A.12 is the desire to constitute student groups in significant sizes since, as it also takes place in the title of the work, student groups are arranged in 7 to 9 people in the problem based learning sessions for the problem based learning system applied in the Department of Statistics at Dokuz Eylül University. Besides this, trials have been carried out also for 5-, 6- or 10-people group constitutions so as to be an example.

Table 5.5 Summary table of error ratio averages.

$n=30$	$m=3$	$m=4$	$m=5$	$m=6$
$N(50,16^2)$	0.0174	0.0948	0.0328	0.0825
$N(70,10^2)$	0.0188	0.0958	0.0337	0.0811
Gamma(14,3)	0.0234	0.0969	0.0419	0.0897
$n=50$	$m=5$	$m=6$	$m=7$	$m=8$
$N(50,16^2)$	0.0167	0.0599	0.1009	0.0796
$N(70,10^2)$	0.0200	0.0565	**0.1099**	0.0732
Gamma(14,3)	0.0227	0.0431	0.1031	0.0573
$n=80$	$m=8$	$m=9$	$m=10$	$m=11$
$N(50,16^2)$	0.0145	0.0605	0.0205	0.1006
$N(70,10^2)$	0.0193	0.0589	0.0213	0.1054
Gamma(14,3)	0.0264	0.0544	0.0301	0.1042
$n=100$	$m=10$	$m=11$	$m=12$	$m=13$
$N(50,16^2)$	0.0172	0.0840	0.0589	0.0905
$N(70,10^2)$	0.0156	0.0881	0.0582	0.0937
Gamma(14,3)	0.0245	0.0858	0.0448	0.0943

As also observed in Table 5.5, the means of the error ratio values are calculated in this way for each sample size and each group number. When this table is observed, the fact that the highest value among the means of error ratio is 0.1099 shows that there is %11 difference among group means. Actually, we also observe that this ratio has quite lower values than %11. Nevertheless, if an optimum solution had been obtained instead of the heuristic solution, these results wouldn't have had "0" value again since it is the discrete optimization problem. Therefore, this algorithm in fact gives closer results to the optimal solution.

CHAPTER SIX
CONCLUSIONS

Fundamentally two mathematical systems are suggested in this work. The first one is the student performance evaluation system while the second one is the student group constitution system in parallel to different strategies. Moreover, in Borland C++ Builder 6.0 program, the algorithms of both systems have been constituted. The results of the work have been published as Nasibov and Kınay (2007a, 2007b, 2007c, 2007d) articles.

Evaluation of student performances, in other words, their gradations can be carried out by different lecturers, hence as a result of each lecturer having different opinions the same student may recieve different grades from different lecturers. Furthermore, these evaluations were carried out in a way that only the concerned lecturer gave a single numerical value to the student on his observations. Nevertheless, the student performance evaluation system was suggested as a result of the necessity of carrying out such evaluations through linguistic terms, which are more inclined to the human way of thinking, and later obtaining the numerical results of these evaluations. Hence, a mathematical model is suggested so as to eliminate the subjective cases as much as possible occurring during evaluation or, in other words, to create a common and unbiased evaluation process by joining the opinions of all lecturers and to estimate parameters. The solution of this model is provided by an iterative algorithm. Thanks to this model, the performance evaluations calculated against linguistic evaluations by using estimated parameters reflecting the evaluation strategy of all of the lecturers, whose knowledge and experience will be taken into consideration, will be the result of a common decision system and will reflect not only one but all decision-makers' opinion.

Compared to other defuzzification methods, the superiority of the WABL method, used in calculations of the iterative method as a basis for the defuzzification of fuzzy data, or in other words, the linguistic variables, has been observed by numerical results. Furthermore, it has been tested and shown that the suggested iterative method reaches the optimum parameter estimation in fewer steps when compared to the Golden Section method as the representative of classical optimization methods.

The main purpose that constitutes the title of this work is to form student groups in parallel to different strategies. The most important information used in forming these student groups is the student performance evaluation results. Therefore, by using the performance evaluation system suggested in this work, the constitution of student groups automatically in parallel to different strategies will give better results within a shorter period of time.

In conclusion, five heuristic assignment methods have been suggested concerning the creation of student groups in parallel to different strategies. Moreover, the mathematical model and algorithms for four of these methods have been provided. It has been observed that the results of one of the suggested assignment methods, balanced assignment method, gave quite effective results like the integer linear programming method and further provide these results within a very short period of time. Due to the identification of the problem, while the division of large groups into smaller groups sometimes cannot be carried out practically by ready integer linear programming software, this does not become a problem thanks to the suggested heuristic assignment method.

Briefly, in this work,

1. For student performance evaluation, the evaluation criteria have been determined and the evaluation form has been created by taking the opinions of the lecturers.
2. For student performance evaluation, a fuzzy-logic-based approach and an iterative algorithm have been suggested.
3. This method provides a common and unbiased evaluation mechanism for all students subjected to evaluation.
4. So as to find the optimal solution, formulae have been given and optimality has been proven.
5. Various defuzzification methods such as WABL (Weighted Averages Based on Levels), COA (Center of Area) and MOM (Mean of Maxima)

have been used in the defuzzification of fuzzy linguistic values and comparative analyses have been carried out. The result of the comparative analyses has shown that the WABL method used is more effective than other defuzzification methods.

6. It has been proven as a result of experiments that the iterative solution algorithm is much more effective when compared to the Golden Section algorithm which is a representative of the classical optimization algorithms.
7. The mathematical formulation of group constitution problems based on various strategies such as balanced random and level-based has been given and the heuristic solution algorithms have been suggested.
8. The programs for the student performance evaluation and group constitution with various strategies algorithms have been implemented in the Borland C++ Builder SDK and calculations have been made. The result of the calculations has showed that the methods used are effective.

For further studies, the improvements to the student groups formation process according to specific characteristics of students, for instance their approximate equal distribution according to their gender, their reevaluation according to the harmony among students, in other words, preventing students, who cannot get on well with each other, to get into the same group, can be studied. In addition, as a further work, a system can be developed in which the course success status of students are grouped according to their courses and which will enable them to be directed for the choice of profession close to the fields where they are successful.

REFERENCES

Babuška, R. (2001). *Fuzzy and neural control-DISC course lecture notes*, Netherlands.

Bardossy, A., Duckstein, L., & Bogardi, L. (1993). Combination of fuzzy numbers representing expert opinions, *Fuzzy Sets and Systems*, 57, 173-181.

Bezdek, J.C. (1974). *Fuzzy mathematics in pattern classification*, Ph.D. Thesis, Applied Mathematics Center, Cornell University, Ithaca.

Bezdek, J.C. (1975). *Mathematical models for systematics and taxonomy*, In proc. 8th Int. Conf. On Numerical Taxonomy, San Fransisco, 143-166.

Bezdek, J.C. (1981). *Pattern recognition with fuzzy objective function algorithms*, Plenum Press, New York.

Bobrowski, L., & Bezdek, J.C. (1991). *C*-means clustering with I_1 and I_∞ norms, *IEEE Transactions on Systems, Man, and Cybernetics*, 21, 545-554.

Bonder, C.G.E., Graan, J.G., & Lootsma, F.A. (1989). Multicriteria decision analysis with fuzzy pairwise comparisons, *Fuzzy Sets and Systems*, 29, 133-143.

Caron, G., Hansen, P., & Jaumard, B. (1999). The assignment problem with seniority and job priority constraints, *Operations Research*, 47(3), 449-454.

Cattrysse, D.G., & Van Wassenhove, L.N. (1992). A survey of algorithms for the generalized assignment problem, *European Journal of Operational Research*, 60(3), 260-272.

Chen, C-T. (2000). Extensions of the TOPSIS for group decision-making under fuzzy environment, *Fuzzy Sets and Systems*, 114, 1-9.

Daskalai, S., Birbas, T., & Housos, E. (2004). An integer programming for a case study in university timetabling, *European Journal of Operational Research*, 153(1), 117-135.

Dell'Amico, M., & Martello, S. (1997). The *k*-cardinality assignment problem, *Discrete Applied Mathematics*, 76(1-3), 103-121.

Duin, C.W., & Volgenant, A. (1991). Minimum deviation and balanced optimization: A unified approach, *Operations Research Letters*, 10(1), 43-48.

Dunn, J.C. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters, *Journal of Cybernetics*, 3(3), 32-57.

Dunn, J.C. (1974). Well-separated clusters and optimal fuzzy partitions, *Journal of Cybernetics*, 4, 95-104.

Ford Jr., L.R., & Fulkerson, D.R. (1966). *Flows in Networks*, Princeton University Press, Princeton NJ.

Fukuyamo, Y., & Sugeno, M. (1989). *A new method for choosing the number of clusters for the fuzzy c-means method*, In proc. 5th Fuzzy System Symp., 247-250.

Gordon, A.D. (1981). *Classification*, University Press, Cambridge.

Gross, O. (1959). *The bottleneck assignment problem*, The RAND Symposium on Mathematical Programming (Linear Programming and Extension), The Rand Corporation, Paper P-1630, March 16-20.

Gupta, S.K., & Punnen, A.P. (1988). Minimum deviation problems, *Operations Research Letters*, 7(4), 201-204.

Han, J., & Kamber, M. (2001). *Data mining concepts and techniques*, Morgan Kaufmann Publishers.

Hathaway, R.J., & Bezdek, J.C. (1993). Switching regression models and fuzzy clustering, *IEEE Transactions on Fuzzy Systems*, 1, 195-204.

Hsu, H.M., & Chen, C.T. (1996). Aggregation of fuzzy opinions under group decision making, *Fuzzy Sets and Systems*, 79, 279-285.

Kahraman, C., Ruan, D., & Doğan, I. (2003). Fuzzy group decision-making for facility location selection, *Information Sciences*, 157; 135-153.

Klir, G.J., & Folger, T.A. (1988). *Fuzzy sets, uncertainty, and information*, Prentice & Hall.

Kuhn, H.W. (1955). The Hungarian method for the assignment problem, *Naval Research Logistics Quarterly,* 2(1&2), 83-97.

Kuo, M-S., Tzeng G-H., & Huang, W-C. (2007). Group decision-making based on concepts of ideal and anti-ideal points in fuzzy environment*, Mathematical and Computer Modeling*, 45, 324-339.

Kwon, S.H. (1998). Cluster validity index for fuzzy clustering, *Electronic Letters*, 34(22), 2176-2177.

Lee, H.S. (2002). Optimal consensus of fuzzy opinions under group decision making environment, *Fuzzy Sets and Systems*, 132, 303-315.

Lin, C-T., & Lee, C.S. G. (1996). *Neural fuzzy systems*, Prentice & Hall.

Ma, J., & Zhou, D. (2000). Fuzzy set approach to the assessment of student-centered learning, *IEEE Transactions on Education*, 43(2), 237-241.

Martello, J., Pulleybank, W.R., Toth, P., & de Werra, D. (1984). Balanced optimization problems, *Operations Research Letters*, 3(5), 275-278.

Mendel, J.M., (2001). *Uncertain rule-based fuzzy logic systems*, Prentice & Hall.

Nasibov, E.N. (2002). Certain integral characteristics of fuzzy numbers and a visual interactive method for choosing the strategy of their calculation, *Journal of Computer and System Sciences International*, 41(4), 584-590.

Nasibov, E.N. (2003a). Aggregation of fuzzy values in linear programming problems, *Automatic Control and Computer Sciences*, 37(2),1-11.

Nasibov, E.N., & Shikhlinskaya, R. (2003b). Adjustment of the parameters of WABL-aggregation for locating the center of gravity of polynomial-type fuzzy number. *Automatic Control and Computer Sciences*, 37(6), 34-42.

Nasibov E.N., Nasibova R.A. (2003c). OWA and MIN Aggregation methods in fuzzy bin-packing problem. *Transac. of the National Academy of Sciences of Azerbaijan*, Phus.-tech. and math. series, 2, 45-50.

Nasibov E.N. (2004). An Algorithm for Constructing an Admissible Solution to the Bin Packing Problem with Fuzzy Constraints, *Journ. of Comp. and Syst. Sci. Int.*, 43(2), 205-212.

Nasibov, E.N., & Mert, A. (2005). *On WABL and COA defuzzifications for polynomial shape triangular and trapezoidal fuzzy numbers*, In Proc. ICSCCW-2005, Antalya, Turkey, 149-158.

Nasibov, E.N., & Ulutagay, G. (2006a). *Program tools for fuzzy clustering analysis*, In Proc. Inf. Tech. and Telecomm. in Education and Science, May 20-26, Antalya, Turkey, 195-197.

Nasibov, E.N., & Ulutagay, G. (2006b). *A new fuzzy joint points criteria for cluster validity,* In Proc. Int. Conf. On Modeling and Simulation (AMSE), Konya, Turkey, 625-629.

Nasibov, E.N., & Kınay, A.Ö. (2006c). *Kaliteli İş Paylaşımı Problemi için Bulanık Mantık Yaklaşımı*, In Proc. Yöneylem Araştırması ve Endüstri Mühendisliği 26. Ulusal Kongresi, July 3-5, Kocaeli, Turkey, 360-363.

Nasibov, E.N., & Kınay, A.Ö. (2006d). Kaliteli İş Paylaşımı Problemi için Bulanık Mantık Yaklaşımı, *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi*, 5(10), 13-22.

Nasibov, E.N., & Kınay, A.Ö. (2007a). *Aktif Eğitimde Öğrenci Performans Değerlendirme Stratejisinin Bulanık Regresyon Yaklaşımıyla Belirlenmesi*, In Proc. 5. İstatistik Kongresi ve Risk Ölçümleri ve Yükümlülük Toplantısı, May 20-24, Antalya, Turkey, 413-418.

Nasibov, E.N., & Kınay, A.Ö. (2007b). *Öğrenci Performanslarının Değerlendirilmesinde Optimal Katsayıların Belirlenmesi için İteratif Bir Yöntem*, In Proc. Yöneylem Araştırması ve Endüstri Mühendisliği 27. Ulusal Kongresi, July 02-04, İzmir, Turkey, 8-13.

Nasibov, E.N., & Kınay, A.Ö. (2007c). *An Iterative Method for Student Performances Assessment in Fuzzy Evaluations*, In Proc. ICSCCW-2007, August 27-28, Antalya, Turkey, 242-250.

Nasibov, E.N., & Kınay, A.Ö. (2007d). An Iterative Approach For Graduation of Student Performances Based on Linguistic Evaluations, *Information Sciences,* submitted.

Nasibov, E.N., & Mert, A. (2007e). On methods of defuzzification of parametrically represented fuzzy numbers, *Automatic Control and Computer Sciences*, 41(5), 265-273.

Pedrycz, W., & Gomide, F. (1998). *An introduction to fuzzy sets*, The MIT Press.

Roychowdhury, S., & Pedrycz, W. (2001). A survey of defuzzification strategies. *International Journal of Intelligent Systems*, 16, 679-695.

Punnen, A.P., & Aneja, Y.P. (1993). Categorized assignment scheduling: A tabu search approach, *Journal of The Operational Research Society*, 44(7), 673-679.

Saghafian, S., & Hejazi, S.R. (2005). *Multi-criteria group decision making using a modified fuzzy TOPSIS procedure*, Proceedings of the CIMCA-IAWTIC'05.

Saaty, T.L. (1990). How to make a decision: The analytic hierarchy process, *European Journal of Operational Research*, 48, 9-26.

Tong, R.M., & Bonissone, P.P. (1980). A linguistic approach to decision making with fuzzy sets, *IEEE Trans. Systems Man Cybernet*, 10, 716-723.

Türkşen, I. B. (2005). *An ontological and epistemological perspective of fuzzy set theory*, Elsevier Science.

Xie, X., & Beni, G. (1991). A validity for fuzzy clustering, IEEE *Transactions On Pattern Analysis Machine Intelligence*, 3, 841-846.

Ulutagay, G. (2004). *Bulanık c-ortalamalar kümeleme analizi ve uygulamaları*, M.Sc. Thesis, Ege Üniversitesi, İzmir, Türkiye.

Votaw, D.F. & Orden, A. (1952). *The personnel assignment problem*, Symposium on Linear Inequalities and Programming, SCOOP 10, US Air Force, USA, 155-163.

Wang, Y-M., & Parkan, C. (2006). Two new approaches for assessing the weights of fuzzy opinions in group decision analysis, *Information Sciences*, 176, 3538-3555.

de Werra, D. (1985). An introduction to timetabling, *European Journal of Operational Research*, 19(2), 151-162.

Yager, R.R. (1988). On ordered weighted averaging aggregation operators in multicriteria decision making, *IEEE Transactions On Systems, Man, And Cybernetics*, 18(1), 183-190.

Yager, R.R. (2004). Generalized OWA aggregation operators, *Fuzzy Optimization and Decision Making*, 3, 93-107.

Yong, D., & Wen-Kang, S. (2003). Aggregating fuzzy opinions under group decision making, *Journal of Computer and Systems Sciences International*, 42(5), 727-731.

Zadeh, L. A. (1965). Fuzzy Sets, *Information and Control*, 8, 338-353.

Zadeh, L. A. (1975). The concept of a linguistic variable and its application to approximate reasoning, Part I, *Information Sciences*, 8, 199-249.

APPENDICES

APPENDIX A – Summary tables for the sample sizes 30, 50, 80 and 100 for each distribution according to different group numbers

A.1 The group averages and error ratio results which are obtained from Balanced Assignment method for $n=30$ and $N(50,16^2)$.

