POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Civile

Tesi di Laurea Magistrale

# A genetic algorithm-based framework for the optimal seismic retrofitting of reinforced concrete buildings by steel-jacketing

**Relatori**
prof. Fabio Di Trapani
prof. Giuseppe Carlo Marano

**Studente**
Antonio Pio Sberna

Anno accademico 2019-2020

*To the memory
of my mother*

# Acknowledgements

At the end of this university course, I would like to express my sincere gratitude to the following people:

My supervisors prof. Fabio Di Trapani for always being close to me despite the critical world situation, wisely accompanying me on this first research experience and prof. Giuseppe Marano who introduced me into the world of structural optimization and assisted in the realization of this thesis. I really appreciate the great trust you have placed in me.

I will never stop feeling sincere gratitude for all the teachers and professors who have led me up to here, all of them sharing a piece of their knowledge have motivated me to do always better.

I would like to mention and especially thank all the colleagues with whom I shared a stretch of this trail, thank you for the fantastic times we had together.

Last, but most importantly, my family for their immeasurable and unconditionally support.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

A large amount of buildings and infrastructures in the world are reinforced concrete (RC) frame structures designed prior to the entry into force of seismic guidelines and seismic detailing rules. Seismic risk associated with these structures is significant due to their low lateral load-carrying capacity and insufficient ductility. In particular, RC columns play a critical role to the seismic performance, being the location of most of the structural deficiencies (poor concrete,inadequate transverse reinforcement, lack of seismic details).

One of the extensively used retrofitting technique for columns is the steel-jacketing. It consists of the installation of a cage made of steel angles and battens providing additional confinement and transverse reinforcement to the RC elements, and compensating their ductility lack.

The main issues that structural engineers face in the design of this kind of interventions regard the determination of the position and the amount of the retrofitting to exploit the maximum effect, reducing costs and invasiveness of the intervention.

Currently, the design of these retrofitting interventions is mainly based on engineer's intuition and experience and, hence, this could lead to an over-estimated design, associated with an increase of economical and downtime costs.

This work of master's degree thesis addressed the use of genetic algorithms (GA), proposing a rational method to optimise the seismic retrofitting of the existing RC structures with steel-jacketing.

The optimisation is performed both for the position of the retrofitting system (topological optimisation) and for the amount of steel used for the jacketing, by varying battens interaxis. The research space consists of all the combination of retrofitted columns with all the different battens spacings.

The metaheuristic procedure allows obtaining the optimal solution without the need of evaluating all the possible solutions that could involve huge computational effort. The main GA operators (selection, crossover, and mutation) concur to explore the research space roughly and evolve the suitable results toward better solutions.

The GA analysis aims to select the cheapest retrofitting solution among the feasible ones. The cost of each candidate solution is evaluated by the objective function, which takes into account material and workmanship related costs. The feasibility of each solution is verified by the results of static pushover analyses in the framework of the N2 method from the results carried out a 3D fibre-section model, developed in the *OpenSees* software platform.

Chapter 2 contains a brief rewiew of the state of art of structural optimization and critical literary review of articles on the use of optimization allgorithm for the structural retrofitting.

In Chapter 3 is presented the structure of the algorithm, in particular the arrangement of each its components, the fundamental hypotheses underlying the method, the application domain and the limitations of use of the proposed framework.

Chapter 4 involves the model used for the following study case, peculiarly the features of the chosen retrofitting system, the theories that were used to model the confining effect of steel-jacketing and to verify the brittle mechanism of shear collapse.

The last Chapter 5 is based on results analysis of the proposed approach applied for different case study structures subject to different structural deficiencies (plan and height irregularities, local shear failures, influence of masonry infills), highly representative of the class of RC existing structures built in the middle of 1900.

# Chapter 2

# Artificial intelligence and structural optimization

Since the early works of Patrick Blackett and the Tizard committee which laid the foundations for the birth of the new branch of mathematics called *operational research*, optimization algorithms have always had a strongly engineering connotation.

From the first works carried out during the Second World War on the search for the best disposition of the British anti-aircraft systems, or of the intercepting systems of submarines in the English Channel, the importance of this type of analysis emerged. which in some cases provided for counter-intuitive but effective solutions [6].

At the end of the conflict, the methods investigated by the research team were published and applied to various civil problems, for example the optimization of production cycles, problems related to industrial planning or the optimization of transport networks (railways and roads).

The main factors that played a key role in the rapid growth of operations research was the computer revolution. A large amount of computation is usually required to deal most effectively with the complex problems considered by optimization. Above all, the development of electronic digital computers, with their capacity to perform arithmetic calculations milions of times faster than a human being can, was the principal incentive to the growth of this discipline.

One of the first problems that emerged was that real-world problems are difficult to analyse and solve for several reason, especially because the number of possible solution in the search space is so large to forbid an exhaustive search, the evaluation function is noisy, the possible solutions are heavily constrained that constructing even some feasible solution is difficult (Michaelewicz et al. (2013) [7]). For all of these and other reasons there is no single method available for solving all optimization problems efficiently. Hence in the past sixty years several optimization methods have been developed for solving different types of optimization problems.

In the simplest case, optimization seeks the maximum or minimum value of an objective function corresponding to variables defined in a feasible range or space. More generally, optimization is the search of the set of variables that produces the best values of one or more objective functions while complying with multiple constraints.

A single-objective constrained optimization problem can be stated as follows:

$$\text{Find } X = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix} \text{ which minimizes } f\left(X\right) \tag{2.1}$$

subject to the contraints:

$$\begin{cases} g_j\left(X\right) \leq 0, & j = 1{,}2, \ldots, m \\ l_j\left(X\right) = 0, & j = 1{,}2, \ldots, p \end{cases} \tag{2.2}$$

where $X$ is a $n$-dimesional vector that is termed *design vector*it is a set of *decision variables*that consitutes a possible solution to the optimization problem, $f\left(X\right)$ is called *objective function*, $g_j\left(X\right)$ and $l_j\left(X\right)$ are known as *inequality* and *equality constraints*, respectively.

Generally the equality constraints are often neglected, for simplicity, in the statement of a constrained optimization problem, although several methods are available for handling problems with equality constraints. The number of decision variables that detemines the dimension of the optimization problem ($n$) and the number of costraints ($m$, $p$) need not be related in any way. In engineering problems, the constraints are generally related to the feasibility of the solution.

The main way to summarizing the usual phases of an optimization study is the following:

1. Define the problem of interest
2. Formulate a mathematica model to represent the problem
3. Develop a computer-base procedure for deriving solutions to the problem from the model
4. Test the model and refine it as needed

The decision variables are inputs to the simulation model. Then, the state variables, which are outputs of the simulation model, are evaluated. Thereafter, the objective function is evaluated. In the next step, the problem constraints are determined, and lastly the fitness value of the current decision variables is calculated. At this time, the optimization algorithm generates a new possible solution of decision variables to continue the iterations if a termination criterion is not reached.

In Figure 2.1 is illustrated the relation between the simulation model and the optimization algorithm [8].

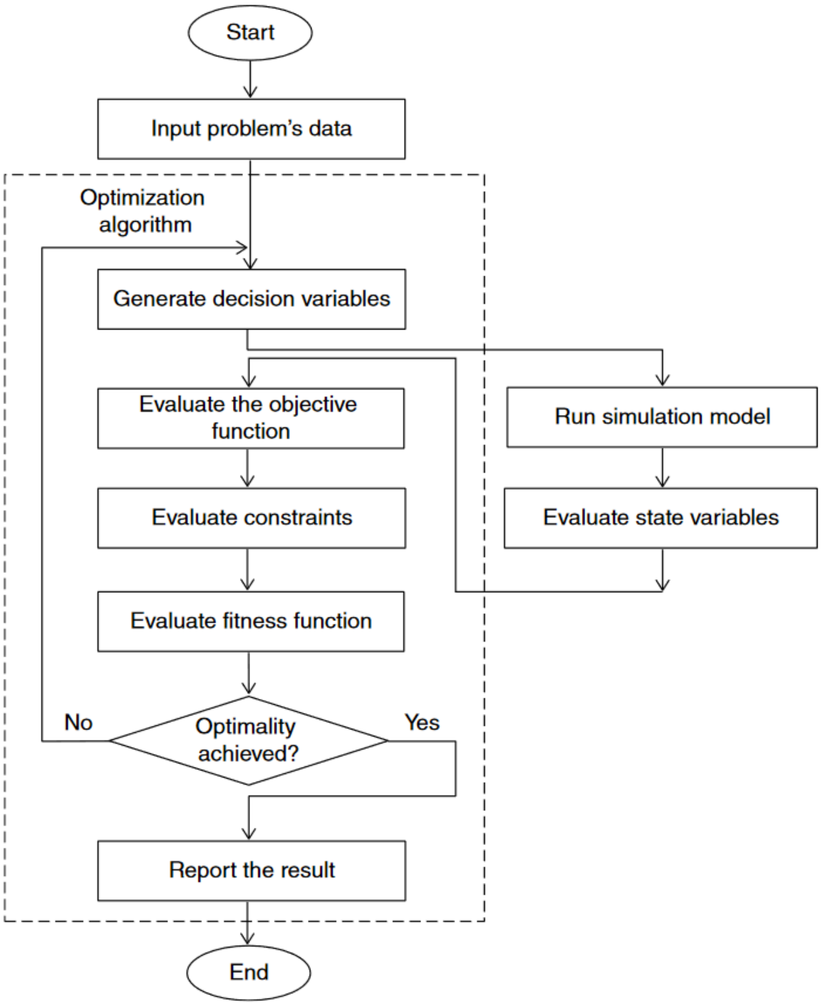



Figure 2.1: Relation between a simulation model and a optimization algorithm

The optimization techniques can be divided according to the number of functions to minimize at the same time (single-objective or multi-objective), the nature of the equations involved (linear or non-linear programming), the permissible values of the design variables (integer or continuum) and on the nature of the variables (deterministic or stochastic).

The main classification of the optimization techniques is related to the nature of the algorithm, they can be divided into *classical optimization methods* (hard-computing) and *modern optimization methods* (soft-computing).

The first ones are related to the differential calculus method laid by Newton and Leibnitz but used for minimization of functionals or variations calculus by Euler, Bernoulli, Lagrange and Weirstrass during the 1700s.

Cauchy made the first application of the steepest descent method to solve unconstrained minimization problems. Despite these early contributions, very little progress was made until the middle of the twentieth century, when digital calculators made implementation of the optimization procedures possible and stimulated further research on new methods. It is necessary to briefly mention the Bellman works on constrained optimization, the contributions of Zoutendijk and Rosen to nonlinear programming and the work of Gomory in integer programming.

The classical methods of optimization are useful in finding the optimum solution of continuous and differentiable functions.

The main peculiarity of these method is related to the to the need to formally define the function to be minimized. Furthermore, these algorithms present problems in research spaces noisy or characterized by steep variations.

The *modern optimization methods*, also sometimes called *non-traditional optimization methods*or *soft-computing algorithm*, have emerged as powerful and popular methods for solving complex engineering optimization problems in recent years.

Most of these method are named *population-based algorith* or *mimetic learning* because they draw inspiration from natural process such as the particle swarm optimization, the ant colony optimization, simulated annealing, and so on.

In Figure 2.2 is illustrated a non-exhaustive classification of soft-computing techniques.

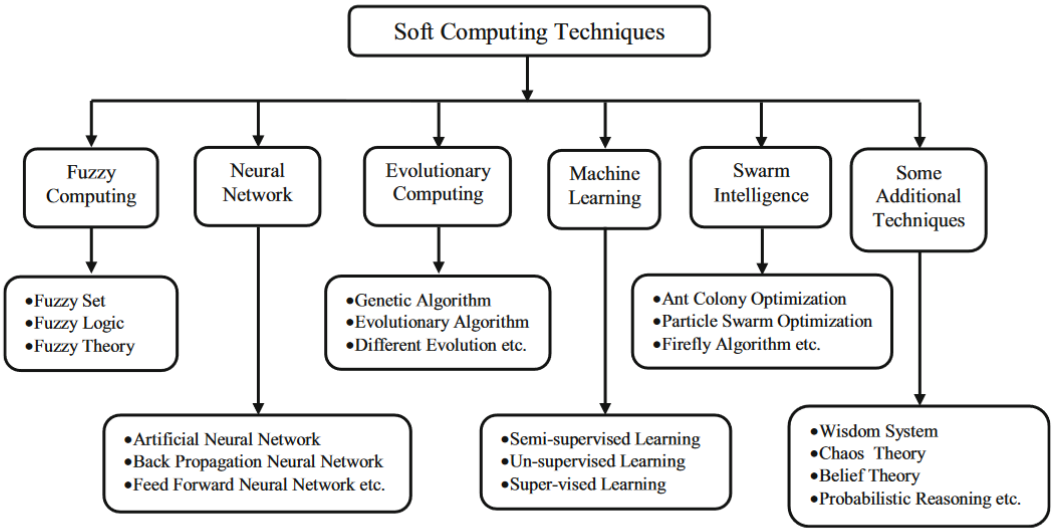



Figure 2.2: Soft-computing algorithm taxonomy (Sharma et al. 2019 [1])

These algorithms work by iteratively moving in the search-space toward better position without knowing its overall characteristics but only punctual values.

Soft-computing method are rahter tolerant of imprecision, uncertainty and partial truth in order to return approximated solutions in quick time. In figure 2.3 are reported the main peculiarity and advantages of both traditional methods (*hard-computing*) and soft-computing techniques.

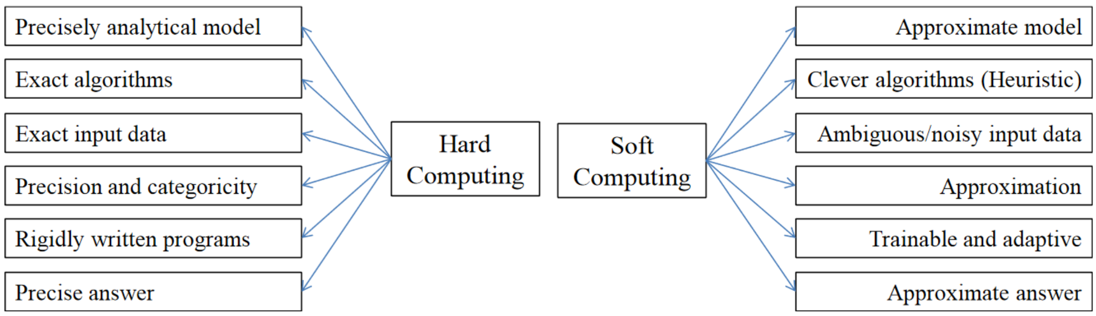



Figure 2.3: Main differences between hard-computing and soft-computing (Falcone et al. 2020 [2])

## 2.1 Genetic algorithm

The framework developed for this master's thesis work is based on Genetic Algorithm. This typology of optimization method belong to the class of Evolutionary Algorithm, the metaheuristic algorithm inspired to the Darwin's "evolution of specie" presented for the first time in the "On the Origin of Species by Means of Natural Selection" [9] and the Mendel's "inheritance laws" [10].

The word *metaheuristic* was coined by Glover in 1986 [11] and can be defined as a Sturzle described in his PhD dissertation [12]:

> Many of the metaheuristic approaches rely on probabilistic decisions made during the search. But, the main difference to pure random search is that in metaheuristic algorithms randomness is not used blindly but in an intelligent, biased form.

With this term call modern nature-inspired algorithms are usually called.

The earliest published record of evolutionary computation work conducted by Barricelli at the Institute for Advanced Study in Princeton on artificial life. His original research was published in Italian during 1954 [13] and three years later republished in English [14].

This algorithm was developed to simulate some of the biological mechanisms observed in natural evolution operating on genetic heritage.

This optimization algorithm was originally developed by Holland [15] and popularized by Goldberg [16].

The essence of an evolutionary approach is to regard candidate solutions of a generic problem as *individuals* belonging to a set called *population* and to introduce the notion of *fitness* as a formal measure of perceived performance of the individual with respect to the optimization objective.

Each individuals are characterized by *chromosome* made up of *genes* that represent decision variables (Figure 2.4).

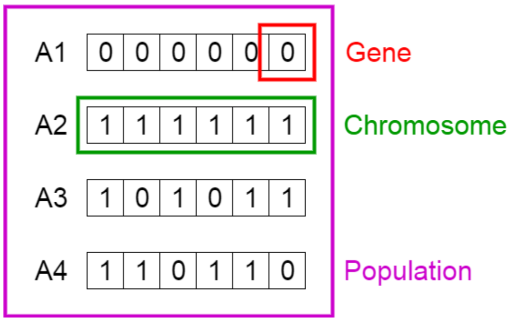



Figure 2.4: Definition of elements objects of genetic algorithm

The basic elements of natural genetics—reproduction, crossover, and mutation are used in the genetic search procedure. Genetic algorithms differ from the traditional methods of optimization in the following respects [17]:

- A population of points is used for starting the procedure instead of a single design point. Since several points are used as candidate solutions, GAs are less likely to get trapped at a local optimum.

- GAs use only the values of the objective function. The derivatives are not used in the search procedure.

- In GAs the design variables are represented as strings of binary variables that correspond to the chromosomes in natural genetics. Thus the search method is naturally applicable for solving discrete and integer programming problems. For continuous design variables, the string length can be varied to achieve any desired resolution.

- The objective function value corresponding to a design vector plays the role of fitness in natural genetics.

- In every new generation, a new set of strings is produced by using randomized parents selection and crossover from the old generation. Although randomized, GAs are not simple random search techniques. They efficiently explore the new combinations with the available knowledge to find a new generation with better fitness or objective function value.

As many of other soft-computing algorithm, GA does not require gradient information and, hence, it is particularly suitable to be used with objective functions that are not continuous and not differentiable.

The fitness of each string is evaluated by performing some type of system analysis to compute a value of the objective function.

The solution of an optimization problem by GAs starts with a population of random strings denoting several design vectors. Each design vector is evaluated to find its fitness value. The population is operated by three operatorsmuechanism inspired by biological evoluation, to produce a new population:

1. selection
2. crossover
3. mutation

The selection operator identify the best individuals of the generation

The crossover operation creates variations in the solution population by producing new solution strings that consist of parts taken from selected parent solution strings.

The mutation operation introduces random changes in the solution population because reproduction does not change the features of parent strings and there is the possibility that some important regions of the search space may never be explored getting stucked into a local minimum.

The new population is further evaluated to find the fitness values and tested for the convergence of the process. One cycle of reproduction, crossover, and mutation and the evaluation of the fitness values is known as a generation in GAs.

If the convergence criterion is not satisfied,the population is iteratively operated by the three operators and the resulting new population is evaluated for the fitness values. The procedure is continued through several generations until the convergence criterion is satisfied and the process is terminated.

GAs basically consist of a series of three processes:

- coding and decoding design variables into strings,
- evaluating the fitness of each solution string,
- applying genetic operators to generate the next generation of solution strings

In Figure 2.5 is illustrated the flowchart of a genetic algorithm.

Basic assumptions include population size, selection-strategy, crossover-type and the probability of mutation. By varying these parameters and strategies, the convergence of the algorithm may be altered.

Therefore, it is important to tune appropriate values for these parameters in order to balance the two main operation that GAs carry out: exploration and exploitation (Eiben et al. 1998 [18]).

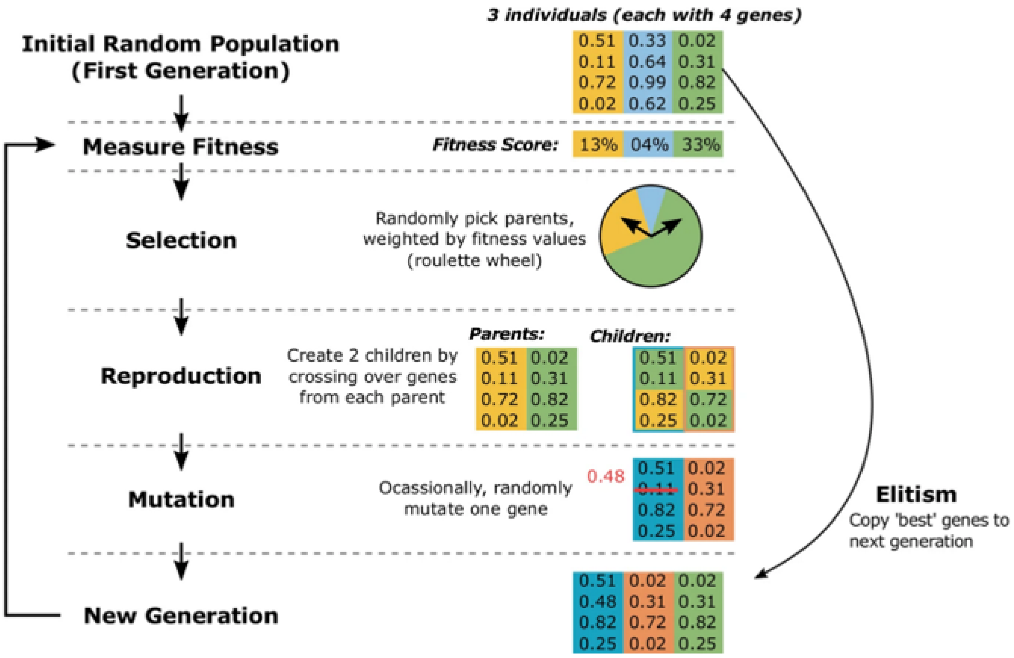



Figure 2.5: Flowchart of a genetic algorithm (Woodward et al. 2016 [3])

During the first phase the algorithm tries to explore as diffusely the entire search space in order to understand the borders of the minimum zones. A deep exploration of the search space is very important to avoid to loss multiple minimun zones.

The second stage is characterized by a deep analysis of the results in order to further refine the solutions.

In the two cases the mutation operator permits to avoid the stall into local minimum.

## 2.2 Optimization of retrofitting system - Literature review

Structural optimization problems, they are typically divided into three categories sometimes: topological optimization, sizing optimization and shape optimization [19].

Topological optimization aims at optimizing the structural layout within a given design space, for a given set of loads, boundary conditions and with the best possible performance of the system.

Shape optimization deals with optimizing the overall shape, or the contour of a structural system whose topology is fixed.

Sizing optimization is aimed at optimizing geometrical parameters, such as length, width or thickness of members in a structural system whose topology and shape are fixed.

During the past thirty years different optimization methods were used to solve structural problems. Typical structural problems solved thanks to the use of metaheuristic algorithms are the optimization of the bridges shape or spatial structures.

Applications of this type of algorithms for seismic adaptation of structures concern the optimization of the characteristics and position of viscous dumpers or tuned massed dumpers.

Only in recent years have avantgarde works on topological optimization of retrofitting system on existing frame structures arise in scientific literature.

The first work published by Seo et al. (2018) [20] presents the usage of ant colony optimization for the topological optimization of retrofitting on a school building. The structure, a three-storeys reinforced concrete frame, was analysed by tridimensional non-linear dynamic analysis.

The authors concluded that the ACO shows that reliable results could be derived, for the target structure the optimal solution saves over the half of the cost related to the intervention on all the columns.

Chronologically, the second work concerning the optimization of retrofitting systems is that published by Falcone et al. in 2019 [21].

In this publication the author presented a framework based on a genetic algorithm for the optimization of two different type of retrofitting system, confinement of columns (local intervention) and concentric steel bracing (global intervention). Optimization is both topological and sizing varying the dimensions of the elements of the bracing system.

The paper published by Mahdavi et al. (2019) [22] concern the usage of both genetic algorithms and particle swarm optimization for the optimization of FRP confinement retrofit for concrete structures. The objective was to confine the columns by different numbers of FRP wraps along their plastic hinges. The criterion to formulate the optimization problem was based on providing a uniform distribution of the plastic hinge rotation. In order to evaluate the capacity of the plastic hinge rotation, the effect of FRP confinement on the moment–curvature backbone curve of the column was quantified by a sectional analysis along with some empirical relations available in the literature.

In 2020 Di Trapani et al. [23] proposed a genetic algorithm approach for the minimization of steel-jacketing retrofitting system on concrete frame structure. The objective function calculates the amount of steel used for the intervention by varying the position and the spacing of the battens.

This paper is the starting point of this work of thesis.

# Chapter 3

# Optimization framework

The proposed optimization framework works by connecting the Matlab genetic algorithm (GA) tool with a FE structural model developed with the OpenSees software platform [24] . The framework is aimed at minimizing an objective function built by computing the retrofitting costs as a function of the defined design variables (number and location of retrofitted columns and respective battens spacing) associated with the steel jacketing reinforcement.

A flowchart of the optimization procedure is shown in the following Figure 3.1.

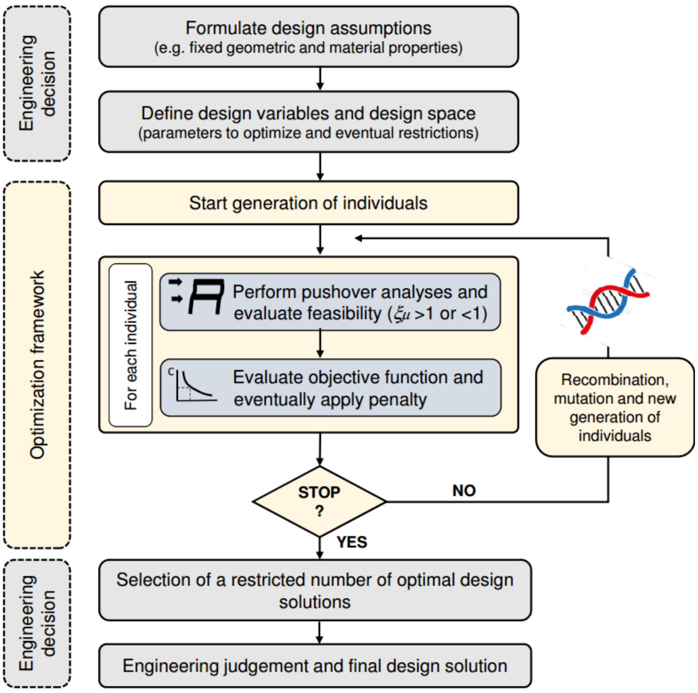



Figure 3.1: Flowchart of the optimization process

The procedure starts with the engineering design choices about fixed geometric and material properties. Then, the individuated design variables are eventually limited to a restricted design space (e.g. limit the number of columns involved in the optimization process, reduce the step of battens spacing variations).

The phase of restriction of the design space, fundamental to reduce the number of possible combinations of design variables and reduce computational effort, has to be carried out specifically for each case.

After this point, the optimization algorithm starts generating the first population of random individuals as described in Chapter 3.4. Each individual is representative of one model of the structure having one possible combination of the design variables.

The feasibility of each solution is assessed by carrying out one pushover analyses and computing the ratios between ductility capacity and demand ($\mu_c/\mu_d$) in the framework of the N2 method (Chapter 4.3). This allows reasonable computational effort and concise identification of seismic performance with a unique parameter.

The retrofitting cost of each solution is then computed by evaluating the objective function calculated as presented in chapter 3.2. The cost is eventually incremented by a penalty factor, fictitiously increasing the amount if one or more solutions are unfeasible ($\mu_c/\mu_d < 1$) (chapter 3.3).

For each generation the GA will combine the best individuals through the crossover and mutation operators as presented in the following paragraphs 3.6 and 3.7. The optimization framework is stopped when the optimization algorithm does not provide significant improvements in terms of cost minimization.

A final engineering judgment phase is necessary to assess potentially equivalent optimal solutions in terms of practice engineering feasibility and to eventually make final design corrections.

The framework consists of different routines (Figure 3.2), the main code where all the parameters of the analysis are defined (characteristics and size of the population, type of genetic operators, etc.) is the *MainCode* (Appendix 7.1. It performs all preliminary analyzes, performs the analysis and organizes the results obtained.

The design vectors, encoded as reported in the Chapter 3.1 are analyzed by the function *ModelCreator* (Appendix 7.4) which creates the structural model by writing the .tcl files containing the definition of the nodes, elements, loads and parameters for the execution of the structural analysis.

The model and analysis parameters created by the function *ModelCreator* are interpreted by *Geometry* (Appendix 7.12) and subsequently analyzed by *PushOver*.

The *Pushover* (Appendix 7.13) function performs the pushover analysis of the strucure returning several text files containing the numerical outputs (base shear, displacements and elements stresses). They are read by the *DuctilityCheck* (Appendix 7.3) routine which verifies the feasibility of the attempt solution by means of the capacity curve that is read, together with the stresses, by *CostFunction*.

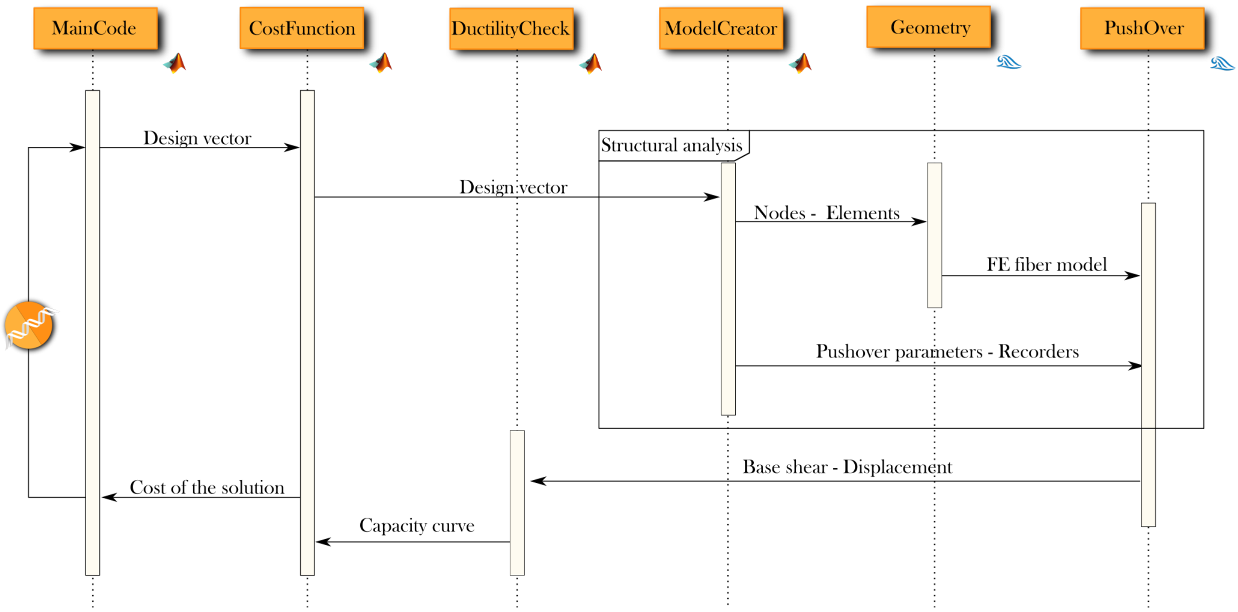



Figure 3.2: Sequence diagram of the framework

This last calculates the cost of the intervention by carrying out the shear verification of the vertical elements, according to the model presented in the following Chapter 4.4, and applying the penalty function if needed.

The cost of intervention are calculated as shown in the following Chapter 3.2. It is interpreted by the genetic algorithm as the value of the objective function associated with the analyzed vector design.

In order to restrict the design optimization variables as much as possible, the following basic assumption are made for the steel jacketing retrofitting system:

1. The angles are constituted by L-shaped steel profiles having fixed lateral length ($l_a$) and thickness ($t_a$) for all the retrofitted columns

2. The battens are constituted by rectangular plates having fixed thickness ($t_b$) and width ($w_b$) for all the retrofitted columns.

3. Battens spacing is the same for all the retrofitted columns.

The consequence of the aforementioned assumptions is that the battens spacing ($s_b$) remains the only variable defining the effect of the jacketing on confinement.

## 3.1 Design vector

The design vector characterize each solution encoding the gene of an individual in order to determine the value of the objective function. The main aim of the optimization algorithm is to search for the design vectors that yield the best value of the objective function.

In the case analysed in this thesis to represent each tentative solution the design vector is composed as:

$$\boldsymbol{b} = \begin{pmatrix} s_b \\ \boldsymbol{p} \end{pmatrix} \tag{3.1}$$

where $s_b$ is a scalar belonging to the interval S so defined:

$$s_b \in S = [s_{b,min}, s_{b,max}] \tag{3.2}$$

in which $s_{b,min}$ and $s_{b,max}$ are the minimum and maximum allowed battens spacings, while $\boldsymbol{p}$ is a vector collecting the positions of the columns included in the design space having the following form:

$$\boldsymbol{p} = (\ldots\ldots c_{ij} \ldots)^T \tag{3.3}$$

The elements belonging to $\boldsymbol{p}$ have the generic shape $c_{ij}$ elements, where $i$ represents the position of the column with reference numbering in plan, and $j$ represents the storey. The $c_{ij}$ elements are binary elements assuming the value 0 if the column is not retrofitted and 1 if the column is retrofitted. Therefore, $c_{ij}$ elements belong to the binary set named $C$ and so defined:

$$c_{ij} \in C = (0{,}1) \subset \mathbb{N} \tag{3.4}$$

In this way every individual (namely a model) can be completely characterized by a $\boldsymbol{b}$ vector defining the position and battens spacing of the retrofitted columns.

## 3.2 Objective function

The objective function monitors the retrofitting costs intended as the material cost and the manpower costs to realise columns steel jacketing ($C_{sj}$) and necessary works for demolition and recostruction of plasters and masonry ($C_M$).

The general form of the objective function can be expressed as:

$$C = C_M + C_{sj}$$

The cost $C_M$ has been estimated considering a fixed amount ($c_m$) equal to 2000 € per reinforced column, hence:

$$C_M = n_c \cdot c_m$$

where $n_c$ is the number of retrofitted columns.

As regard $C_{sj}$, this can be computed as:

$$C_{sj} = c_s \cdot \sum_{i=1}^{n_c} W_{s,i} \tag{3.5}$$

where $W_{s,i}$ is the total weight of steel used to arrange a steel jacketing cage and $c_s$ is the manpower and material cost per unit weight (estimated in 4.5 €/kg).

For the current case, since all the columns of the same storey have the same dimension, Equation 3.5 becomes simply:

$$C_{sj} = (n_{c,1} \cdot W_{s_1} + n_{c,2} \cdot W_{s,2}) \cdot c_s \tag{3.6}$$

where $c_{s,1}$ and $c_{s,2}$ are the number of columns retrofitted on the ground and first floor respectively, and $W_{s,1}$ / $W_{s,2}$ the fixeed weight of the steel cage calculated for the genric $n^{th}$ floor as:

$$W_{s,n} = (V_{A,n} + V_{B,n}) \cdot \gamma_s \tag{3.7}$$

in which $\gamma_s$ is the specific weight of steel ($78.5\,\mathrm{kN\,m^{-3}}$) and $V_{A,n}$ is the total weight of steel angles applied at the corners of the columns, that is:

$$v_{A,n} = 8 \cdot l_a \cdot t_a \cdot l_{c,n} \tag{3.8}$$

$l_a$ is the width of the angle, $t_a$ is the thickness of the angle and $l_{c,n}$ is the length of the columns of the $n^{th}$ storey.

Finally, $V_{B,n}$ is the total volume of the battens, which depends on their spacing ($s_b$) as follows:

$$V_{B,n} = 2 \cdot (V_{bx} + V_{by}) \cdot \left(\frac{l_{c,n}}{s_b}\right) \tag{3.9}$$

where $V_{bx}$ and $V_{by}$ are the volumes of singles batten alogn the two orthogonal directions, that is:

$$V_{bx} = t_b \cdot l_b \cdot (b - l_a)$$
$$V_{by} = t_b \cdot l_b \cdot (h - l_a)$$

For the case of square columns, where $V_{bx} = V_{by} = V_b$ the Equation 3.9 becomes:

$$V_B = 4 \cdot V_b \cdot \left(\frac{l_{c,n}}{s_b}\right) \tag{3.10}$$

## 3.3 Penalty function

The search strategy adopted by the GA considers the fitness of a solution and is unaffected by any violation of problem constraints. For the current case, the feasibility of a solution is represented by the capacity/demand ratio ($\xi_\mu$), which is determined as shown in the following Chapter 4.3.

There are several techniques to take into account feasibility of a solution, and therefore the possible violation of a constraint such as removal method, refinement method or penalty function. For the purpose of this thesis work the last one was introduced.

This can be expressed by changing the objective function ($C$) into the objective function $F$ as follows:

$$F = C + \Pi \tag{3.11}$$

where $\Pi$ is the penalty function having the following form:

$$\begin{cases} 0 & \text{if } \xi_\mu \geq 1 \\ C_{\text{max}} \cdot \left(\frac{1}{\xi_\mu}\right)^3 & \text{if } \xi_\mu < 1 \end{cases} \tag{3.12}$$

and in which $C_{\text{max}}$ is the maximum possible retrofitting cost related to the structure with all first and second floor columns retrofitted with the minimum battens spacing $s_b = 150\,\text{mm}$.

This means that if a solution is not feasible, the current cost is fictitiously increased by $C_{\text{max}}$ multiplied by the factor $(1/\xi_\mu^3)$ which takes into account the distance of the cutrrent solution from the feasibility ($\xi_\mu = 1$).

A graphical exemplification of the penalty function is illustrated in the following Figure 3.3 as a function of the term $\xi_\mu$.

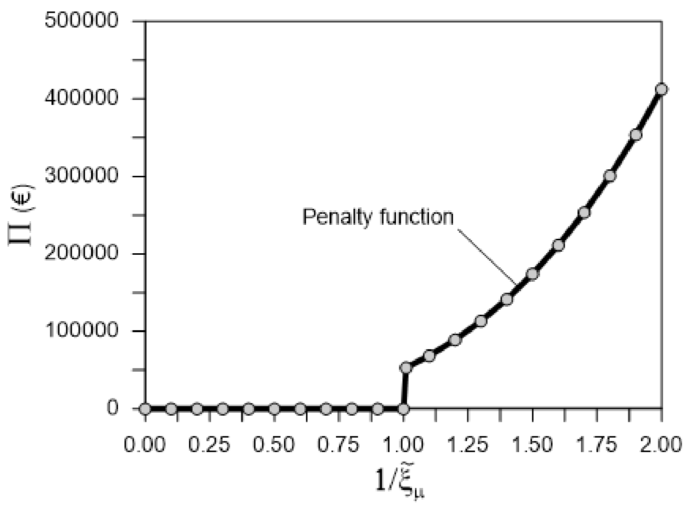



Figure 3.3: Penalty function

## 3.4 Initial population generation

The definition of the population size is essential to the effectiveness of the optimization, especially in terms of computational effort (each individual requires performing a pushover analysis), but this extremely varies case by case.

In particular, the generation of a random initial generation is fundamental to properly accomplish the exploration of the research space. For this purpose the *RandomDVGenerator* function was developed to generate a random design vector in function of its dimension and the percentage of retrofitted columns. The function developed for this purpose is reported in Appendix 7.8.

