Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 1s (2025)

Design and Optimization of Self-Healing Concrete Mix Proportions Using MATLAB

P.Neeraja*, Bala Padmaja S[1] and M.Padmaja[2], A.Sujathaa[4]

[1]Assistant Professor, Mahatma Gandhi Institute of Technology, Hyderabad,
pneeraja_civil@mgit.ac.in

Article History:

Received: 26-08-2024

Revised: 10-12-2024

Accepted: 05-01-2025

Abstract:

Concrete is a heterogeneous material made up of several constituents, each imparting unique characteristics that together determine its overall quality. Achieving desired properties like workability, strength, and durability requires a systematic approach known as concrete mix design, which involves selecting suitable materials and determining their proper proportions. Accurate proportioning not only enhances performance but also helps in reducing construction costs.

The mix design for self-healing concrete follows the provisions of IS 10262:2019. However, performing this process manually is often tedious and prone to errors because it involves interpreting data from multiple tables and performing interpolations to derive intermediate values. Such manual calculations increase the likelihood of inaccuracies.

To address these challenges and improve efficiency, a computer-aided mix design program has been developed. This software cross-verifies results obtained from manual methods, ensuring consistency and precision. The main goal of this project is to design a reliable computational tool capable of quickly generating accurate mix proportions for self-healing concrete that integrates materials like Alccofine and nylon fibers as partial cement replacements, along with an additional crystalline admixture to enhance the self-healing capability.

1. Literature Review:

A study by **A. R. Makenya and M. John Paul** highlights the critical influence of concrete mix design on the overall performance of engineered concrete structures. Traditionally, mix design has been carried out through manual computation or spreadsheet-based methods, both of which are time-consuming and susceptible to human error. To overcome these limitations, the researchers created a MATLAB-based application specifically designed for high-strength concrete (HSC) mix design.

Their program was developed based on the **Erntroy and Shacklock (1954)** mix design method. By converting the method's tabular and graphical information into mathematical expressions, they formulated an algorithm that simplified the mix design process. This algorithm, along with certain predefined assumptions, was incorporated into the MATLAB software.

To verify its accuracy and reliability, the program was tested using examples from established research sources. The outcomes obtained from MATLAB were found to be in close agreement with those derived through manual calculations, while the program required significantly less computation time. The study demonstrated that the MATLAB-based approach can serve as an efficient and dependable substitute for conventional mix design methods, emphasizing the growing importance of **computer-assisted tools** in improving both the precision and productivity of concrete design.

In another research work titled **"Enhancing Self-Healing Concrete Mix Design through Computational Intelligence Techniques"** by **Jain et al. (2018)**, the authors introduced an innovative method for optimizing self-healing concrete mix design. The study employed computational intelligence approaches—including genetic algorithms **and** neural networks implemented in MATLAB—to refine the proportions of healing agents, aggregates, and other essential materials. Experimental results confirmed that this technique significantly improved the mechanical performance and durability of self-healing concrete, demonstrating the potential of intelligent computational methods in advancing modern concrete technology.

In a separate endeavor, Avinash G and Chandra Mohan Rao B.D.V developed a MATLAB code specifically for designing high-strength concrete mixes. Concrete mix design is a critical process that entails selecting suitable ingredients and determining their proportions to achieve desired concrete properties. However, manual calculations for determining these proportions can be labor-intensive and prone to errors due to the variability in material properties. Hence, the researchers aimed to develop a MATLAB code following the IS 10262:2019 standard to streamline the mix design process.

Another notable contribution comes from Satnam Singh, Gurpreet Kaur, and Mandeep Kaur, who introduced a MATLAB-based tool and GUI Development Environment (GUIDE) for concrete mix design. This tool automates the mix design process, adhering to Indian Standard (IS) codes, and provides a user-friendly interface for designing concrete mixes. By abstracting data and preventing human errors, this tool enhances the efficiency and accuracy of mix design calculations.

Lastly, D. Zealakshm, A. Ravichandran, and S. Kothandraman proposed a computer-aided mix design methodology using MATLAB for high-strength concrete, following ACI 211.4R-93 guidelines. This approach automates the mix design process, minimizing human errors and improving efficiency by simplifying data interpretation.

