PRIORITISATION OF ROADS USING MCDM APPROACHES – A CASE STUDY OF ROADS IN SPRAWL AREA OF HYDEREABAD

Project Report submitted
in partial fulfilment of the requirements for
the award of the Degree of

Bachelor Of Engineering

in

Civil Engineering

By

V. Aishwarya Laxmi (1602-18-732-001)

Soumya Shanigarapu (1602-18-732-048)

Tarun Kumar Koram (1602-18-732-054)

Under the guidance of

Dr. K. Jayasree (M. Tech, Ph. D)

Assistant Professor

DEPARTMENT OF CIVIL ENGINEERING

VASAVI COLLEGE OF ENGINEERING(AUTONOMOUS)

(Affiliated To Osmania University)

IBRAHIM BAGH, HYDERBAD-500031

JUNE 2022

DEPARTMENT OF CIVIL ENGINEERING

VASAVI COLLEGE OF ENGINEERING (AUTONOMOUS)

HYDERABAD

CERTIFICATE

This is to certify that the Theme Based Project work "**PRIORITISATION OF ROADS USING MCDM APPROACHES-A CASE STUDY OF ROADS IN SPRAWL AREA OF HYDERABAD**" is being submitted by V.AISHWARYA LAXMI (1602-18-732-001), S. SOUMYA (1602-18-732-048), K. TARUN KUMAR (1602-18-732-054) in partial fulfilment for the award of Bachelor of Engineering in Civil Engineering Department, Vasavi College of Engineering (Autonomous), Ibrahim Bagh, Hyderabad -500031 is a record of bonafide work carried out by him / her under our guidance.

Dr K. Jayashree	**Dr B. Sridhar**
Project Guide	**Professor & Head**
Dept. of Civil Engineering	**Dept. of Civil Engineering**

ACKNOWLEDGEMENT

We express our sincere thanks to our institution for fostering an excellent ambience without which this project would not have been fruitful.

We would like to thank our adored professor **Dr. B. Sridhar, Professor and Head of Department of Civil Engineering** for his valuable suggestions and encouragement throughout the period of our project work.

We take immense pleasure in expressing our deep sense of gratitude to our beloved project guide **Dr. K. Jayasree, Assistant Professor, in civil Engineering**, for her constant support, valuable guidance, constructive criticism and timely suggestions throughout our project work. We choose this moment to acknowledge her contribution gratefully. We would like to offer our special thanks to **Dr. T. Srinivas and K. Pavan Kumar** for their encouragement and patience throughout the project.

We perceive this opportunity as a big milestone in our career development, we will strive to use gained skills and knowledge in the best possible way, and we will continue to work on their improvements, in order to attain desired career objectives.

Sincerely,

V. AISHWARYA LAXMI **1602-18-732-001**

SOUMYA SHANIGARAPU **1602-18-732-048**

TARUN KUMAR KORAM **1602-18-732-054**

ABSTRACT

Pavements are the key asset for a country's economic and industrial development. Maintenance of roads is as significant as laying them. Pavements deteriorate as time passes, due to various factors like excessive vehicular stress, sunlight exposure, water intrusion, unequal expansion, contraction due to seasonal changes, etc. Repairing them not only cost extortionate amounts, but also huge time loss. Thus, an algorithm is required, such that it is convenient and flexible enough for the users to input 'n' number of factors, in order make optimally informed decisions. The study has aimed to provide a fully flexible, user-convenient algorithm, that can take huge amounts of pavement information and can yield results as ranks, so that user can know which road is to be prioritized in an informed order. 21 road stretches, each measuring 1km are taken in the areas Ibrahim Bagh, Alijapur, Neknampur`, Mahalneknampur and Jai hind colony and three Multi Criteria Decision Making (MCDM) methods were integrated namely Fuzzy TOPSIS, AHP and Concordance which were written in Python programming language. Equal weightages are given for all the methods chosen. A sensitivity analysis was also performed on these outputs to understand the level of accuracy from the results displayed. The algorithms provide stretch wise ranks as outputs and furthermore, these ranks were imported into GIS, to display several thematic maps for various factors considered. Finally, the results achieved from the project are almost equal in all the three methods and according to the results furnished, stretch 5 should be given the highest priority, followed by Stretch 12.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

S.NO	Abbreviation	Full form
1	A_L	Alligator Cracking
2	B_L	Block Cracking
3	E_C	Edge Cracking
4	T_C	Transverse Cracking
5	L_C	Longitudinal Cracking
6	M_C	Mud Cracking
7	P	Patching
8	RA	Ravelling
9	R	Rutting
10	D	Depression
11	S	Shoving
12	POP	Population
113	N _TOWN	Nearby Towns
14	CBR	California Bearing Test
15	FS	Facilities Score
16	ST_NO	Stretch Number
17	FIG	Figure
18	NO	Number
19	WT	Weight
20	DEC	Decreases
21	INC	Increases

CHAPTER 1

INTRODUCTION

1.0 GENERAL:

A well-structured, efficient, and well-planned road transportation infrastructure is a critical infrastructural input for a country's long-term economic growth. It also plays a crucial role in encouraging national integration, which is crucial in a varied country like India. India holds the world's second-biggest road network, preceded by the United States, measuring to a total length of 5,603,293 km (Basic Road Statistics, India, 2015–16, 2015). The highway network in India has a density of roughly 0.66 km/km2 of land (World Bank 2014). The building of a road network necessitates significant expenditure and as a result, effective care of these produced assets is required of the utmost significance. Pavements in excellent condition are critical for maintaining road users' comfort and safety. However, most Indian roads are crowded and have poor surface quality, which can pose serious dangers to motorists and be harmful to automobiles. To ensure the availability of year-round high-quality standards, regular monitoring of a set of performance criteria across the whole road infrastructure network, as well as the routine execution of suitable maintenance operations when and when needed, is essential. Pavement preservation and repair should be given the attention it deserves, given the importance of pavements to the country.

The demand for road maintenance resources is far greater than what is available, creating a complicated challenge of coordinating time, materials, equipment, funding, labour, design, and decision-making (Chopra et al. 2017). In light of this, it is nearly impossible for a highway agency to either attend to all road sections in need of care during a particular planning period or to complete all scheduled repair and maintenance work at the same time. The procedure necessitates the use of a priority system to identify and schedule road sections for maintenance. By providing a priority-rating score to each examined road segment, a priority ranking of road sections may be obtained. The ranking is traditionally determined by the professional assessment and judgment of experienced highway engineers and skilled maintenance staff.

Many nations have established a variety of priority-rating-scoring techniques (Chandran et al. 2007, Dalal et al. 2010, Khademi and Sheikholeslami 2010, Baba Shamsi et al. 2016, Ahmed et al. 2017). The analytical hierarchy process (AHP) is a commonly used technique. Dalal et al. (2010), Baba Shamsi et al. (2016), and Ahmed et al. (2016) employed approaches for identifying pavement maintenance priorities (Dalal et al., 2010, Baba Shamsi et al., 2016, Ahmed et al., 2016).2017). The perception of experts is AHP's key input. The validity of AHP is determined by the consistency of the data. These elements Expert perception, on the other hand, is subject to subjectivity, and AHP is validated based on the consistency limitations of these inputs. However, expert perception involves subjectivity, and the model becomes useless if a judgmental error occurs. The fuzzy logic approach is another option for allocating pavement maintenance priorities. Chandra, et al. Fuzzy logic is said to be capable of modelling highly complicated non-linear functions and has a tolerance for erroneous data. Linguistics is used in this strategy. To analyse survey results, variables are employed. Linguistic variables are variables whose values are natural language terms like very good, good, fair, bad, and extremely poor. However, applying this strategy is quite difficult because linguistic variation boundaries have a major impact on accuracy in Multi-Criteria Decision Making (MCDM), the Maintenance Priority Index-based method, and so on. It could also be used to prioritize pavement maintenance.

1.1 PROBLEM CONTEXT:

Pavement Distress is formed due to vehicular stress, sunlight exposure and unequal expansion. As good pavements have been very important to cities. A city must have good pavements to provide comfort to the people who travel on these roads. Many authors and researcher had highlighted that good pavement was a part of a modern urban transportation systems plays an important role in economy growth of the country and drivers' safety. In a few journals researchers have mentioned that prioritization of pavements using by visiting off the site is difficult and complicated process., using various traditional methods like Fuzzy, AHP, Concordance methods leads to make the process somewhat easy parameter by considering all the effective parameters simultaneously. Most of the roads in the sprawl areas are not in good condition. This is because of poor maintenance of roads or by heavy population etc.

Pavement distress has been shown to have negative effects on the roads where population and facilities sectors are high as result this has become a concern in these routes. If this problem is ignored overtime, it will lead to removal of entire road structure and cause inconvenience for the people. Addressing this problem will have practical benefits for the region and contribute to understanding of maintenance of roads regularly.

1.2 NEED OF THE STUDY:

The aim of the project is to prioritize pavements and assign them ranks based on the road condition in one of sprawl areas like Ibrahim Bagh, Neknampur in Hyderabad. The project has focussed on suitable surveys and laboratory works to measure the effectiveness of the research.

Furthermore, the project goal was to evaluate types of distress on different types of roads and their prioritization based on the condition of pavement. V. Sunitha and few others has stated in their paper that, if it is needed to lay new roads on the damaged roads it becomes hectic. Moreover, funds released by the government officials will be very less and time taking process to lay new roads is high. So, within the budgets officials can prioritise the roads based on their structural and functional parameters and perform maintenance within the budget given or released by the government.

By providing maintenance regularly will help in achieving not only the preserving current assets, but also lowering future cost for citizens, road user, taxpayers and road owners. Maintenance can help in in achieving the traffic flow safety and smoothly. It helps in developing the nation's economic growth.

1.3 OBJECTIVE OF THE STUDY:

This research has predominantly discussed areas where we can apply different methods like Analytical Hierarchy Process and Fuzzy Multiple Criteria Decision Making and concordance method to determine the prioritization of different pavements and giving them ranks to consider which road is best and other is worst.

Moreover, in this project we need to identify the criteria that support the acceptability of the site location. Following are the specific objectives framed in the study.

- Collection of all data related to the study like cracks, population. Towns etc.
- Analysing the behaviour of criteria that effect the formation of cracks on different types of roads.
- Application of Analytic Hierarchy Process, Fuzzy (Multicriteria Decision making) and concordance logic to quantify the criteria that affect the objective.

1.4 ASSUMPTIONS IN THE STUDY:

- The attributes obtained in this analysis, have taken for peak hours which include weekends and weekdays
- As carriage width is not constant throughout the stretch so number of lanes are considered into account to further procced.
- There are no undulations and curvatures along the study area, so sight distance is considered as adequate.
- Shortest distances from the stretches are considered as the nearby town distance.
- Population of that town can be calculated by adding all the population of towns less than 2 kms.

1.5 PRESENTATION OF THE WORK:

1. In chapter one, the project has focussed on the introduction of the thesis, problem context, need of the study, assumptions of the study, and objectives.
2. In chapter two, it has dealt with the literature review and the current state of art in prioritising various pavements. It has focussed on the application of fuzzy logic, AHP and concordance in transportation engineering.
3. In chapter three, the project explains about the study area. This project has given details about the stretch and how the site selection and other process have been carried out. Furthermore, the observations are recorded and tabulated for each study stretch

4. In chapter four, the research has highlighted the details of methodology. The steps require for presenting the methods, techniques, procedure, and rules required on how the project has been executed., it is meticulously presented. The process of AHP, FUZZY logic and Concordance and code using NumPy and pandas have been explained.
5. In chapter five states the application of the chosen methodology in the project and it's factors along with the result. The analysis has been done through various surveys, parking patrolling survey, and field works. These have been tabulated and the analysis through GIS and has been explained in detail.
6. In chapter six with the help of the qualitative and quantitative attributes obtained through survey and analytical process, the best and worst road has chosen. In this chapter results have been obtained after the comprehension of findings from the study area, and the validation of the approach from the application of methodology.
7. In chapter seven, summary and the conclusion of the project have been presented, where the results have been summarized.
8. In chapter eight, the references have been presented.
9. In chapter nine, the appendix has been presented.

CHAPTER 2

LITERATURE REVIEW

2.0 GENERAL:

A good transport system is the backbone of a county's economic growth and development and is directly proportional to the available infrastructure. Good and efficient planning of pavements and infrastructure boosts the industrial and socio-economic growth. Deferred pavement maintenance procedures cost exorbitantly and severely affects the country's development. Hence, a comprehensive road maintenance program must be established and need to be implemented to always guarantee the good working status of roads in order to provide safety and comfort to the citizens. Many researchers have embarked on a quest of finding an efficient method to identify the pavements and optimize the resources and funds available. Thus, this chapter specifically covers various available scientific literatures referred, further, proposes identification of defects of pavements on visual basis and methods such as MCDM, Fuzzy approach, Fuzzy Topsis, AHP and Concordance to develop a workable solution with available factors.

2.1 PAVEMENT DISTRESSES INTRODUCTION:

DEFINITION: Pavement distresses are the indication of the performance of an unfavorable pavement and shows the signs of upcoming failures (or) it refers to a variety of types of pavement distresses that occur on the surface of pavements.

Distressed pavement is often a result of a combination of various factors, rather than just one root cause. Different types of pavements develop different surface defects and are often formed on roads due to poor construction, fluctuation in temperature, usage of cheap materials, excess loadings, excess deflections underneath the road surface, etc. Distresses are often visualized on the road by the naked eye. These are classified into several types based on their shape, position, intensity, and depth. They can be broadly classified as cracking, surface deformation, disintegration (potholes, etc.), surface defects (bleeding, etc.)

2.2 IDENTIFICATION OF DISTRESSES ON ROADS:

2.21. IRC 082-2015: Code of Practice for Maintenance of Bituminous Surfaces.

This codebook, defined as per Indian Standards, has focused on different types of pavement distresses and their causes on the Bituminous Road. The code is organized into 13 broad sections and focused on preventive measures have been given to each problem, caused by each crack on the bituminous pavements. Many aspects of maintenance, such as materials, tools and equipment, traffic arrangements, and overall organization and management are also covered. Special problems caused due to drainage or temperature were discussed. Maintenance of the roads has been elaborated and rectification of distress by different materials and methods is also covered. The surveying of existing pavements to assess physical condition, structural capacity, and roughness is also discussed in the article.

2.22. IRC: SP:083-2008: **Guidelines for Maintenance Repair and Rehabilitation of Cement Concrete Pavements**

This codebook, defined as per Indian Standards, has focussed on different types of pavement distresses and their causes on the cement concrete roads. This code is broadly divided into 15 sections. The codebook has focussed on making the engineer understand the types and causes of various defects on a cement concrete pavement and also mentioned way to segregate them according to various evaluation procedures such as visual condition survey, Faulting surveys, GPR surveys, roughness indices etc. in both structural and functional processes respectively. Procedures of planning, aspects of maintenance, tools and materials needed, and several methods of repairing the roads were also detailed accordingly.

2.3 FUZZY LOGIC INTRODUCTION:

DEFINITION: Fuzzy logic is an approach to computing based on “degrees of truth” rather than the usual “true or false” (1 or 0) Boolean logic on which the moderate computer is based. The idea of fuzzy logic was first advanced by Dr. Lofti Zadeh of the University of California of Berkeley in the 1960s. Dr. Zadeh was working on the problem of computer understanding of natural language. Natural language (like most other activities in life and indeed the universe) is not easily translated into the absolute terms of 0 and 1, (whether everything is ultimately describable in binary terms is a philosophical question worth pursuing, but in practice, much data we might want to feed a computer is in some state in between and so, frequently, are the results of computing). It has helped to see fuzzy logic as the way reasoning has worked and binary or Boolean logic was simply a special case of it.

Fuzzy logic has included 0 and 1 as extreme cases of truth (or "the state of matters" or "fact") but it has also included various states of truth in between so that for example, the result of a comparison between two things could be not "tall" or "short" but "0.38 of tallness". Fuzzy logic has been closer to the way our brains work. We have aggregated data and formed several partial truths which we aggregate further into higher truths which in turn, when certain thresholds were exceeded, cause certain further results such as motor reaction. A similar kind of process has been used in neural networks, expert surveys, and other artificial intelligence applications. Fuzzy logic was essential to the development of human-like capabilities for AI, sometimes referred to as artificial intelligence: the representation of generalized human cognitive abilities in software so that, faced with an unfamiliar task, the Al system could find a solution.

2.4 HISTORY OF FUZZY LOGIC APPLICATIONS

2.41 Fuzzy Logic in Transportation

This paper has been presented by Amrita Sarkar and G. Sahoo, in 2012, where fuzzy logic is shown to be a very promising mathematical approach for modelling traffic and transportation processes characterized by subjectively, ambiguity, uncertainty and imprecision. The basic premises of fuzzy logic systems were presented, as well as a detailed analysis of Fuzzy Logic systems have been developed to solve various transportation planning problems. Emphasis has been put on the importance of fuzzy logic systems as universal approximates in solving transportation problems. This paper has presented an analysis of the results achieved using fuzzy logic to model complex transportation process. [Amrita Sarkar,2012]

2.42 Fuzzy Logic Technique for Pavement Condition Rating & Maintenance Needs Assessment

This paper presented by T. F. Fwa and R. Shanmugam, 1998, discusses the development of a fuzzy logic-based system of pavement condition rating and maintenance-needs assessment for a road network of an industrial park located on an island. For this purpose, a user-friendly PC based computer programs has been developed to accept the data directly and process pavement condition survey data, and produce the required pavement distress rating and recommended maintenance actions accordingly. Fuzzy Mathematical approach was incorporated in the tool for subjective analysis, to shrink the uncertainty of PC rating and for assessing the maintenance-needs. The paper has considered certain standards to classify each defect's severity levels thus enabling computation of distress rating score for each pavement distress and with this better ability of differentiating them, thus making possible a more effective way of maintenance. A membership function was established for distress severity and corresponding distress rating scores were developed. Concluding, the procedure proposed has a clear and logical decision framework and generated important information that can be utilised to optimise the scheduling of maintenance tasks.

2.43 Prioritization of Pavement Stretches using Fuzzy MCDM Approach - A Case Study

Logical distribution of funds is the main goal for any effective pavement management techniques to create an optimistic prioritization approach. However, there is involvement of uncertainties of some parameters that are often left inadequately addressed. This paper, a case study, by Sandra. A.K, Vinayaka Rao. V.R, Raju. K.S, Sarkar. A.K, 2010, have identified one such parameter, the severity distress, which is often difficult to assess precisely. A Fuzzy Multi Criteria Decision Making approach (Fuzzy TOPSIS) was chosen and for demonstration purpose, pavement distresses and their severity was collected in the study area. An expert survey was also prepared from 15 different experts for each criterion. Raw data was collected from field on visual basis. This data was put into the prioritization model and after undergoing several detailed procedural steps, the ranks of the stretches were outputted, indicating the optimistic preference for maintenance. This paper produced a versatile approach, to include various other possible factors, to make the prioritization even more accurate.

2.44 A Development of Fuzzy Pavement Condition Assessment

Pavement Management Systems (PMS) are widely employed across the world to aid pavement network administrators in making proper and cost-effective highway improvement decisions. The capacity of PMS to describe the status of pavement networks is one of the most basic and significant properties, and various pavement condition indices, including as PCI, PSI, and MCI, are utilised for this purpose. This paper published by Joni Arliansyah, Teruhiko Maruyama and Osamu, 2003, have focussed on developing a pavement condition assessment method using Fuzzy Weighted Average operation, considering an existing database and some practical aspects, in the country, Japan. The main purpose of the study was (1) proposing a method to determine membership functions in pavement condition assessment based on experts' opinions; (2) investigating the effects of the inclusion or excluding pavement parameters on pavement condition assessment, the effects of weight changes and sensitivity of rating terms using FWA approach; (3) comparison of values obtained with maintenance control index (MCI), developed by Japanese Ministry of Construction. The paper has presented a detailed procedure and methodology, satisfying the objective of the project, thus providing a FPCI model generating better results than existing MCI model.

2.45 Using Fuzzy Logic and Expert System Approaches in Evaluating Flexible Pavement Distress- Case Study

This case study was prepared by Hari Krishan Kodur, Feipeng Xiao, Serji N. Amirkhanian and C. Hsein Juang, 2010. The main objective of the methodology developed in this case study is to use automated techniques for classifying the flexible pavement distresses through a quick, efficient, and consistent classification. For this an expert system was developed in C language using Fuzzy logic for reasoning. Fuzzy logic was used to develop a methodology to categorize pavement distresses. Membership functions represent the gradual transition from membership to non-membership of an element or predicate in a set (MTC 1982; Zimmerman 1995; Lee and Donnell 2007). A membership function basically provides the degree of membership or extent of inclusion of a selected element in a fuzzy set. Linear regression, SCDOT's methods and standard SCDOT's categorization tables are used throughout the model development.

2.5 AHP INTRODUCTION:

DEFINITION: The Analytic Hierarchy Process (AHP) is a method for organizing and analysing complex decisions, using math and psychology. It was developed by Thomas L. Saaty in the 1970s. AHP has been suitable for complex decisions which have involved the comparison of decision elements, which was difficult to quantify. It was based on the assumption that when faced with a complex decision the natural human reaction was to cluster the decision elements according to their common characteristics. AHP provides a rational framework for a needed decision by quantifying its criteria and alternative options, and for relating those elements to the overall goal.

It involves building a hierarchy (Ranking) of decision elements and then making comparisons between each possible pair in each cluster (as a matrix). This has given a weighting for each element within a cluster (or the level of hierarchy) and also a consistency ratio (useful for checking the consistency of data).

2.6 HISTORY OF AHP APPLICATIONS:

2.6.1 The Analytic Hierarchy Process- What it is and how it is Used

This paper written by R.W. Satty, 1980, who is known to be the inventor of the Satty's scale, has introduced the Analytical Hierarchy Process (AHP) as a method of measurement with ratio scales. According to the paper, this can be used to derive the ratio scales from both discrete and continuous paired comparisons. These comparisons, taken from actual measurements or fundamental scales, reflect both strengths of preference and relative strength between the criteria at each hierarchical level defined. AHP has found its extensive application in Multi-Criteria Decision Making (MCDM). This method can be used to measure physical and psychological events. For using AHP in a modelling problem, a hierarchy or network structure is needed to represent the problem, and to establish the relations between structures

Pair Wise matrices must be created. In discrete, dominance matrices are created, whereas Kernel Fredholm Operators in continuous.

Two general hierarchy types namely, forward, and backward hierarchies were discussed along with two examples. According to their homogeneity, elements are clustered together at each level and picked out iteratively hierarchically. In Pair Wise matrices, the main criteria are compared, and preferences are given according to the standard Satty's scale. Then they are compared with sub-criterial divisions and elements within them and so on.

The main question the paper wanted to focus is how Pair Wise Comparison matrices can influence judgments and priorities. Later in this paper, a few axioms along with eigenvector solutions for weights and consistency, and absolute and relative measurement along with structural information were covered. Lastly, it has mentioned certain areas where the AHP model can extend its application and theoretical development of the model.

2.6.2 Application of AHP in Transportation Engineering

The paper has been presented by Valentinas PodvezkoIn, Henrikas Sivilevicius, 2014, which has highlighted the methodologies of solving different transport problems where the best decision has been determined by number of chosen quantitative criteria, which have been incorporated, it has described the qualitative parameters of transport systems in quantitative terms. Often weights which have revealed the importance of such criteria have been evaluated. The realm of proprietary methods which have been used in engineering sampling and experimental studies have not comprised methods of weight evaluation. Consequently, expert evaluation methods which have elicit weights of criteria from experienced, qualified, and fair experts were used. Among the most popular such methods was the method AHP (Analytic Hierarchy Process). They have used this method for investigation of inter-relationship on road traffic safety, and for evaluation of quality of passenger railway transportation service. [Valentinas PodvezkoIn, 2014]

2.6.3 Prioritization of pavement maintenance sections using objective based Analytic Hierarchy Process

The function of pavement maintenance is to diminish pavement deterioration and improve the life of a pavement. Pavement maintenance, if not done at appropriate times in a pre-planned manner, negatively impacts the transport system. This paper was published by Sarfaraz Ahmed, P. Vedagiri, K.V. Krishna Rao, 2017, had primary focus on presenting the study to assess the effectiveness of objective based AHP method in determining the pavement maintenance prioritization for the selected pavement sections, consisting multiple distresses. Also, the solution from priority ratings of AHP was compared with the corresponding solution by the traditional pavement maintenance procedure, the Road Condition Index (RCI) method. The study has used Terrestrial Laser Scanner and Cyclone software to collect the pavement data.

The objective based AHP had generated priority ratings, which were positively correlated with those obtained by the RCI method. Similarly, priority ranking evaluation by objective based AHP and RCI methods were found to be strongly correlated. The paper also shows the comparison between RCI and proposed objective AHP methods from their application point of view, the RCI method were found to be more beneficial than the proposed method with the ease of application of the RCI procedure. This limitation is overcome by using the proposed objective AHP method. The findings of this study suggest that AHP approach is suitable for the purpose of pavement maintenance prioritization. This study has introduced an objective manner of evaluation of pavement maintenance sections for prioritization using the AHP technique.

2.6.4 Pavement maintenance and prioritization using AHP- A case study of Rajkot city

The road network is deteriorating by several factors including climatic factors as well as traffic operations. Neglecting to maintain these roads, deterioration rapidly increases over time leading to inaccessibility and immobility in urban areas. Experts base their decisions solely on their experience while due consideration is not given to the actual quantitative physical condition of the roads. To overcome these difficulties, a paper was presented by Chirag R. Bhuva, Prof. Bindiya Patel, Prof. Mayank Kanani, 2019, where an objective based AHP method is proposed in this study, where pair wise comparison values are assigned based on the field data collected from a road network, in Rajkot city. Final ranking list of road sections takes into consideration the priority weight of alternatives, which reflect the road conditions. The parameter or alternative was ranked first which has highest weightage value and it was more prior than other so, considered first for the maintenance work. This method consists of following steps a) Prioritizing by AHP method; b) Assigning judgemental preference value (0 to 9) by expert's opinion.; c) Priority weight for each parameter by AHP d) Priority weight for each alternative & Priority ranking to pavement sections for maintenance. The tools used in this project are Analytic Hierarchy Process (AHP), Bump integrator for Roughness Index, Benkelmen Beam Deflection to evaluate structural capacity.

2.6.5 Rural Road Maintenance Prioritization Index Based on Functional and Structural Parameters for Rural Road Network In Himachal Pradesh

This paper published by Dr. Aakash Gupta, Dr. Pradeep Kumar, Dr. Ashok Kumar Gupta, Dr. Ashish Kumar, 2021, was an attempt made to prioritize the sections of roads in rural area of Himachal Pradesh, India, in order to provide timely maintenance based on their structural and functional conditions. For this the developed Rural Road Maintenance Priority Index (RRMPI) was used, which provides the users with Overall Functional Condition Index (OFCI) and Overall Structural Condition Index (OSCI), having a scale ranging 0 -100, signifying worst to best condition of a pavement. For Functional evaluation, a road roughness measuring apparatus, MERLIN was used for accurate results. The objective Rider Comfort Rating (RCR) was also performed, provided on a scale of 0 – 100, where 0 is worst condition and vice versa. Skid Resistance was obtained from a skid resistance pendulum machine. For Structural evaluation, as per guidelines in IRC:81-1997, Benkelman Beam Deflection was conducted for characteristic deflection study and a CBR test was performed, and the modulus of subgrade was co-related according to IRC:58-2015. Distresses are obtained on visual basis, from a pre-defined standard and calculated using Total pavement distress index (FCI_{TPO}).

After finding all the required parameters, performing all corrections, expert opinion survey and collecting the raw data, the data was inputted performing the objective

AHP and required weightages were obtained accordingly. An Overall Acceptability Index is defined based on Fuzzy Logic. This method of RRMPI, is a powerful and handy tool, was proposed such that any rural roads with similar climatic, geological and traffic conditions make use of it.

2.7 CONCORDANCE INTRODUCTION:

DEFINITION: Concordance is an MCDM approach, which comes under the branch of Logistic Regression. A pair of observations with different observed responses is said to be concordant if the observation with the lower ordered response value (Y = 0) has a lower predicted probability of 1 than the observation with the higher order response value. (or) A number of pair is said to be concordant, if the probability of “1 in case of 1” is greater than probability of “1 in case of 0”.

Pairs are the total no of distinct pairs with one case having a positive response (Y = 1) and other having a negative response (Y = 0). The calculation of concordance mainly depends on the way of thinking relatively, in terms of probability. For ex., when among two quantities, one should be chosen to be the best quantity, they are chosen in terms of the probability or relative comparison, where no margin is defined. For ex: If a set of numbers [0.3, 0.4, 0.5, 0.7, 0.8] are considered and asked to divide them into best and worst quantities, one would quantify [0.5, 0.7, 0.8] as the best and [0.3, 0.4] as worst, considering 0.5 as a margin. But in case of concordance, one must choose by selecting lower most value as the worst i.e., 0.3 is defined to be the worst and the remaining set to be the best quantities. Same applies for 0.7 and 0.8, if they are to be compared.

2.8 GIS INTRODUCTION

DEFINITION: A Geographic information system (GIS) is a system that creates, manages, analyses, and maps all types of data. (or) GIS is a computer system for capturing, storing, checking, and displaying data related to positions on Earth's surface. GIS connects data to a map, integrating location data with all types of descriptive information. GIS helps users understand patterns, relationships, and geographic context. The benefits include improved communication and efficiency as well as better management and decision making.

As almost every field of life, GIS can help in achieving excellence in transportation as well. It significantly aids in planning, monitoring and managing complex systems involved in transportation planning and management more effectively. GIS helps in determining capacity enhancements, improving operations, and identifying the most strategic investments for keeping the transportation system in any country running optimally. Accident analysis, traffic modelling, route planning, highway management, transport safety management are some of the fields of transportation engineering where GIS has found its applications.

2.9 HISTORY OF GIS APPLICATIONS

2.9.1 Managing Road Maintenance Using Geographic Information System Application

This study was conducted by Mohd Zulkifli, B. Mohd Yunus and Hamidah Bt. Hassan, 2010, in Penang, Malaysia, with an objective to explore the potential of GIS in capturing, storing, updating, retrieving, displaying and printing data to facilitate road database management. The road database is monitored effectively using the ArcView software. The spatial data was provided such that all the data inserted is in digital format. Attribute data was also inserted through Microsoft Excel and ArcView. They are mainly for database features displayed. This data can be an image, a text file or a document. ArcView software created interactive maps with multiple themes or groupings of elements that are related. Thus, activating the related theme, information on the study area was obtained efficiently. This paper has shown that using GIS can minimize the re-doing of data collection of geospatial information and can improve data currency, accuracy and consistency of the data maintained. Also, the obtained map data is shown to be more definite, secured and organized.

2.9.2 Prioritizing Road Maintenance Activities using GIS platform and Vb.net

This paper was presented by Fardeen Nodrat and Dongshik Kang, 2018, where GIS and Vb.net was used to prioritize road maintenance and prioritization activities in the country Afghanistan. A study area in the city of Kabul was considered and accordingly certain constituents to prioritize were chosen and categorized them as positive and negative criteria, where positive indicates highest quantity has the highest priority and vice versa for negative.

Paver 5.2 software had been used to calculate PCI after collecting road distresses physically, for which the values range in a scale of 0 to 100. All the other factors were obtained as well from various civic departments, and this data was arranged in an Excel Sheet. Using Shannon Entropy method, by entering the values of criteria weightage, calculations were performed to obtain the decision matrix. Finally, after obtaining the weightages and decision matrix, the MARP tool was used to prioritize the alternatives. Subsequently through the inbuilt TOPSIS model, the software gives out optimized prioritization table, which later got imported into ARCGIS 10.4 software and various thematic maps were prepared, highlighting the criteria such as Road PCI, traffic volume, rehabilitation cost, rehabilitation activity ranking etc. The results have shown that the positive criterion had the greatest impact on the ranking of road maintenance activities.

2.10 REVIEW OF LITERATURE

Table 2.1 Review of Literature

AUTHOR	YEAR	APPLICATION AREA	TYPE OF AREA	FACTORS CONSIDERED	MODEL/ ANALYSIS TYPE
Navid Khademi, Abdolreza Sheikholeslami	2010	Gilan province, Iran	Rural	Gilan low-class road network survey, pavement condition, pavement type, Traffic volume, physical characteristics, topographical and natural conditions.	AHP, WAMM, ArcView GIS, Overlapping map layers and analysis
Danial Moazami, Ratnasamy Muniandy	2010	Tehran, Iran	Cosmopolitan	Hourly traffic volume, PCI, Road type, Maintenance cost	AHP and Fuzzy Logic Modelling, MATLAB
Lu Sun and Wenjun Gu	2011	-	-	Roughness, deflection, surface deterioration, rutting, and skid resistance	AHP, Fuzzy Logic theory, PCA MGP and DWCI
Yashon O. Ouma, J. Opudo and S. Nyambenya	2015	Eldoret, Kenya	Rural	Distress data, Visual based surface condition rating (ECR), Roughness index	AHP, Fuzzy AHP, Fuzzy TOPSIS, GPS, SQL database, GIS
P.C. Acquah, C. Fosu	2017	Kumasi, Ghana	Metropolitan	Stretches data: length, width, surface type; Roughness, surface distresses, Annual Traffic Data, Maintenance Budget	ArcGIS, SQL database, GPS, PMS, MCA method, VBA
Mansour Fakhri and Reza Dezfoulian	2017	Ilam Province, Iran	Rural	Pavement condition, distress types through rapid data collection from Rater’s judgement, traffic data, functional classification, age.	PMS and GIS integration; Ranking Methods, AI Techniques, AHP, Optimization methods for prioritization

Hussein Ali Ewadh, Raid Almuhannah and Saja Alasadi	2017	Karbala City, Iraq	Metropolitan	Distress types, ADT, pavement condition, functional classification data, Cost of Maintenance.	Paver system integrated with PAVER and GIS, PCI, matrix method, BCR method, SPSS software
Zeinab Maddahi, Ahmad Jalalian, Mir Masoud Kheirkhah Zarkesh and Naser Honarjo	2017	Amol District, Mazandaran Province, Iran	Rural	Soil properties, climate conditions, accessibility, Land Suitability Analysis: Digital topography maps, water bodies nearby, meteorological data, soil mapping, satellite images	Fuzzy AHP, ARCGIS software, FAO, Landsat TM, GPS, Spatial AHP, Cost Benefit Analysis
Kirill Skorokhod	2018	Syracuse, New York, USA	Urban	• Social Factors: Road Functional Classification, Proximity to important places, Population density. • Economic Factors • Environmental Factors: Global Warming Potential and Energy Consumption for Road Construction, Current condition of roads.	TAM, ArcGIS, PMS, PCA, AHP, Excel, MRR technique, Hierarchy process
Shruti Wadalkar, Ravindra K. Lad, Rakesh K. Jain	2020	Pune, Maharashtra, India	Metropolitan	Questionnaire survey for expert data collection, Structural distresses, PCI	Dempster's Shafer's-Fuzzy Evidence Theory Weightage method, PCDI, PMS
Sanjaya Subedi, Keshav Basnet and Nirmal Prasad Baral	2020	Annapurna rural municipality Kaski District, Nepal	Rural	Rapid condition Survey, questionnaire regarding road types, Maintenance Cost, Surface condition, Road users and Traffic volume	GPS

CHAPTER 3

STUDY AREA

3.0 GENERAL:

Hyderabad is the capital of Southern India’s Telangana state with an area of 650 sq.km. It is situated at the Deccan Plateau ,500 meters above the sea level and most of the area is rocky. At over 12.2 million, it has the largest population of any city in the state. It is an A-1 city under the terms of development priorities and has good intercity transport facilities. There are rail and air services to Delhi, Kolkata, Mumbai, Chennai and Bengaluru (Bangalore) as well as to historical sites including the nearby Golkonda fortress and the Ajanta and Ellora caves in neighboring Maharashtra state, the latter two locales having been designated UNESCO World Heritage sites in 1983. Hyderabad has become a hub of trade and commerce and an international center for information technology (IT). Transportation infrastructure within Hyderabad, however, has lagged the city’s rapid population growth, and pavement distress has become common in few areas like sprawl areas of Hyderabad. This is due to lack of release of funds or maintenance of sprawl area roads.

The sprawling areas of Hyderabad include Ibrahim Bagh, Neknampur, Alijapur, and other areas. As Ibrahim Bagh covers 5.58 square kilometers (Hyderabad area), military lands occupy nearly half of it. It links to the Rajiv Gandhi International Airport through the Nehru Outer Ring Road highway. Since the formation of Telangana, Ibrahim Bagh has been considered a part of the urban area. It can also be considered the sprawl area of Hyderabad. Although it is considered an urban area, the pavements are not maintained properly. Most of the roads are cement-concrete roads in towns. As it is covered by military areas, the passage of vehicles on major roads is less compared to minor roads. Ibrahim Bagh serves as a link between Manikonda, Gachibowli, Gandipet, and Narsingh. Minor roads are mostly used by people to reach their destinations easily. These minor roads are not maintained properly and most of them are completely damaged. During busy times, the area is densely populated as there are colleges like Vasavi College of Engineering, schools like primary and public schools, and many mosques in this area. As it is one of the most connected and quickest ways to reach many destinations, the pavements need to be good. The pavement needs maintenance or needs to be repaved. As government funds are limited, utilizing the limited funds to maintain the roads is the best option for prioritizing the pavements. The main towns like Neknampur, IbrahimBagh,JaiHind Colony are shown in map of img3.1

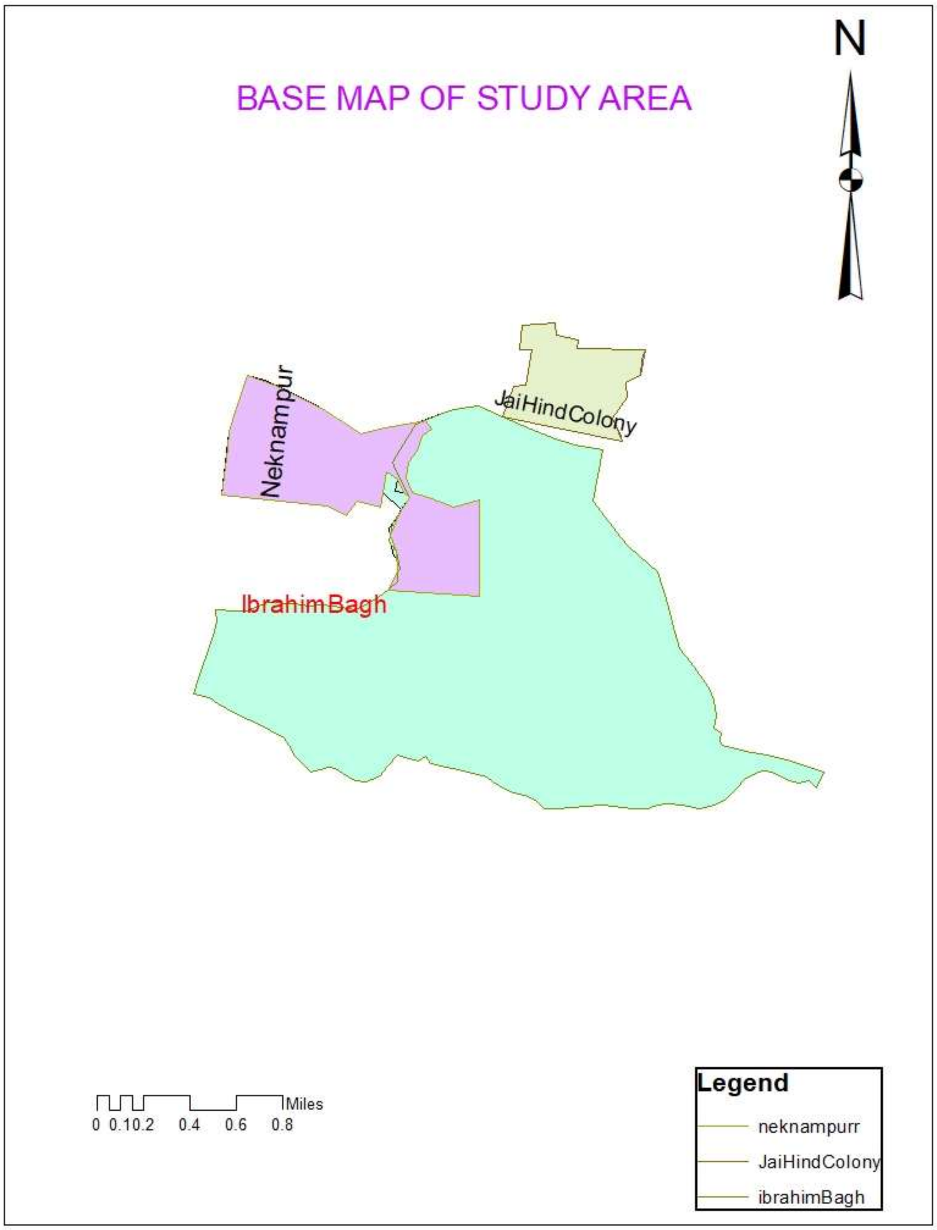


Fig 3.1 Map Showing Base map of Study Area

3.1 CHARACTERISTICES OF THE STUDY AREA:

As most of the study area is part of Ibrahim Bagh, Neknampur, JaiHind Nagar Colony. According to latest census Ibrahim Bagh has population of It falls under few towns in Ibrahim Bagh, Neknampur namely. The few towns which were part of the study area as follows,

1. Neknampur
2. Alijapur
3. Jai-Hind Colony
4. Secretariat Hills

   Keeping in the view, the pavement condition, nearby towns, different roads around the study area, this area was selected. This area has no proper maintenance of roads and unpaved roads in mostly in minor roads.

3.1.1 Area

The total study area is 6,937452 sq.km   that is equal to sq.km which is occupied by the towns mentioned above. The areas engaged by individual colonies are mentioned below.

Table 3.1: Total area of different towns in study area

S.NO	TOWN NAME	TOTAL AREA COVERED (in Sq.m)
1	Neknampur	10,21,116
2	Alijapur	5,72,514.40
3	Jai-Hind Colony	2,75,097
4	Secretariat Hills	49,841.06
5	Ibrahim Bagh	5,081,236

3.1.2 Road Network:

The study area of both major and minor roads. A total of 21 stretches are consider under this road network. The major road network provides easy access to the residential area as well as to the arterial roads. It comprises of 4 major roads. They are stretch 17, stretch 16, stretch 18, stretch 14. Most amount of Bituminous Road is found in this major road network.

The minor road network comprises of different colony roads which connects households to major road network. It contains total of 17 roads, most of them are made of cement concrete roads. Description of each stretch are listed in below table 3.2 and map of road network is displayed below in FIG 3.2.

Table 3.2 Description of Each stretch of Road Network in Study Area

STRECTH NO	LENGTH OF STRECTH(KM)	TYPEOF PAVEMENT	FUNCTIONALITY
1	1	Cement concrete	Local roads
2	1	Cement concrete	Local Roads
3	1	Cement concrete	Local roads
4	1	Cement concrete	Local Roads
5	1	Bituminous	Sub-Arterial Roads
6	1	Cement concrete	Local Roads
7	1	Cement concrete	Local Roads
8	1	Cement concrete	Local Roads
9	1	Cement concrete	Local Roads
10	1	Bituminous	Collector Roads
11	1	Bituminous	Collector Roads
12	1	Bituminous	Sub-Arterial Road
13	1	Bituminous	Subaerial Roads
14	1	Cement Concrete	Local Roads
15	1	Bituminous	Local Roads
16	1	Bituminous	Collector Roads
17	1	Bituminous	Sub-Arterial Roads
18	1	Bituminous	Sub-Arterial Roads
19	1	Bituminous	Sub-Arterial Roads
20	1	Bituminous	Sub-Arterial Roads
21	1	Bituminous	Sub-Arterial Roads

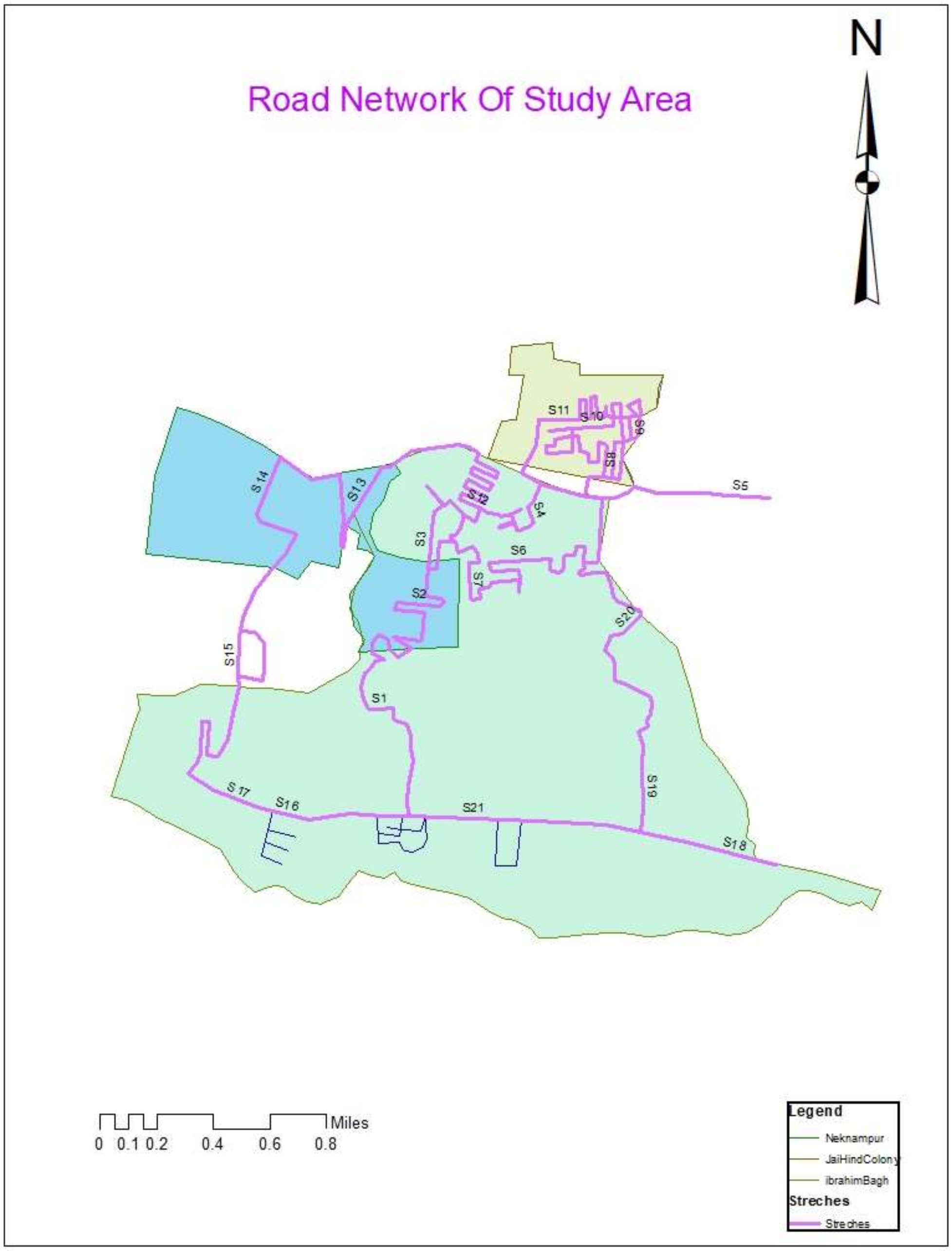


Fig 3.2 Map showing road network of study area

3.2 SURVEYS CARRIED OUT:

Surveys are conducted to collect the required information about the pavement condition and about other criteria like population, strength of soil.

The data required for the analysis is collected with following surveys.

- Pavement distress survey
- Classified volume count survey
- Road Inventory Survey
- Facilities score

3.2.1 Pavement Distress Survey:

Pavement distress survey is to find different types of cracks present on the stretch. In this survey, information is collected on the current condition of different pavements such as,

- The location, condition, type, and number of distresses on different pavements.
- Type of pavements either flexible or rigid pavement

Different types of roads like flexible, rigid pavements are selected and surveyed different types of roads to find different distress among the pavements. Images of that cracks on the different types of roads are to be taken. Different types of cracks are identified visually and analyzed by using different code books and journals.

From this survey it was found that most of the minor roads are badly damaged with respect to the major roads. Most of the roads are filled with water content as people were letting water on to the roads. On the other there were unpaved roads and muddy roads at some stretches. Major roads are maintained regularly at this area, but this is not case with minor roads. As minor roads are busy roads compared to the major roads in this area.

The study area, being mostly sprawl, has good pavements mostly in major road networks. It was observed that there were more facilities sectors in this sprawl area and mostly them are in minor roads. Pavements which have more facilities, population are taken as potential site

3.4.2 Classified Volume count survey

Volume of the road is one of the fundamental measures of traffic on a road system in each interval of time. It is a procedure to determine mainly volume of traffic moving on the roads at a particular section during a particular time. Volume of road is also termed as flow and is expressed in terms of vehicles per hour when the traffic is uniform.

It was carried out manually on all roads of major road and minor roads. On each road, a mark was made and for every vehicle that crosses the mark made, it was counted and continued the process till one hour. This can be repeatedly for three times and peak hour is taken into consideration for the study.

Table 3.3 Volume of Different stretches in study area

STRECTH.NO	VOLUME/hr.
1	98
2	332
3	484
4	311
5	696
6	724
7	724
8	943
9	887
10	112
11	720
12	912
13	728
14	412
15	172
16	153
17	155
18	166
19	97
20	63
21	67

3.4.3 ROAD INVENTORY SURVEY:

Sight distances is one of the geometric characteristics. As length of road visible ahead to the driver at any distance is termed as sight distances. From this survey it was found that there are no undulations and curvatures in the study area. So, there is adequate sight distances in study area.

3.4.4 FACILITIES SCORES:

It is one types of survey where one need to consider all facilities sectors in study area. In this survey the information collected for different sectors available in each stretch such as

- All sectors including educational institutions, hospitals, tourist attractions are to be noted.
- Each sector has different scores so each stretch may get different score.

  From this survey it was found that there are many primary schools, very less hospitals, Bus stops, tourist attractions and different colleges in this study area.

CHAPETR 4

METHODOLOGY

4.0 GENERAL

Methodology is the systematic theoretical analysis of the methods applied to a field of study, or the theoretical analysis of the body of the methods and principles associated with the branch of knowledge. It typically encompasses concepts such as theoretical model, paradigms, phases and quantitative or qualitative techniques.

4.1 METHODOLOGY

The following is the procedure that has been adopted for the study.

4.1.1 Identification of Multiple Criteria for Site Selection.

For the site selection of pavement prioritization, one parameter is not enough to evaluate the sites. So, there is a need for the selection for multi-criteria. From the Literature review, the following criterions have been selected that have significant impact on pavement prioritization site selection.

1) Strength of the soil

2) Population of the nearby towns

3) Facilities Score

4) Distance to the nearest towns

5) Traffic volume

6) Pavement distress

7) Geometric Characteristics

4.1.2 Determination of Criteria value for each site

1) STRENGTH OF THE SOIL

Strength of the soil can be evaluated by performing CALIFORNIA BEARING TEST(CBR). To perform this test field sample is required.

STEPS FOR FINDING OUT STRENGTH OF SOIL:

- Three different 5 kgs field sample from each stretch is to be collected.
- Sample is to be prepared before performing the test

SAMPLE PREPARATION AND SAMPLE TESTING:

- Around 5 kgs sample is to be taken
- Sample is to be sieved by passing 19mm sieve and retained on 4.5mm sieve.
- Sample is mixed thoroughly, and sample is divided into three parts.
- The cylindrical mould is to be filled with an extension collar a perforated base plate and thin film of oil is to be applied to the inside of the mould, the base plate, and the collar.
- The spacer disc is to be inserted over the baseplate and a 150 mm diameter coarse filter paper is to be placed on the top of the disc.
- The well mixed soil is then compacted in the prepared mould. Sample is to be placed in three layers and compacted by 55 blows in each layer in specified manner.
- The collar is to be removed and soil is trimmed to the size of the mould using the straight edge.
- The perforated base plate is removed, and the density of the compacted soil is obtained.
- A disc of coarse filter paper is placed on the perforated base plate. The mould is inverted.
- The surcharge weights is placed in a way to produce an intensity of loading equal to the weight of the base material and pavement within 2.5kgs but not less than 5 kgs.
- The penetration piston is seated at the center of the specimen with the smallest possible load. full contact is allowed between the surface of the specimen and the piston.
- suitable piston is chosen for regulating value on the hydraulic pressure unit.
- Loads are applied on the penetration piston so that the ratio of application is approximately 1.25mm/min
- Load readings are recorded at penetration of 0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 5, 7.5, 10 and 12.5 mm.

- Unfolding the specimen is done after taking final reading and the mould is detached from the loading equipment.
- Similar procedure is to be adopted for other samples also.
- Average value of three samples are taken as CBR value.

CALCULATIONS:

Corresponding to 2.5 mm and 5 mm CBR is calculated as follows.

CBR = Load at corresponding 2.5 mm penetration*100/1370 (for 2.5mm)

CBR = Load at corresponding 5 mm penetration*100/1370*1.5 (for 5 mm)

CBR value of 5 mm penetration is taken as soil strength.

2) POPULATION OF NEARBY TOWNS:

Population of nearby towns less than 2kms are considered, populations of various towns are found by approaching local people and Gram panchayat offices**.** Sum of population nearby towns less than 2 kms gives total population of that stretch. All the stretches are to be considered for the calculating population. This is followed for all stretches to get populations for different stretches.

3) FACILITIES SCORE:

Facilities like educational institutional, Bus stops, banks, tourist attractions etc. are to be considered in each stretch. Score for each facility should be given either by considering journal references or using code book. Sum of the scores for all facilities in each stretch can be considered as "facility score". Different scores for different sectors have taken from the below table 4.1.

Table 4.1 Facilities Score for different stretches in study area

SECTOR	FACILITY	POINT
EDUCATIONAL	• Primary Elementary School (up to class 4)	05
	• Middle school or Junior Secondary (Class 5 to 8) • Matriculation or Secondary	10 15
	• School (Class 9 and 10 th) • Higher Secondary/Puc • College (graduate level and above) • Industrial School • Training School • Adult literacy class/Center • Others When Educational facility is available at a distance of: • Less than 5 kms • 5 to 10 kms • +10kms	20 25 30 50 50 20 15 10 05
COMMUNICATIONS	• Bus stops • Railway stops • Navigate Waterway When communications facility is available at a distance of: • Less than 5 kms	25 50 20 20 15

	• 5-10 kms • 10+kms	10
POST & TELEGRAPH	• Post Office • Telegraph Office • Post and Telegraph office Phone: Telephone connection when facility is available at a distance of • Less than 5kms • 5 to 10kms • 10+kms • Tube well • Hand Pump • River • Fountain • Canal • Lake • Spring • Nallah • Others When drinking water facility is available at distance of, • Less than 5kms • 5 to 10kms • 10+kms	25 25 50 75 20 15 10 30 30 25 20 20 20 20 10 05
Approach to the village	• Pucca Road • Kachha Road • Foot path • Navigable river • Navigable canal • Navigable Waterway	50 40 25 15 15 15

POWER SUPPLY	• Electricity for Domestic purpose only • Electricity for agriculture purpose only • Electricity for other purpose like, industrial commercial etc. • Electricity for all purpose listed above	50 50 50 50
LAND USE	CULTURABLE WATER • More culturable waste, Less points • Less culturable waste, More points Area not available for cultivation • More area, Less points • Less area, More points	- - - -
HOUSEHOLDS AND POPULATION	• More people, more households, more points • Less people, less number of households—less points	- -

4. DISTANCES TO NEAREST TOWN

Distances of nearest town must be considered. Assuming a midpoint on selected stretch as reference, distances from that midpoint to different towns must be noted. Smallest distances from all town's distances are to be considered. This distance is taken as nearest distance for that stretch.

5.TRAFFIC VOLUME:

Since the study locality is a sprawl area, the traffic movement is less to none of the roads expect for the roads which connect two heavily trafficked roads. The study period is taken for 3 hours and considered peak hour for further process.

Below are the steps to find out traffic volume of roads.

STEP-1: A reference line marked on the midblock is to be selected

STEP-2: When a vehicle is passed through the reference line, a tally mark is to be taken.

STEP-3: This process of counting vehicles is continued for one hour.

STEP-4: Similar process is to be done for three hours and peak hour is to be considered.

6. PAVEMENT DISTRESS:

Pavements distresses for each stretch has to be identified. Pictures of different types of cracks on different roads are to be taken from the site. Cracks are identified by visually and classified them into low, medium, high. This classification is done based on codebook and journals data. Different pavement distresses for different roads are given below.

Pavement distress for bituminous roads is different from cement concrete roads. Each has different criteria for cracks. These are classified into low, medium,high based on this table6.1 Various distress for both Rigid and flexible pavements are discussed below in table 4.2.

Table 4.2 Different types of distress and their classification

S.no	Type of Distress	Severity	Description
1	**cracking**	Low	Width of the cracking is less than 3mm.
		Medium	width of the cracking is greater than 3mm and less than 6mm
		High	Width of the cracking is greater than 6mm
2	**potholes**	Low	Depth of the pothole is less than 25mm
		Medium	Depth of the pothole is greater than 25mm and less than 50 mm
		High	Depth of the pothole is more than 50 mm
3	**Ravelling**	Low	The aggregate or binder has started to wear away but has not progressed significantly. The pavement appears onlyslightly aged and slightly rough.
		Medium	The aggregate or binder has worn away and the surface texture is moderately rough and pitted. Loose particles may be present and fine aggregate is partially missing
		High	The aggregate and/or binder have worn away significantly, and the surface texture is deeply pitted and very rough. Fine aggregate is essentially missing from the surface, and pitting extends to a depth approaching one half (or more) of the coarse aggregate size
4	**Patching**	Low	Patch has low severity distress of any type including rutting <6 mm; pumping is not evident
		Medium	Patch has moderate severity distress of any type or rutting from 6 mm to 12 mm; pumping is not evident.

		High	Patch has high severity distress of any type including rutting > 12 mm, or the patch has additional different patch materialwithin it; pumping may be evident.
5	**Rutting**	Low	Barely noticeable, depth less than 6 mm
		Medium	Readily noticeable, depth more than 6 mm less than 25 mm
		High	Definite effect upon vehicle control, depth greater than 25mm
6	**Edge Failure**	Low	Appearance of edge step with a few initial cracks on thebituminous surface along the edge portion of thecarriageway
		Medium	Appearance of edge step with a number of interconnected high intensity cracks on the bituminous surface along the edge portion of the carriageway
		High	Permanent loss of part of carriageway and pothole formation along the edge portion

7.GEOMETRIC CHARACTERISTICS:

7.1 CARRIAGE WIDTH:

The width of the carriage is measured on the road by visiting the site or using a GPS map. The number of lanes on the road must be considered. Each 100m of Cement Concrete Road had a different carriage width. The carriage width of a road is calculated by taking the average of all widths on that road.

7.2 SIGHT DISTANCES:

Sight distance is the distance along the centre line of the road that a driver can see an object that is stationary or moving at a set height above the carriageway.

The length of road visible ahead of the vehicle at any given time is known as the "sight distance." The sight distance standards should meet the following criteria:

- When driving at the design speed, the driver has enough sight distance or road length to safely stop the vehicle in the event of an obstruction on the road ahead.
- Drivers should be able to overtake slow moving cars at appropriate intervals without obstructing or endangering traffic in the opposite direction.
- When entering an uncontrolled crossing, the driver should have enough sight to see clearly.

4.2 PRIORITIZING PAVEMENT USING FUZZY MULTICRITERIA DECISION MAKING APPROACH(TOPSIS)

4.2.1 Data Collection

- Data collected from all the above-mentioned criteria like cracks, population, facilities score etc. are to be arranged in a tabular form.

4.2.2 Performing Normalisation Process to Form Rating Matrix

A simple normalisation is used to normalise data collected in the field on a scale of 0 to 100 with respect to the maximum value in the series if the maximum value needs to be prioritized as highest and with respect to the minimum value, if the minimum value is prioritized as highest.

Normalised Data Point = (Data point) *100/ (Maximum of the Data Series)

These normalised data values obtained are to be arranged in a table. Ratings are to be to be given for each value in each column. The rating criteria for all normalised values ranging from 0-100 are given below

Table 4.3 Ratings for the Normalised Values

Normalised Matrix	0-10	11-20	21-30	31-40	41-50	51-60	61-70	71-80	81-90	91-100
Rating	1	2	3	4	5	6	7	8	9	10

These ratings are to be arranged into a matrix called a "rating matrix" (R), with each row representing an alternative (A) and each column representing a criterion.

4.2.3 Collecting Experts Opinions for Pavement Distresses in each stretch

Criteria values alone are not adequate to find out the best site. To arrive at the objective, weightage of sites must be known. The opinion of selected experts has been sought to ascertain the influence of different distress parameters on the functional condition of the pavement. The distresses considered are cracking, potholes, ravelling etc., with respect to three severity levels namely low, medium, and high. A questionnaire was prepared and sent to selected experts for this purpose. Further, they were asked to indicate their preferences regarding the influence of severity of various distress parameters in terms of linguistic variables such as Negligible (N), Low (L), Moderate (M), High (H) and Very High (VH)

4.2.4 Determination of Fuzzy Weights for experts opinions

From the expert data collected, linguistic variables (negligible, low, medium, high, very high) themselves aren't enough to find the weightages. The severity of various distresses has been expressed using Triangular Fuzzy Numbers (TFN). The TFNs for various linguistic variables are as listed below in table 4.4. These variables are needed to be converted into fuzzy numbers and these fuzzy numbers are expressed in fuzzy weights. Fuzzy weights for all the criteria can be expressed in a row matrix

$$\widetilde{\boldsymbol{W}}=[\widetilde{\boldsymbol{w_1}},\widetilde{\boldsymbol{w_2}}\ldots\ldots\ldots\ldots\ldots\widetilde{\boldsymbol{w_N}}]$$

Where $\widetilde{w_1},\widetilde{w_2}\ldots\ldots\ldots\widetilde{w_N}$ are the fuzzy weights for all criteria expressed in Triangular Fuzzy Numbers

i.e., $\widetilde{w_j} = (\widetilde{w_{j1}}, \widetilde{w_{j2}}, \widetilde{w_{j3}}) \quad \forall$ j = 1,2,3,4, ……………N

All obtained values are arranged in a table as stretch and its corresponding weightage.