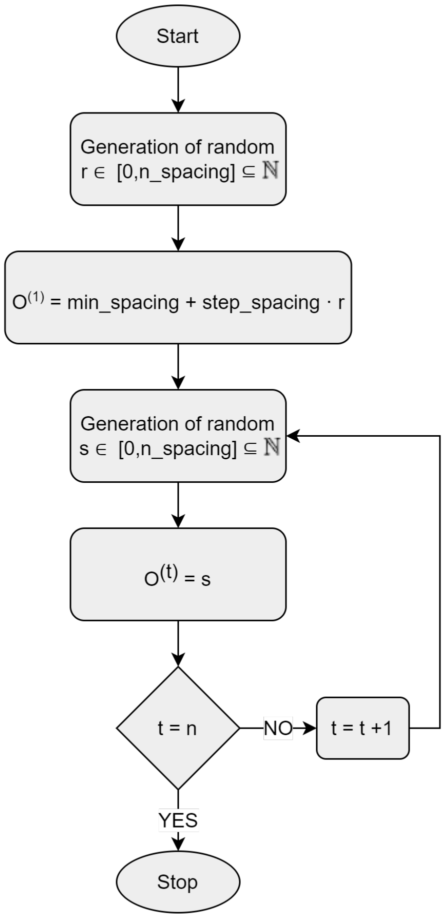



Figure 3.4: Initial population function flowchart

In Figure 3.4 is illustated the flowchart of the algorithm for the generation of a design vector with $n$ elements with minimum costraint for the battens spacing (*min_spacing*) and step of variation of the battens spacing (*step_spacing*).

## 3.5 Selection

The selection subroutine is one of the genetic operator necessary to improve the tentative best solution of each generation. It is the one that affects significantly the convergence of the algorithm. The basic strategy for the fitness-based procedures is based on the rule that the better fitted an individual, the larger the probability of its survival and mating.

For the framework developed during this thesis work, a "Fitness proportionate selection" procedure was chosen (Lipowski et al., 2012 [25]).

The selection probability ($p$) is associate to each individuals of the generation (composed of $n$ genomes) proportionally to their fitness ($w$) as:

$$p_i = \frac{w_i}{\sum_{i=1}^{n} w_i} \qquad \forall i \in [1; n] \subset \mathbb{N} \tag{3.13}$$

By creating a set of random number ($r$) with dimensions equal to the number of elements that have to be selected ($m$) such that:

$$0 < r < \sum_{i=1}^{n} w_i \tag{3.14}$$

the individuals who can pass on to generation are chosen.

This type of selection operator is commonly called *roulette-wheel selection* because it is equivalent to a random extraction from a roulette where each section dimention are propotional to the relative fitness of each individuals.

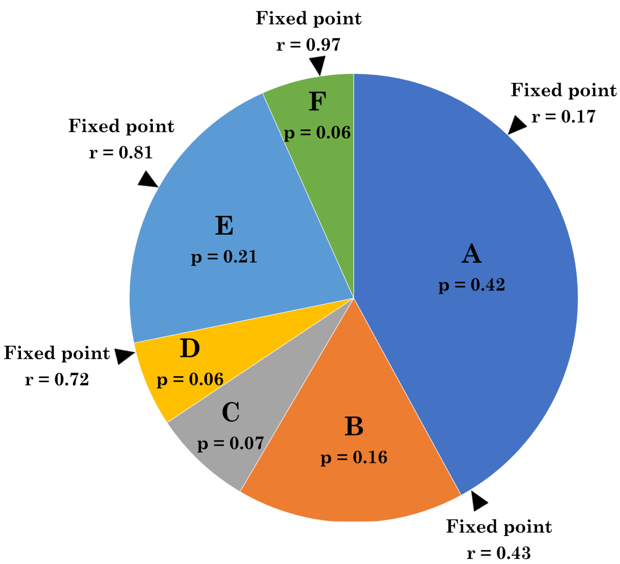



Figure 3.5: Example of roulette selection procedure

Individuals with a lower fitness are more likely to be eliminated during this type of selection process but still remains the possibility that some solutions to pass on the next generation.

This is an advantage because there is the possibility that some weaker solutions may have some genes that can be useful on the crossover process with better solutions.

The size of each slice corresponds to the fitness of the appropriate individual, the circumference of the wheel represents the sum of the fitness of all individuals of the generation.

The function used in the framework presented in this thesis is the *selection-stochunif* developed by Mathwork and reported in Appendix 7.9.

In the following Figure 3.5 is reported a graphical explaination of the procedure of this selection technique for the generation in Table 3.1 to select four chromosomes.

The random number for the extraction are 0.17, 0.43, 0.72, 0.81, 0.97.

| Chromosome | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| Fitness value | 8.2 | 3.2 | 1.4 | 1.2 | 4.2 | 1.3 |

Table 3.1: Generation example for the *roulette-wheel selection* procedure

## 3.6 Crossover

The crossover operator is used to improve the members of the population in the mating pool by mixing good sub-strings from two chromosomes with a view of getting a better individuals.

Among many crossover proposed in the scientific literature in the past years [26] [27], *uniform crossover* is implemented in the present study as given below (Syswerda 1989 [28]).

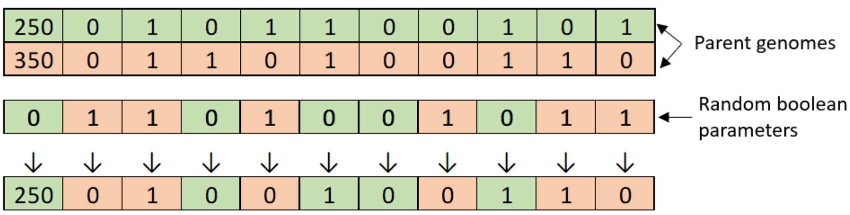



Figure 3.6: Example of uniform crossover

Selected two parent chromosomes randomly from the mating pool, they are mixed randomly from the generation of a random binary string of the same dimention of the parents.

In Figure 3.6 is reported an example of the principle of operation of this typology of crossover operator.

The function developed for this purpose is reported in Appendix 7.10.

In Figure 3.7 is reported the flowchart of the subroutine implemented for the parents individuals $P_1$ and $P_2$ of $n$ -dimension to create two offsprings $O_1$ and $O_2$ with a probability of swapping $p_s$.

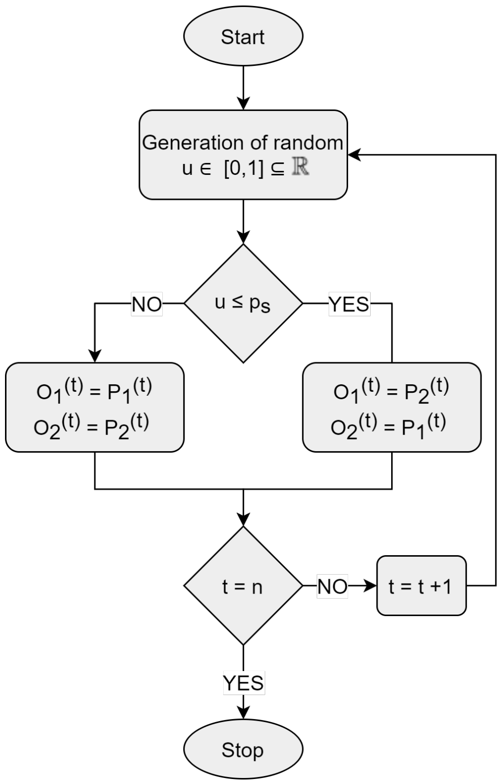



Figure 3.7: Flowchart of uniform crossover

## 3.7 Mutation

Mutation is the operator that is used to bring about random changes in the population. The need for mutation is to keep diversity in the population. This operation is carried out with a view to search unexplored areas and to avoid premature convergence at local optimum solution. At the same time, the higher frequency of applying this operator may also destroy the important information contained in the offspring. Hence, the probability of mutation is kept low (usually $p_m \in [0.001, 0.005]$).

Due to high computational effort requested by the analysis of the objective function, in particular to carry out the push-over analysis, the limited dimension of the population require an high mutation ratio was set $p_m = 0.05$ to explore all the possible optimum solution increasing the algorith's freedom to search outside the current region of variable space.

The relative high value of mutation ratio allows to avoid the stall of the analysis into local optima.

The heterogeneity of the design vector (Chapter 3.1) has required to define a new type of mutation function, in particular the position of retrofitting system follows the standard mutation. This operation is carried out by randomly selecting a binary bit ($u$) from the entire population and flipping the values from 0 to 1 or vice-versa. The battens spacing value needs another random number extraction ($v$) to decide if the mutation will increase or decrease the battens spacing of a battens space. The function developed for this purpose is reported in Appendix 7.11.

In the following Figure 3.8 is reported the flowchart of the mutation subroutine for a design vector of $n$ elements.

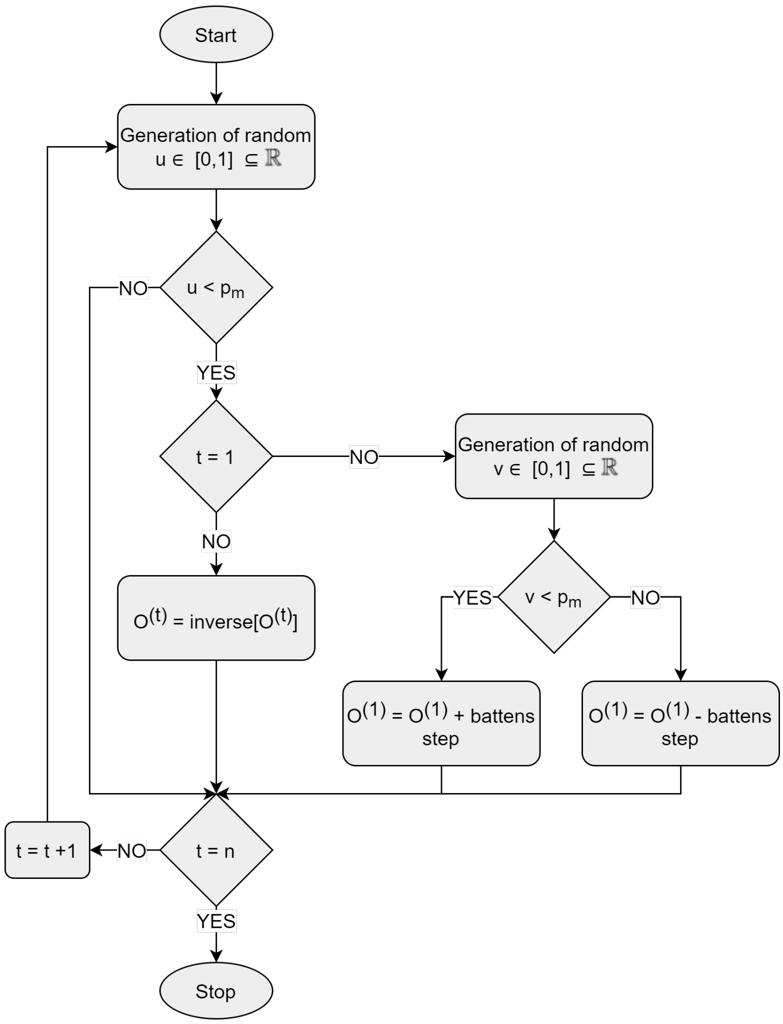



Figure 3.8: Flowchart of mutation operator

## 3.8 Structural analysis

The need to perform a large number of non-linear structural analyzes by varying the parameters that define the characteristics of the material led to the choice of the *Opensees* software which allowed to perform a fibre model of the structural elements.

This proves to be convenient since it allows to carry out non-linear analyzes on these elements since each fibre of the element is assigned a non-linear constitutive law.

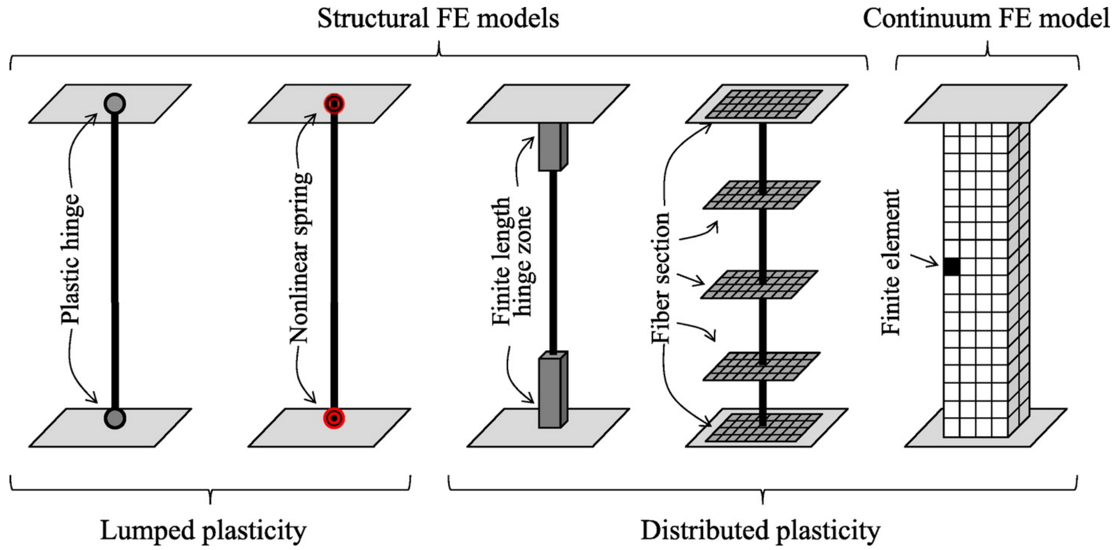



Figure 3.9: Types of models of frame elements (Deierlein et al. 2010 [4])

In this way, it is possible to perform analysis of distributed plasticity elements, overcoming the uncertainty of concentrated plasticity analysis due to the determination of the size of the plastic hinge. However, this method requires a more significant computational effort, in the face of a more realistic behaviour of the element.

Fibre elements are essential of two types Force Based Elements (FBE) and Displacement Based Elements (DBE). The first one is the classic finite element approach in which the deformation of the element is interpolated, starting from the approximation of the displacement field. The principle of virtual works is then used to derive the nodal forces. To interpolate the deformations, linear shape function is used for the axial displacement and quadratic ones for the transverse displacement; for these reasons, a constant axial deformation and a linear curvature are thus obtained. This shape functions are evaluated as the exact solution of the Bernoulli beam equations.

$$N'' = 0 \qquad\qquad H^{IV} = 0 \tag{3.15}$$

where $N$ is the shape function for the axial displacement and $H$ is the shape function for the transverse displacement.

Therefore, due to this approximations, a dense discretization is necessary to be able to grasp the real deformation field.

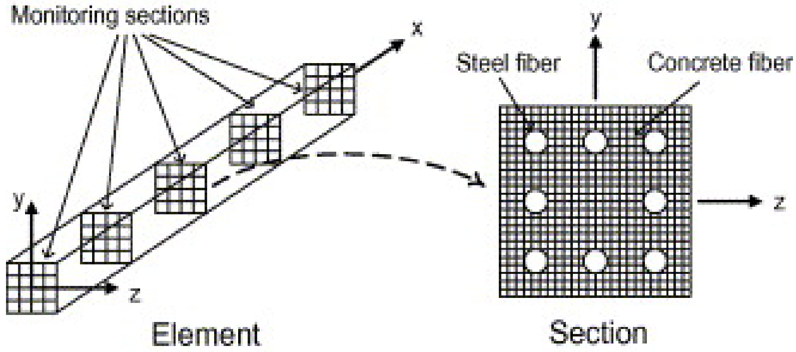



Figure 3.10: Element and section discretization

For Forced Based elements, however, dense discretization is not required, as the approximation will be adequate thanks to the use of control sections defined by the Gauss-Lobatto integration points.

For the purpose of this work a parametric FE model of forced based elements was created to perform a static non-linear analysis in the framework on N2 (Chapter 4.3)

# Chapter 4

# Finite element model

The case study building consists of a five-storey reinforced concrete structure obtained through simulated design to resist only gravity loads. This type of structure is representative of the class of reinforced concrete existing structures built in the middle of 1900. The structure has a very simple construction typology, regular in plan and in elevation. Three-dimensional representation of the structure is reported in Figure 4.1. Dimension in plan are represented in Figure 4.2 as well as dimensions of beams and columns.

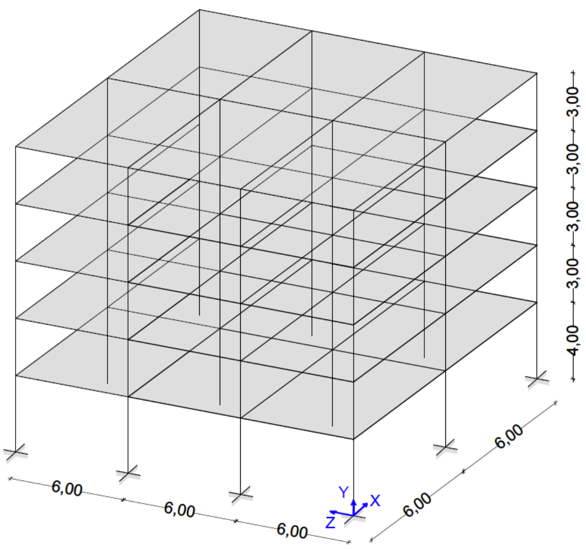



Figure 4.1: 3D frame view of the geometrical dimensions of the case study structure

Reinforcement details of beams and columns are listed in Table 4.1. The building is supposed being located in Cosenza (Italy), soil type C. The reference nominal life (VN) is of 100 years. The resulting return period is TR=975 years.

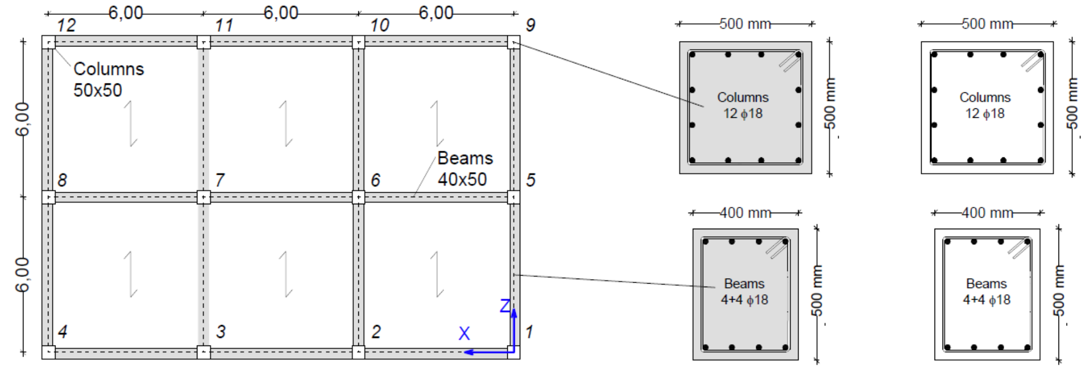



Figure 4.2: Geometrical dimensions of the case study structure

| RC element | b×h (mm) | Longitudinal reinforcement | Transversal reinforcement |
|---|---|---|---|
| Beam | $400 \times 500$ | $4 + 4\phi18$ | $\phi6/200\,\mathrm{mm}$ |
| Columns | $500 \times 500$ | $12\phi$ | |

Table 4.1: Reinforcement details of beams and columns

| | | |
|---|---|---|
| $a_g$ | $0.359\,g$ | design ground acceleration |
| $F_0$ | 2.463 | amplification factor |
| $T_b$ | $0.179\,\mathrm{s}$ | |
| $T_c$ | $0.576\,\mathrm{s}$ | corner periods in the spectrum |
| $T_d$ | $3.037\,\mathrm{s}$ | |
| $S$ | 1.169 | soil factor |
| $\eta$ | 1 | damping correction factor |

Table 4.2: Parameters of elastic response spectrum

The general assumptions at page 31 are applied as follows:

1. Steel angles have lateral length $l_a = 100\,\mathrm{mm}$ and thickness $t_a = 5\,\mathrm{mm}$.

2. The thickness of the battens ($t_b$) is 5 mm, the width ($w_b$) is 50 mm.

3. Yielding strength of steel angles and battens is $f_{yb} = 275\,\mathrm{MPa}$ and their spacing is the same for all the retrofitted columns.

Moreover, as suggested in the the general formulation of the optimization framework, the following restrictions are applied to re- duce the dimension of the designs space:

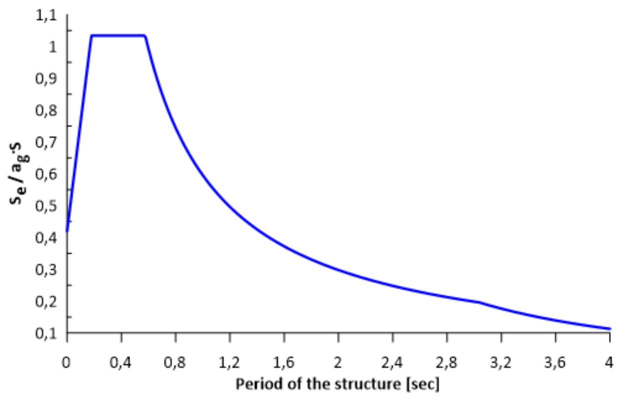



Figure 4.3: Elastic response spectrum

4. Retrofitted columns can be only located within the first the second floor.
5. Minimum and maximum spacings between the battens are 150 mm and 400 mm respectively.
6. Battens spacing can change only by step of 50 mm

Assumption 4) is justified by the fact that the maximum deformation demand is expected at the first two (of five) stories. Assumption 5) is done to limit possible battens spacing into a feasible range of values and assumption 6) is done in order to reduce the research space dimension.

Based on the aforementioned assumptions, the design vector ($\boldsymbol{b}$) components ($s_b$ and $\boldsymbol{p}$) are specialized as:

$$s_b \in S = [150,\ 200,\ 250,\ 300,\ 350,\ 400] \tag{4.1}$$

and $\boldsymbol{p}$ is a $24 \times 1$ vector collecting the positions of the columns at the first two floors.

The resulting size of the design space is then of 25 variables and consequently a research space of $6 \cdot 2^{24} \approx 10^8$ different solutions.

Reinforced concrete frame elements (beams and columns) are modelled adopting distributed plasticity force-based elements with five Gauss-Lobatto integration points available in OpenSees.

The subroutine *ModelCreator* (Appendix 7.4) defines all the nodes and elements of the model. The subroutines *ConfinedConcreteSR* (Appendix 7.14) and *ConfinedConcreteBattens* (Appendix 7.15), developed in Tcl, the constituve laws of confined and unconfined concrete according to the models presented in the next paragraphs of this chapterand the characteristics of steel-jacketing and stirrups arrangements.

According to the design vector transmitted from the *MainCode* function to *ModelCreator*, a different constitutive law is associated to each vertical elements (Figure 4.4).

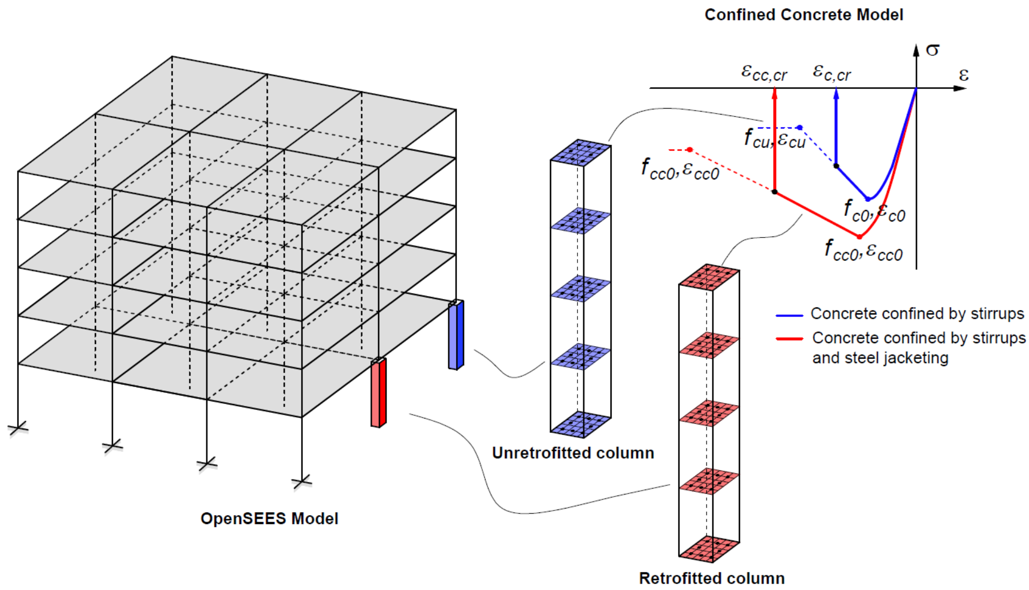



Figure 4.4: Definition of the fiber-section elements in OpenSees with and without considering the steel-jacketing reinforcement

## 4.1 Retrofitting system - Steel jacketing

Several strengthening systems utilize the benefits produced by the lateral confinement of reinforced concrete columns to increase strength and the ductility.

These include traditional steel stirrups, FRP wraps, steel jacketing, concrete jacketing, a system using angles with smoothed edges and pretensioned steel ribbons, etc. Among these the steel jacketing systems stands out for its effectiveness and low cost [29] for these reasons this method is widely used in many countries.

This technique is a decades-old system that utilizes both steel angles and strips. There are several arragements of steeljackets (Wu et al. 2006 [30], Figure 4.5) but the most common is realized applying four steel angles to the corners of RC members.

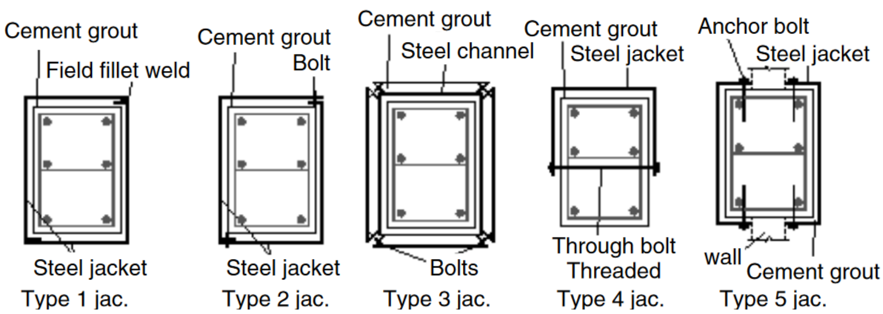
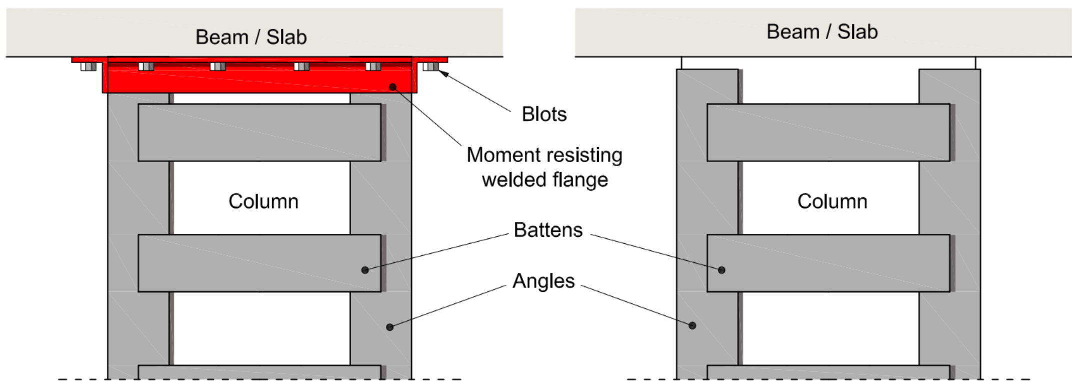



Figure 4.5: Some type of steel jacketing arrangements

The angle pieces are connected transversally by discontinuous steel strips welded to the angles. This strengthening technique for RC columns improves both bearing capacity and ductility, reduces the risk of buckling of main bars under compression, and improves the bond action with concrete (Campione et al. 2010 [31]).

Depending on the structural details of the beam-to-column joint location, the steel angles can be considered to act both in tension and in compression, only in compression or, finally, can be considered as providing a confining effect only.

In fact, only when connection between angles of different storey is effectively realized without interruptions, they can be considered acting in tension and in compression. In this case the angles can be realized with end plates connected to the floor in order to assure that the angle will work in compression, but it is not able to transfer the tension to the angles. In this case the angles can be realized with end plates connected to the floor in order to assure that the angle will work in compression, but it is not able to transfer the tension to the angles (Figure 4.6.a).




*a)* *b)*

Figure 4.6: Column steel-jacketing arrangements: (a) cage with moment resisting end connections; (b) cage without end connections

Finally, when no attention is given to the realization of the structural details regarding the connection of the angles to the relative floors (Figure 4.6.b) the angles cannot be considered as additional longitudinal reinforcement. In this case the angles have to be considered as confining elements only.

Modelling of steel jacketing in fiber-section elements has been addressed by Campione et al (2017) [32] who provided that, for the case in which only confinement is considered, steel angles are not included in the cross-section assembly. On the contrary, in case of full flexural connection, also angels are discretized into fibers having specific uniaxial behaviour. It is supposed that steel-jacketing is arranged without realizing moment resisting connection at the top and the bottom of the columns, while frictional contribution to the resistance (Campione et. al 2017 [32]) is neglected.

## 4.2 Materials

The structure is supposed to be arranged with poor resistance concrete having average unconfined strength $f_{c0m} = 20\,\mathrm{MPa}$. Steel rebars have average yielding strength $f_y = 455\,\mathrm{MPa}$.

In the following sections are presented the model used to define the material of the FEM analysis.

### 4.2.1 Reinforcement steel

The Giuffré-Menegotto-Pinto model with isotropic strain hardening is assumed for modeling the behavior of reinforcement steel.

It is defined by the following mechanical parameters:

| $f_y$ [MPa] | $E_s$ [MPa] | $b\,[-]$ |
|---|---|---|
| 455 | 210000 | 0.01 |

Table 4.3: Mechanical properties of the steel

where $f_y$ is the yielding stress, $E_s$ is the elastic modulus and $b$ is the strain-hardening ratio.

The parameters that control the transition from elastic to plastic branches were defined from recommended values, in particular $R_0 = 15$, $c_{R1} = 0.925$, $c_{R2} = 0.15$.

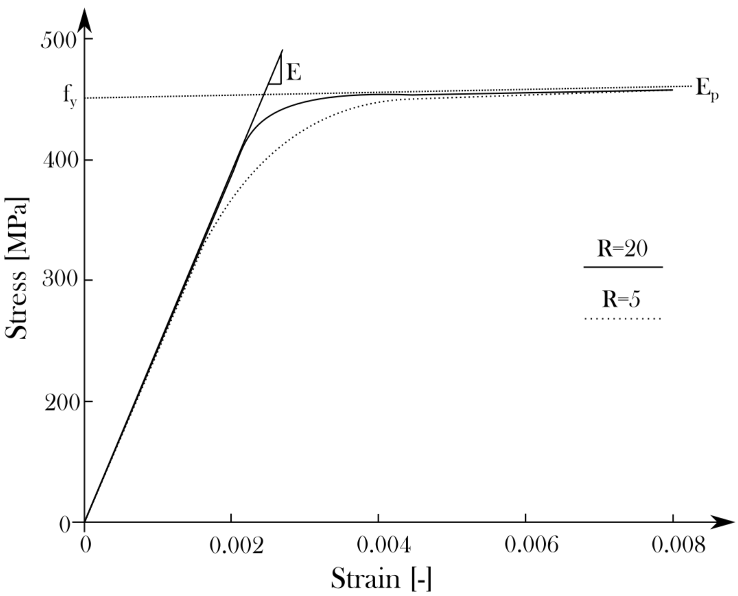



Figure 4.7: Costitutive law of *Steel02*

### 4.2.2 Concrete

Among the materials present in the library of *OpenSees*, the *Concrete 02* uniaxial material model is assigned to the cross-section fibers. For sake of simplicity it is assumed that the effect of confinement is extended to the whole cross-section both for the cases of columns with and without reinforcement. This simplified assumption is used to obtain a formal consistency with the confinement model in the case of concrete confined by stirrups and steel jacketing which provides uniform confinement over the cross-section (Figure 4.9).

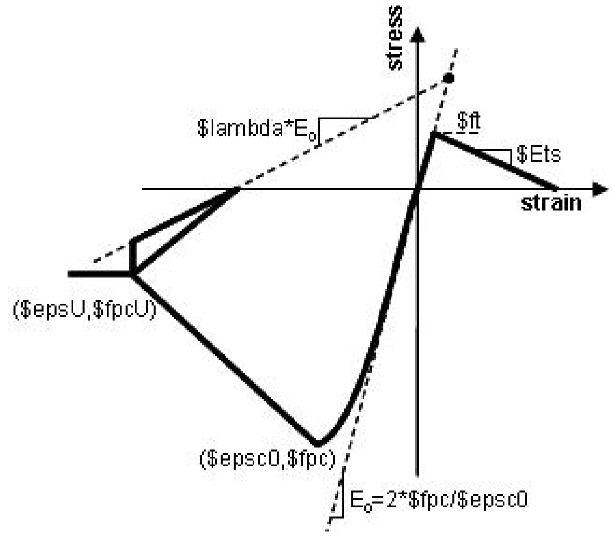



Figure 4.8: Costitutive law of *Concrete02*

In order to simulate the crushing of the cross-section fibers, Concrete02 material is combined with *MinMax* material which removes the contribution of the fiber when a specified strain threshold is achieved. For the current case, it is assumed that the crushing of fibers occurs in correspondence of the compressive strain ($f_{cr}$) attained at a 30% reduction of the peak strength.

Confined concrete parameters for the RC elements confined only by stirrups are evaluated using the stress-strain model by Saatchioglu and Razvi (1992) [5]. As for the columns with steel jacketing retrofitting, confined concrete parameters are obtained following the approach by Montuori and Piluso (2009) [33] as described in detail in the following section.

### Confinement by stirrups

**Strength of confined concrete** For RC elements confined only by stirrups the strength can be calculated as:

$$f_{cc} = f_{c0} + K_1 \cdot f_f$$

where:

$f_{c0}$ the strength of the unconfined concrete

$K_1$ parameter function of Poisson's coefficinent in non-linear beaviour

$f_l$ lateral confining pressure

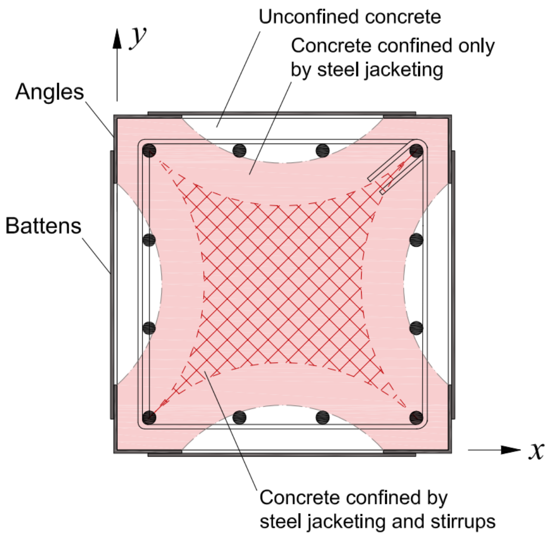



Figure 4.9: Effectively confined area by stirrups and steel jacketing

Due to the difficulty to estimate the Poisson's coefficient in the non-linear branch, the value of $K_1$ can be estimated from experimental results as:

$$K_1 = \frac{f_{cc} - f_{c0}}{f_l}$$

from the results of the experimental campaign accomplished by [34] on cylinder speciments confined by hydrostatic pressure, [5] recommend the following exponential formulation:

$$K_1 = 6.7 \cdot (f_l)^{-0.17}$$

In the case of closed stirrups, the lateral pressure of confinement is detemined from simple equilibrium observation (Figure 4.10):

$$n \cdot A_s \cdot f_y = s \cdot b_c \cdot f_l \qquad \Rightarrow \qquad f_l = \frac{n \cdot A_s \cdot f_y}{s \cdot b_0}$$

The pressure provided by closely spaced circular spirals and vertical column reinforcement can be considered to be uniform around the perimeter of the cross section.

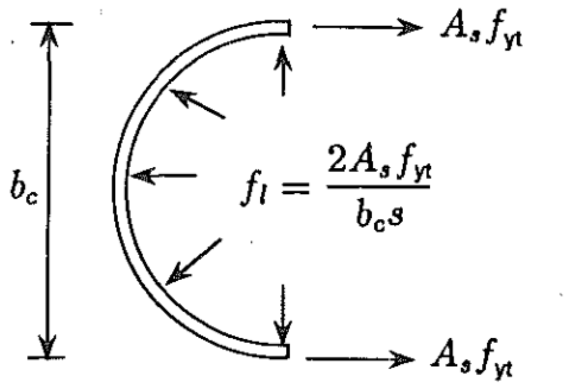



Figure 4.10: Confinement pressure accomplished by closed stirrups

In the case of prismatic elements, the confining is a three-dimentional that can not be reduced to a sectional level. The lateral pressure between the ties reduces with the distance from the longitudinal reinforcement. This reduction occurs at a faster rate than that of the pressure at the tie level.

The reaction of the stirrups between the corners is conditioned by the stiffness of the rebars, the distance between the tie points and the elastic modulus of the steel used.

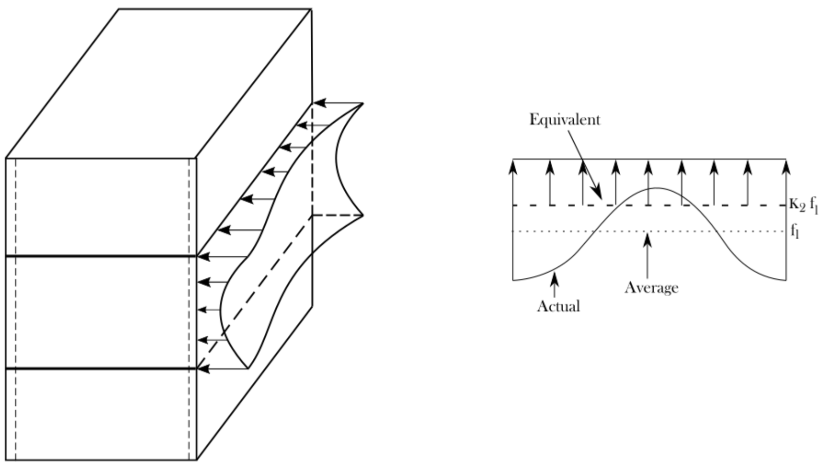



Figure 4.11: Lateral confining pressure on prismatic elements

The equivalent uniform pressure $f_{le}$ can be established by reducing the average pressure with due considerations with the coefficient $k_2$, as:

$$f_{le} = K_2 \cdot f_l \tag{4.2}$$

In the case of premature buckling of longitudinal reinforcement prevented, the value of $K_2$ can be estimated as:

$$K_2 = 0.26 \cdot \sqrt{\left(\frac{b_c}{s}\right)\left(\frac{b_c}{s_l}\right)\left(\frac{1}{f_l}\right)} \leq 1.0 \tag{4.3}$$

For rectangular cross-section element, the lateral confinement pressure is:

$$f_{le} = \frac{f_{lex} \cdot b_{cx} + f_{ley} \cdot b_{cy}}{b_{cx} + b_{cy}} \tag{4.4}$$

where:

$f_{lex/y}$ the effective lateral pressure perpendicular to the direction x/y

$b_{cx/y}$ the dimentions of the cross section

**Ductility of confined concrete** In addition to increasing the strength of the concrete elements, lateral confinement increases deformability Confined concrete can sustain higher strains at the peak load, and may show little strength decay thereafter.