INTRODUCTION:

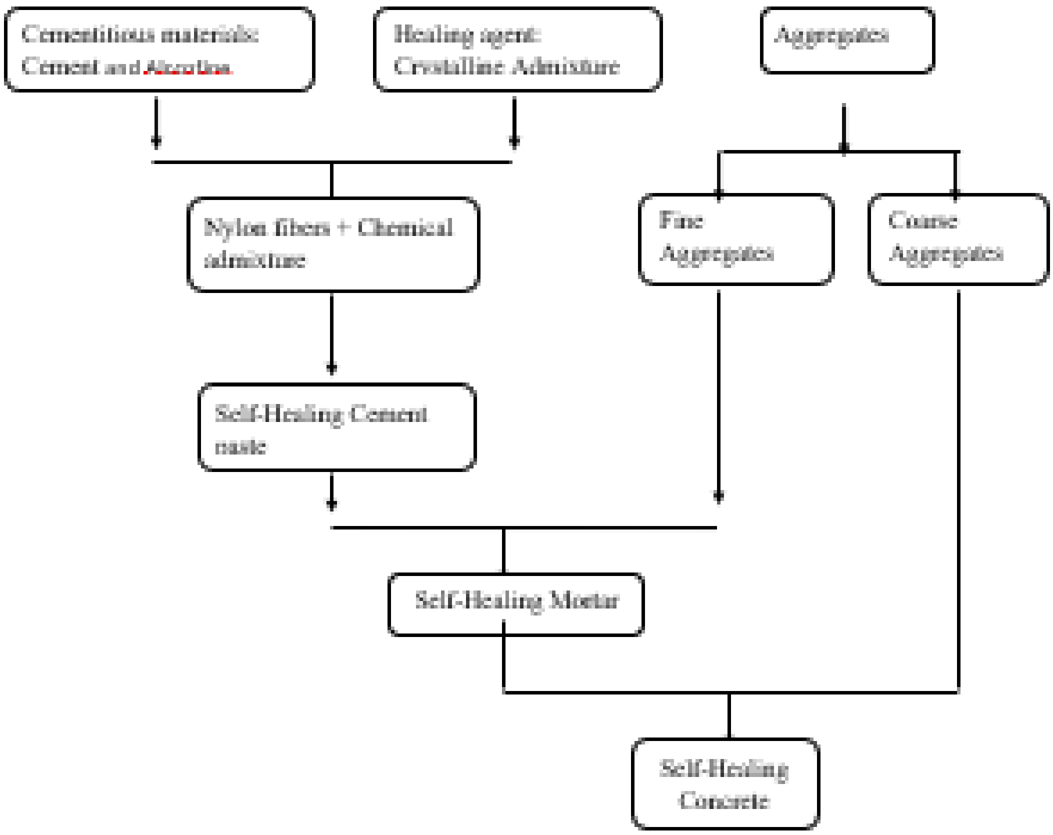
Mix design is an important process in concrete technology, which aims to determine the proportions of various components of concrete that would result in the desired properties. The process involves selecting the appropriate materials and determining the right mix proportions based on various factors, including the intended use, strength requirements, workability, and durability of the concrete. Traditionally, mix design has been carried out using empirical or statistical methods, which are often time-consuming and may not always result in the desired properties. In recent years, optimization-based approaches have gained popularity as a more efficient and accurate way to achieve optimal mix designs.

Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 1s (2025)

MATLAB provides a powerful platform for optimization-based mix design. It offers a flexible and efficient environment for formulating and solving optimization problems, which can be used to minimize the cost or maximize the performance of concrete mixes. MATLAB's optimization toolbox provides a range of algorithms for solving different types of optimization problems, including linear and nonlinear programming, constrained and unconstrained optimization, and multi-objective optimization. MATLAB can also be used to develop more accurate models for predicting the properties of concrete mixes. These models can be based on various algorithms, including support vector machines, artificial neural networks, and regression models. The accuracy of the models can be improved by incorporating more data and refining the model parameters.