Table 4.4 Triangular Fuzzy Numbers for various Linguistic Variables

LINGUISTIC VARIABLE	TFN
Negligible	(0,0,0.1)
Low	(0,0.1,0.3)
Medium	(0.3,0.5,0.7)
High	(0.7,0.9,1)
Very High	(0.9,1,1)

4.2.5 Determination of Fuzzy Evaluation Values

Fuzzy evaluation value can be evaluated by multiplying Rating matrix and weightage matrix obtained from experts' surveys, The expression can be expressed in mathematical terms as follows as

$$\widetilde{P_i} = \sum_{j=1}^{N} R_{ij} \times \widetilde{w_j}, \forall\, i = 1, 2, 3, 4 \ldots\ldots\ldots\ldots\ldots N \text{ and } \forall\, j = 1, 2, 3, 4 \ldots\ldots\ldots. N$$

The fuzzy evaluation values are arranged as stretch and its corresponding fuzzy evaluation value in a tabular form.

4.2.6. Determination of Triangular Fuzzy Numbers

A relative preference is to be evaluated for all the stretches. This can be obtained by obtaining differences among all the combinations of all fuzzy values. These can be expressed in mathematically as shown below.

$$\widetilde{F_{ij}} = (\widetilde{p_i} - \widetilde{p_j}) \qquad \forall\ \text{i= 1 to N} \qquad \forall\ \text{j = 1 to N and i} \neq \text{j}$$

Hence this difference of the combinations can be expressed as Fuzzy numbers. All obtained values from mathematical expression are arranged in table. The subtraction operation on the two fuzzy numbers is performed as:

$$(l, m, n)\ \Theta\ (p, q, r) = (l - r, m - q, n - p)$$

As $\widetilde{p_i}$ and $\widetilde{p_j}$ are triangular fuzzy numbers. So $(\widetilde{p_i} - \widetilde{p_j})$ are also known as Triangular Fuzzy numbers.

4.2.7 Determination of Degree of Preference of Matrix of one stretch over other stretch

- To obtain degree of preference matrix of one stretch over another, the fuzzy preference relation matrix (E) has to be developed.

$$\mathbf{E} = \begin{bmatrix} \boldsymbol{e_{11}} & \boldsymbol{e_{12}} \dots\dots\dots. & \boldsymbol{e_{1N}} \\ \boldsymbol{e_{21}} & \boldsymbol{e_{22}} \dots\dots\dots\dots & \boldsymbol{e_{2N}} \\ \boldsymbol{e_{N1}} & \boldsymbol{e_{N2}} \dots\dots\dots. & \boldsymbol{e_{NN}} \end{bmatrix}$$

The real number e_{ij} represents the degree if preference between the respective i^{th} and j^{th} pavement stretches. It can be determined from both positive (S^{+}_{ij}) and negative area ($|S^{-}_{ij}|$) of the difference between two fuzzy values.

$$\mathbf{eij} = \frac{\mathbf{S^{+}ij}}{\mathbf{S^{+}ij + \ |S^{-}\ ij|}} = \frac{\textbf{Positive area}}{\textbf{Total area}}$$

Where, ($S^{+}_{ij} + |S^{-}_{ij}|$) = Total area of ($\widetilde{p_i} - \widetilde{p_j}$).

- This positive and negative areas are obtained from membership functions of $(\widetilde{p_i - \widetilde{p_j}})$.
- The values are arranged in a matrix form. Here $e_{ii} = 0.5$ and $e_{ij} + e_{ji} = 1.0$. If $e_{ij} > 0.5$ the stretch A_i is to be given priority over stretch A_j and vice versa.

4.2.8 Determination of Priority Index

Calculation for Priority Index for all pavement stretches.

- This Priority Index can be obtained from a mathematical expression.

Priority Index (PI) = $\boldsymbol{\sum_{j=1}^{n}(e_{ij} - 0.5)}$

All obtained values after performing above calculations on the fuzzy preference relational matrix (E) are to be noted in a table. Ranking has to be given according to high to low value. Maximum value obtained gets the least rank 1 i.e., worst case and highest value will get the rank 1 (is in need to be repaired immediately).

To perform these fuzzy operations such as fuzzy weights computation, TFN, preference matrix (E), manual calculations are very difficult and time consuming. So here in this study we used python programming language to write an algorithm in order to perform the calculations. We developed a python code to Fuzzy TOPSIS approach to give priority index based on the inputs provided. This code been placed in appendix.

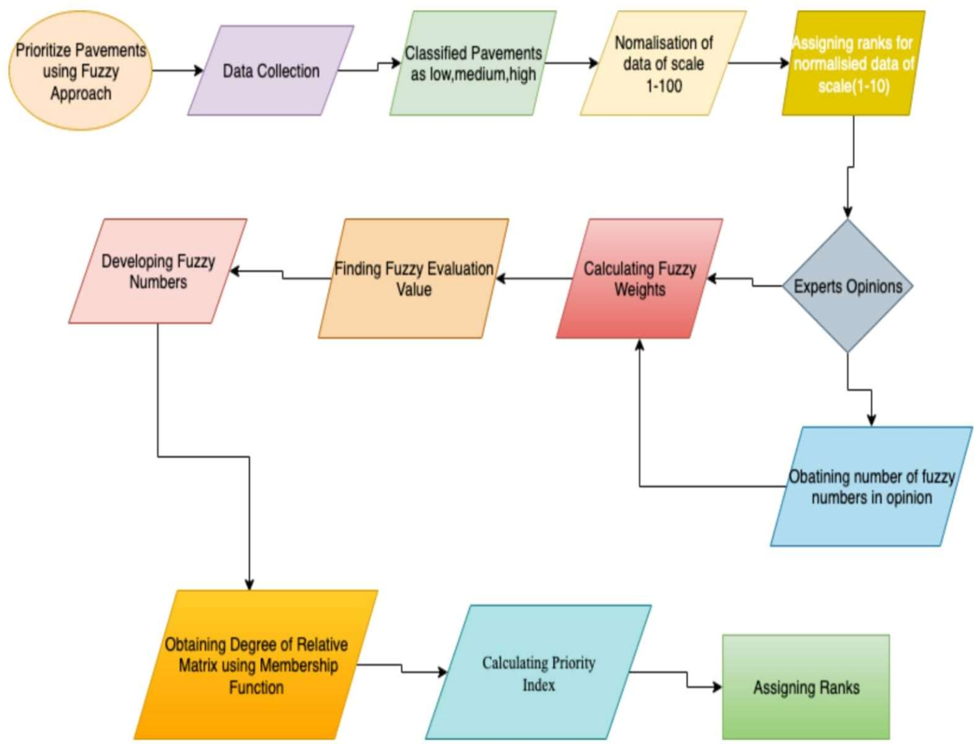


Fig 4.1 Flowchart for methodology of fuzzy MCDM

4.3 PRIORITIZING PAVEMENTS USING ANALYTICAL HEIRACHY PROCESS

4.3.1 Developing Pairwise Comparison Matrix

The pairwise comparison matrices (PCM) are used to compute relative preferences or the priorities of the alternatives with respect to the criteria. They are needed to obtain the weightages of each criterion (or) the criteria weights. To obtain these Pairwise matrices, 15 experts specialized in transportation engineering have been interviewed and their preferences of criteria in the form of a ranks which are further converted into pair wise matrices, which are collected by using Satty's scale, containing ratings from 0 – 9 as shown in table 4.5. The relative preference of criteria in a ranking order is done before putting the values into the pairwise matrix.

Table 4.5 Pair wise comparison matrix scale (Satty's preference scale)

Intensity of Importance	Definition	Explanation
1	Equal Importance	Two activities contribute equally to the objective
2	Weak	
3	Moderate Importance	Experience and judgement slightly favor one activity over another
4	Moderate plus	
5	Strong Importance	Experience and judgment strongly favor one activity over another
6	Strong plus	
7	Very Strong	An activity is favoured very strongly over another
8	Very very Strong	
9	Extreme Importance	The evidence favouring one activity over another is of the highest possible order of affirmation

2,4,6,8 – Intermediate values between two judgements generally taken when compromise is needed.

Reciprocals of above nonzero If activity 'i' has one of the above non-zero numbers assigned to it when compared with activity 'j' as in table 4.5, then j has the reciprocal value when compared with i.

The diagonal elements are always 1.

4.3.2 Determination of Combination Pairwise Matrix

The pairwise matrices collected need to be combined into one single matrix. The steps involved in this as follows.

STEP-1: The geometric mean is calculated to each value with their value in corresponding positions in all the expert matrices. The geometric mean is calculated by the formula as shown below.

$$\mathbf{a_{ij} = (a^1_{ij} + a^2_{ij} + a^3_{ij} + a^4_{ij} + \ldots\ldots\ a^k_{ij} + \ldots\ldots\ a^h_{ij})^{1/h} = (\Sigma^{h}\ _{k=1} = a^k_{ij}\ )^{1/h}}$$

$$A = \begin{pmatrix} a11 & a12\ \ldots\ldots & a1n \\ a21 & a22\ \ldots\ldots & a2n \\ \vdots & \vdots & \vdots \\ an1 & an2\ \ldots\ldots & ann \end{pmatrix} \quad \text{or} \quad C = \begin{pmatrix} C11 & C12 & C13 \\ C21 & C22 & C23 \\ C31 & C32 & C33 \end{pmatrix}$$

Where, k = Expert 1,2,3......n
C is combined pairwise combination matrix.

4.3.3 Determination of Normalised Pairwise Matrix

For the combined pairwise matrix obtained, the weights are determined by normalizing the eigen vector associated with maximum eigen value i.e., each value under its own criteria column is considered and is divided by the sum the whole column, for each criterion, in order to normalize the matrix. The normalized pairwise matrix is obtained as shown below.

$$\text{Normalization of pairwise matrix} = \frac{\text{Criteria value}}{\text{Sum of all the values under the criteria column}}$$

$$C = \begin{pmatrix} X11 & X12 & X13 \\ X21 & X22 & X23 \\ X31 & X32 & X33 \end{pmatrix} \qquad X_{ij} = \frac{Cij}{\Sigma\, Cij}\ ; \quad i = 1 \text{ to } n$$

4.3.4 Determination of weighted performance for each alternative with respect to each criterion

Using the combined pair wise comparison matrix, weightages of criteria can be found by dividing the sum of normalized row of matrix by number of criterion (n) i.e., by finding row wise mean to generate weighted matrix as shown below.

$$W = \begin{pmatrix} W11 \\ W21 \\ W31 \end{pmatrix} \qquad W_{ij} = \frac{\Sigma Xij}{n}\ ; \quad i = 1 \text{ to } n$$

4.3.5 Determination of Consistency ratio:

This check estimates whether the comparisons of pair wise matrix are consistent or not. There are several steps the Pairwise matrix has to undergo to meet the checks' goal i.e., the values obtained should not contain any inconsistency. The consistency ratio (CR) must be attained such that CR < 0.1 and is calculated as follows:

STEP - 1 Calculation for consistency check matrix is done by multiplying the un-normalized pairwise matrix criterias with the corresponding criteria weightages obtained.

STEP Calculation of Consistency vector is done by dividing the weighted sum with criterion weight

STEP- 3: The weight sum values obtained from the previous step are divided by the corresponding criteria weights.

Steps 1,2,3 are achieved as shown below:

$$\begin{pmatrix} C11 & C12 & C13 \\ C21 & C22 & C23 \\ C31 & C32 & C33 \end{pmatrix} * \begin{pmatrix} W11 \\ W21 \\ W31 \end{pmatrix} = \begin{pmatrix} Cv11 \\ Cv21 \\ Cv31 \end{pmatrix}$$

$\mathbf{C_{v11} = 1/W_{11}\ [C_{11}W_{11} + C_{12}W_{21} + C_{13}W_{31}]}$

$\mathbf{C_{V22} = 1/W_{21}\ [C_{21}W_{11} + C_{22}W_{21} + C_{23}W_{31}]}$

$\mathbf{C_{V33} = 1/W_{31}\ [C_{31}W_{11} + C_{32}W_{21} + C_{33}W_{31}]}$

STEP - 4: Determination of λ_{max}, Consistency Index (CI), Random Index (RI) and Consistency Ratio (CR):

1. The λ_{max} is calculated by averaging the value of consistency vector i.e., by adding all the values of WSV divided by number of criteria weights.
2. The calculation of CI is based on the observation that λ_{max} is always greater than or equal to the number of criteria under consideration (n) for positive, reciprocal matrices and $\lambda = n$, if the pair wise comparison matrix is a consistency matrix.

   The consistency provides a measure of deviation from consistency.

   CI = $\frac{\boldsymbol{\lambda - n}}{\boldsymbol{n - 1}}$ **;** n is number of criteria used.
3. Random Index (RI) are defined standard, for number of criteria present in the data. The random indexes are obtained from the Satty's random index as shown in table 4.6.

Table 4.6 Saaty's random index

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0	0	0.58	0.9	1.12	1.24	1.32	1.41	1.45	1.49
11	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1.53	1.54	1.56	1.57	1.59	1.6	1.61	1.61	1.62	1.63

4. The Consistency Ratio (CR) is obtained by dividing the consistency index by random index. The consistency ratio is designed in such a way that if $CR < 0.10$, the ratio indicates considerable level of consistency in the pairwise comparisons. If however, $CR \geq 0.10$, the values of the ratio are indicative of

inconsistent judgements. In such cases, the original values in the pairwise comparison matrix must be reconsidered and revised. CR is calculated as follows:

5. **Consistency Ratio = Consistency Index / Random Index**

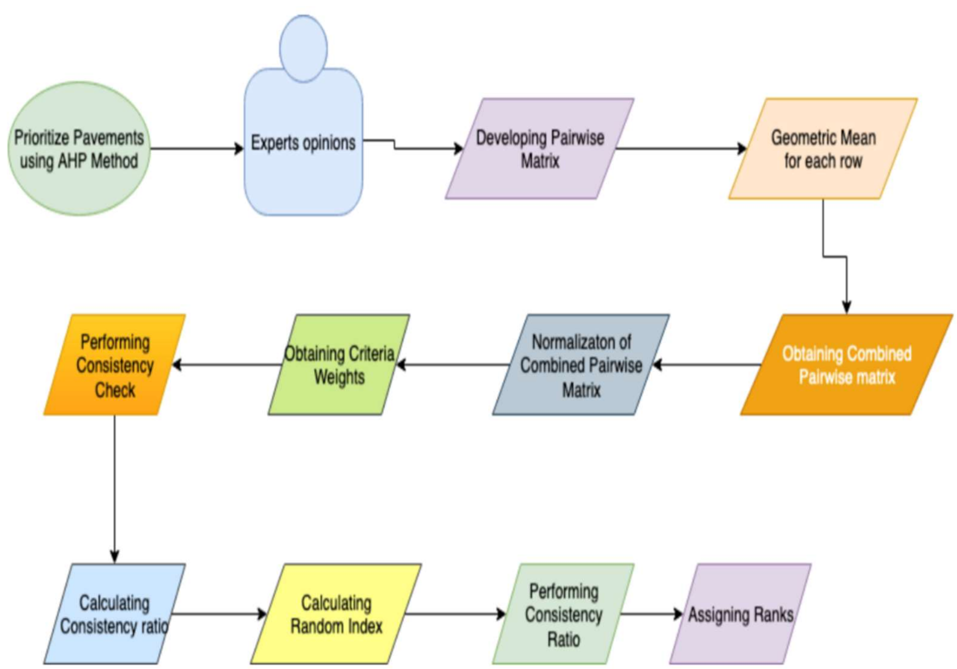
4.3.6 Normalization of Raw Data collected from field study

The raw data from field are normalized on a scale of 0 – 100 and are combined and converted to the main set of criteria, that can be utilized for further operations.

4.3.7 Assigning ranks to the criterias

The normalized data obtained are multiplied with the consistency check satisfied pairwise matrix weightages as a dot product to obtain the priority index (PI) for corresponding stretches. For these priority indices, ranks are assigned based on the maximum value, i.e., greater the value least will be the rank, thus more importance should be given to that stretch.

Fig 4.2 Flowchart for methodology of AHP



4.4 PRIORITISATION OF ROADS USING CONCORDANCE METHOD

4.4.1 Collection of Data:

The raw data obtained from the field are taken and are combined to convert them into main criteria set.

4.4.2 Determination of maximum and minimum value and mathematical expression.

Maximum and Minimum value from each criteria same calculations are to be calculated to every criterion i.e., each column. The difference between maximum and minimum value is calculated as well.

4.4.3 Normalisation of the obtained Raw Data by Linear Transformation:

The linear transformation from 0 to 1 or the logarithmic transformation is one of many types of normalization methods that exist. For concordance we apply Linear function transformation such that the values range in between the range 0 to 1. This can be calculated in such a way that

$$\text{New column value} = \frac{\text{Old column value} - \text{Min value of that column}}{\text{Maximum value of the column} - \text{minim value of the column}}$$

4.4.4 Importing Weightages From AHP:

The Weightages calculated in AHP method as in chapter 4.4.4 are obtained for further operations.

4.4.5 Determination of Relative Comparison of Stretches and Row wise Summation:

A relative preference is to be evaluated for all the stretches. This can be calculated by taking two stretches at once iteratively, and that for m1 - m2 comparison if m1 > m2 then weightage value should be assigned else if m1 < m2 then zero is assigned.

Similar process is done for all the other values. The size of the matrix at this time is n x n rows, where n is total number of stretches and x columns, where x is total number of criteria. After assigning the weightages, the values of each row are added up and sum values are obtained. After assigning the weightages, the values of each row are added up and n x n sum values are obtained.

4.4.6 Obtaining Ranking Matrix

The n x n sum values obtained from the relative matrix are arranged in a n x n matrix, where n is total number of criteria. The matrix is arranged in such a way that if m_i-m_j value is compared, the value should be arranged in a_{ij} position in the ranking matrix.

4.4.7 Determination of Priority Index from Rating matrix:

The row wise summation for rating matrix is performed to obtain the prioritization index (PI) and using these prioritization indices, the ranks are assigned such that the greatest value obtained will be assigned to the least rank 1.

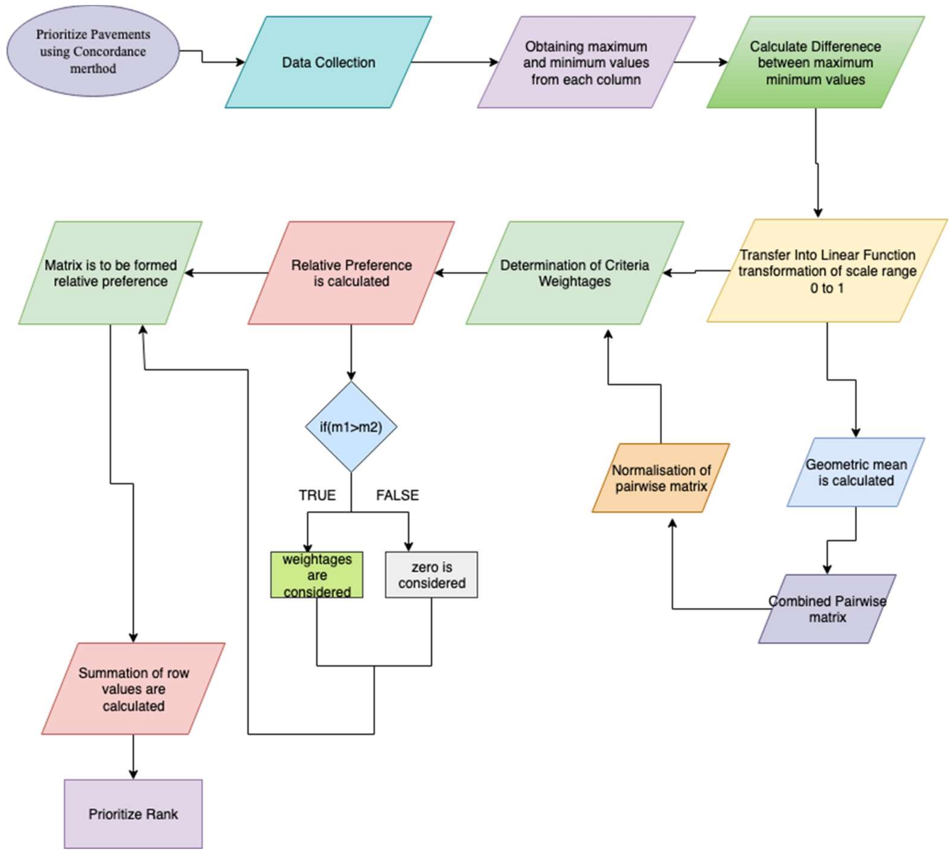


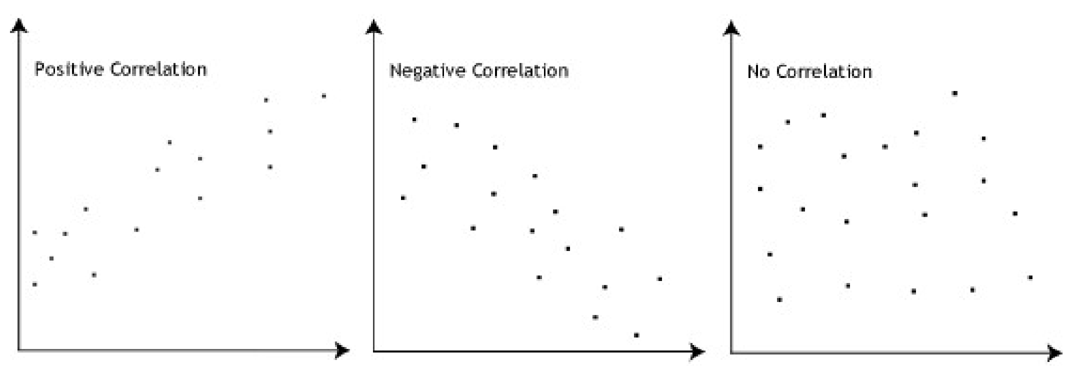
Fig 4.3 Flowchart for methodology of Concordance

4.5 SENSIVITY TEST

The objective of doing this test is to show that the results obtained through the methods used i.e., Fuzzy TOPSIS, AHP, and Concordance are not sensitive to the increase and decrease in the weightage values at a stipulated rate. In this test, all the weightages from the methods are taken and are increased and decreased by 5% and 10% iteratively. The obtained ranks are then compared to the original ranks and thus if the values do not show any variation, then the results obtained are not sensitive to errors that may have been incorporated without knowing. This process is done for all the three methods chosen for their corresponding weightages and compared to check the values if they are sensitive. The sensitivity test results are compared using the Pearson's correlation coefficient (r).

4.6 PEARSON CORRELATION COEFFICIENT (r):

he Pearson correlation method is used to determine the relationship between two sets of variables. It employs a coefficient known as Pearson's correlation coefficient, denoted by "r." The Pearson correlation coefficient r can range from +1 to -1. A value of 0 indicates a positive association; that is, as the value of one variable increases, so does the value of the other variable. A value less than 0 indicates a negative association; that is, as the value of one variable increases, the value of the other variable decreases. The diagram below depicts this.



The greater the link between the two variables, the closer the Pearson correlation coefficient ‘r’ is to +1 or -1, depending on whether the relationship is positive or negative.

4.7 PERFORMING MCDM, AHP, CONCORDANCE METHODS AND SENSIVITY ANALYSIS USING PYTHON APPROACH.

It is difficult task to perform all the operation manually, since the data taken can humongous and can require calculations that might take ages if one chose to do. Thus, an algorithm is needed to make the tasks easier, flexible and user friendly. For this to be achieved, codes were written for each method to compute the tasks automated and make the process easier, we developed a python code to calculate ranks for different methods using several code lines, methods and libraries. Code had been developed in such a way, to do all calculations automatically and give the output as priority index values and with their respective ranks.

4.8 LIBRARIES AND METHODS USED IN ALGORITHM:

4.8.1 Pandas:

Pandas is mainly used for data analysis and associated manipulation of tabular data in Data frames. It is a fast, powerful, flexible, and easy to use open-source data analysis and manipulation tool, built on top of the Python programming language. It has functions for analysing, cleaning, exploring, and manipulating the data. Pandas allows us to analyse big data and make conclusions based on statistical theories. It can clean messy data sets and make them readable and relevant. Pandas allows importing data from various file formats such as comma-separated values, JSON, Parquet, SQL database tables or queries, and Microsoft Excel.

To import this library, use **import pandas as pd.**

4.8.2 Numpy:

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. It is a primarily used for working with arrays. Using NumPy, mathematical and logical operations on arrays can be performed smoothly. It also has functions for working in domain of linear algebra, Fourier transform, and matrices.

In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important.

To import this library, use **import numpy as np.**

4.8.3 Math:

Python has a set of built-in math functions, including an extensive math module, that allows to perform mathematical tasks on numbers. The math module has a set of inbuilt methods and constants. Math module provides values of various constants like pi, tau, infinity etc. It also provides numeric functions to calculate floor, ceil, logarithmic values, GCD, square root, power, trigonometric functions, angular functions and so on.

To import this library, use **import math.**

4.8.4 Matplotlib:

One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib is one such comprehensive library for creating static, animated, and interactive visualizations in Python. It is a plotting library for Python and its numerical mathematics extension NumPy and is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays. It provides an object-oriented API that helps in embedding plots in applications using Python GUI toolkits such as PyQt, WxPythonotTkinter. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

To import this library, use **import matplotlib.**

- **PYPLOT:** matplotlib.pyplot is a state-based interface to matplotlib. It provides an implicit, MATLAB-like, way of plotting. It also opens figures on the screen, and acts as the figure GUI manager. pyplot is mainly intended for interactive plots and simple cases of programmatic plot generation. In each pyplot function various states are preserved across function calls, so that it keeps track of things like the current figure and plotting area, and the plotting functions are directed to the current axes.

  To access this module, use **import matplotlib.pyplot as plt.**

- **PATCHES:** Patches are arbitrary two-dimensional regions. There are a lot of commonly useful wrappers and helpers, like Rectangles, Circles, Boxes, and Ellipses. In order to follow a single approach that will meet nearly all the user needs, **Polygons** are the best. They let you define the outline of any two-dimensional shape you want, no matter how complex.

  To access the Polygon function, use **from matplotlib.patches import Polygon.**

- **%matplotlib inline –** This line sets the backend of matplotlib to the 'inline' backend: With this backend, the output of plotting commands is displayed inline within frontends like the Jupyter notebook, directly below the code cell that produced it. The resulting plots will then be stored in the notebook document.

4.8.5 Decimal:

Python interprets any number that includes a decimal point as a double precision floating point number. The Decimal module contains floating decimal point type functions with more precision and a smaller range than the float. This decimal is used to do some decimal floating point related tasks by providing correctly-rounded floating point arithmetic.

To access this module, use **from decimal import Decimal.**

- **GETCONTEXT:** Contexts are environments for arithmetic operations. They govern precision, set rules for rounding, determine which signals are treated as exceptions, and limit the range for exponents. **decimal.getcontext()** - Return the current context for the active thread.

To access this module, use **from decimal import getcontext.**

4.8.6 Collections:

The collection Module in Python provides different types of containers. A Container is an object that is used to store different objects and provide a way to access the contained objects and iterate over them. Some of the built-in containers in python are Tuple, List, Dictionary, etc.

- **COUNTER:** A Counter is a subclass of dict. It is an unordered collection where elements are stored as Dict keys and their count as dict value. Counter elements count can be positive, zero or negative integers.

To access this module, use **from collections import Counter.**

4.8.7 Itertools:

Itertools is a module in Python, it is used to iterate over data structures that can be stepped over using a for-loop. Such data structures are also known as iterables. This module works as a fast, memory-efficient tool that is used either by themselves or in combination to form iterator algebra.

- **GROUPBY:** This method calculates the keys for each element present in iterable. It returns key and iterable of grouped items. The first value of tuple consists of keys, on which the items of iterable were grouped. The second value of the tuple will be an iterator that contains all the items grouped by the key.

To access this module, use **from itertools import groupby.**

4.8.8 Scipy:

SciPy's high level syntax makes it accessible and productive for programmers from any background or experience level. It provides algorithms for optimization, integration, interpolation, eigenvalue problems, differential equations, statistics and many other classes of problems. The algorithms and data structures provided by SciPy are broadly applicable across domains. SciPy extends NumPy providing additional tools for array computing and provides specialized data structures, such as sparse matrices and k-dimensional trees.

- **STATS:** All the statistics functions are located in the sub-package scipy.stats and a fairly complete listing of these functions can be obtained using **info(stats)** function. This module contains a large number of probability distributions as well as a growing library of statistical functions. **gmean** calculates the geometric mean of the values in the array passed.

To access the gmean function, use **from scipy.stats import gmean.**

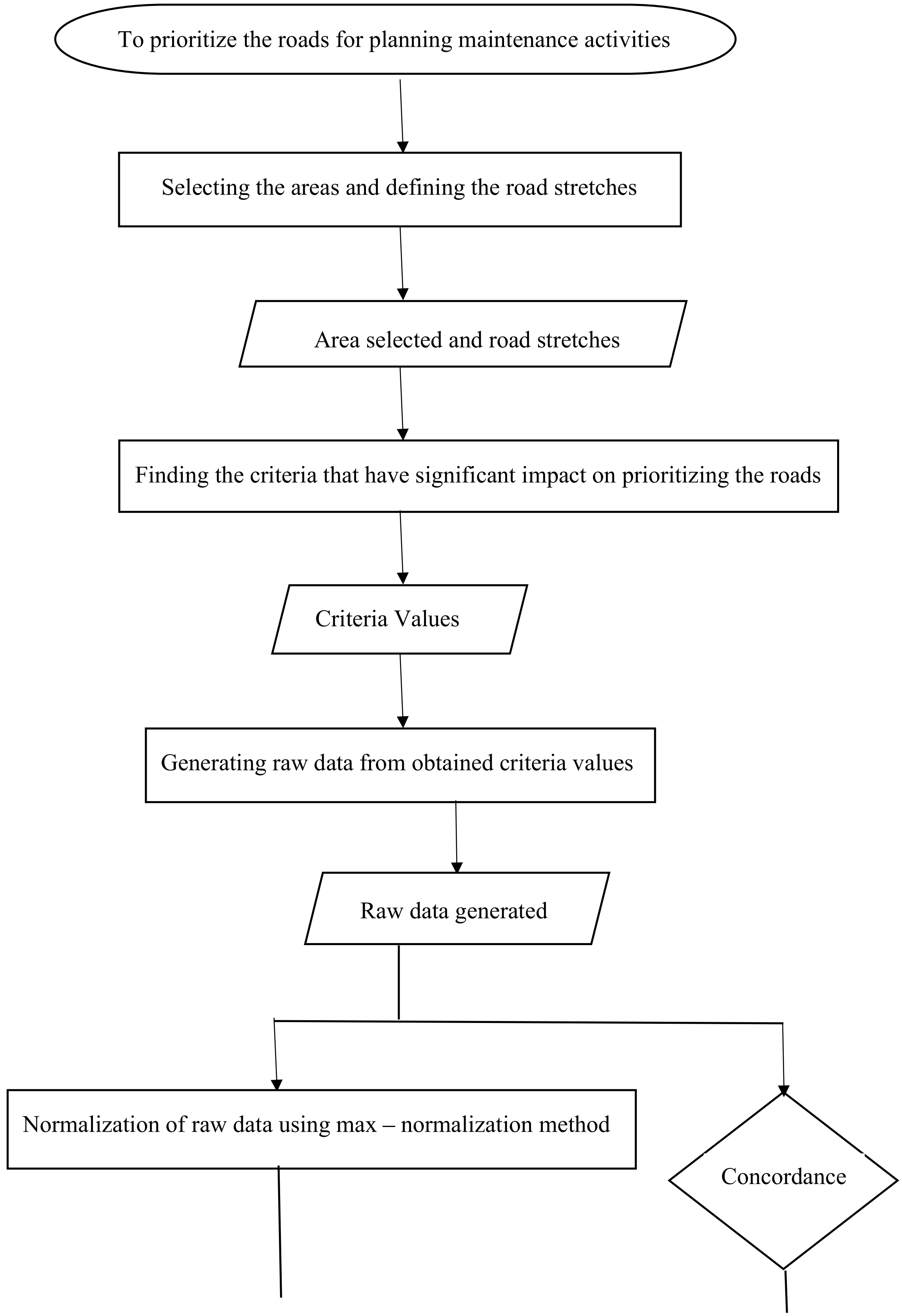
4.8.9 Sympy:

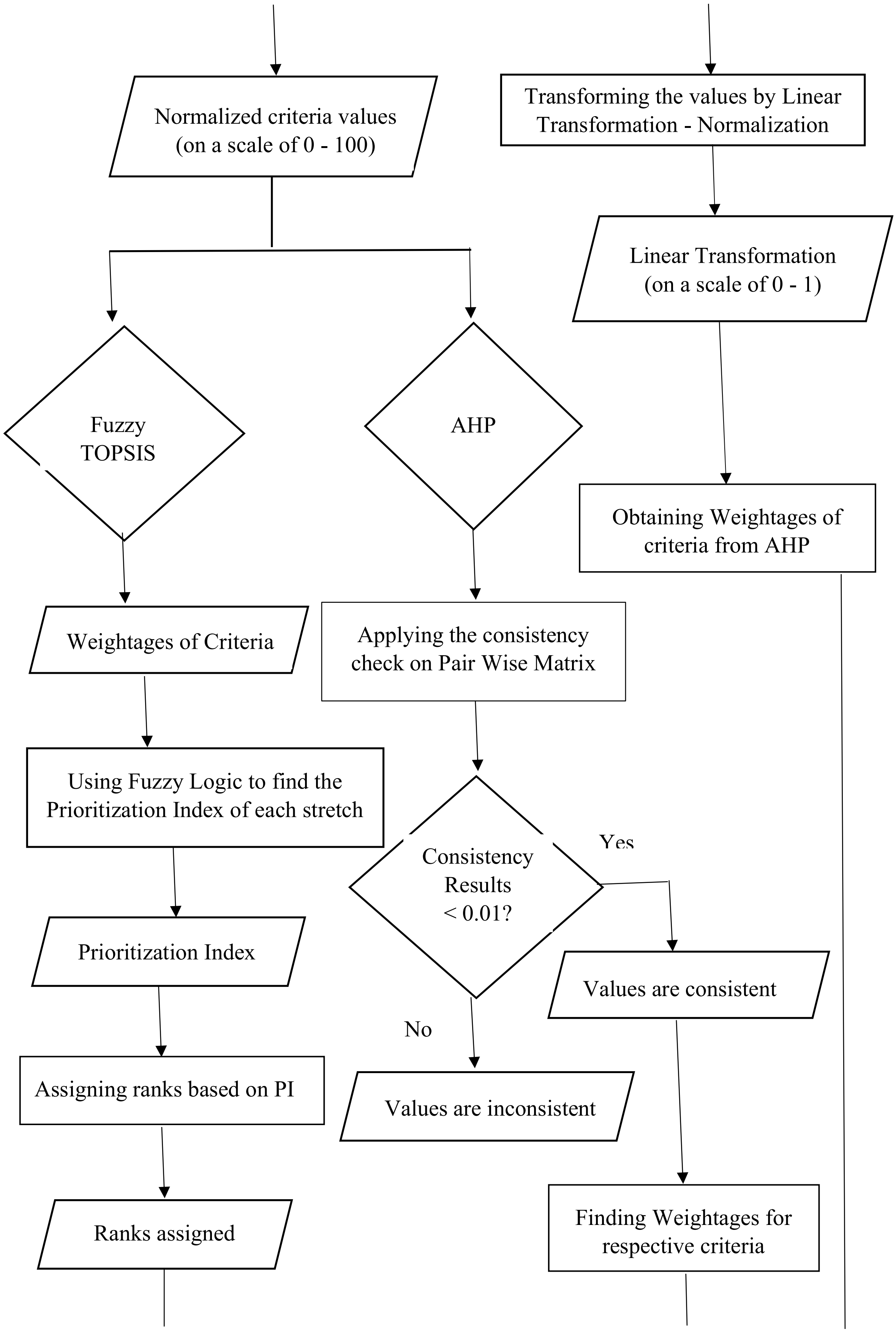
SymPy is a Python library for symbolic mathematics. It aims to become a full-featured computer algebra system (CAS) and an alternative to systems such as Mathematica or Maple while keeping the code as simple as possible in order to be comprehensible and easily extensible.

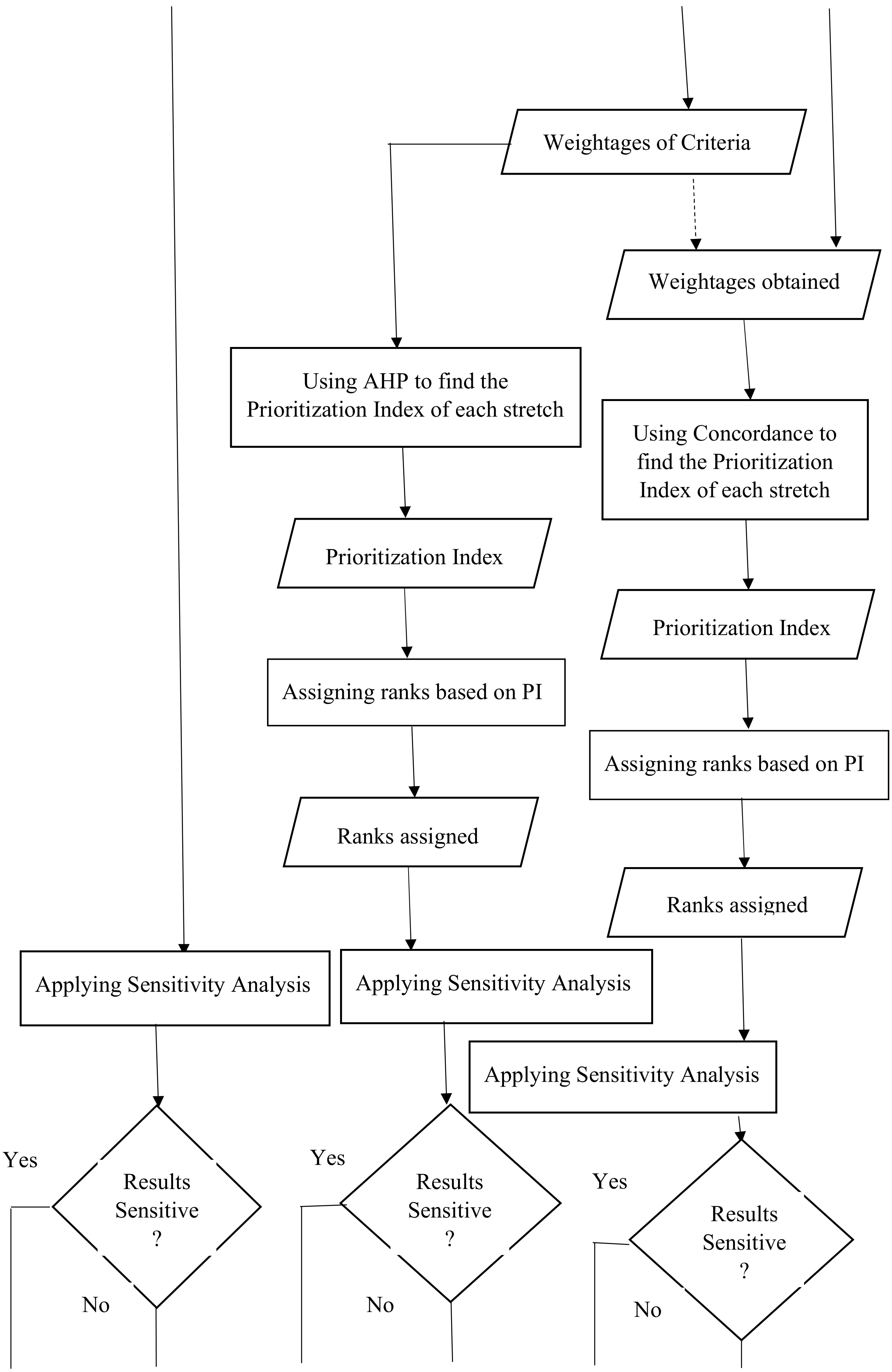
- **POINT, LINE & SEGMENT:** A point is a 1D entity in an n-dimensional Euclidean space. A line is base class for all linear entities in the same n-dimensional Euclidian space. A Segment is a function that returns joins of the point specified and the intersection of the bisector and the segment.

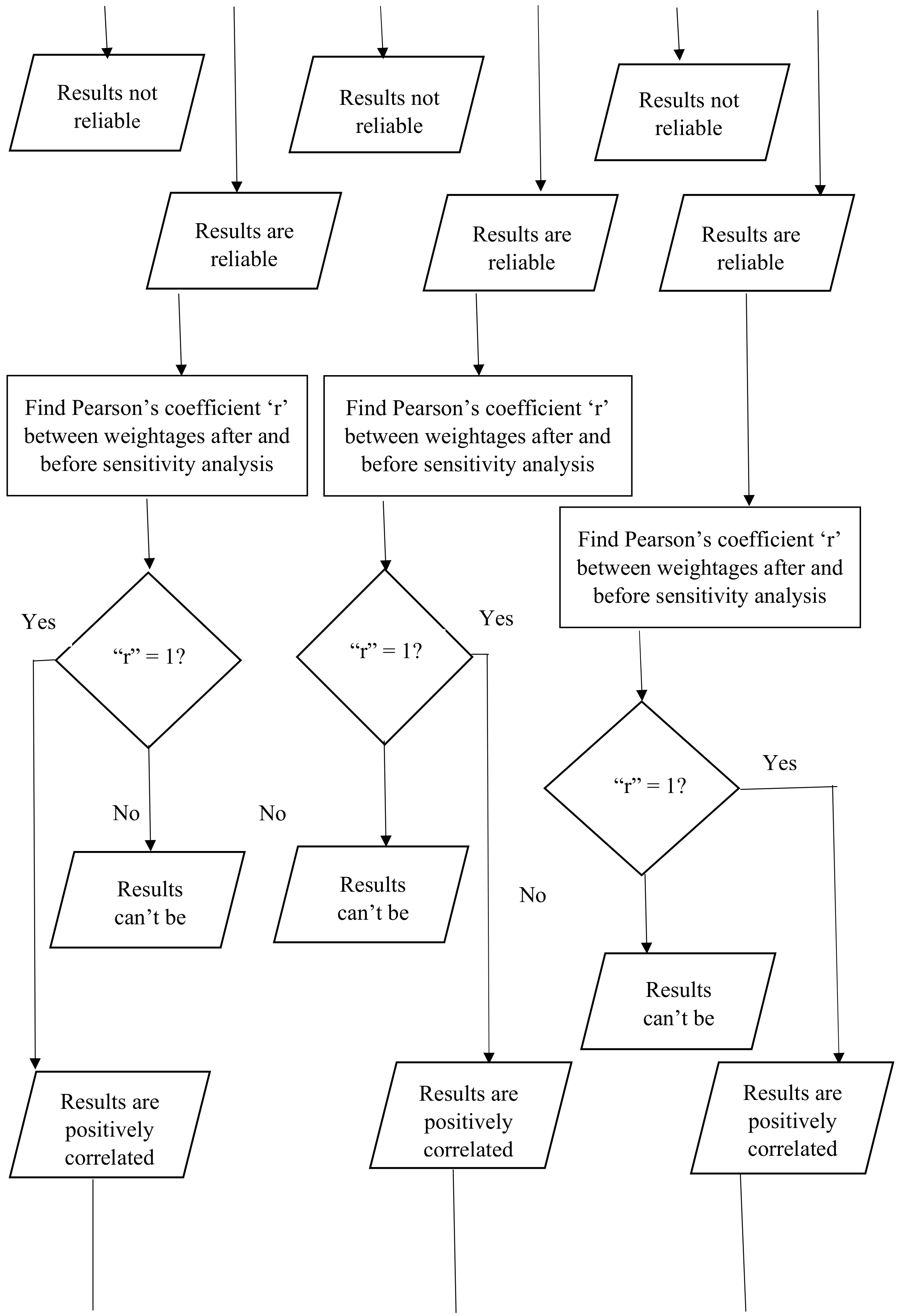
To access all these three modules at once, use **from sympy import Point, Line, Segment.**

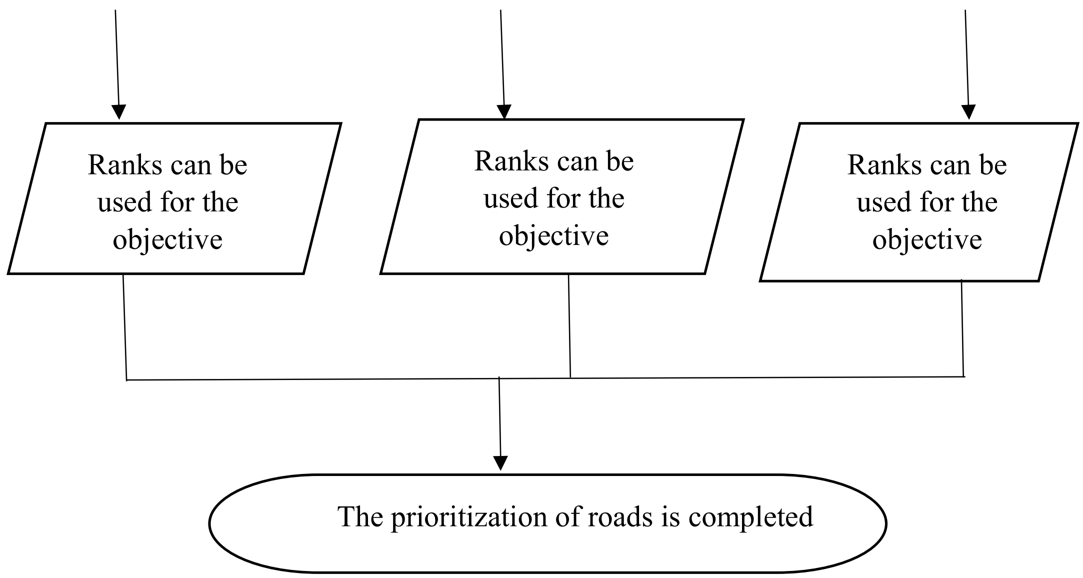
4.9 FLOWCHART




Normalized criteria values
(on a scale of 0 - 100)
Transforming the values by Linear Transformation - Normalization
Linear Transformation
(on a scale of 0 - 1)
Fuzzy TOPSIS
AHP
Obtaining Weightages of criteria from AHP
Weightages of Criteria
Applying the consistency check on Pair Wise Matrix
Using Fuzzy Logic to find the Prioritization Index of each stretch
Consistency Results < 0.01?
Yes
Prioritization Index
Values are consistent
No
Assigning ranks based on PI
Values are inconsistent
Ranks assigned
Finding Weightages for respective criteria


Weightages of Criteria
Weightages obtained
Using AHP to find the Prioritization Index of each stretch
Using Concordance to find the Prioritization Index of each stretch
Prioritization Index
Prioritization Index
Assigning ranks based on PI
Assigning ranks based on PI
Ranks assigned
Ranks assigned
Applying Sensitivity Analysis
Applying Sensitivity Analysis
Applying Sensitivity Analysis
Yes
Results Sensitive ?
No
Yes
Results Sensitive ?
No
Yes
Results Sensitive ?
No


Results not reliable
Results not reliable
Results not reliable
Results are reliable
Results are reliable
Results are reliable
Find Pearson's coefficient 'r' between weightages after and before sensitivity analysis
Find Pearson's coefficient 'r' between weightages after and before sensitivity analysis
Find Pearson's coefficient 'r' between weightages after and before sensitivity analysis
Yes
"r" = 1?
Yes
"r" = 1?
Yes
"r" = 1?
No
No
No
Results can't be
Results can't be
Results can't be
Results are positively correlated
Results are positively correlated
Results are positively correlated


Ranks can be used for the objective
Ranks can be used for the objective
Ranks can be used for the objective
The prioritization of roads is completed

CHAPTER 5

APPLICATION OF METHODOLOGY

5.1 GENERAL:

The main objective of the project is to prioritize the pavements of different roads. The stretches have been chosen for the study is sprawl areas of Hyderabad. The use of GIS application and the concept of Analytical Hierarchy Process, Fuzzy Logic, Concordance are the important tools for the analysis. The process of the collection of data, computations and analysis have been mentioned in detail in the following steps.

5.2 DETERMINATION OF CRITERIA VALUES

In chapter 4, the definitions of criteria have already been explained. How they are calculated is explained here. Major part of the calculations are done using various models like AHP, Fuzzy, Concordance, Python code. The detailed explanation of each criterion has been described as following steps.

5.2.1 STRENGTH OF THE SOIL.

Strength of soil is calculated by performing CBR test. Each stretch has their own strength. The average of three sample strength is taken as strength of soil. The table 5. 1 below shows the observations of the test during experiment for strecth-1 sample.

Table 5.1: Observations of CBR test experiment values for stretch 1

STRECTH NO.	PENETRATION(MM)	LOAD(KGF)
1	0	0
2	0.5	1.9
3	1	2.4
4	1.5	2.5

5	2.0	2.6
6	2.5	2.7
7	3	2.9
8	4	3.7
9	5	4.1
10	7.5	5.4
11	10	6.4
12	12.5	7.4

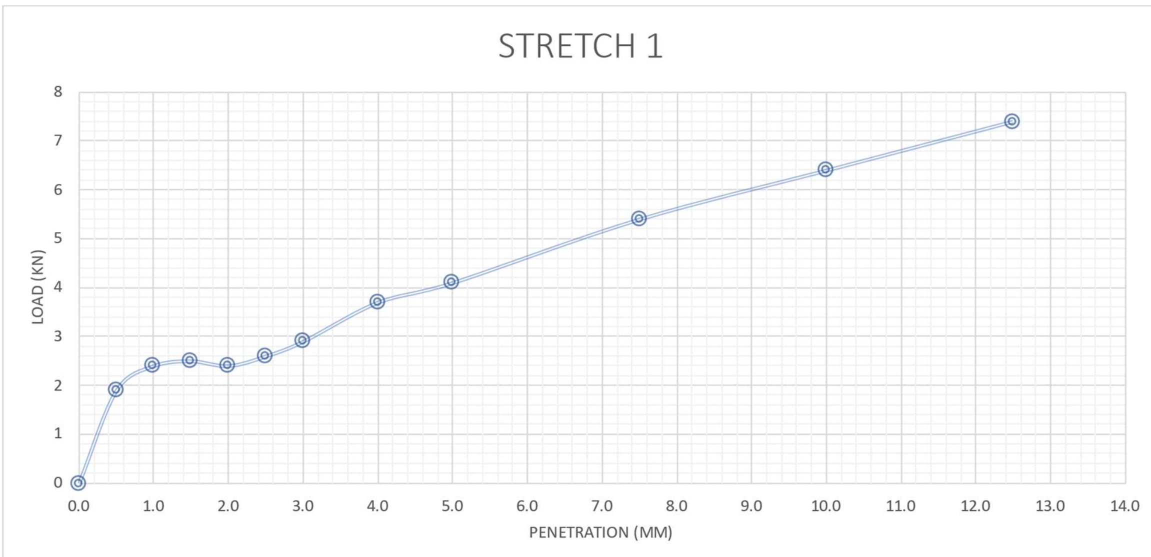


Fig 5.1 Penetration vs Load graph for CBR test of stretch 1 sample

CALCULATIONS:

For 2.5 mm CBR value =0.189 For 5 mm CBR value = 0.199

As 2.5 mm penetration< 5mm penetration so this is suitable

So value considered for stercth1 is CBR VALUE=19%

Remaining sample stretches values are arranged in appendix.

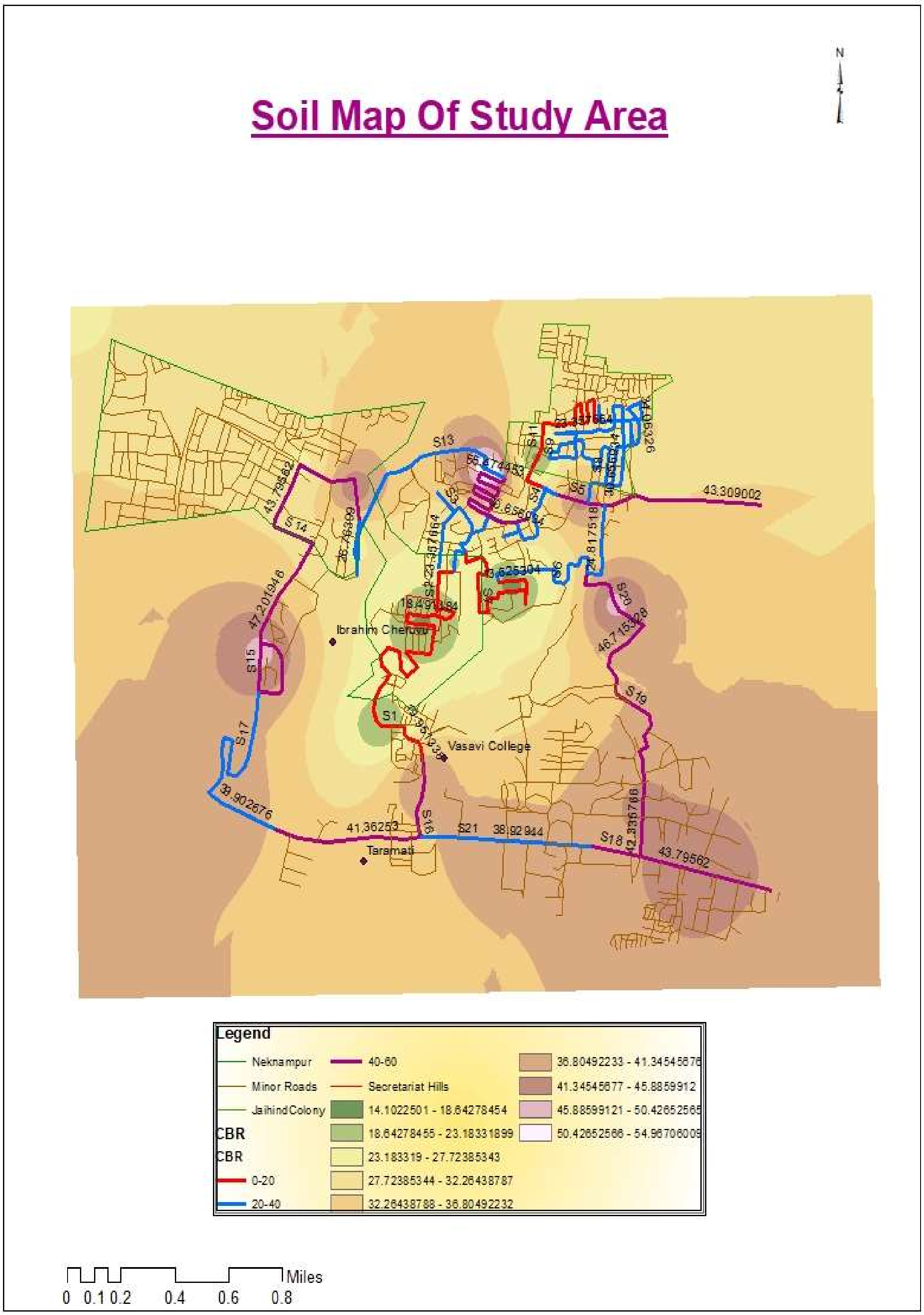


Fig 5.2. Map showing soil characteristics of different stretches in study area

5.2.2. POPULATION OF NEARBY TOWNS.

Population of nearby towns is to be calculated. To know population of each stretch nearby towns less than 2 kms from each stretch must be considered.

Sum of population of nearby towns gives total population of that stretch. The population of each stretch as listed in below table 5.2.2.

Table 5.2: Total Population of Different stretches of Road

Stretch Name	Population
S1	9000
S2	18057
S3	24137
S4	24197
S5	15557
S6	22598
S7	23598
S8	23598
S9	22598
S10	24637
S11	23137
S12	24437
S13	24437
S14	7880
S15	18437
S16	2010
S17	2800
S18	34567
S19	5330
S20	20154
S21	6780

5.2.3 FACILITIES SCORE:

Each stretch has its own facility score. The facilities score for different criteria has discussed in methodology. The score for each stretch has calculated by summing up all facilities in that stretch. For each stretch the calculated score are listed below in a table 5.2.3

Table 5.3 Facilities Score for different stretches

Stretch No	Facility Score
1	135
2	155
3	120
4	250
5	215
6	125
7	190
8	110
9	175
10	70
11	165
12	150
13	100
14	120
15	120
16	160
17	80
18	160
19	200
20	75
21	165

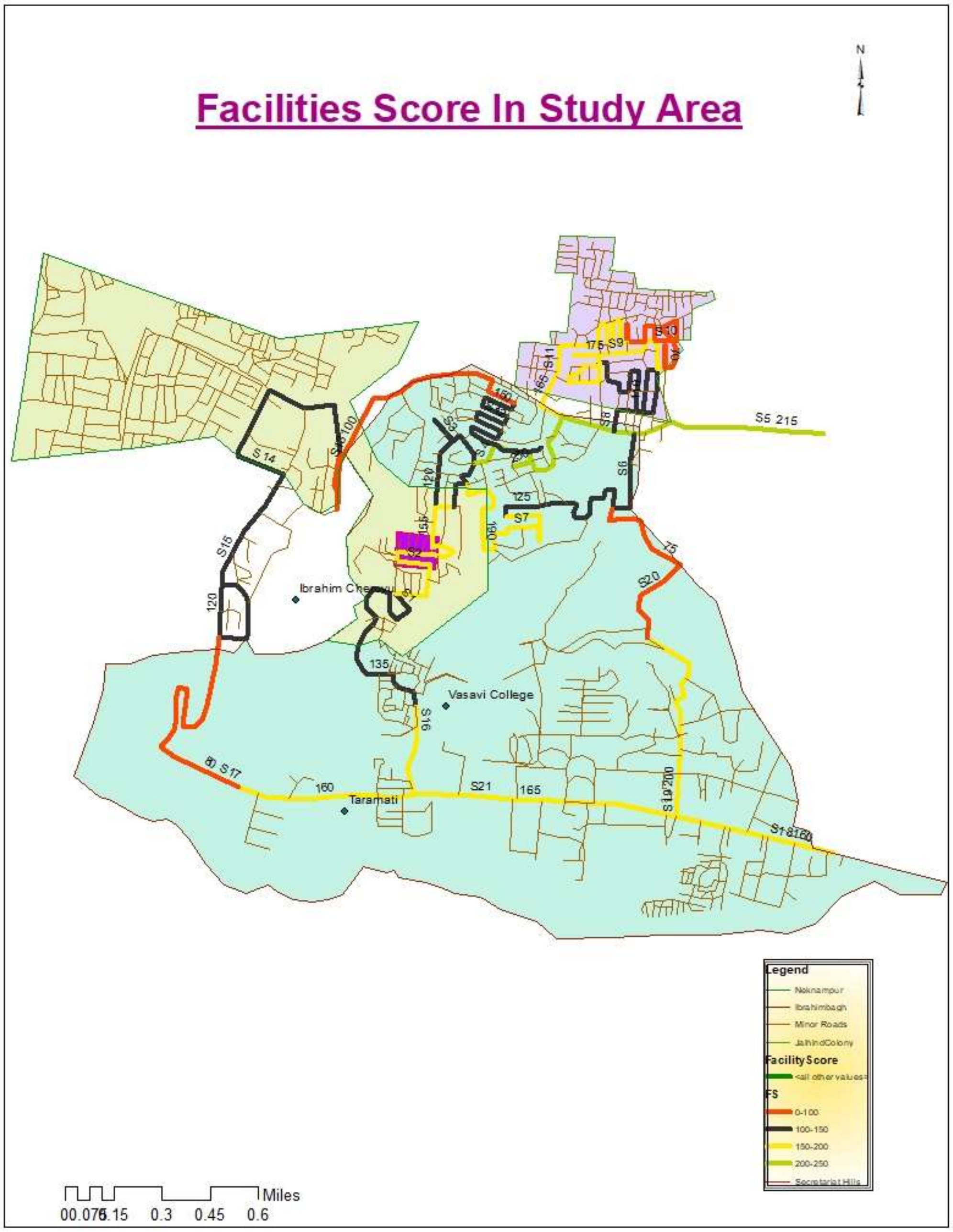


Fig 5.3 Map showing facilities score for each stretch in study area

5.2.4 DISTANCE TO NEARBY TOWNS:

In these distances from mid-point of the stretch to all the towns which located less than 2 kms are to be noted. The least distances among all the towns are considered as least distances from towns. The least distances for each town have been listed in a table 5.4.

Table 5.4 Distances of nearby town for each stretch

STRECTH NO	NEARBY DISTANCE(KM)
1	1.2
2	1.1
3	1.5
4	1
5	1.4
6	0.7
7	0.35
8	0.4
9	0.55
10	0.55
11	0.35
12	0.6
13	0.4
14	0.5
15	1.5
16	1.4
17	0.7
18	0.5
19	0.3
20	0.95
21	0.2

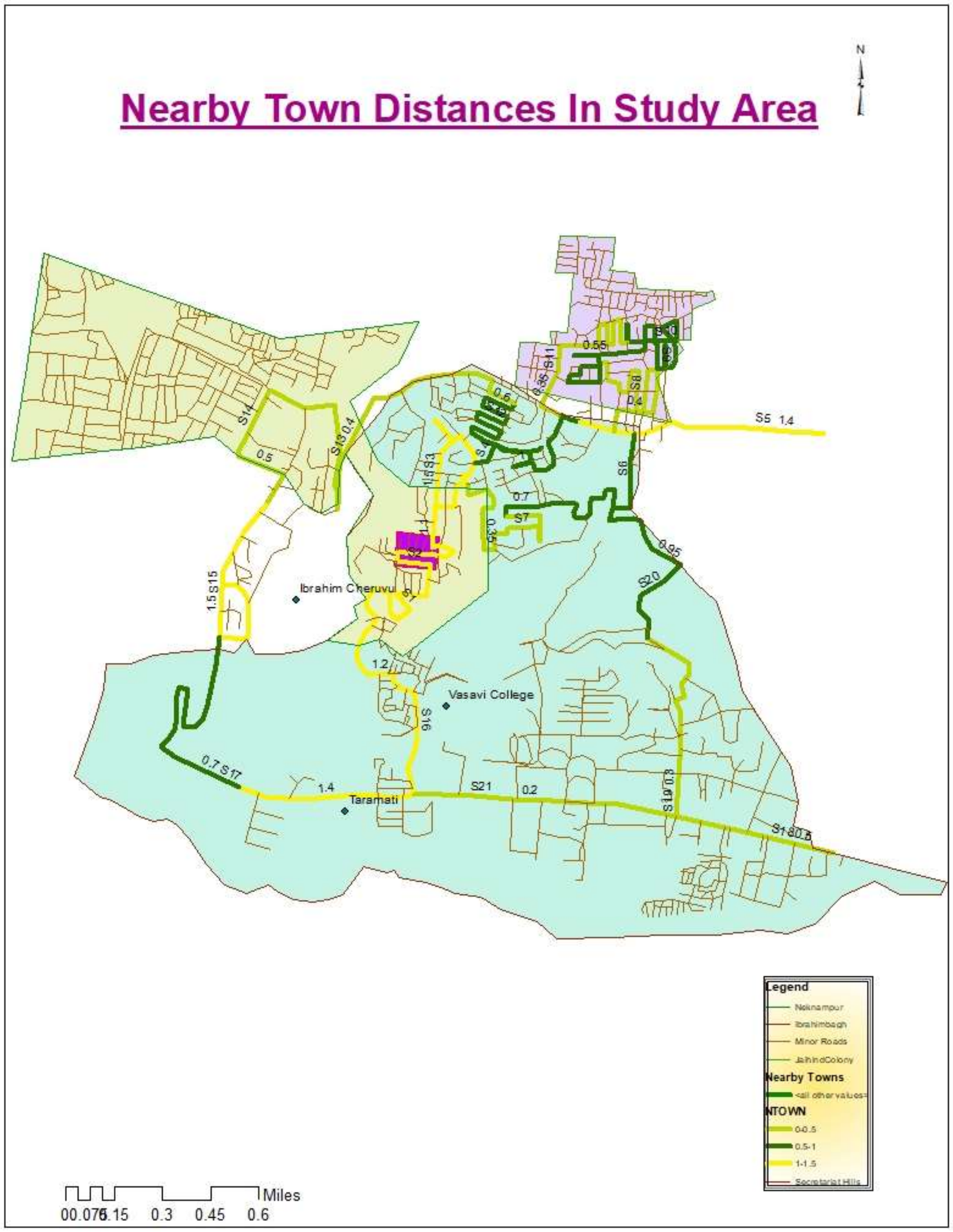


Fig 5.4 Map showing nearby towns distances for each stretch in study area

5.2.5 TRAFFICE VOLUME:

Volume of each stretch has to collected and methodology involved in this is explained in chapter 4. Volume of road for each stretch per hour collected has noted in table 5.2.5.

