XLVI CILAMCE
VITÓRIA - ES - BRAZIL - 2025
XLVI IBERO-LATIN AMERICAN CONGRESS ON COMPUTATIONAL METHODS IN ENGINEERING

Educational tool for determining longitudinal reinforcement layout in reinforced concrete beams

David F. Orquera[1]

[1]*Postgraduate Program in Civil Engineering, Federal University of Ouro Preto*
Professor Paulo Magalhães Gomes Street, 122 - Bauxita, 35400-000, Ouro Preto-MG, Brasil
david.yar@aluno.ufop.edu.br

Abstract. The selection of longitudinal reinforcement in reinforced concrete beams is traditionally carried out manually, based on trial-and-error procedures. This approach considers commercially available rebars configurations and must comply with design code requirements, such as minimum steel area and bar spacing. Although common in practice, this process is repetitive and prone to errors when evaluating multiple alternatives under constructability and regulatory constraints. The complexity increases when design decisions must simultaneously satisfy normative, practical, and economic criteria simultaneously must be satisfied. This study presents a practical educational tool for engineering applications, developed in Python, which assists in defining reinforcement distribution during the structural design process. The program accepts user input for the required steel area, beam geometry, maximum allowable bar diameter, and a tolerance margin. Based on this data, it iterates through a matrix of steel areas associated with various commercial bar types and quantities. The algorithm identifies feasible combinations that comply with ACI 318 provisions and prioritizes those with the least excess steel. The tool supports both students and engineers, enabling faster and code-compliant decision-making in reinforced concrete element design.

Keywords: Educational Tool, Longitudinal Reinforcement, Practical Engineering Tool, Reinforced Concrete, Engineering Programming

1 Introduction

The structural design of reinforced concrete beams requires accurately determining the longitudinal reinforcement to resist design loads. An essential step is selecting a bar configuration that meets the required steel area (*As*) and complies with design codes such as ACI 318 [1].

Traditionally, the selection of longitudinal reinforcement relies on a manual, trial-and-error method. This iterative process is inherently time-consuming and prone to errors, often leading to sub-optimal designs. This specific challenge has been widely recognized and has motivated extensive research into computational methods for optimizing reinforced concrete structures [3].

To overcome these limitations, this study presents a Python-based computational tool that automates the selection of longitudinal reinforcement. The algorithm iterates through a matrix of predefined steel areas, considering user-defined constraints such as target steel area, element geometry, and diameter limits. Viable combinations are filtered to keep only those that meet ACI 318 [1] spacing rules, prioritizing the most steel-efficient options. The objective is to provide a practical and educational tool that streamlines decision-making in reinforced concrete design, ensures code compliance, and improves calculation accuracy [2].

2 Methodology

The functionality of the tool presented in this study operates based on a set of input data provided by the user. These parameters include the geometric dimensions of the cross-section (base), the concrete clear cover,

the diameter of the transverse reinforcement (stirrups), the maximum diameter of the longitudinal reinforcement bars to be considered (expressed in mm), and the required steel area (As) to be fulfilled using a configuration of commercially available bars.

2.1 Imput Data

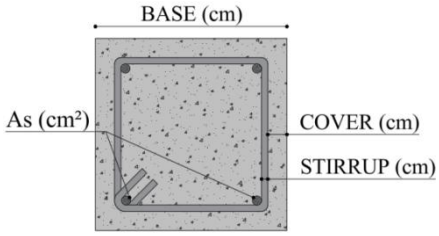


Figure 1. Cross-section of a reinforced concrete beam showing the geometric variables considered in the algorithm.

Figure 1 provides a graphical representation of the program's input data. The required steel area (As), provided by the user in *cm²*, represents the total amount of longitudinal reinforcement the algorithm seeks to distribute among the commercially available bars. The remaining geometric variables are used by the algorithm to verify the spacing between bars, thus ensuring compliance with the requirements established by the ACI code.