Several authors (Balmer (1949) [35], Mander (1989) [36], Saatchioglu et al. (1992) [5]) have experimentally validated the following expression for the determination of the peak deformation:

$$\varepsilon_1 = \varepsilon_{01} \cdot (1 + 5K) \tag{4.5}$$

where:

$$K = \frac{K_1 \cdot f_{le}}{f'_{c0}} \tag{4.6}$$

and $\varepsilon_{01}$ the peak deformation of unconfined concrete, a value of 0.002 may be appropriate under quasi-static loads.

The deformability of concrete in the post-peak branch is strongly influenced by the behavior of the longitudinal bars which, in those load conditions, as spalling effect occur, are no longer tied to instability by concrete cover.

Therefore, at this load stage the lateral support provided by transverse reinforcements becomes the most important.

For this reason, the amount of transverse reinforcement, expressed in terms of reinforcement ratio (p), play a major role on the descending slope of the stress-strain relationship.

$$\rho = \frac{\sum A_s}{s \cdot (b_{cx} \cdot b_{cy})} \tag{4.7}$$

Regression analysis of test data indicates that the following expression can be used to establish the strain at 85% strength level beyond the peak:

$$\varepsilon_{85} = 260 \cdot \rho \cdot \varepsilon_1 + \varepsilon_{085} \tag{4.8}$$

where $\varepsilon_{085}$ is the deformation at 85% of the peak stress of the unconfined concrete. For ordinary concretes, in unavailability of experimental results, a value of 0.0038 may be appropriate.

Definitely, the stress-strain law of concrete confined with stirrups can be defined by the following three equations:

a) elastic parabolic branc
assuming that in this load stage the confinement has a negligible effect, the law proposed by [37] may be appropriate also for the confined concrete:

$$f_c = f'_{cc} \cdot \left(2 \cdot \left(\frac{\varepsilon_c}{\varepsilon_1}\right) - \left(\frac{\varepsilon_c}{\varepsilon_1}\right)^2\right)^{\frac{1}{1+2K}}$$

b) post-peak linear branch
the gradient of the linear law is determined by forcing the passage to point $[0.85 \cdot f_{cc}; \varepsilon_{85}]$ thus:

$$f_c = f'_{cc} \cdot \left(1 - 0.15 \cdot \left(\frac{\varepsilon - \varepsilon_1}{\varepsilon_{85} - \varepsilon_1}\right)\right)$$

c) high displacement constant branch
reached the residual strength of 20% of the peak stress

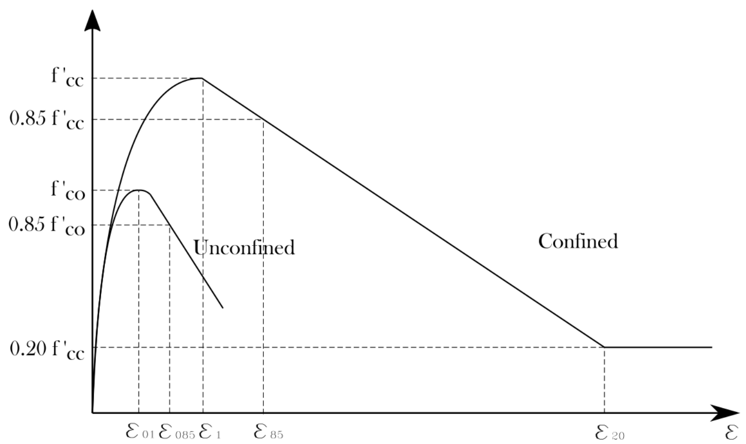



Figure 4.12: Stress-strain law of concrete confined by stirrups Saatchioglu et al. [5]

In the framework subject of this thesis, the definition of the concrete confined by stirrups is performed by a tcl function reported in Appendix 7.14.

### Confinement by battens

The effect of steel jacketing is intoduced in the retrofitted columns only as confinement action, as already described by Campione et al. (2017) [38] by simply modifying the constitutive law of concrete fibers.

The approach proposed by Montuori and Piluso (2009) [33] is combined with the expression provided by Saatchioglu and Razvi (1992) [5] as presented in the previous paragraph. This model has been used for predicting the load carrying capacity of retrofitted columns.

The confinement effect exerted by the battens of the steel jacket system sums up with that of stirrups producing different confinement levels over the cross-section as shown in Figure 4.9.

However, given that the steel jacketing confining action is prevailing,the model provides the use of a single concrete stress–strain law for the entire section. This assumption has demonstrated to be sufficiently reliable in comparison with experimental results (Braga et al. (2006) [39], Campione et al. (2017) [38]).

The lateral confinement pressure along the two directions of the cross-section are evaluated as:

$$\begin{aligned} f_{le,x} &= k_e \cdot \rho_{st,x} \cdot f_y \\ f_{le,y} &= k_e \cdot \rho_{st,y} \cdot f_y \end{aligned} \tag{4.9}$$

in wich the calculation of the transverse reinforcement volumetric ratios consider both the contribution of internal and external reinforcement as:

$$\begin{aligned} \rho_{st,x} &= \frac{n_{bx} \cdot A_{st,x} \cdot b_0}{s \cdot b_0 \cdot h_0} + \frac{2 \cdot A_{sb,e} \cdot b}{s_b \cdot b \cdot h} \\ \rho_{st,y} &= \frac{n_{by} \cdot A_{st,y} \cdot h_0}{s \cdot b_0 \cdot h_0} + \frac{2 \cdot A_{sb,e} \cdot h}{s_b \cdot b \cdot h} \end{aligned} \tag{4.10}$$

the coefficient $k_e$ expresses the effectivelly confined area through the expression:

$$k_e = \left(1 - \frac{s_b - \phi_{st}}{2 \cdot b_0}\right) \cdot \left(1 - \frac{s_b - \phi_{st}}{2 \cdot h_0}\right) \tag{4.11}$$

In Eqs. 4.10 and 4.11

$b$ the cross-section base

$h$ the cross-section height

$c$ the width of concrete cover

$b_0$-$h_0$ the concrete confined by stirrups equal to $b_0 = b - 2 \cdot c$ and $h_0 = h$

$n_{bx}$-$n_{by}$ the number of stirrups arms along x and y

$A_{st,x}$-$A_{st,y}$ the area of the stirrups along x and y

$A_{sb}$ the transverse area of a batten

$\phi_{st}$ the diameter of the stirrups

$s$ the spacing of internal hoops

$s_b$ the spacing of external battens

$A_{sb,e}$ the mechanically equivalent transverse area of battens, calculated as:

$$A_{sb,e} = A_{sb} \cdot \frac{f_{yb}}{f_y}$$

The confinement parameters of the constitutive law are evaluated by using the expressions provided by Saatchioglu et al. [5] as presented in the previous paragraph.

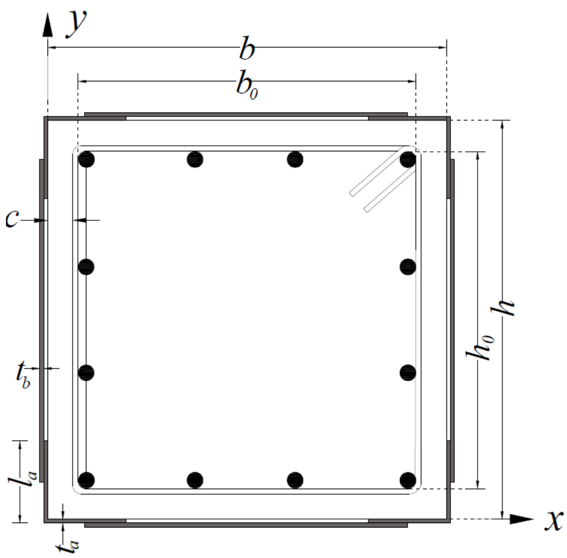



Figure 4.13: Geometric arrangement of cross-section of a column reinforced by steel jacketing

In order to include the effect of the steel jacketing, the term of the reinforcement ratio $\rho_{st}$ is modified as [40] propose to calculate as:

$$\rho_{st} = \frac{A_{st,x} + A_{st,y} + 4 \cdot A_{sb,e}}{\widetilde{s} \cdot (b_0 + h_0)}$$

where $\widetilde{s}$ represents the averege stirrups/battens spacing that is:

$$\widetilde{s} = \frac{s + s_b}{2} \tag{4.12}$$

Samples of the resulting stress–strain response in compression for a reference column cross-section fibers are reported in 4.14 considering the non-retrofitted case and the cases of steel-jacketing reinforcement with different battens spacing.

In the framework subject of this thesis, the definition of the concrete confined by stirrups is performed by a tcl function reported in Appendix 7.15.

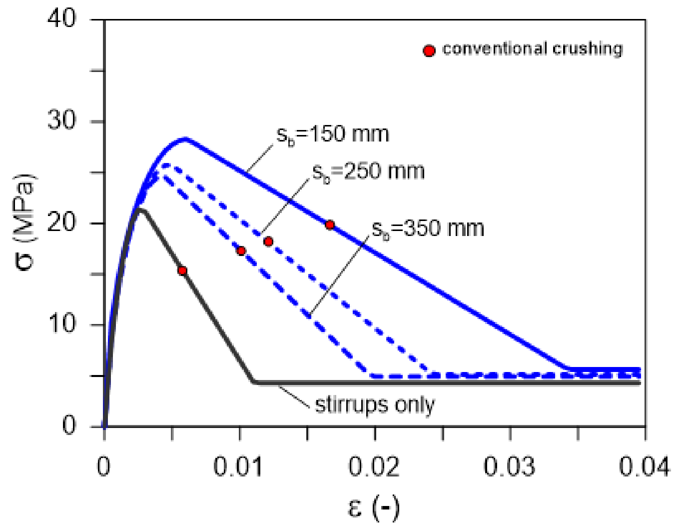



Figure 4.14: Sample of stress–strain response of concrete in compression for a reference column with and without steel-jacketing

### 4.2.3 Infills

Infills are modelled as fiber-section struts according to the model by Di Trapani et al. [41] (Figure 4.4). The model provides using a concrete-type compression-only stress–strain relationship defined by evaluating four parameters, peak stress ($f_{md0}$), ultimate stress ($f_{mdu}$), peak strain ($\varepsilon_{md0}$) and ultimate strain ($\varepsilon_{mdu}$) wich are obtained by semi-empirical equations.

Geometric and mechanical parameters of the struts are reported in the following Table 4.4 are consistent with a clay hollow masonry infill having thickness $t$ = 250 mm, elastic modulus $E_m = 6400$ MPa, compressive strength $f_m = 8.6$ MPa and shear strength $f_{vm} = 1.07$ MPa.

| $w$ [mm] | $t$ [mm] | $f_{md0}$ [MPa] | $f_{mdu}$ [MPa] | $\varepsilon_{md0}$ [−] | $\varepsilon_{mdu}$ [−] |
|---|---|---|---|---|---|
| 250 | 1053 | 1.88 | 0.86 | 0.013 | 0.073 |

Table 4.4: Geometric and mechanical details of the masonry infill equivalent strut

.

where:

$w$ is the width of the strut

$t$ is the thickness of the strut

$f_{md0}$ is the peak stress

$f_{mdu}$ is the stress treshold of linear branch

$\varepsilon_{md0}$ is the strain at the peak

$\varepsilon_{mdu}$ is the strain in correspondence of the linear branch

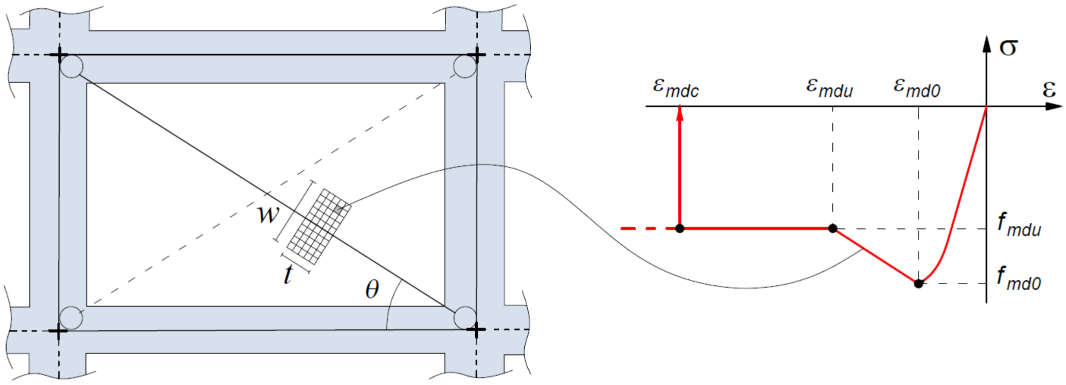



Figure 4.15: Equivalent strut model for the masonry infills

To model the crushing of the infills masonry, *Concrete02* material is combined with *MinMax* material, wich removes the contribution of the elemen when a specified strain threshold is achieved. In particular, for the current case it is assumed that the crushing of the infills occurs in correspondence of a strain value equal to the double of the plastic behaviour threshold ($\varepsilon_{mdc} = 2 \cdot \varepsilon_{mdu} = 0.0145$).

## 4.3 Pushover analysis

The N2 method, introduced by Fajfar [42] and provided as standard procedure in Eurocode 8 [43] and in the Italian Technical Code [44] was used for the aim of this study.

The capacity curve of the structure was determined imposing two monotonically increasing profiles of lateral forces. The first one was proportional to the product of the first modal shape and the diagonal matrix of the storey masses M. A second distribution consisted of the force profile proportional to the storey masses. In the model presented in this thesis, in order to reduce computational effort, pushover analyses are carried out by considering only a uniform profile for lateral loads.

The bilinear base shear against top displacement ($V^*$ - $d^*$) capacity curves of the SDOF systems equivalent to the MDOF one were obtained after dividing both base shear and top displacement of the pushover curve (which was cut off to an ultimate strength not lesser than the 85% of the peak strength) for the first participation factor defined as:

$$\Gamma_1 = \frac{\phi^T \cdot M \cdot I}{\phi^T \cdot M \cdot \phi} = \frac{\sum m_i \cdot \phi_i^2}{\sum m \cdot \phi_i} \tag{4.13}$$

where $M$ is the diagonal mass matrix, $\phi$ is the eigenvector associated to the first vibration mode (normalized to the top displacement $\phi_n = 1$), $I$ is the unit vector and $m_i$ is the concentrated mass at the $i^{th}$ storey. The value in the denominator represents the mass of the equivalent SDOF system ($m^* = \sum m_i \cdot \phi_i$)

Through the bilinearization of the SDOF capacity curve imposing the area under the curves equality (Figure 4.16), the stiffness $k^*$ associated to each SDOF system response was calculated in agreement to the rules of the N2 method as:

$$k^* = \frac{F_y^*}{d_y^*}$$

where $F_y^*$ and $d_y^*$ are respectively the yielding force and the corresponding displacement, from which the related period T* is calculated as:

$$T^* = 2\pi \cdot \sqrt{\frac{m^*}{k^*}}$$

where $m^*$ is the mass of the SDOF system.

The ductility demand ($\mu_d$) of an inelastic SDOF system is calculated according to the method proposed by Vidic et al. [45] as:

$$\begin{cases} \mu_d = (q^* - 1)\,\frac{T_c}{T^*} + 1 & \text{if } T^* \leq T_c \\ \mu_d = q^* & \text{if } T^* > T_c \end{cases} \tag{4.14}$$

where $q^*$ is the reduction factor evaluated from the elasti spectral acceleration $S_{ae}\left(T^*\right)$ as:

$$q^* = \frac{S_{ae}\left(T^*\right) \cdot m^*}{F_y^*} \tag{4.15}$$

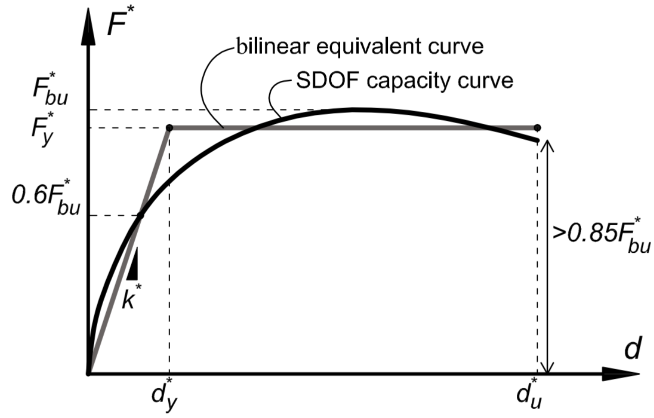



Figure 4.16: Equivalent SDOF capacity curve and bilinear equivalent curve

Eventually, the ductility capacity ($\mu_c$) is calculated from the bilinear curve as:

$$\mu_c = \frac{d^*_u}{d^*_y} \tag{4.16}$$

The capacity/demand ratio ($\xi_\mu$) is finally:

$$\xi_\mu = \frac{\mu_c}{\mu_d} \tag{4.17}$$

The coefficient $\xi_\mu$ is the final output of the processing of pushover curves and is used as a discriminating factor in the optimization process in order to establish if a single individual passes the verification check ($\xi_\mu \geq 1$) or not ($\xi_\mu < 1$). Different reference $\xi_\mu$ values can be eventually adopted if higher or lower target safety factor are selected.

## 4.4 Shear verification

The shear verification of columns is performed according to the model proposed by Biskinis et al. (2004) [46]. This theory is provided as standard procedure in Eurocode 8 (EN1998-3 §A.3.3.1) [47] and in the Italian Technical Code (explanatory circular to NTC18 §C8.7.2.3.5) [48] for the evaluation of shear strength of element subjected to cyclic loads.

For seismic actions, it is necessary to consider the reduction of shear strength in cyclical conditions as a function of the ductility demand on the element. The maximum shear demand in the element can be determined, regardless of the level of action considered, starting from the resistant moments in the end sections, assessed by amplifying the average resistances of the materials.

The cyclic shear resistance ($V_R$) decreases with the plastic part of ductility demand that can be expressed in terms of ductility factor of the transverse deflection of the shear span $\mu_\Delta^{pl} = \mu_\Delta - 1$ where $\mu_\Delta$ is the ductility demand defined as the maximum rotation and the yielding rotation ratio.

The shear resistance under cyclic loads can be evaluated as the sum of three different contribution, the first function of the compressive state of concrete, the second calculated from the axial load of the steel and the last that is function of the interaction interaction with the flexural rotation of the element as a function of the plastic part of the ductility demand $\mu_\Delta^{pl}$.

$$V_R = \frac{1}{\gamma_{el}} \cdot \left[\frac{h-x}{2 \cdot L_v} \cdot \min\left(N; 0.55 \cdot A_c \cdot f_c\right) + \left(1 - 0.05 \cdot \min\left(5; \mu_\Delta^{pl}\right)\right) \cdot \right.$$
$$\left. \left[0.16 \cdot \max\left(0.5; 100 \cdot \rho_{tot}\right) \cdot \left(1 - 0.16 \cdot \min\left(5; \frac{L_v}{h}\right)\right) \cdot \sqrt{f_c} \cdot A_c + V_w\right]\right] \tag{4.18}$$

where:

$\gamma_{el}$ is the partial safety factor, equal to 1.15 for primary seismic element

$h$ is the depth of cross-section

$x$ is the compression zone depth

$N$ is the compressive axial force

$L_v$ is the ratio moment/shear at the end section ($L_v = M/V$)

$A_c$ is the cross-section area, for prismatic elements it is equal to $A_c = b_w \cdot d$ where $b_w$ is the web width (the thickness) and $d$ is the structural depth

$f_c$ is the concrete compressive strength

$\rho_{tot}$ is the total longitudinal reinforcement ratio

$V_W$ is the contribution of transverse reinforcement to shear resistance, for rectangular cross-section it is equal to $V_w = \rho_w \cdot b_w \cdot z \cdot f_{yw}$ where $\rho_w$ is the transverse reinforcement ratio, $z$ is the length of internal lever arm and $f_{yw}$ the yielding stress of the transverse reinforcement.

Without specific assessments, the height of the compressed area of the section ($x$) can be calculated in a simplified way through the relation suggested by the Italian technical code:

$$\frac{x}{h} = 0.25 + 0.85 \cdot \frac{N}{A_c \cdot f_C} \leq 1 \tag{4.19}$$

### 4.4.1 Shear strength of retrofitted elements by steel-jacketing

In case of element retrofitted by steel jacketing the shear resistance must consider the contibution of the steel battens.

According to the Italian Techical code (§C8.7.4.2.2) [48] the contribution of the steel jacketing on shear resistance can be considered additional to the pre-existing resistance as long as the battems remains entirely in the linear elastic branch. This condition is necessary for it to limit the width of the cracks and ensure the integrity of the concrete, allowing the functioning of the resistant mechanism of the existing element.

The additional resistance ($V_j$) related to the steel jacketing can be calculated as:

$$V_j = 0.5 \cdot \frac{2 \cdot t_j}{s} \cdot b \cdot f_{yw} \cdot 0.9 \cdot d \cdot \cot(\vartheta) \tag{4.20}$$

where:

$d$ is the height of the cross-section

$t_j$ is the width of the battens

$b$ is the width of the cross-section

$s$ is the battens spacing

$f_{yw}$ is the yield stress of the steel used for the battens

$\vartheta$ is the inclination of the cracks according to Ritter-Mörsch model [49] [50]

In the Figure 4.17 is illustrated the plot of the functions that model the shear resistance related to compression failure, stirrups tension collapse and the contribution of the steel jacketing varying the inclination of the crack for a beam that has the geometrical and mechanical property reported in Table 4.5 without any axial load ($N = 0\,\mathrm{N}$).

| $d$ [mm] | $b_w$ [mm] | $A_{sw}$ | $s$ [mm] | $f_{cd}$ [MPa] | $f_{yd}$ [MPa] | $t_j$ [mm] | $b_j$ [mm] | $s_j$ [mm] |
|---|---|---|---|---|---|---|---|---|
| 300 | 250 | $2 \times \phi 10$ | 180 | 15 | 400 | 5 | 20 | 250 |

Table 4.5: Parameters of the example illustrated in Figure 4.17

where:

$d$ is the width of the cross-section

$b_w$ is the thickness of the cross-section (web width)

$A_{sw}$ is the area of all stirrups arm

$s$ is the stirrups spacing

$f_{cd}$ is the concrete ultimate strength

$f_{yd}$ is the stirrups yield stress

$N$ is the axial force acting on the section

$t_j$ is the thickness of the battes

$b_j$ is the width of the battens

$s_j$ is the battens spacing

$V_{Rcd}$ is the shear resistance of the member without shear reinforcement

$V_{Rsd}$ is the shear force which can be sustained by the yielding shear reinforcement

It is easy to observe that the presence of a steel jacketing significantly increases the resistance to shear collapse by increasing the inclination of the cracks.

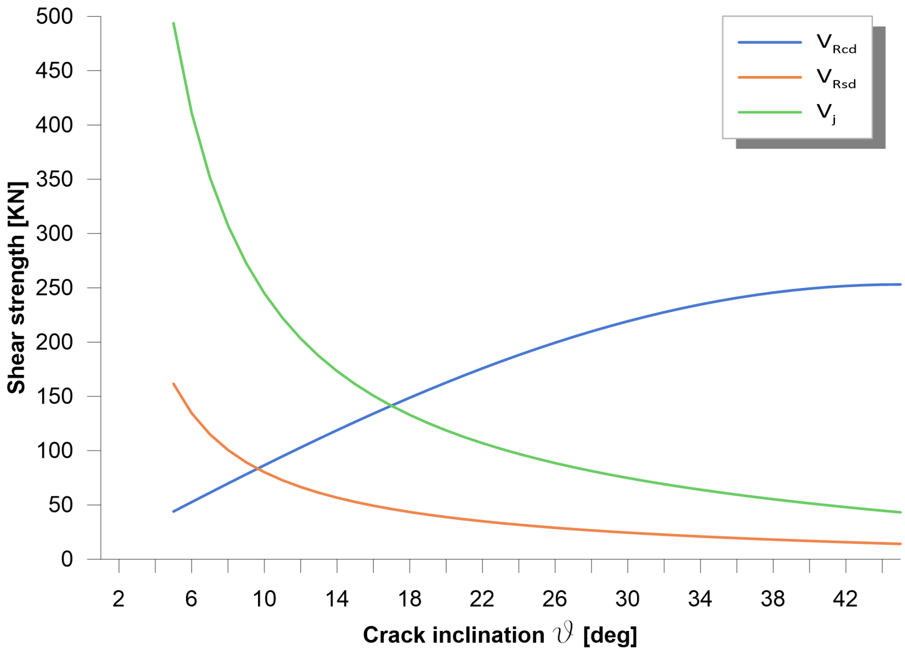



Figure 4.17: Example of shear resistance trend by varying the crack inclination

In the framework subject of this thesis, the resistance of the columns is calculated automatically by the subroutine *ShearStrength* reported in Appendix 7.7.

### 4.4.2 Shear demand on columns for infilled frame

Masonry infills may induce shear collapse of frames because of excess of shear demand at the end of columns. The actual shear demand on columns can be directly evaluated by using a multi-strut macro-model for the infills. In case of single concentric strut the infills contribution to shear demand can be estimated by using the following expression based on simple equilibrium considerations proposed by Di Trapani and Malavisi (2019) [51].

$$V_{C,inf} = P_{str} \cdot \cos\alpha - \mu \cdot P_{str} \cdot \sin\alpha \tag{4.21}$$

where, referring to Figure 4.18 $V_{C,inf}$ is the additional shear demand actually transferred from the infill to the colum, $P_{str}$ the current value of the axial force acting on the equivalent strut, $\alpha$ is the angle of inclination of the strut with respect to horizontal direction, and $\mu$ the friction coefficient associated with the infill-mortar-frame interface.

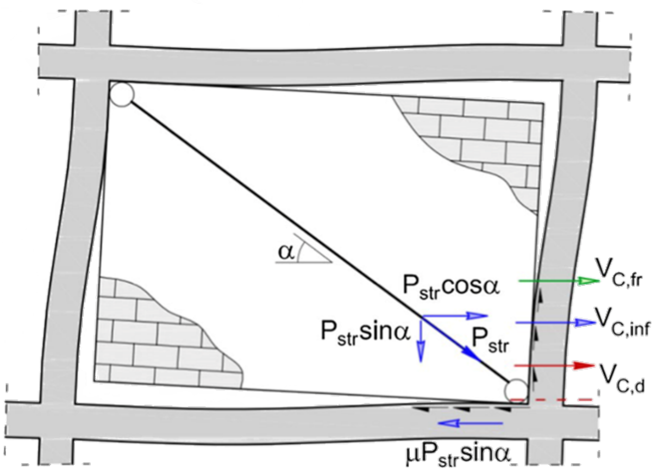



Figure 4.18: Simplified scheme for the determination of actual shear demand on columns for infilled frame

Shear limit state is expressed by the following condition:

$$V_{C,d} = V_{C,fr} + V_{C,inf} \leq V_{Rd} \tag{4.22}$$

where $V_{C,fr}$ is the shear force evaluated on the frame (in any section of a column), and $V_{Rd}$ the shear capacity of the column calculated as presented in previous Chapter 4.4.

For the purpose of this work the frictional coefficient is established equal to $\mu = 0.7$.

# Chapter 5

# Study cases and validation of the method

## 5.1 Validation of the framework and calibration of parameters

The structure reported in Chapter 4 has vertical elements highly shear sensitive, this design choice was opted for testing the effectiveness of the framework to brittle collapse sensitive structures. In these cases the structures require a high number of columns retrofitted by steel-jacketing for flexural-ductility lack but specially for shear strengthening.

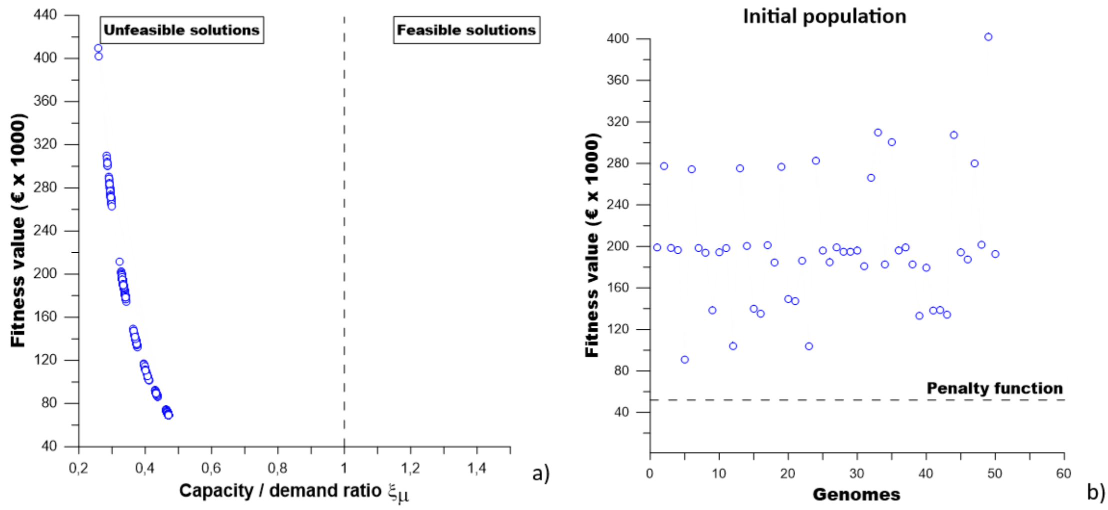



Figure 5.1: Outputs of analysis accomplished by standard GA of Matlab library (a) Objective function values as a function of $\xi_\mu$, (b) Convergence history

As illustrated in Figure 5.1, by choosing a completely random initial population, the probability of obtaining verified solutions is so low that the algorithm explores only the field of unfeasible solutions.

The most favorable solution that was find during this work of thesis, involves the generation of the initial population that presents individuals with a high number of reinforced columns. By means of the *RandomDVGenerator.m* subroutine (Chapter 3.4), several analyses were performed obtaining some feasible solutions (Figure 5.2).

But, by using the standard genetic algorithm present in the official libraries of Matlab, since the vector design consists of integers, the mutation function is suppressed. This can be seen from Figure 5.2 where it is observed that the combination of unfeasible individuals leads to the stall of the algorithm. The framework can not find further feasible solutions only by means of crossover operator. The solution

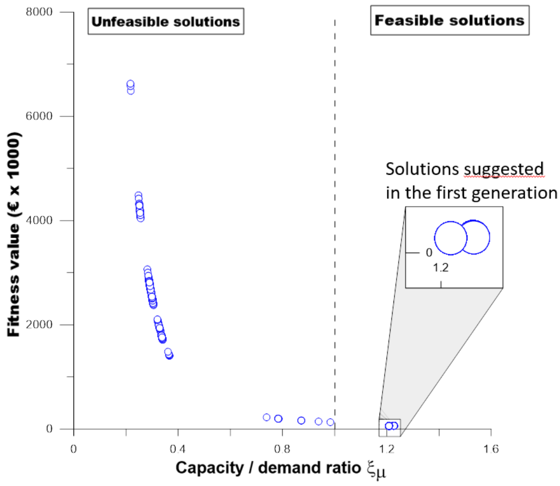



Figure 5.2: Objective function values as a function of $\xi_\mu$ of analysis performed by standard GA of Matlab

found during the work of this thesis and proposed therein concerns the use of an initial population composed of individuals that represent intervention arrangement with a high number of retrofitted columns ($90\% \div 95\%$) in association with a high mutation ratio.

The genomes with high number of retrofitted columns allow to "tend" the generation toward the feasible research subspace, the high mutation ratio ($p_m \simeq 0.025 \div 0.05$) allows thoroughly exploration of all the possible solutions.

As can be seen from the Figure 5.3 this new approach leads to find different solutions that have low ductility ratio ($\xi_\mu$) with optimised intervention costs.

Another factor that improves the algorithm performance is elitism. Elitism involves copying a small number of the fittest candidates, unchanged, into the next generation. This function allows the analysis not to lose good genetic heritage that can be occurs during the crossover operations.

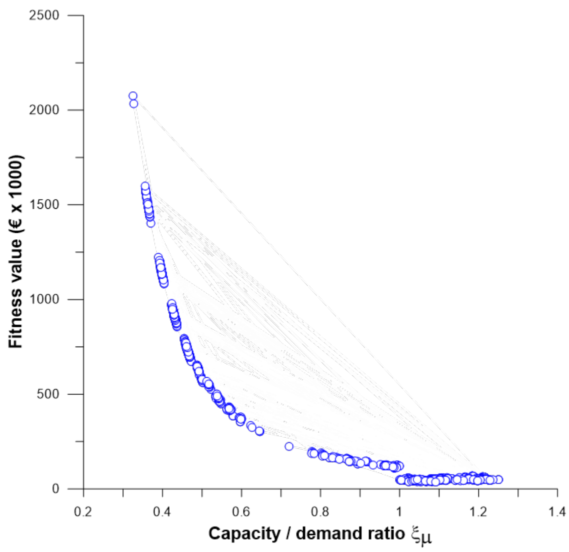



Figure 5.3: Objective function values as a function of $\xi_\mu$ of analysis performed using the proposed recommendation

### 5.1.1 Calibration of parameters

As usual for this type of algorithm, an initial calibration phase of the parameters is necessary to make the framework efficient and fast.

In particular, two diffent parametric set of analyses were accomplished, the first one is performed by varying the population dimension (80 and 120 individuals, Figure 5.4 and Figure 5.5).

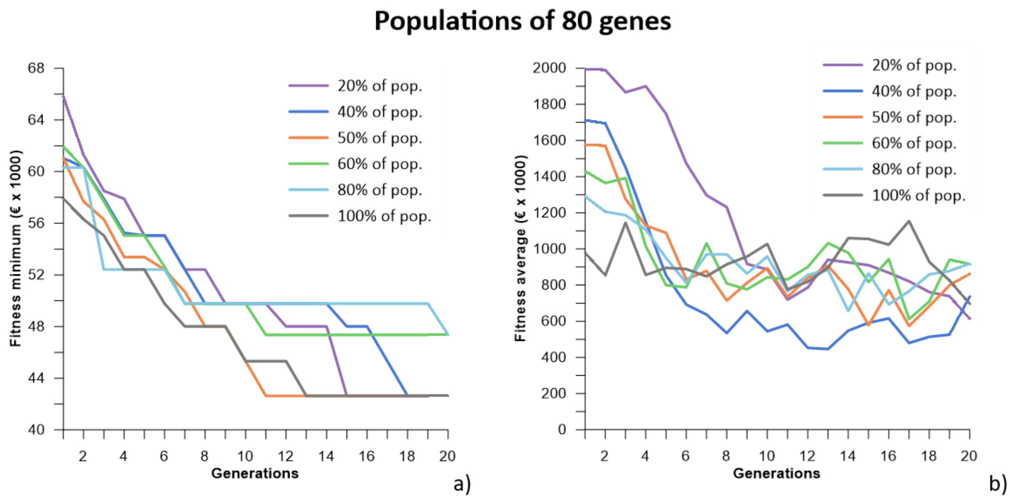



Figure 5.4: Influence of the number of individuals with 95% of retrofitted columns in the initial population: 80 individuals every population - (a) fitness minimum, (b) fitness average

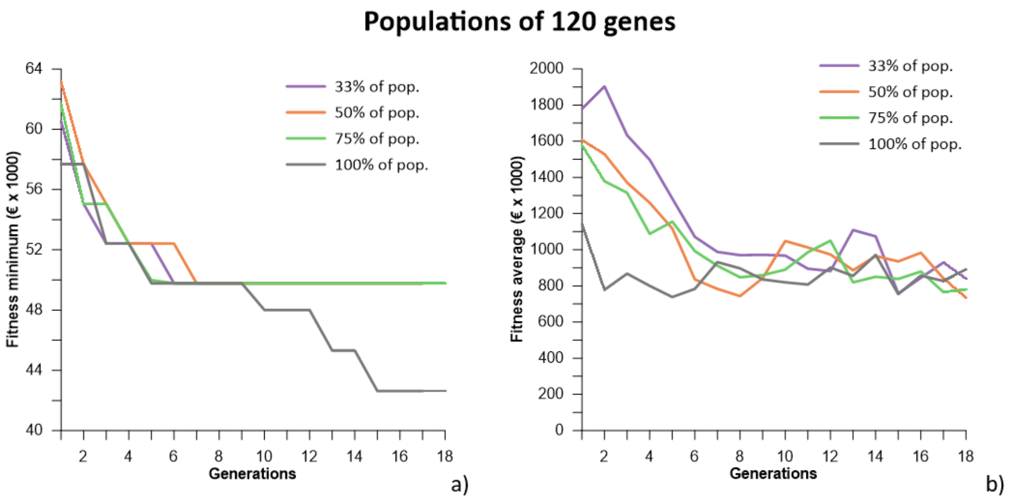



Figure 5.5: Influence of the number of individuals with 95% of retrofitted columns in the initial population: 120 individuals every population - (a) fitness minimum, (b) fitness average

The second parametric set of analyses is carried out by changing the mutation ratio ($m_r$), in Figure 5.6 the trends of the optimal solutions are illustrated.