$n=30$		$m=3$	$m=4$	$m=5$	$m=6$	$\max p_i - \min p_j$	$n=30$		$m=3$	$m=4$	$m=5$	$m=6$	$\max p_i - \min p_j$
	Min	69.20	68.25	69.00	68.17			Min	46.20	44.50	46.50	45.40	
1	Max	70.50	71.86	71.60	70.33	97-53=44	**11**	Max	47.60	49.71	47.83	49.40	79-9=70
	E.R.	**0.0295**	**0.0820**	**0.0591**	**0.0492**			E.R.	**0.0200**	**0.0745**	**0.0190**	**0.0571**	
	Min	69.50	69.12	67.40	69.67			Min	48.50	46.12	47.50	46.20	
2	Max	70.50	71.29	71.60	70.50	86-54=32	**12**	Max	48.70	51.43	50.17	51.00	79-12=67
	E.R.	**0.0312**	**0.0675**	**0.1312**	**0.0260**			E.R.	**0.0030**	**0.0792**	**0.0398**	**0.0716**	
	Min	68.00	66.50	67.00	68.17			Min	45.60	43.00	45.83	42.00	
3	Max	69.60	72.00	72.00	69.00	94-45=49	**13**	Max	46.90	49.71	46.83	49.20	80-9=71
	E.R.	**0.0327**	**0.1122**	**0.1020**	**0.0170**			E.R.	**0.0183**	**0.0946**	**0.0141**	**0.1014**	
	Min	68.90	67.88	68.00	69.17			Min	50.90	47.62	50.33	47.60	
4	Max	70.10	71.71	72.00	70.67	97-51=46	**14**	Max	51.60	55.86	52.67	54.20	91-9=82
	E.R.	**0.0261**	**0.0835**	**0.0870**	**0.0326**			E.R.	**0.0085**	**0.1004**	**0.0285**	**0.0805**	
	Min	70.30	68.50	69.60	69.17			Min	51.20	49.75	51.33	48.60	
5	Max	70.50	72.71	71.00	71.00	89-49=40	**15**	Max	53.60	57.29	54.00	58.60	100-20=80
	E.R.	**0.0050**	**0.1054**	**0.0350**	**0.0458**			E.R.	**0.0300**	**0.0942**	**0.0333**	**0.1250**	
	Min	69.60	68.25	68.20	69.50			Min	45.30	42.62	44.83	43.40	
6	Max	70.30	72.29	71.00	70.67	87-50=37	**16**	Max	45.80	48.43	46.33	47.80	72-23=49
	E.R.	**0.0189**	**0.1091**	**0.0757**	**0.0315**			E.R.	**0.0102**	**0.1184**	**0.0306**	**0.0898**	
	Min	64.50	63.25	63.60	64.17			Min	44.40	43.25	44.17	44.40	
7	Max	65.00	66.86	66.20	65.50	83-48=35	**17**	Max	46.60	48.71	47.50	48.40	86-25=61
	E.R.	**0.0143**	**0.1031**	**0.0743**	**0.0381**			E.R.	**0.0361**	**0.0896**	**0.0546**	**0.0656**	
	Min	71.80	70.75	71.00	71.00			Min	55.60	53.00	55.67	54.20	
8	Max	72.80	74.86	74.60	74.17	100-54=46	**18**	Max	56.70	60.14	56.83	58.60	100-27=73
	E.R.	**0.0217**	**0.0893**	**0.0783**	**0.0688**			E.R.	**0.0151**	**0.0978**	**0.0160**	**0.0603**	
	Min	66.60	64.75	65.80	65.83			Min	46.40	44.38	46.17	44.20	
9	Max	67.40	69.00	68.20	67.83	83-46=37	**19**	Max	48.10	50.57	48.67	50.60	79-23=56
	E.R.	**0.0216**	**0.1149**	**0.0649**	**0.0541**			E.R.	**0.0304**	**0.1107**	**0.0446**	**0.1143**	
	Min	70.90	69.38	69.20	69.83			Min	51.10	48.88	50.67	47.80	
10	Max	71.30	72.71	73.20	72.17	97-54=43	**20**	Max	51.70	54.86	52.00	55.20	79-24=55
	E.R.	**0.0093**	**0.0777**	**0.0930**	**0.0543**			E.R.	**0.0109**	**0.1088**	**0.0242**	**0.1345**	

A.2 The group averages and error ratio results which are obtained from Balanced Assignment method for $n = 50$ and $N(50,16^2)$.

$n = 50$		$m = 5$	$m = 6$	$m = 7$	$m = 8$	$\max p_i - \min p_j$
1	Min	51.40	49.56	46.62	47.29	83–8=75
	Max	53.70	54.88	54.43	55.33	
	E.R.	**0.0306**	**0.0709**	**0.1041**	**0.1072**	
2	Min	52.10	49.00	46.25	47.00	90–13=77
	Max	52.80	54.88	54.57	55.00	
	E.R.	**0.0091**	**0.0763**	**0.1080**	**0.1038**	
3	Min	47.30	44.22	41.25	41.86	90–4=86
	Max	47.70	49.38	50.86	50.00	
	E.R.	**0.0047**	**0.0600**	**0.1117**	**0.0947**	
4	Min	52.90	50.22	48.75	49.14	80–19=61
	Max	53.40	54.88	54.43	54.83	
	E.R.	**0.0082**	**0.0764**	**0.0931**	**0.0933**	
5	Min	48.80	46.56	43.62	45.43	84–15=69
	Max	49.70	50.62	51.14	50.67	
	E.R.	**0.0130**	**0.0588**	**0.1090**	**0.0759**	
6	Min	47.55	48.00	46.62	47.00	87–19=68
	Max	50.80	51.25	51.43	52.17	
	E.R.	**0.0478**	**0.0478**	**0.0707**	**0.0760**	
7	Min	49.00	45.89	43.12	43.57	83–0=83
	Max	50.70	52.25	52.00	53.17	
	E.R.	**0.0205**	**0.0766**	**0.1070**	**0.1157**	
8	Min	48.20	46.56	43.62	46.14	88–19=69
	Max	49.50	50.38	51.29	50.50	
	E.R.	**0.0188**	**0.0554**	**0.1112**	**0.0632**	
9	Min	52.20	48.89	46.38	48.57	94–11=83
	Max	52.50	54.00	54.57	54.33	
	E.R.	**0.0036**	**0.0616**	**0.0987**	**0.0694**	
10	Min	50.30	48.56	45.38	47.43	94–15=79
	Max	51.50	52.75	54.43	52.83	
	E.R.	**0.0152**	**0.0530**	**0.1146**	**0.0684**	
11	Min	49.00	46.33	44.25	45.43	82-6=76
	Max	50.30	51.38	51.86	51.83	
	E.R.	**0.0171**	**0.0663**	**0.1001**	**0.0843**	
12	Min	52.00	50.44	48.88	49.57	91-25=66
	Max	53.50	53.50	54.86	54.17	
	E.R.	**0.0227**	**0.0463**	**0.0906**	**0.0696**	
13	Min	47.50	45.11	42.25	44.29	82-9=73
	Max	48.40	50.12	50.43	49.67	
	E.R.	**0.0123**	**0.0687**	**0.1120**	**0.0737**	
14	Min	49.60	48.56	46.12	47.71	87-20=67
	Max	51.50	52.38	53.29	52.50	
	E.R.	**0.0284**	**0.0570**	**0.1069**	**0.0714**	
15	Min	50.90	48.89	46.00	48.71	88-17=71
	Max	51.50	52.88	54.00	53.33	
	E.R.	**0.0085**	**0.0561**	**0.1127**	**0.0651**	
16	Min	49.90	48.44	46.25	47.43	83-19=64
	Max	50.80	52.00	52.14	52.83	
	E.R.	**0.0141**	**0.0556**	**0.0921**	**0.0844**	
17	Min	52.30	50.44	49.25	49.71	84-20=64
	Max	52.80	54.00	53.86	54.33	
	E.R.	**0.0078**	**0.0556**	**0.0720**	**0.0722**	
18	Min	49.90	48.22	46.12	47.14	100-18=82
	Max	52.00	53.00	55.71	52.83	
	E.R.	**0.0256**	**0.0583**	**0.1169**	**0.0694**	
19	Min	50.70	48.67	47.25	47.29	88-18=70
	Max	51.40	52.38	53.71	52.33	
	E.R.	**0.0100**	**0.0530**	**0.0923**	**0.0721**	
20	Min	51.60	50.44	48.00	49.43	88-23=65
	Max	52.60	53.38	54.14	53.50	
	E.R.	**0.0154**	**0.0451**	**0.0945**	**0.0626**	

A.3 The group averages and error ratio results which are obtained from Balanced Assignment method for $n = 80$ and $N(50,16^2)$.

$n = 80$		$m = 8$	$m = 9$	$m = 10$	$m = 11$	$\max p_i - \min p_j$	$n = 80$		$m = 8$	$m = 9$	$m = 10$	$m = 11$	$\max p_i - \min p_j$
1	Min	49.20	47.89	48.88	45.62	90–15=75	**11**	Min	52.20	50.89	52.12	47.62	88-9=79
	Max	49.80	51.75	50.25	53.14			Max	53.40	56.50	53.38	56.00	
	E.R.	**0.0080**	**0.0514**	**0.0183**	**0.1003**			E.R.	**0.0152**	**0.0710**	**0.0158**	**0.1060**	
2	Min	49.50	48.22	49.00	46.00	84–14=70	**12**	Min	49.60	48.44	49.50	45.62	97-12=85
	Max	50.30	52.75	50.25	51.86			Max	51.20	53.12	51.12	55.29	
	E.R.	**0.0114**	**0.0647**	**0.0178**	**0.0837**			E.R.	**0.0188**	**0.0551**	**0.0191**	**0.1137**	
3	Min	50.30	48.78	50.12	45.62	86–4=82	**13**	Min	50.90	49.67	50.88	46.00	97-9=88
	Max	51.80	53.88	52.38	54.29			Max	52.10	54.62	52.00	55.57	
	E.R.	**0.0183**	**0.0622**	**0.0276**	**0.1057**			E.R.	**0.0136**	**0.0563**	**0.0128**	**0.1088**	
4	Min	52.90	51.56	52.88	48.50	96–14=82	**14**	Min	49.20	47.33	49.12	44.25	88-6=82
	Max	53.70	56.75	53.75	57.14			Max	50.10	52.62	50.25	53.71	
	E.R.	**0.0098**	**0.0633**	**0.0106**	**0.1054**			E.R.	**0.0110**	**0.0645**	**0.0137**	**0.1154**	
5	Min	47.50	46.11	47.50	42.75	89–8=81	**15**	Min	49.60	48.00	49.50	45.62	88-12=76
	Max	48.20	51.88	48.38	51.00			Max	50.10	52.88	50.50	53.14	
	E.R.	**0.0086**	**0.0712**	**0.0109**	**0.1019**			E.R.	**0.0066**	**0.0641**	**0.0132**	**0.0989**	
6	Min	48.70	47.89	48.12	45.25	80–14=66	**16**	Min	48.50	48.33	48.62	46.00	72-18=54
	Max	49.30	52.25	49.50	51.14			Max	50.00	51.88	49.75	50.71	
	E.R.	**0.0091**	**0.0661**	**0.0209**	**0.0892**			E.R.	**0.0278**	**0.0656**	**0.0208**	**0.0873**	
7	Min	49.50	47.67	49.50	46.00	97–11=86	**17**	Min	51.00	50.11	50.62	47.50	92-17=75
	Max	50.80	52.88	50.75	53.29			Max	52.10	54.00	51.62	54.86	
	E.R.	**0.0151**	**0.0606**	**0.0145**	**0.0848**			E.R.	**0.0147**	**0.0519**	**0.0133**	**0.0981**	
8	Min	46.30	45.67	46.00	43.38	85–20=65	**18**	Min	52.10	50.44	51.75	48.75	94-20=74
	Max	47.20	49.38	47.62	49.71			Max	53.00	54.50	53.88	56.29	
	E.R.	**0.0138**	**0.0571**	**0.0249**	**0.0974**			E.R.	**0.0122**	**0.0548**	**0.0287**	**0.1018**	
9	Min	51.80	50.89	51.88	46.88	97–14=83	**19**	Min	50.90	49.56	49.25	47.12	84-9=75
	Max	53.80	55.12	52.88	56.86			Max	52.30	54.75	53.00	53.86	
	E.R.	**0.0241**	**0.0510**	**0.0120**	**0.1202**			E.R.	**0.0187**	**0.0693**	**0.0500**	**0.0898**	
10	Min	48.40	47.33	48.00	44.88	97–19=78	**20**	Min	49.00	47.67	48.50	44.75	84-10=74
	Max	49.80	50.75	51.00	52.57			Max	50.10	52.50	50.50	52.57	
	E.R.	**0.0179**	**0.0438**	**0.0385**	**0.0986**			E.R.	**0.0149**	**0.0653**	**0.0270**	**0.1057**	

A.4 The group averages and error ratio results which are obtained from Balanced Assignment method for $n = 100$ and $N(50,16^2)$.

$n = 100$		$m = 10$	$m = 11$	$m = 12$	$m = 13$	$\max p_i - \min p_j$	$n = 100$		$m = 10$	$m = 11$	$m = 12$	$m = 13$	$\max p_i - \min p_j$
1	Min	51.10	46.20	49.22	49.50	100–14=86	**11**	Min	47.10	42.40	44.78	44.88	88-6=82
	Max	52.60	54.33	53.38	57.00			Max	48.10	49.44	49.00	52.86	
	E.R.	**0.0174**	**0.0945**	**0.0484**	**0.0872**			E.R.	**0.0122**	**0.0859**	**0.0515**	**0.0973**	
2	Min	52.00	48.70	49.89	50.25	96–17=79	**12**	Min	49.40	45.20	46.44	47.38	94-10=84
	Max	53.40	54.67	53.75	57.57			Max	50.40	51.44	51.38	54.86	
	E.R.	**0.0177**	**0.0756**	**0.0488**	**0.0926**			E.R.	**0.0119**	**0.0743**	**0.0587**	**0.0891**	
3	Min	50.70	46.80	48.11	49.25	94–14=80	**13**	Min	46.30	42.50	44.22	45.00	80-10=70
	Max	51.90	53.33	52.62	55.71			Max	47.60	48.67	48.75	51.14	
	E.R.	**0.0150**	**0.0816**	**0.0564**	**0.0808**			E.R.	**0.0186**	**0.0881**	**0.0647**	**0.0878**	
4	Min	49.80	45.30	46.67	47.50	97–8=89	**14**	Min	50.80	46.70	48.22	48.62	94-6=88
	Max	51.10	52.67	52.00	56.43			Max	53.10	53.78	53.62	56.86	
	E.R.	**0.0146**	**0.0828**	**0.0599**	**0.1003**			E.R.	**0.0261**	**0.0804**	**0.0614**	**0.0935**	
5	Min	48.00	44.10	45.11	46.50	81–4=77	**15**	Min	51.50	47.20	48.78	49.88	88-14=74
	Max	49.80	50.67	51.50	53.00			Max	52.10	53.44	53.62	56.14	
	E.R.	**0.0234**	**0.0853**	**0.0830**	**0.0844**			E.R.	**0.0081**	**0.0844**	**0.0655**	**0.0847**	
6	Min	51.20	48.30	49.00	50.00	82–19=73	**16**	Min	51.20	47.10	49.33	50.12	97-10=87
	Max	52.20	52.67	53.50	55.43			Max	52.90	55.22	53.75	58.86	
	E.R.	**0.0137**	**0.0599**	**0.0616**	**0.0744**			E.R.	**0.0195**	**0.0934**	**0.0508**	**0.1004**	
7	Min	53.50	49.30	50.78	51.75	100–15=85	**17**	Min	48.50	44.40	46.00	46.62	88-13=75
	Max	55.50	57.44	55.88	60.86			Max	49.30	50.89	50.12	53.43	
	E.R.	**0.0235**	**0.0958**	**0.0600**	**0.1072**			E.R.	**0.0107**	**0.0865**	**0.0550**	**0.0907**	
8	Min	47.30	43.60	44.78	46.00	85–15=70	**18**	Min	50.20	46.40	47.44	48.62	87-11=76
	Max	48.30	49.78	49.12	52.29			Max	51.30	51.89	52.25	54.71	
	E.R.	**0.0143**	**0.0883**	**0.0620**	**0.0899**			E.R.	**0.0145**	**0.0722**	**0.0632**	**0.0801**	
9	Min	48.60	44.80	46.11	46.62	85–0=85	**19**	Min	47.40	43.40	44.78	45.88	94-11=83
	Max	51.10	51.56	52.12	54.29			Max	48.40	50.33	49.25	53.43	
	E.R.	**0.0294**	**0.0795**	**0.0707**	**0.0902**			E.R.	**0.0120**	**0.0835**	**0.0539**	**0.0910**	
10	Min	50.30	44.00	46.89	47.75	90–0=90	**20**	Min	49.50	46.40	48.56	48.00	94-24=70
	Max	51.60	53.56	52.88	56.14			Max	51.40	52.11	51.12	54.71	
	E.R.	**0.0144**	**0.1062**	**0.0666**	**0.0932**			E.R.	**0.0271**	**0.0816**	**0.0367**	**0.0959**	

A.5 The group averages and error ratio results which are obtained from Balanced Assignment method for $n = 30$ and $N(70,10^2)$.