2.2 Steel Combinations Matrix

(mm)	1	2	3	4	5	6	7	8	9	10	11	12
6	0.28	0.57	0.85	1.13	1.41	1.70	1.98	2.26	2.54	2.83	3.11	3.39
8	0.50	1.01	1.51	2.01	2.51	3.02	3.52	4.02	4.52	5.03	5.53	6.03
10	0.79	1.57	2.36	3.14	3.93	4.71	5.50	6.28	7.07	7.85	8.64	9.42
12	1.13	2.26	3.39	4.52	5.65	6.79	7.92	9.05	10.18	11.31	12.44	13.57
14	1.54	3.08	4.62	6.16	7.70	9.24	10.78	12.32	13.85	15.39	16.93	18.47
16	2.01	4.02	6.03	8.04	10.05	12.06	14.07	16.08	18.10	20.11	22.12	24.13
18	2.54	5.09	7.63	10.18	12.72	15.27	17.81	20.36	22.90	25.45	27.99	30.54
20	3.14	6.28	9.42	12.57	15.71	18.85	21.99	25.13	28.27	31.42	34.56	37.70
22	3.80	7.60	11.40	15.21	19.01	22.81	26.61	30.41	34.21	38.01	41.81	45.62
25	4.91	9.82	14.73	19.63	24.54	29.45	34.36	39.27	44.18	49.09	54.00	58.90
28	6.16	12.32	18.47	24.63	30.79	36.95	43.10	49.26	55.42	61.58	67.73	73.89
32	8.04	16.08	24.13	32.17	40.21	48.25	56.30	64.34	72.38	80.42	88.47	96.51
38	11.34	22.68	34.02	45.36	56.71	68.05	79.39	90.73	102.10	113.40	124.80	136.10

Figure 2. Steel areas by bar diameter and number of bars.

The Figure 2 shows the reinforcement bar diameters in millimeters in the first column, the number of bars from 1 to 12 in the top row, and the total steel area in square centimeters resulting from each specific diameter and quantity combination.

2.3 Search and Verification Algorithm

The algorithm's logic proceeds in two main phases to find a suitable bar combination among 3,575 possible options.

Phase 1: Single-Diameter Search. The algorithm first iterates through 143 combinations using a single bar diameter (from 6 mm to 38 mm are considered valid options). It filters for solutions that meet the required steel area (As) and comply with the ACI 318 minimum spacing requirements.

Phase 2: Mixed-Diameter Search. If no solution is found in Phase 1, the program advances to a more detailed search of up to 3,432 combinations. It evaluates configurations combining bars of two different, consecutive diameters to find an option that satisfies both the steel area and spacing requirements..

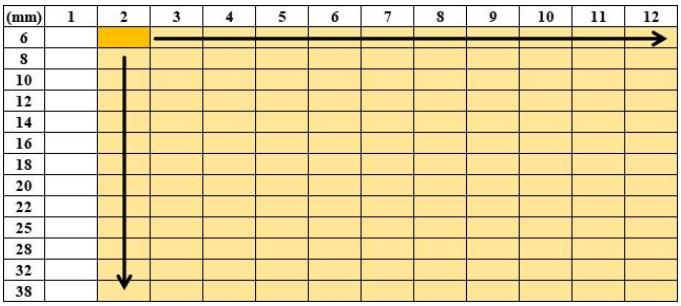


Figure 3. First iterative review of bar combinations.

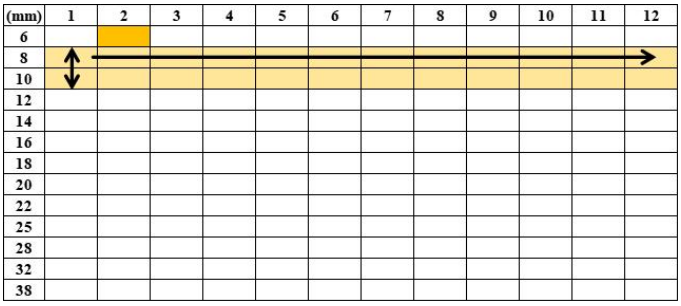


Figure 4. Second iterative review of bar combinations.