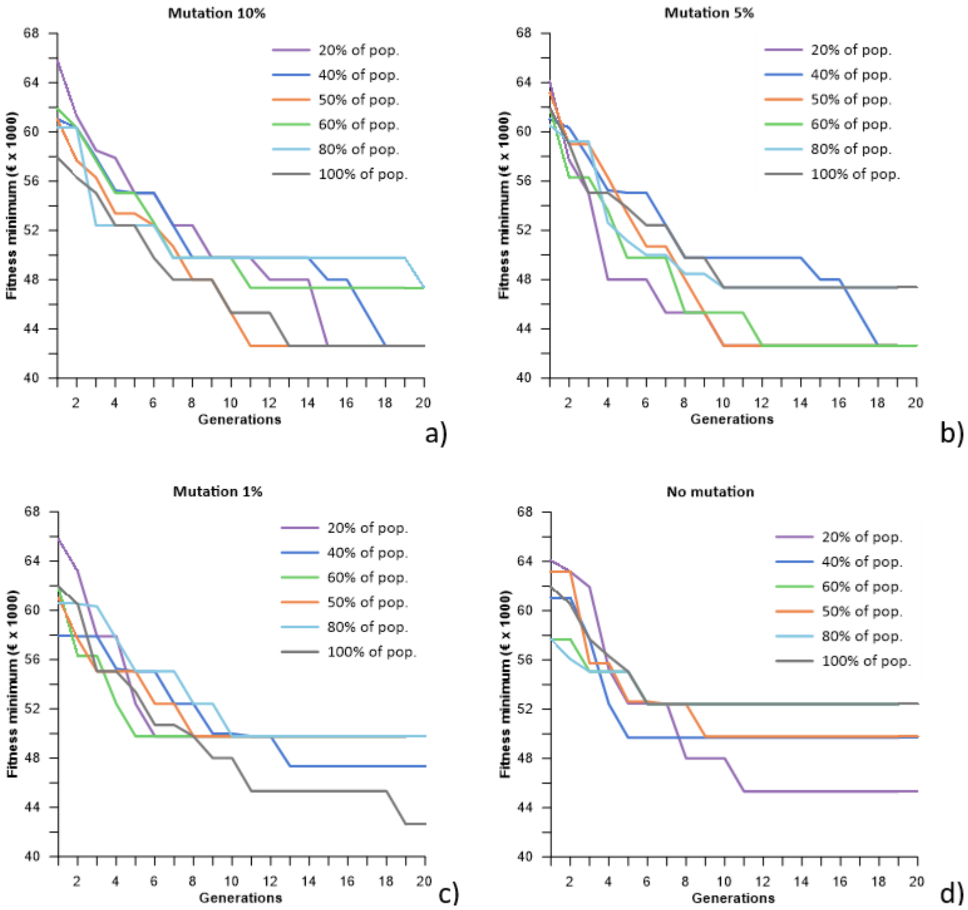



Figure 5.6: Influence on the fitness minimum of the mutation ratio - (a) $p_m = 0.1$, (b) $p_m = 0.05$, (c) $p_m = 0.01$, (d) $p_m = 0$

From the diagrams Figure 5.4, Figure 5.5, and Figure 5.6, observing the parameters that lead to a faster convergence, the following indications can be proposed for analysis with GA for structures having shear sensitive elements:

- The dimension of the population should be almost three times the number of decision variables (genes)
- The number of generation stopping criteria should be equal a quarter of the number of individual of all generations.
- High value of the mutation ratio should be set ($p_m \simeq 0.025 \div 0.05$).
- The individuals of initial population should have a high percentage of reinforced elements.

## 5.2 Study cases

The effectiveness of the method and the recommendation proposed in this thesis is tested by analysing different structural configuration of the building presented in the chapter 4.

The actual potential of the proposed optimization procedure can be well outlined by considering the preliminary example applications reported in this chapter.

In particular the infills were arranged into different configuration to modify the behaviour of the case study structure. In particular the following structure were analysed:

1. a bare frame

2. an infilled frame

3. a structure that is characterized by a soft story mechanism by the definition of the infills only from the first story

4. a structure that has infills only on one side

The third configuration is typical of structures with unobstructed commercial spaces on the ground floor, the last one is characteristic of the buildings in line.

Before starting with the optimization process of the retrofitting, the seismic performance of each structural configuration has been tested without any retrofit and with all the columns subject of analysis retrofitted.

This is first done to get a reference point about the safety of the structure as built. Secondly, the test of a number of trial retrofit configurations allows comparing cost/performance results with those of the solution found through the optimization framework solution.

As well explained in Chapter 4.3 a single (one direction) pushover analysis is carried out for each configuration in order to reduce the computational effort to obtain the capacity/demand ratio $\xi_{\mu}$.

The following test results are illustrated in term of total base shear or column base shear againt top displacement of the structure.

In tridimensional representations of the structures the unretrofitted columns are depicted white, columns retrofitted to increase the ductility capacity are portrayed light red.

The columns that have a shear vulnerability so that they need a shear strengthening are drawn dark red instead the colums that are vulnerable to shear collapse inducted by infills interaction are depicted purple.

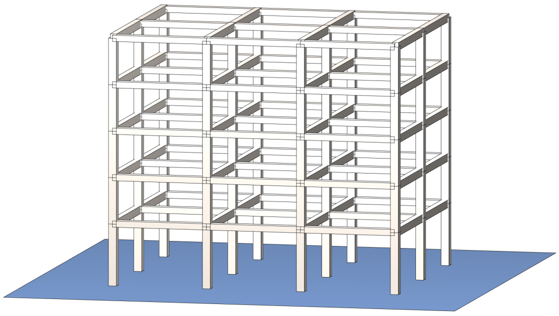
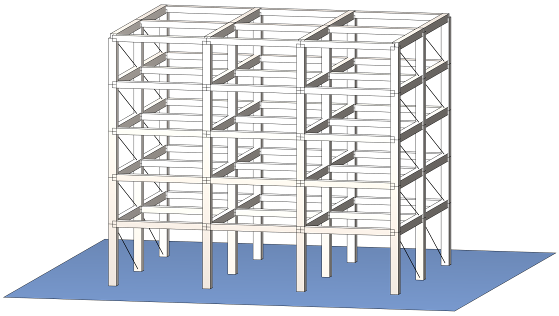
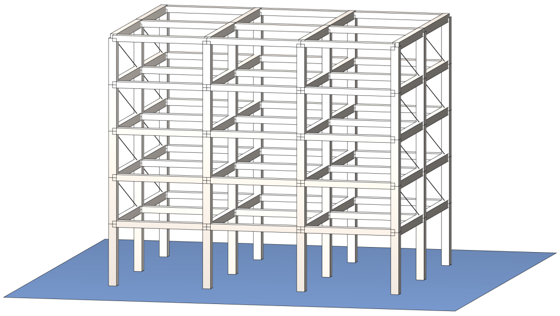
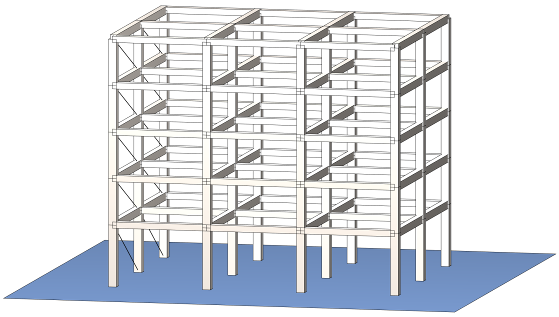
| | | |
|---|---|---|
| **Bare frame** | Section 5.2.1<br>pag. 72 |  |
| **Infilled frame** | Section 5.2.2<br>pag. 85 |  |
| ***Soft-story* structure** | Section 5.2.3<br>pag. 96 |  |
| ***Eccentric* frame** | Section 5.2.4<br>pag. 112 |  |

Table 5.1: Study cases

### 5.2.1 Bare frame

This first structure is analysed to evaluate the effectiveness of the genetic algorithm presented in the previous chapters for a bare frame, a structure without any kind of element except beams and columns without the presence of infills (Figure 5.7).

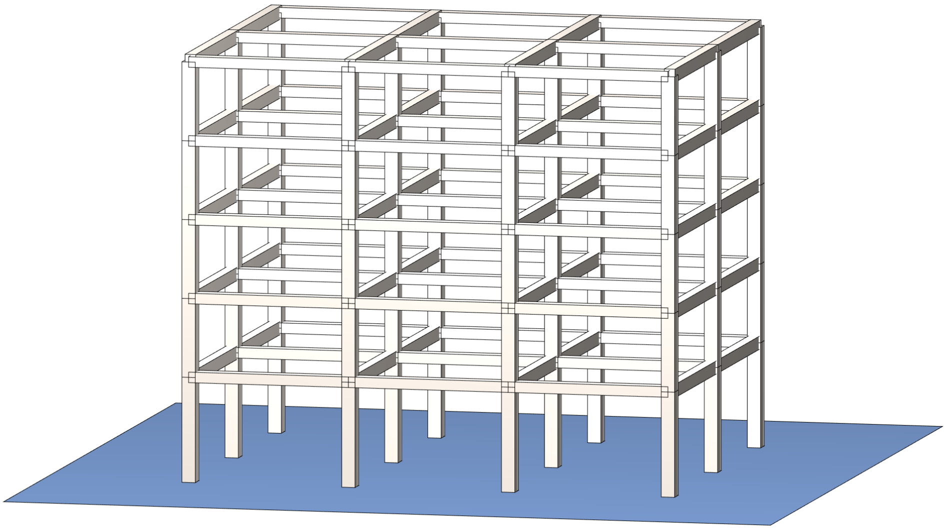


Figure 5.7: Structural configuration of the bare frame structure

For this structure two preliminary test where performed, the first one the analysis of the structure without any retrofit into two differente case with and without the shear verification (as presented in Chapter 4.4). The second preliminary test was performed for the structure with all the columns on the first and second floor retrofitted. This preliminary tests were performed to get a reference point about the safety of the structure as built and for the most expensive solution.

#### Preliminary tests

In Figure 5.9 and 5.12 are illustrated the pushover capacity curve obtained for the bare frame without performing the shear verification respectively for analysis along the Z and X direction.

Comparing to the pushover capacity curve obtained for analysis that consider shear verification (Figure 5.10 and Figure 5.13), the shear collapse of columns are easy to underline and locate in the columns 5, 6, 7 and 8.

The results of these preliminary test are schematically reported in Table 5.2.

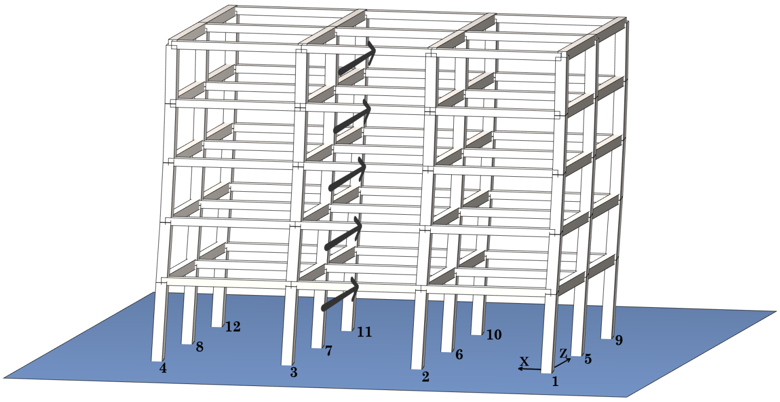



Figure 5.8: Bare frame - Preliminary test 1: Deformed shape (pushover along Z)

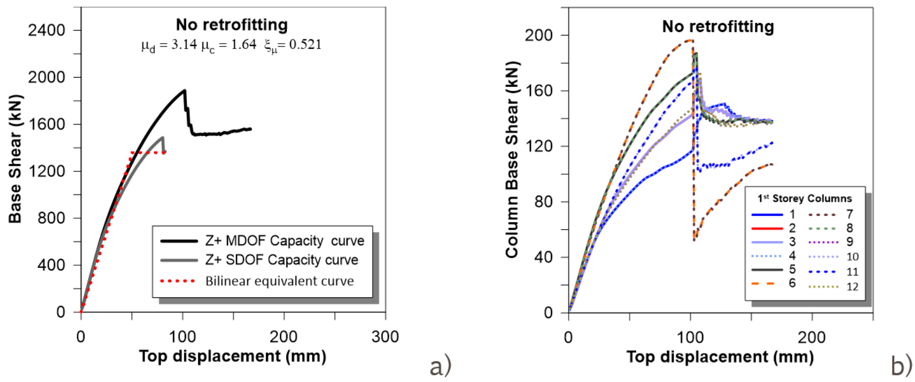



Figure 5.9: Bare frame - Preliminary test 1 - without shear verification: (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

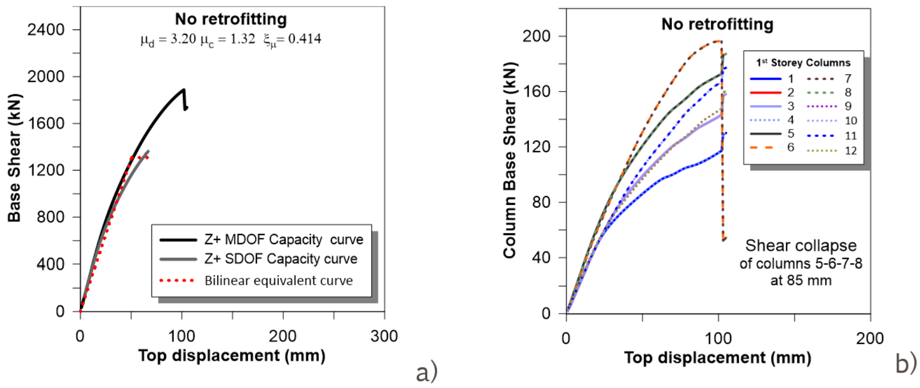



Figure 5.10: Bare frame - Preliminary test 1 - with shear verification: (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

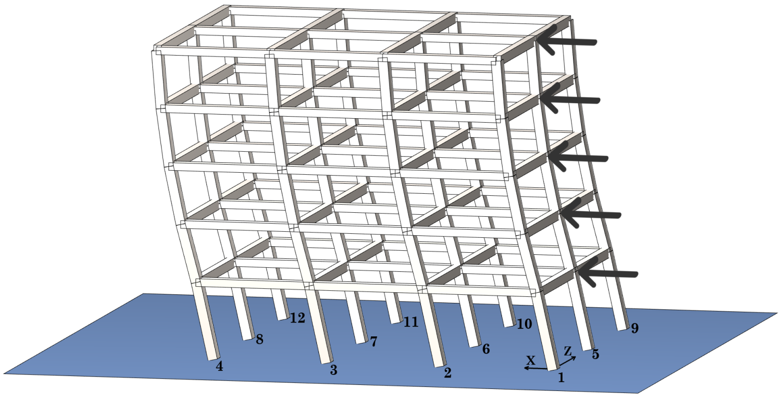



Figure 5.11: Bare frame - Preliminary test 1: Deformed shape (pushover along X)

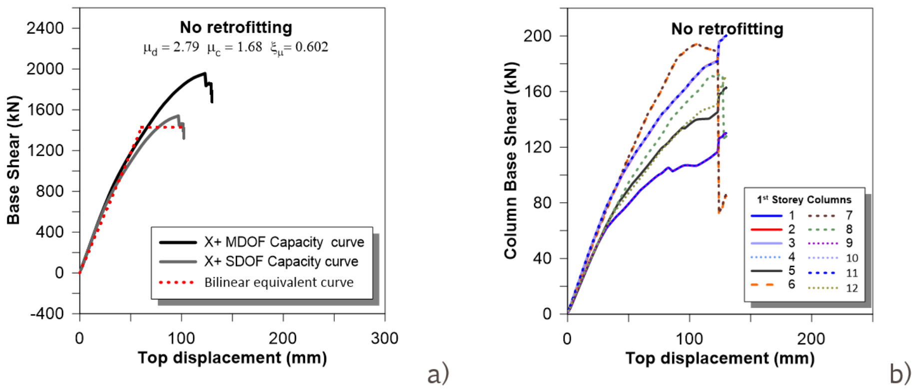



Figure 5.12: Bare frame - Preliminary test 1 - without shear verification: (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

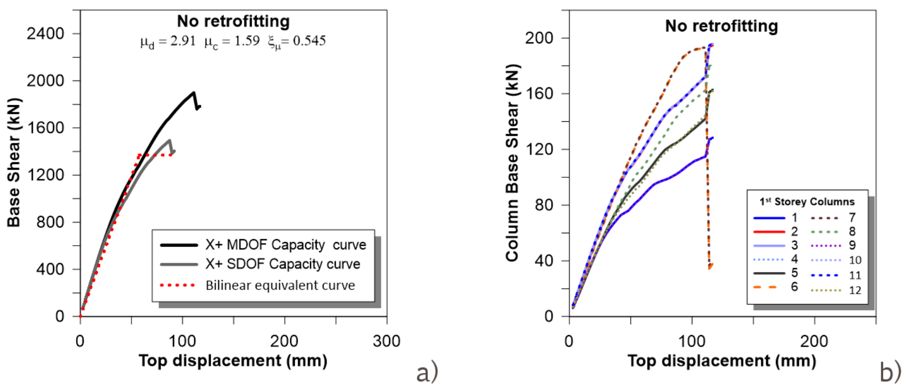



Figure 5.13: Bare frame - Preliminary test 1 - with shear verification: (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

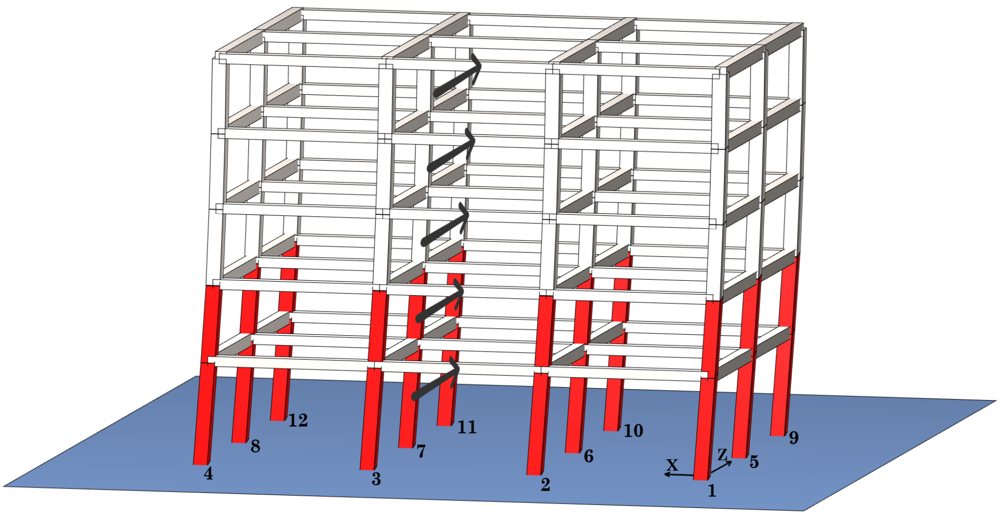
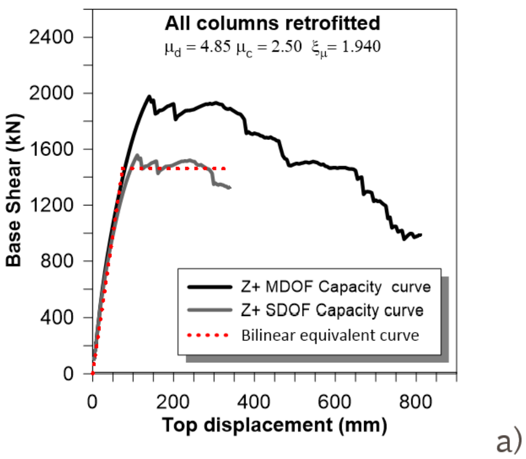



Figure 5.14: Bare frame - Preliminary test 2: Deformed shape (pushover along Z)



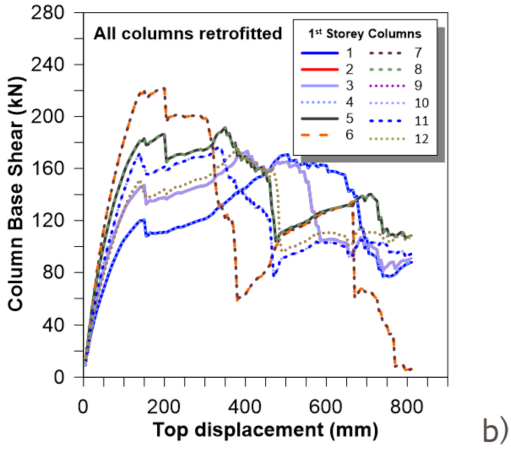




Figure 5.15: Bare frame - Preliminary test 2: (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

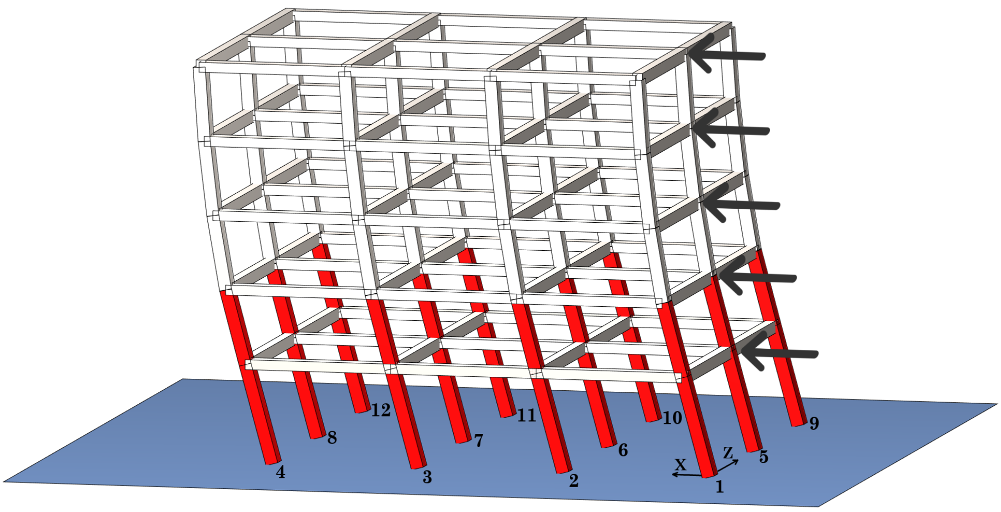



Figure 5.16: Bare frame - Preliminary test 2: Deformed shape (pushover along X)

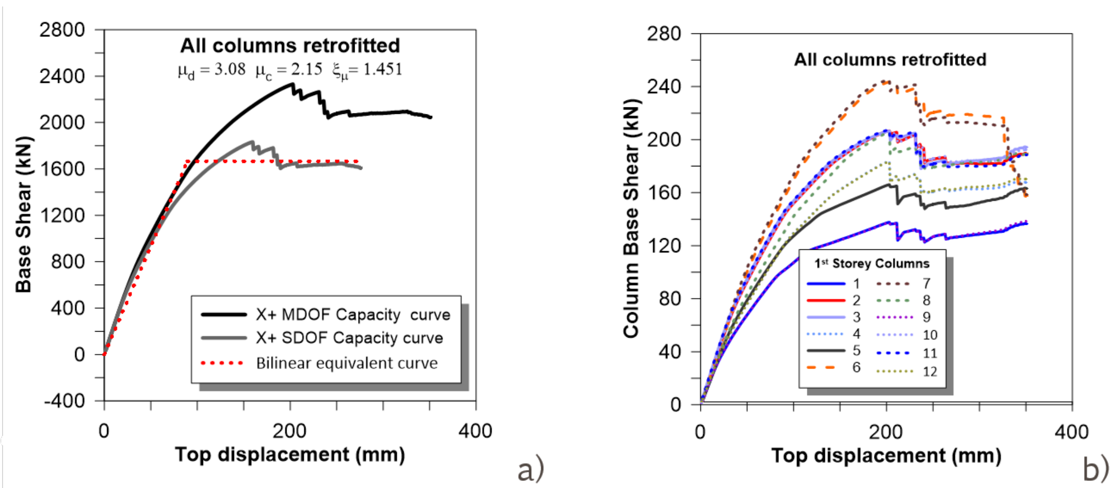



Figure 5.17: Bare frame - Preliminary test 2: (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

### Optimization results

For this structural configuration two different optimization process are performed, in the first one the shear verification of elements was disabled, whereas in the second was enabled.

In the so defined structural configuration, infills significantly reduce interstorey drift demand of the surrounding frames with respect to that of the first floor.

In Figure 5.18 and Figure 5.19 are illustrated the genetic algorithm process trend, in particular are reported the minumum and average value of the individuals of each generation analysed and the stall defined as the number of generation that the best solution does not improve.

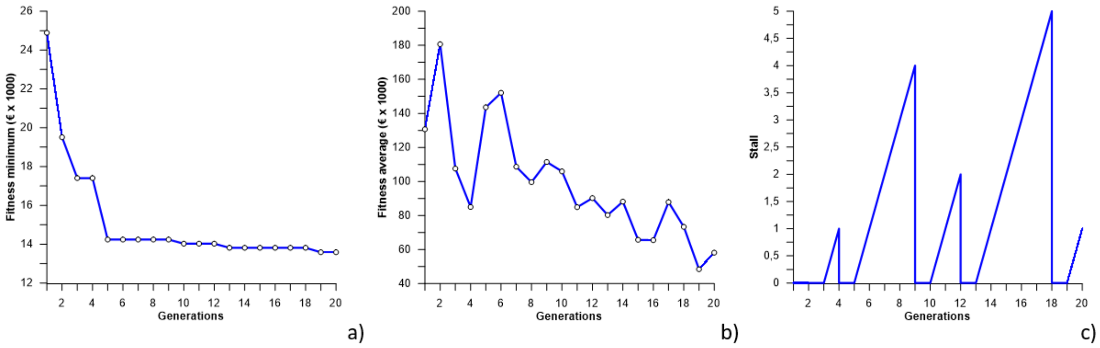



Figure 5.18: Genetic algorithm process parameters - Bare frame(without shear verification): (a) best solution's fitness value each generation (b) average fitness values for each generation (c) stall trend

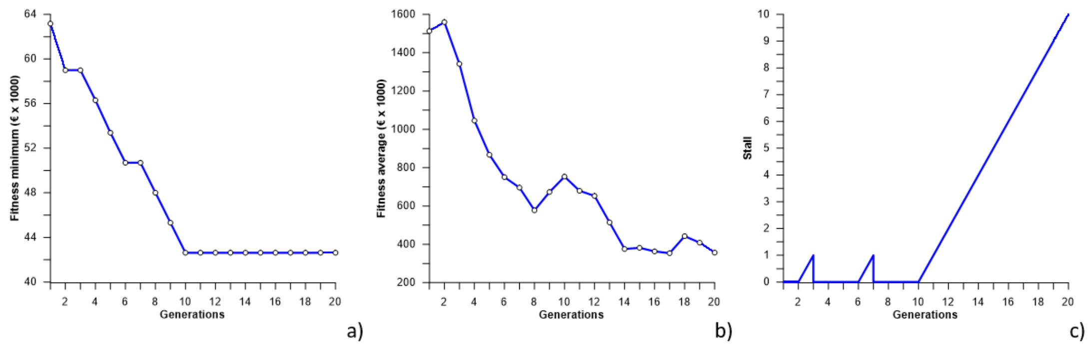



Figure 5.19: Genetic algorithm process parameters - Bare fram (with shear verification): (a) best solution's fitness value each generation (b) average fitness values for each generation (c) stall trend

The results of the two analysis are illustrated in Figure 5.20 and Figure 5.24

together with the respective pushover capacity curves.

As mentioned in the previous Chapter 4, the optimal solution found refers to a pushover force profile acting along Z positive direction. In this case, given the symmetry of the structure in plan and elevation, it can be simply is supposed to retrofit in the same way ($s_b = 150\,\text{mm}$) column 10 and 11 in order to face seismic demand along Z negative direction.

The analyses of this so defined final retrofitting configuration are reported in Figure 5.28 and Figure 5.30 associated with the respective capacity curves.

| Test | $s_b$ | $n_C$ | $C$ | direct. | $\mu_d$ | $\mu_c$ | $\xi_\mu$ | Ver. check |
|---|---|---|---|---|---|---|---|---|
| 1 (without shear ver.) | - | - | 0 € | Z | 3.14 | 1.64 | 0.521 | NO |
| | | | | X | 2.79 | 1.68 | 0.602 | NO |
| 1 (with shear ver.) | - | - | 0 € | Z | 3.20 | 1.32 | 0.414 | NO |
| | | | | X | 2.91 | 1.59 | 0.545 | NO |
| 2 | 150 | 24 | 69 618 € | Z | 4.85 | 2.5 | 1.940 | YES |
| | | | | X | 3.08 | 2.15 | 1.451 | YES |

Table 5.2: Bare frame - Results of preliminary tests

| Test | $s_b$ | $n_C$ | $C$ | direct. | $\mu_d$ | $\mu_c$ | $\xi_\mu$ | Ver. check |
|---|---|---|---|---|---|---|---|---|
| Without shear ver. | 150 | 5 | 15 122 € | Z | 2.83 | 2.77 | 1.023 | YES |
| | | | | X | 2.56 | 2.55 | 1.001 | YES |
| With shear ver. | 150 | 14 | 41 352 € | Z | 4.97 | 2.64 | 1.86 | YES |
| | | | | X | 2.12 | 2.23 | 0.95 | NO |
| Symm. arrangement | 150 | 16 | 47 401 € | Z | 2.54 | 4.81 | 1.891 | YES |
| | | | | X | 2.12 | 2.99 | 1.405 | YES |

Table 5.3: Bare frame - Results of optimization

The capacity demand ratio finally obtained is $\xi_\mu = 1.891$, while the overall cost of the intervention is 47 401.70 €. It is noteworthy observing that the obtained cost is reduced by 32% with respect to the best solution found with preliminary tests (Preliminary test 2 - Chapter 5.2.1). However, in the face of this, the $\xi_\mu$ factor finally obtained for the analysis performed along Z (1.891) differs only by 2.5% with respect to that obtained in preliminary test 2 (1.941) with a retrofitting cost of 69 618.9 €.

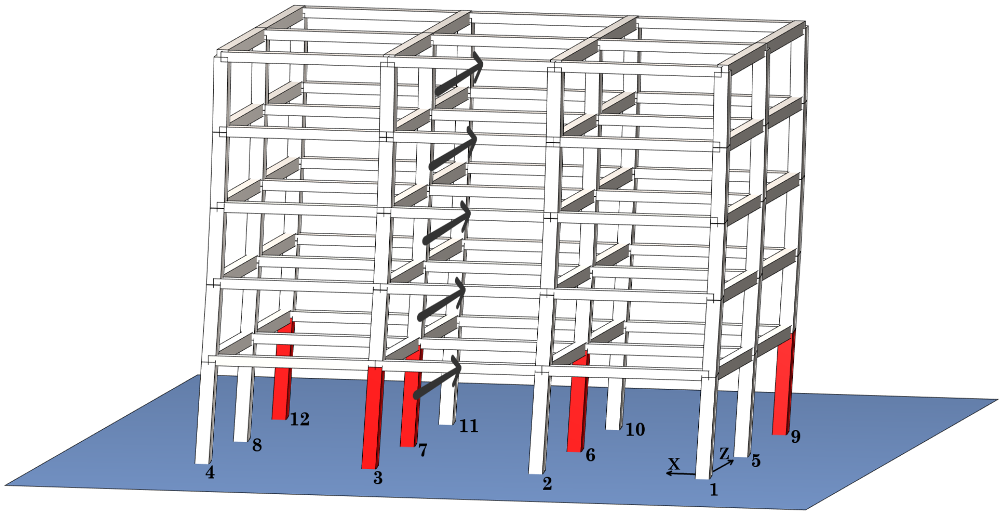



Figure 5.20: Bare frame - Optimal configuration (without shear verification): Deformed shape (pushover along Z)

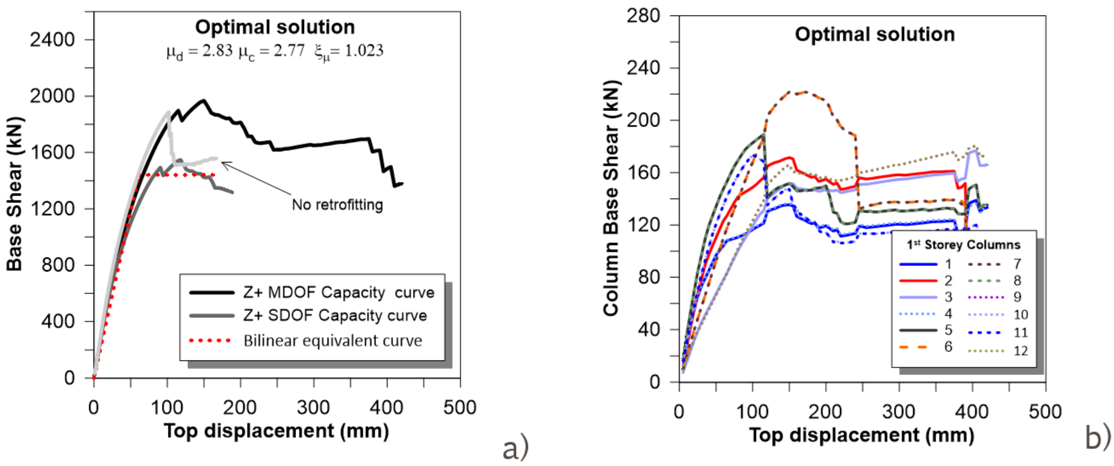



Figure 5.21: Bare frame - Optimal configuration (without shear verification): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

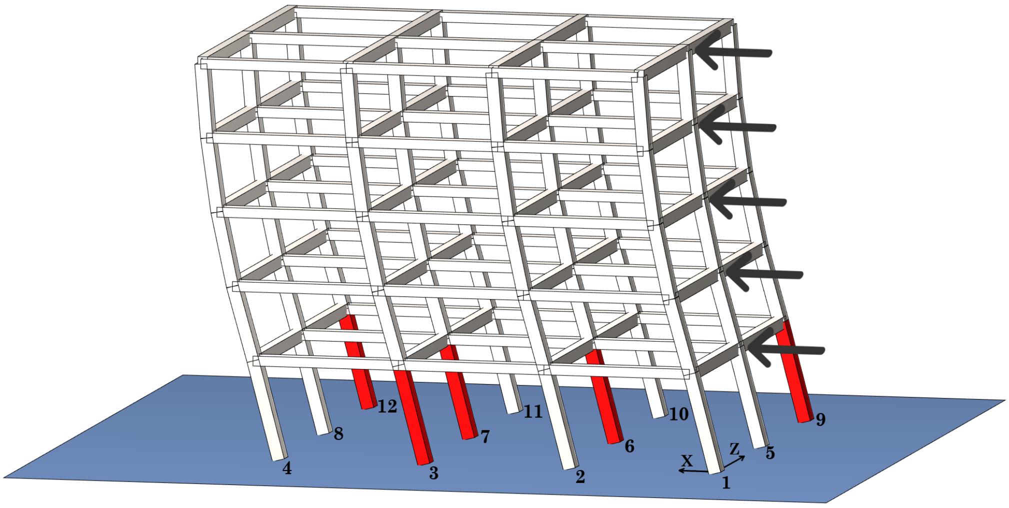



Figure 5.22: Bare frame - Optimal configuration (without shear verification): Deformed shape (pushover along X)

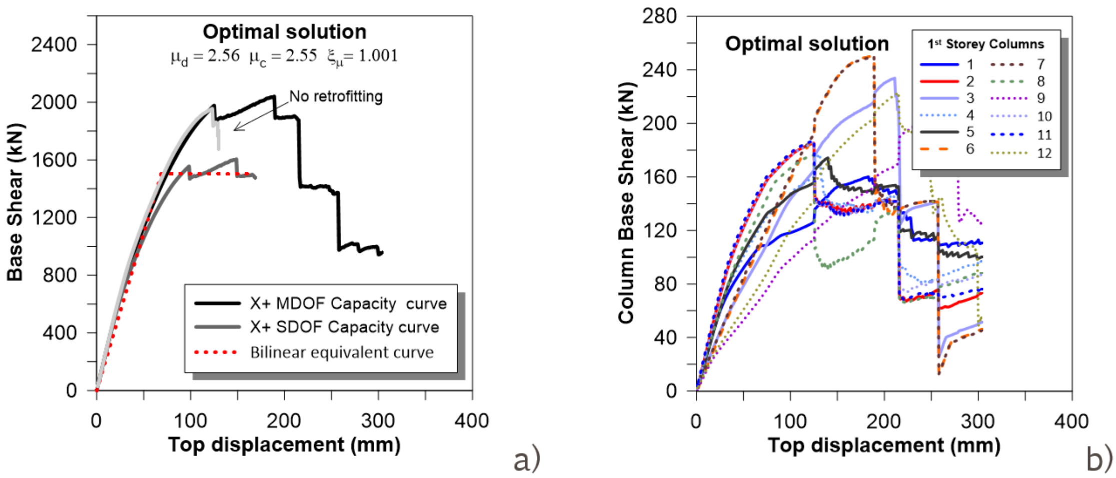



Figure 5.23: Bare frame - Optimal configuration (without shear verification): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

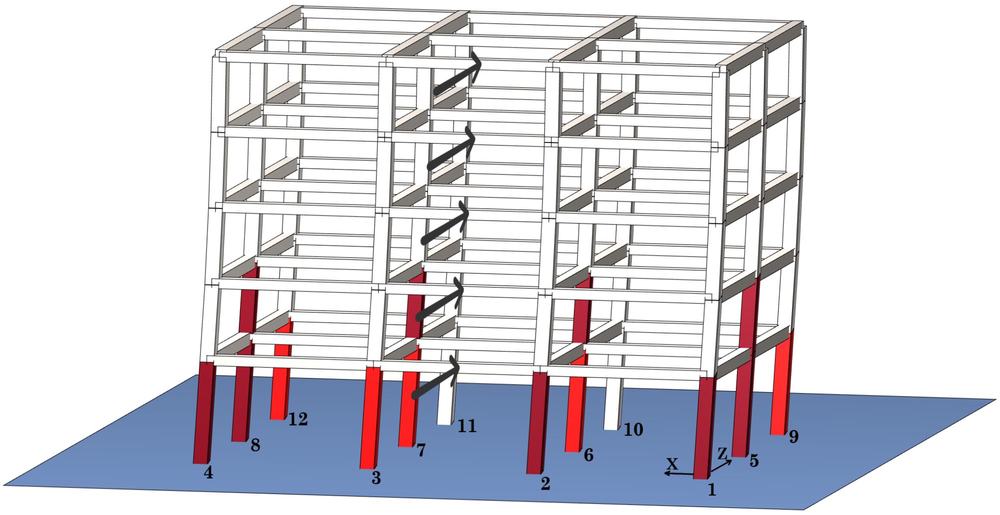



Figure 5.24: Bare frame - Optimal configuration (with shear verification): Deformed shape (pushover along Z)

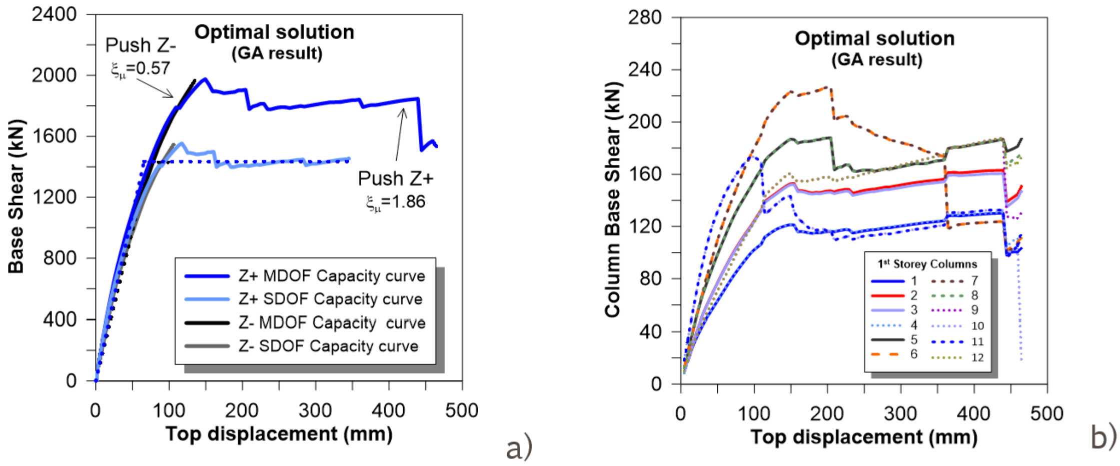



Figure 5.25: Bare frame - Optimal configuration (with shear verification): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