Table 5.5 Volume of traffic per hour for every stretch

STRECTH NO	VOLUME OF TRAFFIC
1	98
2	332
3	484
4	311
5	696
6	724
7	724
8	943
9	887
10	112
11	720
12	912
13	728
14	412
15	172
16	153
17	155
18	166
19	97
20	63
21	67

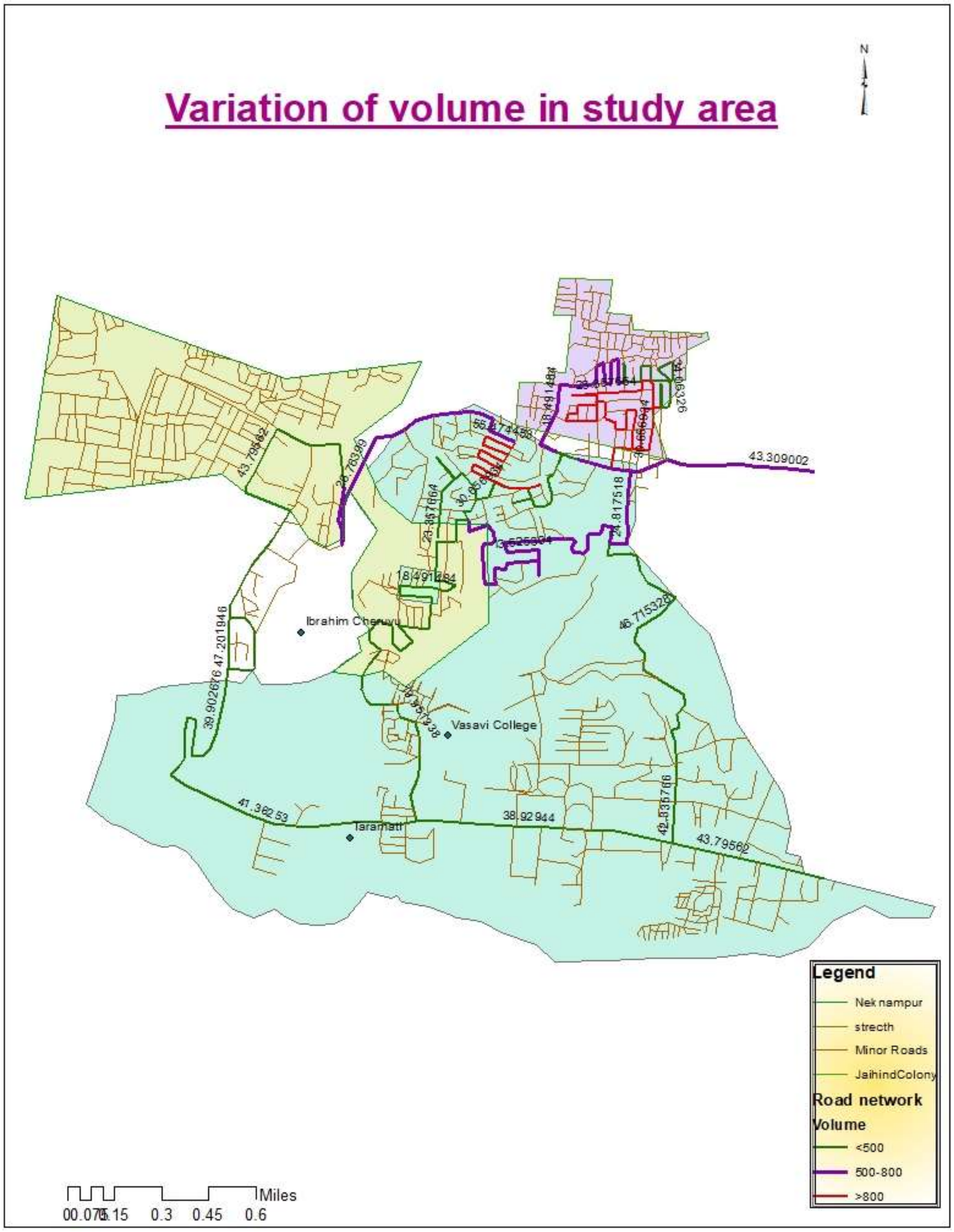


Fig 5.5 Map showing variation of volume for each stretch in study area

5.2.6 PAVEMENT DISTRESSES

Different pavements are subjected to different types of physical damages due to external factors. These pavement distresses are classified as low, medium and high based size, shape and on the intensity of damage they have undergone through. This concept is clearly explained in chapter 4.

Table 5.6 Defects count per each stretch

ST NO.	AC			BC			LC			TC			PH			RA			R			S			EC			D			P			M		
	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H
S1	2	1	0	4	3	1	8	4	3	6	8	1	1	1	0	11	5	8	0	0	0	1	0	0	4	2	1	5	3	1	1	1	0	2	1	1
S2	3	2	1	8	4	2	11	4	2	10	3	2	3	2	1	10	6	4	0	0	0	2	1	0	6	1	1	4	2	2	2	1	0	8	0	0
S3	4	3	2	3	1	1	9	2	1	7	3	1	2	1	1	15	4	2	0	0	0	1	0	0	3	2	0	3	1	1	1	1	0	7	3	1
S4	1	1	1	6	3	2	10	5	2	9	4	2	4	2	1	13	8	5	0	0	0	2	3	0	6	2	1	4	3	1	3	1	0	9	4	1
S5	2	3	2	3	2	1	10	4	1	8	3	1	3	2	1	11	5	6	10	5	3	3	2	1	7	3	2	5	4	2	1	1	1	7	3	1
S6	5	1	2	4	3	1	13	5	3	7	2	2	2	1	1	12	7	4	0	0	0	4	2	2	5	1	1	2	5	1	2	0	0	6	2	2
S7	3	2	1	5	2	0	11	6	2	6	4	2	3	1	1	14	6	1	0	0	0	5	3	1	4	2	2	3	4	2	2	2	0	5	1	1
S8	4	1	0	3	2	1	12	5	3	14	7	0	2	1	1	14	7	2	0	0	0	2	1	1	5	3	1	7	3	1	4	3	2	7	5	2
S9	2	2	1	5	2	1	10	5	3	5	4	1	3	2	1	17	5	1	0	0	0	1	1	0	3	2	1	5	3	1	2	1	1	0	0	0
S10	1	1	0	7	4	3	14	5	0	12	4	0	6	1	1	8	13	6	5	2	1	1	0	0	4	4	0	8	3	1	8	1	1	16	5	2
S11	0	0	0	2	1	0	9	1	0	7	4	1	4	0	0	5	2	0	2	1	1	0	0	0	0	0	0	2	1	0	2	1	0	10	1	3
S12	0	1	1	9	2	3	18	11	0	22	10	0	4	1	1	18	9	7	18	4	0	5	0	0	3	1	0	6	5	0	2	2	0	14	7	5
S13	1	1	1	10	5	3	14	6	2	18	10	0	3	2	0	23	8	0	1	1	0	0	1	0	2	1	0	1	1	0	1	1	0	7	3	0
S14	0	0	0	3	2	0	14	3	0	8	5	1	3	0	0	9	16	6	1	2	0	0	3	0	1	0	0	6	3	0	2	0	0	8	6	10
S15	0	2	4	4	7	2	8	8	8	4	9	0	3	4	0	17	9	4	4	4	0	1	1	0	4	1	0	1	4	0	1	1	0	3	3	1
S16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S17	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S18	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0	1	0	0	3	0	0	0	0	0
S19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
S21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	1	0	0	0	0	0

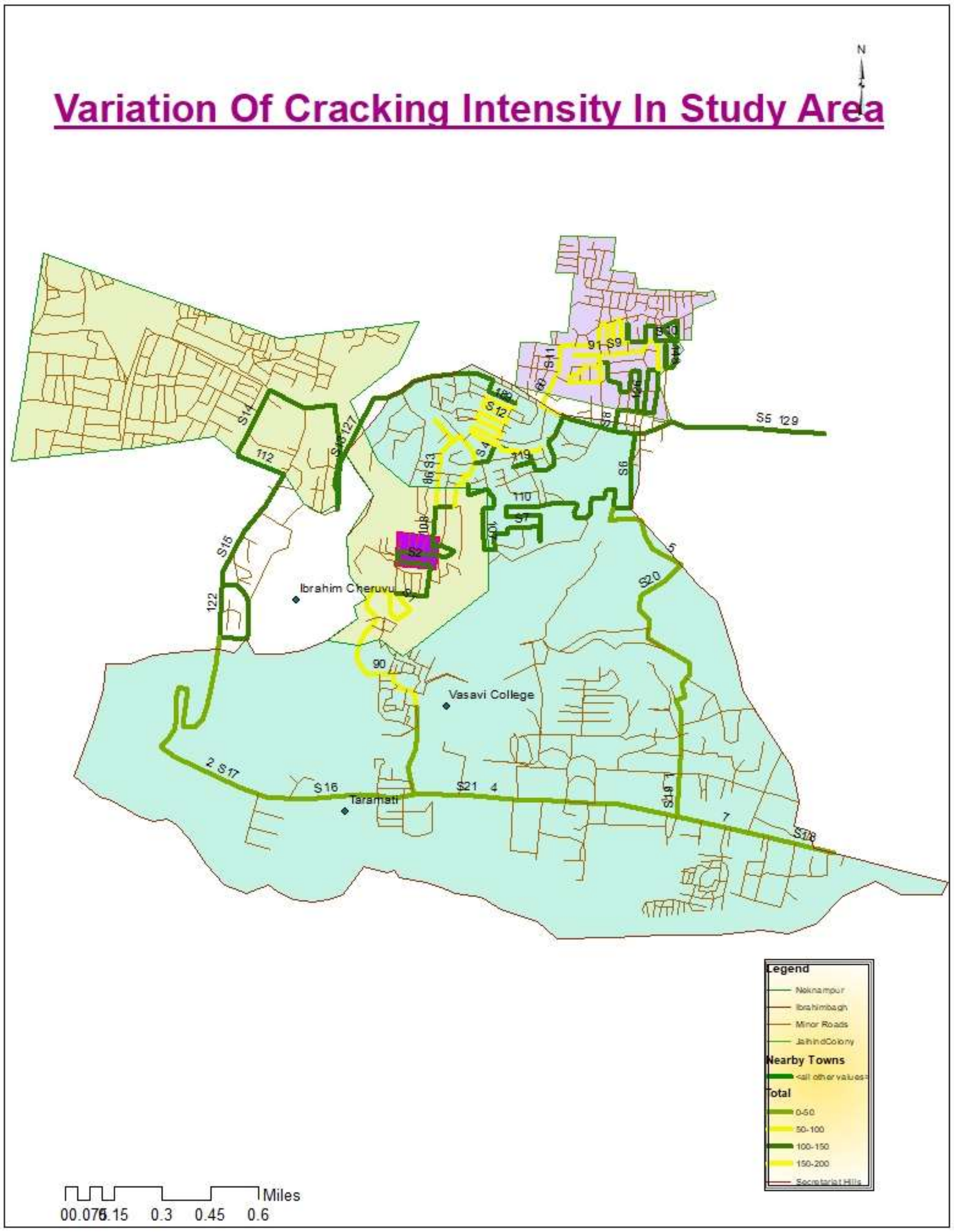


Fig 5.6 Map showing variation of crack intensity for each stretch in study area

5.3 GEOMETRIC CHARACTERISTICS:

5.3.1 CARRAIGE WIDTH:

As carriage width is not same for each stretch. So, carriage width is not considered for further process. No of lanes are considered instead of carriage width. Number of lanes for each stretch is displayed in table 5.3.1

Table 5.6 Number of lanes for each stretch

STRECTH NO	NUMBER OF LANES
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1
11	1
12	1
13	1
14	1
15	1
16	2
17	2
18	2
19	2
20	2
21	2

5.3.2 SIGHT DISTANCES:

Sight distances for all stretches are taken as adequate. As there are no undulations, curvature so sight distances for all stretches are adequate. So, sight distances for all stretches are considered as adequate.

5.4 CALCULATING THE PAVEMENT PRIORITIZATION USING FUZZY TOPSIS APPROACH:

5.4.1 Data Collection:

Data collected from all criteria as in chapter 5.2 is arranged in a table. Collected data is arranged as shown in the table 5.7.

5.4.2 Performing Normalisation process To form Rating Matrix:

Normalization is used on the raw data obtained as mentioned in chapter 4. The normalised values are taken as in table 5.8. The normalized points are then converted into standard ratings to obtain a rating matrix as mentioned in table 4.3. The rating matrix is taken as in table 5.9.

Table 5.7 Data of various parameters collected

ST NO.	AC			BC			LC			TC			PH			RA			R			S			EC			D			P			M			POP	VOL	FS	NTOWN	CBR
	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H					
S1	2	1	0	4	3	1	8	4	3	6	8	1	1	1	0	11	5	8	0	0	0	1	0	0	4	2	1	5	3	1	1	1	0	2	1	1	9000	98	135	1.2	19.95
S2	3	2	1	8	4	2	11	4	2	10	3	2	3	2	1	10	6	4	0	0	0	2	1	0	6	1	1	4	2	2	2	1	0	8	0	0	18057	332	155	1.1	18.49
S3	4	3	2	3	1	1	9	2	1	7	3	1	2	1	1	15	4	2	0	0	0	1	0	0	3	2	0	3	1	1	1	1	0	7	3	1	24137	484	120	1.5	23.36
S4	1	1	1	6	3	2	10	5	2	9	4	2	4	2	1	13	8	5	0	0	0	2	3	0	6	2	1	4	3	1	3	1	0	9	4	1	24194	311	250	1	30.66
S5	2	3	2	3	2	1	10	4	1	8	3	1	3	2	1	11	5	6	10	5	3	3	2	1	7	3	2	5	4	2	1	1	1	7	3	1	15557	696	215	1.4	43.31
S6	5	1	2	4	3	1	13	5	3	7	2	2	2	1	1	12	7	4	0	0	0	4	2	2	5	1	1	2	5	1	2	0	0	6	2	2	22598	724	125	0.7	24.82
S7	3	2	1	5	2	0	11	6	2	6	4	2	3	1	1	14	6	1	0	0	0	5	3	1	4	2	2	3	4	2	2	2	0	5	1	1	23598	724	190	0.35	13.63
S8	4	1	0	3	2	1	12	5	3	14	7	0	2	1	1	14	7	2	0	0	0	2	1	1	5	3	1	7	3	1	4	3	2	7	5	2	23598	943	110	0.4	30.66
S9	2	2	1	5	2	1	10	5	3	5	4	1	3	2	1	17	5	1	0	0	0	1	1	0	3	2	1	5	3	1	2	1	1	0	0	0	22598	887	175	0.55	23.36
S10	1	1	0	7	4	3	14	5	0	12	4	0	6	1	1	8	13	6	5	2	1	1	0	0	4	4	0	8	3	1	8	1	1	16	5	2	24637	112	70	0.55	34.06
S11	0	0	0	2	1	0	9	1	0	7	4	1	4	0	0	5	2	0	2	1	1	0	0	0	0	0	0	2	1	0	2	1	0	10	1	3	23137	720	165	0.35	18.49
S12	0	1	1	9	2	3	18	11	0	22	10	0	4	1	1	18	9	7	18	4	0	5	0	0	3	1	0	6	5	0	2	2	0	14	7	5	24437	912	150	0.6	55.47
S13	1	1	1	10	5	3	14	6	2	18	10	0	3	2	0	23	8	0	1	1	0	0	1	0	2	1	0	1	1	0	1	1	0	7	3	0	24437	728	100	0.4	26.76
S14	0	0	0	3	2	0	14	3	0	8	5	1	3	0	0	9	16	6	1	2	0	0	3	0	1	0	0	6	3	0	2	0	0	8	6	10	7880	412	120	0.5	43.80
S15	0	2	4	4	7	2	8	8	8	4	9	0	3	4	0	17	9	4	4	4	0	1	1	0	4	1	0	1	4	0	1	1	0	3	3	1	18437	172	120	1.5	47.20
S16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2010	153	160	1.4	41.36
S17	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2800	155	80	0.7	39.90
S18	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0	1	0	0	3	0	0	0	0	0	34567	166	160	0.5	43.80
S19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5330	97	200	0.3	42.34
S20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	20154	63	75	0.95	46.72
S21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	1	0	0	0	0	0	6780	67	165	0.2	38.93

Table 5.8 Normalized Data Points obtained from raw data

ST.NO	A_L	A_M	A_H	B_L	B_M	B_H	L_L	L_M	L_H	T_L	T_M	T_H	PH_L	PH_M	PH_H	RA_L	RA_M	RA_H	R_L	R_M	R_H	S_L	S_M	S_H
S1	40	33	0	40	42	33	44	36	37	27	80	50	16	25	0	47	31	100	0	0	0	20	0	0
S2	60	66	25	80	57	66	61	36	25	45	30	100	50	50	100	43	37	50	0	0	0	40	33	0
S3	80	100	50	30	14	33	50	18	12	31	30	50	33	25	100	65	25	25	0	0	0	20	0	0
S4	20	33	25	60	42	66	55	45	25	40	40	100	66	50	100	56	50	62	0	0	0	40	100	0
S5	40	100	50	30	28	33	55	36	12	36	30	50	50	50	100	47	31	75	55	100	100	60	66	50
S6	100	33	50	40	42	33	72	45	37	31	20	100	33	25	100	52	43	50	0	0	0	80	66	100
S7	60	66	25	50	28	0	61	54	25	27	40	100	50	25	100	60	37	12	0	0	0	100	100	50
S8	80	33	0	30	28	33	66	45	37	63	70	0	33	25	100	60	43	25	0	0	0	40	33	50
S9	40	66	25	50	28	33	55	45	37	22	40	50	50	50	100	73	31	12	0	0	0	20	33	0
S10	20	33	0	70	57	100	77	45	0	54	40	0	100	25	100	34	81	75	27	40	33	20	0	0
S11	0	0	0	20	14	0	50	9	0	31	40	50	66	0	0	21	12	0	11	20	33	0	0	0
S12	0	33	25	90	28	100	100	100	0	100	100	0	66	25	100	78	56	87	100	80	0	100	0	0
S13	20	33	25	100	71	100	77	54	25	81	100	0	50	50	0	100	50	0	5	20	0	0	33	0
S14	0	0	0	30	28	0	77	27	0	36	50	50	50	0	0	39	100	75	5	40	0	0	100	0
S15	0	66	100	40	100	66	44	72	100	18	90	0	50	100	0	73	56	50	22	80	0	20	33	0
S16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	40	0	0	0	0
S17	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0	0	20	0	0	0	0
S18	0	0	0	0	0	0	0	0	0	0	0	0	0	25	0	0	0	0	11	0	0	0	0	0
S19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0
S20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	60	0	0	0	0
S21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0

Table 5.8.1 Normalized Data Points obtained from raw data (cont. of table 5.8)

ST N0	E_L	E_M	E_H	D_L	D_M	D_H	P	P	P	M	M	M	POP	VOL	FS	NTOWN	CBR
S1	57	50	50	62	60	50	12	33	0	12	14	10	26	10	54	20	64
S2	85	25	50	50	40	10	25	33	0	50	0	0	52	35	62	26	66
S3	42	50	0	37	20	50	12	33	0	43	42	10	69	51	48	0	57
S4	85	50	50	50	60	50	37	33	0	56	57	10	69	32	100	33	44
S5	100	25	100	62	80	100	12	33	50	43	42	10	45	73	86	6	21
S6	71	75	50	25	100	50	25	0	0	37	28	20	65	76	50	53	55
S7	57	50	100	37	80	100	25	66	0	31	14	10	68	76	76	76	75
S8	71	75	50	87	60	50	50	100	100	43	71	20	68	100	44	73	44
S9	42	50	50	62	60	50	25	33	50	0	0	0	65	94	70	63	57
S10	57	100	0	100	60	50	100	33	50	100	71	20	71	11	28	63	38
S11	0	0	0	25	20	0	25	33	0	6	14	30	66	76	66	76	66
S12	42	25	0	75	100	0	25	66	0	87	100	50	70	96	60	60	0
S13	28	25	0	12	20	0	12	33	0	43	42	0	70	77	40	73	51
S14	14	0	0	75	60	0	25	0	0	50	85	100	22	43	48	66	21
S15	57	25	0	12	18	0	12	33	0	18	42	10	53	18	48	0	14
S16	0	0	0	0	0	0	0	0	0	0	0	0	5	16	64	6	25
S17	0	0	0	0	0	0	0	0	0	0	0	0	8	16	32	53	28
S18	0	0	0	12	0	0	37	0	0	0	0	0	100	17	64	66	21
S19	0	0	0	0	0	0	0	0	0	0	0	0	15	10	80	80	23
S20	0	0	0	0	0	0	0	0	50	0	0	0	58	6	30	36	15
S21	0	0	0	0	40	0	12	0	0	0	0	0	19	7	66	86	29

4	4	1	4	5	4	5	4	4	3	8	5	2	3	1	5	4	10	1	1	1	2	1	1	6	5	5	7	6	5	2	4	1	2	2	1	3	1	6	2	7
6	7	3	8	6	7	7	4	3	5	3	10	5	5	10	5	4	5	1	1	1	4	4	1	9	3	5	5	4	10	3	4	1	5	1	1	6	4	7	3	7
8	10	5	3	2	4	5	2	2	4	3	5	4	3	10	7	3	3	1	1	1	2	1	1	5	5	1	4	2	5	2	4	1	5	5	1	7	6	5	1	6
2	4	3	6	5	7	6	5	3	4	4	10	7	5	10	6	5	7	1	1	1	4	10	1	9	5	5	5	6	5	4	4	1	6	6	1	7	4	10	4	5
4	10	5	3	3	4	6	4	2	4	3	5	5	5	10	5	4	8	6	10	10	6	7	5	10	8	10	7	8	10	2	4	5	5	5	1	5	8	9	1	3
10	4	5	4	5	4	8	5	4	4	2	10	4	3	10	6	5	5	1	1	1	8	7	10	8	3	5	3	10	5	3	1	1	4	3	2	7	8	5	6	6
6	7	3	5	3	1	7	6	3	3	4	10	5	3	10	6	4	2	1	1	1	10	10	5	6	5	10	4	8	10	3	7	1	4	2	1	7	8	8	8	8
8	4	1	3	3	4	7	5	4	7	7	1	4	3	10	6	5	3	1	1	1	4	4	5	8	8	5	9	6	5	5	10	10	5	8	2	7	10	5	8	5
4	7	3	5	3	4	6	5	4	3	4	5	5	5	10	8	4	2	1	1	1	2	4	1	5	5	5	7	6	5	3	4	5	1	1	1	7	10	7	7	6
2	4	1	7	6	10	8	5	1	6	4	1	10	3	10	4	9	8	3	4	4	2	1	1	6	10	1	10	6	5	10	4	5	10	8	2	8	2	3	7	4
1	1	1	2	2	1	5	1	1	4	4	5	7	1	1	3	2	1	2	2	4	1	1	1	1	1	1	3	2	1	3	4	1	7	2	3	7	8	7	8	7
1	4	3	9	3	10	10	10	1	10	10	1	7	3	10	8	6	9	10	8	1	10	1	1	5	3	1	8	10	1	3	7	1	9	10	5	7	10	6	6	1
2	4	3	10	8	10	8	6	3	9	10	1	5	5	1	10	5	1	1	2	1	1	4	1	3	3	1	2	2	1	2	4	1	5	5	1	7	8	4	8	6
1	1	1	3	3	1	8	3	1	4	5	5	5	1	1	4	10	8	1	4	1	1	10	1	2	1	1	8	6	1	3	1	1	5	9	10	3	5	5	7	3
1	7	10	4	10	7	5	8	10	2	9	1	5	10	1	8	6	5	3	8	1	2	4	1	6	3	1	2	8	1	2	4	1	2	5	1	6	2	5	1	2
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	4	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	7	1	3
1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	4	6	3
1	1	1	1	1	1	1	1	1	1	1	1	1	3	1	1	1	1	2	1	1	1	1	1	1	1	1	2	1	1	4	1	1	1	1	1	10	2	7	7	3
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	8	8	3
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	6	1	1	1	1	1	1	1	1	1	1	1	1	5	1	1	1	6	1	3	4	2
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	4	1	2	1	1	1	1	1	2	1	7	9	3

Table 5.9 Rating Matrix obtained from normalized data points

5.4.3 Collecting Experts Opinions for pavement Distresses in each stretch:

The Expert survey is conducted, and data is obtained from different experts in the field as mention in chapter 4. The expert survey data is arranged as show in table 5.10.

Table 5.10 Expert Survey Data Collected

CRITERIA	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11	E12	E13	E14	E15
A_L	L	N	N	L	N	N	N	N	N	N	L	L	N	L	N
A_M	M	M	L	M	N	L	L	L	L	L	M	M	L	M	L
A_H	H	H	M	H	L	M	M	VH	L	M	H	H	M	H	M
B_L	L	N	N	L	N	N	N	N	N	N	L	L	N	L	N
B_M	M	M	L	M	N	L	L	L	L	L	M	M	L	M	L
B_H	H	H	M	H	L	M	M	VH	L	M	H	H	M	H	M
L_L	L	N	N	L	N	N	N	N	N	N	L	L	N	L	N
L_M	M	M	L	M	N	L	L	L	L	L	M	M	L	M	L
L_H	H	H	M	H	L	M	M	VH	L	M	H	H	M	H	M
T_L	L	N	N	L	N	N	N	N	N	N	L	L	N	L	N
T_M	M	M	L	M	N	L	L	L	L	L	M	M	L	M	L
T_H	H	H	M	H	L	M	M	VH	L	M	H	H	M	H	M
PH_L	M	L	L	M	N	M	M	M	M	M	M	L	L	L	M
PH_M	H	H	M	H	L	H	H	H	H	H	H	M	M	M	H
PH_H	VH	VH	H	VH	H	VH	VH	VH	VH	VH	VH	VH	H	H	VH
RA_L	M	L	L	N	N	M	L	L	L	L	N	N	N	N	N
RA_M	H	M	M	L	N	H	M	M	L	M	L	L	L	L	N
RA_H	VH	H	H	M	M	VH	VH	H	M	H	M	H	M	H	M

Table 5.10.1 Expert Survey Data cont.

R_L	M	N	L	N	L	L	N	N	L	L	L	N	L	N	L
R_M	H	M	L	L	M	M	L	L	M	M	M	L	M	M	M
R_H	VH	VH	M	M	VH	H	M	M	H	H	H	H	M	H	H
S_L	L	N	N	L	N	N	N	N	N	N	L	L	N	L	N
S_M	M	M	L	M	N	L	L	L	L	L	M	M	L	M	L
S_H	H	H	M	H	L	M	M	VH	L	M	H	H	M	H	M
E_L	M	L	L	L	N	N	L	L	L	N	N	N	N	L	N
E_M	H	H	M	M	L	L	M	H	M	L	L	L	L	M	N
E_H	VH	VH	H	H	VH	M	H	VH	H	M	M	H	H	H	N
D_L	M	N	L	N	L	L	N	N	L	L	L	N	L	N	L
D_M	H	M	L	L	M	M	L	L	M	M	M	L	M	M	M
D_H	VH	VH	M	M	VH	H	M	M	H	H	H	H	M	H	H
P_L	L	N	L	N	N	M	N	N	M	M	N	N	L	L	L
P_M	M	M	M	L	L	H	L	M	M	H	L	L	M	M	M
P_H	H	H	H	M	M	VH	M	VH	H	VH	M	M	M	H	H
M_L	L	N	L	N	N	M	N	N	M	M	N	N	L	L	L
M_M	M	M	M	L	L	H	L	M	M	H	L	L	M	M	M
M_H	H	H	H	M	M	VH	M	VH	H	VH	M	M	M	H	H
POP	L	N	H	H	L	M	M	N	M	VH	M	VH	VH	L	N
VOL	L	L	N	L	M	M	M	H	H	L	M	H	M	H	VH
FS	M	L	VH	H	L	N	N	L	L	N	L	M	M	H	H
NTOWN	L	M	VH	M	H	M	L	L	M	L	N	VH	N	L	H
CBR	L	N	VH	H	H	L	L	VH	L	H	L	M	H	M	H

5.4.4Determination of Fuzzy Weights:

Determination Of fuzzy weights are explained in chapter 4. Obtained weights for each criterion are shown in table 5.12

CRITERIA	0	1	2
A_L	0	0.033	0.167
A_M	0.12	0.253	0.447
A_H	0.46	0.64	0.787
B_L	0	0.033	0.167
B_M	0.12	0.253	0.447
B_H	0.46	0.64	0.787
L_L	0	0.033	0.167
L_M	0.12	0.253	0.447
L_H	0.46	0.64	0.787
T_L	0	0.033	0.167
T_M	0.12	0.253	0.447
T_H	0.46	0.64	0.787
PH_L	0.18	0.333	0.527
PH_M	0.547	0.74	0.873
PH_H	0.847	0.973	1
RA_L	0.04	0.107	0.26
RA_M	0.193	0.327	0.5
RA_H	0.58	0.76	0.88
R_L	0.02	0.087	0.247
R_M	0.227	0.393	0.587
R_H	0.607	0.787	0.9
S_L	0	0.033	0.167
S_M	0.12	0.253	0.447
S_H	0.46	0.64	0.787
E_L	0.02	0.08	0.233
E_M	0.24	0.387	0.56
E_H	0.627	0.787	0.88
D_L	0.02	0.087	0.247
D_M	0.227	0.393	0.587
D_H	0.607	0.787	0.9
P_L	0.06	0.133	0.287
P_M	0.253	0.42	0.607
P_H	0.58	0.76	0.88
M_L	0.06	0.133	0.287
M_M	0.253	0.42	0.607
M_H	0.58	0.76	0.88
POP	0.353	0.473	0.6
VOL	0.347	0.5	0.653
FS	0.26	0.38	0.527
NTOWN	0.293	0.42	0.567
CBR	0.393	0.533	0.667

Table 5.12 Fuzzy weights obtained from fuzzy number

5.4.5 Determination of Fuzzy Evaluation Values:

The Fuzzy Evaluation Values are obtained by multiplying the rating matrix and their corresponding weightages obtained from expert survey and explained in chapter 4. The Fuzzy Evaluation Values are obtained for each stretch as shown in table 5.12.

Table 5.12 Fuzzy Evaluation Values for each stretch

Stretch No.	l	m	n
A1	38.227	56.923	79.469
A2	53.725	77.293	105.182
A3	42.121	60.97	83.581
A4	55.882	81.619	112.025
A5	70.023	100.834	135.451
A6	56.945	82.369	112.446
A7	59.985	86.898	118.635
A8	57.746	84.475	116.671
A9	51.918	74.878	101.543
A10	54.645	80.873	113.169
A11	30.49	45.393	63.605
A12	54.049	82.085	118.637
A13	36.982	57.624	84.91
A14	37.089	56.646	80.859
A15	44.776	69.073	98.19
A16	14.688	21.615	29.658
A17	15.099	22.122	30.265
A18	20.256	29.266	39.8
A19	16.551	24.122	32.927
A20	18.326	26.513	35.625
A21	17.098	25.081	34.428

5.4.6: Determination of Triangular Fuzzy Numbers

Triangular Fuzzy Numbers is calculated to evaluate relative preference between the stretches and are calculated by obtaining the difference between each stretch's fuzzy weightages, corresponding to other stretch weightages in a iterative manner as mentioned in chapter 4. The Triangular Fuzzy Numbers are obtained as shown in table 5.13.

Table 5.13 Triangular Fuzzy numbers for all stretches

p~i-p~j	l	m	n
p1-p1	-41.242	0	41.242
p1-p2	-66.955	-20.37	25.744
p1-p3	-45.354	-4.047	37.348
p1-p4	-73.798	-24.696	23.587
p1-p5	-97.224	-43.911	9.446
p1-p6	-74.219	-25.446	22.524
p1-p7	-80.408	-29.975	19.484
p1-p8	-78.444	-27.552	21.723
p1-p9	-63.316	-17.955	27.551
p1-p10	-74.942	-23.95	24.824
p1-p11	-25.378	11.53	48.979
p1-p12	-80.41	-25.162	25.42
p1-p13	-46.683	-0.701	42.487
p1-p14	-42.632	0.277	42.38
p1-p15	-59.963	-12.15	34.693
p1-p16	8.569	35.308	64.781
p1-p17	7.962	34.801	64.37
p1-p18	-1.573	27.657	59.213
p1-p19	5.3	32.801	62.918
p1-p20	2.602	30.41	61.143
p1-p21	3.799	31.842	62.371
p2-p1	-25.744	20.37	66.955
p2-p2	-51.457	0	51.457
p2-p3	-29.856	16.323	63.061
p2-p4	-58.3	-4.326	49.3
p2-p5	-81.726	-23.541	35.159
p2-p6	-58.721	-5.076	48.237
p2-p7	-64.91	-9.605	45.197
p2-p8	-62.946	-7.182	47.436
p2-p9	-47.818	2.415	53.264
p2-p10	-59.444	-3.58	50.537
p2-p11	-9.88	31.9	74.692
p2-p12	-64.912	-4.792	51.133

p2-p13	-31.185	19.669	68.2
p2-p14	-27.134	20.647	68.093
p2-p15	-44.465	8.22	60.406
p2-p16	24.067	55.678	90.494
p2-p17	23.46	55.171	90.083
p2-p18	13.925	48.027	84.926
p2-p19	20.798	53.171	88.631
p2-p20	18.1	50.78	86.856
p2-p21	19.297	52.212	88.084
p3-p1	-37.348	4.047	45.354
p3-p2	-63.061	-16.323	29.856
p3-p3	-41.46	0	41.46
p3-p4	-69.904	-20.649	27.699
p3-p5	-93.33	-39.864	13.558
p3-p6	-70.325	-21.399	26.636
p3-p7	-76.514	-25.928	23.596
p3-p8	-74.55	-23.505	25.835
p3-p9	-59.422	-13.908	31.663
p3-p10	-71.048	-19.903	28.936
p3-p11	-21.484	15.577	53.091
p3-p12	-76.516	-21.115	29.532
p3-p13	-42.789	3.346	46.599
p3-p14	-38.738	4.324	46.492
p3-p15	-56.069	-8.103	38.805
p3-p16	12.463	39.355	68.893
p3-p17	11.856	38.848	68.482
p3-p18	2.321	31.704	63.325
p3-p19	9.194	36.848	67.03
p3-p20	6.496	34.457	65.255
p3-p21	7.693	35.889	66.483
p4-p1	-23.587	24.696	73.798
p4-p2	-49.3	4.326	58.3
p4-p3	-27.699	20.649	69.904
p4-p4	-56.143	0	56.143
p4-p5	-79.569	-19.215	42.002
p4-p6	-56.564	-0.75	55.08
p4-p7	-62.753	-5.279	52.04
p4-p8	-60.789	-2.856	54.279
p4-p9	-45.661	6.741	60.107
p4-p10	-57.287	0.746	57.38
p4-p11	-7.723	36.226	81.535
p4-p12	-62.755	-0.466	57.976
p4-p13	-29.028	23.995	75.043
p4-p14	-24.977	24.973	74.936
p4-p15	-42.308	12.546	67.249
p4-p16	26.224	60.004	97.337

p4-p17	25.617	59.497	96.926
p4-p18	16.082	52.353	91.769
p4-p19	22.955	57.497	95.474
p4-p20	20.257	55.106	93.699
p4-p21	21.454	56.538	94.927
p5-p1	-9.446	43.911	97.224
p5-p2	-35.159	23.541	81.726
p5-p3	-13.558	39.864	93.33
p5-p4	-42.002	19.215	79.569
p5-p5	-65.428	0	65.428
p5-p6	-42.423	18.465	78.506
p5-p7	-48.612	13.936	75.466
p5-p8	-46.648	16.359	77.705
p5-p9	-31.52	25.956	83.533
p5-p10	-43.146	19.961	80.806
p5-p11	6.418	55.441	104.961
p5-p12	-48.614	18.749	81.402
p5-p13	-14.887	43.21	98.469
p5-p14	-10.836	44.188	98.362
p5-p15	-28.167	31.761	90.675
p5-p16	40.365	79.219	120.763
p5-p17	39.758	78.712	120.352
p5-p18	30.223	71.568	115.195
p5-p19	37.096	76.712	118.9
p5-p20	34.398	74.321	117.125
p5-p21	35.595	75.753	118.353
p6-p1	-22.524	25.446	74.219
p6-p2	-48.237	5.076	58.721
p6-p3	-26.636	21.399	70.325
p6-p4	-55.08	0.75	56.564
p6-p5	-78.506	-18.465	42.423
p6-p6	-55.501	0	55.501
p6-p7	-61.69	-4.529	52.461
p6-p8	-59.726	-2.106	54.7
p6-p9	-44.598	7.491	60.528
p6-p10	-56.224	1.496	57.801
p6-p11	-6.66	36.976	81.956
p6-p12	-61.692	0.284	58.397
p6-p13	-27.965	24.745	75.464
p6-p14	-23.914	25.723	75.357
p6-p15	-41.245	13.296	67.67
p6-p16	27.287	60.754	97.758
p6-p17	26.68	60.247	97.347
p6-p18	17.145	53.103	92.19
p6-p19	24.018	58.247	95.895
p6-p20	21.32	55.856	94.12

p6-p21	22.517	57.288	95.348
p7-p1	-19.484	29.975	80.408
p7-p2	-45.197	9.605	64.91
p7-p3	-23.596	25.928	76.514
p7-p4	-52.04	5.279	62.753
p7-p5	-75.466	-13.936	48.612
p7-p6	-52.461	4.529	61.69
p7-p7	-58.65	0	58.65
p7-p8	-56.686	2.423	60.889
p7-p9	-41.558	12.02	66.717
p7-p10	-53.184	6.025	63.99
p7-p11	-3.62	41.505	88.145
p7-p12	-58.652	4.813	64.586
p7-p13	-24.925	29.274	81.653
p7-p14	-20.874	30.252	81.546
p7-p15	-38.205	17.825	73.859
p7-p16	30.327	65.283	103.947
p7-p17	29.72	64.776	103.536
p7-p18	20.185	57.632	98.379
p7-p19	27.058	62.776	102.084
p7-p20	24.36	60.385	100.309
p7-p21	25.557	61.817	101.537
p8-p1	-21.723	27.552	78.444
p8-p2	-47.436	7.182	62.946
p8-p3	-25.835	23.505	74.55
p8-p4	-54.279	2.856	60.789
p8-p5	-77.705	-16.359	46.648
p8-p6	-54.7	2.106	59.726
p8-p7	-60.889	-2.423	56.686
p8-p8	-58.925	0	58.925
p8-p9	-43.797	9.597	64.753
p8-p10	-55.423	3.602	62.026
p8-p11	-5.859	39.082	86.181
p8-p12	-60.891	2.39	62.622
p8-p13	-27.164	26.851	79.689
p8-p14	-23.113	27.829	79.582
p8-p15	-40.444	15.402	71.895
p8-p16	28.088	62.86	101.983
p8-p17	27.481	62.353	101.572
p8-p18	17.946	55.209	96.415
p8-p19	24.819	60.353	100.12
p8-p20	22.121	57.962	98.345
p8-p21	23.318	59.394	99.573
p9-p1	-27.551	17.955	63.316
p9-p2	-53.264	-2.415	47.818
p9-p3	-31.663	13.908	59.422

p9-p4	-60.107	-6.741	45.661
p9-p5	-83.533	-25.956	31.52
p9-p6	-60.528	-7.491	44.598
p9-p7	-66.717	-12.02	41.558
p9-p8	-64.753	-9.597	43.797
p9-p9	-49.625	0	49.625
p9-p10	-61.251	-5.995	46.898
p9-p11	-11.687	29.485	71.053
p9-p12	-66.719	-7.207	47.494
p9-p13	-32.992	17.254	64.561
p9-p14	-28.941	18.232	64.454
p9-p15	-46.272	5.805	56.767
p9-p16	22.26	53.263	86.855
p9-p17	21.653	52.756	86.444
p9-p18	12.118	45.612	81.287
p9-p19	18.991	50.756	84.992
p9-p20	16.293	48.365	83.217
p9-p21	17.49	49.797	84.445
p10-p1	-24.824	23.95	74.942
p10-p2	-50.537	3.58	59.444
p10-p3	-28.936	19.903	71.048
p10-p4	-57.38	-0.746	57.287
p10-p5	-80.806	-19.961	43.146
p10-p6	-57.801	-1.496	56.224
p10-p7	-63.99	-6.025	53.184
p10-p8	-62.026	-3.602	55.423
p10-p9	-46.898	5.995	61.251
p10-p10	-58.524	0	58.524
p10-p11	-8.96	35.48	82.679
p10-p12	-63.992	-1.212	59.12
p10-p13	-30.265	23.249	76.187
p10-p14	-26.214	24.227	76.08
p10-p15	-43.545	11.8	68.393
p10-p16	24.987	59.258	98.481
p10-p17	24.38	58.751	98.07
p10-p18	14.845	51.607	92.913
p10-p19	21.718	56.751	96.618
p10-p20	19.02	54.36	94.843
p10-p21	20.217	55.792	96.071
p11-p1	-48.979	-11.53	25.378
p11-p2	-74.692	-31.9	9.88
p11-p3	-53.091	-15.577	21.484
p11-p4	-81.535	-36.226	7.723
p11-p5	-104.961	-55.441	-6.418
p11-p6	-81.956	-36.976	6.66
p11-p7	-88.145	-41.505	3.62

p11-p8	-86.181	-39.082	5.859
p11-p9	-71.053	-29.485	11.687
p11-p10	-82.679	-35.48	8.96
p11-p11	-33.115	0	33.115
p11-p12	-88.147	-36.692	9.556
p11-p13	-54.42	-12.231	26.623
p11-p14	-50.369	-11.253	26.516
p11-p15	-67.7	-23.68	18.829
p11-p16	0.832	23.778	48.917
p11-p17	0.225	23.271	48.506
p11-p18	-9.31	16.127	43.349
p11-p19	-2.437	21.271	47.054
p11-p20	-5.135	18.88	45.279
p11-p21	-3.938	20.312	46.507
p12-p1	-25.42	25.162	80.41
p12-p2	-51.133	4.792	64.912
p12-p3	-29.532	21.115	76.516
p12-p4	-57.976	0.466	62.755
p12-p5	-81.402	-18.749	48.614
p12-p6	-58.397	-0.284	61.692
p12-p7	-64.586	-4.813	58.652
p12-p8	-62.622	-2.39	60.891
p12-p9	-47.494	7.207	66.719
p12-p10	-59.12	1.212	63.992
p12-p11	-9.556	36.692	88.147
p12-p12	-64.588	0	64.588
p12-p13	-30.861	24.461	81.655
p12-p14	-26.81	25.439	81.548
p12-p15	-44.141	13.012	73.861
p12-p16	24.391	60.47	103.949
p12-p17	23.784	59.963	103.538
p12-p18	14.249	52.819	98.381
p12-p19	21.122	57.963	102.086
p12-p20	18.424	55.572	100.311
p12-p21	19.621	57.004	101.539
p13-p1	-42.487	0.701	46.683
p13-p2	-68.2	-19.669	31.185
p13-p3	-46.599	-3.346	42.789
p13-p4	-75.043	-23.995	29.028
p13-p5	-98.469	-43.21	14.887
p13-p6	-75.464	-24.745	27.965
p13-p7	-81.653	-29.274	24.925
p13-p8	-79.689	-26.851	27.164
p13-p9	-64.561	-17.254	32.992
p13-p10	-76.187	-23.249	30.265
p13-p11	-26.623	12.231	54.42

p13-p12	-81.655	-24.461	30.861
p13-p13	-47.928	0	47.928
p13-p14	-43.877	0.978	47.821
p13-p15	-61.208	-11.449	40.134
p13-p16	7.324	36.009	70.222
p13-p17	6.717	35.502	69.811
p13-p18	-2.818	28.358	64.654
p13-p19	4.055	33.502	68.359
p13-p20	1.357	31.111	66.584
p13-p21	2.554	32.543	67.812
p14-p1	-42.38	-0.277	42.632
p14-p2	-68.093	-20.647	27.134
p14-p3	-46.492	-4.324	38.738
p14-p4	-74.936	-24.973	24.977
p14-p5	-98.362	-44.188	10.836
p14-p6	-75.357	-25.723	23.914
p14-p7	-81.546	-30.252	20.874
p14-p8	-79.582	-27.829	23.113
p14-p9	-64.454	-18.232	28.941
p14-p10	-76.08	-24.227	26.214
p14-p11	-26.516	11.253	50.369
p14-p12	-81.548	-25.439	26.81
p14-p13	-47.821	-0.978	43.877
p14-p14	-43.77	0	43.77
p14-p15	-61.101	-12.427	36.083
p14-p16	7.431	35.031	66.171
p14-p17	6.824	34.524	65.76
p14-p18	-2.711	27.38	60.603
p14-p19	4.162	32.524	64.308
p14-p20	1.464	30.133	62.533
p14-p21	2.661	31.565	63.761
p15-p1	-34.693	12.15	59.963
p15-p2	-60.406	-8.22	44.465
p15-p3	-38.805	8.103	56.069
p15-p4	-67.249	-12.546	42.308
p15-p5	-90.675	-31.761	28.167
p15-p6	-67.67	-13.296	41.245
p15-p7	-73.859	-17.825	38.205
p15-p8	-71.895	-15.402	40.444
p15-p9	-56.767	-5.805	46.272
p15-p10	-68.393	-11.8	43.545
p15-p11	-18.829	23.68	67.7
p15-p12	-73.861	-13.012	44.141
p15-p13	-40.134	11.449	61.208
p15-p14	-36.083	12.427	61.101
p15-p15	-53.414	0	53.414

p15-p16	15.118	47.458	83.502
p15-p17	14.511	46.951	83.091
p15-p18	4.976	39.807	77.934
p15-p19	11.849	44.951	81.639
p15-p20	9.151	42.56	79.864
p15-p21	10.348	43.992	81.092
p16-p1	-64.781	-35.308	-8.569
p16-p2	-90.494	-55.678	-24.067
p16-p3	-68.893	-39.355	-12.463
p16-p4	-97.337	-60.004	-26.224
p16-p5	-120.763	-79.219	-40.365
p16-p6	-97.758	-60.754	-27.287
p16-p7	-103.947	-65.283	-30.327
p16-p8	-101.983	-62.86	-28.088
p16-p9	-86.855	-53.263	-22.26
p16-p10	-98.481	-59.258	-24.987
p16-p11	-48.917	-23.778	-0.832
p16-p12	-103.949	-60.47	-24.391
p16-p13	-70.222	-36.009	-7.324
p16-p14	-66.171	-35.031	-7.431
p16-p15	-83.502	-47.458	-15.118
p16-p16	-14.97	0	14.97
p16-p17	-15.577	-0.507	14.559
p16-p18	-25.112	-7.651	9.402
p16-p19	-18.239	-2.507	13.107
p16-p20	-20.937	-4.898	11.332
p16-p21	-19.74	-3.466	12.56
p17-p1	-64.37	-34.801	-7.962
p17-p2	-90.083	-55.171	-23.46
p17-p3	-68.482	-38.848	-11.856
p17-p4	-96.926	-59.497	-25.617
p17-p5	-120.352	-78.712	-39.758
p17-p6	-97.347	-60.247	-26.68
p17-p7	-103.536	-64.776	-29.72
p17-p8	-101.572	-62.353	-27.481
p17-p9	-86.444	-52.756	-21.653
p17-p10	-98.07	-58.751	-24.38
p17-p11	-48.506	-23.271	-0.225
p17-p12	-103.538	-59.963	-23.784
p17-p13	-69.811	-35.502	-6.717
p17-p14	-65.76	-34.524	-6.824
p17-p15	-83.091	-46.951	-14.511
p17-p16	-14.559	0.507	15.577
p17-p17	-15.166	0	15.166
p17-p18	-24.701	-7.144	10.009
p17-p19	-17.828	-2	13.714

p17-p20	-20.526	-4.391	11.939
p17-p21	-19.329	-2.959	13.167
p18-p1	-59.213	-27.657	1.573
p18-p2	-84.926	-48.027	-13.925
p18-p3	-63.325	-31.704	-2.321
p18-p4	-91.769	-52.353	-16.082
p18-p5	-115.195	-71.568	-30.223
p18-p6	-92.19	-53.103	-17.145
p18-p7	-98.379	-57.632	-20.185
p18-p8	-96.415	-55.209	-17.946
p18-p9	-81.287	-45.612	-12.118
p18-p10	-92.913	-51.607	-14.845
p18-p11	-43.349	-16.127	9.31
p18-p12	-98.381	-52.819	-14.249
p18-p13	-64.654	-28.358	2.818
p18-p14	-60.603	-27.38	2.711
p18-p15	-77.934	-39.807	-4.976
p18-p16	-9.402	7.651	25.112
p18-p17	-10.009	7.144	24.701
p18-p18	-19.544	0	19.544
p18-p19	-12.671	5.144	23.249
p18-p20	-15.369	2.753	21.474
p18-p21	-14.172	4.185	22.702
p19-p1	-62.918	-32.801	-5.3
p19-p2	-88.631	-53.171	-20.798
p19-p3	-67.03	-36.848	-9.194
p19-p4	-95.474	-57.497	-22.955
p19-p5	-118.9	-76.712	-37.096
p19-p6	-95.895	-58.247	-24.018
p19-p7	-102.084	-62.776	-27.058
p19-p8	-100.12	-60.353	-24.819
p19-p9	-84.992	-50.756	-18.991
p19-p10	-96.618	-56.751	-21.718
p19-p11	-47.054	-21.271	2.437
p19-p12	-102.086	-57.963	-21.122
p19-p13	-68.359	-33.502	-4.055
p19-p14	-64.308	-32.524	-4.162
p19-p15	-81.639	-44.951	-11.849
p19-p16	-13.107	2.507	18.239
p19-p17	-13.714	2	17.828
p19-p18	-23.249	-5.144	12.671
p19-p19	-16.376	0	16.376
p19-p20	-19.074	-2.391	14.601
p19-p21	-17.877	-0.959	15.829
p20-p1	-61.143	-30.41	-2.602
p20-p2	-86.856	-50.78	-18.1

p20-p3	-65.255	-34.457	-6.496
p20-p4	-93.699	-55.106	-20.257
p20-p5	-117.125	-74.321	-34.398
p20-p6	-94.12	-55.856	-21.32
p20-p7	-100.309	-60.385	-24.36
p20-p8	-98.345	-57.962	-22.121
p20-p9	-83.217	-48.365	-16.293
p20-p10	-94.843	-54.36	-19.02
p20-p11	-45.279	-18.88	5.135
p20-p12	-100.311	-55.572	-18.424
p20-p13	-66.584	-31.111	-1.357
p20-p14	-62.533	-30.133	-1.464
p20-p15	-79.864	-42.56	-9.151
p20-p16	-11.332	4.898	20.937
p20-p17	-11.939	4.391	20.526
p20-p18	-21.474	-2.753	15.369
p20-p19	-14.601	2.391	19.074
p20-p20	-17.299	0	17.299
p20-p21	-16.102	1.432	18.527
p21-p1	-62.371	-31.842	-3.799
p21-p2	-88.084	-52.212	-19.297
p21-p3	-66.483	-35.889	-7.693
p21-p4	-94.927	-56.538	-21.454
p21-p5	-118.353	-75.753	-35.595
p21-p6	-95.348	-57.288	-22.517
p21-p7	-101.537	-61.817	-25.557
p21-p8	-99.573	-59.394	-23.318
p21-p9	-84.445	-49.797	-17.49
p21-p10	-96.071	-55.792	-20.217
p21-p11	-46.507	-20.312	3.938
p21-p12	-101.539	-57.004	-19.621
p21-p13	-67.812	-32.543	-2.554
p21-p14	-63.761	-31.565	-2.661
p21-p15	-81.092	-43.992	-10.348
p21-p16	-12.56	3.466	19.74
p21-p17	-13.167	2.959	19.329
p21-p18	-22.702	-4.185	14.172
p21-p19	-15.829	0.959	17.877
p21-p20	-18.527	-1.432	16.102
p21-p21	-17.33	0	17.33

5.4.7 Determination of Degree of Preference of matrix of one stretch over other stretch:

To obtain the degree of fuzzy preference of one stretch over another, the Fuzzy Preference Relational Matrix (E) is calculated. The fuzzy preference relational matrix E is calculated as mentioned in chapter 4. The values of E are obtained as shown in table 5.14.

Table 5.14 Fuzzy Preference Relational Matrix (E)

0.5	0.155	0.407	0.118	0.016	0.109	0.077	0.096	0.183	0.127	0.765	0.121	0.469	0.502	0.272	1	1	0.999	1	1	1
0.845	0.5	0.792	0.421	0.180	0.408	0.339	0.373	0.550	0.429	0.972	0.403	0.808	0.838	0.642	1	1	1	1	1	1
0.593	0.208	0.5	0.163	0.032	0.152	0.112	0.135	0.242	0.171	0.833	0.162	0.556	0.591	0.338	1	1	1	1	1	1
0.882	0.579	0.837	0.5	0.237	0.487	0.412	0.448	0.624	0.507	0.985	0.476	0.847	0.875	0.702	1	1	1	1	1	1
0.984	0.820	0.968	0.763	0.5	0.756	0.696	0.722	0.850	0.762	1	0.730	0.966	0.980	0.889	1	1	1	1	1	1
0.891	0.592	0.848	0.513	0.244	0.5	0.423	0.460	0.637	0.520	0.989	0.489	0.856	0.884	0.714	1	1	1	1	1	1
0.923	0.661	0.888	0.588	0.304	0.577	0.5	0.538	0.702	0.592	0.997	0.560	0.892	0.917	0.768	1	1	1	1	1	1
0.904	0.627	0.865	0.552	0.278	0.540	0.462	0.5	0.669	0.557	0.992	0.526	0.872	0.898	0.739	1	1	1	1	1	1
0.817	0.450	0.758	0.376	0.150	0.363	0.298	0.331	0.5	0.385	0.960	0.361	0.778	0.810	0.601	1	1	1	1	1	1
0.873	0.571	0.829	0.493	0.238	0.480	0.408	0.443	0.615	0.5	0.980	0.471	0.839	0.867	0.694	1	1	1	1	1	1
0.235	0.028	0.167	0.015	0	0.011	0.003	0.008	0.040	0.020	0.5	0.020	0.225	0.242	0.096	1	1	0.935	0.995	0.978	0.987
0.879	0.597	0.838	0.524	0.270	0.511	0.440	0.474	0.639	0.529	0.980	0.5	0.847	0.873	0.711	1	1	1	1	1	1
0.531	0.192	0.444	0.153	0.034	0.144	0.108	0.128	0.222	0.161	0.775	0.153	0.5	0.532	0.308	1	1	0.996	1	1	1
0.498	0.162	0.409	0.125	0.020	0.116	0.083	0.102	0.190	0.133	0.758	0.127	0.468	0.5	0.276	1	1	0.996	1	1	1
0.728	0.358	0.662	0.298	0.111	0.286	0.232	0.261	0.399	0.306	0.904	0.289	0.692	0.724	0.5	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.5	0.467	0.150	0.351	0.245	0.305
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.533	0.5	0.168	0.380	0.269	0.331
0.001	0	0	0	0	0	0	0	0	0	0.065	0	0.004	0.004	0	0.850	0.832	0.5	0.749	0.646	0.703
0	0	0	0	0	0	0	0	0	0	0.005	0	0	0	0	0.649	0.620	0.251	0.5	0.373	0.443
0	0	0	0	0	0	0	0	0	0	0.022	0	0	0	0	0.755	0.731	0.354	0.627	0.5	0.573
0	0	0	0	0	0	0	0	0	0	0.013	0	0	0	0	0.695	0.669	0.297	0.557	0.427	0.5

5.4.8 Assigning Ranks for all stretches:

After calculating the preference matrix, the row wise sum is performed, and Priority Index (PI) is obtained. These priority indices are assigned with ranks, where max value is assigned the least rank 1. The ranking is given as shown below in table 5.15.

Table 5.15 Assigned ranks for all stretches

Stretch No.	Priority Index	Rank
A1	-0.584075927734375	14
A2	4.00006103515625	8
A3	0.287994384765625	11
A4	4.89788818359375	6
A5	7.88610839843750	1
A6	5.05987548828125	5
A7	5.90686035156250	2
A8	5.48132324218750	3
A9	3.43792724609375	9
A10	4.80108642578125	7
A11	-2.99488830566406	15
A12	5.11279296875000	4
A13	-0.119628906250000	12
A14	-0.537597656250000	13
A15	2.25000000000000	10
A16	-8.48202514648438	21
A17	-8.31906127929688	20
A18	-6.14596557617188	16
A19	-7.65899658203125	19
A20	-6.93807983398438	17
A21	-7.34210205078125	18

5.5 CALCULATING THE PAVEMENT PRIORITIZATION USING ANALYTICAL HIERACHY PROCESS (AHP):

5.5.1 Developing Pairwise Comparison Matrix:

Pairwise Comparison Matrices (PCM) is used to compute for relative priorities of criteria or alternatives. They are marked using the Satty's scale as mention in chapter 4. The pairwise matrices are obtained from 15 different experts as shown in table 5.16 to 5.30.