$n=30$		$m=3$	$m=4$	$m=5$	$m=6$	$\max p_i - \min p_j$	$n=30$		$m=3$	$m=4$	$m=5$	$m=6$	$\max p_i - \min p_j$
	Min	69.20	68.25	68.17	69.00			Min	73.70	72.12	74.00	72.60	
1	Max	70.50	71.86	70.33	71.60	97-53=44	**11**	Max	74.40	76.71	74.17	76.20	93-50=43
	E.R.	**0.0295**	**0.0820**	**0.0492**	**0.0591**			E.R.	**0.0163**	**0.1067**	**0.0039**	**0.0837**	
	Min	69.50	69.12	69.67	67.40			Min	70.50	69.88	70.50	70.00	
2	Max	70.50	71.29	70.50	71.60	86-54=32	**12**	Max	71.30	72.57	71.17	72.60	89-54=35
	E.R.	**0.0312**	**0.0675**	**0.0260**	**0.1312**			E.R.	**0.0229**	**0.0770**	**0.0190**	**0.0743**	
	Min	68.00	66.50	68.17	67.00			Min	68.00	66.88	67.33	67.80	
3	Max	69.60	72.00	69.00	72.00	94-45=49	**13**	Max	68.50	69.71	68.67	69.00	87-55=32
	E.R.	**0.0327**	**0.1122**	**0.0170**	**0.1020**			E.R.	**0.0156**	**0.0887**	**0.0417**	**0.0375**	
	Min	68.90	67.88	69.17	68.00			Min	70.00	68.50	69.83	69.80	
4	Max	70.10	71.71	70.67	72.00	97-51=46	**14**	Max	70.60	72.43	71.17	71.40	87-55=32
	E.R.	**0.0261**	**0.0835**	**0.0326**	**0.0870**			E.R.	**0.0187**	**0.1228**	**0.0417**	**0.0500**	
	Min	70.30	68.50	69.17	69.60			Min	69.90	67.62	69.33	67.60	
5	Max	70.50	72.71	71.00	71.00	89-49=40	**15**	Max	70.00	73.29	70.83	72.40	94-42=52
	E.R.	**0.0050**	**0.1054**	**0.0458**	**0.0350**			E.R.	**0.0019**	**0.1089**	**0.0288**	**0.0923**	
	Min	69.60	68.25	69.50	68.20			Min	68.30	66.75	68.00	66.80	
6	Max	70.30	72.29	70.67	71.00	87-50=37	**16**	Max	69.40	71.57	69.17	72.00	89-45=44
	E.R.	**0.0189**	**0.1091**	**0.0315**	**0.0757**			E.R.	**0.0250**	**0.1096**	**0.0265**	**0.1182**	
	Min	64.50	63.25	64.17	63.60			Min	66.10	65.38	66.67	65.80	
7	Max	65.00	66.86	65.50	66.20	83-48=35	**17**	Max	67.40	68.00	67.17	67.60	84-47=37
	E.R.	**0.0143**	**0.1031**	**0.0381**	**0.0743**			E.R.	**0.0351**	**0.0709**	**0.0135**	**0.0486**	
	Min	71.80	70.75	71.00	71.00			Min	70.30	67.88	70.00	67.80	
8	Max	72.80	74.86	74.17	74.60	100-54=46	**18**	Max	70.50	73.29	70.83	73.20	97-42=55
	E.R.	**0.0217**	**0.0893**	**0.0688**	**0.0783**			E.R.	**0.0036**	**0.0984**	**0.0152**	**0.0982**	
	Min	66.60	64.75	65.83	65.80			Min	67.70	65.75	67.00	64.60	
9	Max	67.40	69.00	67.83	68.20	83-46=37	**19**	Max	68.00	70.57	68.17	70.80	93-45=48
	E.R.	**0.0216**	**0.1149**	**0.0541**	**0.0649**			E.R.	**0.0062**	**0.1004**	**0.0243**	**0.1292**	
	Min	70.90	69.38	69.83	69.20			Min	69.40	68.12	69.17	67.80	
10	Max	71.30	72.71	72.17	73.20	97-54=43	**20**	Max	70.10	71.29	70.67	71.00	86-50=36
	E.R.	**0.0093**	**0.0777**	**0.0543**	**0.0930**			E.R.	**0.0194**	**0.0878**	**0.0417**	**0.0889**	

A.6 The group averages and error ratio results which are obtained from Balanced Assignment method for $n = 50$ and $N(70,10^2)$.

$n = 50$		$m = 5$	$m = 6$	$m = 7$	$m = 8$	$\max p_i - \min p_j$	$n = 50$		$m = 5$	$m = 6$	$m = 7$	$m = 8$	$\max p_i - \min p_j$
1	Min	69.60	68.22	67.12	67.29	86-49=37	**11**	Min	70.90	69.00	67.12	68.43	97-46=51
	Max	70.10	71.00	70.86	71.17			Max	71.40	72.00	73.71	72.00	
	E.R.	**0.0135**	**0.0751**	**0.1009**	**0.1049**			E.R.	**0.0098**	**0.0588**	**0.1292**	**0.0700**	
2	Min	71.30	70.78	68.75	70.29	91-54=37	**12**	Min	71.60	71.00	69.12	70.29	97-55=42
	Max	72.30	73.00	73.57	73.00			Max	72.10	72.12	74.00	72.50	
	E.R.	**0.0270**	**0.0601**	**0.1303**	**0.0734**			E.R.	**0.0119**	**0.0268**	**0.1161**	**0.0527**	
3	Min	69.80	69.22	67.88	68.57	93-53=40	**13**	Min	68.27	69.00	68.50	68.57	94-43=51
	Max	70.30	70.88	71.57	70.83			Max	72.20	72.38	73.29	72.67	
	E.R.	**0.0125**	**0.0413**	**0.0924**	**0.0565**			E.R.	**0.0818**	**0.0703**	**0.0997**	**0.0853**	
4	Min	70.50	69.44	66.50	68.86	96-47=49	**14**	Min	69.50	68.56	66.12	67.86	94-47=47
	Max	71.00	71.88	73.14	72.33			Max	70.10	70.50	72.00	70.67	
	E.R.	**0.0102**	**0.0496**	**0.1356**	**0.0709**			E.R.	**0.0128**	**0.0414**	**0.1250**	**0.0598**	
5	Min	68.20	67.56	66.50	67.29	89-53=36	**15**	Min	69.30	68.11	68.00	67.71	93-54=39
	Max	68.80	69.38	69.86	69.33			Max	70.10	70.38	70.57	70.83	
	E.R.	**0.0167**	**0.0505**	**0.0933**	**0.0569**			E.R.	**0.0205**	**0.0580**	**0.0659**	**0.0800**	
6	Min	71.20	69.11	67.62	68.43	94-46=48	**16**	Min	67.40	65.33	64.62	64.43	98-46=52
	Max	71.60	72.62	73.43	72.67			Max	68.20	68.50	70.43	68.83	
	E.R.	**0.0083**	**0.0732**	**0.1209**	**0.0883**			E.R.	**0.0154**	**0.0609**	**0.1116**	**0.0847**	
7	Min	70.10	70.22	68.50	70.00	100-53=47	**17**	Min	68.80	67.22	65.12	66.43	89-46=43
	Max	71.60	71.33	72.71	71.17			Max	69.40	70.25	71.14	70.33	
	E.R.	**0.0319**	**0.0236**	**0.0897**	**0.0248**			E.R.	**0.0140**	**0.0704**	**0.1400**	**0.0908**	
8	Min	70.10	68.22	66.88	68.00	96-46=50	**18**	Min	69.20	68.33	67.12	67.43	86-48=38
	Max	70.80	71.75	72.00	71.67			Max	70.50	70.75	70.86	71.00	
	E.R.	**0.0140**	**0.0706**	**0.1025**	**0.0733**			E.R.	**0.0342**	**0.0636**	**0.0982**	**0.0940**	
9	Min	68.30	66.78	64.88	66.71	89-45=44	**19**	Min	69.40	68.33	67.25	67.57	94-53=41
	Max	68.90	69.62	70.86	69.83			Max	70.60	70.75	72.00	70.83	
	E.R.	**0.0136**	**0.0647**	**0.1360**	**0.0709**			E.R.	**0.0293**	**0.0589**	**0.1159**	**0.0796**	
10	Min	69.30	68.00	66.38	67.57	96-49=47	**20**	Min	70.70	69.44	68.12	68.71	89-48=41
	Max	69.70	70.38	71.43	70.50			Max	71.30	72.00	71.71	72.17	
	E.R.	**0.0085**	**0.0505**	**0.1075**	**0.0623**			E.R.	**0.0146**	**0.0623**	**0.0875**	**0.0842**	

A.7 The group averages and error ratio results which are obtained from Balanced Assignment method for $n = 80$ and $N(70,10^2)$.

$n=80$		$m=8$	$m=9$	$m=10$	$m=11$	$\max p_i - \min p_j$	$n=80$		$m=8$	$m=9$	$m=10$	$m=11$	$\max p_i - \min p_j$
	Min	70.10	69.22	70.00	67.00			Min	69.20	68.44	68.88	66.38	
1	Max	71.00	72.12	71.25	73.43	100-46=54	**11**	Max	70.10	71.88	70.25	71.71	84-42=42
	E.R.	**0.0167**	**0.0538**	**0.0231**	**0.1190**			E.R.	**0.0187**	**0.0715**	**0.0286**	**0.1112**	
	Min	69.60	68.33	69.38	66.62			Min	70.00	69.33	69.50	67.50	
2	Max	70.20	71.50	70.12	72.86	96-43=53	**12**	Max	71.00	72.12	70.88	73.00	100-46=54
	E.R.	**0.0113**	**0.0597**	**0.0142**	**0.1176**			E.R.	**0.0185**	**0.0517**	**0.0255**	**0.1019**	
	Min	70.40	69.22	70.38	67.88			Min	70.40	70.00	70.38	68.62	
3	Max	70.90	72.00	71.00	72.71	96-46=50	**13**	Max	71.30	72.25	71.62	72.71	94-52=42
	E.R.	**0.0100**	**0.0556**	**0.0125**	**0.0968**			E.R.	**0.0214**	**0.0536**	**0.0298**	**0.0974**	
	Min	70.50	70.56	70.75	68.62			Min	68.50	67.78	68.50	66.25	
4	Max	71.70	72.62	71.38	73.57	93-52=41	**14**	Max	69.70	70.62	69.75	71.57	97-48=49
	E.R.	**0.0293**	**0.0505**	**0.0152**	**0.1206**			E.R.	**0.0245**	**0.0581**	**0.0255**	**0.1086**	
	Min	71.70	70.67	71.38	69.00			Min	70.70	70.11	70.75	68.38	
5	Max	72.40	74.00	72.38	74.29	94-46=48	**15**	Max	71.50	73.25	71.38	73.57	93-47=46
	E.R.	**0.0146**	**0.0694**	**0.0208**	**0.1101**			E.R.	**0.0174**	**0.0682**	**0.0136**	**0.1130**	
	Min	71.10	70.11	71.38	68.62			Min	69.90	68.33	69.88	66.75	
6	Max	72.70	73.12	72.75	74.14	97-46=51	**16**	Max	71.10	71.75	71.25	72.86	98-42=56
	E.R.	**0.0314**	**0.0591**	**0.0270**	**0.1082**			E.R.	**0.0214**	**0.0610**	**0.0246**	**0.1091**	
	Min	68.20	67.67	67.88	66.75			Min	70.30	68.89	70.00	67.88	
7	Max	69.60	69.88	69.62	71.00	94-50=44	**17**	Max	70.80	71.88	71.00	72.57	93-46=47
	E.R.	**0.0318**	**0.0502**	**0.0398**	**0.0966**			E.R.	**0.0106**	**0.0635**	**0.0213**	**0.0999**	
	Min	70.10	69.67	70.25	67.75			Min	69.40	69.00	69.75	67.62	
8	Max	71.00	72.25	70.88	73.00	98-46=52	**18**	Max	70.50	71.88	70.38	71.57	90-46=44
	E.R.	**0.0173**	**0.0497**	**0.0120**	**0.1010**			E.R.	**0.0250**	**0.0653**	**0.0142**	**0.0897**	
	Min	69.20	68.89	69.00	67.12			Min	70.60	69.56	70.75	68.00	
9	Max	69.80	71.25	69.88	71.14	86-48=38	**19**	Max	71.80	72.88	71.75	73.57	98-46=52
	E.R.	**0.0158**	**0.0621**	**0.0230**	**0.1057**			E.R.	**0.0231**	**0.0638**	**0.0192**	**0.1071**	
	Min	71.30	69.67	71.25	68.50			Min	69.00	67.78	68.88	66.62	
10	Max	71.90	73.00	72.00	73.71	98-43=55	**20**	Max	69.80	70.33	70.00	71.57	97-47=50
	E.R.	**0.0109**	**0.0606**	**0.0136**	**0.0948**			E.R.	**0.0160**	**0.0511**	**0.0225**	**0.0989**	

A.8 The group averages and error ratio results which are obtained from Balanced Assignment method for $n=100$ and $N(70,10^2)$.

$n=100$		$m=10$	$m=11$	$m=12$	$m=13$	$\max p_i - \min p_j$	$n=100$		$m=10$	$m=11$	$m=12$	$m=13$	$\max p_i - \min p_j$
1	Min	68.30	66.30	66.67	67.25	90-46=44	**11**	Min	71.00	67.80	69.33	69.62	93-43=50
	Max	68.80	69.11	69.50	71.14			Max	72.00	72.78	72.62	74.71	
	E.R.	**0.0114**	**0.0639**	**0.0644**	**0.0885**			E.R.	**0.0200**	**0.0996**	**0.0658**	**0.1018**	
2	Min	71.90	68.20	70.11	70.38	100-45=55	**12**	Min	69.80	66.90	69.22	68.50	100-45=55
	Max	72.40	74.00	73.25	74.86			Max	71.20	72.44	71.38	73.57	
	E.R.	**0.0091**	**0.1055**	**0.0571**	**0.0815**			E.R.	**0.0255**	**0.1008**	**0.0391**	**0.0922**	
3	Min	69.70	66.70	67.44	67.88	100-42=58	**13**	Min	69.30	66.70	67.89	68.25	94-46=48
	Max	70.30	71.67	70.75	73.71			Max	70.00	70.78	70.62	72.71	
	E.R.	**0.0103**	**0.0856**	**0.0570**	**0.1007**			E.R.	**0.0146**	**0.0850**	**0.0570**	**0.0930**	
4	Min	69.80	66.80	68.22	68.50	98-43=55	**14**	Min	68.70	65.70	67.11	67.75	96-42=54
	Max	70.60	72.22	71.25	74.00			Max	69.60	70.89	70.50	73.00	
	E.R.	**0.0145**	**0.0986**	**0.0551**	**0.1000**			E.R.	**0.0167**	**0.0961**	**0.0628**	**0.0972**	
5	Min	69.00	66.70	67.22	68.00	90-45=45	**15**	Min	68.30	66.50	67.44	67.62	98-50=48
	Max	69.80	70.33	70.50	72.14			Max	69.80	70.56	69.50	72.43	
	E.R.	**0.0178**	**0.0807**	**0.0728**	**0.0921**			E.R.	**0.0312**	**0.0845**	**0.0428**	**0.1001**	
6	Min	70.10	67.70	68.56	68.75	100-48=52	**16**	Min	70.40	67.70	68.78	69.25	90-48=42
	Max	70.80	71.89	71.12	73.57			Max	71.00	71.67	71.62	72.57	
	E.R.	**0.0135**	**0.0806**	**0.0494**	**0.0927**			E.R.	**0.0143**	**0.0944**	**0.0678**	**0.0791**	
7	Min	69.80	67.20	68.11	68.75	90-46=44	**17**	Min	70.70	68.50	69.00	69.50	97-48=49
	Max	70.30	71.22	71.12	72.57			Max	71.10	72.11	71.88	74.57	
	E.R.	**0.0114**	**0.0914**	**0.0685**	**0.0869**			E.R.	**0.0082**	**0.0737**	**0.0587**	**0.1035**	
8	Min	67.80	64.90	65.89	66.50	93-42=51	**18**	Min	70.60	68.30	69.11	69.75	94-45=49
	Max	68.30	69.22	69.00	70.86			Max	71.20	72.22	72.00	74.00	
	E.R.	**0.0098**	**0.0847**	**0.0610**	**0.0854**			E.R.	**0.0122**	**0.0800**	**0.0590**	**0.0867**	
9	Min	69.70	67.00	68.44	68.38	100-45=55	**19**	Min	70.50	68.10	69.00	69.25	98-48=50
	Max	71.10	71.78	71.38	74.29			Max	71.00	72.33	71.62	74.14	
	E.R.	**0.0255**	**0.0869**	**0.0533**	**0.1075**			E.R.	**0.0100**	**0.0847**	**0.0525**	**0.0979**	
10	Min	69.90	67.20	68.22	68.88	96-46=50	**20**	Min	71.30	69.00	69.78	70.38	96-51=45
	Max	70.80	72.11	71.50	73.86			Max	72.10	72.89	72.25	74.29	
	E.R.	**0.0180**	**0.0982**	**0.0656**	**0.0996**			E.R.	**0.0178**	**0.0864**	**0.0549**	**0.0869**	

A.9 The group averages and error ratio results which are obtained from Balanced Assignment method for $n = 30$ and Gamma(14,3) .

$n = 30$		$m = 3$	$m = 4$	$m = 5$	$m = 6$	$\max p_i - \min p_j$	$n = 30$		$m = 3$	$m = 4$	$m = 5$	$m = 6$	$\max p_i - \min p_j$
	Min	38.40	37.25	38.00	37.80			Min	42.70	41.25	42.33	41.60	
1	Max	39.50	41.29	40.50	41.20	64-23=41	**11**	Max	44.00	46.00	44.33	46.40	80-25=55
	E.R.	**0.0268**	**0.0984**	**0.0610**	**0.0829**			E.R.	**0.0236**	**0.0864**	**0.0364**	**0.0873**	
	Min	39.90	38.75	39.50	39.20			Min	42.30	41.62	42.17	41.20	
2	Max	40.60	41.71	40.83	41.20	58-23=35	**12**	Max	43.10	43.57	43.00	43.40	68-24=44
	E.R.	**0.0200**	**0.0847**	**0.0381**	**0.0571**			E.R.	**0.0182**	**0.0442**	**0.0189**	**0.0500**	
	Min	41.60	40.25	41.33	40.60			Min	37.20	36.38	37.17	36.20	
3	Max	42.70	44.86	43.50	44.80	73-20=53	**13**	Max	37.50	38.14	37.50	38.20	52-24=28
	E.R.	**0.0208**	**0.0869**	**0.0409**	**0.0792**			E.R.	**0.0107**	**0.0631**	**0.0119**	**0.0714**	
	Min	41.40	40.00	41.33	40.00			Min	36.70	36.00	36.17	35.00	
4	Max	42.10	44.00	42.17	44.20	61-24=37	**14**	Max	37.30	37.86	37.33	38.60	54-18=36
	E.R.	**0.0189**	**0.1081**	**0.0225**	**0.1135**			E.R.	**0.0167**	**0.0516**	**0.0324**	**0.1000**	
	Min	42.60	40.88	42.67	41.40			Min	43.60	41.62	43.00	41.80	
5	Max	43.00	45.14	43.17	44.60	64-24=40	**15**	Max	45.60	48.29	47.17	49.40	83-23=60
	E.R.	**0.0100**	**0.1067**	**0.0125**	**0.0800**			E.R.	**0.0333**	**0.1110**	**0.0694**	**0.1267**	
	Min	40.80	39.25	40.17	39.00			Min	39.10	38.00	39.00	38.40	
6	Max	41.90	44.43	43.17	44.80	72-22=50	**16**	Max	39.50	40.71	39.67	40.20	58-22=36
	E.R.	**0.0220**	**0.1036**	**0.0600**	**0.1160**			E.R.	**0.0111**	**0.0754**	**0.0185**	**0.0500**	
	Min	40.60	38.75	40.33	37.40			Min	42.20	41.00	42.00	39.60	
7	Max	41.70	43.86	41.67	43.80	69-17=52	**17**	Max	43.00	44.29	42.83	44.80	69-22=47
	E.R.	**0.0212**	**0.0982**	**0.0256**	**0.1231**			E.R.	**0.0170**	**0.0699**	**0.0177**	**0.1106**	
	Min	43.10	42.50	42.67	42.80			Min	38.40	37.38	37.83	36.60	
8	Max	44.40	45.14	44.83	44.80	68-30=38	**18**	Max	39.20	40.14	39.17	40.20	61-18=43
	E.R.	**0.0342**	**0.0695**	**0.0570**	**0.0526**			E.R.	**0.0186**	**0.0644**	**0.0310**	**0.0837**	
	Min	42.20	40.88	42.17	41.00			Min	42.80	40.50	42.67	41.80	
9	Max	43.00	44.29	43.17	43.40	54-23=31	**19**	Max	43.80	46.57	43.67	45.80	74-18=56
	E.R.	**0.0258**	**0.1100**	**0.0323**	**0.0774**			E.R.	**0.0179**	**0.1084**	**0.0179**	**0.0714**	
	Min	46.40	44.75	45.83	44.80			Min	42.80	41.12	42.50	42.60	
10	Max	48.70	51.71	50.50	52.60	92-24=68	**20**	Max	43.90	46.43	44.67	45.20	72-22=50
	E.R.	**0.0338**	**0.1024**	**0.0686**	**0.1147**			E.R.	**0.0216**	**0.1040**	**0.0425**	**0.0510**	

A.10 The group averages and error ratio results which are obtained from Balanced Assignment method for $n = 50$ and Gamma(14,3) .