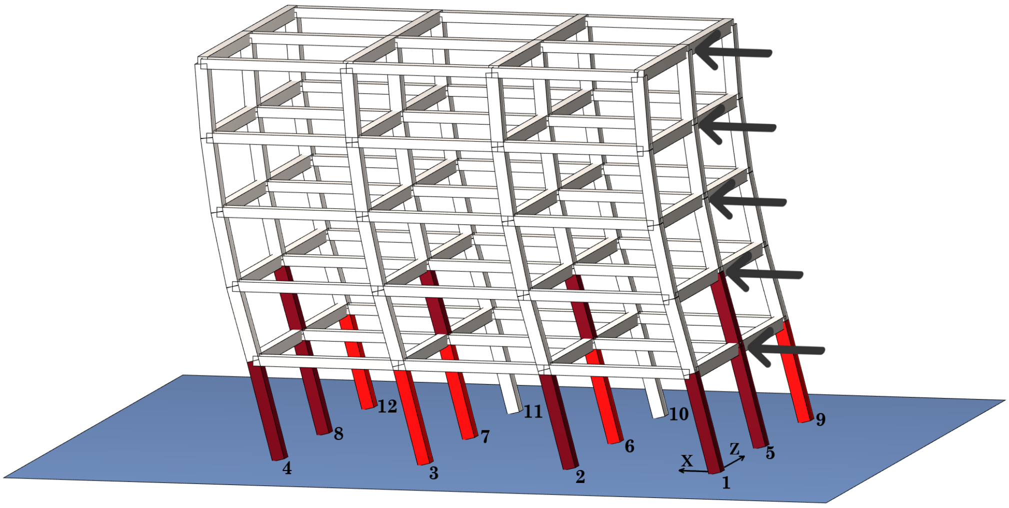



Figure 5.26: Bare frame - Optimal configuration (with shear verification): Deformed shape (pushover along X)

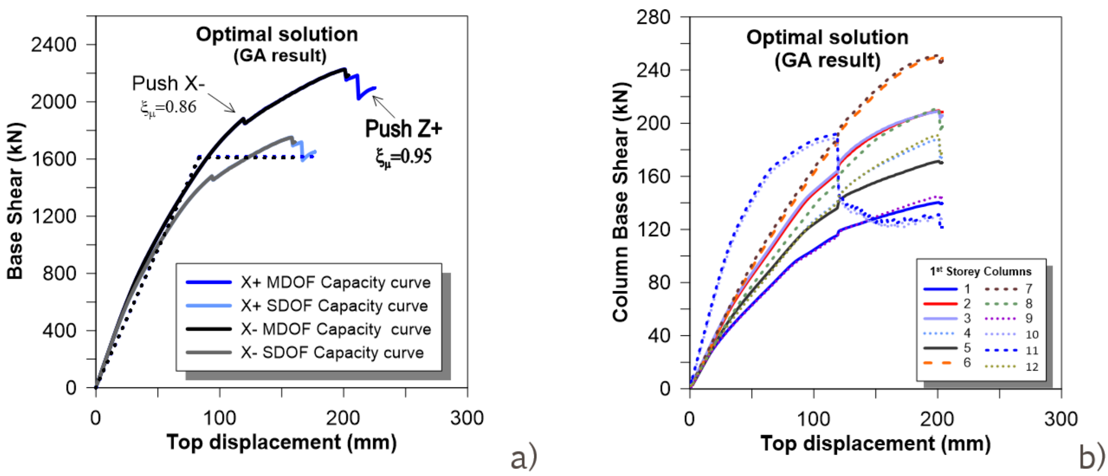



Figure 5.27: Bare frame - Optimal configuration (with shear verification): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

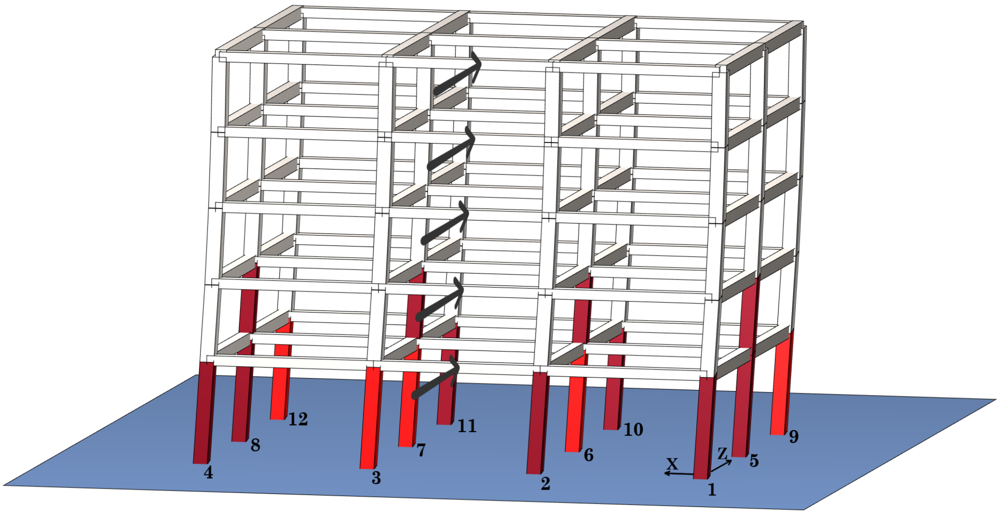



Figure 5.28: Bare frame - Optimal configuration (symmetric arrangement): Deformed shape (pushover along Z)

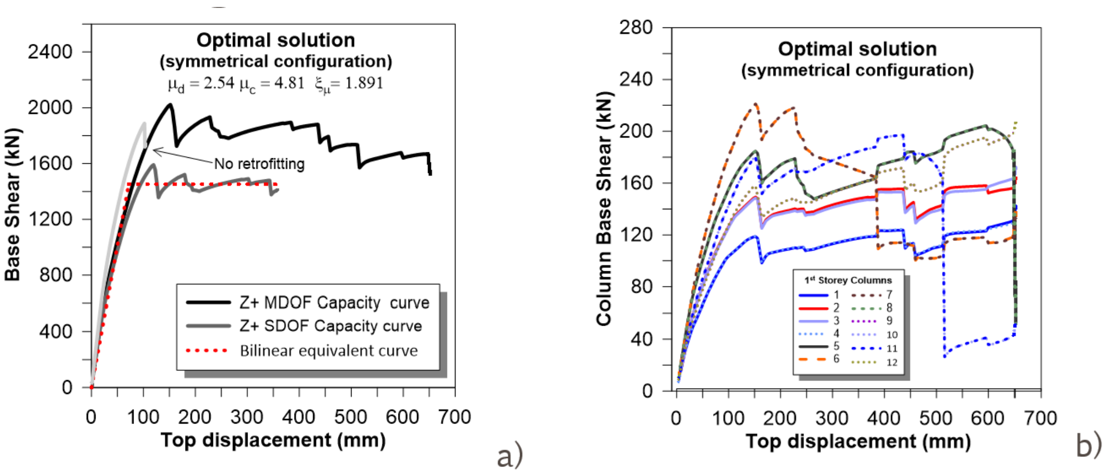



Figure 5.29: Bare frame - Optimal configuration (symmetric arrangement): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

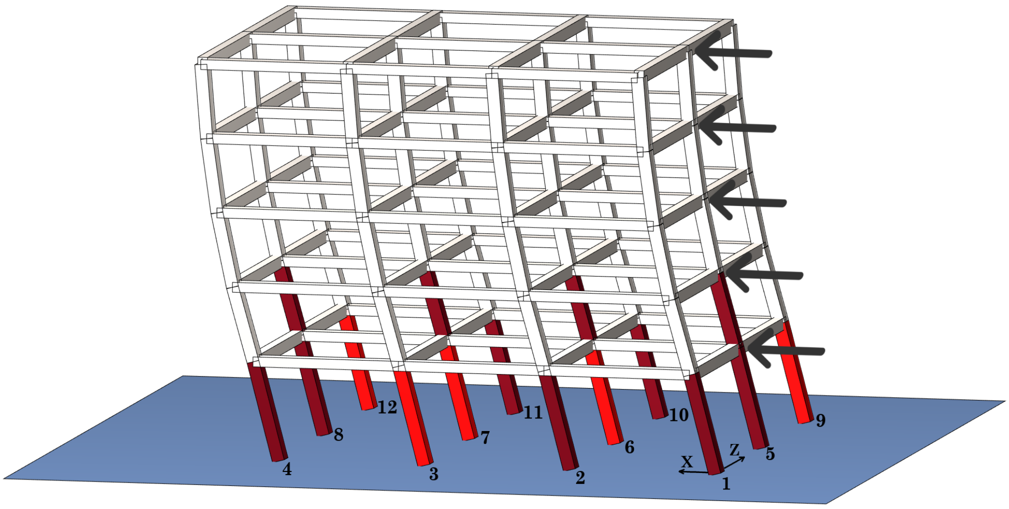



Figure 5.30: Bare frame - Optimal configuration (symmetric arrangement): Deformed shape (pushover along X)

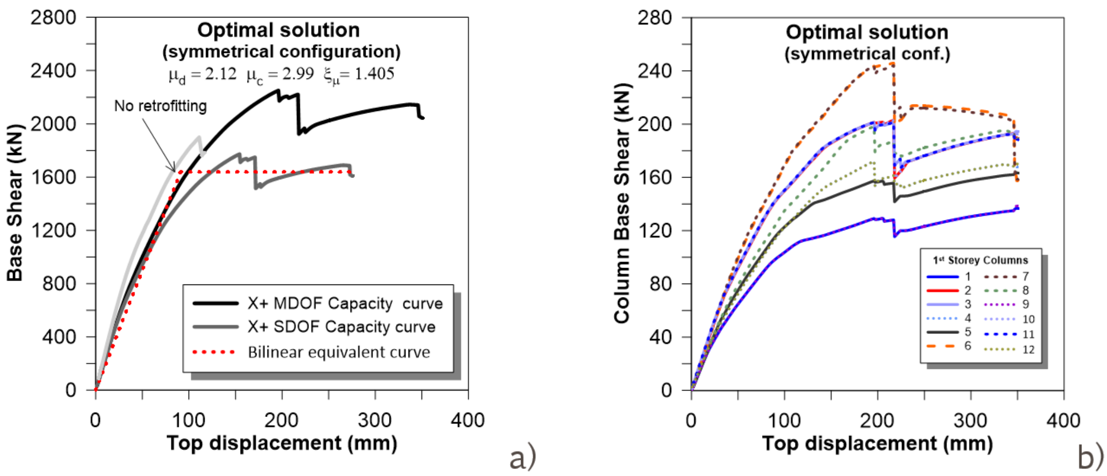



Figure 5.31: Bare frame - Optimal configuration (symmetric arrangement): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

### 5.2.2 Infilled frame

The second case study structure is an infilled frame. It consists of a frame with the presence of infills defined as defined in Chapter 4.2.3 into the two external frame of the structure(Figure 5.7).

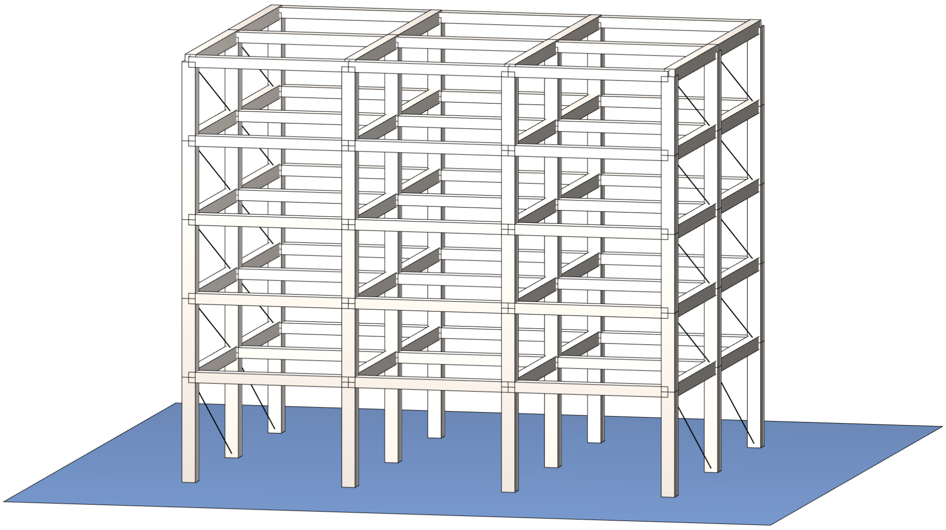


Figure 5.32: Structural configuration of the infilled frame structure

As previously, two preliminary test are performed to verify the structure *as-built* and to verify the structure with all the columns at the first and second floor retrofitted.

#### Preliminary tests

In Figure 5.34 and Figure 5.36 are illustrated the pushover capacity curves for the *as-built* stucture respectively for forces acting along Z and along X.

In Figure 5.38 and Figure 5.40 are reported the pushover results for the structure with all the columns of the first and second floor retrofitted.

The results of these preliminary test are schematically reported in Table 5.4.

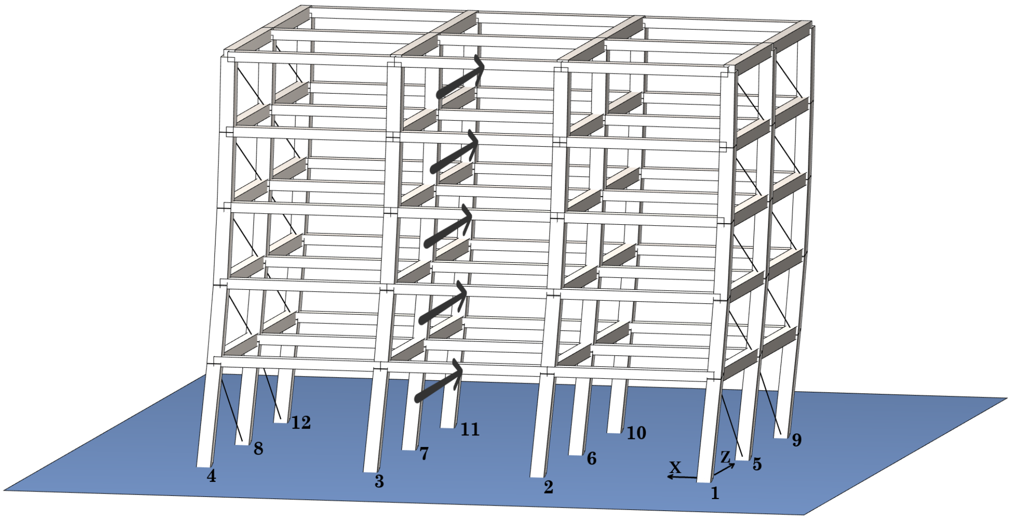
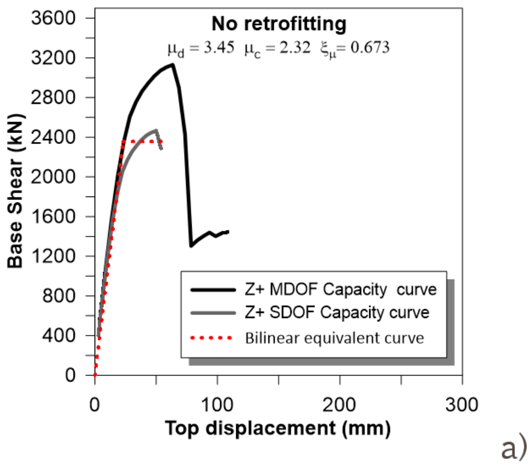



Figure 5.33: Infilled frame - Preliminary test 1: Deformed shape (pushover along Z)



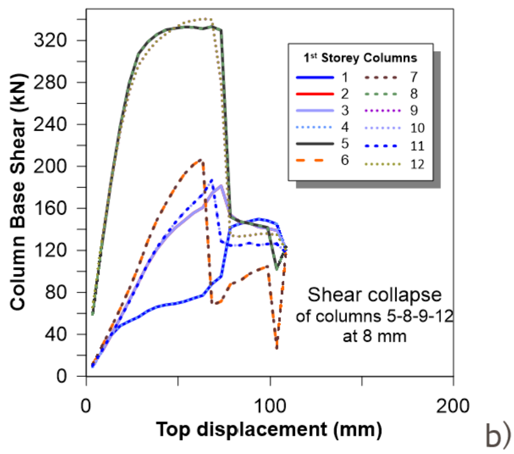




Figure 5.34: Infilled frame - Preliminary test 1: (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

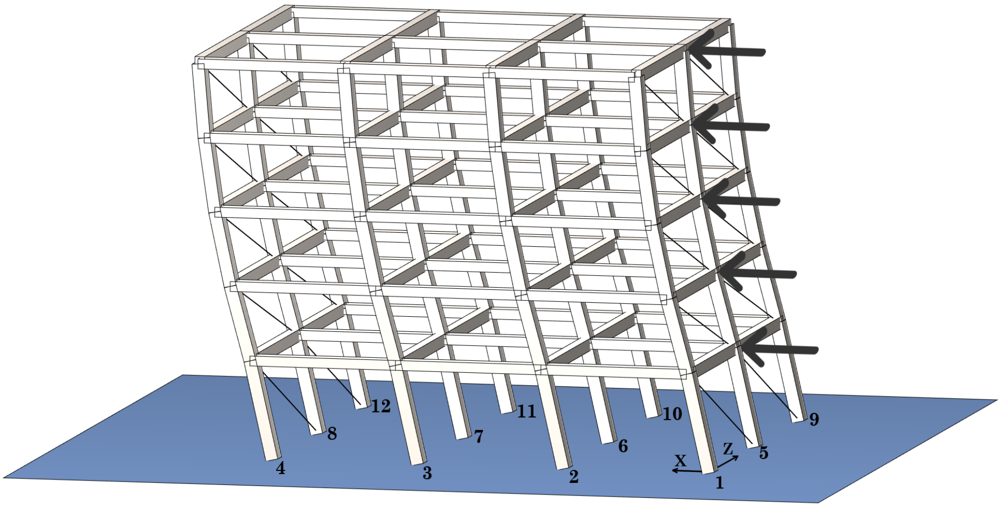



Figure 5.35: Infilled frame - Preliminary test 1: Deformed shape (pushover along X)

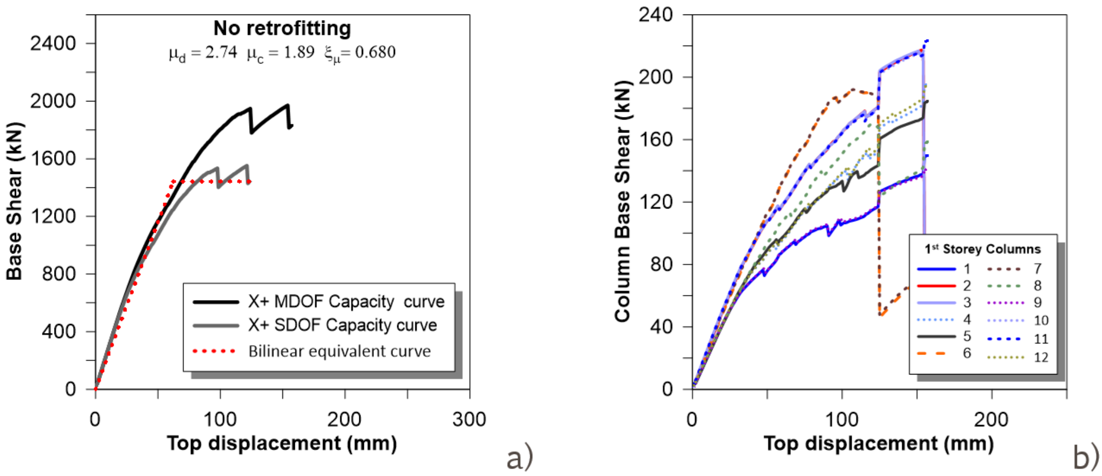



Figure 5.36: Infilled frame - Preliminary test 1: (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

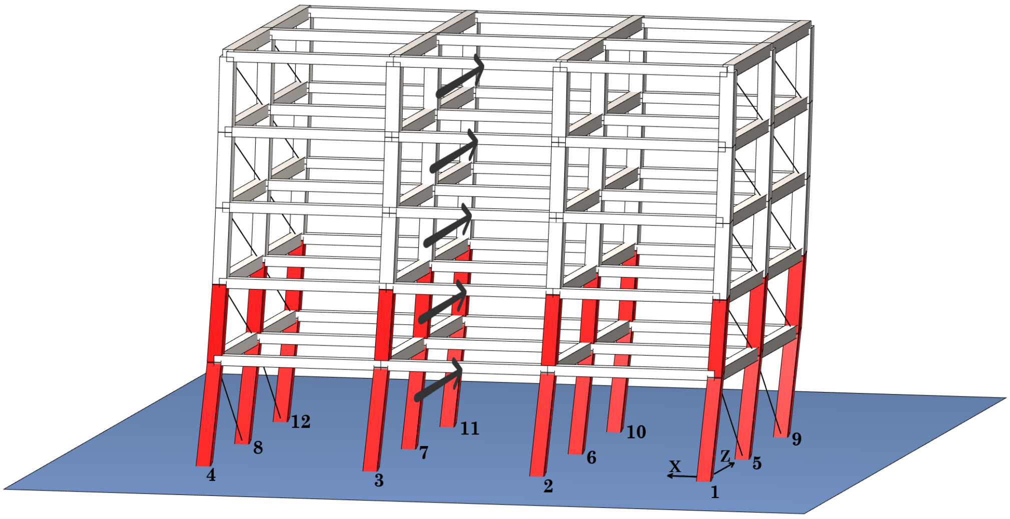



Figure 5.37: Infilled frame - Preliminary test 2: Deformed shape (pushover along Z)

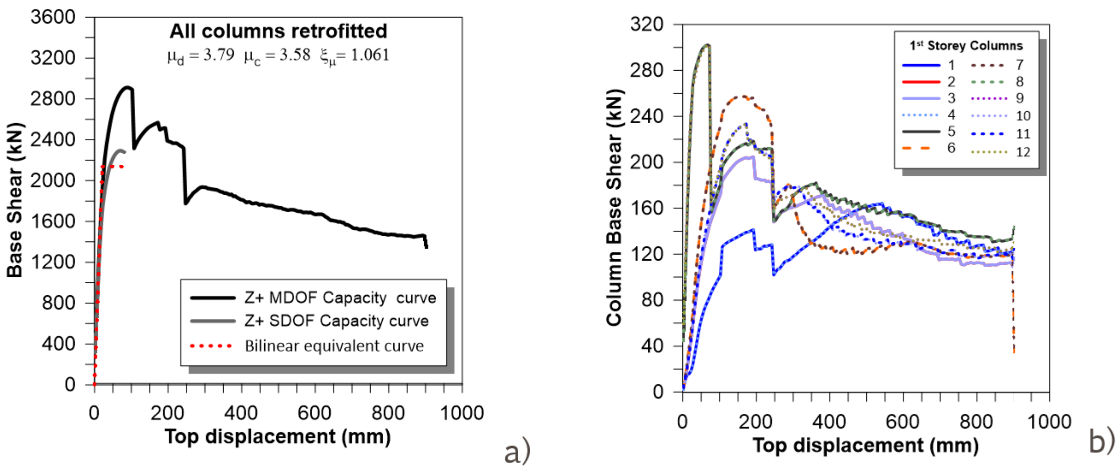



Figure 5.38: Infilled frame - Preliminary test 2: (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

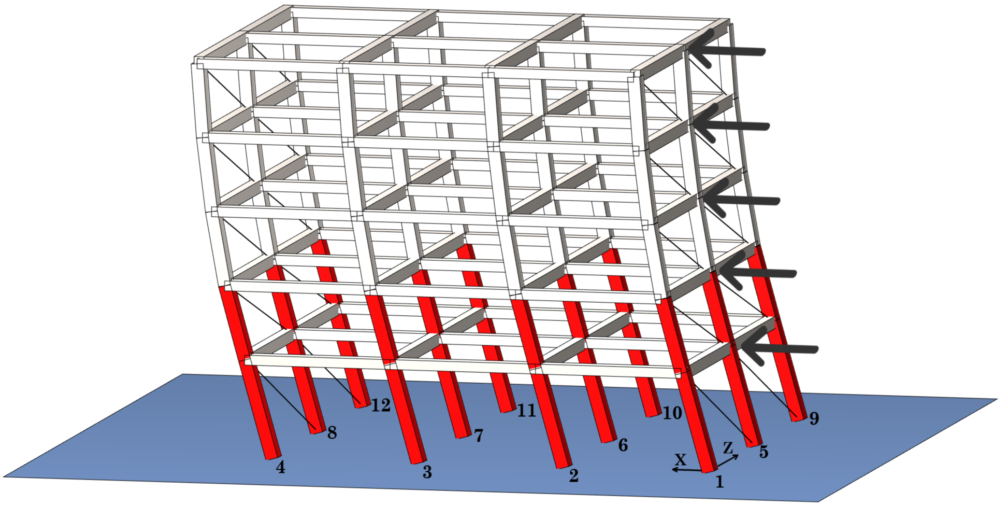



Figure 5.39: Infilled frame - Preliminary test 2: Deformed shape (pushover along X)

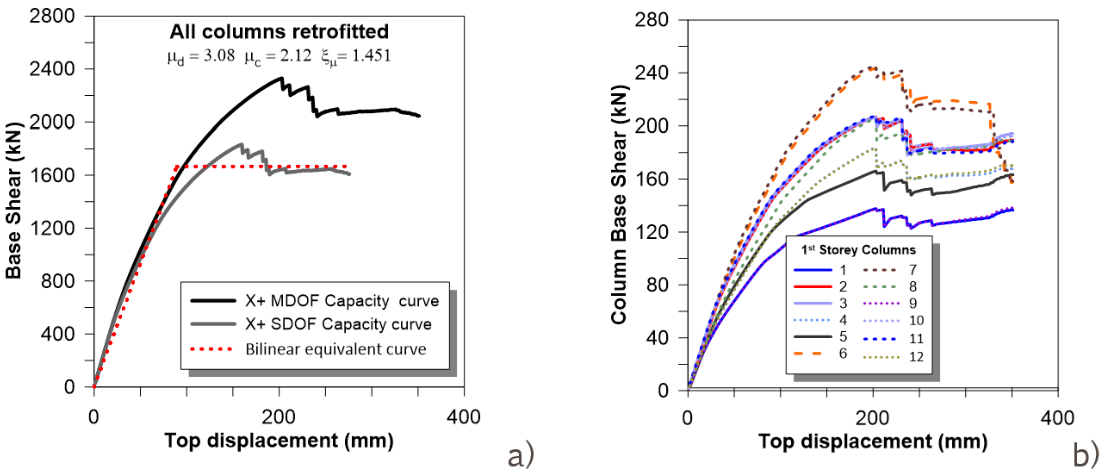



Figure 5.40: Infilled frame - Preliminary test 2: (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

### Optimization results

For this structural configuration three different optimization process are performed, in the first one the shear verification of elements are disabled, in the second shear verification of elements are performed, in the last one the additional shear demand inducted by the infills is calculated (as presented in Chapter 4.4.2).

In Figure 5.41, Figure 5.42, and Figure 5.43 are illustrated the genetic algorithm process trend, in particular are reported the minumum and average value of the individuals of each generation analysed and the stall.

The algorithm output for the first two analysis is exactly coincident. The results of the two analysis are illustrated in Figure 5.44, and Figure 5.48 together with the respective pushover capacity curves.

As explained at pag.77 the final result has to be analysed to verify that the retrofitting arrangement is suitable to react to forces acting on different direction.

In this cases it is not necessary because the algorithm output is symmetric along Z yet.

| Test | $s_b$ | $n_C$ | $C$ | direct. | $\mu_d$ | $\mu_c$ | $\xi_\mu$ | Ver. check |
|---|---|---|---|---|---|---|---|---|
| 1 | - | - | 0€ | Z | 3.45 | 2.32 | 0.673 | NO |
| | | | | X | 2.74 | 1.89 | 0.680 | NO |
| 2 | 150 | 24 | 69 618€ | Z | 3.79 | 3.58 | 1.061 | YES |
| | | | | X | 3.08 | 2.12 | 1.451 | YES |

Table 5.4: Infilled frame - Results of preliminary tests

| Test | $s_b$ | $n_C$ | $C$ | direct. | $\mu_d$ | $\mu_c$ | $\xi_\mu$ | Ver. check |
|---|---|---|---|---|---|---|---|---|
| Without shear ver. | 150 | 12 | 39 070€ | Z | 2.83 | 2.77 | 1.023 | YES |
| | | | | X | 2.56 | 2.55 | 1.001 | YES |
| Infills shear contribution | 150 | 18 | 52 956€ | Z | 3.52 | 3.61 | 1.028 | YES |
| | | | | X | 2.07 | 2.07 | 1.001 | YES |

Table 5.5: Infilled frame - Results of optimization

The capacity demand ratio finally obtained is $\xi_\mu = 1.023$ for the analysis along Z, while the overall cost of the intervention is 52 956.00€. It is noteworthy observing that the obtained cost is reduced by 24% with respect to the best solution found with preliminary tests (Preliminary test 2 - Chapter 5.2.2). However, in the face of this, the $\xi_\mu$ factor finally obtained for the analysis performed along Z (1.023) differs only by 4% with respect to that obtained in preliminary test 2 (1.061) with a retrofitting cost of 69 618.9€.

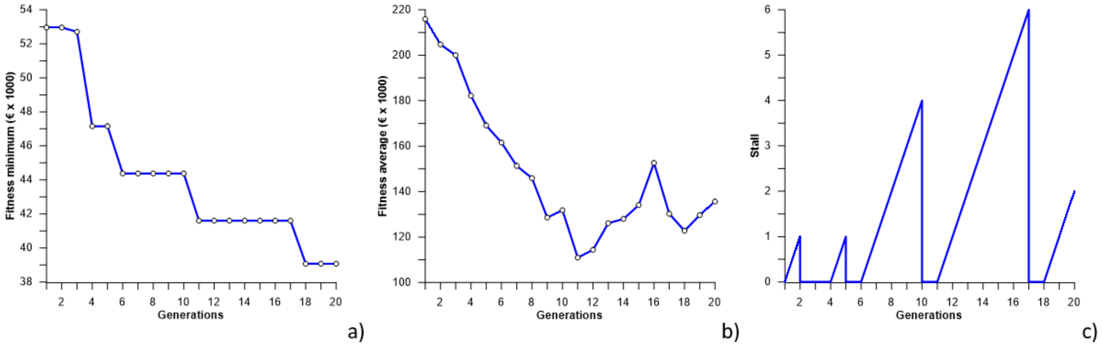



Figure 5.41: Genetic algorithm process parameters - Infilled frame (without shear verification): (a) best solution's fitness value each generation (b) average fitness values for each generation (c) stall trend

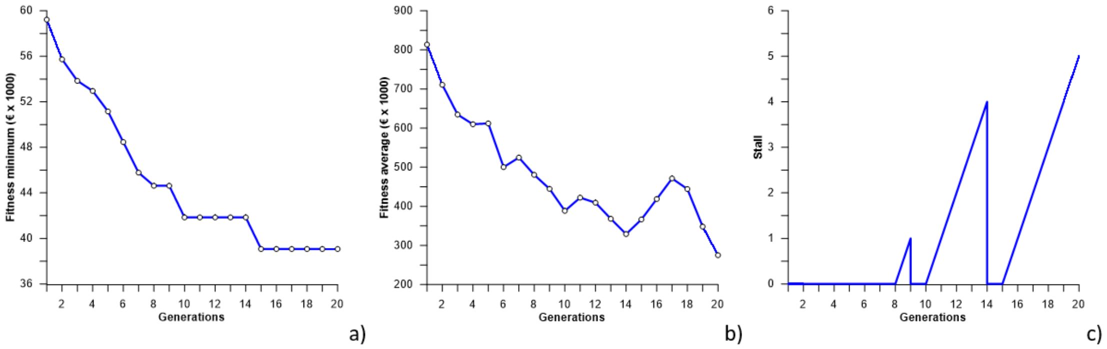



Figure 5.42: Genetic algorithm process parameters - Infilled frame (with shear verification): (a) best solution's fitness value each generation (b) average fitness values for each generation (c) stall trend

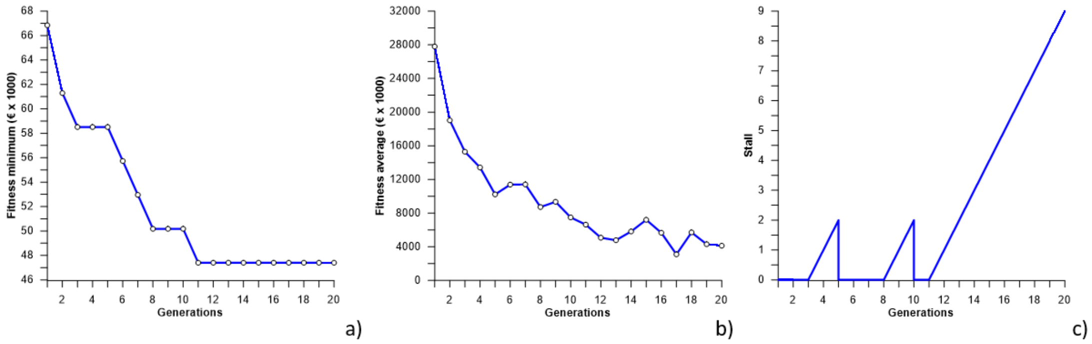



Figure 5.43: Genetic algorithm process parameters - Infilled frame (with infills shear contribution): (a) best solution's fitness value each generation (b) average fitness values for each generation (c) stall trend

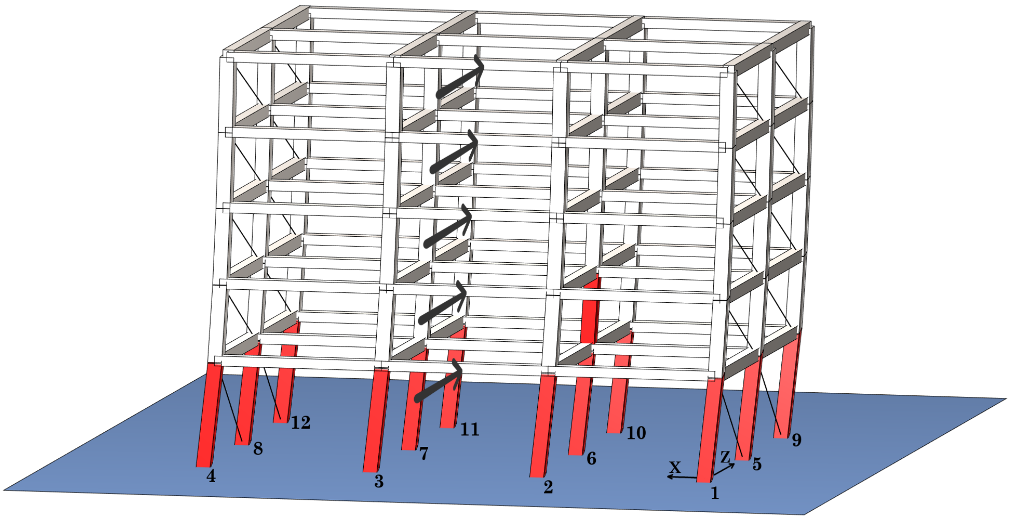
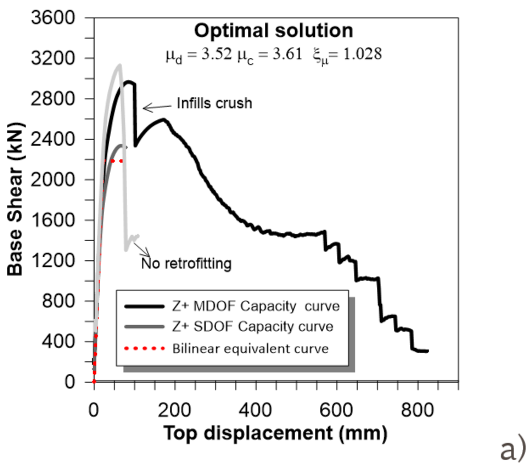



Figure 5.44: Infilled frame - Optimal configuration (without shear verification): Deformed shape (pushover along Z)



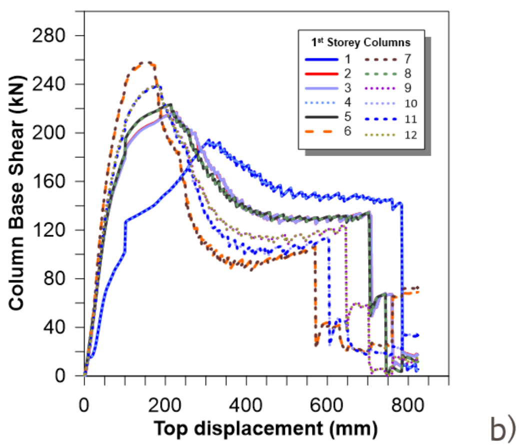




Figure 5.45: Infilled frame - Optimal configuration (without shear verification): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

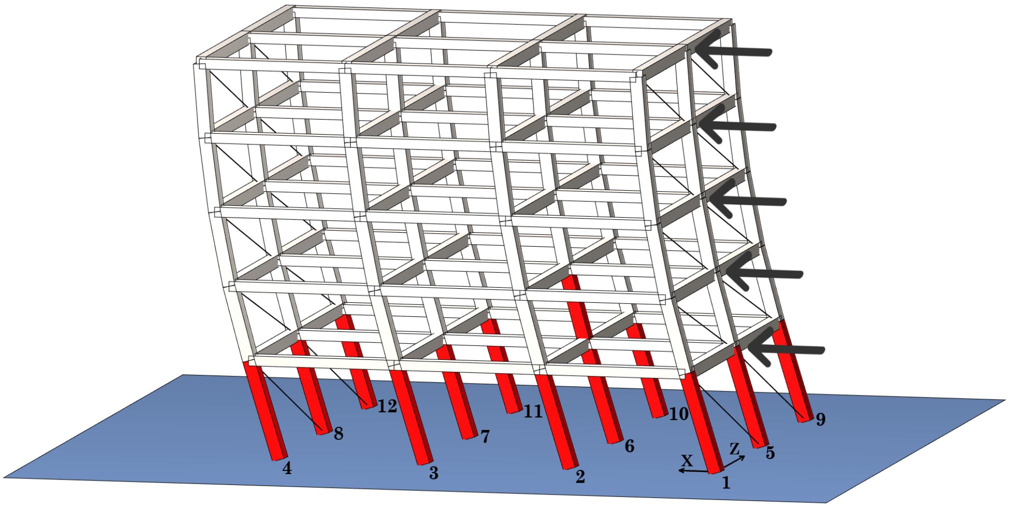



Figure 5.46: Infilled frame - Optimal configuration: Deformed shape (pushover along X)