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR
AC	1	5	7	7	1/3	1/7	1/5	1/3	1/4	8	2	1/5	1/4	1/8	1/2	1/4	1/6
BC	1/5	1	5	5	1/5	1/9	1/6	1/3	1/5	5	1/3	1/6	1/4	1/3	1/2	1/6	1/7
LC	1/7	1/5	1	3	1/6	1/9	1/7	1/5	1/7	2	1/2	1/7	1/6	1/2	1/2	1/7	1/8
TC	1/7	1/5	1/3	1	1/7	1/9	1/6	1/4	1/7	3	1/5	1/7	1/3	1/2	1/2	1/2	1/8
S	3	5	6	7	1	1/5	1/3	5	2	4	1/2	5	3	4	7	2	1/5
PH	7	9	9	9	5	1	7	6	5	8	6	7	5	3	8	9	1/3
RA	5	6	7	6	3	1/7	1	5	6	7	3	1	1/3	1/5	1/6	1/3	1/7
R	3	3	5	4	1/5	1/6	1/5	1	1/5	4	5	4	1/5	5	1/5	2	1/5
D	4	5	7	7	1/2	1/5	1/6	5	1	6	5	3	1/6	1/4	7	3	1/3
EC	1/8	1/5	1/2	1/3	1/4	1/8	1/7	1/4	1/6	1	1/3	3	1/3	1/3	1/5	1	1/9
P	1/2	3	2	5	2	1/6	1/3	1/5	1/5	3	1	1/5	1/7	1/7	1/5	1/3	1/8
MUD	5	6	7	7	1/5	1/7	1	1/4	1/3	1/3	5	1	1/9	1/8	1/6	1/7	1/7
POP	4	4	6	3	1/3	1/5	3	5	6	3	7	9	1	1/5	1/3	7	1/2
VOLT	8	3	2	2	1/4	1/3	5	1/5	4	3	7	8	5	1	7	5	1
FS	2	2	2	2	1/7	1/8	6	5	1/7	5	5	6	3	1/7	1	3	1/5
NTOWN	4	6	7	2	1/2	1/9	3	1/2	1/3	1	3	7	1/7	1/5	1/3	1	1/6
CBR	6	7	8	8	5	3	7	5	3	9	8	7	2	1	5	6	1

Table 5.16 Pair Wise Comparison Matrix for Expert 1

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR
AC	1	1/4	1/7	1/7	5	7	5	3	4	1/8	1/2	5	4	8	2	4	6
BC	4	1	1/5	1/5	5	9	6	3	5	1/5	3	6	4	3	2	6	7
LC	7	5	1	1/3	6	9	7	5	6	1/2	2	7	6	2	2	7	9
TC	7	5	3	1	7	9	6	4	7	1/3	5	7	3	2	2	2	9
S	1/5	1/5	1/6	1/7	1	5	3	1/5	1/3	1/4	2	1/5	1/3	1/4	1/7	1/2	5
PH	1/7	1/9	1/9	1/9	1/5	1	1/7	1/6	1/5	1/8	1/6	1/7	1/5	1/3	1/8	1/9	1
RA	1/5	1/6	1/7	1/6	1/3	7	1	1/5	1/6	1/7	1/3	1	3	5	6	3	7
R	1/3	1/3	1/5	1/4	5	6	5	1	5	1/4	1/5	1/4	5	1/5	5	1/2	5
D	1/4	1/5	1/6	1/7	3	5	6	1/5	1	1/6	1/5	1/3	6	4	1/7	1/3	3
EC	8	5	2	3	4	8	7	4	6	1	3	1/3	3	3	5	1	9
P	2	1/3	1/2	1/5	1/2	6	3	5	5	1/3	1	5	7	7	5	3	8
MUD	1/5	1/6	1/7	1/7	5	7	1	4	3	3	1/5	1	9	8	6	7	7
POP	1/4	1/4	1/6	1/3	3	5	1/3	1/5	1/6	1/3	1/7	1/9	1	5	3	1/7	2
VOLT	1/8	1/3	1/2	1/2	4	3	1/5	5	1/4	1/3	1/7	1/8	1/5	1	1/7	1/5	1
FS	1/2	1/2	1/2	1/2	7	8	1/6	1/5	7	1/5	1/5	1/6	1/3	7	1	1/3	5
NTOWN	1/4	1/6	1/7	1/2	2	9	1/3	2	3	1	1/3	1/7	7	5	3	1	5
CBR	1/6	1/7	1/9	1/9	1/5	1	1/7	1/5	1/3	1/9	1/8	1/7	1/2	1	1/5	1/5	1

Table 5.17 Pair Wise Comparison Matrix for Expert 2

	AC	**BC**	**LC**	**TC**	**S**	**PH**	**RA**	**R**	**D**	**EC**	**P**	**MUD**	**POP**	**VOLT**	**FS**	**NTOWN**	**CBR**
AC	1	1	1/5	1/3	1/3	7	9	8	1/5	1/4	1/6	1/7	4	8	2	4	6
BC	1	1	1/3	1/3	1/5	6	5	3	1/5	3	1/7	1/9	4	3	2	6	7
LC	5	3	1	1/2	1/5	7	7	5	1/5	3	1/5	1/7	6	2	2	7	9
TC	3	3	2	1	1	7	7	5	1/5	3	1/5	1/7	3	2	2	2	9
S	3	5	5	1	1	3	4	1/2	1	1/3	1/8	1/9	1/3	1/4	1/7	1/2	5
PH	1/7	1/6	1/7	1/7	1/3	1	1/5	1/8	1/5	1/8	1/8	1/9	1/5	1/3	1/8	1/9	1
RA	1/9	1/5	1/7	1/7	1/4	5	1	1/2	1/4	1/7	1/5	1/5	3	5	6	3	7
R	1/8	1/3	1/5	1/5	2	8	2	1	1/5	1/4	1/5	1/3	5	1/5	5	1/2	5
D	5	5	5	5	1	5	4	5	1	1/7	1/5	1/7	6	4	1/7	1/3	3
EC	4	1/3	1/3	1/3	3	8	7	4	7	1	1/4	1/7	3	3	5	1	9
P	6	7	5	5	8	8	5	5	5	4	1	1/7	7	7	5	3	8
MUD	7	9	7	7	9	9	5	3	7	7	7	1	9	8	6	7	7
POP	1/4	1/4	1/6	1/3	3	5	1/3	1/5	1/6	1/3	1/7	1/9	1	5	3	1/7	2
VOL	1/8	1/3	1/2	1/2	4	3	1/5	5	1/4	1/3	1/7	1/8	1/5	1	1/7	1/5	1
FS	1/2	1/2	1/2	1/2	7	8	1/6	1/5	7	1/5	1/5	1/6	1/3	7	1	1/3	5
NTOWN	1/4	1/6	1/7	1/2	2	9	1/3	2	3	1	1/3	1/7	7	5	3	1	5
CBR	1/6	1/7	1/9	1/9	1/5	1	1/7	1/5	1/3	1/9	1/8	1/7	1/2	1	1/5	1/5	1

Table 5.18 Pair Wise Comparison Matrix for Expert 3

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR
AC	1	1	1/4	1/3	1/3	5	8	7	1/5	1/4	1/7	1/7	6	8	2	4	6
BC	1	1	1/3	1/2	1/5	6	5	3	1/5	3	1/7	1/7	7	7	3	6	8
LC	4	3	1	1/3	1/5	7	6	5	1/5	3	1/5	1/6	9	7	3	9	5
TC	3	2	3	1	1	7	6	3	1/5	3	1/5	1/7	9	7	3	9	5
S	3	5	5	1	1	3	4	1/2	1	1/3	1/8	1/9	1	5	1/7	4	7
PH	1/5	1/6	1/7	1/7	1/3	1	1/7	1/8	1/5	1/7	1/9	1/9	3	1	1/7	1/5	1
RA	1/8	1/5	1/6	1/6	1/4	7	1	1/2	1/4	1/5	1/7	1/3	5	1/2	6	1/3	2
R	1/7	1/3	1/5	1/3	2	8	2	1	1/4	1/5	1/7	1/3	7	3	5	2	5
D	5	5	5	5	1	5	4	4	1	1/5	1/7	1/7	7	3	1/7	4	5
EC	4	1/3	1/3	1/3	3	7	5	5	5	1	1/2	1/7	9	5	5	1	9
P	7	7	5	5	8	9	7	7	7	2	1	1/5	9	7	5	5	9
MUD	7	7	6	7	9	9	3	3	7	7	5	1	5	1/5	6	1/5	1
POP	1/6	1/7	1/9	1/9	1	1/3	1/5	1/7	1/7	1/9	1/9	1/5	1	3	3	1/3	1/5
VOL	1/8	1/7	1/7	1/7	1/5	1	2	1/3	1/3	1/5	1/7	5	1/3	1	1/7	1/3	1/5
FS	1/2	1/3	1/3	1/3	7	7	1/6	1/5	7	1/5	1/5	1/6	1/3	7	1	1/6	1/3
NTOWN	1/4	1/6	1/9	1/9	1/4	5	3	1/2	1/4	1	1/5	5	3	3	6	1	3
CBR	1/6	1/8	1/5	1/5	1/7	1	1/2	1/5	1/5	1/9	1/9	1	5	5	3	1/3	1

Table 5.19 Pair Wise Comparison Matrix for Expert 4

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR
AC	1	1/3	1/5	1/5	6	9	4	1/3	1/3	1/5	1	4	6	8	2	4	6
BC	3	1	1/4	1/4	7	9	6	3	5	1/5	3	5	7	7	3	6	8
LC	5	4	1	1	5	9	7	5	6	1/2	2	8	9	7	3	9	5
TC	5	4	1	1	5	9	7	7	8	1/3	5	8	9	7	3	9	5
S	1/6	1/7	1/5	1/5	1	5	3	1/5	1	1/4	2	1/5	1	5	1/7	4	7
PH	1/9	1/9	1/9	1/9	1/5	1	5	1/7	1/5	1/8	1/6	1/9	3	1	1/7	1/5	1
RA	1/4	1/6	1/7	1/7	1/3	1/5	1	1/5	1/7	1/7	1/3	1/3	5	1/2	6	1/3	2
R	3	1/3	1/5	1/7	5	7	5	1	5	1/4	1/5	1/2	7	3	5	2	5
D	3	1/5	1/6	1/8	1	5	7	1/5	1	1/6	1/5	1/4	7	3	1/7	4	5
EC	5	5	2	3	4	8	7	4	6	1	1/2	1/3	9	5	5	1	9
P	1	1/3	1/2	1/5	1/2	6	3	5	5	2	1	1/2	9	7	5	5	9
MUD	1/4	1/5	1/8	1/8	5	9	3	2	4	3	2	1	5	1/5	6	1/5	1
POP	1/6	1/7	1/9	1/9	1	1/3	1/5	1/7	1/7	1/9	1/9	1/5	1	3	3	1/3	1/5
VOL	1/8	1/7	1/7	1/7	1/5	1	2	1/3	1/3	1/5	1/7	5	1/3	1	1/7	1/3	1/5
FS	1/2	1/3	1/3	1/3	7	7	1/6	1/5	7	1/5	1/5	1/6	1/3	7	1	1/6	1/3
NTOWN	1/4	1/6	1/9	1/9	1/4	5	3	1/2	1/4	1	1/5	5	3	3	6	1	3
CBR	1/6	1/8	1/5	1/5	1/7	1	1/2	1/5	1/5	1/9	1/9	1	5	5	3	1/3	1

Table 5.20 Pair Wise Comparison Matrix for Expert 5

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR
AC	1	1/5	1/7	1/7	3	7	5	3	4	1/8	1/2	5	4	8	2	4	6
BC	5	1	1/5	1/5	5	9	6	3	5	1/5	3	6	4	3	2	6	7
LC	7	5	1	1	5	9	7	5	6	1/2	2	8	9	7	3	9	5
TC	7	5	1	1	1/5	1/9	1/7	1/7	1/8	3	1/5	1/8	1/9	1/7	1/3	1/9	1/5
S	1/3	1/5	1/5	5	1	1/5	1/3	5	2	4	1/2	5	3	4	7	2	1/5
PH	1/7	1/9	1/9	9	5	1	1/5	1/8	1/5	1/8	1/8	1/9	1/5	1/3	1/8	1/9	1
RA	1/5	1/6	1/7	7	3	5	1	1/2	1/4	1/5	1/7	1/3	5	1/2	6	1/3	2
R	1/3	1/3	1/5	7	1/5	8	2	1	5	4	5	3	1/5	5	1/5	2	1/5
D	1/4	1/5	1/6	8	1/2	5	4	1/5	1	1/5	1/7	1/7	7	3	1/7	4	5
EC	8	5	2	1/3	1/4	8	5	1/4	5	1	3	1/3	3	3	5	1	9
P	2	1/3	1/2	5	2	8	7	1/5	7	1/3	1	5	7	7	5	3	8
MUD	1/5	1/6	1/8	8	1/5	9	3	1/3	7	3	1/5	1	1/5	5	1/6	5	1
POP	1/4	1/4	1/9	9	1/3	5	1/5	5	1/7	1/3	1/7	5	1	1/3	1/3	3	5
VOLT	1/8	1/3	1/7	7	1/4	3	2	1/5	1/3	1/3	1/7	1/5	3	1	1/7	1/3	1/5
FS	1/2	1/2	1/3	3	1/7	8	1/6	5	7	1/5	1/5	6	3	7	1	1/6	1/3
NTOWN	1/4	1/6	1/9	9	1/2	9	3	1/2	1/4	1	1/3	1/5	1/3	3	6	1	3
CBR	1/6	1/7	1/5	5	5	1	1/2	5	1/5	1/9	1/8	1	1/5	5	3	1/3	1

Table 5.21 Pair Wise Comparison Matrix for Expert 6

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR
AC	1	4	7	7	1/5	1/7	1/5	1/3	1/4	8	2	1/5	1/4	1/8	1/2	1/4	1/6
BC	1/4	1	5	5	1/5	1/9	1/6	1/3	1/5	5	1/3	1/6	1/4	1/3	1/2	1/6	1/7
LC	1/7	1/5	1	3	1/6	1/9	1/7	1/5	1/6	2	1/2	1/7	1/6	1/2	1/2	1/7	1/9
TC	1/7	1/5	1/3	1	1/7	1/9	1/6	1/4	1/7	3	1/5	1/7	1/3	1/2	1/2	1/2	1/9
S	5	5	6	7	1	1/5	1/3	5	3	4	1/2	5	3	4	7	2	1/5
PH	7	9	9	9	5	1	7	6	5	8	6	7	5	3	8	9	1
RA	5	6	7	6	3	1/7	1	5	6	7	3	1	1/3	1/5	1/6	1/3	1/7
R	3	3	5	4	1/5	1/6	1/5	1	1/5	4	5	4	1/5	5	1/5	2	1/5
D	4	5	6	7	1/3	1/5	1/6	5	1	6	5	3	1/6	1/4	7	3	1/3
EC	1/8	1/5	1/2	1/3	1/4	1/8	1/7	1/4	1/6	1	1/3	3	1/3	1/3	1/5	1	1/9
P	1/2	3	2	5	2	1/6	1/3	1/5	1/5	3	1	1/5	1/7	1/7	1/5	1/3	1/8
MUD	5	6	7	7	1/5	1/7	1	1/4	1/3	1/3	5	1	1/9	1/8	1/6	1/7	1/7
POP	4	4	6	3	1/3	1/5	3	5	6	3	7	9	1	1/5	1/3	7	1/2
VOLT	8	3	2	2	1/4	1/3	5	1/5	4	3	7	8	5	1	7	5	1
FS	2	2	2	2	1/7	1/8	6	5	1/7	5	5	6	3	1/7	1	3	1/5
NTOWN	4	6	7	2	1/2	1/9	3	1/2	1/3	1	3	7	1/7	1/5	1/3	1	1/5
CBR	6	7	9	9	5	1	7	5	3	9	8	7	2	1	5	5	1

Table 5.22 Pair Wise Comparison Matrix for Expert 7

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR
AC	1	5	7	7	1/3	1/7	1/5	1/3	1/4	8	2	1/5	1/4	1/8	1/2	1/4	1/6
BC	1/5	1	5	5	1/5	1/9	1/6	1/3	1/5	5	1/3	1/6	1/4	1/3	1/2	1/6	1/7
LC	1/7	1/5	1	1	1/5	1/9	1/7	1/5	1/6	2	1/2	1/8	1/9	1/7	1/3	1/9	1/5
TC	1/7	1/5	1	1	5	9	7	7	8	1/3	5	8	9	7	3	9	5
S	3	5	5	1/5	1	5	3	1/5	1/2	1/4	2	1/5	1/3	1/4	1/7	1/2	5
PH	7	9	9	1/9	1/5	1	5	8	5	8	8	9	5	3	8	9	1
RA	5	6	7	1/7	1/3	1/5	1	2	4	5	7	3	1/5	2	1/6	3	1/2
R	3	3	5	1/7	5	1/8	1/2	1	1/5	1/4	1/5	1/3	5	1/5	5	1/2	5
D	4	5	6	1/8	2	1/5	1/4	5	1	5	7	7	1/7	1/3	7	1/4	1/5
EC	1/8	1/5	1/2	3	4	1/8	1/5	4	1/5	1	1/3	3	1/3	1/3	1/5	1	1/9
P	1/2	3	2	1/5	1/2	1/8	1/7	5	1/7	3	1	1/5	1/7	1/7	1/5	1/3	1/8
MUD	5	6	8	1/8	5	1/9	1/3	3	1/7	1/3	5	1	5	1/5	6	1/5	1
POP	4	4	9	1/9	3	1/5	5	1/5	7	3	7	1/5	1	3	3	1/3	1/5
VOLT	8	3	7	1/7	4	1/3	1/2	5	3	3	7	5	1/3	1	7	3	5
FS	2	2	3	1/3	7	1/8	6	1/5	1/7	5	5	1/6	1/3	1/7	1	6	3
NTOWN	4	6	9	1/9	2	1/9	1/3	2	4	1	3	5	3	1/3	1/6	1	1/3
CBR	6	7	5	1/5	1/5	1	2	1/5	5	9	8	1	5	1/5	1/3	3	1

Table 5.23 Pair Wise Comparison Matrix for Expert 8

CRITERIA	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR
AC	1	2	5	5	1/3	1/9	1/5	1/3	1/5	5	3	4	6	8	3	5	8
BC	1/2	1	5	5	1/5	1/9	1/5	1/3	1/5	5	1/4	6	5	6	3	7	8
LC	1/5	1/5	1	1	1/5	1/9	1/7	1/5	1/6	2	1/2	8	9	7	3	9	5
TC	1/5	1/5	1	1	5	9	7	7	8	1/3	5	1/8	1/9	1/7	1/3	1/9	1/5
S	3	5	5	1/5	1	5	3	1/5	1/2	1/4	2	5	3	4	7	2	1/5
PH	9	9	9	1/9	1/5	1	5	8	1	8	8	1/9	1/5	1/3	1/8	1/9	1
RA	5	5	7	1/7	1/3	1/5	1	2	4	5	7	1/3	5	1/2	6	1/3	2
R	3	3	5	1/7	5	1/8	1/2	1	1/5	1/4	1/5	3	1/5	5	1/5	2	1/5
D	5	5	6	1/8	2	1	1/4	5	1	5	7	1/7	7	3	1/7	4	5
EC	1/5	1/5	1/2	3	4	1/8	1/5	4	1/5	1	1/3	1/3	3	3	5	1	9
P	1/3	4	2	1/5	1/2	1/8	1	5	1/7	3	1	5	7	7	5	3	8
MUD	1/4	1/6	1/8	8	1/5	9	1	1/3	7	3	1/5	1	1/5	5	1/6	5	1
POP	1/6	1/5	1/9	9	1/3	5	1/5	5	1/7	1/3	1/7	5	1	1/3	1/3	3	5
VOLT	1/8	1/6	1/7	7	1/4	3	2	1/5	1/3	1/3	1/7	1/5	3	1	1/7	1/3	1/5
FS	1/3	1/3	1/3	3	1/7	8	1/6	5	7	1/5	1/5	6	3	7	1	1/6	1/3
NTOWN	1/5	1/7	1/9	9	1/2	9	3	1/2	1/4	1	1/3	1/5	1/3	3	6	1	3
CBR	1/8	1/8	1/5	5	5	1	1/2	5	1/5	1/9	1/8	1	1/5	5	3	1/3	1

Table 5.24 Pair Wise Comparison Matrix for Expert 9

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOL	FS	NTOWN	CBR
AC	1	3	5	5	1/6	1/9	1/4	3	3	5	1	1/4	1/6	1/8	1/2	1/4	1/6
BC	1/3	1	4	4	1/7	1/9	1/6	1/3	1/5	5	1/3	1/5	1/7	1/7	1/3	1/6	1/8
LC	1/5	1/4	1	1	1/5	1/9	1/7	1/5	1/6	2	1/2	1/8	1/9	1/7	1/3	1/9	1/5
TC	1/5	1/4	1	1	1/5	1/9	1/7	1/7	1/8	3	1/5	1/8	1/9	1/7	1/3	1/9	1/5
S	6	7	5	5	1	1/5	1/3	5	1	4	1/2	5	1	1/5	7	1/4	1/7
PH	9	9	9	9	5	1	1/5	7	5	8	6	9	1/3	1	7	5	1
RA	4	6	7	7	3	5	1	5	7	7	3	3	1/5	2	1/6	3	1/2
R	1/3	3	5	7	1/5	1/7	1/5	1	1/5	4	5	2	1/7	1/3	1/5	1/2	1/5
D	1/3	5	6	8	1	1/5	1/7	5	1	6	5	4	1/7	1/3	7	1/4	1/5
EC	1/5	1/5	1/2	1/3	1/4	1/8	1/7	1/4	1/6	1	2	3	1/9	1/5	1/5	1	1/9
P	1	3	2	5	2	1/6	1/3	1/5	1/5	1/2	1	2	1/9	1/7	1/5	1/5	1/9
MUD	4	5	8	8	1/5	1/9	1/3	1/2	1/4	1/3	1/2	1	1/5	5	1/6	5	1
POP	6	7	9	9	1	3	5	7	7	9	9	5	1	1/3	1/3	3	5
VOL	8	7	7	7	5	1	1/2	3	3	5	7	1/5	3	1	7	3	5
FS	2	3	3	3	1/7	1/7	6	5	1/7	5	5	6	3	1/7	1	6	3
NTOWN	4	6	9	9	4	1/5	1/3	2	4	1	5	1/5	1/3	1/3	1/6	1	1/3
CBR	6	8	5	5	7	1	2	5	5	9	9	1	1/5	1/5	1/3	3	1

Table 5.25 Pair Wise Comparison Matrix for Expert 10

CRITERIA	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOL	FS	NTOWN	CBR
AC	1	1	7	5	3	1/9	1/7	1/8	5	6	4	7	1/6	1/8	1/2	1/4	1/6
BC	1	1	5	5	3	1/5	1/6	1/3	5	1/5	9	7	1/7	1/7	1/3	1/6	1/8
LC	1/7	1/5	1	5	5	1/7	1/5	1/5	4	1/5	7	5	1/9	1/7	1/3	1/9	1/5
TC	1/5	1/5	1/5	1	1	1/5	1/8	1/3	1/5	1/5	9	5	1/9	1/7	1/3	1/9	1/5
S	1/3	1/3	1/5	1	1	3	1/6	3	1	5	6	9	1	1/5	7	1/4	1/7
PH	9	5	7	5	1/3	1	7	8	7	7	7	9	1/3	1	7	5	1
RA	7	6	5	8	6	1/7	1	6	2	5	9	5	1/5	2	1/6	3	1/2
R	8	3	5	3	1/3	1/8	1/6	1	5	4	9	7	1/7	1/3	1/5	1/2	1/5
D	1/5	1/5	1/4	5	1	1/7	1/2	1/5	1	9	5	7	1/7	1/3	7	1/4	1/5
EC	1/6	5	5	5	1/5	1/7	1/5	1/4	1/9	1	6	5	1/9	1/5	1/5	1	1/9
P	1/4	1/9	1/7	1/9	1/6	1/7	1/9	1/9	1/5	1/6	1	5	1/9	1/7	1/5	1/5	1/9
MUD	1/7	1/7	1/5	1/5	1/9	1/9	1/5	1/7	1/7	1/5	1/5	1	1/5	5	1/6	5	1
POP	6	7	9	9	1	3	5	7	7	9	9	5	1	1/3	1/3	3	5
VOL	8	7	7	7	5	1	1/2	3	3	5	7	1/5	3	1	7	3	5
FS	2	3	3	3	1/7	1/7	6	5	1/7	5	5	6	3	1/7	1	6	3
NTOWN	4	6	9	9	4	1/5	1/3	2	4	1	5	1/5	1/3	1/3	1/6	1	1/3
CBR	6	8	5	5	7	1	2	5	5	9	9	1	1/5	1/5	1/3	3	1

Table 5.26 Pair Wise Comparison Matrix for Expert 11

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOL	FS	NTOWN	CBR
AC	1	1	5	3	3	1/7	1/9	1/8	5	4	6	7	1/4	1/8	1/2	1/4	1/6
BC	1	1	3	3	5	1/6	1/5	1/3	5	1/3	7	9	1/4	1/3	1/2	1/6	1/7
LC	1/5	1/3	1	2	5	1/7	1/7	1/5	5	1/3	5	7	1/6	1/2	1/2	1/7	1/9
TC	1/3	1/3	1/2	1	1	1/7	1/7	1/5	5	1/3	5	7	1/3	1/2	1/2	1/2	1/9
S	1/3	1/5	1/5	1	1	1/3	1/4	2	1	3	8	9	3	4	7	2	1/5
PH	7	6	7	7	3	1	5	8	5	8	8	9	5	3	8	9	1
RA	9	5	7	7	4	1/5	1	2	4	7	5	5	1/3	1/5	1/6	1/3	1/7
R	8	3	5	5	1/2	1/8	1/2	1	5	4	5	3	1/5	5	1/5	2	1/5
D	1/5	1/5	1/5	1/5	1	1/5	1/4	1/5	1	7	5	7	1/6	1/4	7	3	1/3
EC	1/4	3	3	3	1/3	1/8	1/7	1/4	1/7	1	4	7	1/3	1/3	1/5	1	1/9
P	1/6	1/7	1/5	1/5	1/8	1/8	1/5	1/5	1/5	1/4	1	7	1/7	1/7	1/5	1/3	1/8
MUD	1/7	1/9	1/7	1/7	1/9	1/9	1/5	1/3	1/7	1/7	1/7	1	1/9	1/8	1/6	1/7	1/7
POP	4	4	6	3	1/3	1/5	3	5	6	3	7	9	1	1/5	1/3	7	1/2
VOL	8	3	2	2	1/4	1/3	5	1/5	4	3	7	8	5	1	7	5	1
FS	2	2	2	2	1/7	1/8	6	5	1/7	5	5	6	3	1/7	1	3	1/5
NTOWN	4	6	7	2	1/2	1/9	3	1/2	1/3	1	3	7	1/7	1/5	1/3	1	1/5
CBR	6	7	9	9	5	1	7	5	3	9	8	7	2	1	5	5	1

Table 5.27 Pair Wise Comparison Matrix for Expert 12

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR
AC	1	1/2	1/5	1/5	3	9	5	3	5	1/5	1/3	1/4	1/6	1/8	1/3	1/5	1/8
BC	2	1	1/5	1/5	5	9	5	3	5	1/5	4	1/6	1/5	1/6	1/3	1/7	1/8
LC	5	5	1	1	5	9	7	5	6	1/2	2	1/8	1/9	1/7	1/3	1/9	1/5
TC	5	5	1	1	1/5	1/9	1/7	1/7	1/8	3	1/5	8	9	7	3	9	5
S	1/3	1/5	1/5	5	1	1/5	1/3	5	2	4	1/2	1/5	1/3	1/4	1/7	1/2	5
PH	1/9	1/9	1/9	9	5	1	1/5	1/8	1	1/8	1/8	9	5	3	8	9	1
RA	1/5	1/5	1/7	7	3	5	1	1/2	1/4	1/5	1	1	1/5	2	1/6	3	1/2
R	1/3	1/3	1/5	7	1/5	8	2	1	5	4	5	1/3	5	1/5	5	1/2	5
D	1/5	1/5	1/6	8	1/2	1	4	1/5	1	1/5	1/7	7	1/7	1/3	7	1/4	1/5
EC	5	5	2	1/3	1/4	8	5	1/4	5	1	3	3	1/3	1/3	1/5	1	1/9
P	3	1/4	1/2	5	2	8	7	1/5	7	1/3	1	1/5	1/7	1/7	1/5	1/3	1/8
MUD	4	6	8	1/8	5	1/9	1/3	3	1/7	1/3	5	1	5	1/5	6	1/5	1
POP	6	5	9	1/9	3	1/5	5	1/5	7	3	7	1/5	1	3	3	1/3	1/5
VOLT	8	6	7	1/7	4	1/3	1/2	5	3	3	7	5	1/3	1	7	3	5
FS	3	3	3	1/3	7	1/8	6	1/5	1/7	5	5	1/6	1/3	1/7	1	6	3
NTOWN	5	7	9	1/9	2	1/9	1/3	2	4	1	3	5	3	1/3	1/6	1	1/3
CBR	8	8	5	1/5	1/5	1	2	1/5	5	9	8	1	5	1/5	1/3	3	1

Table 5.28 Pair Wise Comparison Matrix for Expert 13

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOL	FS	NTOWN	CBR
AC	1	1	1/7	1/5	1/3	9	7	8	1/5	1/6	1/4	1/7	6	8	2	4	6
BC	1	1	1/5	1/5	1/3	5	6	3	1/5	5	1/9	1/7	7	7	3	6	8
LC	7	5	1	1/5	1/5	7	5	5	1/4	5	1/7	1/5	9	7	3	9	5
TC	5	5	5	1	1	5	8	3	5	5	1/9	1/5	9	7	3	9	5
S	3	3	5	1	1	1/3	6	1/3	1	1/5	1/6	1/9	1	5	1/7	4	7
PH	1/9	1/5	1/7	1/5	3	1	1/7	1/8	1/7	1/7	1/7	1/9	3	1	1/7	1/5	1
RA	1/7	1/6	1/5	1/8	1/6	7	1	1/6	1/2	1/5	1/9	1/5	5	1/2	6	1/3	2
R	1/8	1/3	1/5	1/3	3	8	6	1	1/5	1/4	1/9	1/7	7	3	5	2	5
D	5	5	4	1/5	1	7	2	5	1	1/9	1/5	1/7	7	3	1/7	4	5
EC	6	1/5	1/5	1/5	5	7	5	4	9	1	1/6	1/5	9	5	5	1	9
P	4	9	7	9	6	7	9	9	5	6	1	1/5	9	7	5	5	9
MUD	7	7	5	5	9	9	5	7	7	5	5	1	5	1/5	6	1/5	1
POP	1/6	1/7	1/9	1/9	1	1/3	1/5	1/7	1/7	1/9	1/9	1/5	1	3	3	1/3	1/5
VOL	1/8	1/7	1/7	1/7	1/5	1	2	1/3	1/3	1/5	1/7	5	1/3	1	1/7	1/3	1/5
FS	1/2	1/3	1/3	1/3	7	7	1/6	1/5	7	1/5	1/5	1/6	1/3	7	1	1/6	1/3
NTOWN	1/4	1/6	1/9	1/9	1/4	5	3	1/2	1/4	1	1/5	5	3	3	6	1	3
CBR	1/6	1/8	1/5	1/5	1/7	1	1/2	1/5	1/5	1/9	1/9	1	5	5	3	1/3	1

Table 5.29 Pair Wise Comparison Matrix for Expert 14

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOL	FS	NTOWN	CBR
AC	1	1	4	3	3	1/5	1/8	1/7	5	4	7	7	1/6	1/8	1/2	1/4	1/6
BC	1	1	3	2	5	1/6	1/5	1/3	5	1/3	7	7	1/7	1/7	1/3	1/6	1/8
LC	1/4	1/3	1	3	5	1/7	1/6	1/5	5	1/3	5	6	1/9	1/7	1/3	1/9	1/5
TC	1/3	1/2	1/3	1	1	1/7	1/6	1/3	5	1/3	5	7	1/9	1/7	1/3	1/9	1/5
S	1/3	1/5	1/5	1	1	1/3	1/4	2	1	3	8	9	1	1/5	7	1/4	1/7
PH	5	6	7	7	3	1	7	8	5	7	9	9	1/3	1	7	5	1
RA	8	5	6	6	4	1/7	1	2	4	5	7	3	1/5	2	1/6	3	1/2
R	7	3	5	3	1/2	1/8	1/2	1	4	5	7	3	1/7	1/3	1/5	1/2	1/5
D	1/5	1/5	1/5	1/5	1	1/5	1/4	1/4	1	5	7	7	1/7	1/3	7	1/4	1/5
EC	1/4	3	3	3	1/3	1/7	1/5	1/5	1/5	1	2	7	1/9	1/5	1/5	1	1/9
P	1/7	1/7	1/5	1/5	1/8	1/9	1/7	1/7	1/7	1/2	1	5	1/9	1/7	1/5	1/5	1/9
MUD	1/7	1/7	1/6	1/7	1/9	1/9	1/3	1/3	1/7	1/7	1/5	1	1/5	5	1/6	5	1
POP	6	7	9	9	1	3	5	7	7	9	9	5	1	1/3	1/3	3	5
VOL	8	7	7	7	5	1	1/2	3	3	5	7	1/5	3	1	7	3	5
FS	2	3	3	3	1/7	1/7	6	5	1/7	5	5	6	3	1/7	1	6	3
NTOWN	4	6	9	9	4	1/5	1/3	2	4	1	5	1/5	1/3	1/3	1/6	1	1/3
CBR	6	8	5	5	7	1	2	5	5	9	9	1	1/5	1/5	1/3	3	1

Table 5.30 Pair Wise Comparison Matrix for Expert 1

5.5.2: Determination of Combination pairwise matrix

From the expert pairwise combination matrices obtained, the matrices are combined by geometric mean with their respective values of the row to obtain one single pairwise matrix to work. The combined pairwise matrix is obtained as shown in table 5.31.

Table 5.31 Combined Pair Wise Comparison Matrix

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR
AC	1.000	2.236	2.646	2.646	0.577	0.378	0.447	0.577	0.500	2.828	1.414	0.447	0.500	0.354	0.707	0.500	0.408
BC	0.447	1.000	2.236	2.236	0.447	0.333	0.408	0.577	0.447	2.236	0.577	0.408	0.500	0.577	0.707	0.408	0.378
LC	0.378	0.447	1.000	1.732	0.408	0.333	0.378	0.447	0.378	1.414	0.707	0.378	0.408	0.707	0.707	0.378	0.354
TC	0.378	0.447	0.577	1.000	0.378	0.333	0.408	0.500	0.378	1.732	0.447	0.378	0.577	0.707	0.707	0.707	0.354
S	1.732	2.236	2.449	2.646	1.000	0.447	0.577	2.236	1.414	2.000	0.707	2.236	1.732	2.000	2.646	1.414	0.447
PH	2.646	3.000	3.000	3.000	2.236	1.000	2.646	2.449	2.236	2.828	2.449	2.646	2.236	1.732	2.828	3.000	0.577
RA	2.236	2.449	2.646	2.449	1.732	0.378	1.000	2.236	2.449	2.646	4.583	0.577	0.577	0.447	0.408	0.577	0.378
R	1.732	1.732	2.236	2.000	0.447	0.408	0.447	1.000	0.447	2.000	2.236	2.000	0.447	2.236	0.447	1.414	0.447
D	2.000	2.236	2.646	2.646	0.707	0.447	0.408	2.236	1.000	2.449	2.236	1.732	0.408	0.500	2.646	1.732	0.577
EC	0.354	0.447	0.707	0.577	0.500	0.354	0.378	0.500	0.408	1.000	0.577	1.732	0.577	0.577	0.447	1.000	0.333
P	0.707	1.732	1.414	2.236	1.414	0.408	1.528	0.447	0.447	1.732	1.000	0.447	0.378	0.378	0.447	0.577	0.354
MUD	2.236	2.449	2.646	2.646	0.447	0.378	0.577	0.500	0.577	0.577	2.236	1.000	0.333	0.354	0.408	0.378	0.378
POP	2.000	2.000	2.449	1.732	0.577	0.447	1.732	2.236	2.449	1.732	2.646	3.000	1.000	0.447	0.577	2.646	0.707
VOLT	2.828	1.732	1.414	1.414	0.500	0.577	2.236	0.447	2.000	1.732	2.646	2.828	2.236	1.000	2.646	2.236	1.000
FS	1.414	1.414	1.414	1.414	0.378	0.354	2.449	2.236	0.378	2.236	2.236	2.449	1.732	0.378	1.000	1.732	0.447
NTOWN	2.000	2.449	2.646	1.414	0.707	0.333	1.732	0.707	0.577	1.000	1.732	2.646	0.378	0.447	0.577	1.000	0.408
CBR	2.449	2.646	2.828	2.828	2.236	1.732	2.646	2.236	1.732	3.000	2.828	2.646	1.414	1.000	2.236	2.449	1.000

5.5.3 Determination of Normalised Pairwise Matrix:

From the combined pairwise matrix obtained, each value under a criteria column is considered and is divided by the sum the whole column, for each criterion, in order to normalize the matrix. The normalized pairwise matrix is obtained as shown in table 5.32.

Table 5.32 Normalized Pair Wise Matrix

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR
0	0.0377	0.0729	0.0757	0.0764	0.0393	0.0437	0.0224	0.0268	0.0281	0.0853	0.0452	0.0162	0.0324	0.0255	0.0351	0.0226	0.0478
1	0.0169	0.0326	0.0640	0.0646	0.0304	0.0386	0.0204	0.0268	0.0251	0.0675	0.0185	0.0148	0.0324	0.0417	0.0351	0.0184	0.0442
2	0.0142	0.0146	0.0286	0.0500	0.0278	0.0386	0.0189	0.0207	0.0212	0.0427	0.0226	0.0137	0.0264	0.0511	0.0351	0.0171	0.0414
3	0.0142	0.0146	0.0165	0.0289	0.0257	0.0386	0.0204	0.0232	0.0212	0.0523	0.0143	0.0137	0.0374	0.0511	0.0351	0.0319	0.0414
4	0.0653	0.0729	0.0701	0.0764	0.0681	0.0517	0.0289	0.1037	0.0794	0.0603	0.0226	0.0812	0.1122	0.1445	0.1313	0.0638	0.0523
5	0.0997	0.0979	0.0858	0.0867	0.1522	0.1157	0.1323	0.1136	0.1255	0.0853	0.0784	0.0960	0.1449	0.1251	0.1404	0.1354	0.0675
6	0.0843	0.0799	0.0757	0.0708	0.1179	0.0437	0.0500	0.1037	0.1375	0.0798	0.1466	0.0210	0.0374	0.0323	0.0203	0.0261	0.0442
7	0.0653	0.0565	0.0640	0.0578	0.0304	0.0472	0.0224	0.0464	0.0251	0.0603	0.0715	0.0726	0.0290	0.1615	0.0222	0.0638	0.0523
8	0.0754	0.0729	0.0757	0.0764	0.0481	0.0517	0.0204	0.1037	0.0561	0.0739	0.0715	0.0629	0.0264	0.0361	0.1313	0.0782	0.0675
9	0.0133	0.0146	0.0202	0.0167	0.0340	0.0409	0.0189	0.0232	0.0229	0.0302	0.0185	0.0629	0.0374	0.0417	0.0222	0.0451	0.0390
10	0.0266	0.0565	0.0405	0.0646	0.0962	0.0472	0.0764	0.0207	0.0251	0.0523	0.0320	0.0162	0.0245	0.0273	0.0222	0.0261	0.0414
11	0.0843	0.0799	0.0757	0.0764	0.0304	0.0437	0.0289	0.0232	0.0324	0.0174	0.0715	0.0363	0.0216	0.0255	0.0203	0.0171	0.0442
12	0.0754	0.0652	0.0701	0.0500	0.0393	0.0517	0.0866	0.1037	0.1375	0.0523	0.0846	0.1089	0.0648	0.0323	0.0287	0.1194	0.0827
13	0.1066	0.0565	0.0405	0.0409	0.0340	0.0668	0.1118	0.0207	0.1122	0.0523	0.0846	0.1027	0.1449	0.0722	0.1313	0.1010	0.1170
14	0.0533	0.0461	0.0405	0.0409	0.0257	0.0409	0.1225	0.1037	0.0212	0.0675	0.0715	0.0889	0.1122	0.0273	0.0496	0.0782	0.0523

15	0.0754	0.0799	0.0757	0.0409	0.0481	0.0386	0.0866	0.0328	0.0324	0.0302	0.0554	0.0960	0.0245	0.0323	0.0287	0.0451	0.0478
16	0.0923	0.0863	0.0809	0.0817	0.1522	0.2004	0.1323	0.1037	0.0972	0.0905	0.0905	0.0960	0.0916	0.0722	0.1110	0.1106	0.1170

5.5.4: Determination of weighted preferences for each alternative with respect to each criterion

The criteria weights are obtained for each criterion, by computing the row wise mean of the normalized pairwise matrix. The criteria weights obtained for the normalized pairwise matrix as shown in the last column of table 5.33.

Table 5.33 Criteria Weights for all stretches

	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR	Criteria Weights
0	0.0377	0.0729	0.0757	0.0764	0.0393	0.0437	0.0224	0.0268	0.0281	0.0853	0.0452	0.0162	0.0324	0.0255	0.0351	0.0226	0.0478	0.0431
1	0.0169	0.0326	0.0640	0.0646	0.0304	0.0386	0.0204	0.0268	0.0251	0.0675	0.0185	0.0148	0.0324	0.0417	0.0351	0.0184	0.0442	0.0348
2	0.0142	0.0146	0.0286	0.0500	0.0278	0.0386	0.0189	0.0207	0.0212	0.0427	0.0226	0.0137	0.0264	0.0511	0.0351	0.0171	0.0414	0.0285
3	0.0142	0.0146	0.0165	0.0289	0.0257	0.0386	0.0204	0.0232	0.0212	0.0523	0.0143	0.0137	0.0374	0.0511	0.0351	0.0319	0.0414	0.0283
4	0.0653	0.0729	0.0701	0.0764	0.0681	0.0517	0.0289	0.1037	0.0794	0.0603	0.0226	0.0812	0.1122	0.1445	0.1313	0.0638	0.0523	0.0756
5	0.0997	0.0979	0.0858	0.0867	0.1522	0.1157	0.1323	0.1136	0.1255	0.0853	0.0784	0.0960	0.1449	0.1251	0.1404	0.1354	0.0675	0.1107
6	0.0843	0.0799	0.0757	0.0708	0.1179	0.0437	0.0500	0.1037	0.1375	0.0798	0.1466	0.0210	0.0374	0.0323	0.0203	0.0261	0.0442	0.0689
7	0.0653	0.0565	0.0640	0.0578	0.0304	0.0472	0.0224	0.0464	0.0251	0.0603	0.0715	0.0726	0.0290	0.1615	0.0222	0.0638	0.0523	0.0558
8	0.0754	0.0729	0.0757	0.0764	0.0481	0.0517	0.0204	0.1037	0.0561	0.0739	0.0715	0.0629	0.0264	0.0361	0.1313	0.0782	0.0675	0.0664
9	0.0133	0.0146	0.0202	0.0167	0.0340	0.0409	0.0189	0.0232	0.0229	0.0302	0.0185	0.0629	0.0374	0.0417	0.0222	0.0451	0.0390	0.0295
10	0.0266	0.0565	0.0405	0.0646	0.0962	0.0472	0.0764	0.0207	0.0251	0.0523	0.0320	0.0162	0.0245	0.0273	0.0222	0.0261	0.0414	0.0409
11	0.0843	0.0799	0.0757	0.0764	0.0304	0.0437	0.0289	0.0232	0.0324	0.0174	0.0715	0.0363	0.0216	0.0255	0.0203	0.0171	0.0442	0.0429
12	0.0754	0.0652	0.0701	0.0500	0.0393	0.0517	0.0866	0.1037	0.1375	0.0523	0.0846	0.1089	0.0648	0.0323	0.0287	0.1194	0.0827	0.0737

13	0.1066	0.0565	0.0405	0.0409	0.0340	0.0668	0.1118	0.0207	0.1122	0.0523	0.0846	0.1027	0.1449	0.0722	0.1313	0.1010	0.1170	0.0821
14	0.0533	0.0461	0.0405	0.0409	0.0257	0.0409	0.1225	0.1037	0.0212	0.0675	0.0715	0.0889	0.1122	0.0273	0.0496	0.0782	0.0523	0.0613
15	0.0754	0.0799	0.0757	0.0409	0.0481	0.0386	0.0866	0.0328	0.0324	0.0302	0.0554	0.0960	0.0245	0.0323	0.0287	0.0451	0.0478	0.0512
16	0.0923	0.0863	0.0809	0.0817	0.1522	0.2004	0.1323	0.1037	0.0972	0.0905	0.0905	0.0960	0.0916	0.0722	0.1110	0.1106	0.1170	0.1063

5.5.5: Determination of Consistency Ratio

The Consistency check matrix reflects the level of consistency in the results obtained from the combined pair wise matrices of experts. There are several steps the Pairwise matrix has to undergo to meet the checks' goal i.e., the values obtained should not contain inconsistency as mentioned in chapter 4. Below are the step-by-step approach to calculate the consistency check.

Step-1:

The Pairwise matrix columns are multiplied by their corresponding Criteria Weights to obtain the Consistency Check matrix. The table 5.34 shows the consistency check matrix.

Table 5.34 Table showing Consistency Check for all stretches

CRITERIA	**AC**	**BC**	**LC**	**TC**	**S**	**PH**	**RA**	**R**	**D**	**EC**	**P**	**MUD**	**POP**	**VOLT**	**FS**	**NTOWN**	**CBR**
AC	0.043	0.078	0.075	0.075	0.044	0.042	0.031	0.032	0.033	0.083	0.058	0.019	0.037	0.029	0.043	0.026	0.043
BC	0.019	0.035	0.064	0.063	0.034	0.037	0.028	0.032	0.030	0.066	0.024	0.018	0.037	0.047	0.043	0.021	0.040
LC	0.016	0.016	0.029	0.049	0.031	0.037	0.026	0.025	0.025	0.042	0.029	0.016	0.030	0.058	0.043	0.019	0.038
TC	0.016	0.016	0.016	0.028	0.029	0.037	0.028	0.028	0.025	0.051	0.018	0.016	0.043	0.058	0.043	0.036	0.038

S	0.075	0.078	0.070	0.075	0.076	0.050	0.040	0.125	0.094	0.059	0.029	0.096	0.128	0.164	0.162	0.072	0.048
PH	0.114	0.104	0.086	0.085	0.169	0.111	0.182	0.137	0.148	0.083	0.100	0.113	0.165	0.142	0.173	0.154	0.061
RA	0.096	0.085	0.075	0.069	0.131	0.042	0.069	0.125	0.163	0.078	0.188	0.025	0.043	0.037	0.025	0.030	0.040
R	0.075	0.060	0.064	0.057	0.034	0.045	0.031	0.056	0.030	0.059	0.092	0.086	0.033	0.184	0.027	0.072	0.048
D	0.086	0.078	0.075	0.075	0.053	0.050	0.028	0.125	0.066	0.072	0.092	0.074	0.030	0.041	0.162	0.089	0.061
EC	0.015	0.016	0.020	0.016	0.038	0.039	0.026	0.028	0.027	0.030	0.024	0.074	0.043	0.047	0.027	0.051	0.035
P	0.030	0.060	0.040	0.063	0.107	0.045	0.105	0.025	0.030	0.051	0.041	0.019	0.028	0.031	0.027	0.030	0.038
MUD	0.096	0.085	0.075	0.075	0.034	0.042	0.040	0.028	0.038	0.017	0.092	0.043	0.025	0.029	0.025	0.019	0.040
POP	0.086	0.070	0.070	0.049	0.044	0.050	0.119	0.125	0.163	0.051	0.108	0.129	0.074	0.037	0.035	0.135	0.075
VOLT	0.122	0.060	0.040	0.040	0.038	0.064	0.154	0.025	0.133	0.051	0.108	0.121	0.165	0.082	0.162	0.114	0.106
FS	0.061	0.049	0.040	0.040	0.029	0.039	0.169	0.125	0.025	0.066	0.092	0.105	0.128	0.031	0.061	0.089	0.048
NTOWN	0.086	0.085	0.075	0.040	0.053	0.037	0.119	0.039	0.038	0.030	0.071	0.113	0.028	0.037	0.035	0.051	0.043
CBR	0.106	0.092	0.081	0.080	0.169	0.192	0.182	0.125	0.115	0.089	0.116	0.113	0.104	0.082	0.137	0.125	0.106

STEP – 2:

The Weight sum values are obtained by adding the consistency check matrix in row wise. The last column of table 5.35 depicts the weight sum value calculated.

Table 5.35 Table showing weight sum values for all combinations

CRITERIA	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR	WSV
AC	0.043	0.078	0.075	0.075	0.044	0.042	0.031	0.032	0.033	0.083	0.058	0.019	0.037	0.029	0.043	0.026	0.043	0.792
BC	0.019	0.035	0.064	0.063	0.034	0.037	0.028	0.032	0.030	0.066	0.024	0.018	0.037	0.047	0.043	0.021	0.040	0.638
LC	0.016	0.016	0.029	0.049	0.031	0.037	0.026	0.025	0.025	0.042	0.029	0.016	0.030	0.058	0.043	0.019	0.038	0.528
TC	0.016	0.016	0.016	0.028	0.029	0.037	0.028	0.028	0.025	0.051	0.018	0.016	0.043	0.058	0.043	0.036	0.038	0.527
S	0.075	0.078	0.070	0.075	0.076	0.050	0.040	0.125	0.094	0.059	0.029	0.096	0.128	0.164	0.162	0.072	0.048	1.439
PH	0.114	0.104	0.086	0.085	0.169	0.111	0.182	0.137	0.148	0.083	0.100	0.113	0.165	0.142	0.173	0.154	0.061	2.129
RA	0.096	0.085	0.075	0.069	0.131	0.042	0.069	0.125	0.163	0.078	0.188	0.025	0.043	0.037	0.025	0.030	0.040	1.320
R	0.075	0.060	0.064	0.057	0.034	0.045	0.031	0.056	0.030	0.059	0.092	0.086	0.033	0.184	0.027	0.072	0.048	1.051
D	0.086	0.078	0.075	0.075	0.053	0.050	0.028	0.125	0.066	0.072	0.092	0.074	0.030	0.041	0.162	0.089	0.061	1.258
EC	0.015	0.016	0.020	0.016	0.038	0.039	0.026	0.028	0.027	0.030	0.024	0.074	0.043	0.047	0.027	0.051	0.035	0.557
P	0.030	0.060	0.040	0.063	0.107	0.045	0.105	0.025	0.030	0.051	0.041	0.019	0.028	0.031	0.027	0.030	0.038	0.771
MUD	0.096	0.085	0.075	0.075	0.034	0.042	0.040	0.028	0.038	0.017	0.092	0.043	0.025	0.029	0.025	0.019	0.040	0.803
POP	0.086	0.070	0.070	0.049	0.044	0.050	0.119	0.125	0.163	0.051	0.108	0.129	0.074	0.037	0.035	0.135	0.075	1.419
VOLT	0.122	0.060	0.040	0.040	0.038	0.064	0.154	0.025	0.133	0.051	0.108	0.121	0.165	0.082	0.162	0.114	0.106	1.587

FS	0.061	0.049	0.040	0.040	0.029	0.039	0.169	0.125	0.025	0.066	0.092	0.105	0.128	0.031	0.061	0.089	0.048	1.196
NTOWN	0.086	0.085	0.075	0.040	0.053	0.037	0.119	0.039	0.038	0.030	0.071	0.113	0.028	0.037	0.035	0.051	0.043	0.983
CBR	0.106	0.092	0.081	0.080	0.169	0.192	0.182	0.125	0.115	0.089	0.116	0.113	0.104	0.082	0.137	0.125	0.106	2.014

STEP – 3:

The weight sum values are divided by criteria weights and are obtained as shown in the table 5.36.

Table 5.36 Table showing WSV/CW value

CRITERIA	AC	BC	LC	TC	S	PH	RA	R	D	EC	P	MUD	POP	VOLT	FS	NTOWN	CBR	WSV	Criteria Weights	WSV / CW
AC	0.043	0.078	0.075	0.075	0.044	0.042	0.031	0.032	0.033	0.083	0.058	0.019	0.037	0.029	0.043	0.026	0.043	0.792	0.043	18.357
BC	0.019	0.035	0.064	0.063	0.034	0.037	0.028	0.032	0.030	0.066	0.024	0.018	0.037	0.047	0.043	0.021	0.040	0.638	0.035	18.312
LC	0.016	0.016	0.029	0.049	0.031	0.037	0.026	0.025	0.025	0.042	0.029	0.016	0.030	0.058	0.043	0.019	0.038	0.528	0.029	18.534
TC	0.016	0.016	0.016	0.028	0.029	0.037	0.028	0.028	0.025	0.051	0.018	0.016	0.043	0.058	0.043	0.036	0.038	0.527	0.028	18.629
S	0.075	0.078	0.070	0.075	0.076	0.050	0.040	0.125	0.094	0.059	0.029	0.096	0.128	0.164	0.162	0.072	0.048	1.439	0.076	19.035
PH	0.114	0.104	0.086	0.085	0.169	0.111	0.182	0.137	0.148	0.083	0.100	0.113	0.165	0.142	0.173	0.154	0.061	2.129	0.111	19.223
RA	0.096	0.085	0.075	0.069	0.131	0.042	0.069	0.125	0.163	0.078	0.188	0.025	0.043	0.037	0.025	0.030	0.040	1.320	0.069	19.160
R	0.075	0.060	0.064	0.057	0.034	0.045	0.031	0.056	0.030	0.059	0.092	0.086	0.033	0.184	0.027	0.072	0.048	1.051	0.056	18.836
D	0.086	0.078	0.075	0.075	0.053	0.050	0.028	0.125	0.066	0.072	0.092	0.074	0.030	0.041	0.162	0.089	0.061	1.258	0.066	18.951
EC	0.015	0.016	0.020	0.016	0.038	0.039	0.026	0.028	0.027	0.030	0.024	0.074	0.043	0.047	0.027	0.051	0.035	0.557	0.030	18.862
P	0.030	0.060	0.040	0.063	0.107	0.045	0.105	0.025	0.030	0.051	0.041	0.019	0.028	0.031	0.027	0.030	0.038	0.771	0.041	18.836

MUD	0.096	0.085	0.075	0.075	0.034	0.042	0.040	0.028	0.038	0.017	0.092	0.043	0.025	0.029	0.025	0.019	0.040	0.803	0.043	18.734
POP	0.086	0.070	0.070	0.049	0.044	0.050	0.119	0.125	0.163	0.051	0.108	0.129	0.074	0.037	0.035	0.135	0.075	1.419	0.074	19.248
VOLT	0.122	0.060	0.040	0.040	0.038	0.064	0.154	0.025	0.133	0.051	0.108	0.121	0.165	0.082	0.162	0.114	0.106	1.587	0.082	19.322
FS	0.061	0.049	0.040	0.040	0.029	0.039	0.169	0.125	0.025	0.066	0.092	0.105	0.128	0.031	0.061	0.089	0.048	1.196	0.061	19.499
NTOWN	0.086	0.085	0.075	0.040	0.053	0.037	0.119	0.039	0.038	0.030	0.071	0.113	0.028	0.037	0.035	0.051	0.043	0.983	0.051	19.198
CBR	0.106	0.092	0.081	0.080	0.169	0.192	0.182	0.125	0.115	0.089	0.116	0.113	0.104	0.082	0.137	0.125	0.106	2.014	0.106	18.953

STEP – 4: Determining λ_{max}, Consistency Index (CI), Random Index (RI) and Consistency Ratio (CR)

- The λ_{max} is calculated by adding all the values of WSV divided by criteria weights. The λ_{max} obtained from the table is

  λmax: 18.919983800914327.

- From the λmax, the value of consistency index (CI) is obtained from the formula as mentioned in chapter 4. The consistency index obtained is

  CI: 0.11999898755714544

- Random Index (RI) are defined as standard, for number of criteria present. Since the criteria are combined by removing low, medium and high values in a normalized matrix, and the pairwise matrix also contains the same number of criterias n = 17, the random index observed is

  RI: 1.61; for n = 17

- The Consistency Ratio (CR) is obtained by dividing the consistency index by random index. If the consistency ratio is less than 0.1, then the values are said to be within consistency level. Else the values are taken as inconsistent, i.e., the values in Pair Wise matrix have some level of inconsistency in them. The CR value obtained is

  CR: 0.07453353264418972

Therefore, our pairwise matrix is consistent and hence we can continue to use the pairwise matrix for prioritization.

5.5.6 Normalisation of Raw Data collected from filed study

The normalized values are already calculated in the Fuzzy TOPSIS method in the table 5.5. These values are imported as mentioned in chapter 4.

The normalized values obtained are sub-divided into low, medium and high. These 3 sub-criteria are combined and were obtained as 17 main criteria. The combined normalized values are obtained as shown in table 5.37.

Table 5.37 Table showing Combined values for all stretches

ST NO.	A_	B_	L_	T_	PH	RA	R_	S_	E_	D_	P_	M_	PO	VO	FS	NT	CB
S1	73	115	117	157	41	178	0	20	157	172	45	36	26	10	54	20	64
S2	151	203	122	175	200	130	0	73	160	190	58	50	52	35	62	26	66
S3	230	77	80	111	158	115	0	20	92	107	45	95	69	51	48	0	57
S4	78	168	125	180	216	168	0	140	185	160	70	123	69	32	100	33	44
S5	190	91	103	116	200	153	255	176	275	242	95	95	45	73	86	6	21
S6	183	115	154	151	158	145	0	246	146	175	25	85	65	76	50	53	55
S7	151	78	140	167	175	109	0	250	207	217	91	55	68	76	76	76	75
S8	113	91	148	133	158	128	0	123	196	197	250	134	68	100	44	73	44
S9	131	111	137	112	200	116	0	53	142	172	108	0	65	94	70	63	57
S10	53	227	122	94	225	190	100	20	157	210	183	191	71	11	28	63	38
S11	0	34	59	121	66	33	64	0	0	45	58	106	66	76	66	76	66
S12	58	218	200	200	191	221	180	100	67	175	91	237	70	96	60	60	0
S13	78	271	156	181	100	150	25	33	53	32	45	85	70	77	40	73	51
S14	0	58	104	136	50	214	45	100	14	135	25	235	22	43	48	66	21
S15	166	206	216	108	150	179	102	53	82	92	45	70	53	18	48	0	14
S16	0	0	0	0	0	0	40	0	0	0	0	0	5	16	64	6	25
S17	0	0	0	0	16	0	20	0	0	0	0	0	8	16	32	53	28
S18	0	0	0	0	25	0	11	0	0	12	37	0	100	17	64	66	21
S19	0	0	0	0	0	0	20	0	0	0	0	0	15	10	80	80	23
S20	0	0	0	0	0	4	60	0	0	0	50	0	58	6	30	36	15
S21	0	0	0	0	0	4	0	0	0	40	12	0	19	7	66	86	29

5.5.8 Assigning Ranks to the criteria

The Normalized values obtained are multiplied with satisfied Criteria Weights as a dot product. The obtained values are the Priority Index values as mentioned in chapter 4. According to these indices, the ranks are assigned such that, the greatest value is assigned with the least rank 1.

Table 5.38 Assigned Rankings for all stretches

STRETCHES	Priority Index	Rank
S1	71.60726	14
S2	95.18698	8
S3	77.26346	12
S4	105.9539	6
S5	127.5	1
S6	104.4554	7
S7	111.1836	3
S8	108.1342	5
S9	92.51033	9
S10	109.7881	4
S11	53.64031	15
S12	121.6232	2
S13	83.13513	11
S14	74.70059	13
S15	87.50046	10
S16	11.32555	21
S17	12.14108	20
S18	22.81774	16
S19	14.74913	19
S20	16.66704	17
S21	15.62095	18

5.6: PRIORITIZATION OF ROADS USING CONCORDANCE METHOD

5.6.1 Data Collection

The raw data are obtained as shown in table 5.7. These obtained raw data are combined to form 17 main criteria as shown in table 5.39. The raw data is combined in order to match the number of weightages.

Table 5.39 Combined Raw Data for all stretches

STRETCH NO.	A_	B_	L_	T_	PH	RA	R_	S_	E_	D_	P_	M_	PO	VO	FS	NT	CB
S1	3	8	15	15	2	24	0	1	7	9	2	4	9000	98	135	1.2	19.95134
S2	6	14	17	15	6	20	0	3	8	8	3	8	18057	332	155	1.1	18.49148
S3	9	5	12	11	4	21	0	1	5	5	2	11	24137	484	120	1.5	23.35766
S4	3	11	17	15	7	26	0	5	9	8	4	14	24194	311	250	1	30.65693
S5	7	6	15	12	6	22	18	6	12	11	3	11	15557	696	215	1.4	43.309
S6	8	8	21	11	4	23	0	8	7	8	2	10	22598	724	125	0.7	24.81752
S7	6	7	19	12	5	21	0	9	8	9	4	7	23598	724	190	0.35	13.6253
S8	5	6	20	21	4	23	0	4	9	11	9	14	23598	943	110	0.4	30.65693
S9	5	8	18	10	6	23	0	2	6	9	4	0	22598	887	175	0.55	23.35766
S10	2	14	19	16	8	27	8	1	8	12	10	23	24637	112	70	0.55	34.06326
S11	0	3	10	12	4	7	4	0	0	3	3	14	23137	720	165	0.35	18.49148
S12	2	14	29	32	6	34	22	5	4	11	4	26	24437	912	150	0.6	55.47445
S13	3	18	22	28	5	31	2	1	3	2	2	10	24437	728	100	0.4	26.76399
S14	0	5	17	14	3	31	3	3	1	9	2	24	7880	412	120	0.5	43.79562
S15	6	13	24	13	7	30	8	2	5	5	2	7	18437	172	120	1.5	47.20195
S16	0	0	0	0	0	0	2	0	0	0	0	0	2010	153	160	1.4	41.36253
S17	0	0	0	0	1	0	1	0	0	0	0	0	2800	155	80	0.7	39.90268
S18	0	0	0	0	1	0	2	0	0	1	3	0	34567	166	160	0.5	43.79562
S19	0	0	0	0	0	0	1	0	0	0	0	0	5330	97	200	0.3	42.33577
S20	0	0	0	0	0	1	3	0	0	0	1	0	20154	63	75	0.95	46.71533
S21	0	0	0	0	0	1	0	0	0	2	1	0	6780	67	165	0.2	38.92944

4.5.2 Determination of Maximum and minimum value and mathematical expression:

The maximum values, minimum values and the difference between maximum and minimum values for each criterion are obtained from the raw data as below:

Max values: [9.0, 18.0, 29.0, 32.0, 8.0, 34.0, 22.0, 9.0, 12.0, 12.0, 10.0, 26.0, 34567.0, 943.0, 250.0, 1.5, 55.47445255]

Min values: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2010.0, 63.0, 70.0, 0.2, 13.62530414]

Max values - Min values: [9.0, 18.0, 29.0, 32.0, 8.0, 34.0, 22.0, 9.0, 12.0, 12.0, 10.0, 26.0, 32557.0, 880.0, 180.0, 1.3, 41.84914841]

5.6.3 Normalisation of the obtained data Raw Data by Linear Transformation:

The raw data is normalized by transforming using linear transformation ranging between the scale 0 to 1. The data is normalized by using the formula mentioned in chapter 4. The transformed data calculated is as shown in the table 5.40.

5.6.4 Importing weightages from AHP:

The weightages for each criterion are taken from AHP method as mentioned in chapter 4. The weightages of the criteria are as in table 5.41.