$n = 50$		$m = 5$	$m = 6$	$m = 7$	$m = 8$	$\max p_i - \min p_j$	$n = 50$		$m = 5$	$m = 6$	$m = 7$	$m = 8$	$\max p_i - \min p_j$
1	Min	40.30	38.67	37.50	37.86	68-19=49	**11**	Min	38.80	38.56	37.25	37.71	61-25=36
	Max	41.20	41.38	43.29	41.67			Max	39.30	39.62	40.43	39.83	
	E.R.	**0.0184**	**0.0553**	**0.1181**	**0.0777**			E.R.	**0.0139**	**0.0297**	**0.0883**	**0.0589**	
2	Min	40.50	39.11	37.62	38.71	82-21=61	**12**	Min	43.00	41.67	39.38	41.00	65-18=47
	Max	42.00	41.67	44.43	41.71			Max	43.50	44.12	44.86	44.33	
	E.R.	**0.0246**	**0.0419**	**0.1115**	**0.0492**			E.R.	**0.0106**	**0.0523**	**0.1166**	**0.0709**	
3	Min	43.90	43.11	42.75	42.86	77-26=51	**13**	Min	40.00	39.89	36.75	40.00	83-18=65
	Max	45.80	45.50	46.29	46.00			Max	41.70	41.25	44.00	41.50	
	E.R.	**0.0373**	**0.0468**	**0.0693**	**0.0616**			E.R.	**0.0262**	**0.0209**	**0.1115**	**0.0231**	
4	Min	41.30	40.22	38.25	39.14	73-18=55	**14**	Min	40.70	39.78	38.38	39.29	82-22=60
	Max	41.90	42.38	43.14	42.67			Max	41.90	42.00	43.71	42.00	
	E.R.	**0.0109**	**0.0391**	**0.0890**	**0.0641**			E.R.	**0.0200**	**0.0370**	**0.0890**	**0.0452**	
5	Min	41.00	40.11	38.00	40.14	75-21=54	**15**	Min	44.40	43.22	42.00	42.29	68-24=44
	Max	42.60	42.38	44.57	42.67			Max	45.40	45.62	46.14	46.00	
	E.R.	**0.0296**	**0.0419**	**0.1217**	**0.0467**			E.R.	**0.0227**	**0.0546**	**0.0942**	**0.0844**	
6	Min	42.00	40.89	38.62	40.43	71-19=52	**16**	Min	40.90	39.56	37.62	38.71	74-17=57
	Max	42.50	43.50	44.29	43.17			Max	41.80	42.25	44.14	42.17	
	E.R.	**0.0096**	**0.0502**	**0.1089**	**0.0527**			E.R.	**0.0158**	**0.0473**	**0.1143**	**0.0606**	
7	Min	44.90	44.00	43.12	44.00	63-30=33	**17**	Min	39.10	38.33	37.12	37.14	61-17=44
	Max	45.40	46.12	45.86	46.17			Max	40.80	41.25	41.57	41.67	
	E.R.	**0.0152**	**0.0644**	**0.0828**	**0.0657**			E.R.	**0.0386**	**0.0663**	**0.1011**	**0.1028**	
8	Min	36.60	36.22	34.62	36.00	75-22=53	**18**	Min	42.50	42.11	39.12	41.43	83-17=66
	Max	38.80	37.78	40.57	38.29			Max	44.10	43.75	46.71	44.17	
	E.R.	**0.0415**	**0.0294**	**0.1122**	**0.0431**			E.R.	**0.0242**	**0.0248**	**0.1150**	**0.0415**	
9	Min	40.20	39.11	36.75	38.57	77-19=58	**19**	Min	40.40	39.78	37.88	39.14	80-17=63
	Max	41.90	41.38	43.86	41.50			Max	42.40	42.50	44.29	42.50	
	E.R.	**0.0293**	**0.0390**	**0.1225**	**0.0505**			E.R.	**0.0317**	**0.0432**	**0.1018**	**0.0533**	
10	Min	43.90	42.67	41.50	42.57	71-24=47	**20**	Min	42.20	41.89	39.88	42.00	75-23=52
	Max	44.20	44.75	45.57	45.17			Max	43.60	43.62	45.43	44.00	
	E.R.	**0.0064**	**0.0443**	**0.0866**	**0.0552**			E.R.	**0.0269**	**0.0334**	**0.1068**	**0.0385**	

A.11 The group averages and error ratio results which are obtained from Balanced Assignment method for $n = 80$ and Gamma(14,3) .

$n = 80$		$m = 8$	$m = 9$	$m = 10$	$m = 11$	$\max p_i - \min p_j$	$n = 80$		$m = 8$	$m = 9$	$m = 10$	$m = 11$	$\max p_i - \min p_j$
1	Min	42.30	41.56	42.00	39.88	74-19=55	**11**	Min	42.90	42.22	43.00	40.62	83-19=64
	Max	43.30	44.38	43.62	45.14			Max	44.70	45.78	45.12	46.86	
	E.R.	**0.0182**	**0.0513**	**0.0295**	**0.0958**			E.R.	**0.0281**	**0.0556**	**0.0332**	**0.0974**	
2	Min	41.10	40.22	41.00	38.75	75-20=55	**12**	Min	40.00	39.44	40.12	38.12	71-19=52
	Max	42.70	43.22	43.00	44.43			Max	41.10	42.22	41.62	43.57	
	E.R.	**0.0291**	**0.0545**	**0.0364**	**0.1032**			E.R.	**0.0212**	**0.0534**	**0.0288**	**0.1047**	
3	Min	41.40	40.89	41.12	38.88	73-18=55	**13**	Min	40.40	39.44	40.38	37.88	83-18=65
	Max	42.80	44.00	42.88	44.86			Max	42.70	43.67	42.88	44.57	
	E.R.	**0.0255**	**0.0566**	**0.0318**	**0.1088**			E.R.	**0.0354**	**0.0650**	**0.0385**	**0.1030**	
4	Min	42.40	42.22	42.75	40.88	75-23=52	**14**	Min	40.20	39.67	40.25	37.88	77-21=56
	Max	44.20	44.62	44.00	45.57			Max	42.50	42.78	42.62	44.86	
	E.R.	**0.0346**	**0.0462**	**0.0240**	**0.0903**			E.R.	**0.0411**	**0.0556**	**0.0424**	**0.1247**	
5	Min	41.40	41.22	41.75	38.75	83-18=65	**15**	Min	41.40	40.78	41.50	39.12	63-19=44
	Max	43.80	44.38	44.00	46.00			Max	42.10	43.62	42.12	43.43	
	E.R.	**0.0369**	**0.0485**	**0.0346**	**0.1115**			E.R.	**0.0159**	**0.0647**	**0.0142**	**0.0978**	
6	Min	40.80	40.78	41.12	39.50	62-25=37	**16**	Min	43.40	42.67	43.00	40.62	92-23=69
	Max	41.70	42.62	41.50	42.71			Max	46.10	46.33	47.25	48.71	
	E.R.	**0.0243**	**0.0499**	**0.0101**	**0.0869**			E.R.	**0.0391**	**0.0531**	**0.0616**	**0.1172**	
7	Min	45.60	44.11	45.88	42.75	75-17=58	**17**	Min	40.60	40.22	40.62	37.88	83-19=64
	Max	46.50	47.75	46.25	48.29			Max	43.00	43.56	43.62	45.86	
	E.R.	**0.0155**	**0.0627**	**0.0065**	**0.0954**			E.R.	**0.0375**	**0.0521**	**0.0469**	**0.1247**	
8	Min	41.10	40.11	41.00	38.62	92-17=75	**18**	Min	40.70	40.00	40.75	38.25	61-18=43
	Max	43.80	44.78	44.62	47.14			Max	41.50	42.62	41.50	43.00	
	E.R.	**0.0360**	**0.0622**	**0.0483**	**0.1136**			E.R.	**0.0186**	**0.0610**	**0.0174**	**0.1105**	
9	Min	41.40	40.78	41.50	39.50	73-21=52	**19**	Min	41.20	40.56	41.12	39.00	68-19=49
	Max	42.80	43.50	42.50	44.71			Max	41.70	43.12	41.88	43.71	
	E.R.	**0.0269**	**0.0524**	**0.0192**	**0.1003**			E.R.	**0.0102**	**0.0524**	**0.0153**	**0.0962**	
10	Min	41.40	40.78	41.25	39.12	73-21=52	**20**	Min	41.60	41.00	41.25	39.25	72-23=49
	Max	42.40	43.12	42.38	44.14			Max	42.30	43.25	43.25	44.43	
	E.R.	**0.0192**	**0.0451**	**0.0216**	**0.0965**			E.R.	**0.0143**	**0.0459**	**0.0408**	**0.1057**	

A.12 The group averages and error ratio results which are obtained from Balanced Assignment method for $n = 100$ and Gamma(14,3) .

$n = 100$		$m = 10$	$m = 11$	$m = 12$	$m = 13$	$\max p_i - \min p_j$	$n = 100$		$m = 10$	$m = 11$	$m = 12$	$m = 13$	$\max p_i - \min p_j$
	Min	41.50	38.60	40.89	40.75			Min	41.50	39.20	39.78	40.38	
1	Max	44.30	45.56	43.56	47.71	92-21=71	**11**	Max	41.80	42.67	42.62	44.57	64-18=46
	E.R.	**0.0394**	**0.0980**	**0.0376**	**0.0981**			E.R.	**0.0065**	**0.0754**	**0.0619**	**0.0912**	
	Min	41.20	38.80	39.56	40.38			Min	42.10	39.90	40.89	41.12	
2	Max	43.20	44.56	42.50	46.29	82-19=63	**12**	Max	42.70	43.89	43.12	45.57	72-22=50
	E.R.	**0.0317**	**0.0914**	**0.0467**	**0.0938**			E.R.	**0.0120**	**0.0798**	**0.0447**	**0.0889**	
	Min	41.60	38.80	39.78	40.25			Min	42.00	39.40	41.56	41.12	
3	Max	42.60	44.22	42.62	45.86	77-18=59	**13**	Max	43.60	44.89	43.62	46.57	82-19=63
	E.R.	**0.0169**	**0.0919**	**0.0483**	**0.0950**			E.R.	**0.0254**	**0.0871**	**0.0328**	**0.0865**	
	Min	42.60	39.70	41.89	41.50			Min	42.60	40.10	41.00	41.75	
4	Max	44.00	45.33	44.50	47.57	82-18=64	**14**	Max	44.30	45.67	44.00	47.71	80-20=60
	E.R.	**0.0219**	**0.0880**	**0.0408**	**0.0949**			E.R.	**0.0283**	**0.0928**	**0.0500**	**0.0994**	
	Min	43.30	41.10	42.56	42.25			Min	42.40	39.30	40.67	41.12	
5	Max	45.00	46.00	44.75	47.57	83-24=59	**15**	Max	44.10	45.89	43.50	47.71	82-17=65
	E.R.	**0.0288**	**0.0831**	**0.0372**	**0.0902**			E.R.	**0.0262**	**0.1014**	**0.0436**	**0.1014**	
	Min	40.20	38.40	40.11	39.38			Min	39.70	37.50	38.00	38.50	
6	Max	41.80	42.44	42.50	44.29	77-21=56	**16**	Max	40.50	41.44	41.12	42.71	63-17=46
	E.R.	**0.0286**	**0.0722**	**0.0427**	**0.0877**			E.R.	**0.0174**	**0.0857**	**0.0679**	**0.0916**	
	Min	40.60	38.40	39.22	39.75			Min	42.30	40.00	40.78	41.25	
7	Max	41.10	41.67	41.88	43.14	63-20=43	**17**	Max	44.10	45.56	43.50	47.43	82-21=61
	E.R.	**0.0116**	**0.0760**	**0.0617**	**0.0789**			E.R.	**0.0295**	**0.0911**	**0.0446**	**0.1013**	
	Min	39.40	37.30	38.22	38.38			Min	40.18	38.30	40.00	40.12	
8	Max	41.50	42.33	40.56	44.43	83-21=62	**18**	Max	42.10	44.00	42.38	45.71	80-18=62
	E.R.	**0.0339**	**0.0812**	**0.0376**	**0.0976**			E.R.	**0.0309**	**0.0919**	**0.0383**	**0.0901**	
	Min	39.00	36.60	37.67	37.38			Min	41.30	38.90	40.56	40.00	
9	Max	39.40	40.44	39.62	41.57	65-18=47	**19**	Max	42.80	44.33	43.12	47.00	82-17=65
	E.R.	**0.0085**	**0.0818**	**0.0417**	**0.0893**			E.R.	**0.0231**	**0.0836**	**0.0395**	**0.1077**	
	Min	42.40	40.00	41.44	41.50			Min	41.80	39.70	40.89	41.00	
10	Max	44.90	46.00	44.22	48.71	92-21=71	**20**	Max	43.80	44.44	43.25	47.00	82-23=59
	E.R.	**0.0352**	**0.0845**	**0.0391**	**0.1016**			E.R.	**0.0339**	**0.0804**	**0.0400**	**0.1017**	