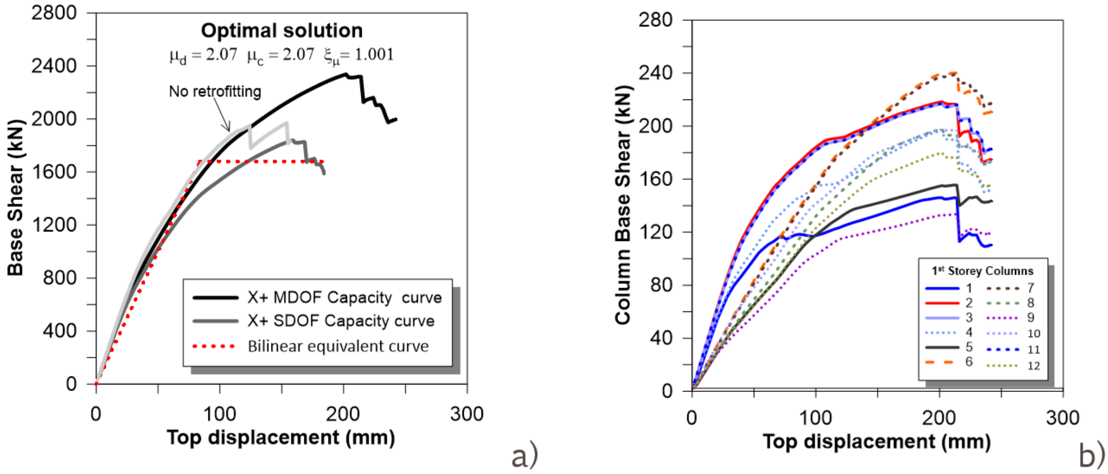



Figure 5.47: Infilled frame - Optimal configuration: (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

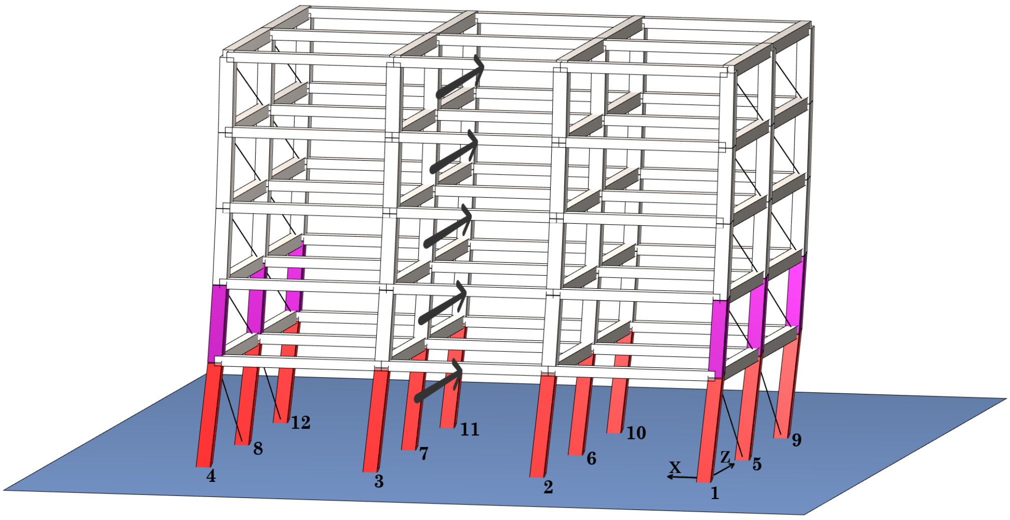



Figure 5.48: Infilled frame - Optimal configuration (column-infill shear interaction): Deformed shape (pushover along Z)

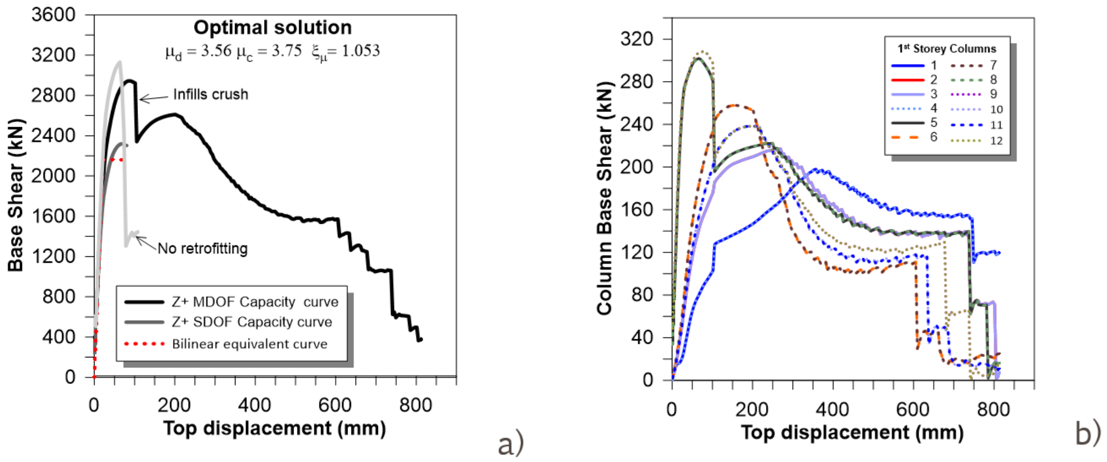



Figure 5.49: Infilled frame - Optimal configuration (column-infill shear interaction): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

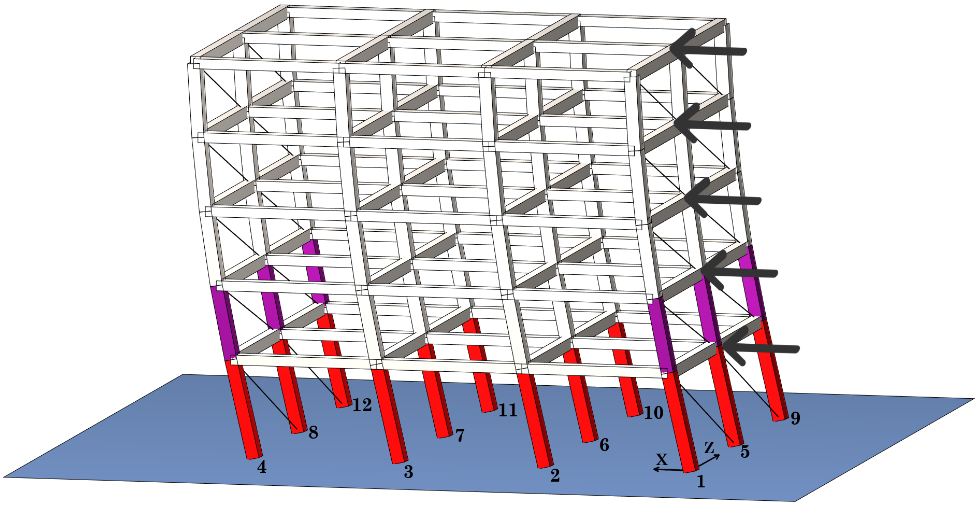
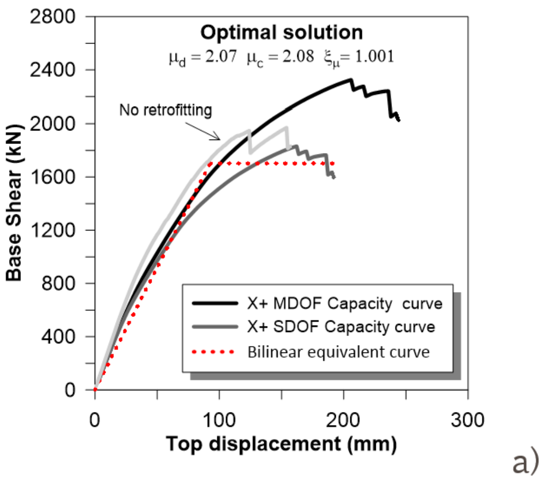



Figure 5.50: Infilled frame - Optimal configuration (column-infill shear interaction): Deformed shape (pushover along X)



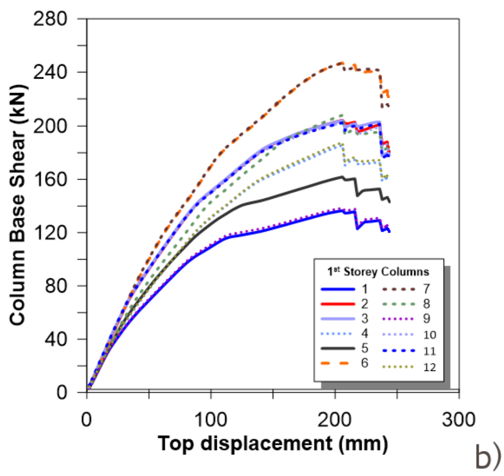




Figure 5.51: Infilled frame - Optimal configuration (column-infill shear interaction): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

### 5.2.3 Soft story mechanism

The third type of structural configuration used for the validation of the method is created to evaluate the effectiveness even in the presence of variation in stiffness in height.

In particular, the behaviour of the algorithm was analyzed in the case of structures characterized by a soft story mechanism. This type of structure is widely present in urban centres where large commercial spaces on the ground floor are constituted by large glass walls that represent a discontinuity in the height of the infills.

The absence of infills leads to a reduction in localized stiffness, leading to the concentration of floor drift on the ground floor with an increase in the shear stresses of the columns on this floor.

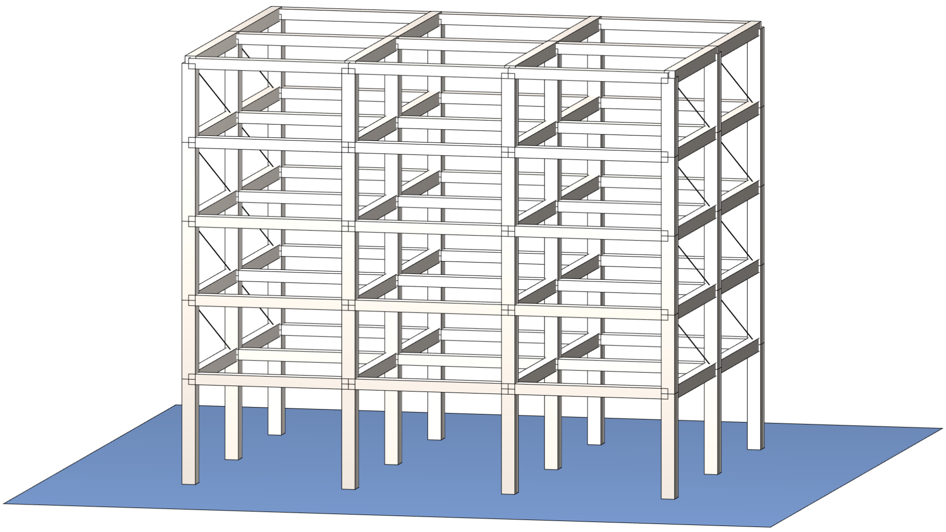


Figure 5.52: Structural configuration of the *soft-storey mechanism* structure

#### Preliminary tests

As previously done, two preliminary test were performed to verify the structure *as-built* and to verify the structure with all the columns at the first and second floor retrofitted.

As expected the pushover carried out along the Z direction highlight a major vulnerability whereas the behaviour along X direction remains quite similar to that of the bare frame configuration (Section 5.2.1).

In Figure 5.54 and Figure 5.56 are illustrated the pushover capacity curves for the *as-built* stucture respectively for forces acting along Z and along X.

In Figure 5.58 and Figure 5.60 are reported the pushover results for the structure with all the columns of the first and second floor retrofitted.

The results of these preliminary test are schematically reported in Table 5.6.

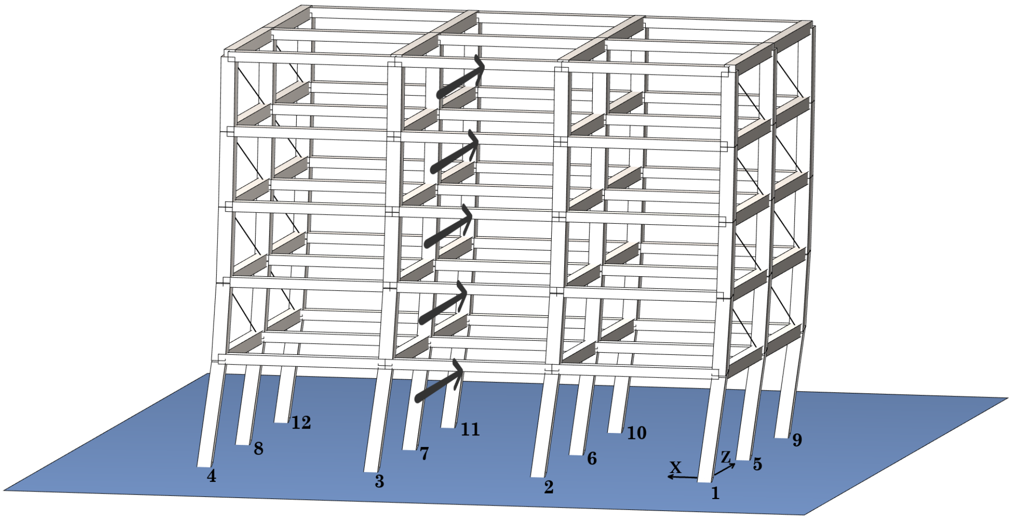



Figure 5.53: *Soft-story mechanism* structure - Preliminary test 1: Deformed shape (pushover along Z)

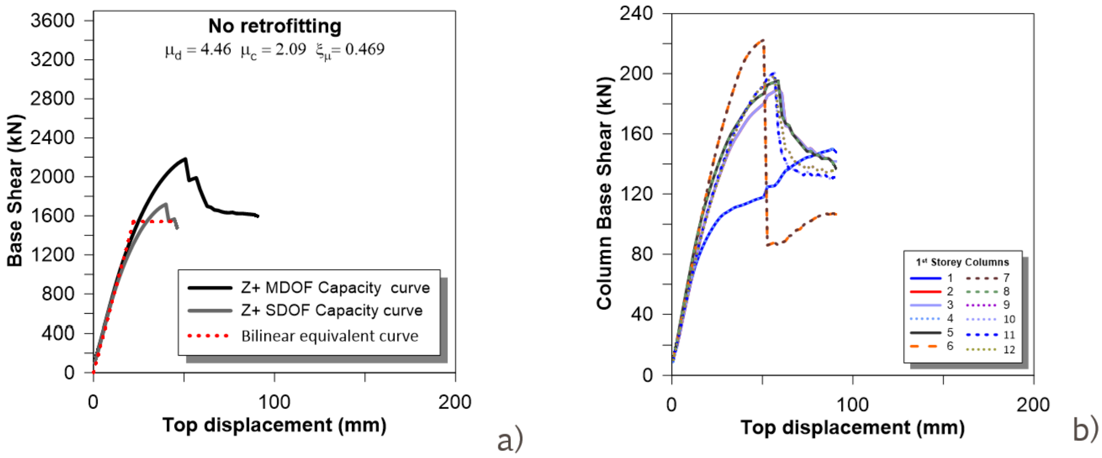



Figure 5.54: *Soft-story mechanism* structure - Preliminary test 1: (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

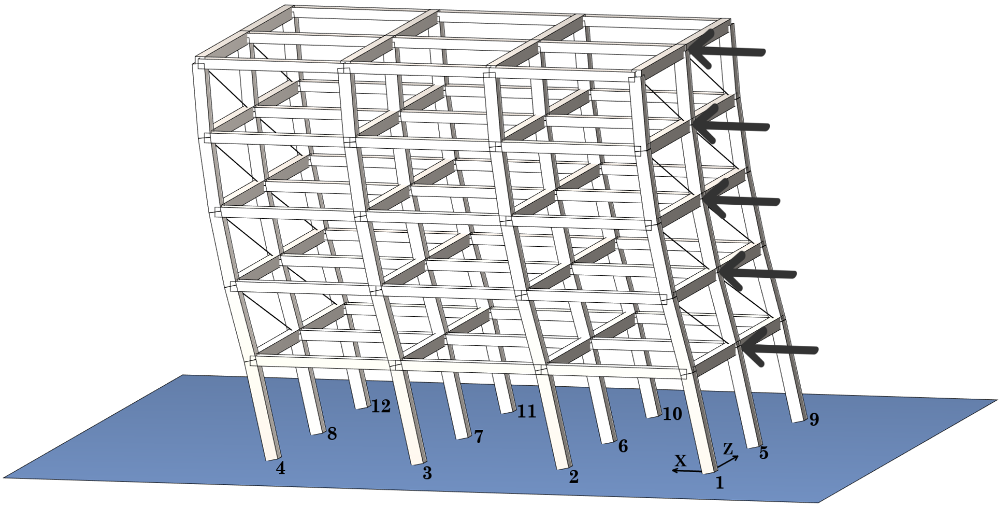



Figure 5.55: *Soft-story mechanism* structure - Preliminary test 1: Deformed shape (pushover along X)

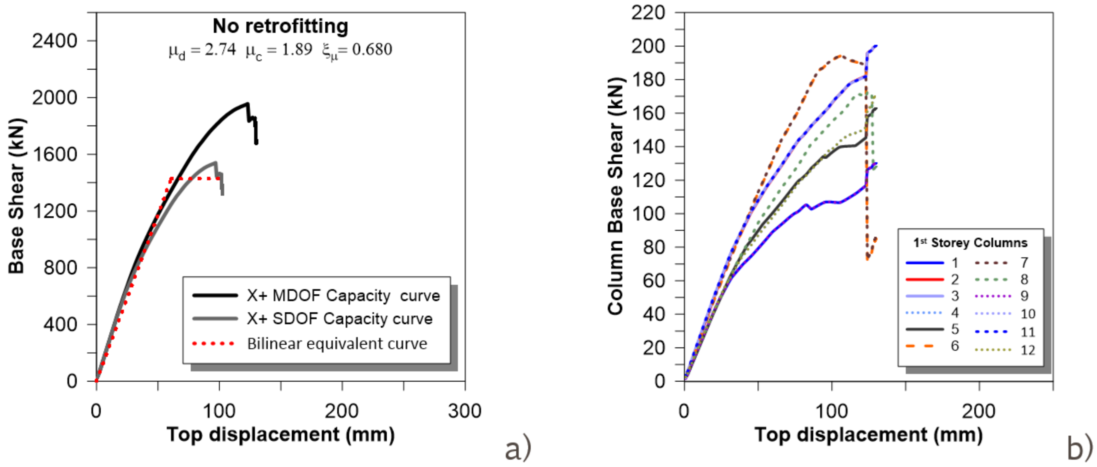



Figure 5.56: *Soft-story mechanism* structure - Preliminary test 1: (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

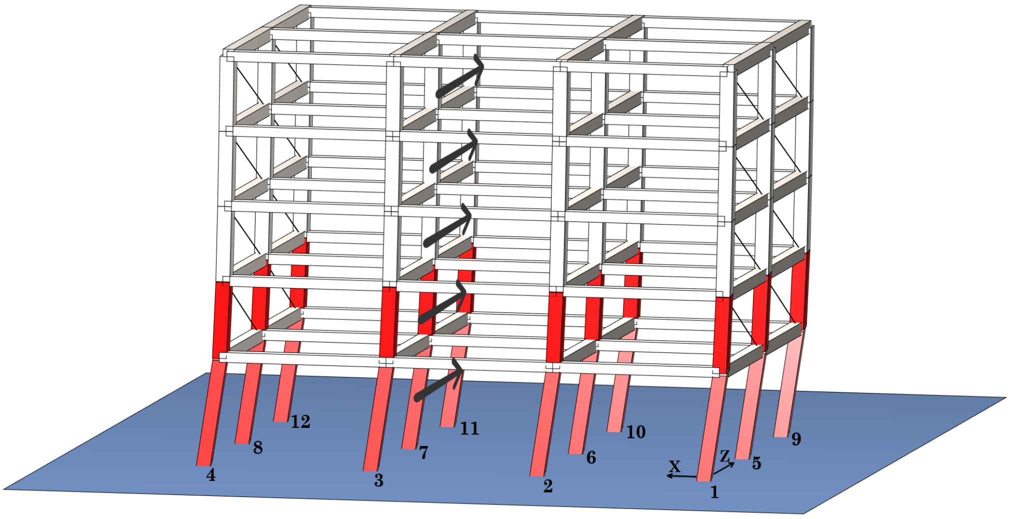



Figure 5.57: *Soft-story mechanism* structure - Preliminary test 2: Deformed shape (pushover along Z)

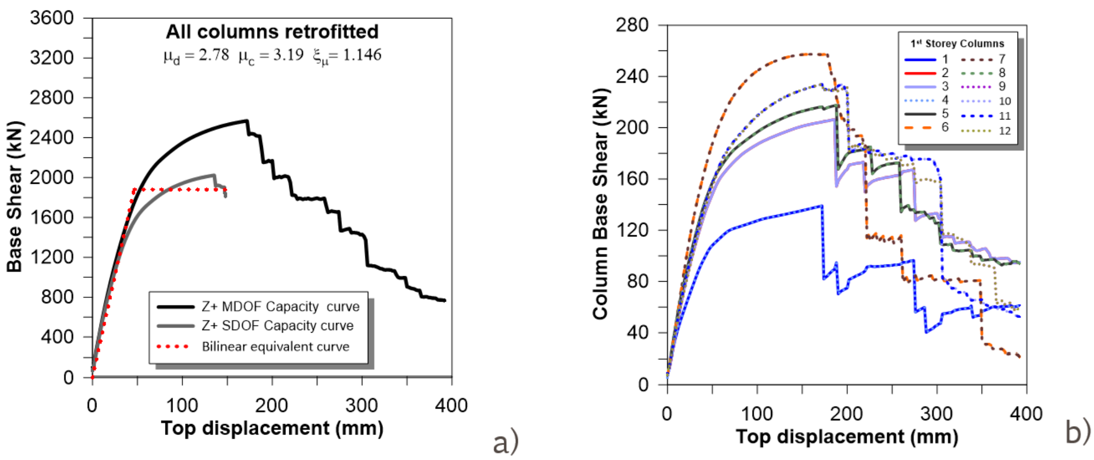



Figure 5.58: *Soft-story mechanism* structure - Preliminary test 2: (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

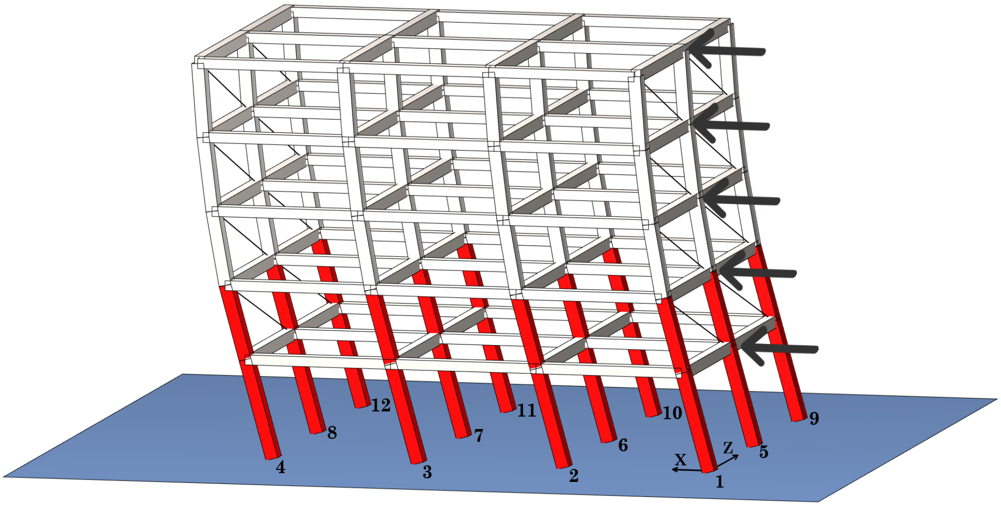



Figure 5.59: *Soft-story mechanism* structure - Preliminary test 2: Deformed shape (pushover along X)

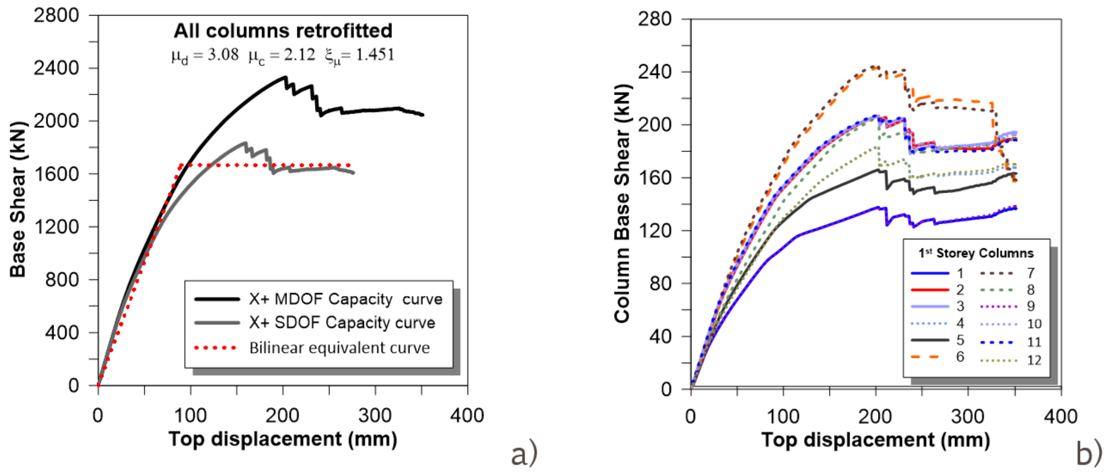



Figure 5.60: *Soft-story mechanism* structure - Preliminary test 2: (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

### Optimization results

For this structural configuration three different optimization process are performed, in the first one the shear verification of elements are disabled, in the second shear verification of elements are performed, in the last one the additional shear demand inducted by the infills is calculated (as presented in Chapter 4.4.2).

In Figure 5.61, Figure 5.62, and Figure 5.63 are illustrated the genetic algorithm process trend, in particular are reported the minumum and average value of the individuals of each generation analysed and the stall.

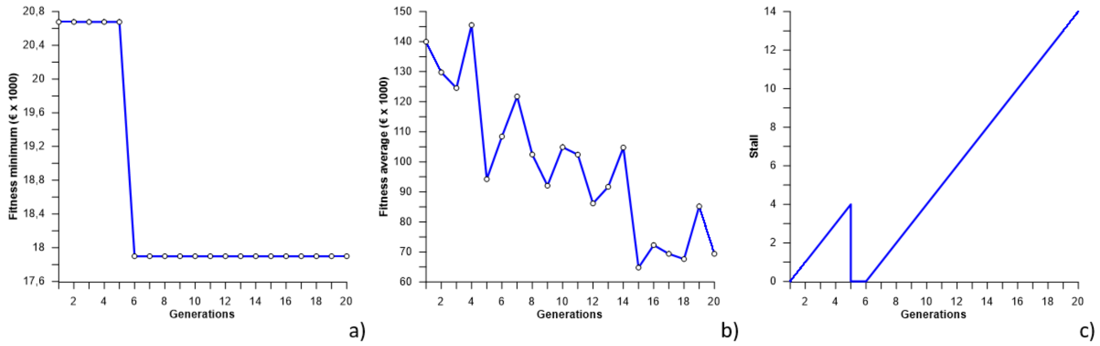



Figure 5.61: Genetic algorithm process parameters - *Soft-story* structure (without shear verification): (a) best solution's fitness value each generation (b) average fitness values for each generation (c) stall trend

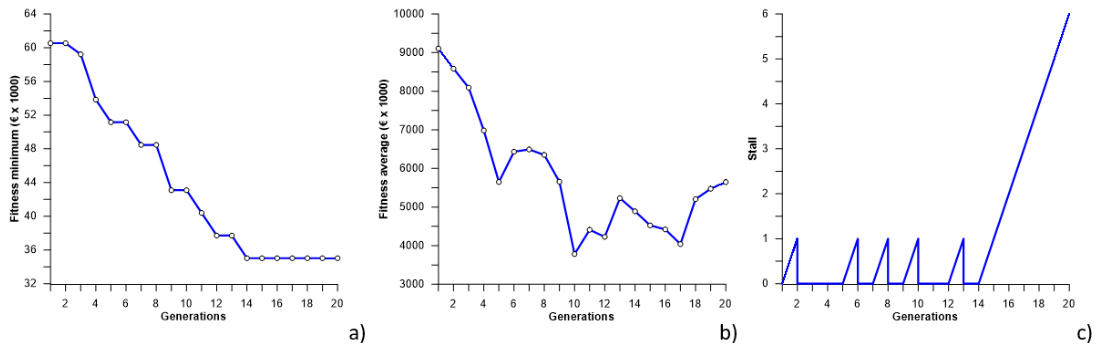



Figure 5.62: Genetic algorithm process parameters - *Soft-story* structure (with shear verification): (a) best solution's fitness value each generation (b) average fitness values for each generation (c) stall trend

The results of the three analysis are illustrated in Figure 5.64, Figure 5.68, and Figure 5.72 together with the respective pushover capacity curves.

As done in the previous cases, the final result has to be analysed to verify that the retrofitting arrangement is suitable to react to forces acting on different direction.

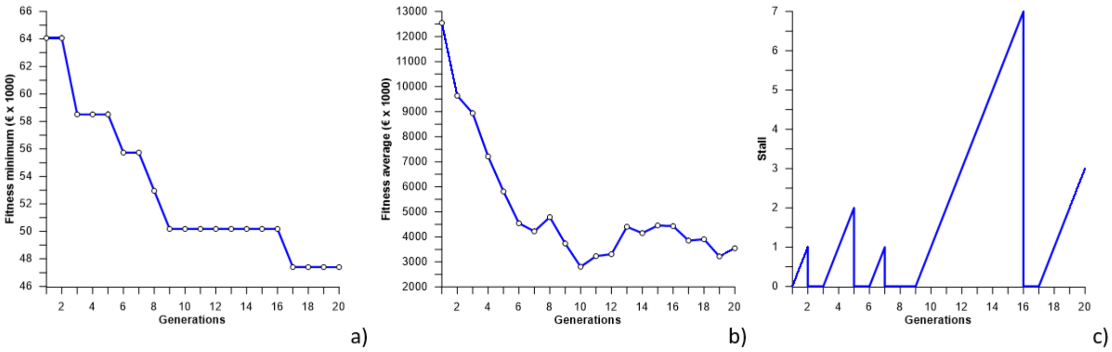



Figure 5.63: Genetic algorithm process parameters - *Soft-story* structure (with infills shear contribution): (a) best solution's fitness value each generation (b) average fitness values for each generation (c) stall trend

In this case the columns 1 and 4 on the first floor were retrofitted in case of force acting along the Z direction on negative verse the shear verification, due to the presence of infills, will increase leading to the collapse of the elements.

The analyses of this so defined final retrofitting configuration are reported in Figure 5.76 and Figure 5.78 associated with the respective capacity curves.

| Test | $s_b$ | $n_C$ | $C$ | direct. | $\mu_d$ | $\mu_c$ | $\xi_\mu$ | Ver. check |
|---|---|---|---|---|---|---|---|---|
| 1 | - | - | 0 € | Z | 4.46 | 2.09 | 0.469 | NO |
| | | | | X | 2.74 | 1.89 | 0.680 | NO |
| 2 | 150 | 24 | 69 618 € | Z | 2.68 | 3.19 | 1.146 | YES |
| | | | | X | 3.08 | 2.12 | 1.452 | YES |

Table 5.6: *Eccentric* structure - Results of preliminary tests

| Test | $s_b$ | $n_C$ | $C$ | direct. | $\mu_d$ | $\mu_c$ | $\xi_\mu$ | Ver. check |
|---|---|---|---|---|---|---|---|---|
| Without shear ver. | 150 | 5 | 24 195 € | Z | 3.51 | 4.04 | 1.149 | YES |
| | | | | X | 2.21 | 1.92 | 0.865 | YES |
| With shear ver. | 150 | 14 | 36 293 € | Z | 2.79 | 3.09 | 1.107 | YES |
| | | | | X | 2.49 | 2.34 | 0.938 | NO |
| Column-infills inter. | 150 | 16 | 47 401 € | Z | 3.19 | 2.78 | 1.185 | YES |
| | | | | X | 1.83 | 2.15 | 0.852 | NO |
| Symm. arrangemen | 150 | 16 | 52 956 € | Z | 3.19 | 2.78 | 1.185 | YES |
| | | | | X | 1.83 | 2.15 | 0.852 | YES |

Table 5.7: *Eccentric* structure - Results of optimization

In this case it is observed that only in the last configuration the structure is verified along the direction X. This is a prove that optimization framework but also the procedure analysed in this thesis it is suitable for finding the optimal solution efficiently.

The capacity demand ratio finally obtained (along Z) is $\xi_\mu = 1.185$, while the overall cost of the intervention is 52 956.00 €. It is noteworthy observing that the obtained cost is reduced by 24% with respect to the best solution found with preliminary tests (Preliminary test 2 - Chapter 5.2.3). However, in the face of this, the $\xi_\mu$ factor finally obtained for the analysis performed along Z (1.186) can be considered almost the same with respect to that obtained in preliminary test 2 (1.941) with a retrofitting cost of 69 618.9 €.

From an engineering point of view, the solution found by the optimization framework is reasonable, in fact, the columns on the ground floor are the elements which require the most significant ductility capacity so that the structure satisfies the verification, following the reduction of stiffness.

The need to reinforce the columns on the first floor is caused by the presence of infills and the increase in shear demand (as presented in Chapter 4.4.2).

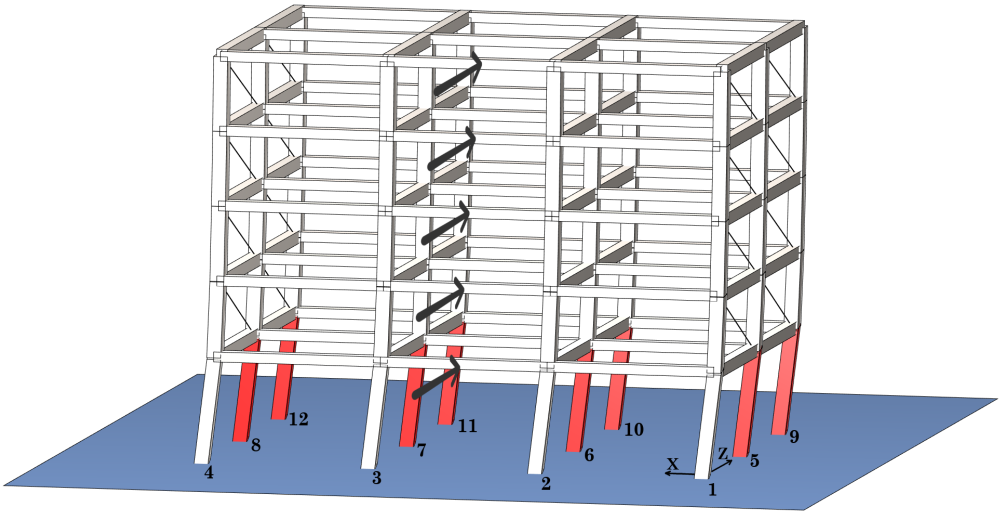



Figure 5.64: *Soft-story* structure - Optimal configuration (without shear verification): Deformed shape (pushover along Z)

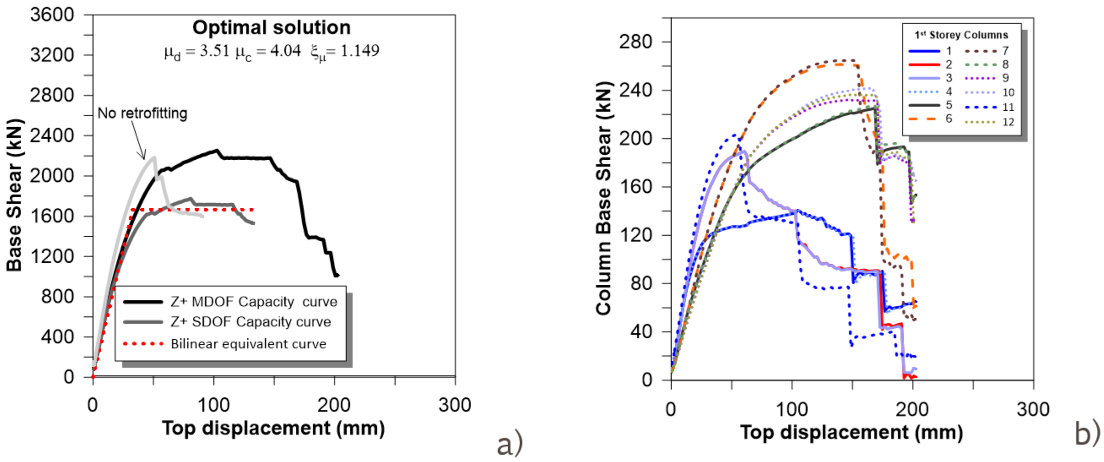



Figure 5.65: *Soft-story* structure - Optimal configuration (without shear verification): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

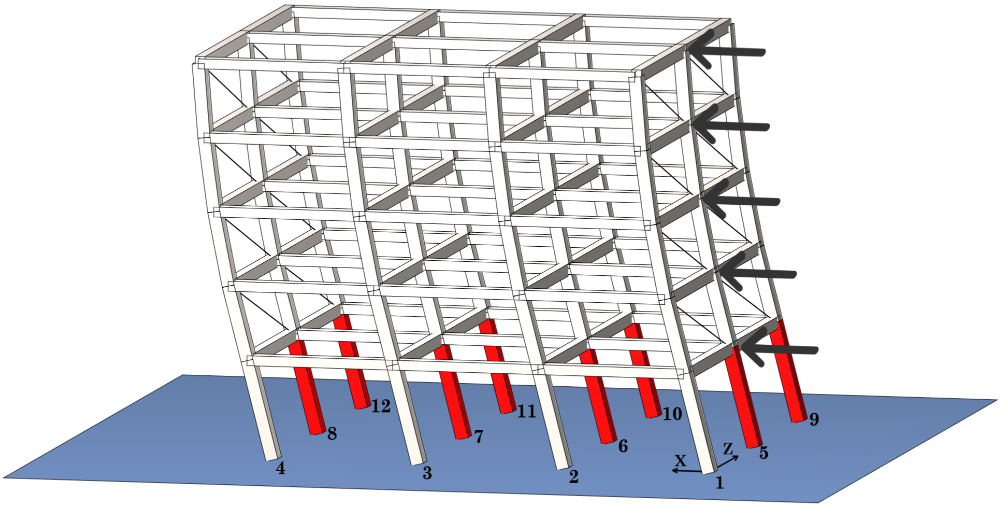



Figure 5.66: *Soft-story* structure - Optimal configuration (without shear verification): Deformed shape (pushover along X)