S.NO	Weightages
0	0.043126
1	0.03482
2	0.028515
3	0.028265
4	0.075574
5	0.110731
6	0.068883

7	0.055786
8	0.06638
9	0.029513
10	0.04093
11	0.042873
12	0.073718
13	0.082116
14	0.061313

Table 5.41 Weightages imported from AHP

Table 5.40: Table showing Raw data after linear transformation

ST NO.	A_	B_	L_	T_	PH	RA	R_	S_	E_	D_	P_	M_	PO	VO	FS	NT	CB
S1	0.333	0.444	0.517	0.469	0.250	0.706	0.000	0.111	0.583	0.750	0.200	0.154	0.215	0.040	0.361	0.769	0.151
S2	0.667	0.778	0.586	0.469	0.750	0.588	0.000	0.333	0.667	0.667	0.300	0.308	0.493	0.306	0.472	0.692	0.116
S3	1.000	0.278	0.414	0.344	0.500	0.618	0.000	0.111	0.417	0.417	0.200	0.423	0.680	0.478	0.278	1.000	0.233
S4	0.333	0.611	0.586	0.469	0.875	0.765	0.000	0.556	0.750	0.667	0.400	0.538	0.681	0.282	1.000	0.615	0.407
S5	0.778	0.333	0.517	0.375	0.750	0.647	0.818	0.667	1.000	0.917	0.300	0.423	0.416	0.719	0.806	0.923	0.709
S6	0.889	0.444	0.724	0.344	0.500	0.676	0.000	0.889	0.583	0.667	0.200	0.385	0.632	0.751	0.306	0.385	0.267
S7	0.667	0.389	0.655	0.375	0.625	0.618	0.000	1.000	0.667	0.750	0.400	0.269	0.663	0.751	0.667	0.115	0.000
S8	0.556	0.333	0.690	0.656	0.500	0.676	0.000	0.444	0.750	0.917	0.900	0.538	0.663	1.000	0.222	0.154	0.407
S9	0.556	0.444	0.621	0.313	0.750	0.676	0.000	0.222	0.500	0.750	0.400	0.000	0.632	0.936	0.583	0.269	0.233
S10	0.222	0.778	0.655	0.500	1.000	0.794	0.364	0.111	0.667	1.000	1.000	0.885	0.695	0.056	0.000	0.269	0.488
S11	0.000	0.167	0.345	0.375	0.500	0.206	0.182	0.000	0.000	0.250	0.300	0.538	0.649	0.747	0.528	0.115	0.116
S12	0.222	0.778	1.000	1.000	0.750	1.000	1.000	0.556	0.333	0.917	0.400	1.000	0.689	0.965	0.444	0.308	1.000
S13	0.333	1.000	0.759	0.875	0.625	0.912	0.091	0.111	0.250	0.167	0.200	0.385	0.689	0.756	0.167	0.154	0.314
S14	0.000	0.278	0.586	0.438	0.375	0.912	0.136	0.333	0.083	0.750	0.200	0.923	0.180	0.397	0.278	0.231	0.721
S15	0.667	0.722	0.828	0.406	0.875	0.882	0.364	0.222	0.417	0.417	0.200	0.269	0.505	0.124	0.278	1.000	0.802
S16	0.000	0.000	0.000	0.000	0.000	0.000	0.091	0.000	0.000	0.000	0.000	0.000	0.000	0.102	0.500	0.923	0.663
S17	0.000	0.000	0.000	0.000	0.125	0.000	0.045	0.000	0.000	0.000	0.000	0.000	0.024	0.105	0.056	0.385	0.628
S18	0.000	0.000	0.000	0.000	0.125	0.000	0.091	0.000	0.000	0.083	0.300	0.000	1.000	0.117	0.500	0.231	0.721
S19	0.000	0.000	0.000	0.000	0.000	0.000	0.045	0.000	0.000	0.000	0.000	0.000	0.102	0.039	0.722	0.077	0.686
S20	0.000	0.000	0.000	0.000	0.000	0.029	0.136	0.000	0.000	0.000	0.100	0.000	0.557	0.000	0.028	0.577	0.791
S21	0.000	0.000	0.000	0.000	0.000	0.029	0.000	0.000	0.000	0.167	0.100	0.000	0.147	0.005	0.528	0.000	0.605

5.6.5 Determination of Relative Comparison of Stretches and Row wise summation

In this step, the stretches are relatively compared with each other in a loop and the weightages are assigned such that, if the first value > second value, the weightage is assigned, else 0 is assigned as mentioned in. After assigning the weightages, the values of each row are added up and sum values are obtained. The weightages are assigned to stretches and summation is performed as shown in the table 5.42.

mi-mj	A_	B_	L_	T_	PH	RA	R_	S_	E_	D_	P_	M_	PO	VO	FS	NT	CB	SUM
m1-m1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m1-m2	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0000	0.0000	0.0000	0.0295	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2977
m1-m3	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0000	0.0000	0.0664	0.0295	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.3595
m1-m4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0295	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.0000	0.0807
m1-m5	0.0000	0.0348	0.0000	0.0283	0.0000	0.1107	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1738
m1-m6	0.0000	0.0000	0.0000	0.0283	0.0000	0.1107	0.0000	0.0000	0.0000	0.0295	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.2810
m1-m7	0.0000	0.0348	0.0000	0.0283	0.0000	0.1107	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.3313
m1-m8	0.0000	0.0348	0.0000	0.0000	0.0000	0.1107	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.2581
m1-m9	0.0000	0.0000	0.0000	0.0283	0.0000	0.1107	0.0000	0.0000	0.0664	0.0000	0.0000	0.0429	0.0000	0.0000	0.0000	0.0512	0.0000	0.2994
m1-m10	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.1556
m1-m11	0.0431	0.0348	0.0285	0.0283	0.0000	0.1107	0.0000	0.0558	0.0664	0.0295	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.5546
m1-m12	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0664	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.0000	0.1607
m1-m13	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0664	0.0295	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.2084
m1-m14	0.0431	0.0348	0.0000	0.0283	0.0000	0.0000	0.0000	0.0000	0.0664	0.0000	0.0000	0.0000	0.0737	0.0000	0.0613	0.0512	0.0000	0.3588
m1-m15	0.0000	0.0000	0.0000	0.0283	0.0000	0.0000	0.0000	0.0000	0.0664	0.0295	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.1855
m1-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0000	0.0000	0.6302
m1-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0000	0.0613	0.0512	0.0000	0.7427
m1-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0000	0.0429	0.0000	0.0000	0.0000	0.0512	0.0000	0.5668
m1-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m1-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0613	0.0512	0.0000	0.7511
m1-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m2-m1	0.0431	0.0348	0.0285	0.0000	0.0756	0.0000	0.0000	0.0558	0.0664	0.0000	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.6052
m2-m2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m2-m3	0.0000	0.0348	0.0285	0.0283	0.0756	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.4211
m2-m4	0.0431	0.0348	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0000	0.0512	0.0000	0.2113

m2-m5	0.0000	0.0348	0.0285	0.0283	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0000	0.0000	0.1653
m2-m6	0.0000	0.0348	0.0000	0.0283	0.0756	0.0000	0.0000	0.0000	0.0664	0.0000	0.0409	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.3585
m2-m7	0.0000	0.0348	0.0000	0.0283	0.0756	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0429	0.0000	0.0000	0.0000	0.0512	0.1063	0.3390
m2-m8	0.0431	0.0348	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.2660
m2-m9	0.0431	0.0348	0.0000	0.0283	0.0000	0.0000	0.0000	0.0558	0.0664	0.0000	0.0000	0.0429	0.0000	0.0000	0.0000	0.0512	0.0000	0.3224
m2-m10	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0512	0.0000	0.2935
m2-m11	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.0000	0.5239
m2-m12	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0664	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.2220
m2-m13	0.0431	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.4238
m2-m14	0.0431	0.0348	0.0000	0.0283	0.0756	0.0000	0.0000	0.0000	0.0664	0.0000	0.0409	0.0000	0.0737	0.0000	0.0613	0.0512	0.0000	0.4753
m2-m15	0.0000	0.0348	0.0000	0.0283	0.0000	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0613	0.0000	0.0000	0.4420
m2-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.0000	0.7123
m2-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.0000	0.8249
m2-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0000	0.0429	0.0000	0.0821	0.0000	0.0512	0.0000	0.6489
m2-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m2-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0613	0.0512	0.0000	0.7511
m2-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m3-m1	0.0431	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.4749
m3-m2	0.0431	0.0000	0.0000	0.0000	0.0000	0.1107	0.0000	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.5100
m3-m3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m3-m4	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0000	0.0512	0.0000	0.1764
m3-m5	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0512	0.0000	0.1680
m3-m6	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0000	0.0000	0.0512	0.0000	0.2109
m3-m7	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0000	0.0000	0.0512	0.1063	0.3172
m3-m8	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0613	0.0512	0.0000	0.2294
m3-m9	0.0431	0.0000	0.0000	0.0283	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0000	0.0000	0.0512	0.0000	0.2392

m3-m10	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0512	0.0000	0.2377
m3-m11	0.0431	0.0348	0.0285	0.0000	0.0000	0.1107	0.0000	0.0558	0.0664	0.0295	0.0000	0.0000	0.0737	0.0000	0.0000	0.0512	0.1063	0.6000
m3-m12	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0664	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.0000	0.1607
m3-m13	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0664	0.0295	0.0000	0.0429	0.0000	0.0000	0.0613	0.0512	0.0000	0.2944
m3-m14	0.0431	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0664	0.0000	0.0000	0.0000	0.0737	0.0821	0.0000	0.0512	0.0000	0.3921
m3-m15	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0000	0.0000	0.0000	0.2418
m3-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m3-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.0000	0.8249
m3-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0000	0.0429	0.0000	0.0821	0.0000	0.0512	0.0000	0.6489
m3-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m3-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.0000	0.8249
m3-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m4-m1	0.0000	0.0348	0.0285	0.0000	0.0756	0.1107	0.0000	0.0558	0.0664	0.0000	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.1063	0.7790
m4-m2	0.0000	0.0000	0.0000	0.0000	0.0756	0.1107	0.0000	0.0558	0.0664	0.0000	0.0409	0.0429	0.0737	0.0000	0.0613	0.0000	0.1063	0.6336
m4-m3	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0000	0.0613	0.0000	0.1063	0.7547
m4-m4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m4-m5	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0000	0.0000	0.0000	0.0409	0.0429	0.0737	0.0000	0.0613	0.0000	0.0000	0.4967
m4-m6	0.0000	0.0348	0.0000	0.0283	0.0756	0.1107	0.0000	0.0000	0.0664	0.0000	0.0409	0.0429	0.0737	0.0000	0.0613	0.0512	0.1063	0.6921
m4-m7	0.0000	0.0348	0.0000	0.0283	0.0756	0.1107	0.0000	0.0000	0.0664	0.0000	0.0000	0.0429	0.0737	0.0000	0.0613	0.0512	0.1063	0.6511
m4-m8	0.0000	0.0348	0.0000	0.0000	0.0756	0.1107	0.0000	0.0558	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0613	0.0512	0.0000	0.4631
m4-m9	0.0000	0.0348	0.0000	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0000	0.0000	0.0429	0.0737	0.0000	0.0613	0.0512	0.1063	0.7069
m4-m10	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0664	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0512	0.0000	0.3599
m4-m11	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0000	0.0613	0.0512	0.1063	0.8061
m4-m12	0.0431	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0664	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.2976
m4-m13	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0000	0.0613	0.0512	0.1063	0.5298
m4-m14	0.0431	0.0348	0.0000	0.0283	0.0756	0.0000	0.0000	0.0558	0.0664	0.0000	0.0409	0.0000	0.0737	0.0000	0.0613	0.0512	0.0000	0.5311

m4-m15	0.0000	0.0000	0.0000	0.0283	0.0000	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.4809
m4-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.7737
m4-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.0000	0.8249
m4-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0613	0.0512	0.0000	0.7511
m4-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.0000	0.8249
m4-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.0000	0.8249
m4-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.0000	0.8249
m5-m1	0.0431	0.0000	0.0000	0.0000	0.0756	0.0000	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.1063	0.7977
m5-m2	0.0431	0.0000	0.0000	0.0000	0.0000	0.1107	0.0689	0.0558	0.0664	0.0295	0.0000	0.0429	0.0000	0.0821	0.0613	0.0512	0.1063	0.7182
m5-m3	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0000	0.0000	0.0821	0.0613	0.0000	0.1063	0.7891
m5-m4	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0558	0.0664	0.0295	0.0000	0.0000	0.0000	0.0821	0.0000	0.0512	0.1063	0.5033
m5-m5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m5-m6	0.0000	0.0000	0.0000	0.0283	0.0756	0.0000	0.0689	0.0000	0.0664	0.0295	0.0409	0.0429	0.0000	0.0000	0.0613	0.0512	0.1063	0.5712
m5-m7	0.0431	0.0000	0.0000	0.0000	0.0756	0.1107	0.0689	0.0000	0.0664	0.0295	0.0000	0.0429	0.0000	0.0000	0.0613	0.0512	0.1063	0.6559
m5-m8	0.0431	0.0000	0.0000	0.0000	0.0756	0.0000	0.0689	0.0558	0.0664	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.1063	0.5285
m5-m9	0.0431	0.0000	0.0000	0.0283	0.0000	0.0000	0.0689	0.0558	0.0664	0.0295	0.0000	0.0429	0.0000	0.0000	0.0613	0.0512	0.1063	0.5536
m5-m10	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0558	0.0664	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0512	0.1063	0.5351
m5-m11	0.0431	0.0348	0.0285	0.0000	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.1063	0.7321
m5-m12	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0664	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.2778
m5-m13	0.0431	0.0000	0.0000	0.0000	0.0756	0.0000	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0000	0.0613	0.0512	0.1063	0.6418
m5-m14	0.0431	0.0348	0.0000	0.0000	0.0756	0.0000	0.0689	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0512	0.0000	0.6834
m5-m15	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0613	0.0000	0.0000	0.4909
m5-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.1063	0.9488
m5-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.1063	1.0000
m5-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0000	0.0429	0.0000	0.0821	0.0613	0.0512	0.0000	0.7791
m5-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.1063	1.0000

m5-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0613	0.0512	0.0000	0.8200
m5-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.1063	1.0000
m6-m1	0.0431	0.0000	0.0285	0.0000	0.0756	0.0000	0.0000	0.0558	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.5080
m6-m2	0.0431	0.0000	0.0285	0.0000	0.0000	0.1107	0.0000	0.0558	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.5431
m6-m3	0.0000	0.0348	0.0285	0.0000	0.0000	0.1107	0.0000	0.0558	0.0664	0.0295	0.0000	0.0000	0.0000	0.0821	0.0613	0.0000	0.1063	0.5754
m6-m4	0.0431	0.0000	0.0285	0.0000	0.0000	0.0000	0.0000	0.0558	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0000	0.0000	0.0000	0.2095
m6-m5	0.0431	0.0348	0.0285	0.0000	0.0000	0.1107	0.0000	0.0558	0.0000	0.0000	0.0000	0.0000	0.0737	0.0821	0.0000	0.0000	0.0000	0.4288
m6-m6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m6-m7	0.0431	0.0348	0.0285	0.0000	0.0000	0.1107	0.0000	0.0000	0.0000	0.0000	0.0000	0.0429	0.0000	0.0000	0.0000	0.0512	0.1063	0.4175
m6-m8	0.0431	0.0348	0.0285	0.0000	0.0000	0.0000	0.0000	0.0558	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.2748
m6-m9	0.0431	0.0000	0.0285	0.0283	0.0000	0.0000	0.0000	0.0558	0.0664	0.0000	0.0000	0.0429	0.0000	0.0000	0.0000	0.0512	0.1063	0.4224
m6-m10	0.0431	0.0000	0.0285	0.0000	0.0000	0.0000	0.0000	0.0558	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0512	0.0000	0.3220
m6-m11	0.0431	0.0348	0.0285	0.0000	0.0000	0.1107	0.0000	0.0558	0.0664	0.0295	0.0000	0.0000	0.0000	0.0821	0.0000	0.0512	0.1063	0.6084
m6-m12	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0664	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.0000	0.2165
m6-m13	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0664	0.0295	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.3073
m6-m14	0.0431	0.0348	0.0285	0.0000	0.0756	0.0000	0.0000	0.0558	0.0664	0.0000	0.0000	0.0000	0.0737	0.0821	0.0613	0.0512	0.0000	0.5725
m6-m15	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0664	0.0295	0.0000	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.4548
m6-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.0000	0.7123
m6-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.7737
m6-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0000	0.0429	0.0000	0.0821	0.0000	0.0512	0.0000	0.6489
m6-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m6-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.7737
m6-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m7-m1	0.0431	0.0000	0.0285	0.0000	0.0756	0.0000	0.0000	0.0558	0.0664	0.0000	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.5703
m7-m2	0.0000	0.0000	0.0285	0.0000	0.0000	0.1107	0.0000	0.0558	0.0000	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.0000	0.4826
m7-m3	0.0000	0.0348	0.0285	0.0283	0.0756	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0000	0.0821	0.0613	0.0000	0.0000	0.5032

m7-m4	0.0431	0.0000	0.0285	0.0000	0.0000	0.0000	0.0000	0.0558	0.0000	0.0295	0.0000	0.0000	0.0000	0.0821	0.0000	0.0000	0.0000	0.2391
m7-m5	0.0000	0.0348	0.0285	0.0000	0.0000	0.0000	0.0000	0.0558	0.0000	0.0000	0.0409	0.0000	0.0737	0.0821	0.0000	0.0000	0.0000	0.3159
m7-m6	0.0000	0.0000	0.0000	0.0283	0.0756	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0000	0.0613	0.0000	0.0000	0.4315
m7-m7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m7-m8	0.0431	0.0348	0.0000	0.0000	0.0756	0.0000	0.0000	0.0558	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.2706
m7-m9	0.0431	0.0000	0.0285	0.0283	0.0000	0.0000	0.0000	0.0558	0.0664	0.0000	0.0000	0.0429	0.0737	0.0000	0.0613	0.0000	0.0000	0.4000
m7-m10	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0000	0.0000	0.2423
m7-m11	0.0431	0.0348	0.0285	0.0000	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.0000	0.7025
m7-m12	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0664	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.2266
m7-m13	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.2970
m7-m14	0.0431	0.0348	0.0285	0.0000	0.0756	0.0000	0.0000	0.0558	0.0664	0.0000	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.0000	0.5623
m7-m15	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.0000	0.4098
m7-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.7737
m7-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.7737
m7-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0613	0.0000	0.0000	0.6999
m7-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m7-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.7737
m7-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.0000	0.8249
m8-m1	0.0431	0.0000	0.0285	0.0283	0.0756	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.6731
m8-m2	0.0000	0.0000	0.0285	0.0283	0.0000	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.6651
m8-m3	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0000	0.0000	0.1063	0.6262
m8-m4	0.0431	0.0000	0.0285	0.0283	0.0000	0.0000	0.0000	0.0000	0.0000	0.0295	0.0409	0.0000	0.0000	0.0821	0.0000	0.0000	0.0000	0.2525
m8-m5	0.0000	0.0000	0.0285	0.0283	0.0000	0.1107	0.0000	0.0000	0.0000	0.0000	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.0000	0.4071
m8-m6	0.0000	0.0000	0.0000	0.0283	0.0000	0.0000	0.0000	0.0000	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.4701
m8-m7	0.0000	0.0000	0.0285	0.0283	0.0000	0.1107	0.0000	0.0000	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0000	0.0512	0.1063	0.5868
m8-m8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

m8-m9	0.0000	0.0000	0.0285	0.0283	0.0000	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.5544
m8-m10	0.0431	0.0000	0.0285	0.0283	0.0000	0.0000	0.0000	0.0558	0.0664	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0000	0.0000	0.3655
m8-m11	0.0431	0.0348	0.0285	0.0283	0.0000	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0821	0.0000	0.0512	0.1063	0.7514
m8-m12	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0664	0.0000	0.0409	0.0000	0.0000	0.0821	0.0000	0.0000	0.0000	0.2326
m8-m13	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0613	0.0000	0.1063	0.5283
m8-m14	0.0431	0.0348	0.0285	0.0283	0.0756	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0821	0.0000	0.0000	0.0000	0.5587
m8-m15	0.0000	0.0000	0.0000	0.0283	0.0000	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.0000	0.4196
m8-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.0000	0.7123
m8-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.7737
m8-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0000	0.0000	0.0000	0.6386
m8-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m8-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.7737
m8-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.7635
m9-m1	0.0431	0.0000	0.0285	0.0000	0.0756	0.0000	0.0000	0.0558	0.0000	0.0000	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.1063	0.5673
m9-m2	0.0000	0.0000	0.0285	0.0000	0.0000	0.1107	0.0000	0.0000	0.0000	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.1063	0.5331
m9-m3	0.0000	0.0348	0.0285	0.0000	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0000	0.0821	0.0613	0.0000	0.0000	0.5857
m9-m4	0.0431	0.0000	0.0285	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0295	0.0000	0.0000	0.0000	0.0821	0.0000	0.0000	0.0000	0.1833
m9-m5	0.0000	0.0348	0.0285	0.0000	0.0000	0.1107	0.0000	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0821	0.0000	0.0000	0.0000	0.3708
m9-m6	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0000	0.0295	0.0409	0.0000	0.0000	0.0821	0.0613	0.0000	0.0000	0.2894
m9-m7	0.0000	0.0348	0.0000	0.0000	0.0756	0.1107	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0000	0.0512	0.1063	0.4607
m9-m8	0.0000	0.0348	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.2229
m9-m9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m9-m10	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0558	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0000	0.0000	0.2423
m9-m11	0.0431	0.0348	0.0285	0.0000	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0000	0.0821	0.0613	0.0512	0.1063	0.7863
m9-m12	0.0431	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0664	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.1708
m9-m13	0.0431	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0000	0.0821	0.0613	0.0512	0.0000	0.5059

m9-m14	0.0431	0.0348	0.0285	0.0000	0.0756	0.0000	0.0000	0.0000	0.0664	0.0000	0.0409	0.0000	0.0737	0.0821	0.0613	0.0512	0.0000	0.5577
m9-m15	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0664	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.0000	0.3540
m9-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.0000	0.7308
m9-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.0000	0.7308
m9-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0000	0.0821	0.0613	0.0512	0.0000	0.7083
m9-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0821	0.0000	0.0512	0.0000	0.7207
m9-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.0000	0.7308
m9-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0512	0.0000	0.7820
m10-m1	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.7886
m10-m2	0.0000	0.0000	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0000	0.1063	0.6053
m10-m3	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0664	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0000	0.1063	0.7065
m10-m4	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0000	0.1063	0.6401
m10-m5	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0000	0.0000	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0000	0.0000	0.4649
m10-m6	0.0000	0.0348	0.0000	0.0283	0.0756	0.1107	0.0689	0.0000	0.0664	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0000	0.1063	0.6780
m10-m7	0.0000	0.0348	0.0000	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0512	0.1063	0.6628
m10-m8	0.0000	0.0348	0.0000	0.0000	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0512	0.1063	0.6345
m10-m9	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0664	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0000	0.1063	0.7065
m10-m10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m10-m11	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0512	0.1063	0.8566
m10-m12	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0664	0.0295	0.0409	0.0000	0.0737	0.0000	0.0000	0.0000	0.0000	0.2861
m10-m13	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0689	0.0000	0.0664	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0512	0.1063	0.5553
m10-m14	0.0431	0.0348	0.0285	0.0283	0.0756	0.0000	0.0689	0.0000	0.0664	0.0295	0.0409	0.0000	0.0737	0.0000	0.0000	0.0512	0.0000	0.5409
m10-m15	0.0000	0.0348	0.0000	0.0283	0.0756	0.0000	0.0000	0.0000	0.0664	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0000	0.0000	0.3921
m10-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0000	0.0000	0.6991
m10-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0000	0.0000	0.0000	0.0000	0.6991
m10-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0000	0.0000	0.0512	0.0000	0.6766

m10-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.8324
m10-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.0000	0.7812
m10-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.8324
m11-m1	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0689	0.0000	0.0000	0.0000	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.4454
m11-m2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.3289
m11-m3	0.0000	0.0000	0.0000	0.0283	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0409	0.0429	0.0000	0.0821	0.0613	0.0000	0.0000	0.3244
m11-m4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0000	0.0000	0.0000	0.1510
m11-m5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0000	0.0000	0.0000	0.1987
m11-m6	0.0000	0.0000	0.0000	0.0283	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0409	0.0429	0.0737	0.0000	0.0613	0.0000	0.0000	0.3160
m11-m7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0429	0.0000	0.0000	0.0000	0.0000	0.1063	0.2180
m11-m8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.1302
m11-m9	0.0000	0.0000	0.0000	0.0283	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0000	0.0000	0.0000	0.0000	0.2137
m11-m10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0000	0.0000	0.1434
m11-m11	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m11-m12	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.0613
m11-m13	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0295	0.0409	0.0429	0.0000	0.0000	0.0613	0.0000	0.0000	0.2435
m11-m14	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0689	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.0000	0.4025
m11-m15	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.3010
m11-m16	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.6773
m11-m17	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.6773
m11-m18	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0000	0.0429	0.0000	0.0821	0.0613	0.0000	0.0000	0.5626
m11-m19	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.6671
m11-m20	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.6773
m11-m21	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.6671
m12-m1	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0000	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.1063	0.8393
m12-m2	0.0000	0.0000	0.0285	0.0283	0.0000	0.1107	0.0689	0.0558	0.0000	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.6676

m12-m3	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0000	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.1063	0.8393
m12-m4	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0689	0.0000	0.0000	0.0295	0.0000	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.6057
m12-m5	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.6171
m12-m6	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.1063	0.7835
m12-m7	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0295	0.0000	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.7325
m12-m8	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0000	0.0000	0.0000	0.0429	0.0737	0.0000	0.0613	0.0512	0.1063	0.7379
m12-m9	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0689	0.0558	0.0000	0.0295	0.0000	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.7127
m12-m10	0.0000	0.0000	0.0285	0.0283	0.0000	0.1107	0.0689	0.0558	0.0000	0.0000	0.0000	0.0429	0.0000	0.0821	0.0613	0.0512	0.1063	0.6359
m12-m11	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.9387
m12-m12	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m12-m13	0.0000	0.0000	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0613	0.0512	0.1063	0.8483
m12-m14	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.1063	1.0000
m12-m15	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0689	0.0558	0.0000	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.1063	0.7637
m12-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.8875
m12-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.1063	0.9488
m12-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0000	0.0512	0.1063	0.8650
m12-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.9387
m12-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.1063	0.9488
m12-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.9387
m13-m1	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.6518
m13-m2	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.5762
m13-m3	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0821	0.0000	0.0000	0.1063	0.6089
m13-m4	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0821	0.0000	0.0000	0.0000	0.4270
m13-m5	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0821	0.0000	0.0000	0.0000	0.3582
m13-m6	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0821	0.0000	0.0000	0.1063	0.6089
m13-m7	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.6274

m13-m8	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0000	0.0000	0.4205
m13-m9	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0000	0.0000	0.0000	0.1063	0.4941
m13-m10	0.0431	0.0348	0.0285	0.0283	0.0000	0.1107	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0000	0.0000	0.3889
m13-m11	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0000	0.0000	0.0000	0.0737	0.0821	0.0000	0.0512	0.1063	0.7565
m13-m12	0.0431	0.0348	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0779
m13-m13	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m13-m14	0.0431	0.0348	0.0285	0.0283	0.0756	0.0000	0.0000	0.0000	0.0664	0.0000	0.0000	0.0000	0.0737	0.0821	0.0000	0.0000	0.0000	0.4325
m13-m15	0.0000	0.0348	0.0000	0.0283	0.0000	0.1107	0.0000	0.0000	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0000	0.0000	0.0000	0.3725
m13-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.0000	0.7123
m13-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.8425
m13-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0000	0.0429	0.0000	0.0821	0.0000	0.0000	0.0000	0.5977
m13-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.8324
m13-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.0000	0.7737
m13-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0000	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.0000	0.8029
m14-m1	0.0000	0.0000	0.0285	0.0000	0.0756	0.1107	0.0689	0.0558	0.0000	0.0000	0.0000	0.0429	0.0000	0.0821	0.0000	0.0000	0.1063	0.5707
m14-m2	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0689	0.0000	0.0000	0.0295	0.0000	0.0429	0.0000	0.0821	0.0000	0.0000	0.1063	0.4404
m14-m3	0.0000	0.0000	0.0285	0.0283	0.0000	0.1107	0.0689	0.0558	0.0000	0.0295	0.0000	0.0429	0.0000	0.0000	0.0000	0.0000	0.1063	0.4708
m14-m4	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0689	0.0000	0.0000	0.0295	0.0000	0.0429	0.0000	0.0821	0.0000	0.0000	0.1063	0.4404
m14-m5	0.0000	0.0000	0.0285	0.0283	0.0000	0.1107	0.0000	0.0000	0.0000	0.0000	0.0000	0.0429	0.0000	0.0000	0.0000	0.0000	0.1063	0.3166
m14-m6	0.0000	0.0000	0.0000	0.0283	0.0000	0.1107	0.0689	0.0000	0.0000	0.0295	0.0000	0.0429	0.0000	0.0000	0.0000	0.0000	0.1063	0.3865
m14-m7	0.0000	0.0000	0.0000	0.0283	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0429	0.0000	0.0000	0.0000	0.0512	0.1063	0.4082
m14-m8	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0429	0.0000	0.0000	0.0613	0.0512	0.1063	0.4413
m14-m9	0.0000	0.0000	0.0000	0.0283	0.0000	0.1107	0.0689	0.0558	0.0000	0.0000	0.0000	0.0429	0.0000	0.0000	0.0000	0.0000	0.1063	0.4128
m14-m10	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0000	0.0558	0.0000	0.0000	0.0000	0.0429	0.0000	0.0821	0.0613	0.0000	0.1063	0.4591
m14-m11	0.0000	0.0348	0.0285	0.0283	0.0000	0.1107	0.0000	0.0558	0.0664	0.0295	0.0000	0.0429	0.0000	0.0000	0.0000	0.0512	0.1063	0.5543
m14-m12	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

m14-m13	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0558	0.0000	0.0295	0.0000	0.0429	0.0000	0.0000	0.0613	0.0512	0.1063	0.4158
m14-m14	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m14-m15	0.0000	0.0000	0.0000	0.0283	0.0000	0.1107	0.0000	0.0558	0.0000	0.0295	0.0000	0.0429	0.0000	0.0821	0.0000	0.0000	0.0000	0.3493
m14-m16	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0000	0.1063	0.8444
m14-m17	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0000	0.1063	0.9057
m14-m18	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0000	0.0429	0.0000	0.0821	0.0000	0.0000	0.0000	0.6235
m14-m19	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.8956
m14-m20	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0613	0.0000	0.0000	0.6568
m14-m21	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.8956
m15-m1	0.0431	0.0348	0.0285	0.0000	0.0756	0.1107	0.0689	0.0558	0.0000	0.0000	0.0000	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.7736
m15-m2	0.0000	0.0000	0.0285	0.0000	0.0756	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0512	0.1063	0.5149
m15-m3	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.5088
m15-m4	0.0431	0.0348	0.0285	0.0000	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.4435
m15-m5	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0512	0.1063	0.5091
m15-m6	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.5042
m15-m7	0.0000	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.5042
m15-m8	0.0431	0.0348	0.0285	0.0000	0.0756	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.1063	0.5804
m15-m9	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0429	0.0000	0.0000	0.0000	0.0512	0.1063	0.5902
m15-m10	0.0431	0.0000	0.0285	0.0000	0.0000	0.1107	0.0000	0.0558	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0512	0.1063	0.5390
m15-m11	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.6990
m15-m12	0.0431	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0664	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.0000	0.2363
m15-m13	0.0431	0.0000	0.0285	0.0000	0.0756	0.0000	0.0689	0.0558	0.0664	0.0295	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.1063	0.5865
m15-m14	0.0431	0.0348	0.0285	0.0000	0.0756	0.0000	0.0689	0.0000	0.0664	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0512	0.1063	0.5485
m15-m15	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m15-m16	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.9387
m15-m17	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0613	0.0512	0.1063	1.0000

m15-m18	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0000	0.0429	0.0000	0.0821	0.0000	0.0512	0.1063	0.8240
m15-m19	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.9387
m15-m20	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0000	0.0821	0.0613	0.0512	0.1063	0.9263
m15-m21	0.0431	0.0348	0.0285	0.0283	0.0756	0.1107	0.0689	0.0558	0.0664	0.0295	0.0409	0.0429	0.0737	0.0821	0.0000	0.0512	0.1063	0.9387
m16-m1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0512	0.1063	0.3698
m16-m2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.1063	0.2877
m16-m3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.2365
m16-m4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2263
m16-m5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m16-m6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.1063	0.2877
m16-m7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2263
m16-m8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.1063	0.2877
m16-m9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2263
m16-m10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0512	0.1063	0.3009
m16-m11	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.1575
m16-m12	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.1125
m16-m13	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.1063	0.2188
m16-m14	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.0000	0.1125
m16-m15	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.0613
m16-m16	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m16-m17	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.1063	0.2877
m16-m18	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.0000	0.0512
m16-m19	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0000	0.0512	0.0000	0.2022
m16-m20	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0512	0.0000	0.1946
m16-m21	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0000	0.0512	0.1063	0.3085
m17-m1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0000	0.0000	0.1063	0.2573

m17-m2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.1751
m17-m3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.1751
m17-m4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.1751
m17-m5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m17-m6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.1751
m17-m7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2263
m17-m8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2263
m17-m9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2263
m17-m10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0512	0.1063	0.3009
m17-m11	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.1575
m17-m12	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.0000	0.0512
m17-m13	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.1575
m17-m14	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.0000	0.0512
m17-m15	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m17-m16	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0821	0.0000	0.0000	0.0000	0.2314
m17-m17	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m17-m18	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.0000	0.0512
m17-m19	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0000	0.0512	0.0000	0.2089
m17-m20	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0000	0.0000	0.2190
m17-m21	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0000	0.0512	0.1063	0.3840
m18-m1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.1063	0.4332
m18-m2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0613	0.0000	0.1063	0.3102
m18-m3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0000	0.0613	0.0000	0.1063	0.3511
m18-m4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0000	0.1063	0.2489
m18-m5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0000	0.1063	0.1800
m18-m6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0000	0.0613	0.0000	0.1063	0.3511

m18-m7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0512	0.1063	0.3001
m18-m8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0613	0.0512	0.1063	0.3614
m18-m9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0000	0.1063	0.2489
m18-m10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0821	0.0613	0.0000	0.1063	0.3234
m18-m11	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0512	0.1063	0.2312
m18-m12	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0613	0.0000	0.0000	0.1350
m18-m13	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0000	0.0613	0.0512	0.1063	0.3334
m18-m14	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0000	0.0613	0.0000	0.0000	0.1760
m18-m15	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0000	0.0613	0.0000	0.0000	0.1760
m18-m16	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0000	0.0295	0.0409	0.0000	0.0737	0.0821	0.0000	0.0000	0.1063	0.4081
m18-m17	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.1063	0.4627
m18-m18	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m18-m19	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0689	0.0000	0.0000	0.0295	0.0409	0.0000	0.0737	0.0821	0.0000	0.0512	0.1063	0.5282
m18-m20	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0000	0.0000	0.0000	0.0295	0.0409	0.0000	0.0737	0.0821	0.0613	0.0000	0.0000	0.3632
m18-m21	0.0000	0.0000	0.0000	0.0000	0.0756	0.0000	0.0689	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0821	0.0000	0.0512	0.1063	0.4987
m19-m1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.2365
m19-m2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.2365
m19-m3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.2365
m19-m4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.1751
m19-m5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m19-m6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.2365
m19-m7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.2365
m19-m8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.2365
m19-m9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.2365
m19-m10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.1676
m19-m11	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.1676

m19-m12	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.0613
m19-m13	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.1676
m19-m14	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.0613
m19-m15	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.0613
m19-m16	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0613	0.0000	0.1063	0.2413
m19-m17	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0613	0.0000	0.1063	0.2413
m19-m18	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.0613
m19-m19	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m19-m20	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0000	0.0000	0.1434
m19-m21	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0821	0.0613	0.0512	0.1063	0.3698
m20-m1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0000	0.1063	0.2489
m20-m2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0000	0.1063	0.2489
m20-m3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.1751
m20-m4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.1751
m20-m5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0000	0.1063	0.1800
m20-m6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2263
m20-m7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2263
m20-m8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2263
m20-m9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2263
m20-m10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0512	0.1063	0.2188
m20-m11	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.1575
m20-m12	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.0000	0.0512
m20-m13	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.2263
m20-m14	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0512	0.1063	0.2312
m20-m15	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0000	0.0000	0.0737
m20-m16	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0000	0.0000	0.0000	0.1063	0.4005

m20-m17	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0000	0.0000	0.0512	0.1063	0.4517
m20-m18	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0512	0.1063	0.3371
m20-m19	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0689	0.0000	0.0000	0.0000	0.0409	0.0000	0.0737	0.0000	0.0000	0.0512	0.1063	0.4517
m20-m20	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m20-m21	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0689	0.0000	0.0000	0.0000	0.0000	0.0000	0.0737	0.0000	0.0000	0.0512	0.1063	0.3001
m21-m1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.1676
m21-m2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.1676
m21-m3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.1676
m21-m4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.1063
m21-m5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
m21-m6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.1676
m21-m7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.1063
m21-m8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.1676
m21-m9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.1063
m21-m10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.1676
m21-m11	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1063	0.1063
m21-m12	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.0613
m21-m13	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.1063	0.1676
m21-m14	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.0613
m21-m15	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.0613
m21-m16	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0000	0.0000	0.0000	0.0295	0.0409	0.0000	0.0737	0.0000	0.0613	0.0000	0.0000	0.3162
m21-m17	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0000	0.0000	0.0000	0.0295	0.0409	0.0000	0.0737	0.0000	0.0613	0.0000	0.0000	0.3162
m21-m18	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0000	0.0000	0.0000	0.0295	0.0000	0.0000	0.0000	0.0000	0.0613	0.0000	0.0000	0.2016
m21-m19	0.0000	0.0000	0.0000	0.0000	0.0000	0.1107	0.0000	0.0000	0.0000	0.0295	0.0409	0.0000	0.0737	0.0000	0.0000	0.0000	0.0000	0.2549
m21-m20	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0295	0.0000	0.0000	0.0000	0.0821	0.0613	0.0000	0.0000	0.1729
m21-m21	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

5.6.6.: Obtaining Ranking Matrix

The obtained sum values are arranged in an n x n matrix, where n is number of stretches. The ranking matrix is obtained as shown in the table 5.42.

5.6.7 Obtaining Prioritization Index From Rating Matrix

For the ranking matrix formed, the row-wise sum is applied to all the rows or criteria to obtain sum values or prioritization index. The sum values are obtained as shown in table 5.43.

ST NO.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
S1	0.000	0.111	0.517	0.247	0.360	0.414	0.146	0.341	0.389	0.239	0.397	0.216	0.374	0.513	0.343	0.788	0.798	0.663	0.736	0.765	0.819
S2	0.820	0.000	0.689	0.296	0.323	0.443	0.249	0.460	0.518	0.443	0.473	0.382	0.479	0.588	0.597	0.870	0.880	0.745	0.819	0.806	0.819
S3	0.317	0.242	0.000	0.232	0.223	0.307	0.270	0.284	0.160	0.321	0.518	0.216	0.425	0.523	0.548	0.819	0.880	0.745	0.819	0.839	0.819
S4	0.685	0.635	0.626	0.000	0.613	0.535	0.427	0.440	0.549	0.472	0.692	0.408	0.483	0.644	0.714	0.931	0.880	0.806	0.880	0.880	0.880
S5	0.640	0.602	0.734	0.387	0.000	0.539	0.574	0.511	0.507	0.435	0.616	0.548	0.595	0.632	0.752	0.843	0.843	0.769	0.843	0.875	0.843
S6	0.482	0.488	0.549	0.397	0.461	0.000	0.260	0.469	0.440	0.405	0.473	0.272	0.438	0.685	0.716	0.870	0.880	0.704	0.819	0.890	0.819
S7	0.786	0.639	0.661	0.504	0.426	0.589	0.000	0.517	0.587	0.552	0.755	0.414	0.530	0.778	0.671	0.931	0.931	0.857	0.819	0.931	0.880
S8	0.590	0.471	0.571	0.385	0.454	0.387	0.340	0.000	0.610	0.564	0.712	0.475	0.498	0.688	0.653	0.870	0.931	0.796	0.819	0.931	0.819
S9	0.512	0.338	0.665	0.382	0.418	0.417	0.344	0.321	0.000	0.405	0.629	0.445	0.561	0.642	0.615	0.888	0.888	0.763	0.776	0.888	0.837
S10	0.639	0.528	0.623	0.528	0.565	0.595	0.448	0.436	0.543	0.000	0.671	0.478	0.509	0.568	0.632	0.857	0.857	0.732	0.887	0.939	0.887
S11	0.603	0.380	0.482	0.308	0.384	0.445	0.112	0.288	0.371	0.329	0.000	0.219	0.401	0.560	0.487	0.835	0.835	0.761	0.722	0.835	0.722
S12	0.784	0.618	0.784	0.592	0.452	0.699	0.545	0.525	0.555	0.522	0.781	0.000	0.691	0.787	0.756	0.832	0.894	0.708	0.781	0.894	0.781
S13	0.585	0.521	0.534	0.474	0.405	0.519	0.470	0.451	0.439	0.491	0.599	0.235	0.000	0.631	0.419	0.870	1.000	0.865	0.887	0.890	0.858
S14	0.487	0.412	0.416	0.356	0.262	0.274	0.222	0.312	0.358	0.432	0.397	0.157	0.369	0.000	0.507	0.789	0.851	0.623	0.738	0.773	0.738
S15	0.616	0.403	0.299	0.286	0.248	0.284	0.329	0.347	0.329	0.368	0.513	0.244	0.540	0.432	0.000	0.781	0.843	0.708	0.781	0.728	0.781
S16	0.212	0.130	0.181	0.069	0.106	0.130	0.069	0.130	0.069	0.143	0.000	0.168	0.130	0.168	0.219	0.000	0.130	0.175	0.257	0.250	0.151
S17	0.202	0.120	0.120	0.120	0.157	0.069	0.069	0.069	0.069	0.143	0.000	0.106	0.000	0.106	0.157	0.307	0.000	0.175	0.264	0.376	0.227
S18	0.337	0.255	0.255	0.194	0.125	0.296	0.143	0.204	0.194	0.268	0.074	0.292	0.135	0.176	0.292	0.353	0.414	0.000	0.302	0.480	0.341
S19	0.181	0.181	0.181	0.120	0.157	0.181	0.181	0.181	0.181	0.113	0.113	0.219	0.113	0.219	0.219	0.186	0.186	0.288	0.000	0.301	0.212
S20	0.235	0.194	0.161	0.120	0.125	0.110	0.069	0.069	0.069	0.061	0.000	0.106	0.110	0.184	0.272	0.345	0.294	0.221	0.294	0.000	0.184
S21	0.113	0.113	0.113	0.051	0.157	0.113	0.051	0.113	0.051	0.113	0.051	0.219	0.142	0.219	0.219	0.474	0.474	0.359	0.412	0.330	0.000

Table 5.43 Ranking matrix for all stretches

ST NO.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	SUM
S1	0.000	0.595	0.719	0.161	0.348	0.562	0.663	0.516	0.599	0.311	1.109	0.321	0.417	0.718	0.371	1.260	1.485	1.134	1.527	1.502	1.527	15.846
S2	1.210	0.000	0.842	0.423	0.331	0.717	0.678	0.532	0.645	0.587	1.048	0.444	0.848	0.951	0.884	1.425	1.650	1.298	1.527	1.502	1.527	19.067
S3	0.950	1.020	0.000	0.353	0.336	0.422	0.634	0.459	0.478	0.475	1.200	0.321	0.589	0.784	0.484	1.527	1.650	1.298	1.527	1.650	1.527	17.684
S4	1.558	1.267	1.509	0.000	0.993	1.384	1.302	0.926	1.414	0.720	1.612	0.595	1.060	1.062	0.962	1.547	1.650	1.502	1.650	1.650	1.650	26.014
S5	1.595	1.436	1.578	1.007	0.000	1.142	1.312	1.057	1.107	1.070	1.464	0.556	1.284	1.367	0.982	1.898	2.000	1.558	2.000	1.640	2.000	28.053
S6	1.016	1.086	1.151	0.419	0.858	0.000	0.835	0.550	0.845	0.644	1.217	0.433	0.615	1.145	0.910	1.425	1.547	1.298	1.527	1.547	1.527	20.594
S7	1.141	0.965	1.006	0.478	0.632	0.863	0.000	0.541	0.800	0.485	1.405	0.453	0.594	1.125	0.820	1.547	1.547	1.400	1.527	1.547	1.650	20.526
S8	1.346	1.330	1.252	0.505	0.814	0.940	1.174	0.000	1.109	0.731	1.503	0.465	1.057	1.117	0.839	1.425	1.547	1.277	1.527	1.547	1.527	23.033
S9	1.135	1.066	1.171	0.367	0.742	0.579	0.921	0.446	0.000	0.485	1.573	0.342	1.012	1.115	0.708	1.462	1.462	1.417	1.441	1.462	1.564	20.467
S10	1.577	1.211	1.413	1.280	0.930	1.356	1.326	1.269	1.413	0.000	1.713	0.572	1.111	1.082	0.784	1.398	1.398	1.353	1.665	1.562	1.665	26.078
S11	0.891	0.658	0.649	0.302	0.397	0.632	0.436	0.260	0.427	0.287	0.000	0.123	0.487	0.805	0.602	1.355	1.355	1.125	1.334	1.355	1.334	14.813
S12	1.679	1.335	1.679	1.211	1.234	1.567	1.465	1.476	1.425	1.272	1.877	0.000	1.697	2.000	1.527	1.775	1.898	1.730	1.877	1.898	1.877	32.499
S13	1.304	1.152	1.218	0.854	0.716	1.218	1.255	0.841	0.988	0.778	1.513	0.156	0.000	0.865	0.745	1.425	1.685	1.195	1.665	1.547	1.606	22.726
S14	1.141	0.881	0.942	0.881	0.633	0.773	0.816	0.883	0.826	0.918	1.109	0.000	0.832	0.000	0.699	1.689	1.811	1.247	1.791	1.314	1.791	20.975
S15	1.547	1.030	1.018	0.887	1.018	1.008	1.008	1.161	1.180	1.078	1.398	0.473	1.173	1.097	0.000	1.877	2.000	1.648	1.877	1.853	1.877	26.210
S16	0.740	0.575	0.473	0.453	0.000	0.575	0.453	0.575	0.453	0.602	0.315	0.225	0.438	0.225	0.123	0.000	0.575	0.102	0.404	0.389	0.617	8.312
S17	0.515	0.350	0.350	0.350	0.000	0.350	0.453	0.453	0.453	0.602	0.315	0.102	0.315	0.102	0.000	0.463	0.000	0.102	0.418	0.438	0.768	6.899
S18	0.866	0.620	0.702	0.498	0.360	0.702	0.600	0.723	0.498	0.647	0.462	0.270	0.667	0.352	0.352	0.816	0.925	0.000	1.056	0.726	0.997	12.841
S19	0.473	0.473	0.473	0.350	0.000	0.473	0.473	0.473	0.473	0.335	0.335	0.123	0.335	0.123	0.123	0.483	0.483	0.123	0.000	0.287	0.740	7.148
S20	0.498	0.498	0.350	0.350	0.360	0.453	0.453	0.453	0.453	0.438	0.315	0.102	0.453	0.462	0.147	0.801	0.903	0.674	0.903	0.000	0.600	9.666
S21	0.335	0.335	0.335	0.213	0.000	0.335	0.213	0.335	0.213	0.335	0.213	0.123	0.335	0.123	0.123	0.632	0.632	0.403	0.510	0.346	0.000	6.088

Table 5.44 Ranking matrix with sum values.

5.6.8 Assigning ranks to different stretches:

From the sums of each row calculated or Priority Indices obtained, the ranks were assigned as shown in table 5.41.

Table 5.45 Concordance Rankings obtained.

STRETCH NO.	SUM	Rank
S1	15.84622	14
S2	19.06728	12
S3	17.68405	13
S4	26.01384	5
S5	28.05267	2
S6	20.59374	9
S7	20.5261	10
S8	23.03313	6
S9	20.46708	11
S10	26.07783	4
S11	14.81337	15
S12	32.49946	1
S13	22.72563	7
S14	20.97549	8
S15	26.20969	3
S16	8.311515	18
S17	6.899148	20
S18	12.84121	16
S19	7.148272	19
S20	9.666222	17
S21	6.087662	21

5.7 SENSITIVITY ANALYSIS AND PEARSON'S COEFFICIENT:

The sensitivity analysis is done such that each weightage is increased one after the other iteratively and the corresponding ranks are compared with the original ranks and the Pierson's coefficient values are obtained for each method as mentioned in chapter 4.

5.7.1 Analysing Pearson Correlation Coefficient For Fuzzy MCDM(Topsis) weightages

The table 5.45 to 5.48 shows the Pierson's values for changes in each weightage when the weightages are increased by 5%, decreased by 5%, increased by 10%, decreased by 10% for TOPSIS.

Table 5.45 Pearson Correlation values 'r' when Topsis weightages are increased by 5%

S No.	Description	Original Wts	After 5% Inc	r
1	Original Rank v/s 5% Inc in Weightage 1	0.167	0.17535	1
2	Original Rank v/s 5% Inc in Weightage 2	0.447	0.46935	1
3	Original Rank v/s 5% Inc in Weightage 3	0.787	0.82635	1
4	Original Rank v/s 5% Inc in Weightage 4	0.167	0.17535	1
5	Original Rank v/s 5% Inc in Weightage 5	0.447	0.46935	1
6	Original Rank v/s 5% Inc in Weightage 6	0.787	0.82635	1
7	Original Rank v/s 5% Inc in Weightage 7	0.167	0.17535	1
8	Original Rank v/s 5% Inc in Weightage 8	0.447	0.46935	1
9	Original Rank v/s 5% Inc in Weightage 9	0.787	0.82635	1
10	Original Rank v/s 5% Inc in Weightage 10	0.167	0.17535	1
11	Original Rank v/s 5% Inc in Weightage 11	0.447	0.46935	1
12	Original Rank v/s 5% Inc in Weightage 12	0.787	0.82635	1
13	Original Rank v/s 5% Inc in Weightage 13	0.527	0.55335	1
14	Original Rank v/s 5% Inc in Weightage 14	0.873	0.91665	1
15	Original Rank v/s 5% Inc in Weightage 15	1	1.05	1
16	Original Rank v/s 5% Inc in Weightage 16	0.26	0.273	1
17	Original Rank v/s 5% Inc in Weightage 17	0.5	0.525	1
18	Original Rank v/s 5% Inc in Weightage 18	0.88	0.924	1
19	Original Rank v/s 5% Inc in Weightage 19	0.247	0.25935	1
20	Original Rank v/s 5% Inc in Weightage 20	0.587	0.61635	1
21	Original Rank v/s 5% Inc in Weightage 21	0.9	0.945	1
22	Original Rank v/s 5% Inc in Weightage 22	0.167	0.17535	1
23	Original Rank v/s 5% Inc in Weightage 23	0.447	0.46935	1
24	Original Rank v/s 5% Inc in Weightage 24	0.787	0.82635	1
25	Original Rank v/s 5% Inc in Weightage 25	0.233	0.24465	1
26	Original Rank v/s 5% Inc in Weightage 26	0.56	0.588	1
27	Original Rank v/s 5% Inc in Weightage 27	0.88	0.924	1
28	Original Rank v/s 5% Inc in Weightage 28	0.247	0.25935	1
29	Original Rank v/s 5% Inc in Weightage 29	0.587	0.61635	1
30	Original Rank v/s 5% Inc in Weightage 30	0.9	0.945	1
31	Original Rank v/s 5% Inc in Weightage 31	0.287	0.30135	1
32	Original Rank v/s 5% Inc in Weightage 32	0.607	0.63735	1
33	Original Rank v/s 5% Inc in Weightage 33	0.88	0.924	1
34	Original Rank v/s 5% Inc in Weightage 34	0.287	0.30135	1
35	Original Rank v/s 5% Inc in Weightage 35	0.607	0.63735	1
36	Original Rank v/s 5% Inc in Weightage 36	0.88	0.924	1

37	Original Rank v/s 5% Inc in Weightage 37	0.6	0.63	1
38	Original Rank v/s 5% Inc in Weightage 38	0.653	0.68565	1
39	Original Rank v/s 5% Inc in Weightage 39	0.527	0.55335	1
40	Original Rank v/s 5% Inc in Weightage 40	0.567	0.59535	1
41	Original Rank v/s 5% Inc in Weightage 41	0.667	0.70035	1

Table 5.46 Pearson Coefficient values 'r' when weightages of Topsis are decreased by 5%

S No.	Description	Original Wts	After 5% Dec	r
1	Original Rank v/s 5% Dec in Weightage 1	0.167	0.15865	1
2	Original Rank v/s 5% Inc in Weightage 2	0.447	0.42465	1
3	Original Rank v/s 5% Inc in Weightage 3	0.787	0.74765	1
4	Original Rank v/s 5% Inc in Weightage 4	0.167	0.15865	1
5	Original Rank v/s 5% Inc in Weightage 5	0.447	0.42465	1
6	Original Rank v/s 5% Inc in Weightage 6	0.787	0.74765	1
7	Original Rank v/s 5% Inc in Weightage 7	0.167	0.15865	1
8	Original Rank v/s 5% Inc in Weightage 8	0.447	0.42465	1
9	Original Rank v/s 5% Inc in Weightage 9	0.787	0.74765	1
10	Original Rank v/s 5% Inc in Weightage 10	0.167	0.15865	1
11	Original Rank v/s 5% Inc in Weightage 11	0.447	0.42465	1
12	Original Rank v/s 5% Inc in Weightage 12	0.787	0.74765	1
13	Original Rank v/s 5% Inc in Weightage 13	0.527	0.50065	1
14	Original Rank v/s 5% Inc in Weightage 14	0.873	0.82935	1
15	Original Rank v/s 5% Inc in Weightage 15	1	0.95	1
16	Original Rank v/s 5% Inc in Weightage 16	0.26	0.247	1
17	Original Rank v/s 5% Inc in Weightage 17	0.5	0.475	1
18	Original Rank v/s 5% Inc in Weightage 18	0.88	0.836	1
19	Original Rank v/s 5% Inc in Weightage 19	0.247	0.23465	1
20	Original Rank v/s 5% Inc in Weightage 20	0.587	0.55765	1
21	Original Rank v/s 5% Inc in Weightage 21	0.9	0.855	1
22	Original Rank v/s 5% Inc in Weightage 22	0.167	0.15865	1
23	Original Rank v/s 5% Inc in Weightage 23	0.447	0.42465	1
24	Original Rank v/s 5% Inc in Weightage 24	0.787	0.74765	1
25	Original Rank v/s 5% Inc in Weightage 25	0.233	0.22135	1
26	Original Rank v/s 5% Inc in Weightage 26	0.56	0.532	1
27	Original Rank v/s 5% Inc in Weightage 27	0.88	0.836	1
28	Original Rank v/s 5% Inc in Weightage 28	0.247	0.23465	1
29	Original Rank v/s 5% Inc in Weightage 29	0.587	0.55765	1
30	Original Rank v/s 5% Inc in Weightage 30	0.9	0.855	1
31	Original Rank v/s 5% Inc in Weightage 31	0.287	0.27265	1
32	Original Rank v/s 5% Inc in Weightage 32	0.607	0.57665	1
33	Original Rank v/s 5% Inc in Weightage 33	0.88	0.836	1
34	Original Rank v/s 5% Inc in Weightage 34	0.287	0.27265	1
35	Original Rank v/s 5% Inc in Weightage 35	0.607	0.57665	1

36	Original Rank v/s 5% Inc in Weightage 36	0.88	0.836	1
37	Original Rank v/s 5% Inc in Weightage 37	0.6	0.57	1
38	Original Rank v/s 5% Inc in Weightage 38	0.653	0.62035	1
39	Original Rank v/s 5% Inc in Weightage 39	0.527	0.50065	1
40	Original Rank v/s 5% Inc in Weightage 40	0.567	0.53865	1
41	Original Rank v/s 5% Inc in Weightage 41	0.667	0.63365	1

Table 5.47 Pearson Coefficient values 'r' when weightages of Topsis are increased by 10%

S No.	Description	Original Wts	After 10% Inc	r
1	Original Rank v/s 5% Inc in Weightage 1	0.167	0.1837	1
2	Original Rank v/s 5% Inc in Weightage 2	0.447	0.4917	1
3	Original Rank v/s 5% Inc in Weightage 3	0.787	0.8657	1
4	Original Rank v/s 5% Inc in Weightage 4	0.167	0.1837	1
5	Original Rank v/s 5% Inc in Weightage 5	0.447	0.4917	1
6	Original Rank v/s 5% Inc in Weightage 6	0.787	0.8657	1
7	Original Rank v/s 5% Inc in Weightage 7	0.167	0.1837	1
8	Original Rank v/s 5% Inc in Weightage 8	0.447	0.4917	1
9	Original Rank v/s 5% Inc in Weightage 9	0.787	0.8657	1
10	Original Rank v/s 5% Inc in Weightage 10	0.167	0.1837	1
11	Original Rank v/s 5% Inc in Weightage 11	0.447	0.4917	1
12	Original Rank v/s 5% Inc in Weightage 12	0.787	0.8657	1
13	Original Rank v/s 5% Inc in Weightage 13	0.527	0.5797	1
14	Original Rank v/s 5% Inc in Weightage 14	0.873	0.9603	1
15	Original Rank v/s 5% Inc in Weightage 15	1	1.1	1
16	Original Rank v/s 5% Inc in Weightage 16	0.26	0.286	1
17	Original Rank v/s 5% Inc in Weightage 17	0.5	0.55	1
18	Original Rank v/s 5% Inc in Weightage 18	0.88	0.968	1
19	Original Rank v/s 5% Inc in Weightage 19	0.247	0.2717	1
20	Original Rank v/s 5% Inc in Weightage 20	0.587	0.6457	1
21	Original Rank v/s 5% Inc in Weightage 21	0.9	0.99	1
22	Original Rank v/s 5% Inc in Weightage 22	0.167	0.1837	1
23	Original Rank v/s 5% Inc in Weightage 23	0.447	0.4917	1
24	Original Rank v/s 5% Inc in Weightage 24	0.787	0.8657	1
25	Original Rank v/s 5% Inc in Weightage 25	0.233	0.2563	1
26	Original Rank v/s 5% Inc in Weightage 26	0.56	0.616	1
27	Original Rank v/s 5% Inc in Weightage 27	0.88	0.968	1
28	Original Rank v/s 5% Inc in Weightage 28	0.247	0.2717	1
29	Original Rank v/s 5% Inc in Weightage 29	0.587	0.6457	1
30	Original Rank v/s 5% Inc in Weightage 30	0.9	0.99	1
31	Original Rank v/s 5% Inc in Weightage 31	0.287	0.3157	1
32	Original Rank v/s 5% Inc in Weightage 32	0.607	0.6677	1
33	Original Rank v/s 5% Inc in Weightage 33	0.88	0.968	1

34	Original Rank v/s 5% Inc in Weightage 34	0.287	0.3157	1
35	Original Rank v/s 5% Inc in Weightage 35	0.607	0.6677	1
36	Original Rank v/s 5% Inc in Weightage 36	0.88	0.968	1
37	Original Rank v/s 5% Inc in Weightage 37	0.6	0.66	1
38	Original Rank v/s 5% Inc in Weightage 38	0.653	0.7183	1
39	Original Rank v/s 5% Inc in Weightage 39	0.527	0.5797	1
40	Original Rank v/s 5% Inc in Weightage 40	0.567	0.6237	1
41	Original Rank v/s 5% Inc in Weightage 41	0.667	0.7337	1

Table 5.48 Pearson Coefficient values ‘r’ when weightages of Topsis are decreased by 10%

S No.	Description	Original Wts	After 10% Dec	r
1	Original Rank v/s 5% Inc in Weightage 1	0.167	0.1503	1
2	Original Rank v/s 5% Inc in Weightage 2	0.447	0.4023	1
3	Original Rank v/s 5% Inc in Weightage 3	0.787	0.7083	1
4	Original Rank v/s 5% Inc in Weightage 4	0.167	0.1503	1
5	Original Rank v/s 5% Inc in Weightage 5	0.447	0.4023	1
6	Original Rank v/s 5% Inc in Weightage 6	0.787	0.7083	1
7	Original Rank v/s 5% Inc in Weightage 7	0.167	0.1503	1
8	Original Rank v/s 5% Inc in Weightage 8	0.447	0.4023	1
9	Original Rank v/s 5% Inc in Weightage 9	0.787	0.7083	1
10	Original Rank v/s 5% Inc in Weightage 10	0.167	0.1503	1
11	Original Rank v/s 5% Inc in Weightage 11	0.447	0.4023	1
12	Original Rank v/s 5% Inc in Weightage 12	0.787	0.7083	1
13	Original Rank v/s 5% Inc in Weightage 13	0.527	0.4743	1
14	Original Rank v/s 5% Inc in Weightage 14	0.873	0.7857	1
15	Original Rank v/s 5% Inc in Weightage 15	1	0.9	1
16	Original Rank v/s 5% Inc in Weightage 16	0.26	0.234	1
17	Original Rank v/s 5% Inc in Weightage 17	0.5	0.45	1
18	Original Rank v/s 5% Inc in Weightage 18	0.88	0.792	1
19	Original Rank v/s 5% Inc in Weightage 19	0.247	0.2223	1
20	Original Rank v/s 5% Inc in Weightage 20	0.587	0.5283	1
21	Original Rank v/s 5% Inc in Weightage 21	0.9	0.81	1
22	Original Rank v/s 5% Inc in Weightage 22	0.167	0.1503	1
23	Original Rank v/s 5% Inc in Weightage 23	0.447	0.4023	1
24	Original Rank v/s 5% Inc in Weightage 24	0.787	0.7083	1
25	Original Rank v/s 5% Inc in Weightage 25	0.233	0.2097	1
26	Original Rank v/s 5% Inc in Weightage 26	0.56	0.504	1
27	Original Rank v/s 5% Inc in Weightage 27	0.88	0.792	1
28	Original Rank v/s 5% Inc in Weightage 28	0.247	0.2223	1
29	Original Rank v/s 5% Inc in Weightage 29	0.587	0.5283	1
30	Original Rank v/s 5% Inc in Weightage 30	0.9	0.81	1
31	Original Rank v/s 5% Inc in Weightage 31	0.287	0.2583	1

32	Original Rank v/s 5% Inc in Weightage 32	0.607	0.5463	1
33	Original Rank v/s 5% Inc in Weightage 33	0.88	0.792	1
34	Original Rank v/s 5% Inc in Weightage 34	0.287	0.2583	1
35	Original Rank v/s 5% Inc in Weightage 35	0.607	0.5463	1
36	Original Rank v/s 5% Inc in Weightage 36	0.88	0.792	0.998701
37	Original Rank v/s 5% Inc in Weightage 37	0.6	0.54	1
38	Original Rank v/s 5% Inc in Weightage 38	0.653	0.5877	1
39	Original Rank v/s 5% Inc in Weightage 39	0.527	0.4743	1
40	Original Rank v/s 5% Inc in Weightage 40	0.567	0.5103	1
41	Original Rank v/s 5% Inc in Weightage 41	0.667	0.6003	1

5.7.2Analysis of Pearson Correlation Coefficient for AHP Weightages:

The table 5.49 to 5.52 shows the Pierson's values for changes in each weightage when the weightages are increased by 5%, decreased by 5%, increased by 10%, decreased by 10% for AHP.