Self-healing concrete stands at the forefront of innovative materials in the construction industry, offering a transformative solution to a common problem, cracks in concrete structures. Traditional concrete, while strong and durable, is prone to cracking over time due to various factors such as environmental conditions, loading, and shrinkage. These cracks not only compromise the structural integrity of the concrete but also necessitate costly and time-consuming repairs.

In response to this challenge, self-healing concrete has emerged as a groundbreaking alternative. Unlike conventional concrete, self-healing concrete possesses the ability to autonomously repair cracks, thereby extending the lifespan of structures and reducing maintenance requirements. This remarkable capability is achieved through the incorporation of specialized additives within the concrete matrix.



2. METHODOLOGY:

CODE:

3.1 Clear Memory:

To begin coding self-healing concrete in MATLAB, it's essential to start with a clean slate by clearing the previous memory in the workspace and command window. This initial step involves using the commands 'clc' and 'clear all' at the outset of every program code, as shown below.

```
clc;

clear all;
```

3.2 Self-healing Concrete Grade Parameters:

To initiate the process, it's imperative to specify the type of Self-healing concrete grade for which the mix is intended. According to IS 10262:2019 and IS 456:2000 standards, various parameters such as standard deviation, compressive strength, and water-cement ratio vary for each grade. To achieve this, the 'case' command solicits real-time input from the user, allowing selection from options like M25, M30, M35, M40, M45, and M50 concrete grades. Subsequently, parameter values are assigned based on the user's selection utilizing 'case' statements, as illustrated below:

These commands help in enhancing code readability and preventing any potential conflicts or clutter from previous executions.

```
% Prompt user to input the grade of concrete

Cemgrade = input('Enter the grade of concrete (25/30/35/40/45/50): ');

% Define standard deviation, compressive strength, and water-cement ratio based on input grade

switch Cemgrade

  case 25

    standard_deviation = 4;

    compressive_strength = 25;

    water_cement_ratio = 0.50;

  case 30

    standard_deviation = 5;

    compressive_strength = 30;

    water_cement_ratio = 0.5;

  case 35

    standard_deviation = 5;
```

```
        compressive_strength = 35;
        water_cement_ratio = 0.45;
    case 40
        standard_deviation = 5;
        compressive_strength = 40;
        water_cement_ratio = 0.42;
    case 45
        standard_deviation = 5;
        compressive_strength = 45;
        water_cement_ratio = 0.37;
    case 50
        standard_deviation = 5;
        compressive_strength = 50;
        water_cement_ratio = 0.35;
    otherwise
        warning('Incorrect value of concrete grade');
end
```

3.3 Target Strength for Mix Calculation:

The next step is to target strength of concrete using the formula given in below as per IS 10262:2019

$F'_C = F_C + 1.65S$

Where,

F'_C = Target average compressive strength at 28 days

F_C = characteristic compressive strength at 28 days

S = Standard deviation

Using the above formula, the target strength is computed in MATLAB

```
Target_Strength = Compressive_Strength + (1.65*Standard_deviation);
```

3.4 Calculate the water content and volume of coarse aggregates:

To calculate the water content and volume of coarse aggregates, the process involves first determining the zone of fine aggregates by comparing the results of a sieve analysis test with Table 3 of IS 383:2016. Based on this comparison and the user-provided nominal maximum size of aggregates (10mm, 20mm, or 40mm), the code utilizes 'case' statements to ascertain the water content per cubic meter and the volume of coarse aggregates, along with approximating the air content, as per IS 10262:2019.

```
% Prompt user to input the nominal size of the aggregate
aggsize = input('Enter the nominal size of the aggregate (10/20/40 mm): ');

% Define water content, air content, and volume of coarse aggregate based on input aggregate size
switch aggsize
   case 10
      water_content_per_cubic_meter = 208;
      approximate_air_content_in_volume = 1.5;
      fine_zone = input('Enter the zone of fine aggregates (1/2/3/4): ');
      if fine_zone >= 1 && fine_zone <= 4
         volume_of_coarse_aggregate = 0.46 + 0.02 * fine_zone;
      else
         warning('Incorrect value of fine aggregate zone');
      end
   case 20
      water_content_per_cubic_meter = 186;
      approximate_air_content_in_volume = 1.0;
      fine_zone = input('Enter the zone of fine aggregates (1/2/3/4): ');
      if fine_zone >= 1 && fine_zone <= 4
         volume_of_coarse_aggregate = 0.58 + 0.02 * fine_zone;
      else
         warning('Incorrect value of fine aggregate zone');
      end
   case 40
      water_content_per_cubic_meter = 165;
      approximate_air_content_in_volume = 0.8;
      fine_zone = input('Enter the zone of fine aggregates (1/2/3/4): ');
      if fine_zone >= 1 && fine_zone <= 4
```

```
            volume_of_coarse_aggregate = 0.67 + 0.01 * fine_zone;
        else
            warning('Incorrect value of fine aggregate zone');
        end
    otherwise
        warning('Incorrect value of aggregate size');
end
```