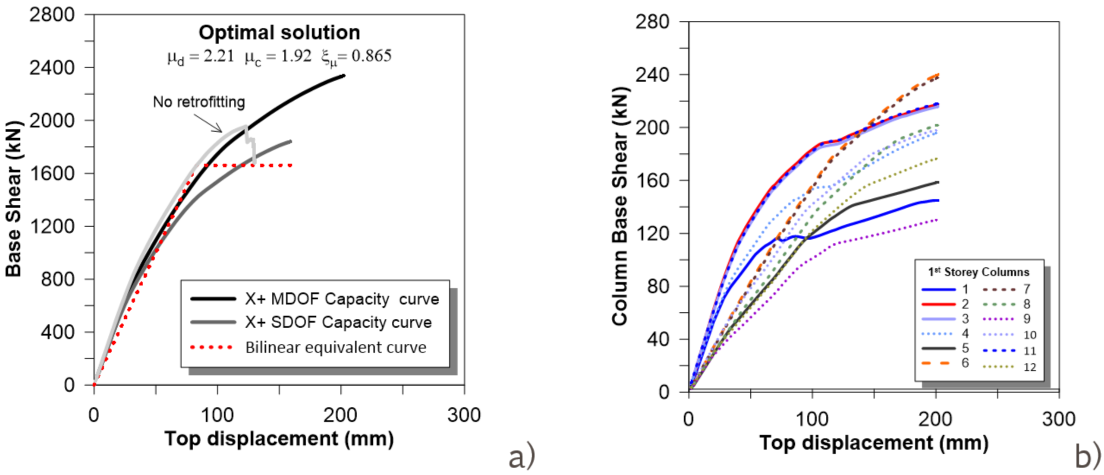



Figure 5.67: *Soft-story* structure - Optimal configuration (without shear verification): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

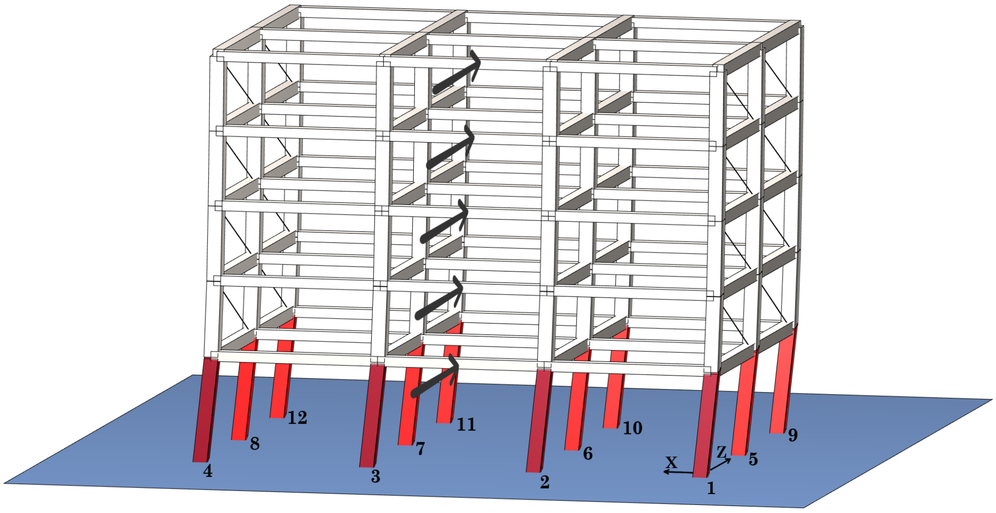



Figure 5.68: *Soft-story* structure - Optimal configuration (with shear verification): Deformed shape (pushover along Z)

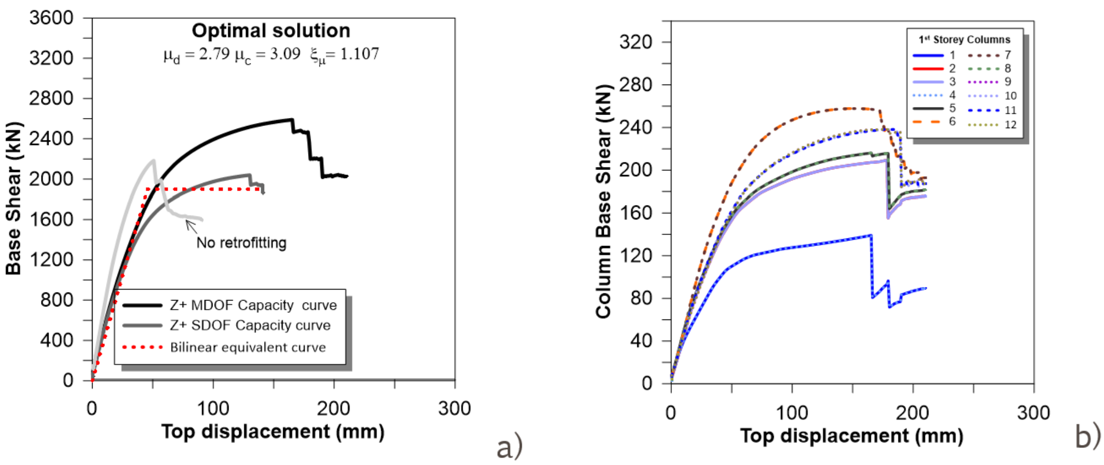



Figure 5.69: *Soft-story* structure - Optimal configuration (with shear verification): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

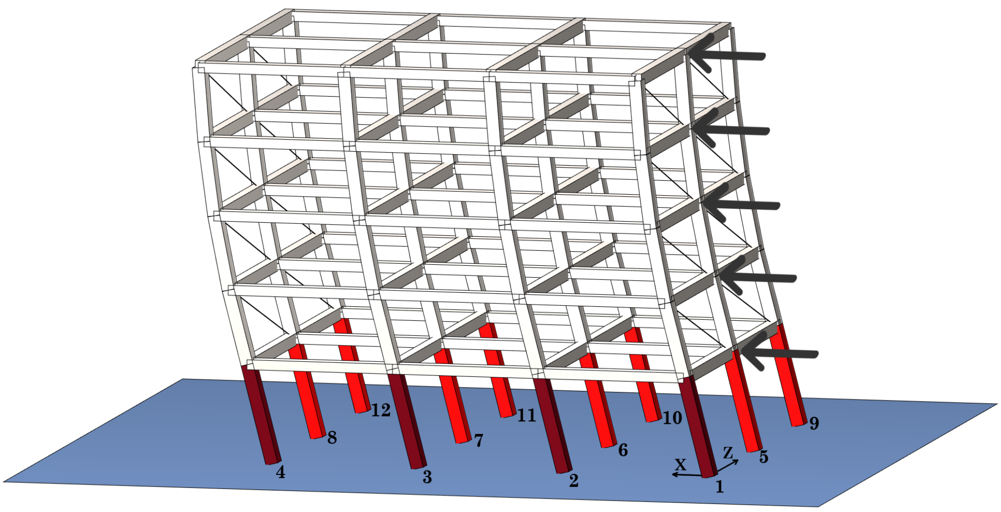



Figure 5.70: *Soft-story* structure - Optimal configuration (with shear verification): Deformed shape (pushover along X)

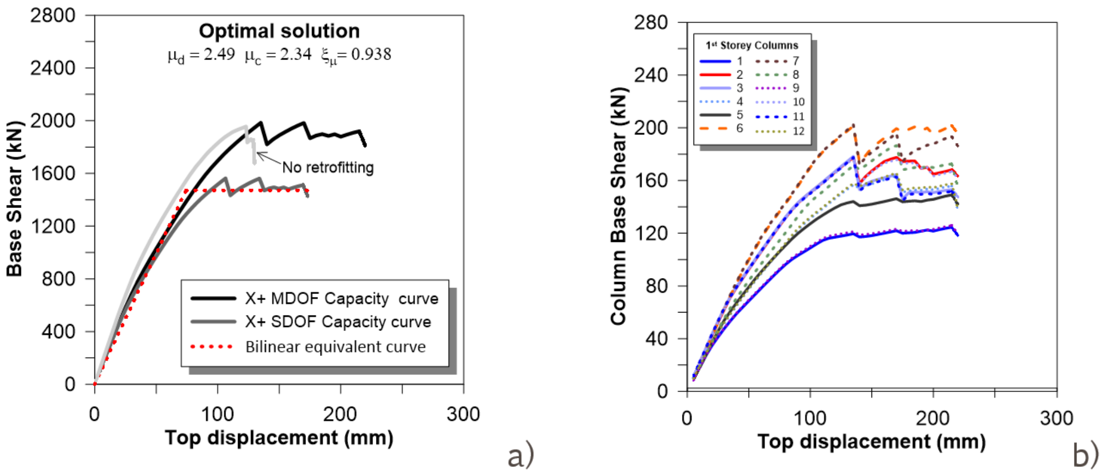



Figure 5.71: *Soft-story* structure - Optimal configuration (with shear verification): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

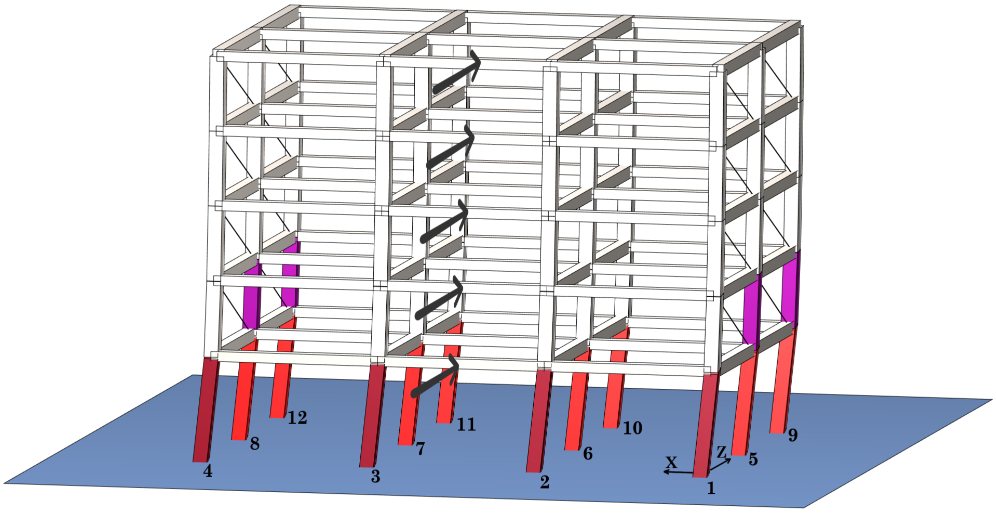



Figure 5.72: *Soft-story* structure - Optimal configuration (with column-infill interaction): Deformed shape (pushover along Z)

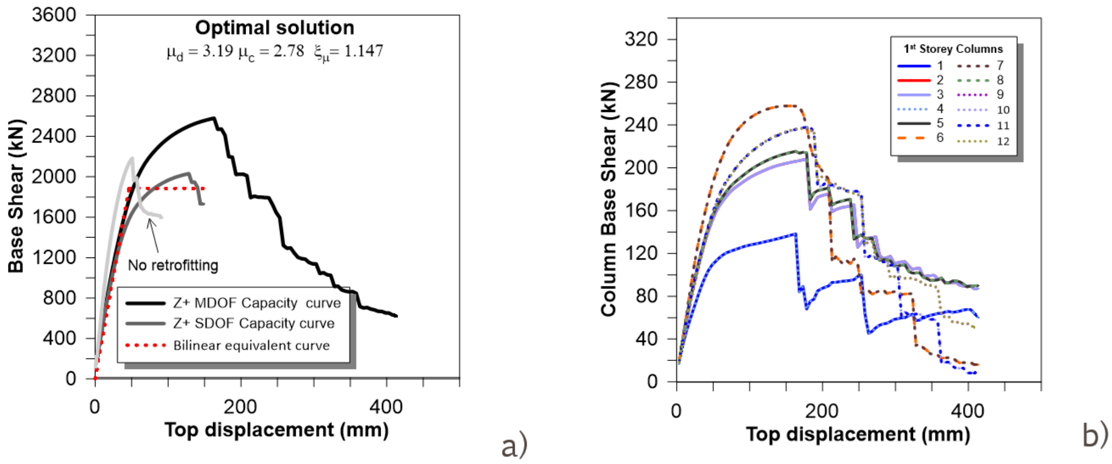



Figure 5.73: *Soft-story* structure - Optimal configuration (with column-infill interaction): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

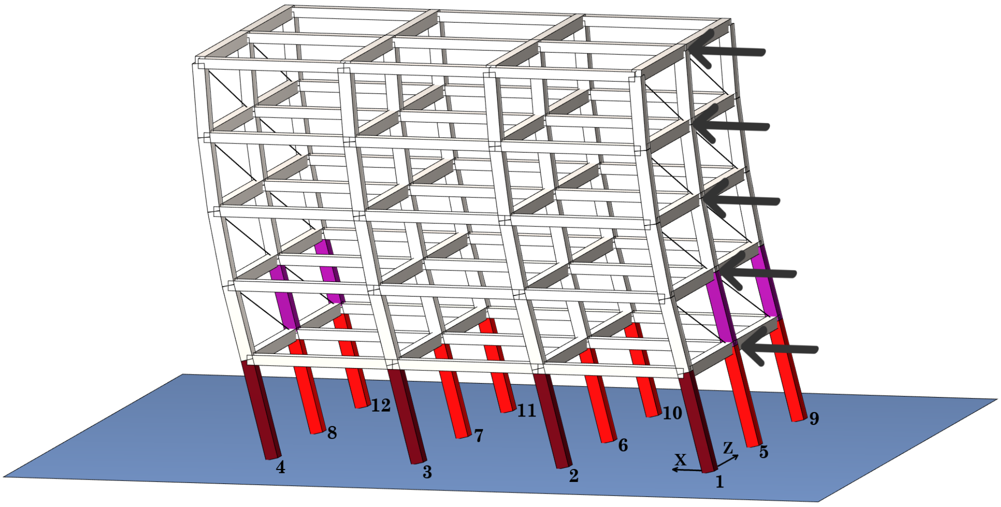



Figure 5.74: *Soft-story* structure - Optimal configuration (with column-infill interaction): Deformed shape (pushover along X)

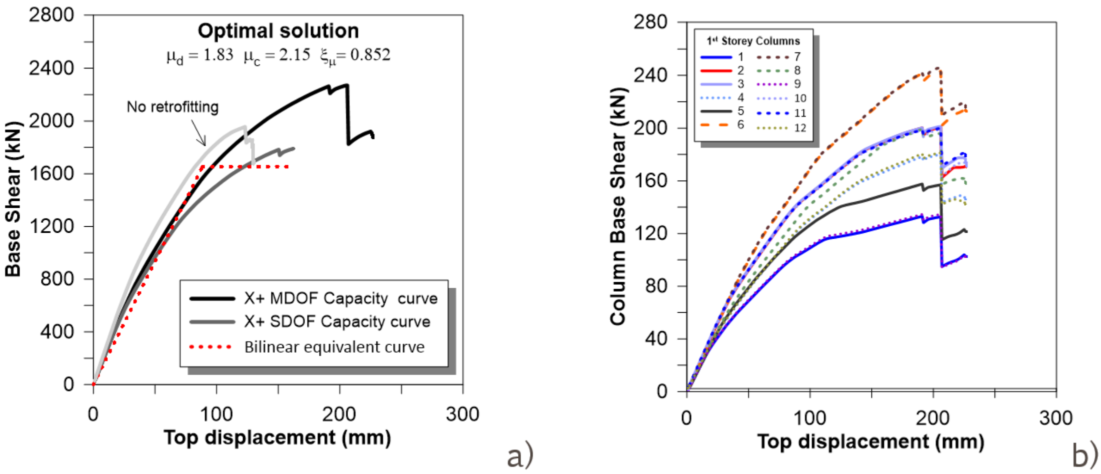



Figure 5.75: *Soft-story* structure - Optimal configuration (with column-infill interaction): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

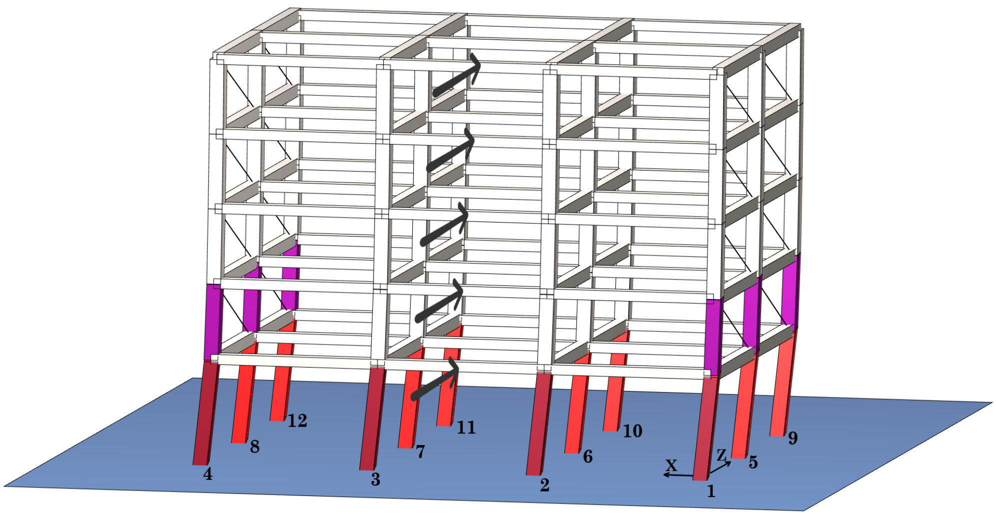



Figure 5.76: *Soft-story* structure - Optimal configuration (symmetric arrangement): Deformed shape (pushover along Z)

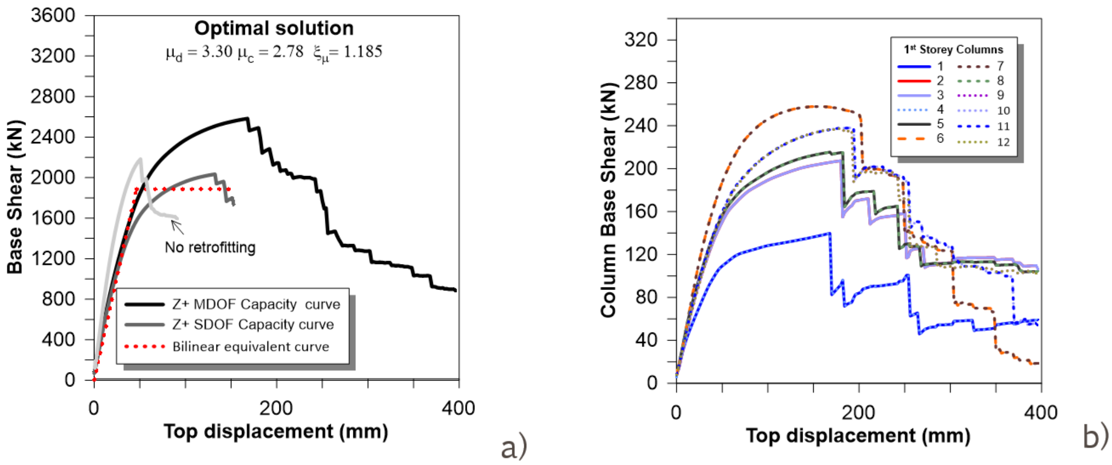



Figure 5.77: *Soft-story* structure - Optimal configuration (symmetric arrangement): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

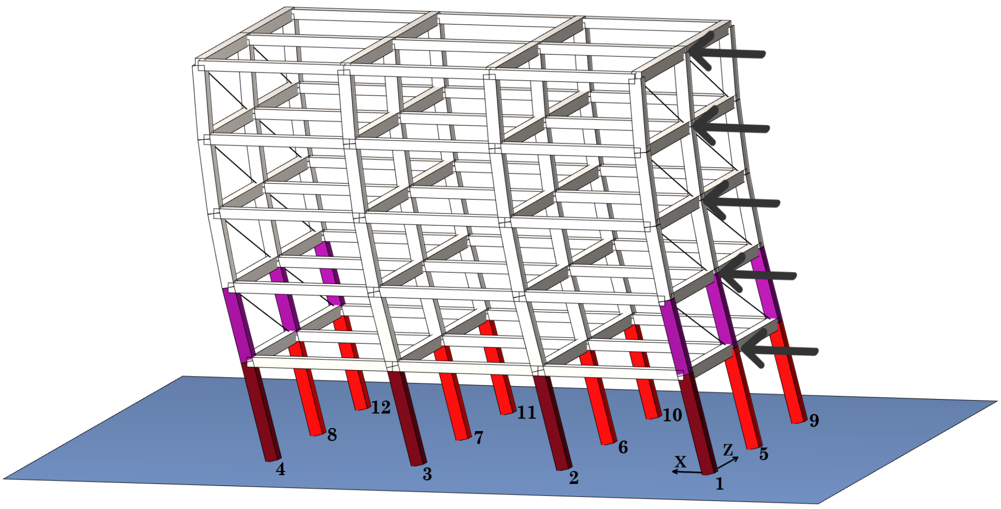



Figure 5.78: *Soft-story* structure - Optimal configuration (symmetric arrangement): Deformed shape (pushover along X)

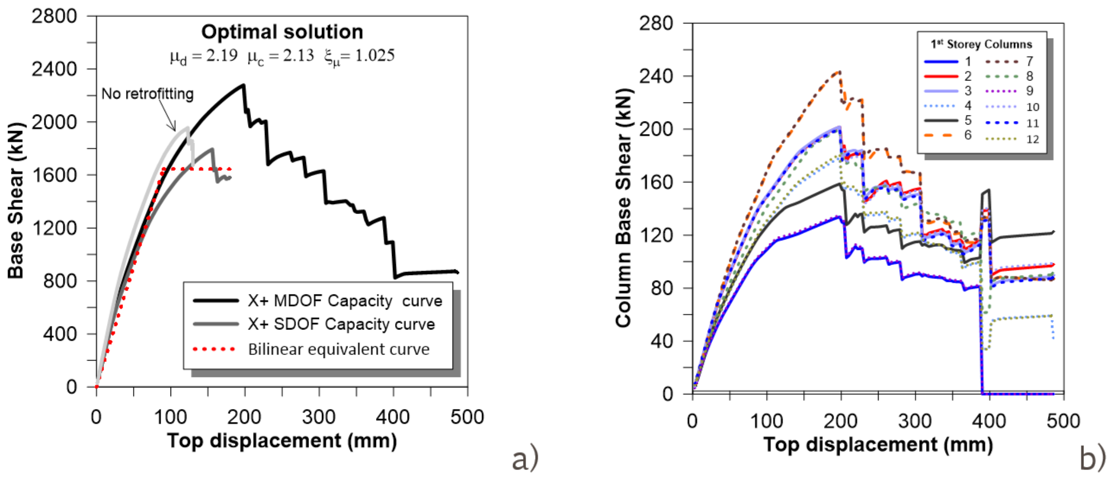



Figure 5.79: *Soft-story* structure - Optimal configuration (symmetric arrangement): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

### 5.2.4 Infilled frame with eccentric elements

The need to carry out this latest case study arises from the necessity to exploit the algorithm also for irregular in-plan structures. The need to carry out this latest case study arises from the necessity In detail, masonry infills are supposed being placed on in one of the external frames at all storeys (Figure 5.80). Infills are modelled as fiber-section struts as described in Chapter 4.2.3.

In the so defined structural configuration, the lateral response of the system is significantly modified along the Z direction, due to the increase in stiffness and the migration of the stiffness center toward the infilled frame. On the contrary, the behaviour along X direction remains quite similar to that of the bare frame configuration (Section 5.2.1).

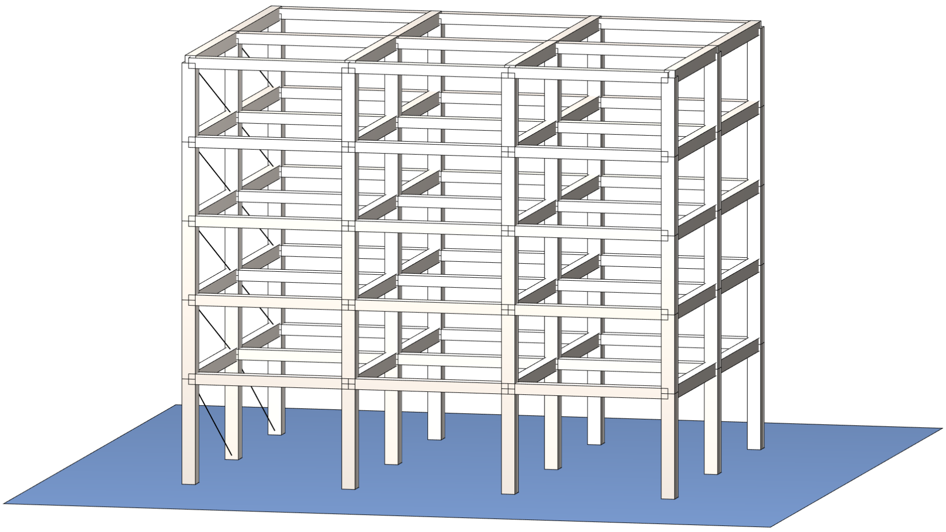


Figure 5.80: Structural configuration of the *Eccentric* structure

This is confirmed by the preliminary pushover analysis carried out, as in the previous cases, for the non-retrofitted structure (Figure 5.82) and for the structural configuration with all the columns on the first two floor retrofitted (Figure 5.86).

#### Preliminary tests

As previously done, two preliminary test were performed to verify the structure *as-built* and to verify the structure with all the columns at the first and second floor retrofitted.

As previously done, two preliminary test were performed to verify the structure *as-built* and to verify the structure with all the columns at the first and second floor retrofitted.

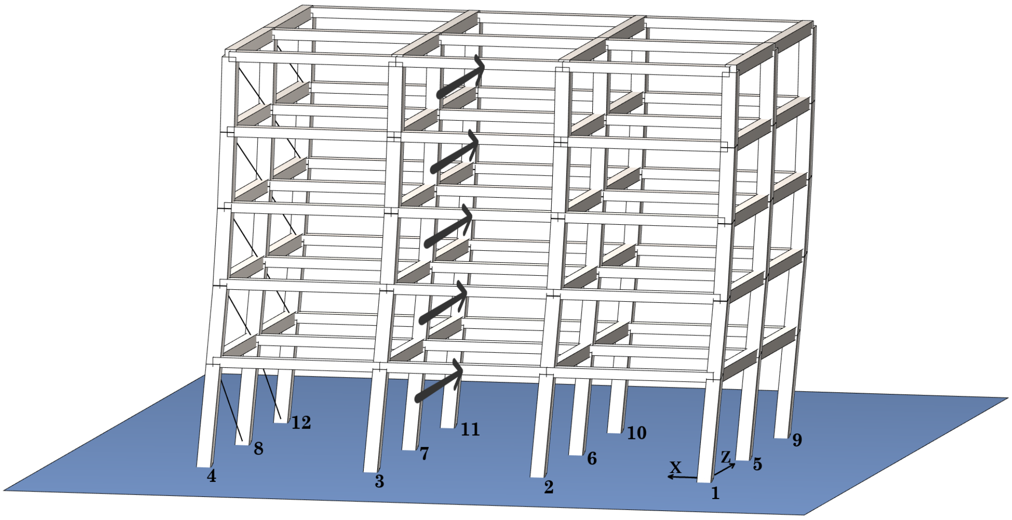



Figure 5.81: *Eccentric* structure - Preliminary test 1: Deformed shape (pushover along Z)

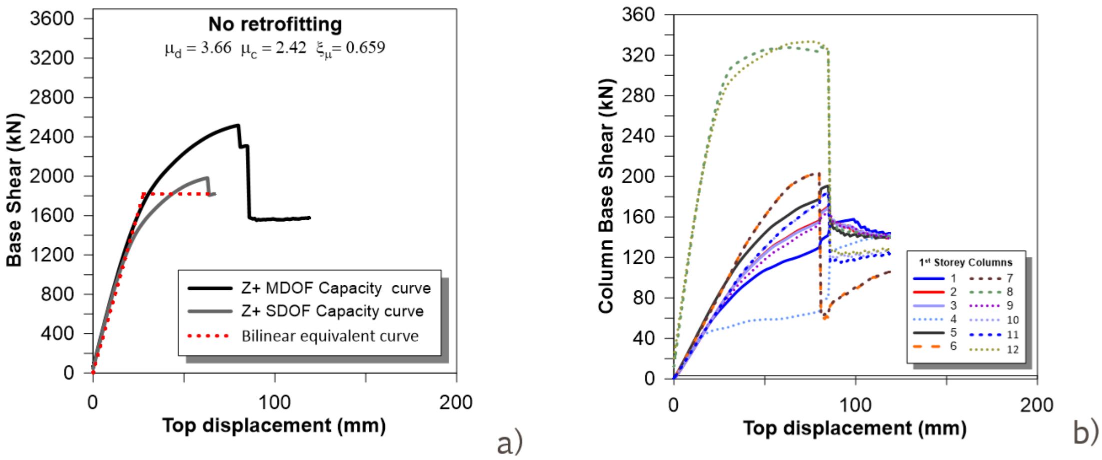



Figure 5.82: *Eccentric* structure - Preliminary test 1: (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

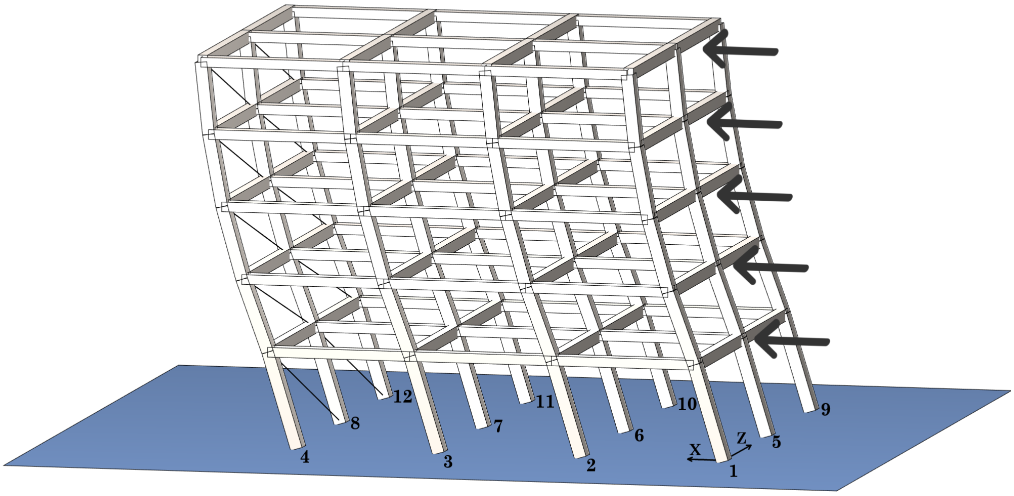



Figure 5.83: *Eccentric* structure - Preliminary test 1: Deformed shape (pushover along X)

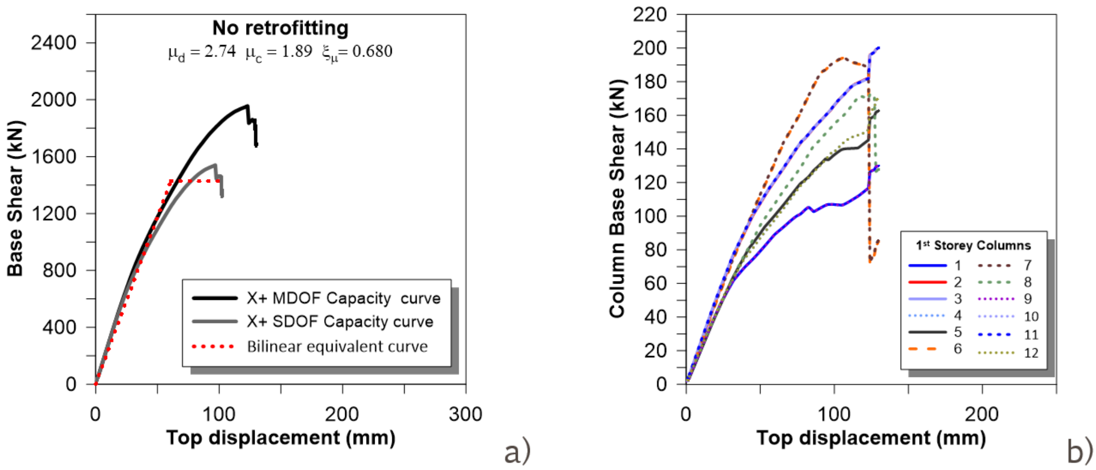



Figure 5.84: *Eccentric* structure - Preliminary test 1: (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

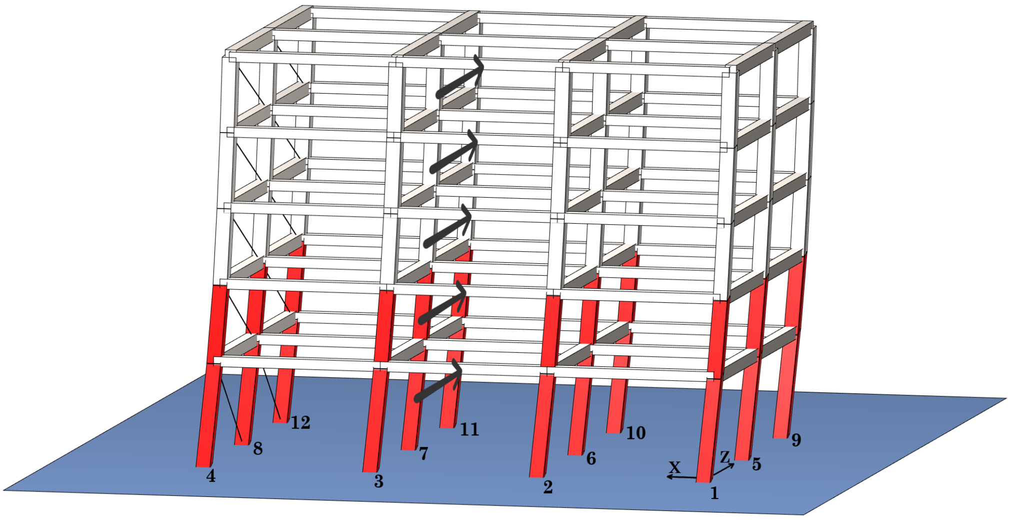



Figure 5.85: *Eccentric* structure - Preliminary test 2: Deformed shape (pushover along Z)

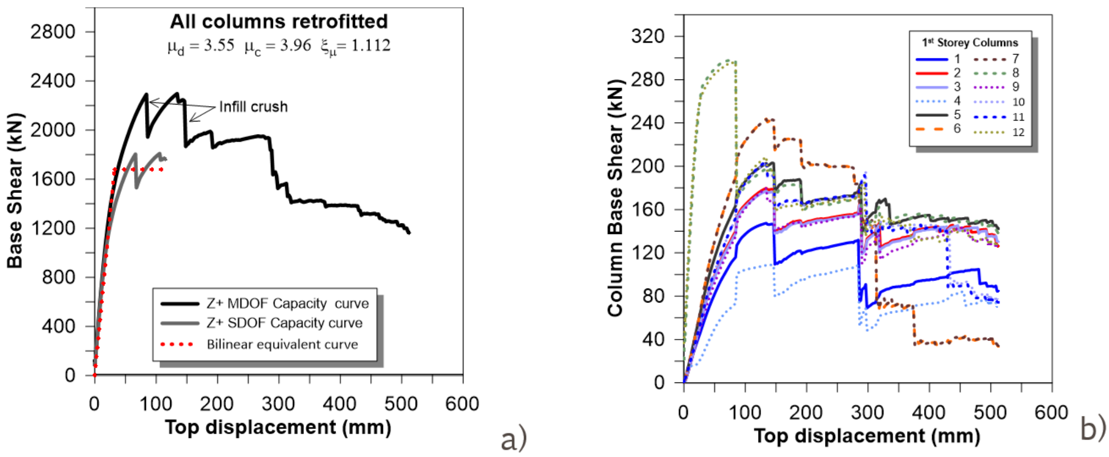



Figure 5.86: *Eccentric* structure - Preliminary test 2: (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

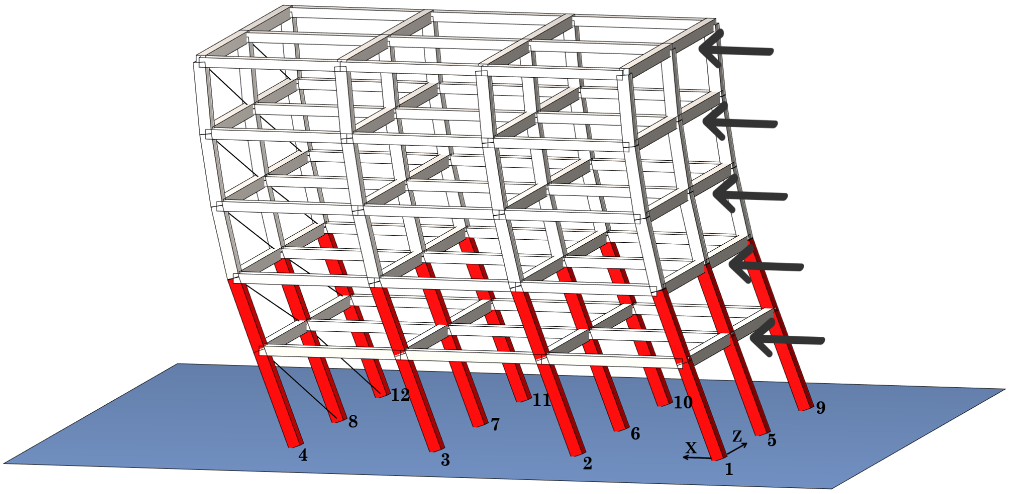



Figure 5.87: *Eccentric* structure - Preliminary test 2: Deformed shape (pushover along X)

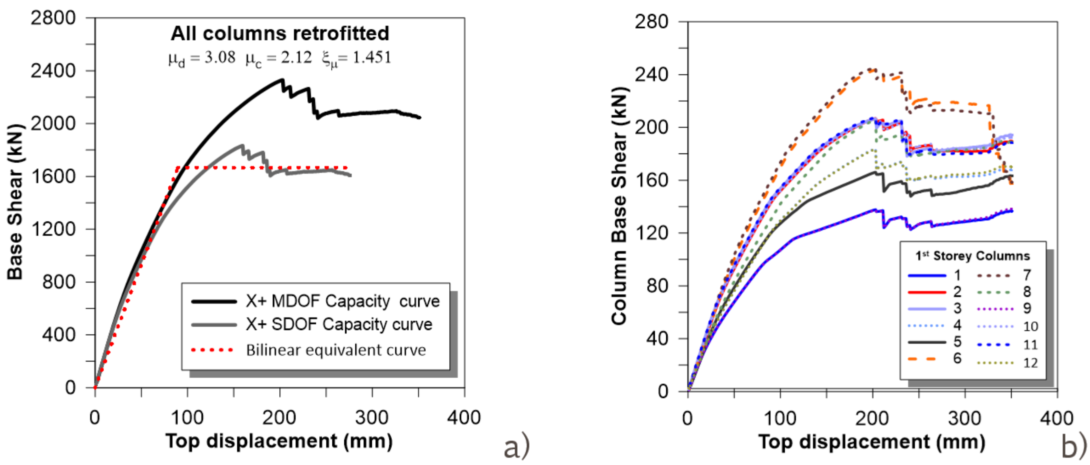



Figure 5.88: *Eccentric* structure - Preliminary test 2: (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

### Optimization results

As previously, for this structural configuration three different optimization process are performed, in the first one the shear verification of elements are disabled, in the second shear verification of elements are performed, in the last one the additional shear demand inducted by the infills is calculated (as presented in Chapter 4.4.2).