Table 5.49 Pearson Coefficient Correlation values 'r' when weightages of AHP are increased by 5%

S No.	Description	Original Wts	After 5% Inc	r
1	Original Rank v/s 5% Inc in Weightage 1	0.043126399	0.045282719	1
2	Original Rank v/s 5% Inc in Weightage 2	0.034819612	0.036560592	1
3	Original Rank v/s 5% Inc in Weightage 3	0.028514517	0.029940243	1
4	Original Rank v/s 5% Inc in Weightage 4	0.028264686	0.029677921	1
5	Original Rank v/s 5% Inc in Weightage 5	0.075574484	0.079353208	1
6	Original Rank v/s 5% Inc in Weightage 6	0.110730666	0.1162672	1
7	Original Rank v/s 5% Inc in Weightage 7	0.068883027	0.072327178	1
8	Original Rank v/s 5% Inc in Weightage 8	0.055786368	0.058575686	1
9	Original Rank v/s 5% Inc in Weightage 9	0.066380394	0.069699414	1
10	Original Rank v/s 5% Inc in Weightage 10	0.029512839	0.030988481	1
11	Original Rank v/s 5% Inc in Weightage 11	0.04092978	0.042976269	1
12	Original Rank v/s 5% Inc in Weightage 12	0.042873194	0.045016854	1
13	Original Rank v/s 5% Inc in Weightage 13	0.073718384	0.077404303	1
14	Original Rank v/s 5% Inc in Weightage 14	0.082115959	0.086221757	1
15	Original Rank v/s 5% Inc in Weightage 15	0.061312889	0.064378534	1
16	Original Rank v/s 5% Inc in Weightage 16	0.051192861	0.053752504	1
17	Original Rank v/s 5% Inc in Weightage 17	0.10626394	0.111577137	1

Table 5.50 Pearson Coefficient Correlation values 'r' when weightages of AHP are decreased by 5%

S No.	Description	Original Wts	After 5% Dec	r
1	Original Rank v/s 5% Dec in Weightage 1	0.043126399	0.040970079	1
2	Original Rank v/s 5% Dec in Weightage 2	0.034819612	0.033078631	1
3	Original Rank v/s 5% Dec in Weightage 3	0.028514517	0.027088792	1
4	Original Rank v/s 5% Dec in Weightage 4	0.028264686	0.026851452	1
5	Original Rank v/s 5% Dec in Weightage 5	0.075574484	0.07179576	1
6	Original Rank v/s 5% Dec in Weightage 6	0.110730666	0.105194133	1
7	Original Rank v/s 5% Dec in Weightage 7	0.068883027	0.065438875	1
8	Original Rank v/s 5% Dec in Weightage 8	0.055786368	0.052997049	1
9	Original Rank v/s 5% Dec in Weightage 9	0.066380394	0.063061375	1
10	Original Rank v/s 5% Dec in Weightage 10	0.029512839	0.028037197	1
11	Original Rank v/s 5% Dec in Weightage 11	0.04092978	0.038883291	1
12	Original Rank v/s 5% Dec in Weightage 12	0.042873194	0.040729534	1
13	Original Rank v/s 5% Dec in Weightage 13	0.073718384	0.070032465	1
14	Original Rank v/s 5% Dec in Weightage 14	0.082115959	0.078010161	1
15	Original Rank v/s 5% Dec in Weightage 15	0.061312889	0.058247245	1
16	Original Rank v/s 5% Dec in Weightage 16	0.051192861	0.048633218	1
17	Original Rank v/s 5% Dec in Weightage 17	0.10626394	0.100950743	1

Table 5.51 Pearson Coefficient Correlation values 'r' when weightages of AHP are increased by 10%

S No.	Description	Original Wts	After 10% Inc	r
1	Original Rank v/s 10% Inc in Weightage 1	0.043126399	0.047439039	1
2	Original Rank v/s 10% Inc in Weightage 2	0.034819612	0.038301573	1
3	Original Rank v/s 10% Inc in Weightage 3	0.028514517	0.031365969	1
4	Original Rank v/s 10% Inc in Weightage 4	0.028264686	0.031091155	1
5	Original Rank v/s 10% Inc in Weightage 5	0.075574484	0.083131932	1
6	Original Rank v/s 10% Inc in Weightage 6	0.110730666	0.121803733	1
7	Original Rank v/s 10% Inc in Weightage 7	0.068883027	0.075771329	1
8	Original Rank v/s 10% Inc in Weightage 8	0.055786368	0.061365005	1
9	Original Rank v/s 10% Inc in Weightage 9	0.066380394	0.073018434	1
10	Original Rank v/s 10% Inc in Weightage 10	0.029512839	0.032464123	1
11	Original Rank v/s 10% Inc in Weightage 11	0.04092978	0.045022758	1
12	Original Rank v/s 10% Inc in Weightage 12	0.042873194	0.047160514	1
13	Original Rank v/s 10% Inc in Weightage 13	0.073718384	0.081090222	1
14	Original Rank v/s 10% Inc in Weightage 14	0.082115959	0.090327555	1
15	Original Rank v/s 10% Inc in Weightage 15	0.061312889	0.067444178	1
16	Original Rank v/s 10% Inc in Weightage 16	0.051192861	0.056312147	1
17	Original Rank v/s 10% Inc in Weightage 17	0.10626394	0.116890334	1

Table 5.52 Pearson Coefficient Correlation values 'r' when weightages of AHP are decreased 10%

S No.	Description	Original Wts	After 10% Dec	r
1	Original Rank v/s 10% Dec in Weightage 1	0.04312639	0.038813759	1
2	Original Rank v/s 10% Dec in Weightage 2	0.03481961	0.03133765	1
3	Original Rank v/s 10% Dec in Weightage 3	0.02851451	0.025663066	1
4	Original Rank v/s 10% Dec in Weightage 4	0.02826468	0.025438218	1
5	Original Rank v/s 10% Dec in Weightage 5	0.07557448	0.068017035	1
6	Original Rank v/s 10% Dec in Weightage 6	0.11073066	0.0996576	1
7	Original Rank v/s 10% Dec in Weightage 7	0.06888302	0.061994724	1
8	Original Rank v/s 10% Dec in Weightage 8	0.05578636	0.050207731	1
9	Original Rank v/s 10% Dec in Weightage 9	0.06638039	0.059742355	1
10	Original Rank v/s 10% Dec in Weightage 10	0.02951283	0.026561555	1
11	Original Rank v/s 10% Dec in Weightage 11	0.04092978	0.036836802	1
12	Original Rank v/s 10% Dec in Weightage 12	0.04287319	0.038585875	1
13	Original Rank v/s 10% Dec in Weightage 13	0.07371838	0.066346545	1
14	Original Rank v/s 10% Dec in Weightage 14	0.08211595	0.073904363	1
15	Original Rank v/s 10% Dec in Weightage 15	0.06131288	0.055181601	1
16	Original Rank v/s 10% Dec in Weightage 16	0.05119286	0.046073575	1
17	Original Rank v/s 10% Dec in Weightage 17	0.10626394	0.095637546	1

5.7.3 Analysis of Pearson Coefficient Correlation for Concordance Approach

The table 5.53 to 5.56 shows the Pearson coefficient correlation values for changes in each weightage when the weightages are increased by 5%, decreased by 5%, increased by 10%, decreased by 10% for Concordance.

Table 5.53 Pearson Coefficient Correlation values 'r' when weightages of Concordance are increased by 5%

S No.	Description	Original Wts	After 5% Inc	r
1	Original Rank v/s 5% Inc in Weightage 1	0.0431263	0.04528271	0.83636363
2	Original Rank v/s 5% Inc in Weightage 2	0.0348196	0.03656059	0.83636363
3	Original Rank v/s 5% Inc in Weightage 3	0.0285145	0.029940243	0.83636363
4	Original Rank v/s 5% Inc in Weightage 4	0.0282646	0.0296779	0.83636363
5	Original Rank v/s 5% Inc in Weightage 5	0.0755744	0.07935320	0.83636363
6	Original Rank v/s 5% Inc in Weightage 6	0.1107306	0.1162672	0.84285714
7	Original Rank v/s 5% Inc in Weightage 7	0.0688830	0.07232717	0.84285714
8	Original Rank v/s 5% Inc in Weightage 8	0.0557863	0.05857568	0.83636363
9	Original Rank v/s 5% Inc in Weightage 9	0.0663803	0.06969941	0.83636363
10	Original Rank v/s 5% Inc in Weightage 10	0.0295128	0.03098848	0.83636363
11	Original Rank v/s 5% Inc in Weightage 11	0.0409297	0.04297626	0.83636363
12	Original Rank v/s 5% Inc in Weightage 12	0.0428731	0.04501685	0.83636363
13	Original Rank v/s 5% Inc in Weightage 13	0.0737183	0.07740430	0.83636363
14	Original Rank v/s 5% Inc in Weightage 14	0.0821159	0.08622175	0.83636363

15	Original Rank v/s 5% Inc in Weightage 15	0.0613128	0.06437853	0.83636363
16	Original Rank v/s 5% Inc in Weightage 16	0.0511928	0.05375250	0.84285714
17	Original Rank v/s 5% Inc in Weightage 17	0.1062639	0.11157713	0.836363636

Table 5.54 Pearson Coefficient Correlation values 'r' when weightages of Concordance are decreased by 5%

S No.	Description	Original Wts	After 5% Dec	r
1	Original Rank v/s 5% Dec in Weightage 1	0.0431263	0.04097007	0.842857
2	Original Rank v/s 5% Dec in Weightage 2	0.0348196	0.03307863	0.836363
3	Original Rank v/s 5% Dec in Weightage 3	0.0285145	0.02708879	0.836363
4	Original Rank v/s 5% Dec in Weightage 4	0.0282646	0.02685145	0.836363
5	Original Rank v/s 5% Dec in Weightage 5	0.0755744	0.07179576	0.836363
6	Original Rank v/s 5% Dec in Weightage 6	0.1107306	0.10519413	0.836363
7	Original Rank v/s 5% Dec in Weightage 7	0.0688830	0.06543887	0.836363
8	Original Rank v/s 5% Dec in Weightage 8	0.0557863	0.05299704	0.8363636
9	Original Rank v/s 5% Dec in Weightage 9	0.0663803	0.06306137	0.836363
10	Original Rank v/s 5% Dec in Weightage 10	0.0295128	0.02803719	0.836363
11	Original Rank v/s 5% Dec in Weightage 11	0.0409297	0.03888329	0.836363
12	Original Rank v/s 5% Dec in Weightage 12	0.0428731	0.04072953	0.836363
13	Original Rank v/s 5% Dec in Weightage 13	0.0737183	0.07003246	0.842857
14	Original Rank v/s 5% Dec in Weightage 14	0.0821159	0.07801016	0.836363
15	Original Rank v/s 5% Dec in Weightage 15	0.0613128	0.05824724	0.833766
16	Original Rank v/s 5% Dec in Weightage 16	0.0511928	0.04863321	0.836363
17	Original Rank v/s 5% Dec in Weightage 17	0.1062639	0.10095074	0.842857

Table 5.55 Pearson Coefficient Correlation values 'r' when weightages of Concordance are increased by 10%

S No.	Description	Original Wts	After 10% Inc	r
1	Original Rank v/s 10% Inc in Weightage 1	0.043126399	0.047439039	0.836363636
2	Original Rank v/s 10% Inc in Weightage 2	0.034819612	0.038301573	0.836363636
3	Original Rank v/s 10% Inc in Weightage 3	0.028514517	0.031365969	0.836363636
4	Original Rank v/s 10% Inc in Weightage 4	0.028264686	0.031091155	0.836363636
5	Original Rank v/s 10% Inc in Weightage 5	0.075574484	0.083131932	0.836363636
6	Original Rank v/s 10% Inc in Weightage 6	0.110730666	0.121803733	0.842857143
7	Original Rank v/s 10% Inc in Weightage 7	0.068883027	0.075771329	0.846753247
8	Original Rank v/s 10% Inc in Weightage 8	0.055786368	0.061365005	0.842857143
9	Original Rank v/s 10% Inc in Weightage 9	0.066380394	0.073018434	0.836363636
10	Original Rank v/s 10% Inc in Weightage 10	0.029512839	0.032464123	0.836363636
11	Original Rank v/s 10% Inc in Weightage 11	0.04092978	0.045022758	0.836363636
12	Original Rank v/s 10% Inc in Weightage 12	0.042873194	0.047160514	0.836363636
13	Original Rank v/s 10% Inc in Weightage 13	0.073718384	0.081090222	0.836363636
14	Original Rank v/s 10% Inc in Weightage 14	0.082115959	0.090327555	0.836363636
15	Original Rank v/s 10% Inc in Weightage 15	0.061312889	0.067444178	0.836363636

16	Original Rank v/s 10% Inc in Weightage 16	0.051192861	0.056312147	0.84025974
17	Original Rank v/s 10% Inc in Weightage 17	0.10626394	0.116890334	0.836363636

Table 5.56 Pearson Coefficient Correlation values 'r' when weightages of Concordance are decreased by 10%

S No.	Description	Original Wts	After 10% Dec	r
1	Original Rank v/s 10% Dec in Weightage 1	0.043126399	0.038813759	0.842857143
2	Original Rank v/s 10% Dec in Weightage 2	0.034819612	0.03133765	0.836363636
3	Original Rank v/s 10% Dec in Weightage 3	0.028514517	0.025663066	0.836363636
4	Original Rank v/s 10% Dec in Weightage 4	0.028264686	0.025438218	0.836363636
5	Original Rank v/s 10% Dec in Weightage 5	0.075574484	0.068017035	0.836363636
6	Original Rank v/s 10% Dec in Weightage 6	0.110730666	0.0996576	0.836363636
7	Original Rank v/s 10% Dec in Weightage 7	0.068883027	0.061994724	0.836363636
8	Original Rank v/s 10% Dec in Weightage 8	0.055786368	0.050207731	0.836363636
9	Original Rank v/s 10% Dec in Weightage 9	0.066380394	0.059742355	0.836363636
10	Original Rank v/s 10% Dec in Weightage 10	0.029512839	0.026561555	0.836363636
11	Original Rank v/s 10% Dec in Weightage 11	0.04092978	0.036836802	0.836363636
12	Original Rank v/s 10% Dec in Weightage 12	0.042873194	0.038585875	0.836363636
13	Original Rank v/s 10% Dec in Weightage 13	0.073718384	0.066346545	0.842857143
14	Original Rank v/s 10% Dec in Weightage 14	0.082115959	0.073904363	0.835064935
15	Original Rank v/s 10% Dec in Weightage 15	0.061312889	0.055181601	0.833766234
16	Original Rank v/s 10% Dec in Weightage 16	0.051192861	0.046073575	0.836363636
17	Original Rank v/s 10% Dec in Weightage 17	0.10626394	0.095637546	0.842857143

5.7.4 Comparison Of Pearson Coefficient Correlation Between Fuzzy Topsis, AHP and Concordance ranks

The ranks from each method are arranged as shown in Table 5.57. The Pearson's coefficient is calculated for ranks of 2 methods at once as shown below.

Table 5.57 Table showing Ranks of Fuzzy Topsis, AHP, Concordance

Stretch No.	RANKS OBTAINED FROM		
	Fuzzy TOPSIS	AHP	Concordance
1	14	14	14
2	8	8	12
3	11	12	13
4	6	6	5
5	1	1	2
6	5	7	9
7	2	3	10
8	3	5	6
9	9	9	11
10	7	4	4
11	15	15	15
12	4	2	1
13	12	11	7
14	13	13	8
15	10	10	3
16	21	21	18
17	20	20	20
18	16	16	16
19	19	19	19
20	17	17	17
21	18	18	21

Pearson's coefficient for Fuzzy TOPSIS and AHP 0.9844

Pearson's coefficient for Fuzzy TOPSIS and Concordance 0.8377

Pearson's coefficient for Fuzzy Concordance and AHP 0.8792

CHAPTER 6

FINDINGS AND RESULTS

From various values obtained from the calculations and the study area, the following findings were found:

1. The total study area has an area of 69,37,452 m^2. In that, highest area is occupied by Ibrahim Bagh with a value 5,02,123 m^2 and the least area is occupied by Secretariat colony with a value 49,841.06 m^2.
2. Total 21 stretches were considered, out of which Stretch 16 to Stretch 21 are two laned and remaining others are single laned.
3. From the traffic volume recorded, the highest volume is recorded in S8 with a value of 943 vehicles/ hour and the least value is recorded in Stretch 20 with a value of 63 vehicles/ hour.
4. The highest population count from the data obtained is present in Stretch 10 with a population of 24,637 individuals and the lowest population count is present in Stretch 16 with a population of 2010 individuals.
5. From the nearby facilities available, the highest facility score was recorded in Stretch 19 with a score of 200 and least score is recorded in Stretch 10 with a score of 70.
6. Nearby town distance is observed least for Stretch 21 with a distance of 200 m and the greatest distance was observed for Stretch 3 with a distance of 1.5 kms.
7. Raveling is observed to be the most common defect on pavements and patching are recorded to be the least.
8. From the CBR values obtained, stretch 15 is observed to be having greatest value 47.202 KN, and thus showing that the soil present in the stretch is of greatest strength. The least value is obtained for stretch 7, thus requiring higher focus.

9. According to the ranks obtained from all, MCDM methods, stretch 5 should be given the highest priority, followed by stretch 7 and stretch 12.
10. Some of the ranks were found to be varying from one method to another to some noticeable extent, which can be considered to be negligible.
11. The Pearson coefficient correlation value obtained from the Fuzzy TOPSIS, and Concordance is lesser than the value obtained from the Concordance and AHP, showing that the correlation between AHP and Concordance in stronger and more significant than the other correlation.
12. All the results display Pierson's coefficient in positive values, this shows that the ranks of all the three methods are positively correlated, even when they are containing some level of discrepancies within them.
13. From the sensitivity analysis carried out for each method, Concordance displayed more sensitive results and constantly varying for every change in the weightages.

From the above finding's it is clear that there is need for developing an algorithm that is more precise. The goal of the project is to prioritize the roads based on various distresses they are subjected to, due to many known and unknown factors and provide an optimal choice for the users, to choose the pavements to prioritize for their maintenance. This research has been done by using 6 main criteria i.e., type and severity of distresses, strength of soil, population of each stretch considered, Facility scores by identifying the major public services and assigning them scores, volume of traffic, nearest towns present. Three main algorithms written in python for TOPSIS, AHP and concordance are used in this study. More accurate results can be obtained if even more criteria such as roughness index, skid resistance, temperate analysis etc., data are added. This shows that the accuracy of the algorithm increases taking more criteria into consideration.

CHAPTER 7

SUMMARY AND CONCLUSION

7.1 SUMMARY

The objective of this study is to prioritize the road stretches based on various criteria considered and analyzing them by integrating them into Multi-Criteria Decision Making (MCDM) techniques. Fuzzy TOPSIS, Analytical Hierarchy Process and Concordance are used for this purpose are they are accurate and not complex in nature. GIS is used as a supporting tool for the analysis. 21 study stretches, each measuring 1 km, were taken in the areas Ibrahimbagh, Alijapur, Neknampur, Mahalneknampur and Jai hind colony. Upon contemplation, 12 distress criteria each subdivided as low, medium and high are considered along with 5 other criteria, summing to a total of 41 criteria. The criteria taken are well defined which is found of to be having a significant impact on our objective. The criteria are:

1. Types of distresses
    - Alligator Cracking
    - Block Cracking
    - Longitudinal Cracking
    - Transverse Cracking
    - Potholes
    - Raveling
    - Rutting
    - Shoving
    - Edge Cracking
    - Depression
    - Patching
    - Mud
2. Population for each stretch
3. Facility scores

4. Nearby towns
5. Volume of traffic (At peak hours)
6. Strength of soil

The types of distress signify the severity of stretch damage due to external factors like excessive vehicular loads, sunlight exposure, water intrusion, unequal expansion, contraction due to seasonal changes, etc. The roads can only be brought to their restructured form, if one can understand the distresses it has undergone.

Population of each stretch is the summation of population of all the nearby towns lying within a proximity of 2 kms. It signifies the number of users using the particular stretch. More the population, more the level of distresses is observed.

Facility score is the summation of the measure of importance given to a various public commodities based on the level of importance assigned to them. It basically helps us to identify the roads that are extremely important and are used often such as for emergency purposes, for tourist attractions etc. For this study, the facilities of each stretch such as schools, hospitals, post-office, tourist places, type of road, proximity to the nearby lake etc., are taken into consideration and a predefined score is assigned and summed up.

Volume of traffic is the measure of traffic recorded in the peak hour period on a particular stretch. For this, the traffic intensity was recorded for 3 hours at each stretch in weekdays and the peak value was considered. More the volume of traffic, more can be the distress intensities.

Strength of soil is a very important characteristic for laying of roads which is calculated by California Bearing Ratio (CBR) test. For this study, 3 samples each weighing 6 – 7 kgs were collected from each stretch and by CBR test the soil strength values were obtained. Lower the value of CBR, higher is the level of maintenance required for the stretch.

In the Fuzzy TOPSIS method, the raw field data was obtained and normalization was done on each criterion by taking the maximum value under each criterion. From these normalized values obtained in a scale of 0 to 100, a rating matrix is obtained by assigning them in between a scale of 1 to 10.

15 Expert's decisions were taken from 15 different experts, to give an importance level for each criterion in a 5-scale range of negligible, low, medium, high, very high and by using these standard linguistic variables of the fuzzy topsis method, Triangular Fuzzy Numbers (TFN) were calculated for each variable. From the fuzzy numbers obtained and by summing them up, Fuzzy weights for each criterion are obtained.

Using the Rating matrix and Fuzzy weights, the Fuzzy evaluation values of each stretch is calculated and to establish a relative preference difference between all combinations of the fuzzy values has been computed. A fuzzy preference matrix is created to understand the difference between all combinations of the fuzzy values that has been computed. Each computed difference produced a graph with a total of 441 graphs.

From the graphs obtained, the negative and positive areas are calculated for each difference and these values are inputted into the fuzzy preference matrix(E). Using this matrix priority index (PI) is calculated for all the stretches and based on these values obtained, greatest value has been assigned with the least rank 1.

In the Analytical Hierarchy Process (AHP), 15 expert Pair Wise matrices for all the 17 major criteria were obtained from 15 different experts in the field, comparing their relative importance by Satty's scale, scaling from 1 to 9 and these Pair Wise matrices are combined into a single matrix using geometric mean on each corresponding element of the matrices. These values are arranged into the matrix again and the row-wise mean is performed calculating the criteria weights.

A Consistency check was performed on the values obtained by multiplying the criteria weights obtained to the combined pair wise matrix. From the calculated matrix a row-wise sum is performed to obtain Weight Sum Value (WSV) and by dividing the WSV with criteria weights and summing up all the values, λ_{max} value is obtained. From the λ_{max} values, Consistency Index (CI) is obtained. A standard

Random Index (RI) value is assigned based on the number of criterias present and by dividing the CI with RI value, Consistency Ratio (CR) is obtained. The consistency ratio should be < 0.01, else the Pair Wise matrix is defined to be inconsistent.

After the consistency check is performed and if the check has got satisfied, the Normalized values of the raw data were imported from the Fuzzy TOPSIS algorithm and they were combined into 17 major criteria. The Normalized values are multiplied with the consistency check satisfied Weightages and the priority indexes for each stretch are obtained. The priority indexes are then used for raking the stretches, where the greatest value signifies more importance, and hence ranked as 1.

In the Concordance method, the raw data values are imported and they are summed up into 17 major criterias. The maximum and minimum values of each criterion are observed and the data is normalized using logarithmic transformation between 0 to 1.

The Weightages for each criterion are obtained from AHP. For the normalized values computed, to establish a relative preference between all combinations, each value is under a criterion is compared with other values under the same criteria and if the first value > second value, the corresponding criteria weightage is assigned, else 0 is given. From this, n x n comparisons are observed under each criterion, where n is number of stretches and are arranged into a matrix. Now for all comparisons in a row, i.e., the comparisons between all the combinations, a row-wise sum is performed and the values obtained are arranged in an n x n matrix. In this matrix again the row-wise sum is performed to obtain the sum or Priority Index (PI). Using this PI, the rankings are given accordingly, wherein the highest value is given the least rank 1.

A sensitivity analysis was performed on the ranks obtained from each method. The weightage obtained in each method are taken one by one, increased by 5% and the ranks are exported. In the same way, sensitivity by decreasing by 5%, increasing by 10% and decreasing by 10% are computed. All the variations of ranks, if any, are compared with the original ranks using Pearson's coefficient and the correlation values are obtained. If the "r" values obtained is 1, the model chosen is defined as a reliable method. Else it is an unreliable method.

Using ARCGIS 10.1, a GIS tool, the thematic maps were prepared to visualize each and every criterion, and also according to the criteria relative importance between each stretch can be clearly understood from the maps generated. The final rankings for the order of prioritization for all the stretches chosen is also displayed using these maps.

By comparing all the methods, it has been found from the results that stretch 5 should be given the highest priority, succeeded by stretch 7 and stretch 12. Hence, Multi Criteria Decision Making and fuzzy logic can be used in transport projects to evaluate and prioritize the road maintenance activities and can have a wide impact. The fuzzy logic of TOPSIS, AHP and concordance are given similar weightages for the ranking they provide. They provide weightages according to their mathematical methodologies and provides ranks for each stretch, that are almost equal and comparable. Keeping the methods used apart, this approach is a tool for planning of road networks to identify the road stretches with maximum distresses and in need for repair. The method can also be used in optimum budget or resource allocations. This helps in understanding the importance of maintaining a road in their initial stages and to take measures using any constraints. The study has been generalized in a frame work, that can be used and implemented at any part of the country.

Our primary objective of the project is to develop a computer algorithm in a user-friendly and currently in trend programming language, the python programming language, such that it is convenient and flexible enough for the users to input 'n' number of criterias, in order make optimally informed decisions. This project mainly focuses on the computation through three MCDM mathematical models chosen for road maintenance priority order. For this project, choosing of the additional criteria such as roughness index, skid resistance, pavement conditions while laying the road, climatic change over the year etc., are allowed, in addition to the chosen criteria and can be added as many as possible to make the algorithm furnish the results as accurate as possible. Following which, more such methods can be integrated, to decrease the level of uncertainty in the results.

7.2 CONCLUSION

Prioritisation of road stretches in urban sprawl area has been achieved in the study using Multi criteria decision making approaches. This study applies the three MCDM methodologies, Fuzzy logic, AHP and concordance analysis, to the same problem under the same conditions. The identical sets of alternatives and criteria are used. The same solutions for road network prioritisation, are arrived at by using all the methodologies. The study also gives a sensitivity analysis. The initial experiment (with the same solution) and 300 additional experiments were carried out applying the Fuzzy logic, AHP and concordance analysis, by changing each criterion conditions. It is shown that the concordance analysis is sensitive to this kind of changing, while the other approaches are not sensitive at all. The solutions obtained by AHP, and fuzzy logic showed reliable results. The existing distress condition of Rank 1 has been correlated in the field and the rank was found to be true representation of field condition. The approaches proved to be a strong tool for pavement prioritisation for road maintenance needs and these criterions for evaluation can be applied for urban sprawl roads.

REFERENCES

1. Sandra. A.K, Vinayaka Rao. V.R, Raju. K.S, Sarkar. A.K, "Prioritization of Pavement Stretches using Fuzzy MCDM Approach - A Case Study" Soft Computing in Industrial Applications (pp.265-278) (2010).
2. Amrita Sarkar et al, "Application of Fuzzy Logic in Transport Planning" International Journal on Soft Computing (IJSC) (AIRCC) Vol.3, Issue.2, Pages 1-21. (2012)
3. Dr.Aakash Gupta et al, "Rural Road Maintenance Prioritization Index Based on Functional and Structural Parameters for Rural Road Network In Himachal Pradesh" International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8, Issue-1, May 2019.
4. R.W. Satty, "The Analytic Hierarchy Process- what it is and how it is used" Mathematical Modelling, Vol. 9, No. 3-5, pp. 161-176, 1987.
5. T.F. Fwa and R. Shanmugam, "Fuzzy Logic Technique for Pavement Condition Rating & Maintenance Needs Assessment" 4th International Conference on Managing Pavements (1998).
6. Danial Moazami and Ratnasamy Muniandy, "Fuzzy Inference and Multi-criteria Decision-Making Applications in Pavement Rehabilitation Prioritization" Australian Journal of Basic and Applied Sciences, 4(10): 4740-4748, 2010 ISSN 1991-8178.
7. Yashon O. Ouma et al, "Comparison of Fuzzy AHP and Fuzzy TOPSIS for Road Pavement Maintenance Prioritization: Methodological Exposition and Case Study" Volume 2015, Article ID 140189, 17 pages.
8. Fardeen Nodrat and Dongshik Kang, "Prioritizing Road Maintenance Activities using GIS Platform and Vb.net" (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 9, No. 2, 2018.

9. Hussein Ali Ewadh et al "Developing optimized prioritizing road maintenance" MATEC Web of Conferences 162, 01044 (2018).

10. Navid Khademi and Abdolreza Sheikholeslami, "Multicriteria Group Decision-Making Technique for a Low-Class Road Maintenance Program" Journal of Infrastructure Systems, Vol. 16, No. 3, American Society of Civil Engineers, ISSN 1076-0342, pp 188-198, 2010.

11. Joni Arliansyah et al, "A Development of Fuzzy Pavement Condition Assessment" J. Materials, Conc. Struct. Pavements, JSCE, No. 746/V-61, 275-285, 2003 November.

12. "Managing Road Maintenance Using Geographic Information System Application"

13. Mohd Zulkifli B. Mohd Yunus and Hamidah Bt. Hassan, "Managing Road Maintenance Using Geographic Information System Application" Journal of Geographic Information System, 2010, 2, 215-219.

14. Prof. Mayank Kanani et al, "Pavement maintenance and prioritization using AHP- A case study of Rajkot city" International Journal of Pavement Research & Technology Mar 2015, Vol. 8 Issue 2, p112-122. 11p.

15. "Using Fuzzy Logic and Expert System Approaches in Evaluating Flexible Pavement Distress- Case Study"

16. Koduru Hari Krishan, Xiao Feipeng, Amir khanian Serji N. and Juang C. Hsein, "Using Fuzzy Logic and Expert System Approaches in Evaluating Flexible Pavement Distress: Case Study" Journal of Transportation Engineering, ASCE. (February 2010):149-157.

17. IRC:82-2015 "Code of Practise for Maintenance of Bituminous Road Surfaces".

18. IRC: SP:83-2018 "Guidelines for Maintenance, Repair and Rehabilitation of Cement Concrete Pavements".

APPENDIX A

A.1 TYPES OF DISTRESSES OBSERVED IN THE FIELD

Fig A.1: Alligator Cracking

Fig A.2: Block Cracking

Fig A.3: Longitudinal Cracks

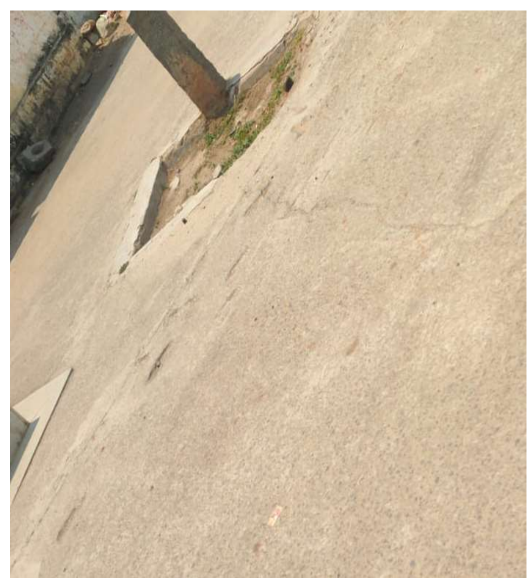


Fig A.4: Transverse Crack

Fig: A.5 Ravelling

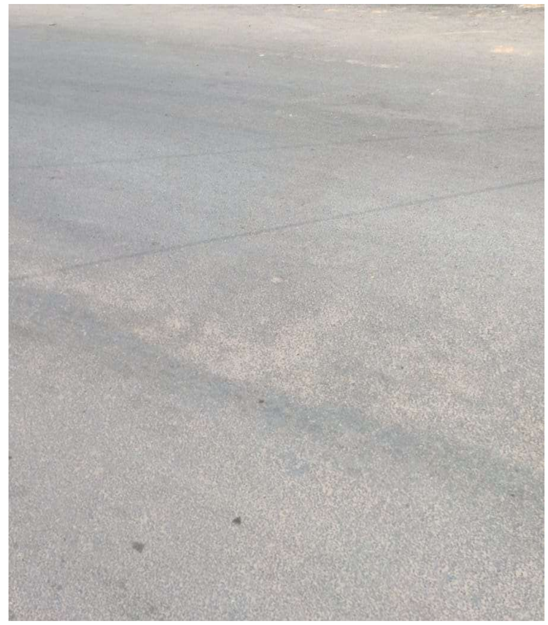


Fig: A.6 Rutting

Fig A.7 Pothole

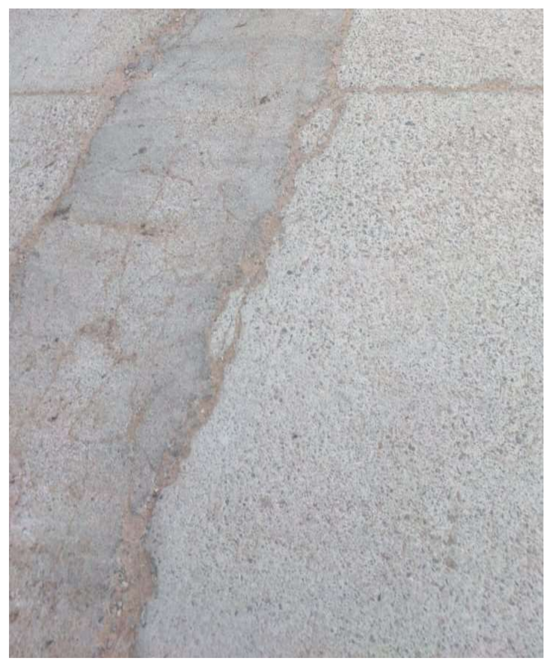


Fig A.8 Patching

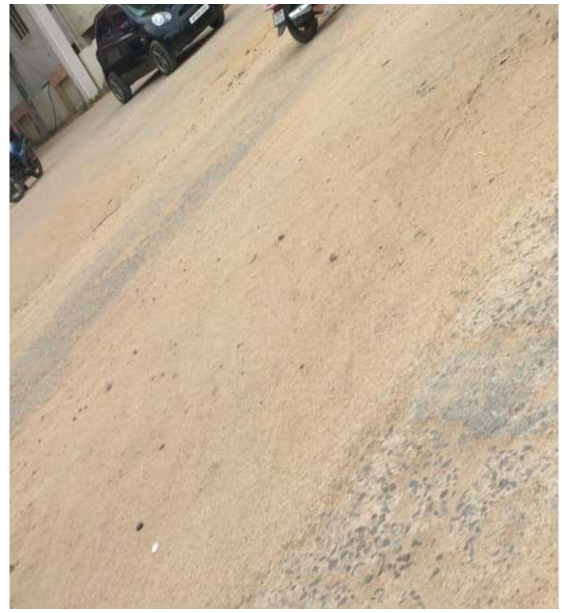



Fig: A.9 Shoving **Fig: A.10 Mud**


Fig A.11 Depression **Fig A.12 Edge Cracking**

APPENDIX B

B.1 CBR TEST

The California Bearing Ratio (CBR) is a measure of the strength of the subgrade of a road or other paved area, and of the materials used in its construction. In this project, three samples weighing, 6 to 7 kgs, were collected from 3 parts of each stretch for all 21 stretches, and were averaged to single values. The test was performed as mentioned in _.

B.1.1 CBR TEST PROCEDURE

The following pictures demonstrates the step-by-step process performed by the team to obtain the load and penetration values from the test apparatus.

Fig B.1: Collection of samples from site **Fig B.2: Peforming Sieve Analysis**

Fig B.3: Preparation of mould

Fig B.4: Putting Sample into the mould

Fig B.5: Giving blows to the sample

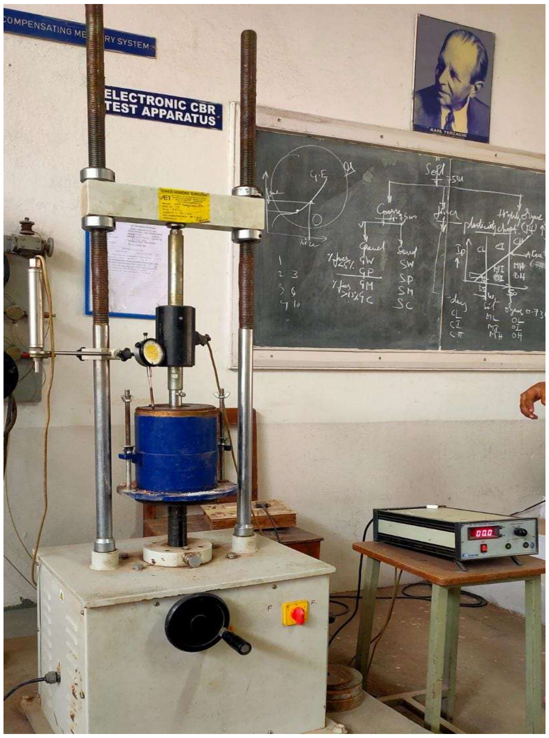


Fig 9.6: CBR test apparatus

Fig 9.7: Performing CBR test

B.1.2 SAMPLES COLLECTED

The Table 9.1 below represents the values of penetration of plunger (mm) and load applied (KN) on each soil sample.

B.1.3 CBR TEST GRAPHS

The CBR test graphs for all stretches are plotted as shown in Fig 9.8 to Fig 9.28. The graphs are plotted between applied load (KN) on the Y-axis and penetration values (mm) on the X-axis from the values in Table 9.1. Corrections are applied to the graphs if required.

Table B.1 CBR test sample values for each str

Penetration of Plunger, (mm)	S 1 (kgf)	S 2 (kgf)	S 3 (kgf)	S 4 (kgf)	S 5 (kgf)	S 6 (kgf)	S 7 (kgf)	S 8 (kgf)	S 9 (kgf)	S 10 (kgf)	S 11 (kgf)	S 12 (kgf)	S 13 (kgf)	S 14 (kgf)	S 15 (kgf)	S 16 (kgf)	S 17 (kgf)	S 18 (kgf)	S 19 (kgf)	S 20 (kgf)	S 21 (kgf)
0.0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.5	1.9	0.3	0.4	1.4	0.7	0.4	1.5	2	0.9	1.1	0.5	0.9	1.2	1.1	0.9	0.3	0.5	1	0.9	1.2	1
1.0	2.4	0.7	1.2	2	2.3	0.9	1.4	2.8	1.7	2.2	1	2.7	2	3.7	2.4	1.3	1.5	2.1	1.5	2.3	2
1.5	2.5	1.1	1.8	2.7	3.6	1.3	1.2	3.3	1.9	3	1.2	4.2	2.6	4.7	3.9	2.5	3.3	3.6	2.9	3.6	2.9
2.0	2.4	1.5	2.4	3.3	4.9	1.8	1.1	3.8	2.4	3.8	1.5	5.7	3.1	5.3	5	4	4.3	4.9	3.9	4.9	4.1
2.5	2.6	1.9	2.9	3.8	5.8	2.3	1.2	4.2	2.9	4.4	1.9	6.8	3.5	5.9	6.3	5.5	5.3	5.9	5.5	6.1	5.1
3.0	2.9	2.3	3.4	4.3	6.4	2.8	1.3	4.6	3.3	5	2.5	7.9	3.9	6.4	7.3	6.4	6.5	6.4	6	6.8	6.4
4.0	3.7	3.1	4.2	5.3	7.5	3.9	2	5.3	3.9	6.1	3.1	9.8	4.7	7.3	8.9	7.5	7.3	7.5	7.5	7.5	7.3
5.0	4.1	3.8	4.8	6.3	8.9	5.1	2.8	6.3	4.8	7	3.8	11.4	5.5	9	9.7	8.5	8.2	9	8.7	9.6	8
7.5	5.4	5.2	6.3	8.7	10.7	6.9	4.2	7.5	5.4	9.1	5.3	15.1	6.4	9.9	11.2	10.3	9.7	10.7	9.7	10.8	9.5
10.0	6.4	6.6	7.4	10.9	12.5	8.3	5	8.8	6.4	11	6.6	18.2	7.3	11.6	13.2	12	11.5	12.5	11.9	12.1	11
12.5	7.4	8	8.3	13	14.2	9.5	5.7	10	7.4	12.8	7.8	19.3	8.3	13.4	14.5	13.2	12.3	14	13.2	13.6	12.5

2.5 mm	0.190	0.139	0.212	0.277	0.423	0.168	0.088	0.307	0.212	0.321	0.139	0.496	0.255	0.431	0.460	0.401	0.387	0.431	0.401	0.445	0.372
5 mm	0.200	0.185	0.234	0.307	0.433	0.248	0.136	0.307	0.234	0.341	0.185	0.555	0.268	0.438	0.472	0.414	0.399	0.438	0.423	0.467	0.389

5mm x 100	19.951	18.491	23.358	30.657	43.309	24.818	13.625	30.657	23.358	34.063	18.491	55.474	26.764	43.796	47.202	41.363	39.903	43.796	42.336	46.715	38.929

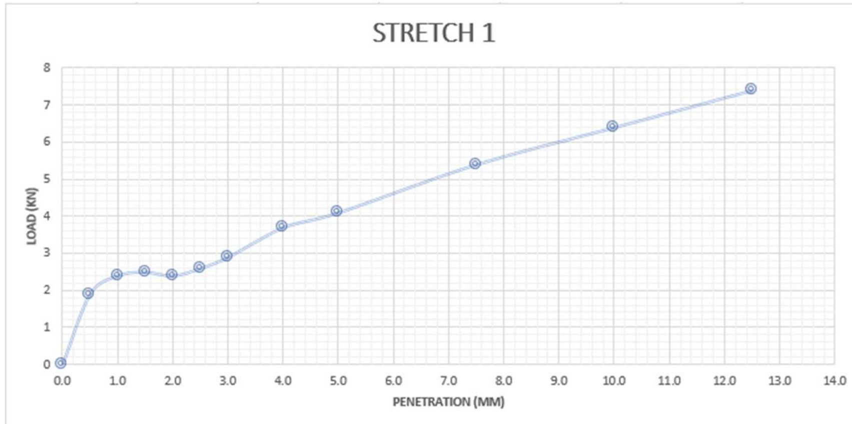


Fig: 9.8 CBR test curve for stretch 1

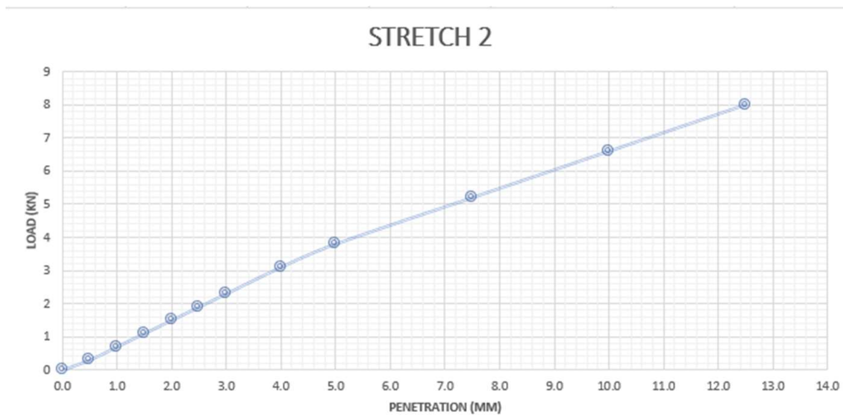


Fig: 9.9 CBR test curve for stretch 2

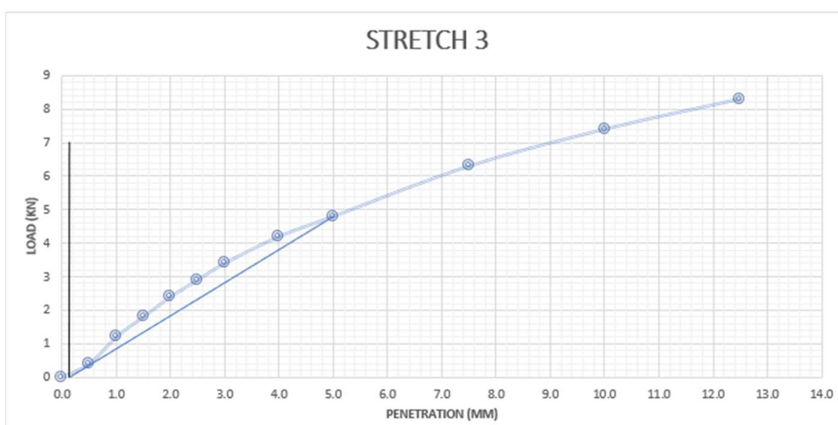


Fig: 9.10 CBR test curve for stretch 3

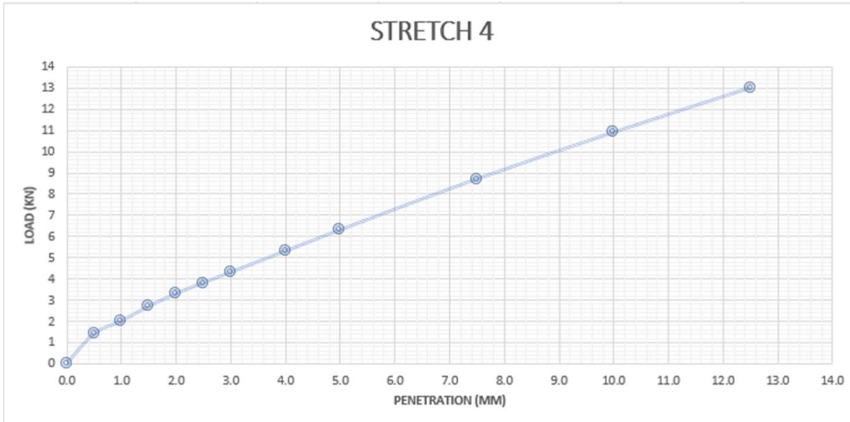


Fig: 9.11 CBR test curve for stretch 4

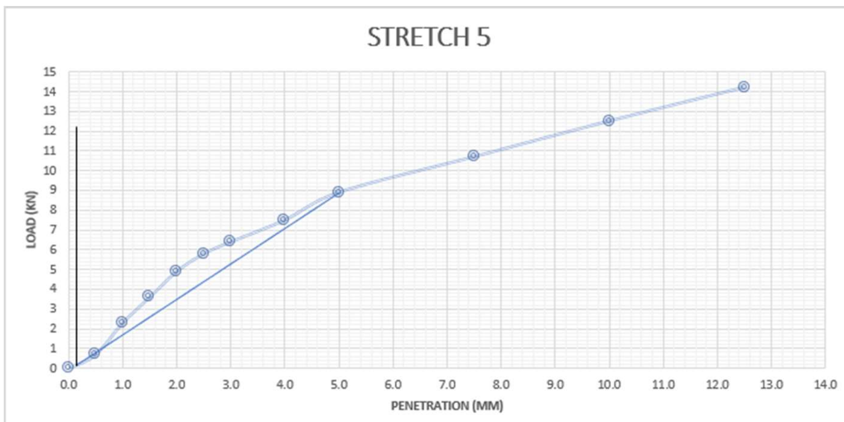


Fig: 9.12 CBR test curve for stretch 5

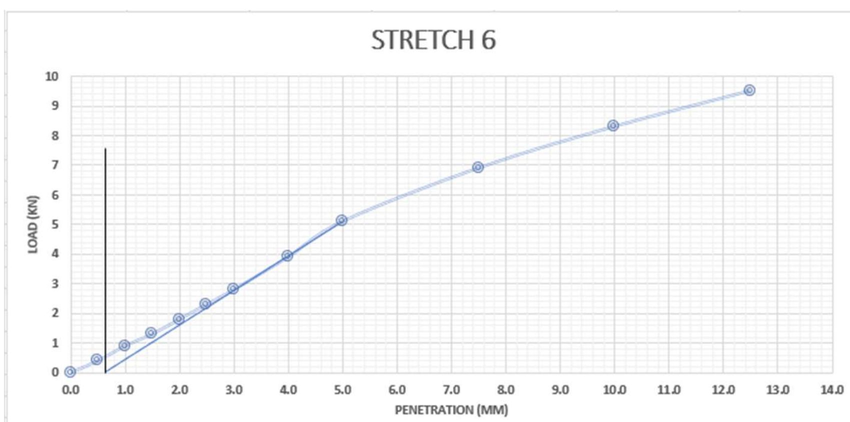


Fig: 9.13 CBR test curve for stretch 6

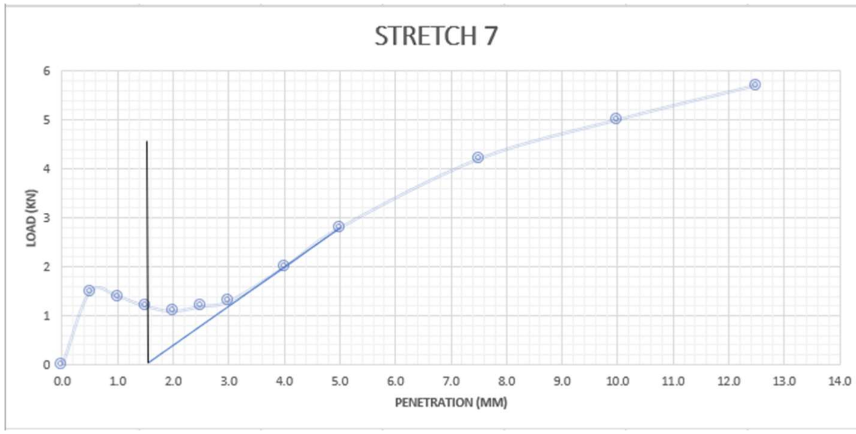


Fig: 9.14 CBR test curve for stretch 7

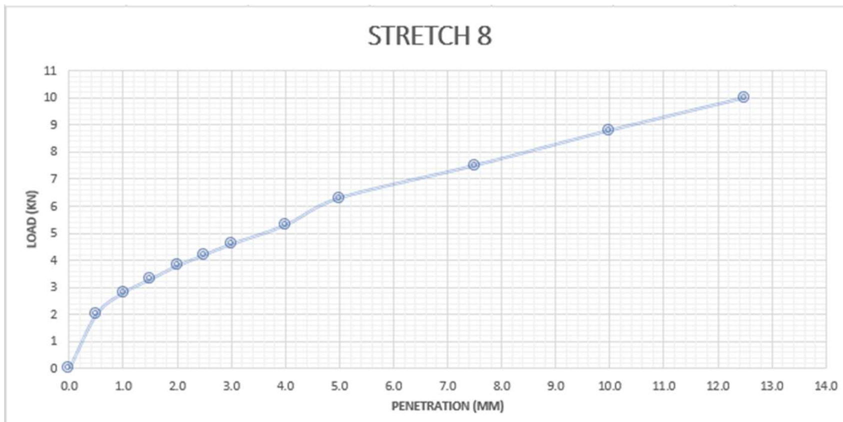


Fig: 9.15 CBR test curve for stretch 8

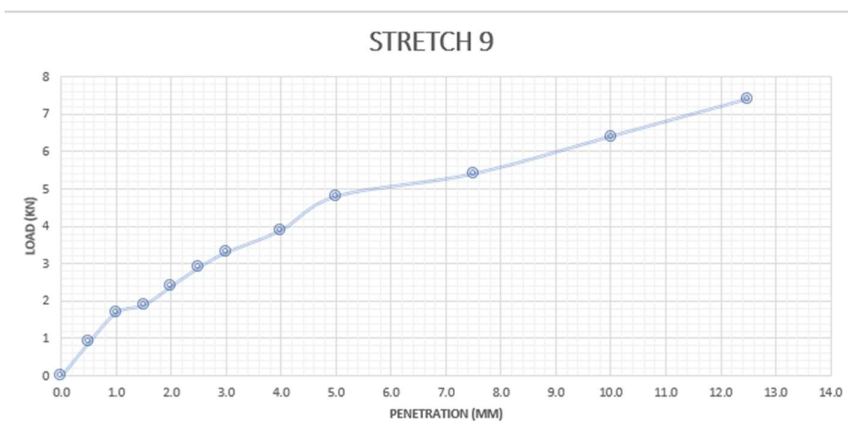


Fig: 9.16 CBR test curve for stretch 9

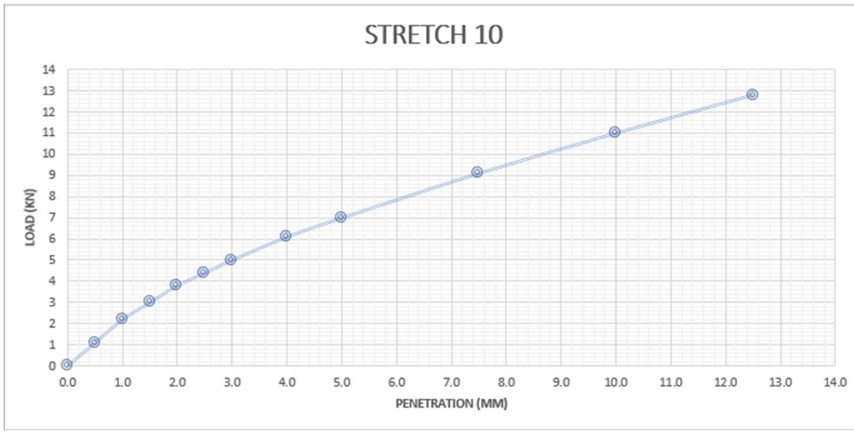


Fig: 9.17 CBR test curve for stretch 10

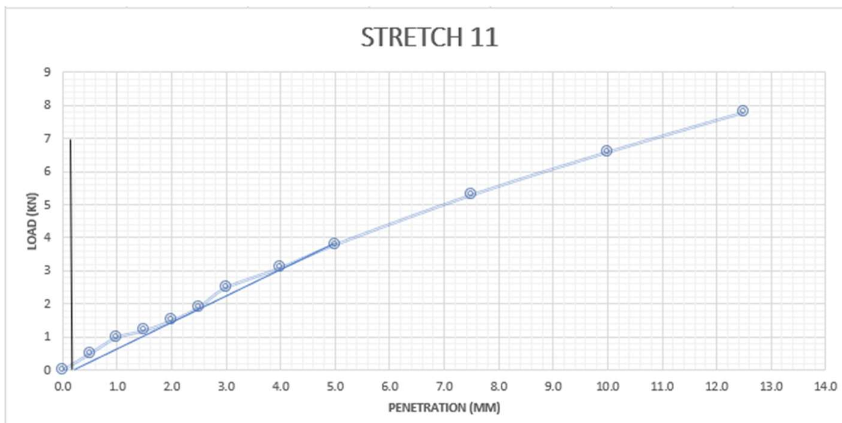


Fig: 9.18 CBR test curve for stretch 11

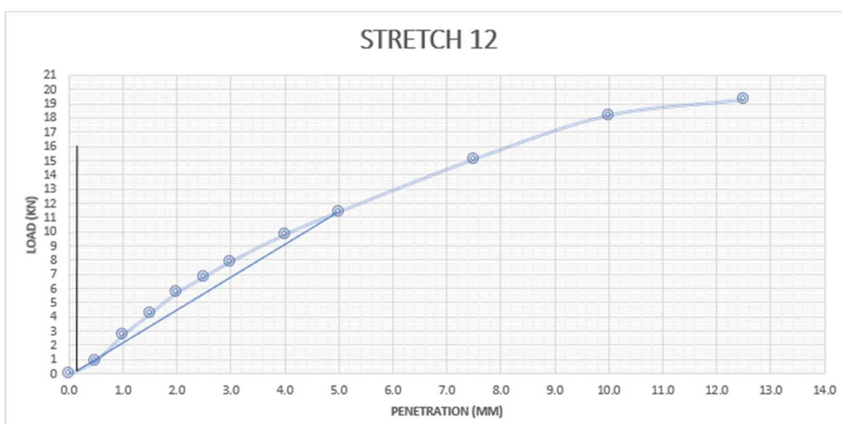


Fig: 9.19 CBR test curve for stretch 12

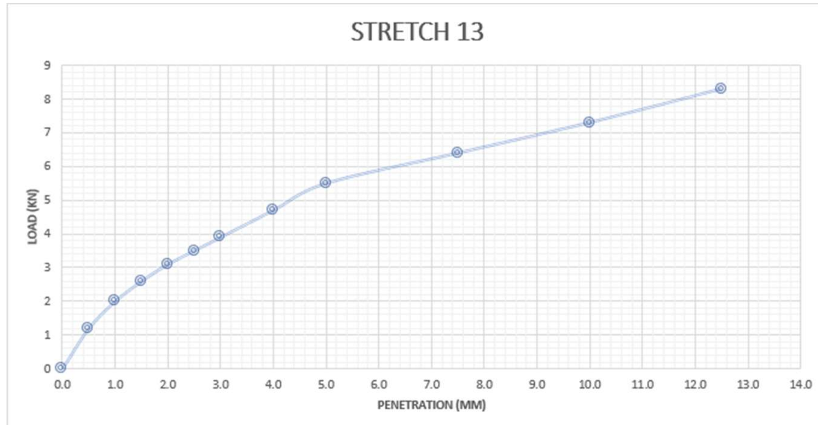


Fig: 9.20 CBR test curve for stretch 13

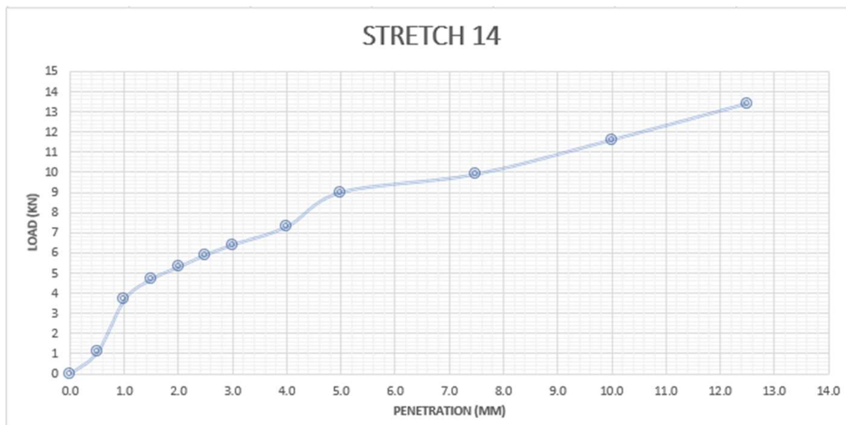


Fig: 9.21 CBR test curve for stretch 14

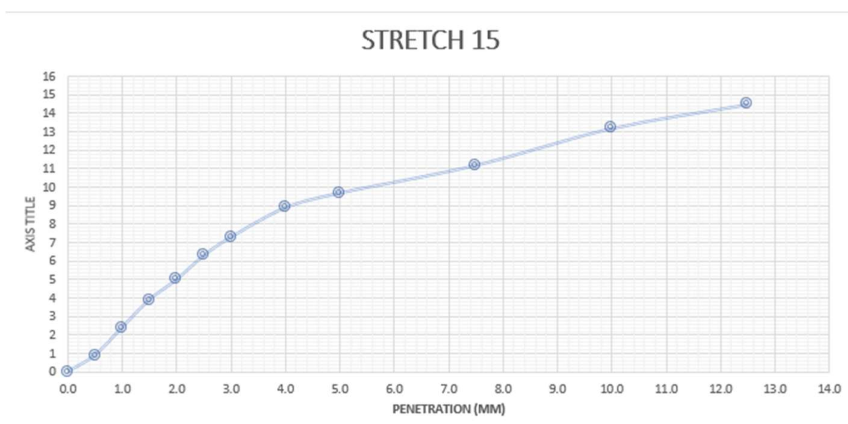


Fig: 9.22 CBR test curve for stretch 15

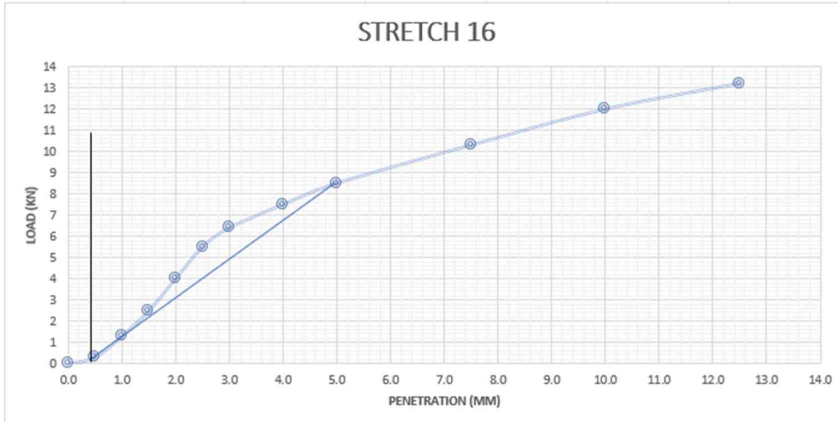


Fig: 9.23 CBR test curve for stretch 16

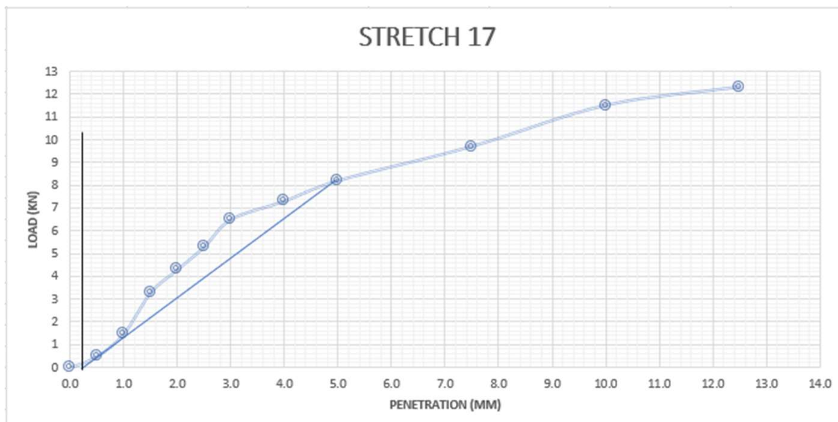


Fig: 9.24 CBR test curve for stretch 17

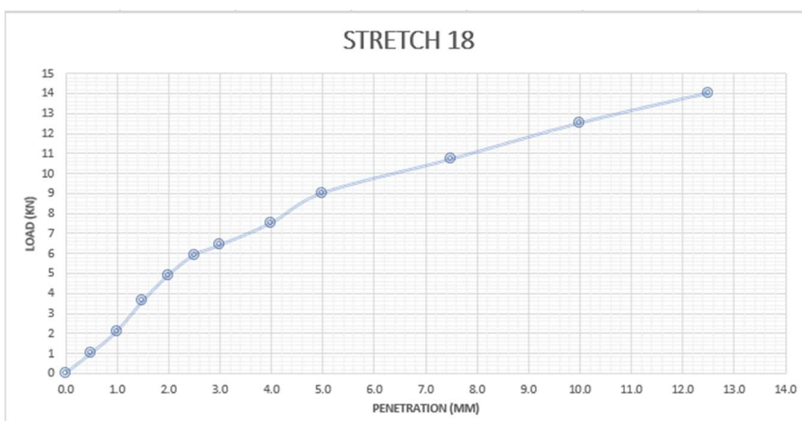


Fig: 9.25 CBR test curve for stretch 18

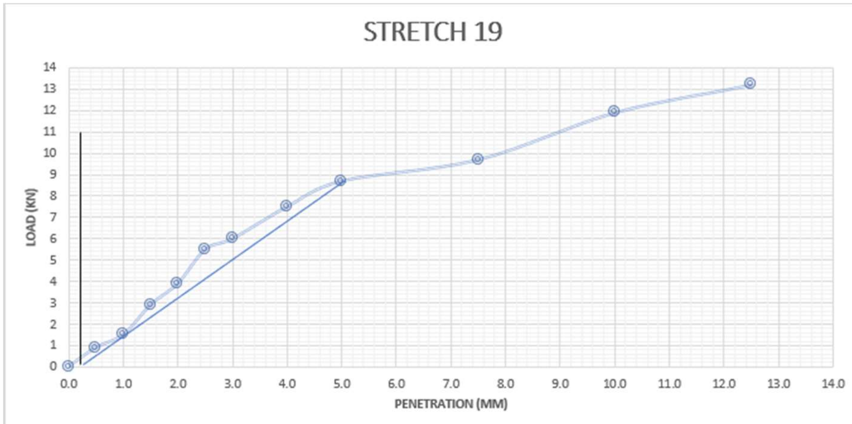


Fig: 9.26 CBR test curve for stretch 19

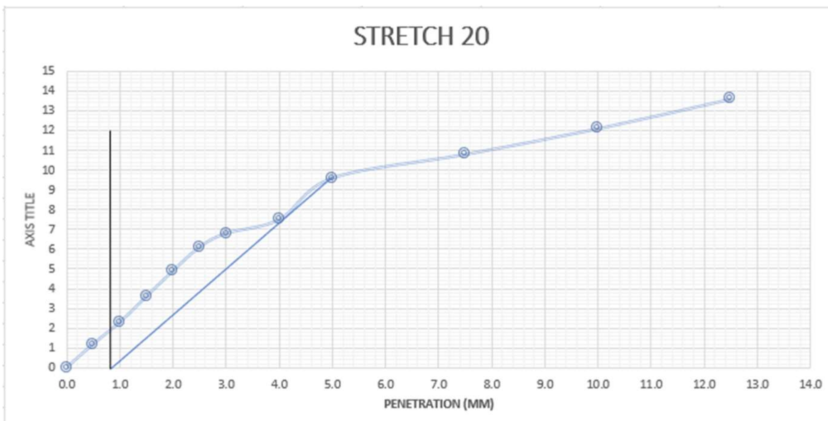


Fig: 9.27 CBR test curve for stretch 20

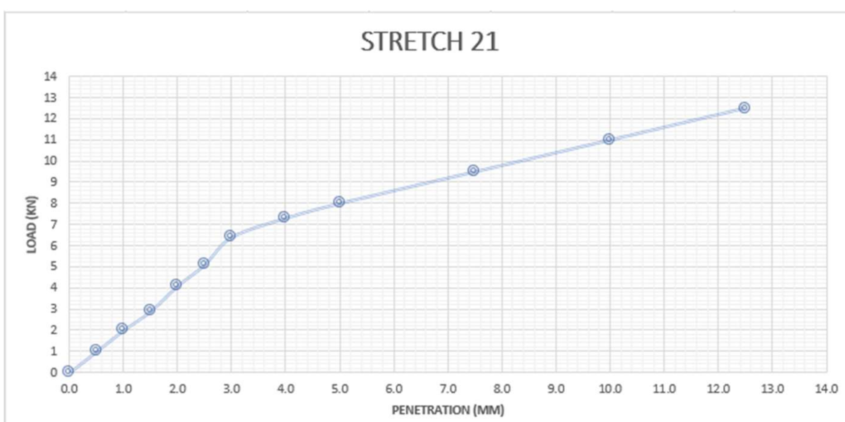


Fig: 9.28 CBR test curve for stretch 21

APPENDIX C

C.1 ALGORITHM FOR FUZZY TOPSIS APPROACH – WITH OUTPUTS

The algorithm below is a step-by-step execution of the Fuzzy TOPSIS method, which is written in python programming language.

```
#Importing Necessary Libraries.
import pandas as pd
import numpy as np
import math
from collections import Counter
from sympy import Point, Line, Segment
import matplotlib.pyplot as plt
from matplotlib.patches import Polygon
%matplotlib inline
from decimal import Decimal, getcontext


#Importing raw data and normalizing it.
Normalized_PCdata = pd.read_excel("Defect count per
stretch.xlsx")
Normalized_PCdata_1 = Normalized_PCdata.set_index("STRETCH NO.")
print("Raw Input Data: \n", Normalized_PCdata_1)
print("\n")

max_values_cols = Normalized_PCdata_1.max()
max_values_cols = list(max_values_cols)

max_values_row = pd.Series(max_values_cols, index =
Normalized_PCdata_1.columns)
Normalized_PCdata_2 = Normalized_PCdata_1.append(max_values_row,
ignore_index=True)

Norm_decision = dict()
print("Enter for the given criterias - MAX given High Priority or
MIN given High Priority.")
print("If Max given High Priority, Enter 1.")
print("If Min given High Priority, Enter 0.\n")
for i in Normalized_PCdata_2.columns:
    print("Criteria: ", i)
    pos_neg = int(input("Max/Min: "))
    Norm_decision[i] = pos_neg
Norm_decision_list = list(Norm_decision.values())
```

```
for i in range(0, len(Normalized_PCdata_2.columns)):
    m = Normalized_PCdata_2.iloc[-1,i]
    n = Norm_decision_list.pop(0)
    for j in range(0, len(Normalized_PCdata_2)-1):
        if n == 1:
            Normalized_PCdata_1.iloc[j,i] =
math.floor((Normalized_PCdata_1.iloc[j,i] * 100) / m)
        else:
            Normalized_PCdata_1.iloc[j,i] = math.floor(100-
((Normalized_PCdata_1.iloc[j,i] * 100) / m))
print("\n")
print("Normalized Data: \n",Normalized_PCdata_1)
Normalized_PCdata_1.to_excel("Normalized Data Sheet.xlsx")
```

```
Raw Input Data:
              A_L  A_M  A_H  B_L  B_M  B_H  L_L  L_M  L_H  T_L
...  P_M  P_H  \
STRETCH NO.
...
S1              2    1    0    4    3    1    8    4    3    6
...    1    0
S2              3    2    1    8    4    2   11    4    2   10
...    1    0
S3              4    3    2    3    1    1    9    2    1    7
...    1    0
S4              1    1    1    6    3    2   10    5    2    9
...    1    0
S5              2    3    2    3    2    1   10    4    1    8
...    1    1
S6              5    1    2    4    3    1   13    5    3    7
...    0    0
S7              3    2    1    5    2    0   11    6    2    6
...    2    0
S8              4    1    0    3    2    1   12    5    3   14
...    3    2
S9              2    2    1    5    2    1   10    5    3    5
...    1    1
S10             1    1    0    7    4    3   14    5    0   12
...    1    1
S11             0    0    0    2    1    0    9    1    0    7
...    1    0
S12             0    1    1    9    2    3   18   11    0   22
...    2    0
S13             1    1    1   10    5    3   14    6    2   18
...    1    0
S14             0    0    0    3    2    0   14    3    0    8
...    0    0
```

```
S15           0    2    4    4    7    2    8    8    8    4
...    1    0
S16           0    0    0    0    0    0    0    0    0    0
...    0    0
S17           0    0    0    0    0    0    0    0    0    0
...    0    0
S18           0    0    0    0    0    0    0    0    0    0
...    0    0
S19           0    0    0    0    0    0    0    0    0    0
...    0    0
S20           0    0    0    0    0    0    0    0    0    0
...    0    1
S21           0    0    0    0    0    0    0    0    0    0
...    0    0

             M_L  M_M  M_H    POP  VOL   FS  NTOWN        CBR
STRETCH NO.
S1             2    1    1   9000   98  135   1.20  19.951338
S2             8    0    0  18057  332  155   1.10  18.491484
S3             7    3    1  24137  484  120   1.50  23.357664
S4             9    4    1  24194  311  250   1.00  30.656934
S5             7    3    1  15557  696  215   1.40  43.309002
S6             6    2    2  22598  724  125   0.70  24.817518
S7             5    1    1  23598  724  190   0.35  13.625304
S8             7    5    2  23598  943  110   0.40  30.656934
S9             0    0    0  22598  887  175   0.55  23.357664
S10           16    5    2  24637  112   70   0.55  34.063260
S11           10    1    3  23137  720  165   0.35  18.491484
S12           14    7    5  24437  912  150   0.60  55.474453
S13            7    3    0  24437  728  100   0.40  26.763990
S14            8    6   10   7880  412  120   0.50  43.795620
S15            3    3    1  18437  172  120   1.50  47.201946
S16            0    0    0   2010  153  160   1.40  41.362530
S17            0    0    0   2800  155   80   0.70  39.902676
S18            0    0    0  34567  166  160   0.50  43.795620
S19            0    0    0   5330   97  200   0.30  42.335766
S20            0    0    0  20154   63   75   0.95  46.715328
S21            0    0    0   6780   67  165   0.20  38.929440

[21 rows x 41 columns]


Enter for the given criterias - MAX given High Priority or MIN
given High Priority.
If Max given High Priority, Enter 1.
If Min given High Priority, Enter 0.
```