To illustrate this functionality, a design case for a reinforced concrete beam has been proposed with the following constraints: :

Beam width: *40 cm;* Concrete cover: *3 cm;*
Stirrup diameter: *1 cm;* Maximum rebar diameter to be considered: *25 mm;*
Search tolerance: *0.5* cm^2*;* Required steel area *(As): 25.13* cm^2

However, when applying the spacing filter, it is determined that the spacing between bars *2.28 cm* is less than the minimum requirement of *2.5 cm.* The clear spacing between reinforcement bars, "s", is calculated using the following formula, which considers the geometric constraints of the section:

$$s = \frac{b - 2r_c - 2d_c - nd_b}{n - 1} \tag{1}$$

s: Clear spacing between bars; b: Width of the beam's base.; r_c: Clear cover.;
d_c: Stirrup diameter.; n: Number of reinforcement bars.; d_b: Reinforcement bar diameter.

For the target As of *25.13* cm2, the algorithm's Phase 1 search identifies that 8 bars of 20 mm match the required area. However, this option is discarded because the calculated spacing of 2.28 cm is less than the 2.5 cm minimum requirement. The algorithm then automatically proceeds to Phase 2, where it finds a valid mixed combination that satisfies all constraints: you need 2 bars of 20 mm diameter and 5 bars of 22 mm diameter. This configuration provides a total steel area of 25.16 cm2 and a clear spacing of 2.83 cm.

3 Implementation

The source code of the tool is publicly available in a GitHub repository. Below is a detailed description of the algorithm's logic for searching the optimal combination of steel bars based on the user's input demand. The explanation covers the process from initial data entry to the fulfillment of code requirements and the presentation of possible solutions.

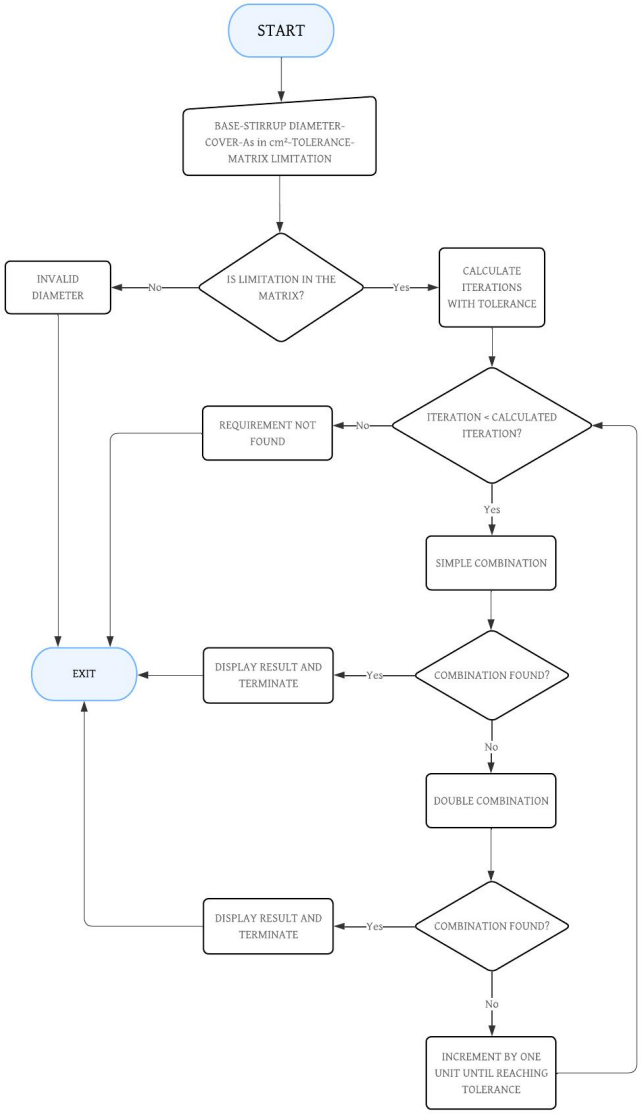


Figure 5. Flowchart of the algorithm

4 Results and Discussion

To demonstrate the tool's practical application and validate its effectiveness, a typical design scenario for a reinforced concrete beam was analyzed. This section presents the results of this case study and discusses their implications regarding design efficiency and accuracy.