APPENDIX B – Borland C++ Builder 6.0 Code for Optimal Weights Evaluation

```
//---------------------------------------------------------------------
#include <vcl.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
#pragma hdrstop
#include "OED_GS.h"
#include "Unit2_GS.h"
#include "Unit3_GS.h"
int n1,h,z,p,k,k1,k3,i,i1,j,j1,ogrno,MAXOGR,y;
double
xx,yyy,x,yy,X,XX[200],Y,v,t,pot,mem[200],po[200],R[200][5],R1[200][5],R2[200][5],G[20
0][200],GI[200][200],e[200],et[200];
double
A[200],c[200],w[200],d[200],f[200],g[200],fg,o[200],Z[200],GIT,tahmin[200],beta;
double Duru(int a,int b,int c,int d,double beta);
double Duru1(int a,int b,int c,int d);
double Duru2(int a,int b,int c,int d);
char s1[20],s[100];
bool dosya=false;
double nbeta0,nbeta1,pay1,pay2,pay3,pay4,payda1,payda2;
double eb=0.0,ek=1000000.0,son=0.0;
#define m 5
#define n 100
#define qq 10
//---------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "CSPIN"
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
DecimalSeparator='.';
}
//---------------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
FILE *ff=fopen("c:\\RRR.txt","a");
if(!dosya)
  {
  MAXOGR=Edit1->Text.ToInt();
  ogrno=CSpinEdit1->Value;
  A[ogrno]=Edit3->Text.ToDouble();
  fprintf(ff,"%d %6.2f %d %d %d %d %d\n",ogrno,A[ogrno],
   RadioGroup1->ItemIndex,
   RadioGroup2->ItemIndex,
   RadioGroup3->ItemIndex,
   RadioGroup4->ItemIndex,
   RadioGroup5->ItemIndex);
  fclose(ff);
  }
dosya=false;
{
switch(RadioGroup1->ItemIndex)
        {
        case 0: R[ogrno][0]=Duru(0,0,10,50,beta);
               R1[ogrno][0]=Duru1(0,0,10,50);
               R2[ogrno][0]=Duru2(0,0,10,50);break;
        case 1: R[ogrno][0]=Duru(10,50,50,90,beta);
               R1[ogrno][0]=Duru1(10,50,50,90);
               R2[ogrno][0]=Duru2(10,50,50,90);break;
        case 2: R[ogrno][0]=Duru(50,90,100,100,beta);
               R1[ogrno][0]=Duru1(50,90,100,100);
               R2[ogrno][0]=Duru2(50,90,100,100);break;
        }
switch(RadioGroup2->ItemIndex)
        {
```

```
        case 0: R[ogrno][1]=Duru(0,0,10,50,beta);
               R1[ogrno][1]=Duru1(0,0,10,50);
               R2[ogrno][1]=Duru2(0,0,10,50);break;
        case 1: R[ogrno][1]=Duru(10,50,50,90,beta);
               R1[ogrno][1]=Duru1(10,50,50,90);
               R2[ogrno][1]=Duru2(10,50,50,90);break;
        case 2: R[ogrno][1]=Duru(50,90,100,100,beta);
               R1[ogrno][1]=Duru1(50,90,100,100);
               R2[ogrno][1]=Duru2(50,90,100,100);break;
        }
switch(RadioGroup3->ItemIndex)
        {
        case 0: R[ogrno][2]=Duru(0,0,10,30,beta);
               R1[ogrno][2]=Duru1(0,0,10,30);
               R2[ogrno][2]=Duru2(0,0,10,30);break;
        case 1: R[ogrno][2]=Duru(10,30,30,50,beta);
               R1[ogrno][2]=Duru1(10,30,30,50);
               R2[ogrno][2]=Duru2(10,30,30,50);break;
        case 2: R[ogrno][2]=Duru(30,50,50,70,beta);
               R1[ogrno][2]=Duru1(30,50,50,70);
               R2[ogrno][2]=Duru2(30,50,50,70);break;
        case 3: R[ogrno][2]=Duru(50,70,70,90,beta);
               R1[ogrno][2]=Duru1(50,70,70,90);
               R2[ogrno][2]=Duru2(50,70,70,90);break;
        case 4: R[ogrno][2]=Duru(70,90,100,100,beta);
               R1[ogrno][2]=Duru1(70,90,100,100);
               R2[ogrno][2]=Duru2(70,90,100,100);break;
        }
switch(RadioGroup4->ItemIndex)
        {
        case 0: R[ogrno][3]=Duru(0,0,10,30,beta);
               R1[ogrno][3]=Duru1(0,0,10,30);
               R2[ogrno][3]=Duru2(0,0,10,30);break;
        case 1: R[ogrno][3]=Duru(10,30,30,50,beta);
               R1[ogrno][3]=Duru1(10,30,30,50);
               R2[ogrno][3]=Duru2(10,30,30,50);break;
        case 2: R[ogrno][3]=Duru(30,50,50,70,beta);
               R1[ogrno][3]=Duru1(30,50,50,70);
               R2[ogrno][3]=Duru2(30,50,50,70);break;
        case 3: R[ogrno][3]=Duru(50,70,70,90,beta);
               R1[ogrno][3]=Duru1(50,70,70,90);
               R2[ogrno][3]=Duru2(50,70,70,90);break;
        case 4: R[ogrno][3]=Duru(70,90,100,100,beta);
               R1[ogrno][3]=Duru1(70,90,100,100);
               R2[ogrno][3]=Duru2(70,90,100,100);break;
        }
switch(RadioGroup5->ItemIndex)
        {
        case 0: R[ogrno][4]=Duru(0,0,10,30,beta);
               R1[ogrno][4]=Duru1(0,0,10,30);
               R2[ogrno][4]=Duru2(0,0,10,30);break;
        case 1: R[ogrno][4]=Duru(10,30,30,50,beta);
               R1[ogrno][4]=Duru1(10,30,30,50);
               R2[ogrno][4]=Duru2(10,30,30,50);break;
        case 2: R[ogrno][4]=Duru(30,50,50,70,beta);
               R1[ogrno][4]=Duru1(30,50,50,70);
               R2[ogrno][4]=Duru2(30,50,50,70);break;
        case 3: R[ogrno][4]=Duru(50,70,70,90,beta);
               R1[ogrno][4]=Duru1(50,70,70,90);
               R2[ogrno][4]=Duru2(50,70,70,90);break;
        case 4: R[ogrno][4]=Duru(70,90,100,100,beta);
               R1[ogrno][4]=Duru1(70,90,100,100);
               R2[ogrno][4]=Duru2(70,90,100,100);break;
        }
}
}
//---------------------------------------------------------------------
double Duru(int a,int b,int c,int d,double beta)
{
double Sonuc1;
beta=Form1->Edit2->Text.ToDouble();
Sonuc1=beta*(d-((qq+1.0)/(qq+2.0))*(d-c))+(1-beta)*(a+((qq+1.0)/(qq+2.0))*(b-a));
return Sonuc1;
}
```

```
//------------------------------------------------------------------
double Duru1(int a,int b,int c,int d)
{
double Sonuc2;
Sonuc2=(d-((qq+1.0)/(qq+2.0))*(d-c));
return Sonuc2;
}
//------------------------------------------------------------------
double Duru2(int a,int b,int c,int d)
{
double Sonuc3;
Sonuc3=(a+((qq+1.0)/(qq+2.0))*(b-a));
return Sonuc3;
}
//------------------------------------------------------------------
void __fastcall TForm1::Button3Click(TObject *Sender)
{
MAXOGR=Edit1->Text.ToInt();
Memo1->Lines->Clear();
beta=Form1->Edit2->Text.ToDouble();
{
sprintf(s,"Beta=%.4lf ------------------------",beta);
Memo1->Lines->Add(s);
for(i=0;i<MAXOGR;i++)
  {
  char s1[20],s[100];
  strcpy(s,"");
  for(j=0;j<5;j++)
    {
    sprintf(s1,"%6.2lf ",R[i][j]);
    strcat(s,s1);
    }
    sprintf(s1,"%7.2lf ",A[i]);
    strcat(s,s1);
    Memo1->Lines->Add(s);
  }
  Memo1->Lines->Add("");
}
}
//------------------------------------------------------------------
void __fastcall TForm1::Button4Click(TObject *Sender)
{
Memo1->Lines->SaveToFile("c:\\RRRMemolines.txt");
}
//------------------------------------------------------------------
void __fastcall TForm1::Button2Click(TObject *Sender)
{
int ki=0;
int flag=0;

Memo1->Lines->Clear();

l1:
beta=Form1->Edit2->Text.ToDouble();
nbeta1=beta;
do
{
Button6->Click();
sprintf(s,"===============");
Memo1->Lines->Add(s);
sprintf(s," %d. ITERATION",++ki);
Memo1->Lines->Add(s);
sprintf(s,"===============");
Memo1->Lines->Add(s);
nbeta0=nbeta1;
XX[ki]=nbeta0;
sprintf(s,"BETA = %.4lf ",nbeta0);
Memo1->Lines->Add(s);

char s[100],s1[100];
double e[m]= {1.0,1.0,1.0,1.0,1.0};

for(i=0;i<m;i++)
  {
```

```
  G[i][j]=0.0;
  for(j=0;j<m;j++)
      {
      for(p=0;p<n;p++)
      G[i][j]=G[i][j]+R[p][i]*R[p][j];
      }
  }

for(i=0;i<m;i++)
  {
  et[i]=e[i];
  }

for(i=0;i<m;i++)
  {
  for(j=0;j<m;j++)
      {
      GI[i][j]=G[i][j];
      }
  }

h=m*2;
n1=m;

for(i=0;i<m;i++)
  {
  for(j=n1;j<h;j++)
      {
      j1=j-i;
      if(j1-m<0) GI[i][j]=0;
      if(j1-m==0) GI[i][j]=1;
      }
  }

for(k=0;k<m;k++)
  {
  k1=k;
  k3=k+m;

  uc:
  if(GI[k][k]!=0) goto bir;
  i1=k;

  if(GI[i1][k]!=0) goto iki;

goto uc;

bir:
  for(j=0;j<h;j++)
  GI[k][j]=GI[k][j]+GI[i1][j];

iki:
  for(j=0;j<h;j++)
  GI[k+m][j]=GI[k][j]/GI[k][k];

    for(i=k1;i<k3;i++)
      {
      t=GI[i][k];
      for(j=0;j<h;j++)
      GI[i][j]=GI[i][j]-GI[k+m][j]*t;
      }
  }

for(i=0;i<m;i++)
  {
  for(j=0;j<m;j++)
      {
      GI[i][j]=GI[i+m][j+m];
      }
  }
for(i=0;i<m;i++)
  {
  d[i]=0.0;
  for(p=0;p<m;p++)
  d[i]=d[i]+GI[i][p]*e[p];
  }

for(j=0;j<m;j++)
  {
  f[j]=0.0;
  for(p=0;p<m;p++)
  f[j]=f[j]+et[p]*GI[p][j];
  }

for(i=0;i<m;i++)
  {
  g[i]=0.0;
  for(p=0;p<n;p++)
  g[i]=g[i]+R[p][i]*A[p];
  }

for(i=0;i<1;i++)
  {
  fg=0.0;
  for(p=0;p<m;p++)
  fg=fg+f[p]*g[p];
  }

v=1.0-fg;

for(i=0;i<m;i++)
  {
  o[i]=0.0;
  o[i]=o[i]+d[i]*v;
  }

GIT=0.0;
for(i=0;i<m;i++)
  {
  for(j=0;j<m;j++)
  GIT=GIT+GI[i][j];
  }

for(j=0;j<m;j++)
  Z[j]=1.0*o[j]/GIT;

for(j=0;j<m;j++)
  {
  c[j]=0.0;
  for(p=0;p<m;p++)
  c[j]=c[j]+GI[j][p]*g[p];
  }

for(j=0;j<m;j++)
  {
  w[j]=Z[j]+c[j];
  }
Memo1->Lines->Add("");
sprintf(s,"----w[j]----");
Memo1->Lines->Add(s);
for(j=0;j<m;j++)
  {
  sprintf(s,"   %1.5f",w[j]);
  Memo1->Lines->Add(s);
  }

for(i=0;i<n;i++)
  {
  tahmin[i]=0.0;
  for(p=0;p<n;p++)
  tahmin[i]=tahmin[i]+R[i][p]*w[p];
  }

for(i=0;i<n;i++)
  po[i]=pow(A[i]-tahmin[i],2);

sprintf(s,"");
Memo1->Lines->Add(s);
```

```
pot=0.0;
for(i=0;i<n;i++) pot+=po[i]; mem[ki]=pot;
sprintf(s,"SSE= %.4lf",pot);
Memo1->Lines->Add(s);

pay4=0.0;
payda2=0.0;

for(i=0;i<n;i++)
  {
  payda1=pay1=pay2=pay3=0.0;
  for(j=0;j<m;j++)
  pay1=pay1+(R2[i][j]*w[j]);
  pay2=pay1-A[i];
  for(j=0;j<m;j++)
  pay3=pay3+w[j]*(R1[i][j]-R2[i][j]);
  pay4=pay4+pay2*pay3;
  for(j=0;j<m;j++)
  payda1=payda1+w[j]*(R1[i][j]-R2[i][j]);
  payda2=payda2+pow(payda1,2);
  }

nbeta1=-1.0*pay4/payda2;
Memo1->Lines->Add("");

{
Form1->Edit2->Text=nbeta1;
Form2->Edit2->Text=nbeta1;
}
Form2->Edit4->Text=w[0];
Form2->Edit5->Text=w[1];
Form2->Edit6->Text=w[2];
Form2->Edit7->Text=w[3];
Form2->Edit8->Text=w[4];

for(i=0;i<m;i++)
for(j=0;j<m*2;j++)
G[i][j]=0.0;
Memo1->Lines->Add("");
}

while((flag==0) && (fabs(nbeta0-nbeta1)>0.000001));
if(flag==0){
  if(nbeta1<0.0) nbeta1=0.0;
  if(nbeta1>1.0) nbeta1=1.0;
  Form1->Edit2->Text=nbeta1;
  Form2->Edit2->Text=nbeta1;
  flag=1; goto l1;
}
Form1->Edit2->Text=nbeta0;
Form2->Edit2->Text=nbeta0;
}
//---------------------------------------------------------------------
void __fastcall TForm1::Button5Click(TObject *Sender)
{
FILE *ff=fopen("c:\\RRR.txt","w");
fclose(ff);
}
//---------------------------------------------------------------------
void __fastcall TForm1::Button6Click(TObject *Sender)
{
FILE *ff=fopen("c:\\RRR.txt","r");
int k=0,k1,k2,k3,k4,k5;
while(!feof(ff)){
  fscanf(ff,"%d",&ogrno);
  fscanf(ff,"%lf %d %d %d %d %d",&A[ogrno],&k1,&k2,&k3,&k4,&k5);
  RadioGroup1->ItemIndex=k1;
  RadioGroup2->ItemIndex=k2;
  RadioGroup3->ItemIndex=k3;
  RadioGroup4->ItemIndex=k4;
  RadioGroup5->ItemIndex=k5;
  k++;
  dosya=true;
  Button1->Click();
```

```
}
fclose(ff);
Edit1->Text=k;
MAXOGR=k;
}
//-------------------------------------------------------------------
void __fastcall TForm1::FormCreate(TObject *Sender)
{
dosya=false;
}
//-------------------------------------------------------------------
void __fastcall TForm1::Button7Click(TObject *Sender)
{
FILE *ff=fopen("c:\\RRRresult.txt","w");
if(!dosya){
  fprintf(ff,"%2.5f %2.5f %2.5f %2.5f %2.5f %2.5f",nbeta1,w[0],w[1],w[2],w[3],w[4]);
  fclose(ff);
}
dosya=false;
}
//-------------------------------------------------------------------
void __fastcall TForm1::Button8Click(TObject *Sender)
{
Form2->Show();
}
//-------------------------------------------------------------------
void __fastcall TForm1::Button9Click(TObject *Sender)
{
Form3->Show();
}
//-------------------------------------------------------------------
```

Golden Section Unit

```
//-------------------------------------------------------------------
#include <vcl.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
#pragma hdrstop
#include "OED_GS.h"
#include "Unit2_GS.h"
int n1,h,z,p,k,k1,k3,i2,i1,j2,j1,ogrno,MAXOGR1,y;
double x,yy,X,Y,v,t,pot,po[100],R3[100][5],G[100][100],GI[100][100],e[100],et[100];
double
A1[100],c[100],w[100],d[100],f[100],g[100],fg,o[100],Z[100],GIT,tahmin[100],beta;
double Duru(int a,int b,int c,int d,double beta);
char s1[20],s[100];
bool dosya=false;
double nbeta0,nbeta1,pay1,pay2,pay3,pay4,payda1,payda2;
AnsiString NAME[100];
#define m 5
#define qq 10
//-------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "CSPIN"
#pragma resource "*.dfm"
TForm2 *Form2;
//-------------------------------------------------------------------
__fastcall TForm2::TForm2(TComponent* Owner)
        : TForm(Owner)
{
}
//-------------------------------------------------------------------
void __fastcall TForm2::Button1Click(TObject *Sender)
{
w[0]=Form2->Edit4->Text.ToDouble();
w[1]=Form2->Edit5->Text.ToDouble();
w[2]=Form2->Edit6->Text.ToDouble();
w[3]=Form2->Edit7->Text.ToDouble();
w[4]=Form2->Edit8->Text.ToDouble();
MAXOGR1=Form2->Edit1->Text.ToInt();

FILE *ff=fopen("c:\\RRR1.txt","a");
if(!dosya)
{
  ogrno=Form2->CSpinEdit1->Value;
  fprintf(ff,"%d %d %d %d %d %d %5s\n",ogrno,
   RadioGroup1->ItemIndex,
   RadioGroup2->ItemIndex,
   RadioGroup3->ItemIndex,
   RadioGroup4->ItemIndex,
   RadioGroup5->ItemIndex,
   NAME[ogrno]=Edit9->Text);
  fclose(ff);
}
dosya=false;
{
switch(RadioGroup1->ItemIndex)
        {
        case 0: R3[ogrno][0]=Duru(0,0,10,50,beta);break;
        case 1: R3[ogrno][0]=Duru(10,50,50,90,beta);break;
        case 2: R3[ogrno][0]=Duru(50,90,100,100,beta);break;
        }
switch(RadioGroup2->ItemIndex)
        {
        case 0: R3[ogrno][1]=Duru(0,0,10,50,beta);break;
        case 1: R3[ogrno][1]=Duru(10,50,50,90,beta);break;
        case 2: R3[ogrno][1]=Duru(50,90,100,100,beta);break;
        }
switch(RadioGroup3->ItemIndex)
        {
        case 0: R3[ogrno][2]=Duru(0,0,10,30,beta);break;
        case 1: R3[ogrno][2]=Duru(10,30,30,50,beta);break;
        case 2: R3[ogrno][2]=Duru(30,50,50,70,beta);break;
        case 3: R3[ogrno][2]=Duru(50,70,70,90,beta);break;
        case 4: R3[ogrno][2]=Duru(70,90,100,100,beta);break;
        }
switch(RadioGroup4->ItemIndex)
        {
        case 0: R3[ogrno][3]=Duru(0,0,10,30,beta);break;
        case 1: R3[ogrno][3]=Duru(10,30,30,50,beta);break;
        case 2: R3[ogrno][3]=Duru(30,50,50,70,beta);break;
        case 3: R3[ogrno][3]=Duru(50,70,70,90,beta);break;
        case 4: R3[ogrno][3]=Duru(70,90,100,100,beta);break;
        }
switch(RadioGroup5->ItemIndex)
        {
        case 0: R3[ogrno][4]=Duru(0,0,10,30,beta);break;
        case 1: R3[ogrno][4]=Duru(10,30,30,50,beta);break;
        case 2: R3[ogrno][4]=Duru(30,50,50,70,beta);break;
        case 3: R3[ogrno][4]=Duru(50,70,70,90,beta);break;
        case 4: R3[ogrno][4]=Duru(70,90,100,100,beta);break;
        }
}

for(i2=0;i2<MAXOGR1;i2++)
  {
  A1[i2]=0.0;
  for(j2=0;j2<m;j2++)
  A1[i2]=A1[i2]+R3[i2][j2]*w[j2];
  }
if (A1[ogrno]<0.0)
A1[ogrno]=0.0;
if(A1[ogrno]>100.0)
A1[ogrno]=100.0;
Form2->Edit3->Text=A1[ogrno];
}
//---------------------------------------------------------------------
double Duru(int a,int b,int c,int d,double beta)
{
double Sonuc1;
beta=Form2->Edit2->Text.ToDouble();
Sonuc1=beta*(d-((qq+1.0)/(qq+2.0))*(d-c))+(1-beta)*(a+((qq+1.0)/(qq+2.0))*(b-a));
return Sonuc1;
}
//---------------------------------------------------------------------
void __fastcall TForm2::Button3Click(TObject *Sender)
{
MAXOGR1=Form2->Edit1->Text.ToInt();
Memo1->Lines->Clear();
beta=Form2->Edit2->Text.ToDouble();

sprintf(s,"      NAME-SURNAME       GRADE");
Memo1->Lines->Add(s);
sprintf(s,"===============================");
Memo1->Lines->Add(s);


for(i2=0;i2<MAXOGR1;i2++)
  {
  char s1[20],s[100];
  strcpy(s,"");
    {
    sprintf(s1,"%19s   %7.0f",NAME[i2],A1[i2]);
    strcat(s,s1);
    }
    Memo1->Lines->Add(s);
  }
  Memo1->Lines->Add("");
}
//---------------------------------------------------------------------
```