Criteria: A_L

Max/Min: 1

Criteria: A_M

Max/Min: 1

Criteria: A_H

Max/Min: 1

Criteria: B_L

Max/Min: 1

Criteria: B_M

Max/Min: 1

Criteria: B_H

Max/Min: 1

Criteria: PH_L

Max/Min: 1

Criteria: PH_M

Max/Min: 1

Criteria: PH_H

Max/Min: 1

Criteria: R_L

Max/Min: 1

Criteria: R_M

Max/Min: 1

Criteria: R_H

Max/Min: 1

Criteria: S_L

Max/Min: 1

Criteria: S_M

Max/Min: 1

Criteria: L_L

Max/Min: 1

Criteria: L_M

Max/Min: 1

Criteria: L_H

Max/Min: 1

Criteria: T_L

Max/Min: 1

Criteria: T_M

Max/Min: 1

Criteria: T_H

Max/Min: 1

Criteria: RA_L

Max/Min: 1

Criteria: RA_M

Max/Min: 1

Criteria: RA_H

Max/Min: 1

Criteria: S_H

Max/Min: 1

Criteria: E_L

Max/Min: 1

Criteria: E_M

Max/Min: 1

Criteria: E_H

Max/Min: 1

Criteria: D_L

Max/Min: 1

Criteria: D_M

Max/Min: 1

Criteria: D_H

Max/Min: 1

Criteria: P_L

Max/Min: 1

Criteria: P_M

Max/Min: 1

Criteria: P_H

Max/Min: 1

Criteria: M_L

Max/Min: 1

Criteria: M_M

Max/Min: 1

Criteria: M_H

Max/Min: 1

Criteria: POP

Max/Min: 1

Criteria: VOL

Max/Min: 1

Criteria: FS

Max/Min: 1

Criteria: NTOWN

Max/Min: 0

Criteria: CBR

Max/Min: 0

```
Normalized Data:
              A_L  A_M  A_H  B_L  B_M  B_H  L_L  L_M  L_H  T_L
...  P_M  P_H  \
STRETCH NO.
...
S1             40   33    0   40   42   33   44   36   37   27
...   33    0
S2             60   66   25   80   57   66   61   36   25   45
...   33    0
S3             80  100   50   30   14   33   50   18   12   31
...   33    0
S4             20   33   25   60   42   66   55   45   25   40
...   33    0
S5             40  100   50   30   28   33   55   36   12   36
...   33   50
S6            100   33   50   40   42   33   72   45   37   31
...    0    0
S7             60   66   25   50   28    0   61   54   25   27
...   66    0
S8             80   33    0   30   28   33   66   45   37   63
...  100  100
S9             40   66   25   50   28   33   55   45   37   22
...   33   50
S10            20   33    0   70   57  100   77   45    0   54
...   33   50
```

```
S11            0    0    0   20   14    0   50    9    0   31
...    33    0
S12            0   33   25   90   28  100  100  100    0  100
...    66    0
S13           20   33   25  100   71  100   77   54   25   81
...    33    0
S14            0    0    0   30   28    0   77   27    0   36
...     0    0
S15            0   66  100   40  100   66   44   72  100   18
...    33    0
S16            0    0    0    0    0    0    0    0    0    0
...     0    0
S17            0    0    0    0    0    0    0    0    0    0
...     0    0
S18            0    0    0    0    0    0    0    0    0    0
...     0    0
S19            0    0    0    0    0    0    0    0    0    0
...     0    0
S20            0    0    0    0    0    0    0    0    0    0
...     0   50
S21            0    0    0    0    0    0    0    0    0    0
...     0    0

             M_L  M_M  M_H  POP  VOL   FS  NTOWN   CBR
STRETCH NO.
S1            12   14   10   26   10   54   20.0  64.0
S2            50    0    0   52   35   62   26.0  66.0
S3            43   42   10   69   51   48    0.0  57.0
S4            56   57   10   69   32  100   33.0  44.0
S5            43   42   10   45   73   86    6.0  21.0
S6            37   28   20   65   76   50   53.0  55.0
S7            31   14   10   68   76   76   76.0  75.0
S8            43   71   20   68  100   44   73.0  44.0
S9             0    0    0   65   94   70   63.0  57.0
S10          100   71   20   71   11   28   63.0  38.0
S11           62   14   30   66   76   66   76.0  66.0
S12           87  100   50   70   96   60   60.0   0.0
S13           43   42    0   70   77   40   73.0  51.0
S14           50   85  100   22   43   48   66.0  21.0
S15           18   42   10   53   18   48    0.0  14.0
S16            0    0    0    5   16   64    6.0  25.0
S17            0    0    0    8   16   32   53.0  28.0
S18            0    0    0  100   17   64   66.0  21.0
S19            0    0    0   15   10   80   80.0  23.0
S20            0    0    0   58    6   30   36.0  15.0
S21            0    0    0   19    7   66   86.0  29.0

[21 rows x 41 columns]
```

```
#Creating Rating Matrix for Normalized Values.
Rating_Matrix = []
for i in range(0, len(Normalized_PCdata_1.columns)):
    for j in range(0, len(Normalized_PCdata_1)):
        if Normalized_PCdata_1.iloc[j,i] >= 0 and
Normalized_PCdata_1.iloc[j,i] <= 10:
            Rating_Matrix.append(1)
        elif Normalized_PCdata_1.iloc[j,i] >= 11 and
Normalized_PCdata_1.iloc[j,i] <= 20:
            Rating_Matrix.append(2)
        elif Normalized_PCdata_1.iloc[j,i] >= 21 and
Normalized_PCdata_1.iloc[j,i] <= 30:
            Rating_Matrix.append(3)
        elif Normalized_PCdata_1.iloc[j,i] >= 31 and
Normalized_PCdata_1.iloc[j,i] <= 40:
            Rating_Matrix.append(4)
        elif Normalized_PCdata_1.iloc[j,i] >= 41 and
Normalized_PCdata_1.iloc[j,i] <= 50:
            Rating_Matrix.append(5)
        elif Normalized_PCdata_1.iloc[j,i] >= 51 and
Normalized_PCdata_1.iloc[j,i] <= 60:
            Rating_Matrix.append(6)
        elif Normalized_PCdata_1.iloc[j,i] >= 61 and
Normalized_PCdata_1.iloc[j,i] <= 70:
            Rating_Matrix.append(7)
        elif Normalized_PCdata_1.iloc[j,i] >= 71 and
Normalized_PCdata_1.iloc[j,i] <= 80:
            Rating_Matrix.append(8)
        elif Normalized_PCdata_1.iloc[j,i] >= 81 and
Normalized_PCdata_1.iloc[j,i] <= 90:
            Rating_Matrix.append(9)
        elif Normalized_PCdata_1.iloc[j,i] >= 91 and
Normalized_PCdata_1.iloc[j,i] <= 100:
            Rating_Matrix.append(10)

cols = len(Normalized_PCdata_1)
rows = len(Normalized_PCdata_1.columns)
Rating_Matrix = np.array(Rating_Matrix)
Rating_Matrix = Rating_Matrix.reshape(rows,cols)
Rating_Matrix = Rating_Matrix.transpose()
print(Rating_Matrix)
```

```
[[ 4  4  1  4  5  4  5  4  4  3  8  5  2  3  1  5  4 10  1  1  1
2  1  1
   6  5  5  7  6  5  2  4  1  2  2  1  3  1  6  2  7]
 [ 6  7  3  8  6  7  7  4  3  5  3 10  5  5 10  5  4  5  1  1  1
4  4  1
   9  3  5  5  4 10  3  4  1  5  1  1  6  4  7  3  7]
```

```
 [ 8 10  5  3  2  4  5  2  2  4  3  5  4  3 10  7  3  3  1  1  1
2  1  1
   5  5  1  4  2  5  2  4  1  5  5  1  7  6  5  1  6]
 [ 2  4  3  6  5  7  6  5  3  4  4 10  7  5 10  6  5  7  1  1  1
4 10  1
   9  5  5  5  6  5  4  4  1  6  6  1  7  4 10  4  5]
 [ 4 10  5  3  3  4  6  4  2  4  3  5  5  5 10  5  4  8  6 10 10
6  7  5
  10  8 10  7  8 10  2  4  5  5  5  1  5  8  9  1  3]
 [10  4  5  4  5  4  8  5  4  4  2 10  4  3 10  6  5  5  1  1  1
8  7 10
   8  3  5  3 10  5  3  1  1  4  3  2  7  8  5  6  6]
 [ 6  7  3  5  3  1  7  6  3  3  4 10  5  3 10  6  4  2  1  1  1
10 10  5
   6  5 10  4  8 10  3  7  1  4  2  1  7  8  8  8  8]
 [ 8  4  1  3  3  4  7  5  4  7  7  1  4  3 10  6  5  3  1  1  1
4  4  5
   8  8  5  9  6  5  5 10 10  5  8  2  7 10  5  8  5]
 [ 4  7  3  5  3  4  6  5  4  3  4  5  5  5 10  8  4  2  1  1  1
2  4  1
   5  5  5  7  6  5  3  4  5  1  1  1  7 10  7  7  6]
 [ 2  4  1  7  6 10  8  5  1  6  4  1 10  3 10  4  9  8  3  4  4
2  1  1
   6 10  1 10  6  5 10  4  5 10  8  2  8  2  3  7  4]
 [ 1  1  1  2  2  1  5  1  1  4  4  5  7  1  1  3  2  1  2  2  4
1  1  1
   1  1  1  3  2  1  3  4  1  7  2  3  7  8  7  8  7]
 [ 1  4  3  9  3 10 10 10  1 10 10  1  7  3 10  8  6  9 10  8  1
10  1  1
   5  3  1  8 10  1  3  7  1  9 10  5  7 10  6  6  1]
 [ 2  4  3 10  8 10  8  6  3  9 10  1  5  5  1 10  5  1  1  2  1
1  4  1
   3  3  1  2  2  1  2  4  1  5  5  1  7  8  4  8  6]
 [ 1  1  1  3  3  1  8  3  1  4  5  5  5  1  1  4 10  8  1  4  1
1 10  1
   2  1  1  8  6  1  3  1  1  5  9 10  3  5  5  7  3]
 [ 1  7 10  4 10  7  5  8 10  2  9  1  5 10  1  8  6  5  3  8  1
2  4  1
   6  3  1  2  8  1  2  4  1  2  5  1  6  2  5  1  2]
 [ 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  4  1
1  1  1
   1  1  1  1  1  1  1  1  1  1  1  1  1  2  7  1  3]
 [ 1  1  1  1  1  1  1  1  1  1  1  1  2  1  1  1  1  1  1  2  1
1  1  1
   1  1  1  1  1  1  1  1  1  1  1  1  1  2  4  6  3]
 [ 1  1  1  1  1  1  1  1  1  1  1  1  1  3  1  1  1  1  2  1  1
1  1  1
   1  1  1  2  1  1  4  1  1  1  1  1 10  2  7  7  3]
 [ 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  2  1
1  1  1
```

```
   1  1  1  1  1  1  1  1  1  1  1  1  2  1  8  8  3]
  [ 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  6  1
1  1  1
   1  1  1  1  1  1  1  1  5  1  1  1  6  1  3  4  2]
  [ 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
1  1  1
   1  1  1  1  4  1  2  1  1  1  1  1  2  1  7  9  3]]
```

```
#Importing Experts Opinion and obtaining Fuzzy weights for
various distress parameters.
Expert_survey = pd.read_excel("MCDM Expert Survey_1.xlsx")
Expert_survey_1 = Expert_survey.set_index("CRITERIA")

Triangular_Fuzzy_nos_1 = {"N": 0, "L": 0, "M": 0.3, "H": 0.7,
"VH": 0.9}
Triangular_Fuzzy_nos_2 = {"N": 0, "L": 0.1, "M": 0.5, "H": 0.9,
"VH": 1}
Triangular_Fuzzy_nos_3 = {"N": 0.1, "L": 0.3, "M": 0.7, "H": 1,
"VH": 1}

Expert_survey_counts = pd.get_dummies(Expert_survey_1,
prefix_sep='', prefix='').sum(axis=1, level=0)
Expert_survey_counts =
Expert_survey_counts.to_dict(orient='records')

Fuzzy_wts_1 = []
inp1 = Counter(Triangular_Fuzzy_nos_1)
for i in Expert_survey_counts:
    inp2 = Counter(i)
    res = Counter({key : inp1[key] * inp2[key] for key in inp1})
    Fuzzy_wts_1.append(res)

Fuzzy_wts_2 = []
inp1 = Counter(Triangular_Fuzzy_nos_2)
for i in Expert_survey_counts:
    inp2 = Counter(i)
    res = Counter({key : inp1[key] * inp2[key] for key in inp1})
    Fuzzy_wts_2.append(res)

Fuzzy_wts_3 = []
inp1 = Counter(Triangular_Fuzzy_nos_3)
for i in Expert_survey_counts:
    inp2 = Counter(i)
    res = Counter({key : inp1[key] * inp2[key] for key in inp1})
    Fuzzy_wts_3.append(res)

Fuzzy_wts = []
```

```
for i in range(0,len(Expert_survey_1)):
    weight_vals = []
    col1 = round(sum(list(Fuzzy_wts_1[i].values())),3)
    col2 = round(sum(list(Fuzzy_wts_2[i].values())),3)
    col3 = round(sum(list(Fuzzy_wts_3[i].values())),3)
    weight_vals.extend([col1,col2,col3])
    Fuzzy_wts.append(weight_vals)
Fuzzy_wts = [[round(x/15, 3) for x in inner_list] for inner_list
in Fuzzy_wts]

Fuzzy_wts = pd.DataFrame(Fuzzy_wts)
Fuzzy_wts["Criteria"] = Expert_survey["CRITERIA"]
Fuzzy_wts = Fuzzy_wts.set_index("Criteria")
print(Fuzzy_wts)
Fuzzy_wts.to_excel("Fuzzy wts.xlsx")
```

```
              0      1      2
Criteria
A_L       0.000  0.033  0.167
A_M       0.120  0.253  0.447
A_H       0.460  0.640  0.787
B_L       0.000  0.033  0.167
B_M       0.120  0.253  0.447
B_H       0.460  0.640  0.787
L_L       0.000  0.033  0.167
L_M       0.120  0.253  0.447
L_H       0.460  0.640  0.787
T_L       0.000  0.033  0.167
T_M       0.120  0.253  0.447
T_H       0.460  0.640  0.787
PH_L      0.180  0.333  0.527
PH_M      0.547  0.740  0.873
PH_H      0.847  0.973  1.000
RA_L      0.040  0.107  0.260
RA_M      0.193  0.327  0.500
RA_H      0.580  0.760  0.880
R_L       0.020  0.087  0.247
R_M       0.227  0.393  0.587
R_H       0.607  0.787  0.900
S_L       0.000  0.033  0.167
S_M       0.120  0.253  0.447
S_H       0.460  0.640  0.787
E_L       0.020  0.080  0.233
E_M       0.240  0.387  0.560
E_H       0.627  0.787  0.880
D_L       0.020  0.087  0.247
D_M       0.227  0.393  0.587
D_H       0.607  0.787  0.900
P_L       0.060  0.133  0.287
P_M       0.253  0.420  0.607
```

```
P_H       0.580  0.760  0.880
M_L       0.060  0.133  0.287
M_M       0.253  0.420  0.607
M_H       0.580  0.760  0.880
POP       0.353  0.473  0.600
VOL       0.347  0.500  0.653
FS        0.260  0.380  0.527
NTOWN     0.293  0.420  0.567
CBR       0.393  0.533  0.667
```

```
#Obtaining Fuzzy Evaluation Values for all stretches.
Rating_Matrix = np.array(Rating_Matrix)
Fuzzy_wts = np.array(Fuzzy_wts)
Fuzzy_Eval_vals = np.dot(Rating_Matrix, Fuzzy_wts)

Fuzzy_Eval_vals = pd.DataFrame(Fuzzy_Eval_vals)
Fuzzy_Eval_vals.columns = ["l","m","n"]
print("Total no of stretches: " + str(len(Fuzzy_Eval_vals)))

row_index = ['A' + str(i) for i in range(1,
len(Fuzzy_Eval_vals)+1)]
Fuzzy_Eval_vals["Stretch No."] = row_index
Fuzzy_Eval_vals_1 = Fuzzy_Eval_vals.set_index("Stretch No.")
print(Fuzzy_Eval_vals_1)
Fuzzy_Eval_vals_1.to_excel("Fuzzy Evaluation Values.xlsx")
```

```
Total no of stretches: 21
                  l        m        n
Stretch No.
A1           38.227   56.923   79.469
A2           53.725   77.293  105.182
A3           42.121   60.970   83.581
A4           55.882   81.619  112.025
A5           70.023  100.834  135.451
A6           56.945   82.369  112.446
A7           59.985   86.898  118.635
A8           57.746   84.475  116.671
A9           51.918   74.878  101.543
A10          54.645   80.873  113.169
A11          30.490   45.393   63.605
A12          54.049   82.085  118.637
A13          36.982   57.624   84.910
A14          37.089   56.646   80.859
A15          44.776   69.073   98.190
A16          14.688   21.615   29.658
A17          15.099   22.122   30.265
A18          20.256   29.266   39.800
A19          16.551   24.122   32.927
```

```
A20           18.326   26.513   35.625
A21           17.098   25.081   34.428
```

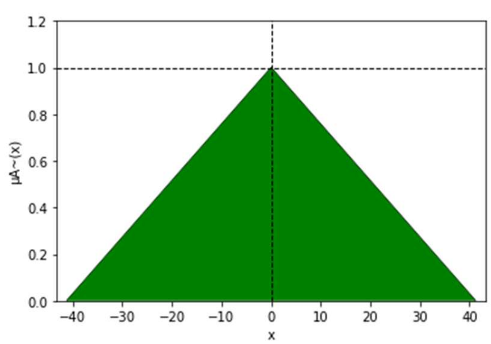
```
#Triangular Fuzzy Numbers Calculation.
Triangular_FuzzyNums = pd.DataFrame(columns = ['l','m','n'])
for i in range(0,len(Fuzzy_Eval_vals)):
    for j in range(0,len(Fuzzy_Eval_vals)):
        arr1 = Fuzzy_Eval_vals.loc[i]
        arr2 = Fuzzy_Eval_vals.loc[j]
        diff = [arr1[0]-arr2[2],arr1[1]-arr2[1],arr1[2]-arr2[0]]
        stretch_diff = pd.Series(diff, index =
Triangular_FuzzyNums.columns)
        Triangular_FuzzyNums =
Triangular_FuzzyNums.append(stretch_diff, ignore_index=True)

row_diff = ['p' + str(i) + "-" + "p" + str(j) for i in range(1,
len(Fuzzy_Eval_vals)+1) for j in range(1,
len(Fuzzy_Eval_vals)+1)]
Triangular_FuzzyNums["p~i-p~j"] = row_diff
Triangular_FuzzyNums_1 = Triangular_FuzzyNums.set_index("p~i-
p~j")
print(Triangular_FuzzyNums_1)
Triangular_FuzzyNums_1.to_excel("Triangular Fuzzy Numbers.xlsx")
```

```
              l       m       n
p~i-p~j
p1-p1   -41.242   0.000  41.242
p1-p2   -66.955 -20.370  25.744
p1-p3   -45.354  -4.047  37.348
p1-p4   -73.798 -24.696  23.587
p1-p5   -97.224 -43.911   9.446
...         ...     ...     ...
p21-p17 -13.167   2.959  19.329
p21-p18 -22.702  -4.185  14.172
p21-p19 -15.829   0.959  17.877
p21-p20 -18.527  -1.432  16.102
p21-p21 -17.330   0.000  17.330

[441 rows x 3 columns]
```

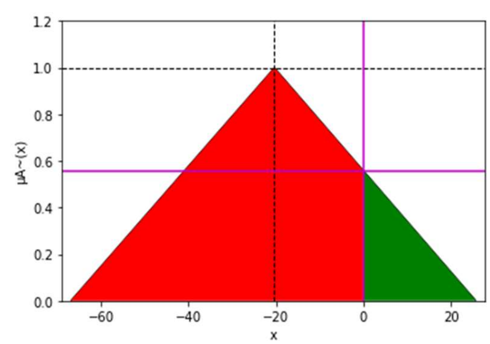
```
#Graphs for all Triangular Fuzzy Numbers.
Intersection_pnts = []
for i in range(0,len(Triangular_FuzzyNums)):
    l = Triangular_FuzzyNums.iloc[i,0]
    m = Triangular_FuzzyNums.iloc[i,1]
```

```
    n = Triangular_FuzzyNums.iloc[i,2]

    plt.xlabel("x")
    plt.ylabel("μA~(x)")
    plt.ylim(0,1.2)
    plt.xlim(l - 2, n + 2)

    pnts = np.array([[l,0], [m,1], [n,0]])
    f_triangle = Polygon(pnts, closed=False, edgecolor = "black",
facecolor = "white")
    ax = plt.gca()
    ax.add_patch(f_triangle)

    if l < 0 and m > 0:
        p1, p2, p3, p4 = Point(0, 0), Point(0, 1), Point(l, 0),
Point(m, 1)
        l1 = Line(p1, p2)
        s1 = Segment(p3, p4)
        showIntersection = l1.intersection(s1)
        print(showIntersection)
        Intersection_pnts.append(showIntersection)
    elif m < 0 and n > 0:
        p1, p2, p3, p4 = Point(0, 0), Point(0, 1), Point(n, 0),
Point(m, 1)
        l1 = Line(p1, p2)
        s1 = Segment(p3, p4)
        showIntersection = l1.intersection(s1)
        print(showIntersection)
        Intersection_pnts.append(showIntersection)
    else:
        Intersection_pnts.append([[0,0]])

    plt.axvline(x = m, color = "black", linestyle = "--",
linewidth = 1)

    if l < 0 and m > 0 and n > 0:
        plt.axhline(y = 1, color = "black", linestyle = "--",
linewidth = 1)
        plt.axvline(x = 0, color = "m")
        plt.axhline(y = showIntersection[0][1], color = "m")
    elif m < 0 and n > 0:
        plt.axhline(y = 1, color = "black", linestyle = "--",
linewidth = 1)
        plt.axvline(x = 0, color = "m")
        plt.axhline(y = showIntersection[0][1], color = "m")
    elif n < 0:
        plt.axhline(y = 1, color = "black", linestyle = "--",
linewidth = 1)
    else:
```

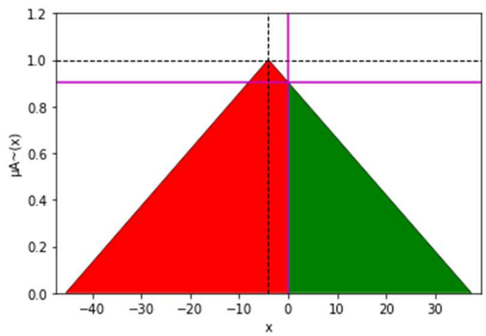
```
        plt.axhline(y = 1, color = "black", linestyle = "--",
linewidth = 1)

    plt.figure()

    if l < 0  and m > 0 and n > 0:
        ax.add_patch(Polygon([(0, 0), (0,
showIntersection[0][1]), (m, 1), (n, 0)],
                              closed=True, facecolor='green'))
        ax.add_patch(Polygon([(0, 0), (0,
showIntersection[0][1]), (l, 0)],
                              closed=True, facecolor='red'))
    elif m < 0 and n > 0:
        ax.add_patch(Polygon([(0, 0), (0,
showIntersection[0][1]), (n, 0)],
                              closed=True, facecolor='green'))
        ax.add_patch(Polygon([(l, 0), (m, 1), (0,
showIntersection[0][1]), (0, 0)],
                              closed=True, facecolor='red'))
    elif n < 0:
        ax.add_patch(Polygon([(l, 0), (m, 1), (n, 0)],
                              closed=True, facecolor='red'))
    else:
        ax.add_patch(Polygon([(l, 0), (m, 1), (n, 0)],
                              closed=True, facecolor='green'))


    plt.show()
```




```
<Figure size 432x288 with 0
Axes>

[Point2D(0, 12872/23057)]
```

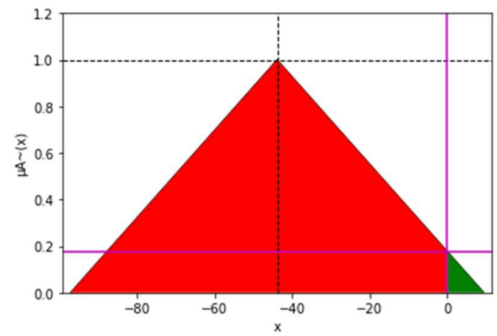



```
<Figure size 432x288 with 0
Axes>

[Point2D(0, 37348/41395)]
```



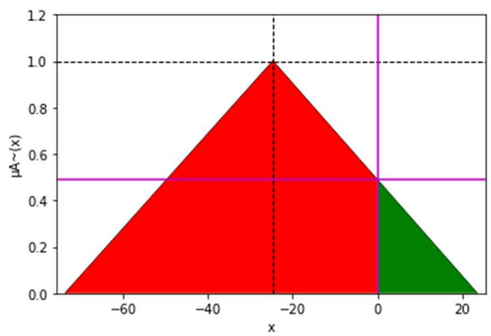
<Figure size 432x288 with 0 Axes>

[Point2D(0, 23587/48283)]



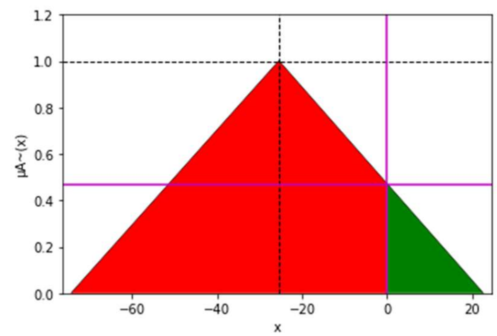
<Figure size 432x288 with 0 Axes>

[Point2D(0, 3754/7995)]



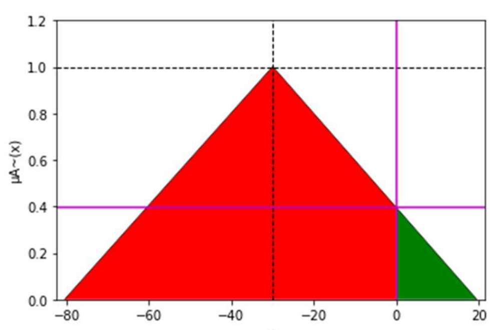
<Figure size 432x288 with 0 Axes>

[Point2D(0, 157433333333333/889283333333333)]



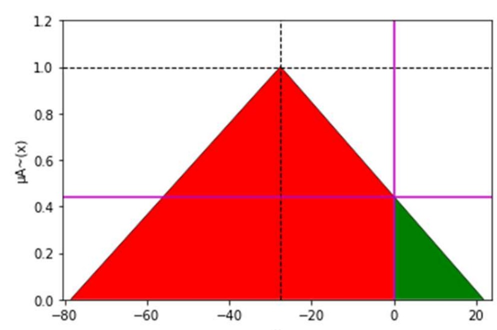
<Figure size 432x288 with 0 Axes>

[Point2D(0, 19484/49459)]



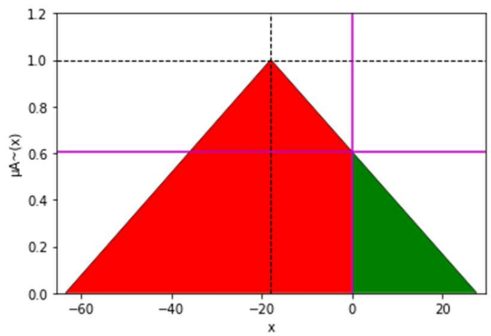
<Figure size 432x288 with 0 Axes>

[Point2D(0, 7241/16425)]



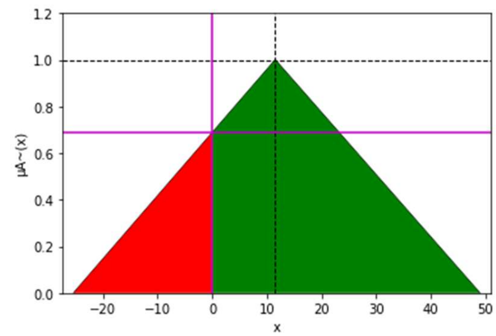
<Figure size 432x288 with 0 Axes>

[Point2D(0, 27551/45506)]



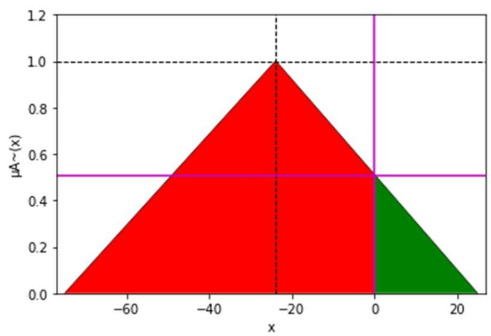
<Figure size 432x288 with 0 Axes>

[Point2D(0, 12412/24387)]



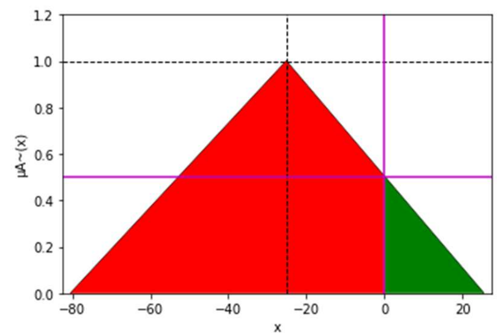
<Figure size 432x288 with 0 Axes>

[Point2D(0, 12710/25291)]

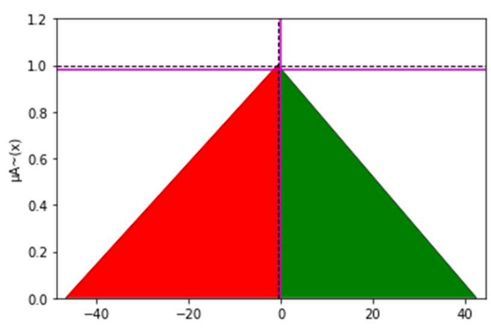


<Figure size 432x288 with 0 Axes>

[Point2D(0, 12689/18454)]

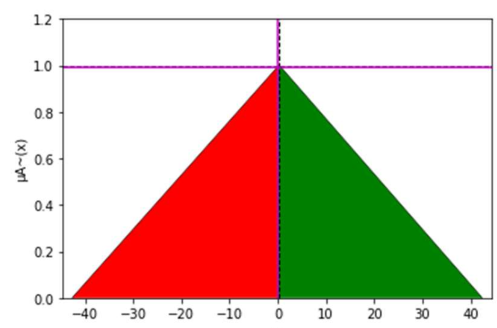


<Figure size 432x288 with 0 Axes>



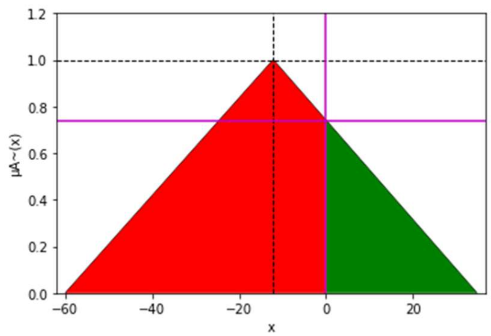
<Figure size 432x288 with 0 Axes>

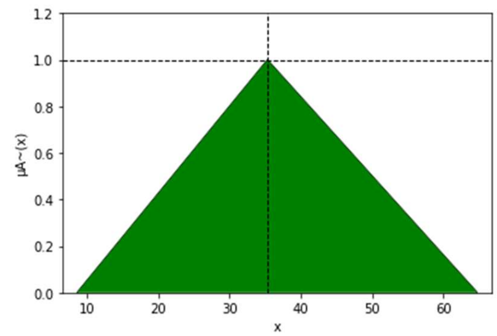
[Point2D(0, 5329000000000000/5363625000000001)]



<Figure size 432x288 with 0 Axes>

[Point2D(0, 34693/46843)]

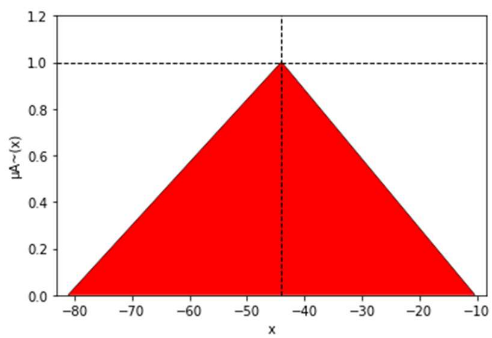


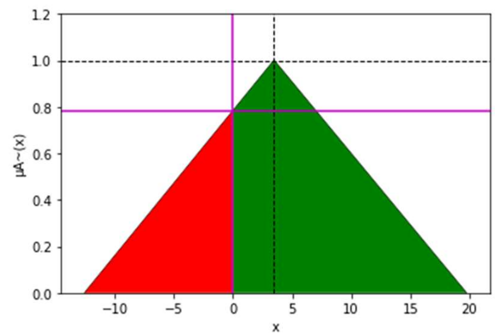



```
<Figure size 432x288 with 0
Axes>
```

```
<Figure size 432x288 with 0
Axes>
```

```
[Point2D(0, 1197/1466)]
```






```
<Figure size 432x288 with 0
Axes>
```

(The remaining graphs are displayed in similar manner; Total no of graphs – 441)

```
#Fuzzy Preference Relational Matrix[E].
Intersection = []
for i in range(0,len(Triangular_FuzzyNums_1)):
    Intersection.append(Intersection_pnts[i][0][1])

Triangular_FuzzyNums_1["Intersection height"] = Intersection

E = []

for i in range(0,len(Triangular_FuzzyNums)):
    l = Triangular_FuzzyNums_1.iloc[i,0]
    m = Triangular_FuzzyNums_1.iloc[i,1]
    n = Triangular_FuzzyNums_1.iloc[i,2]
    h = round(Triangular_FuzzyNums_1.iloc[i,3], 3)

    if l < 0 and m > 0 and n > 0:
        pos_Area = (1/2 * (n-m)) + (1/2 * (m) * (1-h)) + (h * m)
        neg_Area = 1/2 * (0-l) * h
    elif m < 0 and n > 0:
        pos_Area = 1/2 * (n) * h
        neg_Area = (1/2 * abs((m-l))) + (1/2 * (0-m) * (1-h)) +
((0-m) * h)
    elif n < 0:
        pos_Area = 0
        neg_Area = (1/2 * (n-m)) + (1/2 * (m-l))
    else:
        pos_Area = (1/2 * (n-m)) + (1/2 * (m-l))
        neg_Area = 0

    total_Area = pos_Area + neg_Area
    corr_e = round(pos_Area/total_Area, 3)
    E.append(corr_e)

E_array = np.array(E)
TriFuzzNo_sqrt = int(np.sqrt(len(Triangular_FuzzyNums)))
print("Matrix Size: " + str(TriFuzzNo_sqrt) + "*" +
str(TriFuzzNo_sqrt) + "\n")
E_2D_array = np.reshape(E_array, (TriFuzzNo_sqrt,
TriFuzzNo_sqrt))
np.fill_diagonal(E_2D_array, 0.5)
print(E_2D_array)
```

```
Matrix Size: 21*21

[[0.5 0.155 0.407 0.118 0.016 0.109 0.077 0.096 0.183 0.127 0.765
0.121
```

```
  0.469 0.502 0.272 1.0 1.0 0.999 1.0 1.0 1.0]
 [0.845 0.5 0.792 0.421 0.180 0.408 0.339 0.373 0.550 0.429 0.972
0.403
  0.808 0.838 0.642 1.0 1.0 1.0 1.0 1.0 1.0]
 [0.593 0.208 0.5 0.163 0.032 0.152 0.112 0.135 0.242 0.171 0.833
0.162
  0.556 0.591 0.338 1.0 1.0 1.0 1.0 1.0 1.0]
 [0.882 0.579 0.837 0.5 0.237 0.487 0.412 0.448 0.624 0.507 0.985
0.476
  0.847 0.875 0.702 1.0 1.0 1.0 1.0 1.0 1.0]
 [0.984 0.820 0.968 0.763 0.5 0.756 0.696 0.722 0.850 0.762 1.0
0.730
  0.966 0.980 0.889 1.0 1.0 1.0 1.0 1.0 1.0]
 [0.891 0.592 0.848 0.513 0.244 0.5 0.423 0.460 0.637 0.520 0.989
0.489
  0.856 0.884 0.714 1.0 1.0 1.0 1.0 1.0 1.0]
 [0.923 0.661 0.888 0.588 0.304 0.577 0.5 0.538 0.702 0.592 0.997
0.560
  0.892 0.917 0.768 1.0 1.0 1.0 1.0 1.0 1.0]
 [0.904 0.627 0.865 0.552 0.278 0.540 0.462 0.5 0.669 0.557 0.992
0.526
  0.872 0.898 0.739 1.0 1.0 1.0 1.0 1.0 1.0]
 [0.817 0.450 0.758 0.376 0.150 0.363 0.298 0.331 0.5 0.385 0.960
0.361
  0.778 0.810 0.601 1.0 1.0 1.0 1.0 1.0 1.0]
 [0.873 0.571 0.829 0.493 0.238 0.480 0.408 0.443 0.615 0.5 0.980
0.471
  0.839 0.867 0.694 1.0 1.0 1.0 1.0 1.0 1.0]
 [0.235 0.028 0.167 0.015 0.0 0.011 0.003 0.008 0.040 0.020 0.5
0.020
  0.225 0.242 0.096 1.0 1.0 0.935 0.995 0.978 0.987]
 [0.879 0.597 0.838 0.524 0.270 0.511 0.440 0.474 0.639 0.529
0.980 0.5
  0.847 0.873 0.711 1.0 1.0 1.0 1.0 1.0 1.0]
 [0.531 0.192 0.444 0.153 0.034 0.144 0.108 0.128 0.222 0.161
0.775 0.153
  0.5 0.532 0.308 1.0 1.0 0.996 1.0 1.0 1.0]
 [0.498 0.162 0.409 0.125 0.020 0.116 0.083 0.102 0.190 0.133
0.758 0.127
  0.468 0.5 0.276 1.0 1.0 0.996 1.0 1.0 1.0]
 [0.728 0.358 0.662 0.298 0.111 0.286 0.232 0.261 0.399 0.306
0.904 0.289
  0.692 0.724 0.5 1.0 1.0 1.0 1.0 1.0 1.0]
 [0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5
0.467
  0.150 0.351 0.245 0.305]
 [0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.533 0.5
  0.168 0.380 0.269 0.331]
 [0.001 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.065 0.0 0.004 0.004
```

```
0.0
  0.850 0.832 0.5 0.749 0.646 0.703]
 [0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.005 0.0 0.0 0.0 0.0
0.649
  0.620 0.251 0.5 0.373 0.443]
 [0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.022 0.0 0.0 0.0 0.0
0.755
  0.731 0.354 0.627 0.5 0.573]
 [0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.013 0.0 0.0 0.0 0.0
0.695
  0.669 0.297 0.557 0.427 0.5]]
```

```
#Priority Index/ Ranking of Stretches.
PI = []
for i in range(0,len(E_2D_array)):
    PI_Sum = sum(E_2D_array[i]) - (len(E_2D_array) * 0.5)
    PI.append(PI_Sum)

Ranking = pd.DataFrame()
Ranking["Priority Index"] = PI
Ranking["Stretch No."] = row_index
Ranking["Rank"] = Ranking["Priority Index"].rank(ascending =
False, method = 'max')
Ranking = Ranking.set_index("Stretch No.")
print(Ranking)
Ranking.to_excel("Fuzzy_TOPSIS_Ranking.xlsx")
```

Stretch No.	Priority Index	Rank
A1	-0.584075927734375	14.0
A2	4.00006103515625	8.0
A3	0.287994384765625	11.0
A4	4.89788818359375	6.0
A5	7.88610839843750	1.0
A6	5.05987548828125	5.0
A7	5.90686035156250	2.0
A8	5.48132324218750	3.0
A9	3.43792724609375	9.0
A10	4.80108642578125	7.0
A11	-2.99488830566406	15.0
A12	5.11279296875000	4.0
A13	-0.119628906250000	12.0
A14	-0.537597656250000	13.0
A15	2.25000000000000	10.0
A16	-8.48202514648438	21.0
A17	-8.31906127929688	20.0
A18	-6.14596557617188	16.0
A19	-7.65899658203125	19.0
A20	-6.93807983398438	17.0
A21	-7.34210205078125	18.0

C.2 ALGORITHM FOR AHP METHOD– WITH OUTPUTS

The algorithm below is a step-by-step execution of the method, AHP, which is written in python programming language.

```
#Importing Necessary Libraries.
import pandas as pd
import numpy as np
from scipy.stats import gmean
from itertools import groupby


#Importing Expert PairWise Matrices and combining them into a
single PairWise Matrix.
PairWise_Matrices = pd.read_excel("AHP Expert Data.xlsx",
sheet_name = None)
keys = list(PairWise_Matrices.keys())
print(keys)
print("\n")

i = 1
count_Experts = 0
for n in range(len(PairWise_Matrices)):
    globals()["Expert" + str(i)] =
PairWise_Matrices[keys[count_Experts]]
    globals()["Expert" + str(i)] = globals()["Expert" +
str(i)].set_index("CRITERIA")
    globals()["Expert" + str(i) + "_1"] =
np.array(globals()["Expert" + str(i)])
    i += 1
    count_Experts += 1

print("Total no of Experts: ", count_Experts)
print("\n")

PairWise_Matrix_comb = np.ones([len(Expert1),len(Expert1)])
for i in range(1, count_Experts+1):
    PairWise_Matrix_comb = PairWise_Matrix_comb *
globals()["Expert" + str(i) + "_1"]

PairWise_Matrix_comb = np.sqrt(PairWise_Matrix_comb)
np.shape(PairWise_Matrix_comb)

PairWise_Matrix_comb_1 = pd.DataFrame(PairWise_Matrix_comb,
columns = Expert1.columns)
PairWise_Matrix_comb_1["CRITERIA"] = Expert1.index
PairWise_Matrix_comb_1 =
```

```
PairWise_Matrix_comb_1.set_index("CRITERIA")
print(PairWise_Matrix_comb_1)
PairWise_Matrix_comb_1.to_excel("Pair Wise Matrix Combined.xlsx")
```

```
['Exp1', 'Exp2', 'Exp3', 'Exp4', 'Exp5', 'Exp6', 'Exp7', 'Exp8',
'Exp9', 'Exp10', 'Exp11', 'Exp12', 'Exp13', 'Exp14', 'Exp15']


Total no of Experts:  15


                AC        BC        LC        TC         S
PH  \
CRITERIA
AC        1.000000  2.236068  2.645751  2.645751  0.577350
0.377964
BC        0.447214  1.000000  2.236068  2.236068  0.447214
0.333333
LC        0.377964  0.447214  1.000000  1.732051  0.408248
0.333333
TC        0.377964  0.447214  0.577350  1.000000  0.377964
0.333333
S         1.732051  2.236068  2.449490  2.645751  1.000000
0.447214
PH        2.645751  3.000000  3.000000  3.000000  2.236068
1.000000
RA        2.236068  2.449490  2.645751  2.449490  1.732051
0.377964
R         1.732051  1.732051  2.236068  2.000000  0.447214
0.408248
D         2.000000  2.236068  2.645751  2.645751  0.707107
0.447214
EC        0.353553  0.447214  0.707107  0.577350  0.500000
0.353553
P         0.707107  1.732051  1.414214  2.236068  1.414214
0.408248
MUD       2.236068  2.449490  2.645751  2.645751  0.447214
0.377964
POP       2.000000  2.000000  2.449490  1.732051  0.577350
0.447214
VOLT      2.828427  1.732051  1.414214  1.414214  0.500000
0.577350
FS        1.414214  1.414214  1.414214  1.414214  0.377964
0.353553
NTOWN     2.000000  2.449490  2.645751  1.414214  0.707107
0.333333
CBR       2.449490  2.645751  2.828427  2.828427  2.236068
1.732051
```

MUD \ CRITERIA	RA	R	D	EC	P	
AC	0.447214	0.577350	0.500000	2.828427	1.414214	0.447214
BC	0.408248	0.577350	0.447214	2.236068	0.577350	0.408248
LC	0.377964	0.447214	0.377964	1.414214	0.707107	0.377964
TC	0.408248	0.500000	0.377964	1.732051	0.447214	0.377964
S	0.577350	2.236068	1.414214	2.000000	0.707107	2.236068
PH	2.645751	2.449490	2.236068	2.828427	2.449490	2.645751
RA	1.000000	2.236068	2.449490	2.645751	4.582576	0.577350
R	0.447214	1.000000	0.447214	2.000000	2.236068	2.000000
D	0.408248	2.236068	1.000000	2.449490	2.236068	1.732051
EC	0.377964	0.500000	0.408248	1.000000	0.577350	1.732051
P	1.527525	0.447214	0.447214	1.732051	1.000000	0.447214
MUD	0.577350	0.500000	0.577350	0.577350	2.236068	1.000000
POP	1.732051	2.236068	2.449490	1.732051	2.645751	3.000000
VOLT	2.236068	0.447214	2.000000	1.732051	2.645751	2.828427
FS	2.449490	2.236068	0.377964	2.236068	2.236068	2.449490
NTOWN	1.732051	0.707107	0.577350	1.000000	1.732051	2.645751
CBR	2.645751	2.236068	1.732051	3.000000	2.828427	2.645751

CRITERIA	POP	VOLT	FS	NTOWN	CBR
AC	0.500000	0.353553	0.707107	0.500000	0.408248
BC	0.500000	0.577350	0.707107	0.408248	0.377964
LC	0.408248	0.707107	0.707107	0.377964	0.353553
TC	0.577350	0.707107	0.707107	0.707107	0.353553
S	1.732051	2.000000	2.645751	1.414214	0.447214
PH	2.236068	1.732051	2.828427	3.000000	0.577350
RA	0.577350	0.447214	0.408248	0.577350	0.377964
R	0.447214	2.236068	0.447214	1.414214	0.447214

```
D          0.408248  0.500000  2.645751  1.732051  0.577350
EC         0.577350  0.577350  0.447214  1.000000  0.333333
P          0.377964  0.377964  0.447214  0.577350  0.353553
MUD        0.333333  0.353553  0.408248  0.377964  0.377964
POP        1.000000  0.447214  0.577350  2.645751  0.707107
VOLT       2.236068  1.000000  2.645751  2.236068  1.000000
FS         1.732051  0.377964  1.000000  1.732051  0.447214
NTOWN      0.377964  0.447214  0.577350  1.000000  0.408248
CBR        1.414214  1.000000  2.236068  2.449490  1.000000
```

```
#Obtaining Criteria Weights.
colWise_Sum = PairWise_Matrix_comb_1.loc[:].sum(axis = 0)
PWMatrix_with_colSum = PairWise_Matrix_comb_1.append(colWise_Sum,
ignore_index=True)

for i in range(0, len(PWMatrix_with_colSum.columns)):
    sum_val = PWMatrix_with_colSum.iloc[-1,i]
    for j in range(0, len(PWMatrix_with_colSum)-1):
        if sum_val > 0:
            PWMatrix_with_colSum.iloc[j,i] =
PWMatrix_with_colSum.iloc[j,i] / sum_val

PWMatrix_with_colSum.drop(PWMatrix_with_colSum.tail(1).index,inpl
ace=True)
PWMatrix_with_colSum.to_excel("Normalized Pair Wise Matrix.xlsx")
print("\n")
n = len(PWMatrix_with_colSum)
rowWise_Sum = PWMatrix_with_colSum.loc[:].mean(axis = 1)
PWMatrix_with_colSum["Criteria Weights"] = rowWise_Sum
print(PWMatrix_with_colSum)
PWMatrix_with_colSum.to_excel("Matrix with Criteria
Weights.xlsx")

Criteria_Wts = PWMatrix_with_colSum["Criteria Weights"].tolist()
Criteria_Wts_1 = pd.DataFrame(Criteria_Wts)
Criteria_Wts_1.to_excel("Weightages.xlsx")
Criteria_Wts = np.array(Criteria_Wts)
```

```
          AC        BC        LC          TC          S         PH
RA   \
0   0.037682  0.072944  0.075689  0.076429  0.039294  0.043736
0.022362
1   0.016852  0.032622  0.063969  0.064594  0.030437  0.038572
0.020414
2   0.014242  0.014589  0.028608  0.050034  0.027785  0.038572
0.018900
```

```
3   0.014242  0.014589  0.016517  0.028887  0.025724  0.038572
0.020414
4   0.065267  0.072944  0.070075  0.076429  0.068059  0.051750
0.028870
5   0.099697  0.097865  0.085824  0.086662  0.152185  0.115716
0.132298
6   0.084259  0.079907  0.075689  0.070759  0.117882  0.043736
0.050004
7   0.065267  0.056502  0.063969  0.057775  0.030437  0.047241
0.022362
8   0.075364  0.072944  0.075689  0.076429  0.048125  0.051750
0.020414
9   0.013323  0.014589  0.020229  0.016678  0.034030  0.040912
0.018900
10  0.026645  0.056502  0.040458  0.064594  0.096250  0.047241
0.076382
11  0.084259  0.079907  0.075689  0.076429  0.030437  0.043736
0.028870
12  0.075364  0.065243  0.070075  0.050034  0.039294  0.051750
0.086609
13  0.106581  0.056502  0.040458  0.040853  0.034030  0.066808
0.111812
14  0.053290  0.046134  0.040458  0.040853  0.025724  0.040912
0.122484
15  0.075364  0.079907  0.075689  0.040853  0.048125  0.038572
0.086609
16  0.092301  0.086309  0.080915  0.081706  0.152185  0.200425
0.132298

           R         D        EC         P       MUD       POP
VOLT  \
0   0.026767  0.028059  0.085338  0.045242  0.016232  0.032393
0.025543
1   0.026767  0.025096  0.067465  0.018470  0.014818  0.032393
0.041711
2   0.020734  0.021210  0.042669  0.022621  0.013719  0.026449
0.051085
3   0.023181  0.021210  0.052258  0.014307  0.013719  0.037404
0.051085
4   0.103669  0.079362  0.060343  0.022621  0.081160  0.112212
0.144491
5   0.113563  0.125482  0.085338  0.078362  0.096030  0.144866
0.125133
6   0.103669  0.137459  0.079826  0.146602  0.020955  0.037404
0.032309
7   0.046362  0.025096  0.060343  0.071534  0.072592  0.028973
0.161546
8   0.103669  0.056117  0.073904  0.071534  0.062866  0.026449
0.036123
```

```
9   0.023181  0.022910  0.030171  0.018470  0.062866  0.037404
0.041711
10  0.020734  0.025096  0.052258  0.031991  0.016232  0.024487
0.027306
11  0.023181  0.032399  0.017419  0.071534  0.036296  0.021595
0.025543
12  0.103669  0.137459  0.052258  0.084641  0.108888  0.064786
0.032309
13  0.020734  0.112235  0.052258  0.084641  0.102660  0.144866
0.072245
14  0.103669  0.021210  0.067465  0.071534  0.088907  0.112212
0.027306
15  0.032783  0.032399  0.030171  0.055410  0.096030  0.024487
0.032309
16  0.103669  0.097198  0.090514  0.090485  0.096030  0.091621
0.072245

          FS     NTOWN       CBR  Criteria Weights
0   0.035104  0.022574  0.047760          0.043126
1   0.035104  0.018431  0.044218          0.034820
2   0.035104  0.017064  0.041362          0.028515
3   0.035104  0.031924  0.041362          0.028265
4   0.131348  0.063848  0.052319          0.075574
5   0.140417  0.135441  0.067543          0.110731
6   0.020267  0.026066  0.044218          0.068883
7   0.022202  0.063848  0.052319          0.055786
8   0.131348  0.078197  0.067543          0.066380
9   0.022202  0.045147  0.038996          0.029513
10  0.022202  0.026066  0.041362          0.040930
11  0.020267  0.017064  0.044218          0.042873
12  0.028663  0.119448  0.082724          0.073718
13  0.131348  0.100952  0.116989          0.082116
14  0.049645  0.078197  0.052319          0.061313
15  0.028663  0.045147  0.047760          0.051193
16  0.111010  0.110587  0.116989          0.106264
```

```
#Importing Normalized data sheet and rearranging the columns.
Normalized_Data = pd.read_excel("Normalized Data Sheet.xlsx")
Normalized_Data = Normalized_Data.set_index("STRETCH NO.")
Normalized_Data_1 =
Normalized_Data.groupby(Normalized_Data.columns.str[:2],axis=1,
sort = False).sum()
print(Normalized_Data_1)
Normalized_Data_1.to_excel("Check Normalized Data.xlsx")
Normalized_Data_Matrix = np.array(Normalized_Data_1)
```

```
              A_    B_    L_    T_    PH    RA    R_    S_    E_    D_
P_    M_    PO \
```

STRETCH NO.													
S1	73	115	117	157	41	178	0	20	157	172	45	36	26
S2	151	203	122	175	200	130	0	73	160	190	58	50	52
S3	230	77	80	111	158	115	0	20	92	107	45	95	69
S4	78	168	125	180	216	168	0	140	185	160	70	123	69
S5	190	91	103	116	200	153	255	176	275	242	95	95	45
S6	183	115	154	151	158	145	0	246	146	175	25	85	65
S7	151	78	140	167	175	109	0	250	207	217	91	55	68
S8	113	91	148	133	158	128	0	123	196	197	250	134	68
S9	131	111	137	112	200	116	0	53	142	172	108	0	65
S10	53	227	122	94	225	190	100	20	157	210	183	191	71
S11	0	34	59	121	66	33	64	0	0	45	58	106	66
S12	58	218	200	200	191	221	180	100	67	175	91	237	70
S13	78	271	156	181	100	150	25	33	53	32	45	85	70
S14	0	58	104	136	50	214	45	100	14	135	25	235	22
S15	166	206	216	108	150	179	102	53	82	92	45	70	53
S16	0	0	0	0	0	0	40	0	0	0	0	0	5
S17	0	0	0	0	16	0	20	0	0	0	0	0	8
S18	0	0	0	0	25	0	11	0	0	12	37	0	100
S19	0	0	0	0	0	0	20	0	0	0	0	0	15
S20	0	0	0	0	0	4	60	0	0	0	50	0	58
S21	0	0	0	0	0	4	0	0	0	40	12	0	19

STRETCH NO.	VO	FS	NT	CB
S1	10	54	20	64
S2	35	62	26	66
S3	51	48	0	57

```
S4              32  100  33  44
S5              73   86   6  21
S6              76   50  53  55
S7              76   76  76  75
S8             100   44  73  44
S9              94   70  63  57
S10             11   28  63  38
S11             76   66  76  66
S12             96   60  60   0
S13             77   40  73  51
S14             43   48  66  21
S15             18   48   0  14
S16             16   64   6  25
S17             16   32  53  28
S18             17   64  66  21
S19             10   80  80  23
S20              6   30  36  15
S21              7   66  86  29
```

```
#CONSISTENCY CHECK:
PairWise_Matrix_comb_1 = np.array(PairWise_Matrix_comb_1)
Consistency_check_Matrix = PairWise_Matrix_comb_1 * Criteria_Wts
Consistency_check_Matrix = pd.DataFrame(Consistency_check_Matrix,
columns = Expert1.columns)
Consistency_check_Matrix["CRITERIA"] = Expert1.index
Consistency_check_Matrix =
Consistency_check_Matrix.set_index("CRITERIA")

rowSum = Consistency_check_Matrix.loc[:].sum(axis = 1)
Consistency_check_Matrix["WSV"] = rowSum
Consistency_check_Matrix.to_excel("Consistency Check
Matrix.xlsx")
Consistency_check_Matrix["Criteria Weights"] = list(Criteria_Wts)

Consistency_check_Matrix['WSV / CW'] =
Consistency_check_Matrix['WSV'] /
Consistency_check_Matrix['Criteria Weights']
print(Consistency_check_Matrix)
Consistency_check_Matrix.to_excel("Consistency Check.xlsx")
print("\n")

λmax = gmean(Consistency_check_Matrix["WSV / CW"])
print("λmax: ", λmax)
```

```
                AC         BC        LC          TC          S
PH   \
```