3.5 Revised Water Content (using Slump):

The next essential input for the design process is the slump value. This value is determined by researchers through the slump test conducted on concrete. Typically, the slump can range from 50mm to 120mm. However, there's a 3% increase in water content when the slump varies between 50mm to 75mm and a 6% increase in water content when the slump increases from 75mm to 120mm. Accordingly, the following code snippet adjusts the water content (from cubic meter to liters) based on the provided slump value:

```
% Prompt user to input the slump value
slump = input('Enter the value of slump between 50 to 120 (in mm): ');

% Define water content required based on input slump value
if slump == 50
    water_content_required_in_litre = water_content_per_cubic_meter;
elseif slump > 50 && slump <= 75
    water_content_required_in_litre = water_content_per_cubic_meter + 0.03 * water_content_per_cubic_meter;
elseif slump > 75 && slump <= 120
    water_content_required_in_litre = water_content_per_cubic_meter + 0.084 * water_content_per_cubic_meter;
else
    warning('Incorrect value of slump');
end
```

3.6 Exposure Conditions:

Exposure conditions are pivotal factors considered in concrete design as per the IS456:2000 code. This code specifies the minimum cement content required for various exposure conditions in both plain and reinforced concrete. Additionally, it outlines the maximum permissible water-cement ratio and the minimum grade of concrete suitable for each exposure condition. Adhering to these provisions ensures that concrete structures meet the necessary standards for durability and performance under specific environmental circumstances.

matlab

```
% Prompt user to input the exposure condition

exposure_condition = input('Enter the exposure condition (mild/moderate/severe/very severe/extreme): ', 's');

% Define minimum cement content based on input exposure condition

switch lower(exposure_condition)

   case 'mild'

      minimum_cement_content_in_kg_per_m3 = 300;

   case 'moderate'

      minimum_cement_content_in_kg_per_m3 = 300;

   case 'severe'

      minimum_cement_content_in_kg_per_m3 = 320;

   case 'very severe'

      minimum_cement_content_in_kg_per_m3 = 340;

   case 'extreme'

      minimum_cement_content_in_kg_per_m3 = 360;

   otherwise

      warning('Incorrect value of exposure condition');

end
```

3.7 Chemical Admixture Percentage Input and Water Content Reduction:

In many construction and concrete applications, chemical admixtures are utilized to enhance the properties of concrete. These admixtures play a crucial role in modifying various aspects of concrete, including its workability, strength, durability, and setting time. Among the significant parameters associated with chemical admixtures is the percentage of their incorporation into the concrete mix. This document outlines a program segment designed to prompt users for the percentage of chemical admixture and calculate the corresponding reduction in water content based on the input percentage.

matlab

```
chemical_admixture_percent = input('Enter the percentage of chemical admixture (10/15/20/23/25/30): ');
```

3.8 Calculating Water Content Reduction:

Upon receiving the input percentage of chemical admixture, the program segment proceeds to calculate the reduction in water content. This reduction is determined based on a predefined relationship between the percentage of chemical admixture and the corresponding water content reduction percentage.

```
% Prompt user to input the percentage of chemical admixture
chemical_admixture_percent = input('Enter the percentage of chemical admixture (10/15/20/23/25/30): ');


% Define water content reduction percent based on input percentage of chemical admixture
switch chemical_admixture_percent
   case {10, 15, 20, 23, 25, 30}
      water_content_reduction_percent = 1 - chemical_admixture_percent / 100;
   otherwise
      warning('Incorrect value of water content reduction percent');
end
```

The presented program segment facilitates the integration of chemical admixtures into concrete mixtures by allowing users to specify the desired percentage of admixture. Subsequently, it calculates the corresponding reduction in water content, which is pivotal for maintaining the desired concrete properties. This document serves as a guide for developers and engineers involved in concrete mix design and construction projects.