Grade Evaluation Unit

```
//---------------------------------------------------------------------------
#include <vcl.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
#pragma hdrstop
#include "Unit3_GS.h"
int n1,h,z,p,k,k1,k3,i,i1,j,j1,ogrno,MAXOGR;
double
a1,b1,v,t,pot,mem[100],po[100],R[100][5],R1[100][5],G[100][100],GI[100][100],e[100],e
t[100];
double
A[100],c[100],w[100],d[100],f[100],g[100],fg,o[100],Z[100],GIT,tahmin[100],beta;
double nbeta0,nbeta1,beta1,beta2;
double Duru2(int a,int b,int c,int d,double beta1);
double Duru1(int a,int b,int c,int d,double beta2);
char s1[20],s[100];
bool dosya=false;
double v1,t1,pot1,mem1[100],po1[100],G1[100][100],GI1[100][100],e1[100],et1[100];
double c1[100],w1[100],d1[100],f1[100],g1[100],fg1,o1[100],Z1[100],GIT1,tahmin1[100];
#define m 5
#define n 100
#define qq 10
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "CSPIN"
#pragma resource "*.dfm"
TForm3 *Form3;
//---------------------------------------------------------------------------
__fastcall TForm3::TForm3(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TForm3::Button1Click(TObject *Sender)
{
FILE *ff=fopen("c:\\RRR.txt","a");
if(!dosya)
{
  MAXOGR=Edit1->Text.ToInt();
  ogrno=CSpinEdit1->Value;
  A[ogrno]=Edit3->Text.ToDouble();
  fprintf(ff,"%d %6.2lf %d %d %d %d %d\n",ogrno,A[ogrno],
   RadioGroup1->ItemIndex,
   RadioGroup2->ItemIndex,
   RadioGroup3->ItemIndex,
   RadioGroup4->ItemIndex,
   RadioGroup5->ItemIndex);
  fclose(ff);
}
dosya=false;
{
switch(RadioGroup1->ItemIndex)
        {
        case 0: R[ogrno][0]=Duru2(0,0,10,50,beta1);
               R1[ogrno][0]=Duru1(0,0,10,50,beta2);break;
        case 1: R[ogrno][0]=Duru2(10,50,50,90,beta1);
               R1[ogrno][0]=Duru1(10,50,50,90,beta2);break;
        case 2: R[ogrno][0]=Duru2(50,90,100,100,beta1);
               R1[ogrno][0]=Duru1(50,90,100,100,beta2);break;
        }
switch(RadioGroup2->ItemIndex)
        {
        case 0: R[ogrno][1]=Duru2(0,0,10,50,beta1);
               R1[ogrno][1]=Duru1(0,0,10,50,beta2);break;
        case 1: R[ogrno][1]=Duru2(10,50,50,90,beta1);
               R1[ogrno][1]=Duru1(10,50,50,90,beta2);break;
        case 2: R[ogrno][1]=Duru2(50,90,100,100,beta1);
               R1[ogrno][1]=Duru1(50,90,100,100,beta2);break;
        }
switch(RadioGroup3->ItemIndex)
        {
```

```
        case 0: R[ogrno][2]=Duru2(0,0,10,30,beta1);
               R1[ogrno][2]=Duru1(0,0,10,30,beta2);break;
        case 1: R[ogrno][2]=Duru2(10,30,30,50,beta1);
               R1[ogrno][2]=Duru1(10,30,30,50,beta2);break;
        case 2: R[ogrno][2]=Duru2(30,50,50,70,beta1);
               R1[ogrno][2]=Duru1(30,50,50,70,beta2);break;
        case 3: R[ogrno][2]=Duru2(50,70,70,90,beta1);
               R1[ogrno][2]=Duru1(50,70,70,90,beta2);break;
        case 4: R[ogrno][2]=Duru2(70,90,100,100,beta1);
               R1[ogrno][2]=Duru1(70,90,100,100,beta2);break;
        }
switch(RadioGroup4->ItemIndex)
        {
        case 0: R[ogrno][3]=Duru2(0,0,10,30,beta1);
               R1[ogrno][3]=Duru1(0,0,10,30,beta2);break;
        case 1: R[ogrno][3]=Duru2(10,30,30,50,beta1);
               R1[ogrno][3]=Duru1(10,30,30,50,beta2);break;
        case 2: R[ogrno][3]=Duru2(30,50,50,70,beta1);
               R1[ogrno][3]=Duru1(30,50,50,70,beta2);break;
        case 3: R[ogrno][3]=Duru2(50,70,70,90,beta1);
               R1[ogrno][3]=Duru1(50,70,70,90,beta2);break;
        case 4: R[ogrno][3]=Duru2(70,90,100,100,beta1);
               R1[ogrno][3]=Duru1(70,90,100,100,beta2);break;
        }
switch(RadioGroup5->ItemIndex)
        {
        case 0: R[ogrno][4]=Duru2(0,0,10,30,beta1);
               R1[ogrno][4]=Duru1(0,0,10,30,beta2);break;
        case 1: R[ogrno][4]=Duru2(10,30,30,50,beta1);
               R1[ogrno][4]=Duru1(10,30,30,50,beta2);break;
        case 2: R[ogrno][4]=Duru2(30,50,50,70,beta1);
               R1[ogrno][4]=Duru1(30,50,50,70,beta2);break;
        case 3: R[ogrno][4]=Duru2(50,70,70,90,beta1);
               R1[ogrno][4]=Duru1(50,70,70,90,beta2);break;
        case 4: R[ogrno][4]=Duru2(70,90,100,100,beta1);
               R1[ogrno][4]=Duru1(70,90,100,100,beta2);break;
        }
}
}
//---------------------------------------------------------------------
double Duru2(int a,int b,int c,int d,double beta1)
{
double Sonuc1;
beta1=Form3->Edit2->Text.ToDouble();
Sonuc1=beta1*(d-((qq+1.0)/(qq+2.0))*(d-c))+(1-beta1)*(a+((qq+1.0)/(qq+2.0))*(b-a));
return Sonuc1;
}
//---------------------------------------------------------------------
double Duru1(int a,int b,int c,int d,double beta2)
{
double Sonuc2;
beta2=Form3->Edit4->Text.ToDouble();
Sonuc2=beta2*(d-((qq+1.0)/(qq+2.0))*(d-c))+(1-beta2)*(a+((qq+1.0)/(qq+2.0))*(b-a));
return Sonuc2;
}
//---------------------------------------------------------------------
void __fastcall TForm3::Button6Click(TObject *Sender)
{
FILE *ff=fopen("c:\\RRR.txt","r");
int k=0,k1,k2,k3,k4,k5;
while(!feof(ff))
{
  fscanf(ff,"%d",&ogrno);
  fscanf(ff,"%lf %d %d %d %d %d",&A[ogrno],&k1,&k2,&k3,&k4,&k5);
  RadioGroup1->ItemIndex=k1;
  RadioGroup2->ItemIndex=k2;
  RadioGroup3->ItemIndex=k3;
  RadioGroup4->ItemIndex=k4;
  RadioGroup5->ItemIndex=k5;
  k++;
  dosya=true;
  Button1->Click();
}
fclose(ff);
```

```
Edit1->Text=k;
MAXOGR=k;
}
//---------------------------------------------------------------------
void __fastcall TForm3::FormCreate(TObject *Sender)
{
dosya=false;
}
//---------------------------------------------------------------------
void __fastcall TForm3::Button9Click(TObject *Sender)
{
Memo1->Lines->Clear();
sprintf(s,"Golden Section Solution:");
Memo1->Lines->Add(s);
sprintf(s,"==================================================================");
Memo1->Lines->Add(s);
sprintf(s,"Iter.   a         b       lamda       mu       SSE1      SSE2");
Memo1->Lines->Add(s);
sprintf(s,"==================================================================");
Memo1->Lines->Add(s);
double a2=0.0,b2=1.0;
int ki=1;
char s[100],s1[100];
beta1=a2+0.382*fabs(b2-a2);
beta2=a2+0.618*fabs(b2-a2);

do
{
double e[m]= {1.0,1.0,1.0,1.0,1.0};

for(i=0;i<m;i++)
  {
  G[i][j]=0.0;
  for(j=0;j<m;j++)
      {
      for(p=0;p<n;p++)
      G[i][j]=G[i][j]+R[p][i]*R[p][j];
      }
  }

for(i=0;i<m;i++)
  et[i]=e[i];

for(i=0;i<m;i++)
  {
  for(j=0;j<m;j++)
      {
      GI[i][j]=G[i][j];
      }
  }

h=m*2;
n1=m;

for(i=0;i<m;i++)
  {
  for(j=n1;j<h;j++)
      {
      j1=j-i;
      if(j1-m<0) GI[i][j]=0;
      if(j1-m==0) GI[i][j]=1;
      }
  }

for(k=0;k<m;k++)
  {
  k1=k;
  k3=k+m;

  uc:
  if(GI[k][k]!=0) goto bir;
  i1=k;

  if(GI[i1][k]!=0) goto iki;
goto uc;

bir:
  for(j=0;j<h;j++)
  GI[k][j]=GI[k][j]+GI[i1][j];

iki:
  for(j=0;j<h;j++)
  GI[k+m][j]=GI[k][j]/GI[k][k];

    for(i=k1;i<k3;i++)
      {
      t=GI[i][k];
      for(j=0;j<h;j++)
      GI[i][j]=GI[i][j]-GI[k+m][j]*t;
      }
  }

for(i=0;i<m;i++)
  {
  for(j=0;j<m;j++)
      {
      GI[i][j]=GI[i+m][j+m];
      }
  }

for(i=0;i<m;i++)
  {
  d[i]=0.0;
  for(p=0;p<m;p++)
  d[i]=d[i]+GI[i][p]*e[p];
  }

for(j=0;j<m;j++)
  {
  f[j]=0.0;
  for(p=0;p<m;p++)
  f[j]=f[j]+et[p]*GI[p][j];
  }

for(i=0;i<m;i++)
  {
  g[i]=0.0;
  for(p=0;p<n;p++)
  g[i]=g[i]+R[p][i]*A[p];
  }

for(i=0;i<1;i++)
  {
  fg=0.0;
  for(p=0;p<m;p++)
  fg=fg+f[p]*g[p];
  }

v=1.0-fg;

for(i=0;i<m;i++)
  {
  o[i]=0.0;
  o[i]=o[i]+d[i]*v;
  }

GIT=0.0;
for(i=0;i<m;i++)
  {
  for(j=0;j<m;j++)
  GIT=GIT+GI[i][j];
  }

for(j=0;j<m;j++)
  Z[j]=1.0*o[j]/GIT;


for(j=0;j<m;j++)
```

```
  {
  c[j]=0.0;
  for(p=0;p<m;p++)
  c[j]=c[j]+GI[j][p]*g[p];
  }

for(j=0;j<m;j++)
  {
  w[j]=Z[j]+c[j];
  }

for(i=0;i<n;i++)
  {
  tahmin[i]=0.0;
  for(p=0;p<n;p++)
  tahmin[i]=tahmin[i]+R[i][p]*w[p];
  }

for(i=0;i<n;i++)
  {
  po[i]=pow(tahmin[i]-A[i],2);
  }

pot=0.0;
for(i=0;i<n;i++) pot+=po[i]; mem[ki]=pot;

double e1[m]= {1.0,1.0,1.0,1.0,1.0};

for(i=0;i<m;i++)
  {
  G1[i][j]=0.0;
  for(j=0;j<m;j++)
      {
      for(p=0;p<n;p++)
      G1[i][j]=G1[i][j]+R1[p][i]*R1[p][j];
      }
  }

for(i=0;i<m;i++)
  et1[i]=e1[i];

for(i=0;i<m;i++)
  {
  for(j=0;j<m;j++)
      {
      GI1[i][j]=G1[i][j];
      }
  }

h=m*2;
n1=m;

for(i=0;i<m;i++)
  {
  for(j=n1;j<h;j++)
      {
      j1=j-i;
      if(j1-m<0) GI1[i][j]=0;
      if(j1-m==0) GI1[i][j]=1;
      }
  }

for(k=0;k<m;k++)
  {
  k1=k;
  k3=k+m;

  alti:
  if(GI1[k][k]!=0) goto dort;
  i1=k;

  if(GI1[i1][k]!=0) goto bes;

goto alti;
```

```
dort:
  for(j=0;j<h;j++)
  GI1[k][j]=GI1[k][j]+GI1[i1][j];

bes:
  for(j=0;j<h;j++)
  GI1[k+m][j]=GI1[k][j]/GI1[k][k];

    for(i=k1;i<k3;i++)
      {
      t1=GI1[i][k];
      for(j=0;j<h;j++)
      GI1[i][j]=GI1[i][j]-GI1[k+m][j]*t1;
      }
  }

for(i=0;i<m;i++)
  {
  for(j=0;j<m;j++)
      {
      GI1[i][j]=GI1[i+m][j+m];
      }
  }

for(i=0;i<m;i++)
  {
  d1[i]=0.0;
  for(p=0;p<m;p++)
  d1[i]=d1[i]+GI1[i][p]*e1[p];
  }

for(j=0;j<m;j++)
  {
  f1[j]=0.0;
  for(p=0;p<m;p++)
  f1[j]=f1[j]+et1[p]*GI1[p][j];
  }

for(i=0;i<m;i++)
  {
  g1[i]=0.0;
  for(p=0;p<n;p++)
  g1[i]=g1[i]+R1[p][i]*A[p];
  }

for(i=0;i<1;i++)
  {
  fg1=0.0;
  for(p=0;p<m;p++)
  fg1=fg1+f1[p]*g1[p];
  }

v1=1.0-fg1;

for(i=0;i<m;i++)
  {
  o1[i]=0.0;
  o1[i]=o1[i]+d1[i]*v1;
  }

GIT1=0.0;
for(i=0;i<m;i++)
  {
  for(j=0;j<m;j++)
  GIT1=GIT1+GI1[i][j];
  }

for(j=0;j<m;j++)
  Z1[j]=1.0*o1[j]/GIT1;

for(j=0;j<m;j++)
  {
  c1[j]=0.0;
  for(p=0;p<m;p++)
  c1[j]=c1[j]+GI1[j][p]*g1[p];
  }


for(j=0;j<m;j++)
  {
  w1[j]=Z1[j]+c1[j];
  }

for(i=0;i<n;i++)
  {
  tahmin1[i]=0.0;
  for(p=0;p<n;p++)
  tahmin1[i]=tahmin1[i]+R1[i][p]*w1[p];
  }

for(i=0;i<n;i++)
  {
  po1[i]=pow(tahmin1[i]-A[i],2);
  }

pot1=0.0;
for(i=0;i<n;i++) pot1+=po1[i]; mem1[ki]=pot1;


for(i=0;i<m;i++)
  for(j=0;j<m*2;j++)
  {
  G[i][j]=0.0;
  G1[i][j]=0.0;
  }
/*************************GOLDEN SECTION**********************/

sprintf(s,"%2d  %8.4lf  %8.4lf  %8.4lf  %8.4lf  %8.4lf
%8.4lf",ki,a2,b2,beta1,beta2,mem[ki],mem1[ki]);
Memo1->Lines->Add(s);

  if(mem[ki]>mem1[ki])
       a2=beta1;
  else b2=beta2;

beta1=a2+0.382*fabs(b2-a2);
beta2=a2+0.618*fabs(b2-a2);

Form3->Edit2->Text=beta1;
Form3->Edit4->Text=beta2;
beta1=Form3->Edit2->Text.ToDouble();
beta2=Form3->Edit4->Text.ToDouble();

mem[ki]=mem1[ki]=0.0;
ki++;
Button6->Click();
}
while(fabs(b2-a2)>0.000001);
sprintf(s,"==================================================================");
Memo1->Lines->Add(s);
sprintf(s,"Uncertainty interval of function is [%5.4lf,%5.4lf].",a2,b2);
Memo1->Lines->Add(s);
sprintf(s,"Interval midpoint is %5.4lf.",0.5*(a2+b2));
Memo1->Lines->Add(s);
}
//-------------------------------------------------------------------
```