CRITERIA						
AC	0.043126	0.077859	0.075442	0.074781	0.043633	0.041852
BC	0.019287	0.034820	0.063760	0.063202	0.033798	0.036910
LC	0.016300	0.015572	0.028515	0.048956	0.030853	0.036910
TC	0.016300	0.015572	0.016463	0.028265	0.028564	0.036910
S	0.074697	0.077859	0.069846	0.074781	0.075574	0.049520
PH	0.114102	0.104459	0.085544	0.084794	0.168990	0.110731
RA	0.096434	0.085290	0.075442	0.069234	0.130899	0.041852
R	0.074697	0.060309	0.063760	0.056529	0.033798	0.045206
D	0.086253	0.077859	0.075442	0.074781	0.053439	0.049520
EC	0.015247	0.015572	0.020163	0.016319	0.037787	0.039149
P	0.030495	0.060309	0.040326	0.063202	0.106878	0.045206
MUD	0.096434	0.085290	0.075442	0.074781	0.033798	0.041852
POP	0.086253	0.069639	0.069846	0.048956	0.043633	0.049520
VOLT	0.121980	0.060309	0.040326	0.039972	0.037787	0.063930
FS	0.060990	0.049242	0.040326	0.039972	0.028564	0.039149
NTOWN	0.086253	0.085290	0.075442	0.039972	0.053439	0.036910
CBR	0.105638	0.092124	0.080651	0.079945	0.168990	0.191791

\ CRITERIA	RA	R	D	EC	P	MUD
AC	0.030805	0.032208	0.033190	0.083475	0.057883	0.019173
BC	0.028121	0.032208	0.029686	0.065993	0.023631	0.017503
LC	0.026035	0.024948	0.025089	0.041737	0.028942	0.016205
TC	0.028121	0.027893	0.025089	0.051118	0.018304	0.016205
S	0.039770	0.124742	0.093876	0.059026	0.028942	0.095867

PH	0.182247	0.136648	0.148431	0.083475	0.100257	0.113432
RA	0.068883	0.124742	0.162598	0.078084	0.187564	0.024753
R	0.030805	0.055786	0.029686	0.059026	0.091522	0.085746
D	0.028121	0.124742	0.066380	0.072291	0.091522	0.074259
EC	0.026035	0.027893	0.027100	0.029513	0.023631	0.074259
P	0.105221	0.024948	0.029686	0.051118	0.040930	0.019173
MUD	0.039770	0.027893	0.038325	0.017039	0.091522	0.042873
POP	0.119309	0.124742	0.162598	0.051118	0.108290	0.128620
VOLT	0.154027	0.024948	0.132761	0.051118	0.108290	0.121264
FS	0.168728	0.124742	0.025089	0.065993	0.091522	0.105017
NTOWN	0.119309	0.039447	0.038325	0.029513	0.070892	0.113432
CBR	0.182247	0.124742	0.114974	0.088539	0.115767	0.113432

WSV \ CRITERIA	POP	VOLT	FS	NTOWN	CBR	
AC	0.036859	0.029032	0.043355	0.025596	0.043382	0.791655
BC	0.036859	0.047410	0.043355	0.020899	0.040164	0.637606
LC	0.030095	0.058065	0.043355	0.019349	0.037570	0.528497
TC	0.042561	0.058065	0.043355	0.036199	0.037570	0.526555
S	0.127684	0.164232	0.162219	0.072398	0.047523	1.438556
PH	0.164839	0.142229	0.173419	0.153579	0.061352	2.128526
RA	0.042561	0.036723	0.025031	0.029556	0.040164	1.319811
R	0.032968	0.183617	0.027420	0.072398	0.047523	1.050797
D	0.030095	0.041058	0.162219	0.088669	0.061352	1.258003
EC	0.042561	0.047410	0.027420	0.051193	0.035421	0.556673
P	0.027863	0.031037	0.027420	0.029556	0.037570	

```
0.770938
MUD       0.024573  0.029032  0.025031  0.019349  0.040164
0.803169
POP       0.073718  0.036723  0.035399  0.135444  0.075140
1.418948
VOLT      0.164839  0.082116  0.162219  0.114471  0.106264
1.586621
FS        0.127684  0.031037  0.061313  0.088669  0.047523
1.195561
NTOWN     0.027863  0.036723  0.035399  0.051193  0.043382
0.982785
CBR       0.104254  0.082116  0.137100  0.125396  0.106264
2.013969

          Criteria Weights   WSV / CW
CRITERIA
AC                0.043126  18.356618
BC                0.034820  18.311691
LC                0.028515  18.534303
TC                0.028265  18.629414
S                 0.075574  19.034939
PH                0.110731  19.222555
RA                0.068883  19.160172
R                 0.055786  18.836082
D                 0.066380  18.951420
EC                0.029513  18.862052
P                 0.040930  18.835624
MUD               0.042873  18.733584
POP               0.073718  19.248223
VOLT              0.082116  19.321715
FS                0.061313  19.499338
NTOWN             0.051193  19.197698
CBR               0.106264  18.952515

λmax:  18.919983800914327
```

```
#Obtaining Consistency Ratio.
Consistency_index = (λmax - len(Expert1)) / (len(Expert1) - 1)
print("CI: ", Consistency_index)

n = len(Expert1)
if n > 0 and n < 51:
    Random_Index_vals={1:0, 2:0, 3:0.58, 4:0.90, 5:1.12, 6:1.24,
7:1.32, 8:1.41, 9:1.45, 10:1.49, 11:1.51, 12:1.48, 13:1.56,
14:1.57, 15:1.59, 16:1.6, 17:1.61, 18:1.61, 19:1.62, 20:1.63,
21:1.63, 22:1.64, 23:1.65, 24:1.65, 25:1.66, 26:1.66, 27:1.66,
28:1.67, 29:1.67, 30:1.67, 31:1.67, 32:1.68, 33:1.68, 34:1.68,
35:1.68, 36:1.69, 37:1.69, 38:1.69, 39:1.69, 40: 1.69, 41:1.70,
```

```
42:1.70, 43:1.70, 44:1.70, 45:1.70, 46:1.70, 47:1.70, 48:1.70,
49:1.71, 50:1.71}

    Random_Index = Random_Index_vals.get(n)
else:
    Random_Index = float(input("Enter the Random Index according
to no. of criterias: "))


print("RI: ", Random_Index)
Consistency_Ratio = Consistency_index / Random_Index
print("CR: ", Consistency_Ratio)



CI:  0.11999898755714544
RI:  1.61
CR:  0.07453353264418972



#Ranking of Stretches - if values are consistent.
if Consistency_Ratio < 0.10:
    Consistency_Ratio = np.dot(Normalized_Data_Matrix,
Criteria_Wts)
    AHP_Ranking = pd.DataFrame()
    AHP_Ranking["Priority Index"] = Consistency_Ratio
    AHP_Ranking["STRETCHES"] = Normalized_Data_1.index
    AHP_Ranking = AHP_Ranking.set_index("STRETCHES")
    AHP_Ranking["Rank"] = AHP_Ranking["Priority
Index"].rank(ascending = False, method = 'max')
    print(AHP_Ranking)
    AHP_Ranking.to_excel("AHPRanking.xlsx")
else:
    print('The provided metrics are "Inconsistent".')
```

STRETCHES	Priority Index	Rank
S1	71.607265	14.0
S2	95.186984	8.0
S3	77.263455	12.0
S4	105.953905	6.0
S5	127.499959	1.0
S6	104.455392	7.0
S7	111.183579	3.0
S8	108.134182	5.0
S9	92.510333	9.0
S10	109.788088	4.0
S11	53.640310	15.0
S12	121.623187	2.0
S13	83.135125	11.0
S14	74.700589	13.0
S15	87.500465	10.0
S16	11.325549	21.0
S17	12.141079	20.0
S18	22.817738	16.0
S19	14.749127	19.0
S20	16.667044	17.0
S21	15.620946	18.0

C.3 ALGORITHM FOR CONCORDANCE APPROACH– WITH OUTPUTS

The algorithm below is a step-by-step execution of the method, Concordance, which is written in python programming language.

```
#Importing Necessary Libraries.
import pandas as pd
from collections import Counter
import numpy as np


#Imporing Normalized Data and combining the same criterias.
FieldData = pd.read_excel("Check Normalized Data.xlsx")
FieldData = FieldData.set_index("STRETCH NO.")
FieldData.to_excel("Raw Data Combined.xlsx")
print(FieldData)
```

```
              A_   B_   L_   T_   PH   RA   R_   S_   E_   D_  
P_   M_   PO  \
STRETCH NO.
S1            73  115  117  157   41  178    0   20  157  172
45   36   26
S2           151  203  122  175  200  130    0   73  160  190
58   50   52
S3           230   77   80  111  158  115    0   20   92  107
45   95   69
S4            78  168  125  180  216  168    0  140  185  160
70  123   69
S5           190   91  103  116  200  153  255  176  275  242
95   95   45
S6           183  115  154  151  158  145    0  246  146  175
25   85   65
S7           151   78  140  167  175  109    0  250  207  217
91   55   68
S8           113   91  148  133  158  128    0  123  196  197
250  134   68
S9           131  111  137  112  200  116    0   53  142  172
108    0   65
S10           53  227  122   94  225  190  100   20  157  210
183  191   71
S11            0   34   59  121   66   33   64    0    0   45
58  106   66
S12           58  218  200  200  191  221  180  100   67  175
91  237   70
S13           78  271  156  181  100  150   25   33   53   32
45   85   70
```

```
S14             0    58   104   136    50   214    45   100    14   135
25   235    22
S15           166   206   216   108   150   179   102    53    82    92
45    70    53
S16             0     0     0     0     0     0    40     0     0     0
0     0     5
S17             0     0     0     0    16     0    20     0     0     0
0     0     8
S18             0     0     0     0    25     0    11     0     0    12
37     0   100
S19             0     0     0     0     0     0    20     0     0     0
0     0    15
S20             0     0     0     0     0     4    60     0     0     0
50     0    58
S21             0     0     0     0     0     4     0     0     0    40
12     0    19

              VO   FS   NT   CB
STRETCH NO.
S1            10   54   20   64
S2            35   62   26   66
S3            51   48    0   57
S4            32  100   33   44
S5            73   86    6   21
S6            76   50   53   55
S7            76   76   76   75
S8           100   44   73   44
S9            94   70   63   57
S10           11   28   63   38
S11           76   66   76   66
S12           96   60   60    0
S13           77   40   73   51
S14           43   48   66   21
S15           18   48    0   14
S16           16   64    6   25
S17           16   32   53   28
S18           17   64   66   21
S19           10   80   80   23
S20            6   30   36   15
S21            7   66   86   29
```

```
#Finding maximum values, minimum values and difference of maximum
and minimum values.
max_Values = list(FieldData.max())
print("Max values: ", max_Values, "\n")
min_Values = list(FieldData.min())
print("Min values: ", min_Values, "\n")
```

```
max_values_row = pd.Series(max_Values, index = FieldData.columns)
FieldData_1 = FieldData.append(max_values_row, ignore_index=True)
min_values_row = pd.Series(min_Values, index =
FieldData_1.columns)
FieldData_2 = FieldData_1.append(min_values_row,
ignore_index=True)
#print(FieldData_2)

max_min_Diff = list(np.array(max_Values) - np.array(min_Values))
print("Max values - Min values: ", max_min_Diff, "\n")

max_min_diffRow = pd.Series(max_min_Diff, index =
FieldData.columns)
FieldData_Filled = FieldData_2.append(max_min_diffRow,
ignore_index=True)
```

```
Max values:  [230, 271, 216, 200, 225, 221, 255, 250, 275, 242,
250, 237, 100, 100, 100, 86, 75]

Min values:  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 6, 28, 0, 0]

Max values - Min values:  [230, 271, 216, 200, 225, 221, 255,
250, 275, 242, 250, 237, 95, 94, 72, 86, 75]
```

```
#Field Data Transformed - Linear Function Transformation from 0
to 1
for i in range(0, len(FieldData.columns)):
    denom = FieldData_Filled.iloc[-1,i]
    min_val = FieldData_Filled.iloc[-2,i]
    for j in range(0, len(FieldData)):
            FieldData.iloc[j,i] = (FieldData.iloc[j,i] -
min_val)/denom

print(FieldData)
FieldData.to_excel("Field Data Concordance.xlsx")
```

```
                   A_        B_        L_     T_        PH
RA  \
STRETCH NO.
S1            0.317391  0.424354  0.541667  0.785  0.182222
0.805430
S2            0.656522  0.749077  0.564815  0.875  0.888889
0.588235
S3            1.000000  0.284133  0.370370  0.555  0.702222
0.520362
```

S4	0.339130	0.619926	0.578704	0.900	0.960000	0.760181
S5	0.826087	0.335793	0.476852	0.580	0.888889	0.692308
S6	0.795652	0.424354	0.712963	0.755	0.702222	0.656109
S7	0.656522	0.287823	0.648148	0.835	0.777778	0.493213
S8	0.491304	0.335793	0.685185	0.665	0.702222	0.579186
S9	0.569565	0.409594	0.634259	0.560	0.888889	0.524887
S10	0.230435	0.837638	0.564815	0.470	1.000000	0.859729
S11	0.000000	0.125461	0.273148	0.605	0.293333	0.149321
S12	0.252174	0.804428	0.925926	1.000	0.848889	1.000000
S13	0.339130	1.000000	0.722222	0.905	0.444444	0.678733
S14	0.000000	0.214022	0.481481	0.680	0.222222	0.968326
S15	0.721739	0.760148	1.000000	0.540	0.666667	0.809955
S16	0.000000	0.000000	0.000000	0.000	0.000000	0.000000
S17	0.000000	0.000000	0.000000	0.000	0.071111	0.000000
S18	0.000000	0.000000	0.000000	0.000	0.111111	0.000000
S19	0.000000	0.000000	0.000000	0.000	0.000000	0.000000
S20	0.000000	0.000000	0.000000	0.000	0.000000	0.018100
S21	0.000000	0.000000	0.000000	0.000	0.000000	0.018100

PO \ STRETCH NO.	R_	S_	E_	D_	P_	M_	
S1	0.000000	0.080	0.570909	0.710744	0.180	0.151899	0.221053
S2	0.000000	0.292	0.581818	0.785124	0.232	0.210970	0.494737
S3	0.000000	0.080	0.334545	0.442149	0.180	0.400844	0.673684
S4	0.000000	0.560	0.672727	0.661157	0.280	0.518987	0.673684
S5	1.000000	0.704	1.000000	1.000000	0.380	0.400844	

							0.421053
S6	0.000000	0.984	0.530909	0.723140	0.100	0.358650	0.631579
S7	0.000000	1.000	0.752727	0.896694	0.364	0.232068	0.663158
S8	0.000000	0.492	0.712727	0.814050	1.000	0.565401	0.663158
S9	0.000000	0.212	0.516364	0.710744	0.432	0.000000	0.631579
S10	0.392157	0.080	0.570909	0.867769	0.732	0.805907	0.694737
S11	0.250980	0.000	0.000000	0.185950	0.232	0.447257	0.642105
S12	0.705882	0.400	0.243636	0.723140	0.364	1.000000	0.684211
S13	0.098039	0.132	0.192727	0.132231	0.180	0.358650	0.684211
S14	0.176471	0.400	0.050909	0.557851	0.100	0.991561	0.178947
S15	0.400000	0.212	0.298182	0.380165	0.180	0.295359	0.505263
S16	0.156863	0.000	0.000000	0.000000	0.000	0.000000	0.000000
S17	0.078431	0.000	0.000000	0.000000	0.000	0.000000	0.031579
S18	0.043137	0.000	0.000000	0.049587	0.148	0.000000	1.000000
S19	0.078431	0.000	0.000000	0.000000	0.000	0.000000	0.105263
S20	0.235294	0.000	0.000000	0.000000	0.200	0.000000	0.557895
S21	0.000000	0.000	0.000000	0.165289	0.048	0.000000	0.147368

STRETCH NO.	VO	FS	NT	CB
S1	0.042553	0.361111	0.232558	0.853333
S2	0.308511	0.472222	0.302326	0.880000
S3	0.478723	0.277778	0.000000	0.760000
S4	0.276596	1.000000	0.383721	0.586667
S5	0.712766	0.805556	0.069767	0.280000
S6	0.744681	0.305556	0.616279	0.733333
S7	0.744681	0.666667	0.883721	1.000000
S8	1.000000	0.222222	0.848837	0.586667
S9	0.936170	0.583333	0.732558	0.760000
S10	0.053191	0.000000	0.732558	0.506667
S11	0.744681	0.527778	0.883721	0.880000
S12	0.957447	0.444444	0.697674	0.000000
S13	0.755319	0.166667	0.848837	0.680000

```
S14          0.393617  0.277778  0.767442  0.280000
S15          0.127660  0.277778  0.000000  0.186667
S16          0.106383  0.500000  0.069767  0.333333
S17          0.106383  0.055556  0.616279  0.373333
S18          0.117021  0.500000  0.767442  0.280000
S19          0.042553  0.722222  0.930233  0.306667
S20          0.000000  0.027778  0.418605  0.200000
S21          0.010638  0.527778  1.000000  0.386667
```

```
#Importing the Weightages and assigning weightages by comparing
the values of each column.
Weightages = pd.read_excel("Weightages.xlsx")
Weightages = list(Weightages.iloc[:,-1])
print("Weightages: ",Weightages, "\n")

stretch_comp = []
for i in (range(len(FieldData))):
    val1 = FieldData.iloc[i,:]
    for j in range(len(FieldData)):
        val2 = FieldData.iloc[j,:]

        for k in range(len(val1)):
            if val1[k] > val2[k]:
                comp_vals = Weightages[k]
                stretch_comp.append(comp_vals)
            else:
                #print(0)
                stretch_comp.append(0)

#print(stretch_comp)
n = len(stretch_comp)// len(FieldData.columns)
D1_array = np.array(stretch_comp)
D2_array = np.reshape(D1_array, (n, len(FieldData.columns)))
stretch_Df = pd.DataFrame(D2_array, columns = FieldData.columns)

row_diff = ['m' + str(i) + "-" + "m" + str(j) for i in range(1,
len(FieldData)+1) for j in range(1, len(FieldData)+1)]
stretch_Df["mi-mj"] = row_diff
stretch_Df = stretch_Df.set_index("mi-mj")
print(stretch_Df)
stretch_Df.to_excel("Stretch Diff Weights.xlsx")
```

```
Weightages:  [0.04312639892298784, 0.03481961150848359,
0.02851451737123218, 0.02826468640934807, 0.07557448386619237,
0.1107306662952266, 0.06888302674115064, 0.05578636789311761,
0.06638039441853488, 0.02951283921423371, 0.04092977999686601,
```

```
0.04287319413218346, 0.07371838387391301, 0.08211595877378293,
0.06131288944500066, 0.05119286128165851, 0.106263939856088]

          A_       B_        L_        T_   PH        RA   R_
S_       E_  \
mi-mj
m1-m1    0.0  0.00000  0.000000  0.000000  0.0  0.000000  0.0
0.0  0.00000
m1-m2    0.0  0.00000  0.000000  0.000000  0.0  0.110731  0.0
0.0  0.00000
m1-m3    0.0  0.03482  0.028515  0.028265  0.0  0.110731  0.0
0.0  0.06638
m1-m4    0.0  0.00000  0.000000  0.000000  0.0  0.110731  0.0
0.0  0.00000
m1-m5    0.0  0.03482  0.028515  0.028265  0.0  0.110731  0.0
0.0  0.00000
...      ...      ...       ...       ...  ...       ...  ...
...      ...
m21-m17  0.0  0.00000  0.000000  0.000000  0.0  0.110731  0.0
0.0  0.00000
m21-m18  0.0  0.00000  0.000000  0.000000  0.0  0.110731  0.0
0.0  0.00000
m21-m19  0.0  0.00000  0.000000  0.000000  0.0  0.110731  0.0
0.0  0.00000
m21-m20  0.0  0.00000  0.000000  0.000000  0.0  0.000000  0.0
0.0  0.00000
m21-m21  0.0  0.00000  0.000000  0.000000  0.0  0.000000  0.0
0.0  0.00000

               D_       P_   M_        PO        VO        FS
NT  \
mi-mj
m1-m1    0.000000  0.00000  0.0  0.000000  0.000000  0.000000
0.000000
m1-m2    0.000000  0.00000  0.0  0.000000  0.000000  0.000000
0.000000
m1-m3    0.029513  0.00000  0.0  0.000000  0.000000  0.061313
0.051193
m1-m4    0.029513  0.00000  0.0  0.000000  0.000000  0.000000
0.000000
m1-m5    0.000000  0.00000  0.0  0.000000  0.000000  0.000000
0.051193
...           ...      ...  ...       ...       ...       ...
...
m21-m17  0.029513  0.04093  0.0  0.073718  0.000000  0.061313
0.051193
m21-m18  0.029513  0.00000  0.0  0.000000  0.000000  0.061313
0.051193
m21-m19  0.029513  0.04093  0.0  0.073718  0.000000  0.000000
```

```
0.051193
m21-m20  0.029513  0.00000  0.0  0.000000  0.082116  0.061313
0.051193
m21-m21  0.000000  0.00000  0.0  0.000000  0.000000  0.000000
0.000000

               CB
mi-mj
m1-m1    0.000000
m1-m2    0.000000
m1-m3    0.106264
m1-m4    0.106264
m1-m5    0.106264
...           ...
m21-m17  0.106264
m21-m18  0.106264
m21-m19  0.106264
m21-m20  0.106264
m21-m21  0.000000

[441 rows x 17 columns]
```

```
#Obtaining Row-Wise Sum.
sDF_rowWise_Sum = stretch_Df.loc[:].sum(axis = 1)
stretch_Df["SUM"] = sDF_rowWise_Sum
print(stretch_Df)
stretch_Df.to_excel("Stretch diff weights with rowwise sum.xlsx")
```

```
          A_       B_        L_        T_   PH        RA   R_
S_       E_  \
mi-mj
m1-m1    0.0  0.00000  0.000000  0.000000  0.0  0.000000  0.0
0.0  0.00000
m1-m2    0.0  0.00000  0.000000  0.000000  0.0  0.110731  0.0
0.0  0.00000
m1-m3    0.0  0.03482  0.028515  0.028265  0.0  0.110731  0.0
0.0  0.06638
m1-m4    0.0  0.00000  0.000000  0.000000  0.0  0.110731  0.0
0.0  0.00000
m1-m5    0.0  0.03482  0.028515  0.028265  0.0  0.110731  0.0
0.0  0.00000
...      ...      ...       ...       ...  ...       ...  ...
...      ...
m21-m17  0.0  0.00000  0.000000  0.000000  0.0  0.110731  0.0
0.0  0.00000
m21-m18  0.0  0.00000  0.000000  0.000000  0.0  0.110731  0.0
```

```
0.0  0.00000
m21-m19  0.0  0.00000  0.000000  0.000000  0.0  0.110731  0.0
0.0  0.00000
m21-m20  0.0  0.00000  0.000000  0.000000  0.0  0.000000  0.0
0.0  0.00000
m21-m21  0.0  0.00000  0.000000  0.000000  0.0  0.000000  0.0
0.0  0.00000

               D_       P_   M_        PO        VO        FS
NT  \
mi-mj
m1-m1    0.000000  0.00000  0.0  0.000000  0.000000  0.000000
0.000000
m1-m2    0.000000  0.00000  0.0  0.000000  0.000000  0.000000
0.000000
m1-m3    0.029513  0.00000  0.0  0.000000  0.000000  0.061313
0.051193
m1-m4    0.029513  0.00000  0.0  0.000000  0.000000  0.000000
0.000000
m1-m5    0.000000  0.00000  0.0  0.000000  0.000000  0.000000
0.051193
...           ...      ...  ...       ...       ...       ...
...
m21-m17  0.029513  0.04093  0.0  0.073718  0.000000  0.061313
0.051193
m21-m18  0.029513  0.00000  0.0  0.000000  0.000000  0.061313
0.051193
m21-m19  0.029513  0.04093  0.0  0.073718  0.000000  0.000000
0.051193
m21-m20  0.029513  0.00000  0.0  0.000000  0.082116  0.061313
0.051193
m21-m21  0.000000  0.00000  0.0  0.000000  0.000000  0.000000
0.000000

               CB       SUM
mi-mj
m1-m1    0.000000  0.000000
m1-m2    0.000000  0.110731
m1-m3    0.106264  0.516992
m1-m4    0.106264  0.246507
m1-m5    0.106264  0.359786
...           ...       ...
m21-m17  0.106264  0.473661
m21-m18  0.106264  0.359013
m21-m19  0.106264  0.412348
m21-m20  0.106264  0.330398
m21-m21  0.000000  0.000000

[441 rows x 18 columns]
```

```
#Arranging the obtained sum values in n*n columns and obtaining
Row-Wise sum.
SUM_vals = np.array(stretch_Df["SUM"].tolist())
SUM_vals_2D = np.reshape(SUM_vals, (len(FieldData),
len(FieldData)))
#print(SUM_vals_2D)
Ranking = pd.DataFrame(SUM_vals_2D)
Ranking["STRETCH NO."] = FieldData.index
Ranking = Ranking.set_index("STRETCH NO.")
#print(Ranking)
Ranking.to_excel("Ranking Matrix.xlsx")

Ranking_rowWise_Sum = Ranking.loc[:].sum(axis = 1)
Ranking["SUM"] = Ranking_rowWise_Sum
print(Ranking)
Ranking.to_excel("Ranking Matrix Conc.xlsx")
```

```
                    0         1         2         3         4
5  \
STRETCH NO.
S1           0.000000  0.110731  0.516992  0.246507  0.359786
0.413882
S2           0.820386  0.000000  0.689283  0.295839  0.322774
0.443059
S3           0.317408  0.241834  0.000000  0.231506  0.223109
0.306912
S4           0.684610  0.635278  0.625892  0.000000  0.613265
0.534604
S5           0.640214  0.601652  0.734018  0.386735  0.000000
0.539324
S6           0.482415  0.488058  0.548631  0.396513  0.460676
0.000000
S7           0.785567  0.638601  0.660668  0.503823  0.425857
0.588937
S8           0.589725  0.471024  0.571121  0.384646  0.453873
0.386723
S9           0.512272  0.337784  0.665135  0.381656  0.417555
0.417390
S10          0.638865  0.528235  0.623130  0.528235  0.564617
0.594615
S11          0.602865  0.380096  0.481836  0.308456  0.384429
0.445174
S12          0.784229  0.617829  0.784229  0.591556  0.452230
0.698930
S13          0.584869  0.521113  0.534026  0.473773  0.404890
0.519169
```

```
S14          0.487157  0.411582  0.415758  0.355796  0.261576
0.273680
S15          0.616143  0.402666  0.298734  0.286074  0.247783
0.283878
S16          0.212312  0.130196  0.181389  0.068883  0.106264
0.130196
S17          0.202192  0.120076  0.120076  0.120076  0.157457
0.068883
S18          0.337223  0.255107  0.255107  0.193794  0.124911
0.296037
S19          0.181389  0.181389  0.181389  0.120076  0.157457
0.181389
S20          0.234724  0.193794  0.161006  0.120076  0.124911
0.109813
S21          0.112506  0.112506  0.112506  0.051193  0.157457
0.112506



                    6         7         8         9  ...
12        13  \
STRETCH NO.                                          ...
S1           0.145550  0.341392  0.389332  0.238968  ...
0.374201  0.512843
S2           0.249389  0.460093  0.517758  0.443251  ...
0.478887  0.588418
S3           0.270449  0.284422  0.159718  0.321084  ...
0.425044  0.522929
S4           0.427294  0.440207  0.549461  0.471765  ...
0.483101  0.644204
S5           0.574143  0.511307  0.506870  0.435383  ...
0.595110  0.632160
S6           0.260064  0.468819  0.440009  0.405385  ...
0.437957  0.685390
S7           0.000000  0.517415  0.586946  0.552471  ...
0.530080  0.777513
S8           0.339984  0.000000  0.610020  0.563888  ...
0.497612  0.687936
S9           0.344171  0.321097  0.000000  0.405385  ...
0.561003  0.642269
S10          0.447529  0.436112  0.543422  0.000000  ...
0.508603  0.567723
S11          0.111756  0.287653  0.371196  0.329150  ...
0.400969  0.559991
S12          0.545495  0.524691  0.555219  0.521608  ...
0.690879  0.786757
S13          0.469920  0.451195  0.438997  0.491397  ...
0.000000  0.630901
S14          0.222487  0.312064  0.357731  0.432277  ...
0.369099  0.000000
```

```
S15          0.328947  0.347387  0.328947  0.368004  ...
0.539822  0.431947
S16          0.068883  0.130196  0.068883  0.143429  ...
0.130196  0.167577
S17          0.068883  0.068883  0.068883  0.143429  ...
0.000000  0.106264
S18          0.142601  0.203914  0.193794  0.268340  ...
0.135031  0.175961
S19          0.181389  0.181389  0.181389  0.112506  ...
0.112506  0.218770
S20          0.068883  0.068883  0.068883  0.061313  ...
0.109813  0.183531
S21          0.051193  0.112506  0.051193  0.112506  ...
0.142019  0.218770

                   14        15        16        17        18
19  \
STRETCH NO.
S1           0.342928  0.787688  0.797808  0.662777  0.736495
0.765276
S2           0.597334  0.869804  0.879924  0.744893  0.818611
0.806206
S3           0.547830  0.818611  0.879924  0.744893  0.818611
0.838994
S4           0.713926  0.931117  0.879924  0.806206  0.879924
0.879924
S5           0.752217  0.842543  0.842543  0.768825  0.842543
0.875089
S6           0.716122  0.869804  0.879924  0.703963  0.818611
0.890187
S7           0.671053  0.931117  0.931117  0.857399  0.818611
0.931117
S8           0.652613  0.869804  0.931117  0.796086  0.818611
0.931117
S9           0.615266  0.888244  0.888244  0.763333  0.775738
0.888244
S10          0.631996  0.856571  0.856571  0.731660  0.887494
0.938687
S11          0.486672  0.834707  0.834707  0.760988  0.722201
0.834707
S12          0.755715  0.832423  0.893736  0.707512  0.781230
0.893736
S13          0.419249  0.869804  1.000000  0.864969  0.887494
0.890187
S14          0.506741  0.789297  0.850610  0.623456  0.738104
0.773342
S15          0.000000  0.781230  0.842543  0.707512  0.781230
0.727895
S16          0.218770  0.000000  0.130196  0.175147  0.257263
```

```
0.249693
S17          0.157457  0.306750  0.000000  0.175147  0.263954
0.376460
S18          0.292488  0.353044  0.414357  0.000000  0.301851
0.479691
S19          0.218770  0.186224  0.186224  0.287653  0.000000
0.300886
S20          0.272105  0.345455  0.294262  0.220543  0.294262
0.000000
S21          0.218770  0.473661  0.473661  0.359013  0.412348
0.330398

                   20        SUM
STRETCH NO.
S1           0.818611   9.174675
S2           0.818611  11.699401
S3           0.818611   9.505824
S4           0.879924  13.180615
S5           0.842543  13.086560
S6           0.818611  11.515408
S7           0.879924  13.756726
S8           0.818611  12.562167
S9           0.837051  11.735421
S10          0.887494  12.920803
S11          0.722201  10.078524
S12          0.781230  13.980465
S13          0.857981  12.144370
S14          0.738104   9.473198
S15          0.781230   9.859586
S16          0.150999   2.888047
S17          0.226573   2.857707
S18          0.341222   5.130683
S19          0.212312   3.714379
S20          0.183531   3.222052
S21          0.000000   3.884673

[21 rows x 22 columns]
```

```
#Ranking of Stretches.
Concordance_Ranking = pd.DataFrame(list(Ranking["SUM"]), columns
= ["SUM"])
Concordance_Ranking["STRETCH NO."] = Ranking.index
Concordance_Ranking = Concordance_Ranking.set_index("STRETCH
NO.")
Concordance_Ranking["Rank"] =
Concordance_Ranking["SUM"].rank(ascending = False, method =
'max')
print(Concordance_Ranking)
```

STRETCH NO.	SUM	Rank
S1	9.174675	15.0
S2	11.699401	9.0
S3	9.505824	13.0
S4	13.180615	3.0
S5	13.086560	4.0
S6	11.515408	10.0
S7	13.756726	2.0
S8	12.562167	6.0
S9	11.735421	8.0
S10	12.920803	5.0
S11	10.078524	11.0
S12	13.980465	1.0
S13	12.144370	7.0
S14	9.473198	14.0
S15	9.859586	12.0
S16	2.888047	20.0
S17	2.857707	21.0
S18	5.130683	16.0
S19	3.714379	18.0
S20	3.222052	19.0
S21	3.884673	17.0

C.4 ALGORITHM FOR SENSITIVITY ANALYSIS – WITH OUTPUTS

This algorithm is written to get the sensitive weightages from each method when the value is changed iteratively. This algorithm is written in the python programming language.

SENSITIVITY CODE FOR AHP AND CONCORDANCE

```
import pandas as pd
import xlsxwriter
import numpy as np
from scipy.stats import gmean
from sympy import Point, Line, Segment

Weightages = pd.read_excel("Weightages.xlsx")
Weightages_lst = list(Weightages[0])
#print(Weightages_lst)

user_sensitivity_val = int(input("Enter the % of
increment/decrement want to add for each value one at a time: "))
user_val = user_sensitivity_val/100

writer = pd.ExcelWriter('Sensitivity Weights for AHP and
Concordance.xlsx', engine='xlsxwriter')
j = 0

for i in range(0,len(Weightages_lst)):
    Weightages_1 = Weightages_lst[:]
    sens_val_inc_dec = Weightages_1[i] + (Weightages_1[i] *
user_val)
    Weightages_1[i] = sens_val_inc_dec
    globals()["sens_wts" + str(i)] = Weightages_1
    weights_df = globals()["sens_wts" + str(i)]
    weights_df = pd.DataFrame(weights_df)

    j += 1
    weights_df.to_excel(writer, sheet_name = "sens_wts" + str(j))

writer.save()
```

CONCORDANCE CODE

```
FieldData = pd.read_excel("Field Data Concordance.xlsx")
FieldData = FieldData.set_index("STRETCH NO.")
#print(FieldData)
```

```
concordance = pd.ExcelWriter('Concordance rankings.xlsx', 
engine='xlsxwriter')
j = 1
for i in range(0,len(Weightages_lst)):
    sens_wts = pd.read_excel("Sensitivity Weights for AHP and 
Concordance.xlsx", sheet_name ="sens_wts" + str(j))
    sens_wts = list(sens_wts.iloc[:,-1])
    j += 1

    stretch_comp = []
    for k in (range(len(FieldData))):
        val1 = FieldData.iloc[k,:]
        for l in range(len(FieldData)):
            val2 = FieldData.iloc[l,:]

            for l in range(len(val1)):
                if val1[l] > val2[l]:
                    comp_vals = sens_wts[l]
                    stretch_comp.append(comp_vals)
                else:
                    #print(0)
                    stretch_comp.append(0)

    #print(stretch_comp)
    n = len(stretch_comp)// len(FieldData.columns)
    D1_array = np.array(stretch_comp)
    D2_array = np.reshape(D1_array, (n, len(FieldData.columns)))
    #print(D2_array)

    stretch_Df = pd.DataFrame(D2_array, columns = 
FieldData.columns)

    row_diff = ['m' + str(i) + "-" + "m" + str(j) for i in 
range(1, len(FieldData)+1) for j in range(1, len(FieldData)+1)]
    stretch_Df["mi-mj"] = row_diff
    stretch_Df = stretch_Df.set_index("mi-mj")

    sDF_rowWise_Sum = stretch_Df.loc[:].sum(axis = 1)
    stretch_Df["SUM"] = sDF_rowWise_Sum
    #print(stretch_Df)

    SUM_vals = np.array(stretch_Df["SUM"].tolist())
    SUM_vals_2D = np.reshape(SUM_vals, (len(FieldData), 
len(FieldData)))

    Ranking = pd.DataFrame(SUM_vals_2D)
    Ranking["STRETCH NO."] = FieldData.index
    Ranking = Ranking.set_index("STRETCH NO.")
```

```
    Ranking_rowWise_Sum = Ranking.loc[:].sum(axis = 1)
    Ranking["SUM"] = Ranking_rowWise_Sum
    #print(Ranking)

    Concordance_Ranking = pd.DataFrame(list(Ranking["SUM"]),
columns = ["SUM"])
    Concordance_Ranking["STRETCH NO."] = Ranking.index
    Concordance_Ranking = Concordance_Ranking.set_index("STRETCH
NO.")
    Concordance_Ranking["Rank"] =
Concordance_Ranking["SUM"].rank(ascending = False, method =
'max')
    Concordance_Ranking.to_excel(concordance, sheet_name =
"conc_rank" + str(i+1) + "_" +str(user_sensitivity_val))
    #print(str(i+1), Concordance_Ranking)
concordance.save()
```

AHP CODE

```
C_check_matrix_WSV = pd.read_excel("Consistency Check
Matrix.xlsx")
C_check_matrix_WSV = C_check_matrix_WSV.set_index("CRITERIA")
Len_Matrix = len(C_check_matrix_WSV)
#print(Len_Matrix)
#print(C_check_matrix_WSV)

Normalized_Data = pd.read_excel("Check Normalized Data.xlsx")
Normalized_Data = Normalized_Data.set_index("STRETCH NO.")
Normalized_Data_Matrix = np.array(Normalized_Data)
#print(Normalized_Data_Matrix)

AHP = pd.ExcelWriter('AHP rankings.xlsx', engine='xlsxwriter')
j = 1
for i in range(0,len(Weightages_lst)):
    sens_wts = pd.read_excel("Sensitivity Weights for AHP and
Concordance.xlsx", sheet_name ="sens_wts" + str(j))
    sens_wts = list(sens_wts.iloc[:,-1])
    j += 1

    C_check_matrix_WSV["Criteria Weights"] = list(sens_wts)
    #print(C_check_matrix_WSV)

    C_check_matrix_WSV['WSV / CW'] = C_check_matrix_WSV['WSV'] /
C_check_matrix_WSV['Criteria Weights']
    λmax = gmean(C_check_matrix_WSV["WSV / CW"])

    Consistency_index = (λmax - Len_Matrix) / (Len_Matrix - 1)

    n = Len_Matrix
```

```
    if n > 0 and n < 51:
        Random_Index_vals={1:0, 2:0, 3:0.58, 4:0.90, 5:1.12,
6:1.24, 7:1.32, 8:1.41, 9:1.45, 10:1.49, 11:1.51, 12:1.48,
13:1.56, 14:1.57, 15:1.59,
                           16:1.6, 17:1.61, 18:1.61, 19:1.62,
20:1.63, 21:1.63, 22:1.64, 23:1.65, 24:1.65, 25:1.66, 26:1.66,
27:1.66, 28:1.67, 29:1.67, 30:1.67,
                           31:1.67, 32:1.68, 33:1.68, 34:1.68,
35:1.68, 36:1.69, 37:1.69, 38:1.69, 39:1.69, 40: 1.69, 41:1.70,
42:1.70, 43:1.70, 44:1.70, 45:1.70,
                           46:1.70, 47:1.70, 48:1.70, 49:1.71,
50:1.71}

        Random_Index = Random_Index_vals.get(n)
    else:
        Random_Index = float(input("Enter the Random Index
according to no. of criterias: "))


    Consistency_Ratio = Consistency_index / Random_Index

    if Consistency_Ratio < 0.10:
        Consistency_Ratio = np.dot(Normalized_Data_Matrix,
sens_wts)
        AHP_Ranking = pd.DataFrame()
        AHP_Ranking["Priority Index"] = Consistency_Ratio
        AHP_Ranking["STRETCHES"] = Normalized_Data.index
        AHP_Ranking = AHP_Ranking.set_index("STRETCHES")
        AHP_Ranking["Rank"] = AHP_Ranking["Priority
Index"].rank(ascending = False, method = 'max')
        AHP_Ranking.to_excel(AHP, sheet_name = "ahp_rank" +
str(i+1) + "_" +str(user_sensitivity_val))
        #print(str(i+1), AHP_Ranking)
    else:
        print('The provided metrics are "Inconsistent".')

AHP.save()
```

SENSITIVITY CODE FOR TOPSIS

```
Fuzzy_wts = pd.read_excel("Fuzzy wts.xlsx")
Fuzzy_wts = Fuzzy_wts.set_index("Criteria")
#print(len(Fuzzy_wts))
Fuzzy_wts_list = list(Fuzzy_wts[2])
#print(Fuzzy_wts_list)

writer1 = pd.ExcelWriter('Sensitivity Weights for TOPSIS.xlsx',
engine='xlsxwriter')
j = 0
```

```
for i in range(0,len(Fuzzy_wts)):
    Fuzzy_wts_list_1 = Fuzzy_wts_list[:]
    Fuzzy_wts_1 = Fuzzy_wts[:]
    sens_val_inc_dec = Fuzzy_wts_list_1[i] + (Fuzzy_wts_list_1[i]
* user_val)
    Fuzzy_wts_list_1[i] = sens_val_inc_dec
    globals()["sens_wts" + str(i)] = Fuzzy_wts_list_1
    weights_df = globals()["sens_wts" + str(i)]

    Fuzzy_wts_1[2] = weights_df
    weights_df_comp = pd.DataFrame(Fuzzy_wts_1)
    #print(weights_df_comp)

    j += 1
    weights_df_comp.to_excel(writer1, sheet_name =
"sens_wts_TOPSIS" + str(j))

writer1.save()
```

TOPSIS CODE

```
Normalized_data_topsis = pd.read_excel("Normalized Data
Sheet.xlsx")
Normalized_data_topsis =
Normalized_data_topsis.set_index("STRETCH NO.")
#print(Normalized_data_topsis)

Rating_Matrix = []
for i in range(0, len(Normalized_data_topsis.columns)):
    for j in range(0, len(Normalized_data_topsis)):
        if Normalized_data_topsis.iloc[j,i] >= 0 and
Normalized_data_topsis.iloc[j,i] <= 10:
            Rating_Matrix.append(1)
        elif Normalized_data_topsis.iloc[j,i] >= 11 and
Normalized_data_topsis.iloc[j,i] <= 20:
            Rating_Matrix.append(2)
        elif Normalized_data_topsis.iloc[j,i] >= 21 and
Normalized_data_topsis.iloc[j,i] <= 30:
            Rating_Matrix.append(3)
        elif Normalized_data_topsis.iloc[j,i] >= 31 and
Normalized_data_topsis.iloc[j,i] <= 40:
            Rating_Matrix.append(4)
        elif Normalized_data_topsis.iloc[j,i] >= 41 and
Normalized_data_topsis.iloc[j,i] <= 50:
            Rating_Matrix.append(5)
        elif Normalized_data_topsis.iloc[j,i] >= 51 and
Normalized_data_topsis.iloc[j,i] <= 60:
            Rating_Matrix.append(6)
        elif Normalized_data_topsis.iloc[j,i] >= 61 and
```

```
Normalized_data_topsis.iloc[j,i] <= 70:
                Rating_Matrix.append(7)
            elif Normalized_data_topsis.iloc[j,i] >= 71 and
Normalized_data_topsis.iloc[j,i] <= 80:
                Rating_Matrix.append(8)
            elif Normalized_data_topsis.iloc[j,i] >= 81 and
Normalized_data_topsis.iloc[j,i] <= 90:
                Rating_Matrix.append(9)
            elif Normalized_data_topsis.iloc[j,i] >= 91 and
Normalized_data_topsis.iloc[j,i] <= 100:
                Rating_Matrix.append(10)

cols = len(Normalized_data_topsis)
rows = len(Normalized_data_topsis.columns)
Rating_Matrix = np.array(Rating_Matrix)
Rating_Matrix = Rating_Matrix.reshape(rows,cols)
Rating_Matrix = Rating_Matrix.transpose()
#print(Rating_Matrix)

Topsis = pd.ExcelWriter('Fuzzy TOPSIS rankings.xlsx',
engine='xlsxwriter')
b = 1
for a in range(0,len(Fuzzy_wts)):
    sens_wts_topsis = pd.read_excel("Sensitivity Weights for
TOPSIS.xlsx", sheet_name ="sens_wts_TOPSIS" + str(b))
    sens_wts_topsis = sens_wts_topsis.set_index("Criteria")
    #print(sens_wts_topsis)
    b += 1

    Rating_Matrix = np.array(Rating_Matrix)
    Fuzzy_wts = np.array(sens_wts_topsis)
    Fuzzy_Eval_vals = np.dot(Rating_Matrix, sens_wts_topsis)
    Fuzzy_Eval_vals = pd.DataFrame(Fuzzy_Eval_vals)
    Fuzzy_Eval_vals.columns = ["l","m","n"]
    #print(Fuzzy_Eval_vals)


    row_index = ['A' + str(i) for i in range(1,
len(Fuzzy_Eval_vals)+1)]
    Fuzzy_Eval_vals["Stretch No."] = row_index
    Fuzzy_Eval_vals_1 = Fuzzy_Eval_vals.set_index("Stretch No.")
    #print(Fuzzy_Eval_vals_1)


    Triangular_FuzzyNums = pd.DataFrame(columns = ['l','m','n'])
    for e in range(0,len(Fuzzy_Eval_vals)):
        for f in range(0,len(Fuzzy_Eval_vals)):
            arr1 = Fuzzy_Eval_vals.loc[e]
            arr2 = Fuzzy_Eval_vals.loc[f]
```

```
            diff = [arr1[0]-arr2[2],arr1[1]-arr2[1],arr1[2]-
arr2[0]]
            stretch_diff = pd.Series(diff, index =
Triangular_FuzzyNums.columns)
            Triangular_FuzzyNums =
Triangular_FuzzyNums.append(stretch_diff, ignore_index=True)
    row_diff = ['p' + str(e) + "-" + "p" + str(f) for e in
range(1, len(Fuzzy_Eval_vals)+1) for f in range(1,
len(Fuzzy_Eval_vals)+1)]
    Triangular_FuzzyNums["p~i-p~j"] = row_diff
    Triangular_FuzzyNums_1 = Triangular_FuzzyNums.set_index("p~i-
p~j")
    #print(Triangular_FuzzyNums_1)


    Intersection_pnts = []
    for k in range(0,len(Triangular_FuzzyNums)):
        l = Triangular_FuzzyNums.iloc[k,0]
        m = Triangular_FuzzyNums.iloc[k,1]
        n = Triangular_FuzzyNums.iloc[k,2]

        if l < 0 and m > 0:
            p1, p2, p3, p4 = Point(0, 0), Point(0, 1), Point(l,
0), Point(m, 1)
            l1 = Line(p1, p2)
            s1 = Segment(p3, p4)
            showIntersection = l1.intersection(s1)
            #print(showIntersection)
            Intersection_pnts.append(showIntersection)
        elif m < 0 and n > 0:
            p1, p2, p3, p4 = Point(0, 0), Point(0, 1), Point(n,
0), Point(m, 1)
            l1 = Line(p1, p2)
            s1 = Segment(p3, p4)
            showIntersection = l1.intersection(s1)
            #print(showIntersection)
            Intersection_pnts.append(showIntersection)
        else:
            Intersection_pnts.append([[0,0]])


    Intersection = []
    for z in range(0,len(Triangular_FuzzyNums_1)):
        Intersection.append(Intersection_pnts[z][0][1])

    Triangular_FuzzyNums_1["Intersection height"] = Intersection
    #print(Triangular_FuzzyNums_1)
```

```
    E = []
    for d in range(0,len(Triangular_FuzzyNums)):
        l = Triangular_FuzzyNums_1.iloc[d,0]
        m = Triangular_FuzzyNums_1.iloc[d,1]
        n = Triangular_FuzzyNums_1.iloc[d,2]
        h = round(Triangular_FuzzyNums_1.iloc[d,3], 3)

        if l < 0 and m > 0 and n > 0:
            pos_Area = (1/2 * (n-m)) + (1/2 * (m) * (1-h)) + (h *
m)
            neg_Area = 1/2 * (0-l) * h
        elif m < 0 and n > 0:
            pos_Area = 1/2 * (n) * h
            neg_Area = (1/2 * abs((m-l))) + (1/2 * (0-m) * (1-h))
+ ((0-m) * h)
        elif n < 0:
            pos_Area = 0
            neg_Area = (1/2 * (n-m)) + (1/2 * (m-l))
        else:
            pos_Area = (1/2 * (n-m)) + (1/2 * (m-l))
            neg_Area = 0

        total_Area = pos_Area + neg_Area
        corr_e = round(pos_Area/total_Area, 3)
        E.append(corr_e)

    E_array = np.array(E)
    TriFuzzNo_sqrt = int(np.sqrt(len(Triangular_FuzzyNums)))
    E_2D_array = np.reshape(E_array, (TriFuzzNo_sqrt,
TriFuzzNo_sqrt))
    np.fill_diagonal(E_2D_array, 0.5)
    #print(E_2D_array)
    PI = []
    for x in range(0,len(E_2D_array)):
        PI_Sum = sum(E_2D_array[x]) - (len(E_2D_array) * 0.5)
        PI.append(PI_Sum)

    Ranking = pd.DataFrame()
    Ranking["Priority Index"] = PI
    Ranking["Stretch No."] = row_index
    Ranking["Rank"] = Ranking["Priority Index"].rank(ascending =
False, method = 'max')
    Ranking = Ranking.set_index("Stretch No.")

    #print(str(a+1), Ranking)
    Ranking.to_excel(Topsis, sheet_name = "topsis_rank" +
str(a+1) + "_" +str(user_sensitivity_val))
Topsis.save()
```

C.5 ALGORITHM FOR PEARSON'S COEFFICIENT– WITH OUTPUTS

The algorithm below is written to find the correlation between the sensitive analysis values obtained for each method, varying the weightages by 5%, 10%, -5%, -10%, for each weightage iteratively, which is written in python programming language.

```
#Importing required modules.
import pandas as pd
import numpy as np


#Importing excel sheet of original rankings.
Topsis = pd.read_excel("Fuzzy_TOPSIS_Ranking.xlsx")
Topsis_lst = Topsis.iloc[:,-1].to_list()
AHP = pd.read_excel("AHPRanking.xlsx")
AHP_lst = AHP.iloc[:,-1].to_list()
Concordance = pd.read_excel("Concordance Ranking.xlsx")
Concordance_lst = Concordance.iloc[:,-1].to_list()
```

TOPSIS

```
xl_Topsis = pd.ExcelFile("Fuzzy TOPSIS rankings_5%.xlsx")
res_Topsis = len(xl_Topsis.sheet_names)
print("Total no. of weightages: ",res_Topsis)
print("\n")

Topsis_lst = Topsis.iloc[:,-1].to_list()

col_1_topsis = []
col_2_topsis = []
col_3_topsis = []
col_4_topsis = []
for i in range(1, res_Topsis+1):
    Topsis_lst_1 = np.array(Topsis_lst[:])
    topsis_wts_inc_5 = pd.read_excel("Fuzzy TOPSIS 
rankings_5%.xlsx", sheet_name = "topsis_rank"+str(i)+"_5")
    topsis_wts_inc_5 = list(topsis_wts_inc_5.iloc[:,-1])
    topsis_wts_inc_10 = pd.read_excel("Fuzzy TOPSIS 
rankings_10%.xlsx", sheet_name = "topsis_rank"+str(i)+"_10")
    topsis_wts_inc_10 = list(topsis_wts_inc_10.iloc[:,-1])
    topsis_wts_dec_5 = pd.read_excel("Fuzzy TOPSIS rankings_-
```

```
5%.xlsx", sheet_name = "topsis_rank"+str(i)+"_-5")
    topsis_wts_dec_5 = list(topsis_wts_dec_5.iloc[:,-1])
    topsis_wts_dec_10 = pd.read_excel("Fuzzy TOPSIS rankings_-
10%.xlsx", sheet_name = "topsis_rank"+str(i)+"_-10")
    topsis_wts_dec_10 = list(topsis_wts_dec_10.iloc[:,-1])

    x_arr = Topsis_lst_1

    y_arr = np.array(topsis_wts_inc_5)
    r_5_inc = np.corrcoef(x_arr,y_arr)
    col_1_topsis.append(r_5_inc[0][1])

    y_arr = np.array(topsis_wts_inc_10)
    r_10_inc = np.corrcoef(x_arr,y_arr)
    col_2_topsis.append(r_10_inc[0][1])

    y_arr = np.array(topsis_wts_dec_5)
    r_5_dec = np.corrcoef(x_arr,y_arr)
    col_3_topsis.append(r_5_dec[0][1])

    y_arr = np.array(topsis_wts_dec_10)
    r_10_dec = np.corrcoef(x_arr,y_arr)
    col_4_topsis.append(r_10_dec[0][1])

PC_Topsis = pd.DataFrame()
PC_Topsis["Inc 5%"] = col_1_topsis
PC_Topsis["Inc 10%"] = col_2_topsis
PC_Topsis["Dec 5%"] = col_3_topsis
PC_Topsis["Dec 10%"] = col_4_topsis
PC_Topsis["Weightage Increased"] = list(range(1, res_Topsis+1))
PC_Topsis = PC_Topsis.set_index("Weightage Increased")

print(PC_Topsis)
PC_Topsis.to_excel("Topsis Sensitivity Results.xlsx")
```

```
Total no. of weightages:  41


                     Inc 5%  Inc 10%  Dec 5%   Dec 10%
Weightage Increased
1                       1.0      1.0     1.0  1.000000
2                       1.0      1.0     1.0  1.000000
3                       1.0      1.0     1.0  1.000000
4                       1.0      1.0     1.0  1.000000
5                       1.0      1.0     1.0  1.000000
6                       1.0      1.0     1.0  1.000000
7                       1.0      1.0     1.0  1.000000
```

```
8                        1.0      1.0      1.0  1.000000
9                        1.0      1.0      1.0  1.000000
10                       1.0      1.0      1.0  1.000000
11                       1.0      1.0      1.0  1.000000
12                       1.0      1.0      1.0  1.000000
13                       1.0      1.0      1.0  1.000000
14                       1.0      1.0      1.0  1.000000
15                       1.0      1.0      1.0  1.000000
16                       1.0      1.0      1.0  1.000000
17                       1.0      1.0      1.0  1.000000
18                       1.0      1.0      1.0  1.000000
19                       1.0      1.0      1.0  1.000000
20                       1.0      1.0      1.0  1.000000
21                       1.0      1.0      1.0  1.000000
22                       1.0      1.0      1.0  1.000000
23                       1.0      1.0      1.0  1.000000
24                       1.0      1.0      1.0  1.000000
25                       1.0      1.0      1.0  1.000000
26                       1.0      1.0      1.0  1.000000
27                       1.0      1.0      1.0  1.000000
28                       1.0      1.0      1.0  1.000000
29                       1.0      1.0      1.0  1.000000
30                       1.0      1.0      1.0  1.000000
31                       1.0      1.0      1.0  1.000000
32                       1.0      1.0      1.0  1.000000
33                       1.0      1.0      1.0  1.000000
34                       1.0      1.0      1.0  1.000000
35                       1.0      1.0      1.0  1.000000
36                       1.0      1.0      1.0  0.998701
37                       1.0      1.0      1.0  1.000000
38                       1.0      1.0      1.0  1.000000
39                       1.0      1.0      1.0  1.000000
40                       1.0      1.0      1.0  1.000000
41                       1.0      1.0      1.0  1.000000
```

```
AHP
xl_AHP = pd.ExcelFile("AHP rankings_%5.xlsx")
res_AHP = len(xl_AHP.sheet_names)
print("Total no. of weightages: ",res_AHP)
print("\n")

AHP_lst = AHP.iloc[:,-1].to_list()

col_1_AHP = []
col_2_AHP = []
col_3_AHP = []
```

```
col_4_AHP = []
for i in range(1, res_AHP+1):
    AHP_lst_1 = np.array(AHP_lst[:])
    AHP_wts_inc_5 = pd.read_excel("AHP rankings_%5.xlsx",
sheet_name = "ahp_rank"+str(i)+"_5")
    AHP_wts_inc_5 = list(AHP_wts_inc_5.iloc[:,-1])
    AHP_wts_inc_10 = pd.read_excel("AHP rankings_%10.xlsx",
sheet_name = "ahp_rank"+str(i)+"_10")
    AHP_wts_inc_10 = list(AHP_wts_inc_10.iloc[:,-1])
    AHP_wts_dec_5 = pd.read_excel("AHP rankings_-5%.xlsx",
sheet_name = "ahp_rank"+str(i)+"_-5")
    AHP_wts_dec_5 = list(AHP_wts_dec_5.iloc[:,-1])
    AHP_wts_dec_10 = pd.read_excel("AHP rankings_-10%.xlsx",
sheet_name = "ahp_rank"+str(i)+"_-10")
    AHP_wts_dec_10 = list(AHP_wts_dec_10.iloc[:,-1])


    x_arr = AHP_lst_1
    y_arr = np.array(AHP_wts_inc_5)
    r_5_inc = np.corrcoef(x_arr,y_arr)
    col_1_AHP.append(r_5_inc[0][1])

    x_arr = AHP_lst_1
    y_arr = np.array(AHP_wts_inc_10)
    r_10_inc = np.corrcoef(x_arr,y_arr)
    col_2_AHP.append(r_10_inc[0][1])

    x_arr = AHP_lst_1
    y_arr = np.array(AHP_wts_dec_5)
    r_5_dec = np.corrcoef(x_arr,y_arr)
    col_3_AHP.append(r_5_dec[0][1])

    x_arr = AHP_lst_1
    y_arr = np.array(AHP_wts_dec_10)
    r_10_dec = np.corrcoef(x_arr,y_arr)
    col_4_AHP.append(r_10_dec[0][1])

PC_AHP = pd.DataFrame()
PC_AHP["Inc 5%"] = col_1_AHP
PC_AHP["Inc 10%"] = col_2_AHP
PC_AHP["Dec 5%"] = col_3_AHP
PC_AHP["Dec 10%"] = col_4_AHP
PC_AHP["Weightage Increased"] = list(range(1, res_AHP+1))
PC_AHP = PC_AHP.set_index("Weightage Increased")

print(PC_AHP)
PC_AHP.to_excel("AHP Sensitivity Results.xlsx")
```

```
Total no. of weightages:  17


                        Inc 5%  Inc 10%  Dec 5%  Dec 10%
Weightage Increased
1                          1.0      1.0     1.0      1.0
2                          1.0      1.0     1.0      1.0
3                          1.0      1.0     1.0      1.0
4                          1.0      1.0     1.0      1.0
5                          1.0      1.0     1.0      1.0
6                          1.0      1.0     1.0      1.0
7                          1.0      1.0     1.0      1.0
8                          1.0      1.0     1.0      1.0
9                          1.0      1.0     1.0      1.0
10                         1.0      1.0     1.0      1.0
11                         1.0      1.0     1.0      1.0
12                         1.0      1.0     1.0      1.0
13                         1.0      1.0     1.0      1.0
14                         1.0      1.0     1.0      1.0
15                         1.0      1.0     1.0      1.0
16                         1.0      1.0     1.0      1.0
17                         1.0      1.0     1.0      1.0
```

CONCORDANCE

```
xl_conc = pd.ExcelFile("Concordance rankings_%5.xlsx")
res_conc = len(xl_conc.sheet_names)
print("Total no. of weightages: ",res_conc)
print("\n")

conc_lst = Concordance.iloc[:,-1].to_list()

col_1_conc = []
col_2_conc = []
col_3_conc = []
col_4_conc = []
for i in range(1, res_conc+1):
    conc_lst_1 = np.array(conc_lst[:])
    conc_wts_inc_5 = pd.read_excel("Concordance
rankings_%5.xlsx", sheet_name = "conc_rank"+str(i)+"_5")
    conc_wts_inc_5 = list(conc_wts_inc_5.iloc[:,-1])
    conc_wts_inc_10 = pd.read_excel("Concordance
rankings_%10.xlsx", sheet_name = "conc_rank"+str(i)+"_10")
    conc_wts_inc_10 = list(conc_wts_inc_10.iloc[:,-1])
    conc_wts_dec_5 = pd.read_excel("Concordance rankings_-
5%.xlsx", sheet_name = "conc_rank"+str(i)+"_-5")
    conc_wts_dec_5 = list(conc_wts_dec_5.iloc[:,-1])
```

```
    conc_wts_dec_10 = pd.read_excel("Concordance rankings_-
10%.xlsx", sheet_name = "conc_rank"+str(i)+"_-10")
    conc_wts_dec_10 = list(conc_wts_dec_10.iloc[:,-1])

    x_arr = conc_lst_1

    y_arr = np.array(conc_wts_inc_5)
    r_5_inc = np.corrcoef(x_arr,y_arr)
    col_1_conc.append(r_5_inc[0][1])

    y_arr = np.array(conc_wts_inc_10)
    r_10_inc = np.corrcoef(x_arr,y_arr)
    col_2_conc.append(r_10_inc[0][1])

    y_arr = np.array(conc_wts_dec_5)
    r_5_dec = np.corrcoef(x_arr,y_arr)
    col_3_conc.append(r_5_dec[0][1])

    y_arr = np.array(conc_wts_dec_10)
    r_10_dec = np.corrcoef(x_arr,y_arr)
    col_4_conc.append(r_10_dec[0][1])

PC_conc = pd.DataFrame()
PC_conc["Inc 5%"] = col_1_conc
PC_conc["Inc 10%"] = col_2_conc
PC_conc["Dec 5%"] = col_3_conc
PC_conc["Dec 10%"] = col_4_conc
PC_conc["Weightage Increased"] = list(range(1, res_conc+1))
PC_conc = PC_conc.set_index("Weightage Increased")

print(PC_conc)
PC_conc.to_excel("Concordance Sensitivity Results.xlsx")
```

```
Total no. of weightages:  17
```

```
                     Inc 5%   Inc 10%    Dec 5%   Dec 10%
Weightage Increased
1                  0.836364  0.836364  0.842857  0.842857
2                  0.836364  0.836364  0.836364  0.836364
3                  0.836364  0.836364  0.836364  0.836364
4                  0.836364  0.836364  0.836364  0.836364
5                  0.836364  0.836364  0.836364  0.836364
6                  0.842857  0.842857  0.836364  0.836364
7                  0.842857  0.846753  0.836364  0.836364
```

8	0.836364	0.842857	0.836364	0.836364
9	0.836364	0.836364	0.836364	0.836364
10	0.836364	0.836364	0.836364	0.836364
11	0.836364	0.836364	0.836364	0.836364
12	0.836364	0.836364	0.836364	0.836364
13	0.836364	0.836364	0.842857	0.842857
14	0.836364	0.836364	0.836364	0.835065
15	0.836364	0.836364	0.833766	0.833766
16	0.842857	0.840260	0.836364	0.836364
17	0.836364	0.836364	0.842857	0.842857