Note:

It's important to ensure that the input values for the chemical admixture percentage align with the predefined options (10, 15, 20, 23, 25, or 30) to guarantee accurate calculations. In case of an incorrect input, the program issues a warning to prompt users to provide a valid percentage.

```
% Calculate final water content and cement content required
final_water_content = water_content_required_in_litre * water_content_reduction_percent;
cement_content_required_in_kg_per_m3 = final_water_content / water_cement_ratio;
```

3.9 Enhancing Concrete Mixture with Additional Cementitious and Admixture Materials:

Concrete mix design is a crucial process in construction, aiming to achieve desired properties and performance characteristics. This document presents a program segment designed to increase cementitious material content by 10%, incorporate mineral admixtures (specifically Alccofine material), nylon fibers, and crystalline admixtures into the concrete mixture. The purpose is to enhance the strength, durability, and other desirable properties of the concrete.

3.10 Increasing Cementitious Material Content:

The initial step in the program segment involves increasing the cementitious material content by 10%. This adjustment is essential for improving the overall strength and durability of the concrete mixture.

matlab

```
cementitious_material_content = cement_content_required_in_kg_per_m3 * 1.10;
```

3.11 Incorporating Mineral Admixtures:

Mineral admixtures such as Alccofine material are known for their ability to enhance various properties of concrete. In this segment, users are prompted to input the percentage of Alccofine material and nylon fibers they intend to incorporate into the mixture.

matlab

```
x = input('Enter the percentage of Alccofine material (x):');

y = input('Enter the percentage of nylon fibers (y):');
```

3.12 Including Crystalline Admixture:

Additionally, the program allows users to input the percentage of crystalline admixture (CA) to be included in the concrete mix. Crystalline admixtures contribute to the waterproofing and durability of concrete structures.

matlab

```
CA = input('Enter the percentage of crystalline admixture (CA):');
```

3.13 Calculating Final Cement Content:

Based on the input percentages of Alccofine material, nylon fibers, and crystalline admixture, the program calculates the final cement content by subtracting the contributions of these additional materials from the increased cementitious material content.

matlab

```
Alccofine_material = cementitious_material_content * (x / 100);

Nylon_fibers = cementitious_material_content * (y / 100);

final_cement_content = cementitious_material_content - Alccofine_material - Nylon_fibers;
```

The presented program segment offers a systematic approach to enhance concrete mixtures by increasing cementitious material content and incorporating mineral and crystalline admixtures. By allowing users to specify the percentages of these additional materials, the program facilitates tailored concrete mix designs that meet specific project requirements for strength, durability, and performance.

Note:

It is essential for users to input accurate percentages of Alccofine material, nylon fibers, and crystalline admixture to ensure the desired properties of the concrete mixture are achieved. Additionally, adherence to recommended guidelines for material proportions is recommended for optimal concrete performance.

```
% increase of 10 percent cementitious material content

cementitious_material_content = cement_content_required_in_kg_per_m3 * 1.10;

% mineral admixture(alccofine material,Fly ash, GGBS, Metakaoline and Silica fume)

% here considered alccofine admixture and nylon fibers

x = input ('Enter the percentage of Alccofine material (x):');

y = input ('enter the percentage of nylon fibers(y):');

% crystalline admixture as additional material

CA = input ('enter the percentage of crystalline admixture(CA):');

Alccofine_material = cementitious_material_content * (x / 100);

Nylon_fibers = cementitious_material_content * (y / 100);

final_cement_content = cementitious_material_content - Alccofine_material  - Nylon_fibers;

if final_cement_content > 450

   fprintf ('Warning! The cement content is exceeding the required value (%.2f)\n', final_cement_content);

end

specific_gravity_of_cement = input ('Enter the value of specific gravity of cement (between 3 to 3.15): ');

specific_gravity_of_fine_aggregate = input ('Enter the value of specific gravity of sand or fine aggregates (between 2.5 to 3): ');

specific_gravity_of_coarse_aggregate = input ('Enter the value of specific gravity of gravels or coarse aggregates (between 2.5 to 3): ');

specific_gravity_of_alccofine_material = input ('Enter the value of specific gravity of alccofine material:');

specific_gravity_of_chemical_admixture = input ('Enter the value of specific gravity of chemical admixture:');

specific_gravity_of_nylon_fibers = input ('Enter the value of specific Nylon fibers admixture:');
```

```
mass_of_chemical_admixture = (cementitious_material_content) * (0.5 / 100);
volume_of_chemical_admixture = ((mass_of_chemical_admixture) / (specific_gravity_of_chemical_admixture)) * (1 / 1000);
mass_of_chemical_admixture_kg = (volume_of_chemical_admixture) * (specific_gravity_of_chemical_admixture) * 1000;
mass_of_crystalline_admixture = (cementitious_material_content) * (1.1 / 100);
mass_of_alccofine =( cementitious_material_content) * (x/100);
mass_of_Nylon =( cementitious_material_content) * (y/100);
Volume_of_cement = (cement_content_required_in_kg_per_m3 / specific_gravity_of_cement) / 1000;
Volume_of_water = final_water_content / 1000;
Volume_of_Alccofine = ((mass_of_alccofine)/(specific_gravity_of_alccofine_material))*1/1000;
Volume_of_Nylon_fibers = ((mass_of_Nylon)/(specific_gravity_of_nylon_fibers))*1/1000;
Volume_of_aggregates = ((1-0.01) - (Volume_of_cement + Volume_of_water + volume_of_chemical_admixture + Volume_of_Alccofine + Volume_of_Nylon_fibers));
Revised_water_cement_ratio = final_water_content / cementitious_material_content;
volume_of_fine_aggregate = 1 - volume_of_coarse_aggregate;
coarse_aggregates_required_in_kg = Volume_of_aggregates * volume_of_coarse_aggregate * specific_gravity_of_coarse_aggregate * 1000;
fine_aggregates_required_in_kg = Volume_of_aggregates * volume_of_fine_aggregate * specific_gravity_of_fine_aggregate * 1000;
fprintf('Mix Design Results:\n');
fprintf('Final Cement Content: %.2f kg/m3\n', final_cement_content);
fprintf('Alccofine_material: %.2f kg/m3\n', Alccofine_material);
fprintf('Nylon_fibers : %.2f kg/m3\n', Nylon_fibers);
fprintf('Final Water Content: %.2f kg/m3\n', final_water_content);
fprintf('Fine Aggregates Required: %.2f kg/m3\n', fine_aggregates_required_in_kg);
fprintf('Coarse Aggregates Required: %.2f kg/m3\n', coarse_aggregates_required_in_kg);
fprintf('mass_of_chemical_admixture_kg: %.2f kg/m3\n', mass_of_chemical_admixture_kg);
fprintf('mass_of_crystalline_admixture : %.2f kg/m3\n', mass_of_crystalline_admixture);
fprintf('Revised Water-Cement Ratio: %.3f\n', Revised_water_cement_ratio);
```

3. Demonstration Using Sample Data:

INPUT:

Enter the grade of concrete to be used in mix 25/30/35/40/45/50:

25

Enter the nominal size of the aggregate to be used in mix25/30/35/40/45/50 (in mm):

20

Enter the zone of fine aggregates (1/2/3/4):

2

Enter the value of slump between 50 to 120 (in mm):

75

Enter the exposure condition (mild/moderate/severe/very severe/extreme):

severe

Enter the water content reduction percent:

20

Enter the percentage of Alccofine material (x):

10

enter the percentage of nylon fibers(y):

0.5

enter the percentage of crystalline admixture(CA):

1.1

Enter the value of specific gravity of cement (between 3 to 3.15):

3.15

Enter the value of specific gravity of sand or fine aggregates (between 2.5 to 3):