APPENDIX C – Borland C++ Builder 6.0 Code for Group Constitution

```
//-------------------------------------------------------------------
#include <vcl.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <stdlib.h>
#include "Unit1.h"
#include "Unit2.h"
#define MAX(a,b) (((a)>(b))?(a):(b))
#pragma hdrstop
int h=0,k=0,i,j,ogrno,MAXOGR,GROUPNO,a,b,c,d,e,f,x=0,xx=0,y=0,yy=0,g;
double k1,k2,k3,k4,k5,top[4][4],ort[4][4],ort1[4],SORT[4],SORT1[25][25][4],hkt[4][4],
po[200][200],var[4][4],A[100][4],A1[4][100][4],A2[4][100][4];
double RA[100],RAM[25][100][4],DA[4][100],DAM[25][25][4],DAMtop[25],DAMort[25],
SA[4][100],SAM[25][100][4],DAM1[25][100][4],SAM1[25][100][4];
double DAMstdev[25],DAMHKT[25];
char s1[20],s[100];
int t=0,m=0,l=0,p=0;
AnsiString AD[61]={
"Arda Can CANÇALAR","Samet ŞENOL","Murat UĞURLU","Bahadır AĞCA","Serdar
ÇORLULUOĞLU","Dinçer GÖKSÜLÜK","Serdar KUZU","A.Gökhan KÜÇÜKKATİPOĞLU","İ.Çağlar
PALAVAR","B. Kenan TELCİ","Sevil AKKAŞ","Sertaç AKSAKAL","Kenan ALIR","Esra
ALTINER","Tuğba ASLAN","H.Resul ÇAĞLAYAN","Selim ÇAM","Beyhan ÇOBAN","Hakan
EKİZ","Elif EMREM","Tamer EROL","Caner HATİPOĞLU","Gizem KAYA","Gökhan KAYMAK",
"Ender KILIÇ","Özgür KORKMAZ","Murat MANSUROĞLU","Sinem NALBANT","Cemile
ÖZDEMİR","Emre ÖZKUL","Ufuk SÖNMEZ","Onur SUBAŞI","Caner SUNGUR","Alper ŞAHİN","Aydın
ŞANAL","Simge TORTOP","Onur TÜYSÜZOĞLU","Belma YIKILMAZ","Eylül YILDIRIM","Merve
AKDEDE","Ufuk AKTAŞ","Servet ARSLAN","Erkan ASLANEL","Emel ÇİNKAYA","Alev
DOĞAN","Onur ELALMAZ","Başak ERDUR","Denizhan GÖNEN","İlknur GÜMÜŞBAŞ","Fatih
GÜRSOY","Sırma KAYITKEN","Raşit KORUMAZ","Can Ceki LEVİ","Ayşegül ÖNER","Şeyda
SOFUOĞLU","Fatma SOĞANLI","Esra SOYLU","Şeyma TEKİN","Ümit TÜNALP","Seher
VATANSEVER","Gülçin YANAR"};
//-------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//-------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
DecimalSeparator='.';
}
//-------------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
int k=0;
t=0;m=0;l=0;p=0;
int x=0,xx=0,y=0,yy=0;
top[0][0]=top[1][0]=top[2][0]=top[3][0]=0;
hkt[0][0]=hkt[1][0]=hkt[2][0]=hkt[3][0]=0;
Form1->Memo1->Lines->Clear();
Form1->Memo2->Lines->Clear();
Form1->Memo3->Lines->Clear();
Form1->Memo4->Lines->Clear();
Form2->Memo1->Lines->Clear();


FILE *ff=fopen("c:\\clusters.txt","r");
while(!feof(ff)){
  fscanf(ff,"%lf %lf %lf %lf",&k1,&k2,&k3,&k4);
  A[k][0]=k1;
  A[k][1]=k2;
  A[k][2]=k3;
  A[k][3]=k4;
  k++;
}
fclose(ff);
Edit1->Text=k;
MAXOGR=k;
```

```
for(k=0;k<MAXOGR;k++)
   {
   if(A[k][1]==0)      {t++;A1[0][x][0]=A2[0][x][0]=A[k][0];
A1[0][x][1]=A2[0][x][1]=A[k][1]; A1[0][x][2]=A2[0][x][2]=A[k][2];
A1[0][x][3]=A2[0][x][3]=A[k][3]; x++;}
   else if(A[k][1]==1)
{m++;A1[1][xx][0]=A2[1][xx][0]=A[k][0];A1[1][xx][1]=A2[1][xx][1]=A[k][1];A1[1][xx][2]
=A2[1][xx][2]=A[k][2];A1[1][xx][3]=A2[1][xx][3]=A[k][3]; xx++;}
   else if(A[k][1]==2) {l++;A1[2][y][0]=A2[2][y][0]=A[k][0];
A1[2][y][1]=A2[2][y][1]=A[k][1]; A1[2][y][2]=A2[2][y][2]=A[k][2];
A1[2][y][3]=A2[2][y][3]=A[k][3]; y++;}
   else
{p++;A1[3][yy][0]=A2[3][yy][0]=A[k][0];A1[3][yy][1]=A2[3][yy][1]=A[k][1];A1[3][yy][2]
=A2[3][yy][2]=A[k][2];A1[3][yy][3]=A2[3][yy][3]=A[k][3]; yy++;}
   }

k=0,j=0,a=0;
int z=0;
double enbuyuk=0;
int kk=0;

ZZ1:    for(i=0;i<t;i++)
        if(A2[0][i][0]!=0) {enbuyuk=MAX(enbuyuk,A2[0][i][3]);}
        kk=1;
        for(i=0;i<t;i++)
          if (A2[0][i][0]!=0 && A2[0][i][3]==enbuyuk && kk==1) {kk++;
SORT1[0][k][0]=A2[0][i][0]; SORT1[0][k][1]=A2[0][i][1]; SORT1[0][k][2]=A2[0][i][2];
SORT1[0][k][3]=A2[0][i][3]; A2[0][i][0]=A2[0][i][1]=A2[0][i][2]=A2[0][i][3]=0;}
          {
          enbuyuk=0;
          k++;
          kk=0;
          if(i!=k) goto ZZ1;
          }

enbuyuk=0;
k=0;
kk=0;
ZZZ1:   for(i=0;i<m;i++)
        if(A2[1][i][0]!=0) {enbuyuk=MAX(enbuyuk,A2[1][i][3]);}
        kk=1;
        for(i=0;i<m;i++)
          if (A2[1][i][0]!=0 && A2[1][i][3]==enbuyuk && kk==1) {kk++;
SORT1[1][k][0]=A2[1][i][0]; SORT1[1][k][1]=A2[1][i][1]; SORT1[1][k][2]=A2[1][i][2];
SORT1[1][k][3]=A2[1][i][3]; A2[1][i][0]=A2[1][i][1]=A2[1][i][2]=A2[1][i][3]=0;}
          {
          enbuyuk=0;
          k++;
          kk=0;
          if(i!=k) goto ZZZ1;
          }

enbuyuk=0;
k=0;
kk=0;
ZZZZ1:  for(i=0;i<l;i++)
        if(A2[2][i][0]!=0) {enbuyuk=MAX(enbuyuk,A2[2][i][3]);}
        kk=1;
        for(i=0;i<l;i++)
          if (A2[2][i][0]!=0 && A2[2][i][3]==enbuyuk && kk==1) {kk++;
SORT1[2][k][0]=A2[2][i][0]; SORT1[2][k][1]=A2[2][i][1]; SORT1[2][k][2]=A2[2][i][2];
SORT1[2][k][3]=A2[2][i][3]; A2[2][i][0]=A2[2][i][1]=A2[2][i][2]=A2[2][i][3]=0;}
          {
          enbuyuk=0;
          k++;
          kk=0;
          if(i!=k) goto ZZZZ1;
          }

enbuyuk=0;
k=0;
kk=0;
ZZZZZ1: for(i=0;i<p;i++)
        if(A2[3][i][0]!=0) {enbuyuk=MAX(enbuyuk,A2[3][i][3]);}
```

```
          kk=1;
          for(i=0;i<p;i++)
            if (A2[3][i][0]!=0 && A2[3][i][3]==enbuyuk && kk==1) {kk++;
SORT1[3][k][0]=A2[3][i][0]; SORT1[3][k][1]=A2[3][i][1]; SORT1[3][k][2]=A2[3][i][2];
SORT1[3][k][3]=A2[3][i][3]; A2[3][i][0]=A2[3][i][1]=A2[3][i][2]=A2[3][i][3]=0;}
            {
            enbuyuk=0;
            k++;
            kk=0;
            if(i!=k) goto ZZZZZ1;
            }

    char s1[20],s[100];
    strcpy(s,"");
    sprintf(s1,"      NAME-SURNAME          MEAN");
    strcat(s,s1);
    Form1->Memo1->Lines->Add(s);
    Form1->Memo2->Lines->Add(s);
    Form1->Memo3->Lines->Add(s);
    Form1->Memo4->Lines->Add(s);
    strcpy(s,"");
    sprintf(s1,"================================");
    strcat(s,s1);
    Form1->Memo1->Lines->Add(s);
    Form1->Memo2->Lines->Add(s);
    Form1->Memo3->Lines->Add(s);
    Form1->Memo4->Lines->Add(s);

for(k=0;k<t;k++)
    {
    top[0][0]+=SORT1[0][k][3];
    ort[0][0]=top[0][0]/t;
    }

for(k=0;k<m;k++)
    {
    top[1][0]+=SORT1[1][k][3];
    ort[1][0]=top[1][0]/m;
    }

for(k=0;k<l;k++)
    {
    top[2][0]+=SORT1[2][k][3];
    ort[2][0]=top[2][0]/l;
    }

for(k=0;k<p;k++)
    {
    top[3][0]+=SORT1[3][k][3];
    ort[3][0]=top[3][0]/p;
    }

for(k=0;k<MAXOGR;k++)
  {
    if (A[k][1]==0)
    {
    po[k][0]=pow((A[k][3]-ort[0][0]),2);
    hkt[0][0]+=po[k][0];
    }
    if (A[k][1]==1)
    {
    po[k][0]=pow((A[k][3]-ort[1][0]),2);
    hkt[1][0]+=po[k][0];
    }
    if (A[k][1]==2)
    {
    po[k][0]=pow((A[k][3]-ort[2][0]),2);
    hkt[2][0]+=po[k][0];
    }
    if (A[k][1]==3)
    {
    po[k][0]=pow((A[k][3]-ort[3][0]),2);
    hkt[3][0]+=po[k][0];
    }
  }

if(t==1) var[0][0]=hkt[0][0]/(t);
else var[0][0]=hkt[0][0]/(t-1);
if(m==1) var[1][0]=hkt[1][0]/(m);
else var[1][0]=hkt[1][0]/(m-1);
if(l==1)var[2][0]=hkt[2][0]/(l);
else var[2][0]=hkt[2][0]/(l-1);
if(p==1)var[3][0]=hkt[3][0]/(p);
else var[3][0]=hkt[3][0]/(p-1);

for(i=0;i<4;i++)
ort1[i]=ort[i][0];

k=0;
enbuyuk=0;

Z: for(i=0;i<4;i++)
   enbuyuk=MAX(enbuyuk,ort1[i]);
   for(i=0;i<4;i++)
   if (ort1[i]==enbuyuk) {ort1[i]=0;}
   {
   SORT[k]=enbuyuk;
   enbuyuk=0;
   k++;
   if(i!=k) goto Z;
   }

for(i=0;i<4;i++)
if (SORT[0]==ort[i][0])
     {
     h=0;
     k=0;
A11: char s1[20],s[100];
     strcpy(s,"");
     sprintf(s1,"%23s %7.1lf",AD[int(SORT1[i][k][0])-1],SORT1[i][k][3]);
     strcat(s,s1);
     Form1->Memo1->Lines->Add(s);
     A2[0][h][0]=SORT1[i][k][0];
     A2[0][h][1]=SORT1[i][k][1];
     A2[0][h][2]=SORT1[i][k][2];
     A2[0][h][3]=SORT1[i][k][3];
     h++;
     k++;
     if (SORT1[i][k][0]!=0) goto A11;
     strcpy(s,"");
     sprintf(s1,"===============================");
     strcat(s,s1);
     Form1->Memo1->Lines->Add(s);
     strcpy(s,"");
     sprintf(s1,"MEAN= %.2f",ort[i][0]);
     strcat(s,s1);
     Form1->Memo1->Lines->Add(s);
     strcpy(s,"");
     sprintf(s1,"STDEV= %.2lf ",sqrt(var[i][0]));
     strcat(s,s1);
     Form1->Memo1->Lines->Add(s);
     }
Edit3->Text=k;
t=k;

for(i=0;i<4;i++)
if (SORT[1]==ort[i][0])
     {
     k=0;
A12: char s1[20],s[100];
     strcpy(s,"");
     sprintf(s1,"%23s %7.1lf",AD[int(SORT1[i][k][0])-1],SORT1[i][k][3]);
     strcat(s,s1);
     Form1->Memo2->Lines->Add(s);
     A2[0][h][0]=SORT1[i][k][0];
     A2[0][h][1]=SORT1[i][k][1];
     A2[0][h][2]=SORT1[i][k][2];
     A2[0][h][3]=SORT1[i][k][3];
```

```
      h++;
      k++;
      if (SORT1[i][k][0]!=0) goto A12;
      strcpy(s,"");
      sprintf(s1,"================================");
      strcat(s,s1);
      Form1->Memo2->Lines->Add(s);
      strcpy(s,"");
      sprintf(s1,"MEAN= %.2f",ort[i][0]);
      strcat(s,s1);
      Form1->Memo2->Lines->Add(s);
      strcpy(s,"");
      sprintf(s1,"STDEV= %.2lf ",sqrt(var[i][0]));
      strcat(s,s1);
      Form1->Memo2->Lines->Add(s);
      }
Edit4->Text=k;
m=k;

for(i=0;i<4;i++)
if (SORT[2]==ort[i][0])
      {
      k=0;
A13:char s1[20],s[100];
      strcpy(s,"");
      sprintf(s1,"%23s %7.1lf",AD[int(SORT1[i][k][0])-1],SORT1[i][k][3]);
      strcat(s,s1);
      Form1->Memo3->Lines->Add(s);
      A2[0][h][0]=SORT1[i][k][0];
      A2[0][h][1]=SORT1[i][k][1];
      A2[0][h][2]=SORT1[i][k][2];
      A2[0][h][3]=SORT1[i][k][3];
      h++;
      k++;
      if (SORT1[i][k][0]!=0) goto A13;
      strcpy(s,"");
      sprintf(s1,"================================");
      strcat(s,s1);
      Form1->Memo3->Lines->Add(s);
      strcpy(s,"");
      sprintf(s1,"MEAN= %.2f",ort[i][0]);
      strcat(s,s1);
      Form1->Memo3->Lines->Add(s);
      strcpy(s,"");
      sprintf(s1,"STDEV= %.2lf ",sqrt(var[i][0]));
      strcat(s,s1);
      Form1->Memo3->Lines->Add(s);
      }
Edit5->Text=k;
l=k;

for(i=0;i<4;i++)
if (SORT[3]==ort[i][0])
      {
      k=0;
A14: char s1[20],s[100];
      strcpy(s,"");
      sprintf(s1,"%23s %7.1lf",AD[int(SORT1[i][k][0])-1],SORT1[i][k][3]);
      strcat(s,s1);
      Form1->Memo4->Lines->Add(s);
      A2[0][h][0]=SORT1[i][k][0];
      A2[0][h][1]=SORT1[i][k][1];
      A2[0][h][2]=SORT1[i][k][2];
      A2[0][h][3]=SORT1[i][k][3];
      h++;
      k++;
      if (SORT1[i][k][0]!=0) goto A14;
      strcpy(s,"");
      sprintf(s1,"================================");
      strcat(s,s1);
      Form1->Memo4->Lines->Add(s);
      strcpy(s,"");
      sprintf(s1,"MEAN= %.2f",ort[i][0]);
      strcat(s,s1);
```

```
     Form1->Memo4->Lines->Add(s);
     strcpy(s,"");
     sprintf(s1,"STDEV= %.2lf ",sqrt(var[i][0]));
     strcat(s,s1);
     Form1->Memo4->Lines->Add(s);
     }
Edit6->Text=k;
p=k;
}
//---------------------------------------------------------------------
void __fastcall TForm1::Button2Click(TObject *Sender)
{
Form1->Memo1->Lines->Clear();
Form1->Memo2->Lines->Clear();
Form1->Memo3->Lines->Clear();
Form1->Memo4->Lines->Clear();

GROUPNO=Edit2->Text.ToInt();
Button1->Click();
int z=(MAXOGR/GROUPNO)+1;
for(z=0;z<GROUPNO;z++)
DAMtop[z]=0.0;
DAMort[z]=0.0;

switch(RadioGroup1->ItemIndex)
{
case 0:{//BALANCED RANDOM ASSIGNMENT***************************************

for(int i=0;i<25;i++)
   for(int j=0;j<25;j++)
      for(int k=0;k<4;k++)
      DAM[i][j][k]=0.0;

 srand(time(NULL));
 for(i=0;i<t;i++)
    {
A:  DA[0][i]=1+rand()%t;
        for(k=0;k<i;k++)
          if(DA[0][i]==DA[0][k])
          goto A;
     }

   for(i=0;i<m;i++)
     {
B:   DA[1][i]=1+rand()%m;
        for(k=0;k<i;k++)
          if(DA[1][i]==DA[1][k])
          goto B;
     }

   for(i=0;i<l;i++)
     {
C:   DA[2][i]=1+rand()%l;
        for(k=0;k<i;k++)
          if(DA[2][i]==DA[2][k])
          goto C;
     }

   for(i=0;i<p;i++)
     {
D:   DA[3][i]=1+rand()%p;
        for(k=0;k<i;k++)
          if(DA[3][i]==DA[3][k])
          goto D;
     }

int k=0,j=0,a=0,z=0;

     for(i=1;i<=t;i++)
        {
E:        for(k=0;k<t;k++)

      if(DA[0][k]==i)
           {
           DAM[z][a][0]=A2[0][k][0];
           DAM[z][a][1]=A2[0][k][1];
           DAM[z][a][2]=A2[0][k][2];
           DAM[z][a][3]=A2[0][k][3];
           }
        DAMtop[z]+=DAM[z][a][3];
        z++;
        if(i==t) goto F;
        if(z==GROUPNO) {z=0; i++; a++;  goto E;}
        }

F:    if (z==GROUPNO) {z=0;a++;}
        for(i=1;i<=m;i++)
        {
G:        for(k=0;k<m;k++)

      if(DA[1][k]==i)
           {
           DAM[z][a][0]=A2[0][k+t][0];
           DAM[z][a][1]=A2[0][k+t][1];
           DAM[z][a][2]=A2[0][k+t][2];
           DAM[z][a][3]=A2[0][k+t][3];
           }
        DAMtop[z]+=DAM[z][a][3];
        z++;
        if(i==m) goto H;
        if(z==GROUPNO) {z=0;i++;a++;  goto G;}
        }

H:    if (z==GROUPNO) {z=0;a++;}
        for(i=1;i<=l;i++)
        {
I:        for(k=0;k<l;k++)

      if(DA[2][k]==i)
           {
           DAM[z][a][0]=A2[0][k+t+m][0];
           DAM[z][a][1]=A2[0][k+t+m][1];
           DAM[z][a][2]=A2[0][k+t+m][2];
           DAM[z][a][3]=A2[0][k+t+m][3];
           }
        DAMtop[z]+=DAM[z][a][3];
        z++;
        if(i==l) goto J;
        if(z==GROUPNO) {z=0;i++;a++;  goto I;}
        }

J:    if (z==GROUPNO) {z=0;a++;}
        for(i=1;i<=p;i++)
        {
K:        for(k=0;k<p;k++)

      if(DA[3][k]==i)
           {
           DAM[z][a][0]=A2[0][k+t+m+l][0];
           DAM[z][a][1]=A2[0][k+t+m+l][1];
           DAM[z][a][2]=A2[0][k+t+m+l][2];
           DAM[z][a][3]=A2[0][k+t+m+l][3];
           }
        DAMtop[z]+=DAM[z][a][3];