4.1 Case Study: Rectangular Beam Design

The algorithm was tasked with finding a suitable longitudinal reinforcement layout for a beam with the following design parameters:

Beam width: *35 cm;*
Concrete cover: *2.5 cm;*
Stirrup diameter: *1 cm;*
Maximum rebar diameter to be considered: *22 mm;*
Search tolerance: *0.5* cm^2*;*
Required steel area *(As): 16.54* cm^2

The objective was to find a bar combination that satisfied the required steel area while adhering to the minimum spacing requirements stipulated by ACI 318.

4.2 Comparison with Manual Design Approach

Faced with this problem, a design engineer would typically begin a manual, trial-and-error process. A plausible first attempt would be to use a single bar diameter to simplify the layout. To meet the *16.54* cm^2 requirement, a designer might select *8* bars of *16 mm* (providing As= *16.08* cm^2). However, a manual spacing check using Equation (1) would reveal a clear spacing of only *2.17 cm*, which is below the *2.5 cm* minimum required by ACI 318.

This initial configuration would be invalid. The designer would then be forced to try other options, such as using fewer but larger bars (e.g., *6* bars of *20 mm*, which provides excessive steel) or begin the more complex task of testing mixed-diameter combinations. This iterative process consumes valuable time, is susceptible to calculation errors, and offers no guarantee that the first valid solution found is the most optimal one.

4.3 Tool-Generated Solution and Discussion

In contrast to the manual approach, the developed tool instantaneously processes the input data to provide an optimized, code-compliant solution: *2* bars of *16 mm* combined with *4* bars of *20 mm*. This configuration yields a total steel area of *16.59* cm^2 and a valid clear spacing of *3.36 cm*.

This result serves a dual purpose. First, it validates the algorithm's reliability by providing a solution that is verifiably correct according to ACI 318 provisions. Second, it highlights its advantages in efficiency and optimization. While other computational approaches focus on advanced layout generation, such as aligning reinforcement with principal stress directions [4], the primary contribution of this tool lies in its practicality and educational focus. It automates the tedious manual iterations and systematically explores thousands of combinations to identify an optimal, code-compliant layout that a designer might otherwise overlook.

5 Conclusions

This study demonstrates the development of a practical and educational tool, written in Python, which addresses the inefficiencies and error-prone nature inherent in the manual process of selecting longitudinal reinforcement in reinforced concrete beams. The proposed algorithm transforms the traditional trial-and-error method into a systematic and automated procedure.

An improvement for the algorithm would be the inclusion of a module that allows the selection of additional longitudinal reinforcement. Currently, the program focuses on a uniform reinforcement configuration along the entire length of the element. However, in structural design practice, it is common to require extra bars in high-stress areas, such as beam ends, to resist negative moments.

Acknowledgements. The research has been sponsored by the Brazilian Research Council (CNPq), the Coordination for Improvement of Higher-Education Personnel (CAPES), and the Minas Gerais Research Support Foundation (FAPEMIG).

References

[1] American Concrete Institute (ACI). (2019). Building Code Requirements for Structural Concrete (ACI 318-19) and Commentary. Farmington Hills, MI: American Concrete Institute.
[2] J. A. Silva, T. H. Medeiros, and F. Goes, “Large Language Models on Solving Integral Calculus Problems,” in \textit{Proc. 54th ACM Tech. Symp. Comput. Sci. Educ. V. 1}, Toronto, ON, Canada, 2023, pp. 563-569.
[3] M. Afzal, Y. Liu, S. Nazar, and M. U. Farooq, “Reinforced concrete structural design optimization: A critical review,” Journal of Building Engineering, vol. 50, p. 104149, Jun. 2022.
[4] P. Gali, L. E. Miguel, M. Popescu, and B. Dillenburger, “Computing Rebar Layouts Aligned with the Principal Stress Directions,” in ACADIA 2023: Habits of the Anthropocene, Denver, CO, USA, 2023, pp. 314-323.