2.90

Enter the value of specific gravity of gravels or coarse aggregates (between 2.5 to 3):

2.80

Enter the value of specific gravity of alccofine material:

2.86

Enter the value of specific gravity of chemical admixture:

1.08

Enter the value of specific Nylon fibers admixture:

1.14

OUT PUT:

Mix Design Results:
Final Cement Content: 301.78 kg/m3
Alccofine_material: 33.72 kg/m3
Nylon_fibers : 1.69 kg/m3
Final Water Content: 153.26 kg/m3
Fine Aggregates Required: 779.11 kg/m3
Coarse Aggregates Required: 1280.85 kg/m3
mass_of_chemical_admixture_kg: 1.69 kg/m3
mass_of_crystalline_admixture : 3.71 kg/m3
Revised Water-Cement Ratio: 0.455

Quantities	Manual calculations		MATLAB calculations		Difference(D)= V1-V2
	Volume(V1) Kg/m^3	Ratio	Volume(V2) Kg/m^3	Ratio	
Cement	301.615Kg/m^3	1	301.78	1	-0.165
Alccofine	33.7 Kg/m^3	0.11	33.72	0.11	-0.02
Nylon fibers	1.685	0.005	1.69	0.005	-0.005
Crystalline admixture	3.707	0.01	3.71	0.01	-0.003
Fine aggregates	779.107	2.58	779.11	2.58	-0.003
Coarse aggregates	1280.66	4.24	1280.85	4.24	-0.19
Water	153.264	0.45	153.26	0.45	-0.004
Super plasticizer	1.685	0.005	1.69	0.005	-0.005

Table 1: Comparative MATLAB and manual calculations

4. Conclusion:

This paper describes a step-by-step methodology using MATLAB to design concrete mixes for various grades in accordance with IS codes. It also demonstrates the conversion of self-healing concrete and presents simulated results. These results are then compared to those obtained through manual calculations. During this comparison, it was found that MATLAB provides higher accuracy because it includes very small values up to the fourth decimal place or beyond, which are typically neglected in manual computations. This reduces quantization errors—errors caused by rounding off values—and slightly decreases the required amounts of coarse and fine aggregates compared to manual calculations. This highlights the superior accuracy achieved using MATLAB. Additionally, the time needed to design a concrete mix in MATLAB is significantly reduced, as the user only needs to click a few buttons to determine the mix ratios, whereas manual calculations involve numerous steps. Furthermore, unlike Excel sheets, MATLAB provides data abstraction, where the user interacts only with the front end while the underlying formulas remain hidden. This eliminates human errors associated with incorrect formula application, calculator misuse, or typographical errors.

REFERENCES:

[1]. Aitcin, P. C. 2000, "Review: Cements of Yesterday and Today, Concrete of tomorrow", Cement and Concrete Research(30), Elsevier, pp. 1349 – 1350.

[2]. Singh, B. "Concrete Technology", Kaption Publishing House in association with Katson Publishing House, second edition,pp. 1 – 4.

[3]. How Concrete is Made. [Online]. Available: http://www.cement.org/cement-concrete-basics/how-concrete-is-made.[Accessed: 26-Jun-2019].

[4]. IS 10262, 2009: Guidelines for concrete mix design proportioning [CED 2: Cement and Concrete], First Revision, Bureauof Indian Standards, pp. 1 – 2

[5]. Zealakshmi, D., Ravichandran, A., and Kothandaraman, S. 2013, "Computer Aided High Strength Concrete MixtureProportion using Mat Lab," International Journal of Computer Applications, vol. 70 (8), pp. 18–25.

[6]. Gupta, A., Mittal, N. and Saini, B., 2011, Computer-aided Proportioning of Concrete Mixes, Delhi: National Institute ofTechnology India.

[7]. Makenya A. R., John Paul M., 2017. Potentiality of Using a Developed MatLab Program for the Design of High StrengthConcrete Mixes, Journal of Civil Engineering Research, 7(3), pp. 81-98.

[8]. Okoloekwe, R.C., Okafor, F.O., 2007. A New Approach to Concrete Mix Design using Computer Techniques, NigerianJournal Of Technology, Vol. 26(1).