        z++;
        if(i==p) break;
        if(z==GROUPNO) {z=0;i++;a++;  goto K;}
        }

Form2->Show();
Form2->Label1->Caption="BALANCED RANDOM ASSIGNMENT";

for(k=0;k<GROUPNO;k++)
  {
  char s1[20],s[100];
  strcpy(s,"");
  sprintf(s1,"                  GROUP %d",k+1);
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);
  strcpy(s,"");
  sprintf(s1,"NO              NAME-SURNAME    CLUSTER   MEAN");
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);

  strcpy(s,"");
  sprintf(s1,"=============================================");
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);

     for(g=0;g<=a;g++)
       {
       if (DAM[k][g][0]!=0)
         {
         strcpy(s,"");
         sprintf(s1,"%2d   %23s   %5.0lf   %7.1lf",g+1,AD[int(DAM[k][g][0])-
1],DAM[k][g][1],DAM[k][g][3]);
         strcat(s,s1);
         Form2->Memo1->Lines->Add(s);
         }
       }
       strcpy(s,"");
       sprintf(s1,"=============================================");
       strcat(s,s1);
       Form2->Memo1->Lines->Add(s);
       if(DAM[k][g-1][0]==0) {g=g-1;}
       DAMort[k]=DAMtop[k]/g;

       DAMHKT[k]=0;
       DAMstdev[k]=0;

       for(g=0;g<=a;g++)
       if(DAM[k][g][3]!=0) {DAMHKT[k]+=pow((DAM[k][g][3]-DAMort[k]),2);}
       if(DAM[k][g-1][0]==0) {g=g-1;}
       DAMstdev[k]=DAMHKT[k]/(g-1);
       strcpy(s,"");
       sprintf(s1,"MEAN=%3.2lf   STDEV=%3.2lf ",DAMort[k],sqrt(DAMstdev[k]));
       strcat(s,s1);
       Form2->Memo1->Lines->Add(s);
       Form2->Memo1->Lines->Add("");
  }
break;
}

case 1:{//SIMPLE RANDOM ASSIGNMENT*****************************************

for(int i=0;i<25;i++)
   for(int j=0;j<25;j++)
      for(int k=0;k<4;k++)
      RAM[i][j][k]=0.0;

   srand(time(NULL));
   for(i=0;i<MAXOGR;i++)
        {
L:
        RA[i]=1+rand()%MAXOGR;
        for(k=0;k<i;k++)
          if(RA[i]==RA[k])
          goto L;

        strcpy(s,"");
        sprintf(s1," %.0lf   %.0lf ",A[i][0],RA[i]);
        strcat(s,s1);
        Memo1->Lines->Add(s);*/
        }

int k=0,j=0,a=0;

     for(i=0;i<MAXOGR;i++)
        {
M:      j=RA[i];
        RAM[k][a][0]=A[j-1][0];
        RAM[k][a][1]=A[j-1][1];
        RAM[k][a][2]=A[j-1][2];
        RAM[k][a][3]=A[j-1][3];
        DAMtop[k]+=RAM[k][a][3];
        k++;
        if(k==GROUPNO &&  i!=MAXOGR-1) {k=0;i++;a++; goto M;}
        if(k==GROUPNO &&  i==MAXOGR-1) break;
        }

Form2->Show();
Form2->Label1->Caption="SIMPLE RANDOM ASSIGNMENT";

for(k=0;k<GROUPNO;k++)
  {
  char s1[20],s[100];
  strcpy(s,"");
  sprintf(s1,"                  GROUP %d",k+1);
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);
  strcpy(s,"");
  sprintf(s1,"NO              NAME-SURNAME    CLUSTER   MEAN");
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);

  strcpy(s,"");
  sprintf(s1,"==============================================");
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);
z=(MAXOGR/GROUPNO)+1;
     for(a=0;a<z;a++)
       {
       if (RAM[k][a][0]!=0)
         {
         strcpy(s,"");
         sprintf(s1,"%2d   %23s   %5.0lf   %7.1lf",a+1,AD[int(RAM[k][a][0])-
1],RAM[k][a][1],RAM[k][a][3]);
         strcat(s,s1);
         Form2->Memo1->Lines->Add(s);
         }
       }
       strcpy(s,"");
       sprintf(s1,"==============================================");
       strcat(s,s1);
       Form2->Memo1->Lines->Add(s);
       if(RAM[k][a-1][0]==0) {a=a-1;}
       DAMort[k]=DAMtop[k]/a;

       DAMHKT[k]=0;
       DAMstdev[k]=0;

       for(a=0;a<z;a++)
       if(RAM[k][a][3]!=0) {DAMHKT[k]+=pow((RAM[k][a][3]-DAMort[k]),2);}
       if(RAM[k][a-1][0]==0) {a=a-1;}
       DAMstdev[k]=DAMHKT[k]/(a-1);
       strcpy(s,"");
       sprintf(s1,"MEAN=%3.2lf   STDEV=%3.2lf ",DAMort[k],sqrt(DAMstdev[k]));
       strcat(s,s1);
       Form2->Memo1->Lines->Add(s);
       Form2->Memo1->Lines->Add("");
  }
break;
}

case 2:{//LEVEL-BASED RANDOM ASSIGNMENT*****************************

for(int i=0;i<25;i++)
   for(int j=0;j<25;j++)
      for(int k=0;k<4;k++)
      SAM[i][j][k]=0.0;

 srand(time(NULL));
 for(i=0;i<t;i++)
     {
N:
     SA[0][i]=1+rand()%t;
```

```
        for(k=0;k<i;k++)
          if(SA[0][i]==SA[0][k])
          goto N;
     }

   for(i=0;i<m;i++)
     {
O:
     SA[1][i]=1+rand()%m;
        for(k=0;k<i;k++)
          if(SA[1][i]==SA[1][k])
          goto O;
     }

   for(i=0;i<l;i++)
     {
P:
     SA[2][i]=1+rand()%l;
        for(k=0;k<i;k++)
          if(SA[2][i]==SA[2][k])
          goto P;
     }

   for(i=0;i<p;i++)
     {
Q:
     SA[3][i]=1+rand()%p;
        for(k=0;k<i;k++)
          if(SA[3][i]==SA[3][k])
          goto Q;
     }

int k=0,j=0,a=0,z=0;
int MAXOGR1=MAXOGR;

     for(i=1;i<=t;i++)
        {
R:        for(k=0;k<t;k++)

      if(SA[0][k]==i)
           {
           SAM[z][a][0]=A2[0][k][0];
           SAM[z][a][1]=A2[0][k][1];
           SAM[z][a][2]=A2[0][k][2];
           SAM[z][a][3]=A2[0][k][3];
           DAMtop[z]+=SAM[z][a][3];
           }
        a++;
        if(i==t) goto S;
        if(a==(MAXOGR1/GROUPNO)) {if(z==0){MAXOGR1=MAXOGR-a; GROUPNO=GROUPNO-1; z++;
i++; a=0;goto R;}else {MAXOGR1=MAXOGR1-a;GROUPNO=GROUPNO-1; z++; a=0;}}
        }

S:   if (a==(MAXOGR1/GROUPNO)) {if(z==0){MAXOGR1=MAXOGR-a; GROUPNO=GROUPNO-1; z++;
a=0;}else {MAXOGR1=MAXOGR1-a; GROUPNO=GROUPNO-1; z++; a=0;}}
        for(i=1;i<=m;i++)
        {
T:       for(k=0;k<m;k++)

      if(SA[1][k]==i)
           {
           SAM[z][a][0]=A2[0][k+t][0];
           SAM[z][a][1]=A2[0][k+t][1];
           SAM[z][a][2]=A2[0][k+t][2];
           SAM[z][a][3]=A2[0][k+t][3];
           DAMtop[z]+=SAM[z][a][3];
           }
        a++;
        if(i==m) goto U;
        if(a==(MAXOGR1/GROUPNO)) {MAXOGR1=MAXOGR1-a; GROUPNO=GROUPNO-1; z++; i++;
a=0;  goto T;}
        }

U:  if (a==(MAXOGR1/GROUPNO)) {MAXOGR1=MAXOGR1-a; GROUPNO=GROUPNO-1; z++; a=0;}
```

```
        for(i=1;i<=l;i++)
        {
X:       for(k=0;k<l;k++)

      if(SA[2][k]==i)
           {
           SAM[z][a][0]=A2[0][k+t+m][0];
           SAM[z][a][1]=A2[0][k+t+m][1];
           SAM[z][a][2]=A2[0][k+t+m][2];
           SAM[z][a][3]=A2[0][k+t+m][3];
           DAMtop[z]+=SAM[z][a][3];
           }
        a++;
        if(i==l) goto W;
        if(a==(MAXOGR1/GROUPNO)) {MAXOGR1=MAXOGR1-a; GROUPNO=GROUPNO-1; z++; i++;
a=0;  goto X;}
        }

W:   if (a==(MAXOGR1/GROUPNO)) {MAXOGR1=MAXOGR1-a; GROUPNO=GROUPNO-1; z++; a=0;}
        for(i=1;i<=p;i++)
        {
Y:       for(k=0;k<p;k++)

      if(SA[3][k]==i)
           {
           SAM[z][a][0]=A2[0][k+t+m+l][0];
           SAM[z][a][1]=A2[0][k+t+m+l][1];
           SAM[z][a][2]=A2[0][k+t+m+l][2];
           SAM[z][a][3]=A2[0][k+t+m+l][3];
           DAMtop[z]+=SAM[z][a][3];
           }
        a++;
        if(i==p) break;
        if(a==(MAXOGR1/GROUPNO)) {MAXOGR1=MAXOGR1-a; GROUPNO=GROUPNO-1; z++; i++;
a=0;  goto Y;}
        }

Form2->Show();
Form2->Label1->Caption="LEVEL-BASED RANDOM ASSIGNMENT";
GROUPNO=Edit2->Text.ToInt();
for(k=0;k<GROUPNO;k++)
  {
  char s1[20],s[100];
  strcpy(s,"");
  sprintf(s1,"                  GROUP %d",k+1);
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);
  strcpy(s,"");
  sprintf(s1,"NO              NAME-SURNAME    CLUSTER   MEAN");
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);

  strcpy(s,"");
  sprintf(s1,"=============================================");
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);

     for(g=0;g<(MAXOGR/GROUPNO)+1;g++)
       {
       if (SAM[k][g][0]!=0)
         {
         strcpy(s,"");
         sprintf(s1,"%2d   %23s   %5.0lf   %7.1lf",g+1,AD[int(SAM[k][g][0])-
1],SAM[k][g][1],SAM[k][g][3]);
         strcat(s,s1);
         Form2->Memo1->Lines->Add(s);
         }
       }
       strcpy(s,"");
       sprintf(s1,"=============================================");
       strcat(s,s1);
       Form2->Memo1->Lines->Add(s);
       if(SAM[k][g-1][0]==0) {g=g-1;}
       DAMort[k]=DAMtop[k]/g;
       DAMHKT[k]=0;
       DAMstdev[k]=0;

       for(g=0;g<(MAXOGR/GROUPNO)+1;g++)
       if(SAM[k][g][3]!=0) {DAMHKT[k]+=pow((SAM[k][g][3]-DAMort[k]),2);}
       if(SAM[k][g-1][0]==0) {g=g-1;}
       DAMstdev[k]=DAMHKT[k]/(g-1);
       strcpy(s,"");
       sprintf(s1,"MEAN=%3.2lf   STDEV=%3.2lf ",DAMort[k],sqrt(DAMstdev[k]));
       strcat(s,s1);
       Form2->Memo1->Lines->Add(s);
       Form2->Memo1->Lines->Add("");
  }
break;
}

case 3:{//BALANCED ASSIGNMENT**************************************

for(int i=0;i<25;i++)
   for(int j=0;j<25;j++)
      for(int k=0;k<4;k++)
      DAM1[i][j][k]=0.0;

k=0,j=0,a=0,z=0;

      for(k=0;k<MAXOGR;k++)
           {
EE:        DAM1[z][a][0]=A2[0][k][0];
           DAM1[z][a][1]=A2[0][k][1];
           DAM1[z][a][2]=A2[0][k][2];
           DAM1[z][a][3]=A2[0][k][3];
           DAMtop[z]+=DAM1[z][a][3];
           z++;
           if (k==MAXOGR-1) {goto TT;}
           if (z==GROUPNO) {z--; a++; k++; goto EEE;}
           if (k!=MAXOGR-1 && z!=GROUPNO) {k++; goto EE;}
           }

EEE: for (k=k;k<MAXOGR;k++)
           {
           DAM1[z][a][0]=A2[0][k][0];
           DAM1[z][a][1]=A2[0][k][1];
           DAM1[z][a][2]=A2[0][k][2];
           DAM1[z][a][3]=A2[0][k][3];
           DAMtop[z]+=DAM1[z][a][3];
           z--;
           if (k==MAXOGR-1) {goto TT;}
           if (z==-1) {z++; a++; k++; goto EE;}
           if (k!=MAXOGR-1 && z!=-1) {k++; goto EEE;}
           }

TT: Form2->Show();
Form2->Label1->Caption="BALANCED ASSIGNMENT";
for(k=0;k<GROUPNO;k++)
  {
  char s1[20],s[100];
  strcpy(s,"");
  sprintf(s1,"                  GROUP %d",k+1);
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);
  strcpy(s,"");
  sprintf(s1,"NO             NAME-SURNAME    CLUSTER   MEAN");
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);
  strcpy(s,"");
  sprintf(s1,"==============================================");
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);


     for(g=0;g<=a;g++)
       {
       if (DAM1[k][g][0]!=0)
         {
```

```
         strcpy(s,"");
         sprintf(s1,"%2d   %23s   %5.0lf   %7.1lf",g+1,AD[int(DAM1[k][g][0])-
1],DAM1[k][g][1],DAM1[k][g][3]);
         strcat(s,s1);
         Form2->Memo1->Lines->Add(s);
         }
        }
        strcpy(s,"");
        sprintf(s1,"=============================================");
        strcat(s,s1);
        Form2->Memo1->Lines->Add(s);
        if(DAM1[k][g-1][0]==0) {g=g-1;}
        DAMort[k]=DAMtop[k]/g;

        DAMHKT[k]=0;
        DAMstdev[k]=0;

        for(g=0;g<=a;g++)
        if(DAM1[k][g][3]!=0) {DAMHKT[k]+=pow((DAM1[k][g][3]-DAMort[k]),2);}
        if(DAM1[k][g-1][0]==0) {g=g-1;}
        DAMstdev[k]=DAMHKT[k]/(g-1);
        strcpy(s,"");
        sprintf(s1,"MEAN=%3.2lf   STDEV=%3.2lf ",DAMort[k],sqrt(DAMstdev[k]));
        strcat(s,s1);
        Form2->Memo1->Lines->Add(s);
        Form2->Memo1->Lines->Add("");
  }
break;
}

case 4:{//LEVEL-BASED ASSIGNMENT***********************************

for(int i=0;i<25;i++)
   for(int j=0;j<25;j++)
      for(int k=0;k<4;k++)
      SAM1[i][j][k]=0.0;

k=0,j=0,a=0,z=0;
int MAXOGR1=MAXOGR;

      for(k=0;k<MAXOGR;k++)
        {
EE1:    SAM1[z][a][0]=A2[0][k][0];
        SAM1[z][a][1]=A2[0][k][1];
        SAM1[z][a][2]=A2[0][k][2];
        SAM1[z][a][3]=A2[0][k][3];
        DAMtop[z]+=SAM1[z][a][3];
        a++;
        if(k==MAXOGR-1) {goto TT1;}
        if(k!=MAXOGR-1 && a==(MAXOGR1/GROUPNO)) {MAXOGR1=MAXOGR1-a; GROUPNO=GROUPNO-
1; z++; k++; a=0;  goto EE1;}
        if(k!=MAXOGR-1 && a!=(MAXOGR1/GROUPNO)) {k++; goto EE1;}
        }

TT1: Form2->Show();
Form2->Label1->Caption="LEVEL-BASED ASSIGNMENT";
GROUPNO=Edit2->Text.ToInt();
for(k=0;k<GROUPNO;k++)
  {
  char s1[20],s[100];
  strcpy(s,"");
  sprintf(s1,"                 GROUP %d",k+1);
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);
  strcpy(s,"");
  sprintf(s1,"NO              NAME-SURNAME    CLUSTER   MEAN");
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);
  strcpy(s,"");
  sprintf(s1,"=============================================");
  strcat(s,s1);
  Form2->Memo1->Lines->Add(s);

     for(g=0;g<a;g++)
```

```
       {
       if (SAM1[k][g][0]!=0)
         {
         strcpy(s,"");
         sprintf(s1,"%2d   %23s   %5.0lf   %7.1lf",g+1,AD[int(SAM1[k][g][0])-
1],SAM1[k][g][1],SAM1[k][g][3]);
         strcat(s,s1);
         Form2->Memo1->Lines->Add(s);
         }
       }
       strcpy(s,"");
       sprintf(s1,"==============================================");
       strcat(s,s1);
       Form2->Memo1->Lines->Add(s);
       if(SAM1[k][g-1][0]==0) {g=g-1;}
       DAMort[k]=DAMtop[k]/g;

       DAMHKT[k]=0;
       DAMstdev[k]=0;

       for(g=0;g<a;g++)
       if(SAM1[k][g][3]!=0) {DAMHKT[k]+=pow((SAM1[k][g][3]-DAMort[k]),2);}
       if(SAM1[k][g-1][0]==0) {g=g-1;}
       DAMstdev[k]=DAMHKT[k]/(g-1);
       strcpy(s,"");
       sprintf(s1,"MEAN=%3.2lf   STDEV=%3.2lf ",DAMort[k],sqrt(DAMstdev[k]));
       strcat(s,s1);
       Form2->Memo1->Lines->Add(s);
       Form2->Memo1->Lines->Add("");
  }
break;
}
}
}
//---------------------------------------------------------------------
```