In Figure 5.89, Figure 5.90, and Figure 5.91 are illustrated the genetic algorithm process trend, in particular are reported the minumum and average value of the individuals of each generation analysed and the stall.

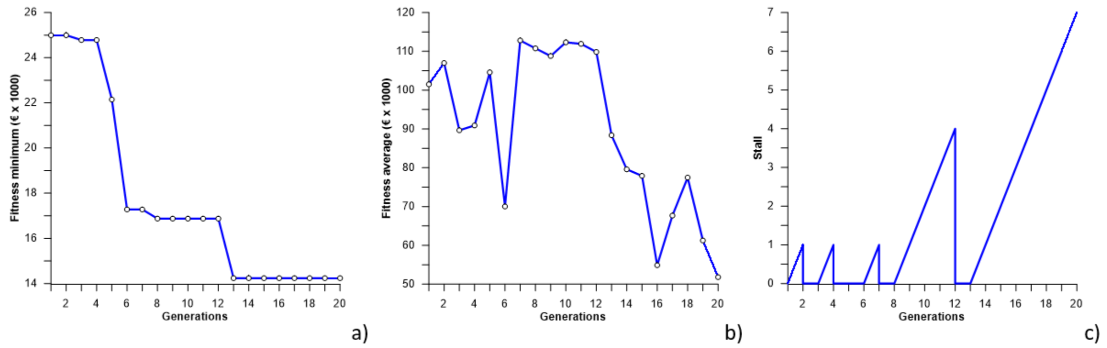



Figure 5.89: Genetic algorithm process parameters - *Eccentric* structure (without shear verification): (a) best solution's fitness value each generation (b) average fitness values for each generation (c) stall trend

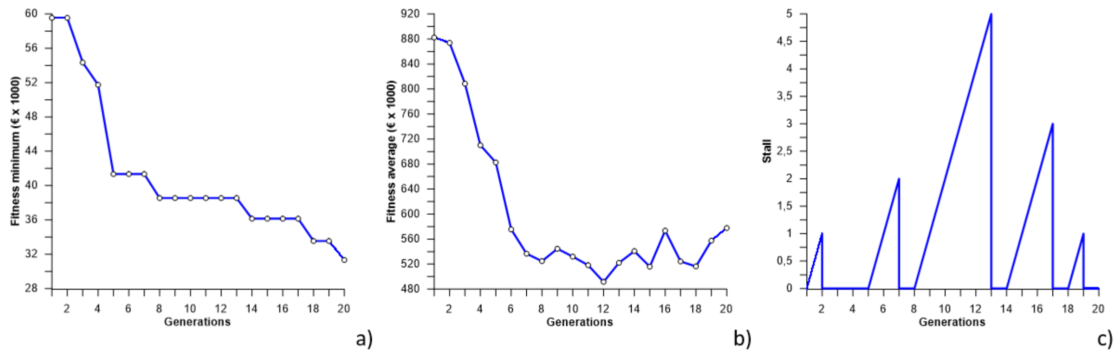



Figure 5.90: Genetic algorithm process parameters - *Eccentric* structure (with shear verification): (a) best solution's fitness value each generation (b) average fitness values for each generation (c) stall trend

The results of the three analysis are illustrated in the following Figure 5.92, Figure 5.96, and Figure 5.100 together with the respective pushover capacity curves.

As done in the previous cases, the final result has to be analysed to verify that the retrofitting arrangement is suitable to react to forces acting on different direction.

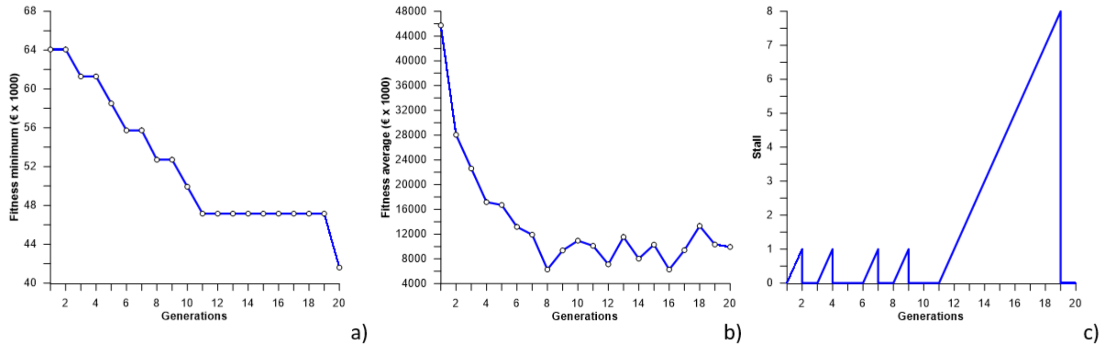



Figure 5.91: Genetic algorithm process parameters - *Eccentric* structure (with infills shear contribution): (a) best solution's fitness value each generation (b) average fitness values for each generation (c) stall trend

In this case the columns 1 and 4 on the first floor were retrofitted in case of force acting along the Z direction on negative verse the shear verification, due to the presence of infills, will increase leading to the collapse of the elements.

The analyses of this so defined final retrofitting configuration are reported in Figure 5.104 and Figure 5.106 associated with the respective capacity curves.

| Test | $s_b$ | $n_C$ | $C$ | direct. | $\mu_d$ | $\mu_c$ | $\xi_\mu$ | Ver. check |
|---|---|---|---|---|---|---|---|---|
| 1 | - | - | 0 € | Z | 3.66 | 2.42 | 0.659 | NO |
| | | | | X | 2.74 | 1.89 | 0.680 | NO |
| 2 | 150 | 24 | 69 618 € | Z | 3.55 | 3.96 | 1.112 | YES |
| | | | | X | 3.08 | 2.12 | 1.451 | YES |

Table 5.8: *Soft-storey* structure - Results of preliminary tests

| Test | $s_b$ | $n_C$ | $C$ | direct. | $\mu_d$ | $\mu_c$ | $\xi_\mu$ | Ver. check |
|---|---|---|---|---|---|---|---|---|
| Without shear ver. | 150 | 5 | 15 122 € | Z | 3.61 | 3.87 | 1.070 | YES |
| | | | | X | 2.21 | 1.92 | 0.865 | YES |
| With shear ver. | 150 | 11 | 33 268 € | Z | 3.46 | 4.26 | 1.203 | YES |
| | | | | X | 2.14 | 1.81 | 0.848 | NO |
| Column-infills inter. | 150 | 13 | 38 822 € | Z | 3.57 | 4.23 | 1.184 | YES |
| | | | | X | 1.83 | 2.15 | 0.851 | NO |
| Symm. arrangemen | 150 | 15 | 44 624 € | Z | 3.96 | 3.36 | 1.180 | YES |
| | | | | X | 2.19 | 2.13 | 1.025 | YES |

Table 5.9: *Soft-storey* structure - Results of optimization

As in the previous case (pag.118), only in the last configuration the structure is verified along the direction X. This confirms that optimization framework but also the procedure analysed in this thesis it is suitable for finding the optimal solution efficiently.

The capacity demand ratio finally obtained (along Z) is $\xi_\mu = 1.025$, while the overall cost of the intervention is 44 624.55 €. It is noteworthy observing that the obtained cost is reduced by 36% with respect to the best solution found with preliminary tests (Preliminary test 2 - Chapter 5.2.4).

However, in the face of this, the $\xi_\mu$ factor finally obtained for the analysis performed along Z ($-8\%$) can be considered almost the same with respect to that obtained in preliminary test 2 (1.112) with a retrofitting cost of 69 618.9 €.

From an engineering point of view, the solution found by the optimization framework is reasonable, in fact, the columns on the ground floor are the elements which require the most significant ductility capacity so that the structure satisfies the verification, following the reduction of stiffness.

The need to reinforce the columns on the first floor is caused by the presence of infills and the increase in shear demand (as presented in Chapter 4.4.2).

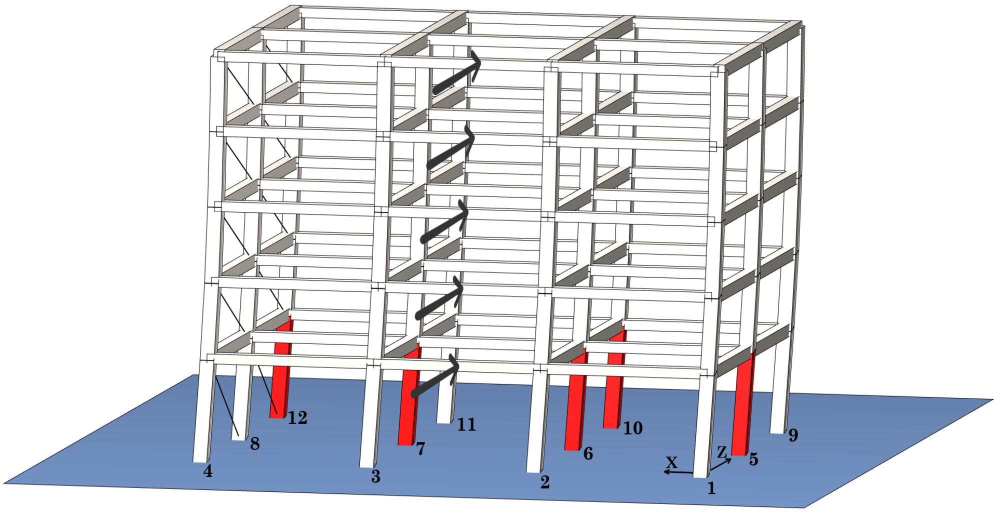



Figure 5.92: *Eccentric* structure - Optimal configuration (without shear verification): Deformed shape (pushover along Z)

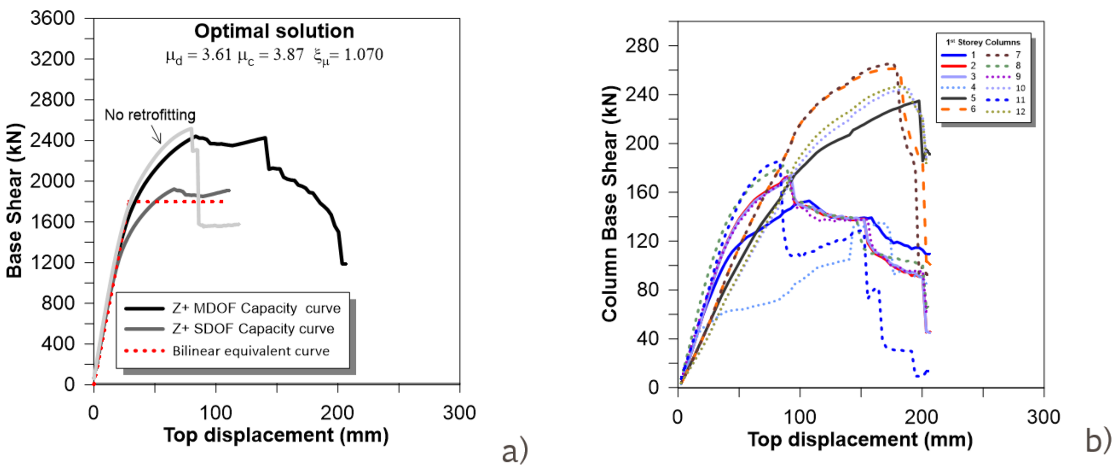



Figure 5.93: *Eccentric* structure - Optimal configuration (without shear verification): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

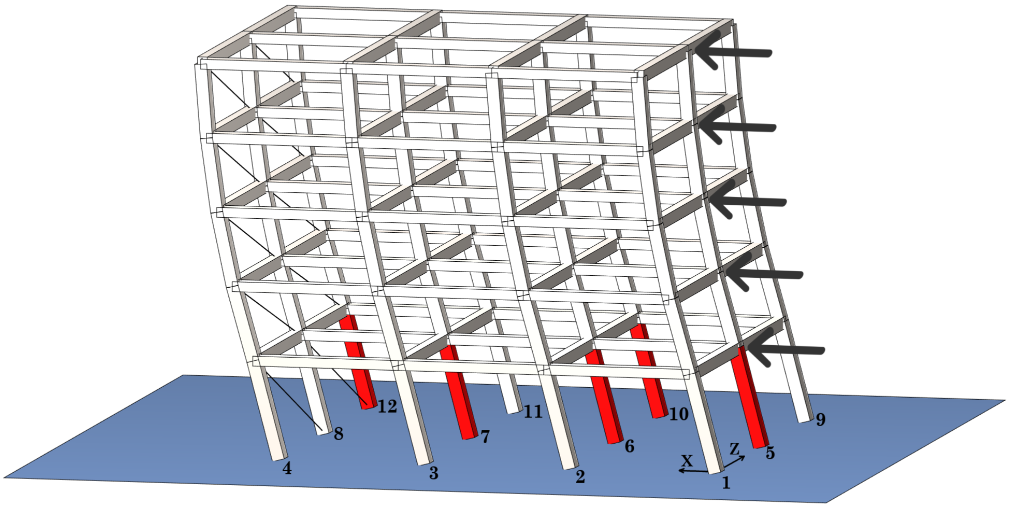



Figure 5.94: *Eccentric* structure - Optimal configuration (without shear verification): Deformed shape (pushover along X)

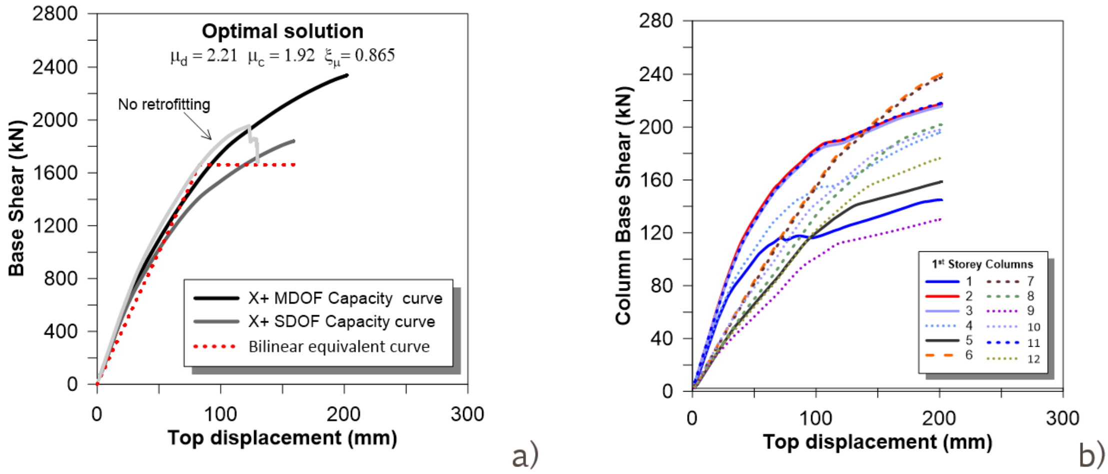



Figure 5.95: *Eccentric* structure - Optimal configuration (without shear verification): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

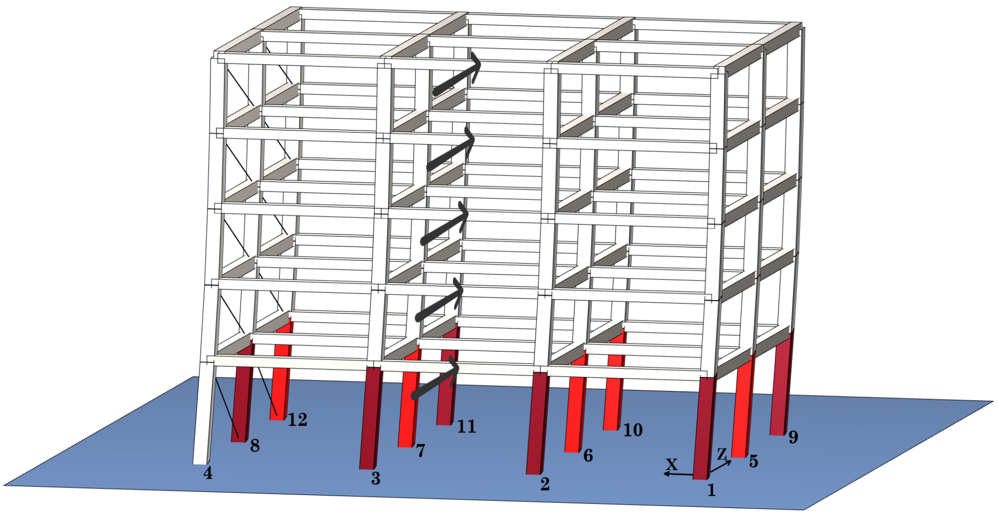



Figure 5.96: *Eccentric* structure - Optimal configuration (with shear verification): Deformed shape (pushover along Z)

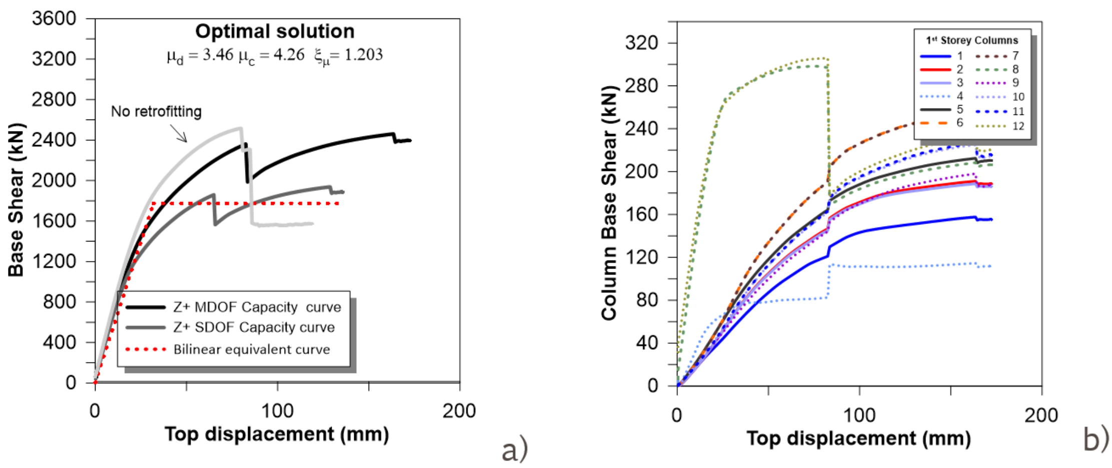



Figure 5.97: *Eccentric* structure - Optimal configuration (with shear verification): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

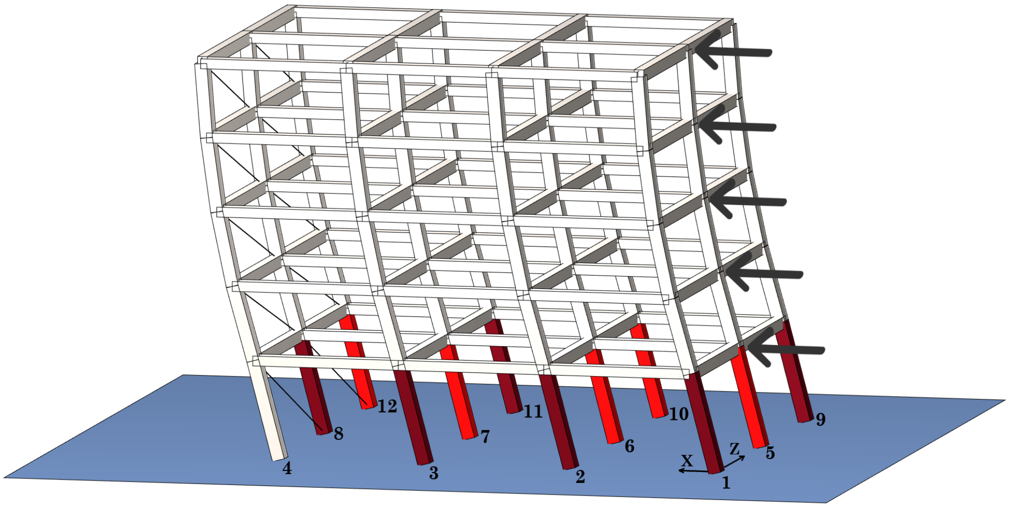



Figure 5.98: *Eccentric* structure - Optimal configuration (with shear verification): Deformed shape (pushover along X)

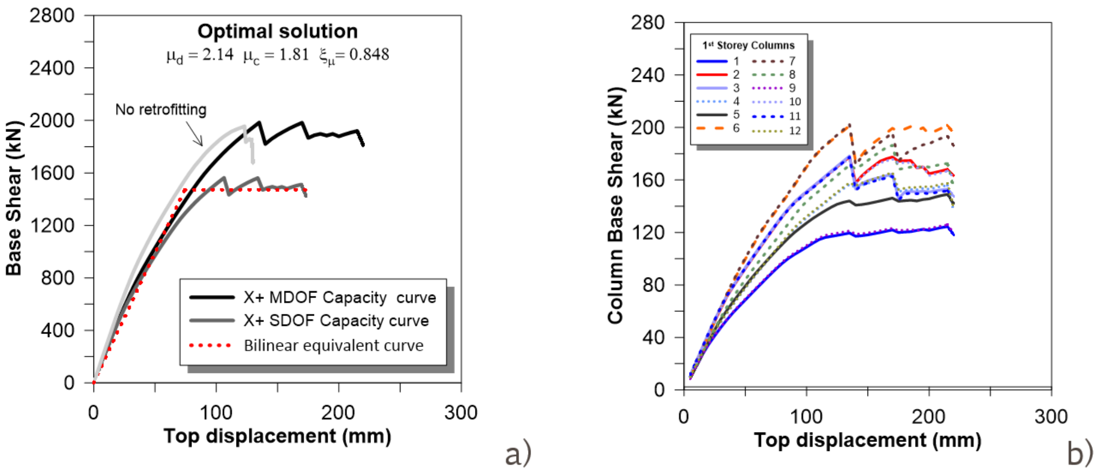



Figure 5.99: *Eccentric* structure - Optimal configuration (with shear verification): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

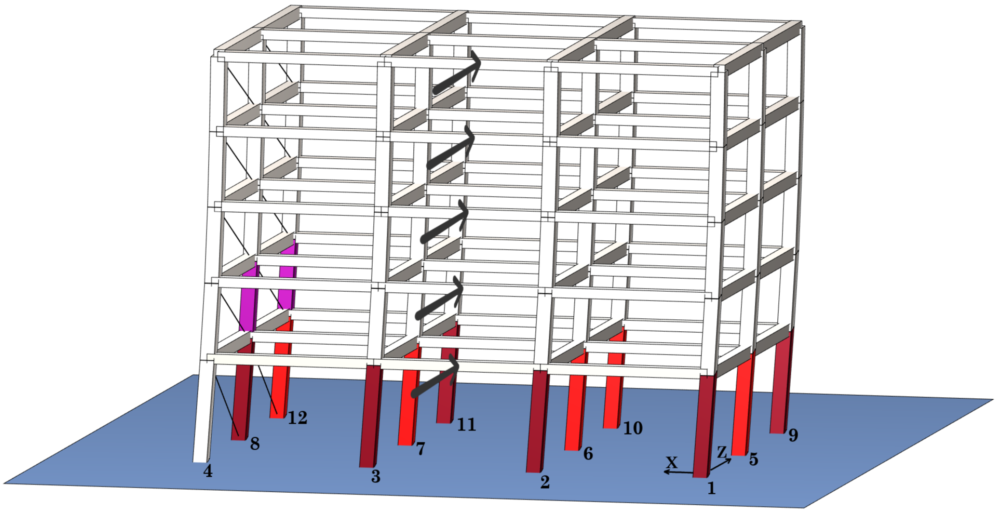



Figure 5.100: *Eccentric* structure - Optimal configuration (with column-infill interaction): Deformed shape (pushover along Z)

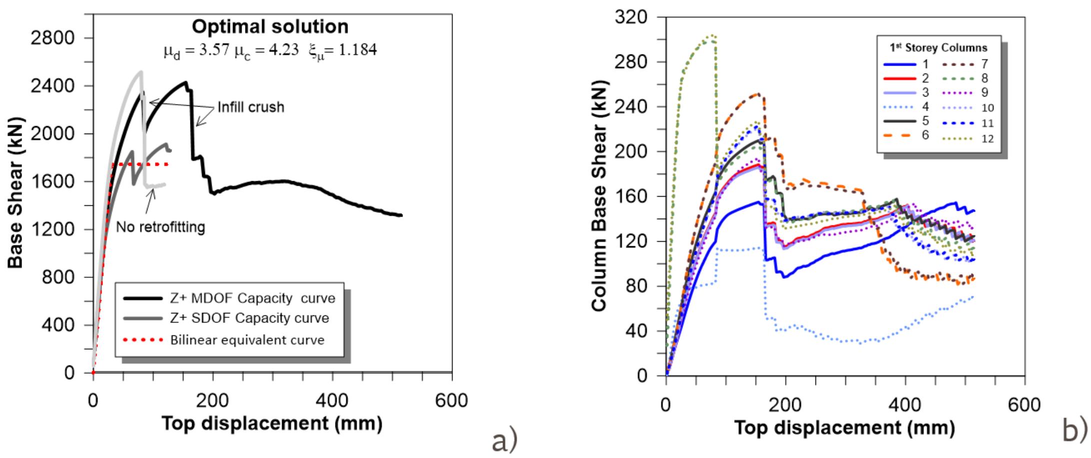



Figure 5.101: *Eccentric* structure - Optimal configuration (with column-infill interaction): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

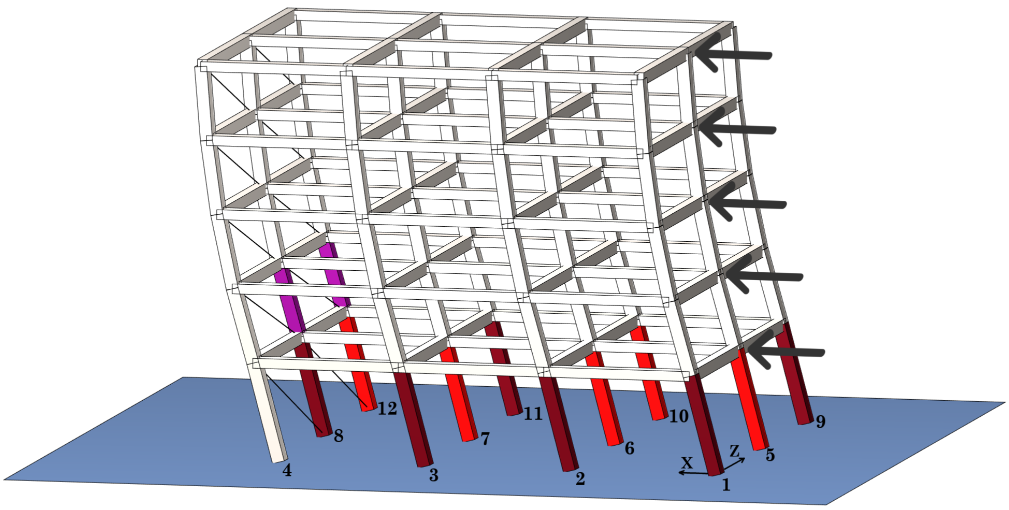



Figure 5.102: *Eccentric* structure - Optimal configuration (with column-infill interaction): Deformed shape (pushover along X)

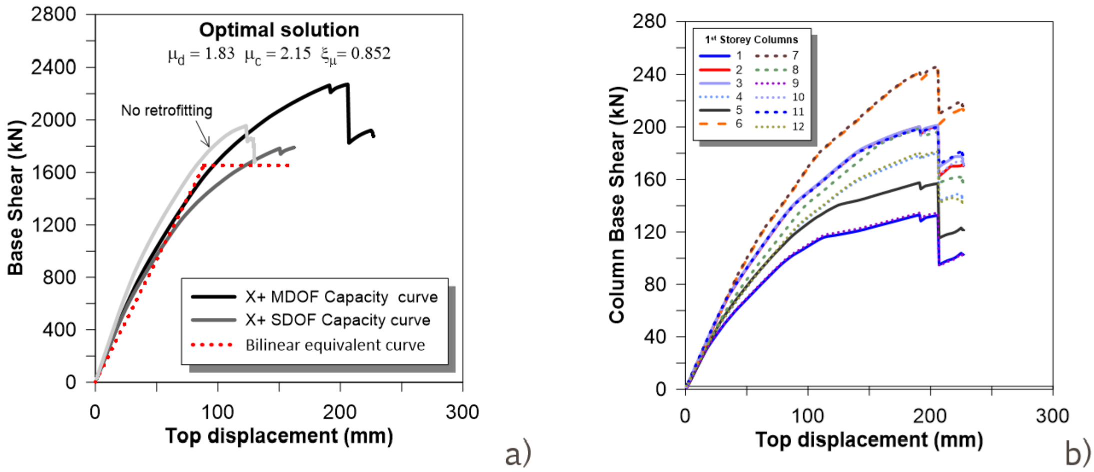



Figure 5.103: *Eccentric* structure - Optimal configuration (with column-infill interaction): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

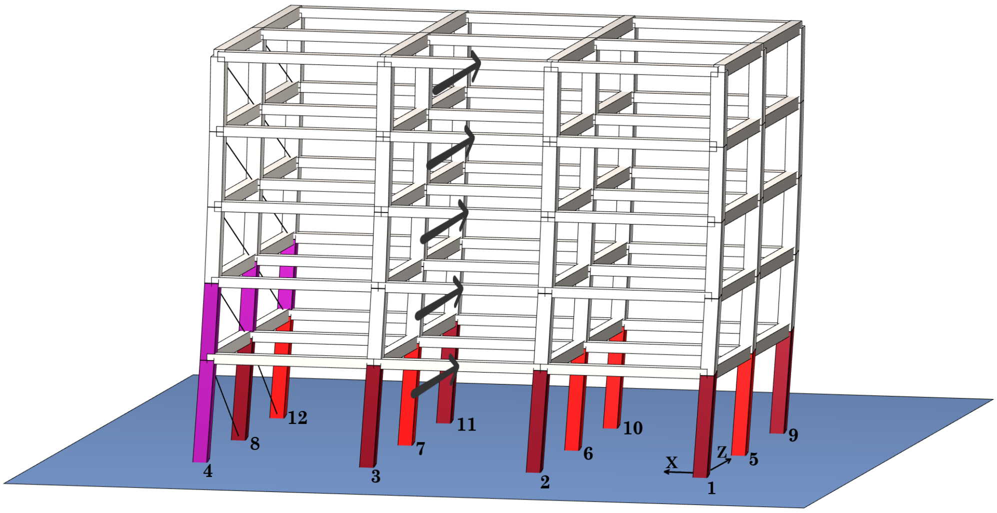



Figure 5.104: *Eccentric* structure - Optimal configuration (symmetric arrangement): Deformed shape (pushover along Z)

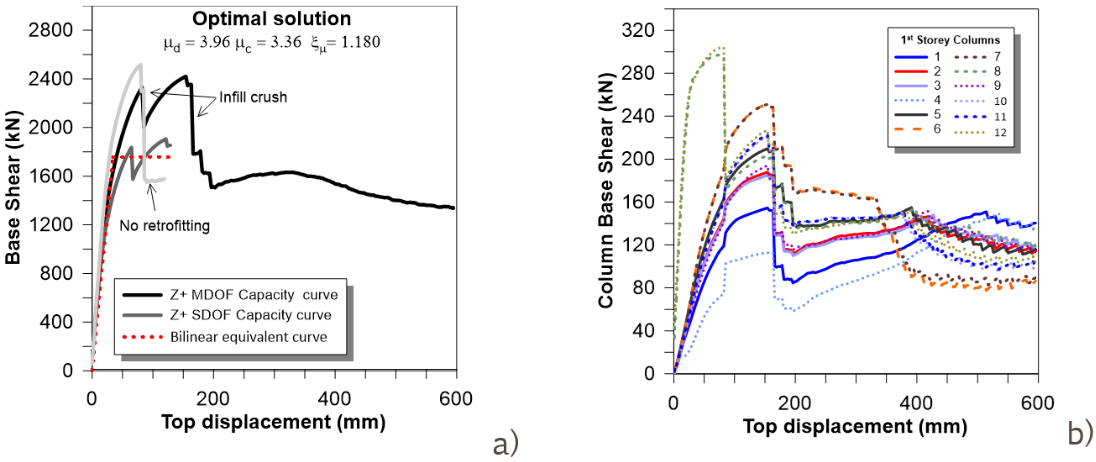



Figure 5.105: *Eccentric* structure - Optimal configuration (symmetric arrangement): (a) Overall pushover capacity curve along Z (b) First storey columns capacity curve

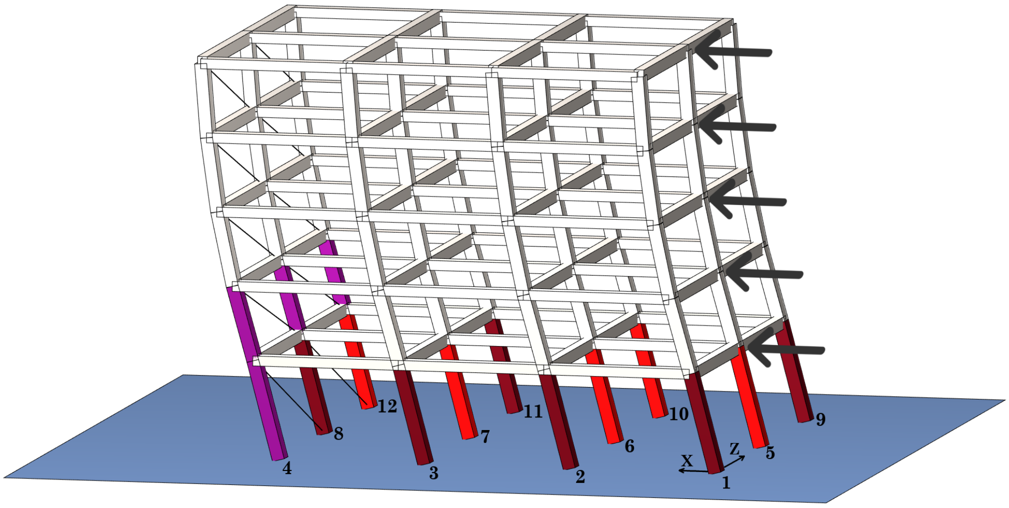



Figure 5.106: *Eccentric* structure - Optimal configuration (symmetric arrangement): Deformed shape (pushover along X)

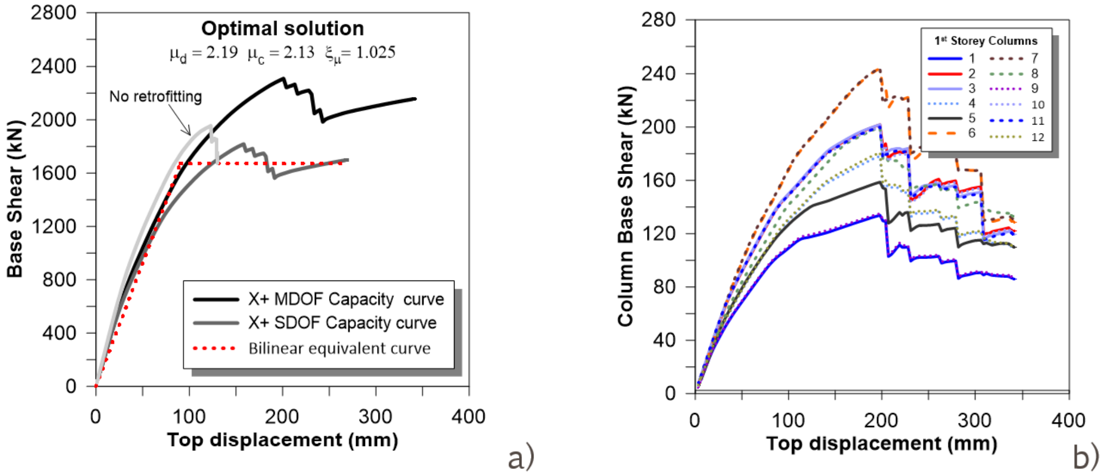



Figure 5.107: *Eccentric* structure - Optimal configuration (symmetric arrangement): (a) Overall pushover capacity curve along X (b) First storey columns capacity curve

# Chapter 6

# Conclusions

This thesis is concerned with the optimization of steel jacketing capable of selecting the cheapest retrofit arrangement and design for existing vulnerable reinforced concrete frame structure.

The method is associated with the adoption of nonlinear static analysis (pushover) as assessment procedure, in the framework of the N2 method. The optimization strategy used a genetic algorithm to minimize retrofitting costs operating on position of reinforced columns (topological optimization) and the spacing of battens.

The structural analyses were performed automatically by connecting an parametric fiber section model of the structure realized with OpenSees.

The feasibility of generated retrofitting solutions was controlled by the ductility capacity/demand ratio ($\xi_\mu = \mu_c/\mu_d$). Shear verification of the columns are accomplished using the model proposed by Biskinis for elements under cyclic loads. The procedure was tested on different configuration of a 5-storey reinforced concrete structure.

From the obtained results, it can be concluded that the proposed optimisation framework can effectively reduce RC building retrofitting and downtime costs controlling safety levels.

From the obtained results, it can be concluded that the proposed optimization framework can effectively reduce RC building retrofitting and downtime costs controlling safety levels above a specified value.

The framework proposed, fine-tuning the parameters that rules the genetic algorithm process, is also efficient in case of shear-sensitive structure.

The cost minimization correlated with a reduction of the amount of steel-jacketing reinforcement is not directly associated with a decrease of safety levels, but on the contrary, the optimization allows discarding ineffective retrofitting solutions for which higher costs are connected with lower safety.

The current approach has been tested on simplistic frame structures, however, for larger RC structures having a significant number of columns, it expected to get noticeable advantages in terms of economical and downtime costs.

The framework is shown to be effective even increasing the degree of the complexity of the structure, however, in the case of regular structural configurations, of structures presenting some symmetries, the optimization can be carried out for a reduced number of directions of action of the lateral force profile to reduce computational effort. Retrofitting along with directions not considered in the optimization can be designed based on simple suppositions of extension of the optimization results which are eventually verified.

This method could be an efficient support to the designer for choosing the cost-effective configuration of the intervention who eventually will have the final decision based on his engineering judgment.

The massive usage of this type of algorithm could increase the effectiveness of retrofitting design reducing the waste of private and public capitals, enhancing the safety of the building heritage.

Further research and case study testing is undoubtedly needed to address, among the other aspects, the development of algorithm that performs multi-directional analysis to assess the structural optimization for irregular structures.

More studies should be done in order to formulate a more comprehensive objective function that could contemplate multiple retrofitting technique. This type of algorithm could be more significant for the retrofitting of existing masonry structures.

Finally, as part of the future developments of the present work of thesis, it could be interesting to extend this metodology to the minimisation of the structure's lifetime earthquake-related repairing costs, with specific referral to the expected annual loss.

# Chapter 7

# Appendix

## 7.1 MainCode.m

```
clear all
close all
clc

% Type of structure to collect the options
opts = optimoptions('ga');

global COSTO_OPT
global COSTO_MAX
global DISTRIBUTION
global DUTT_R
global DUTT_D
global D_max
global POPOLATION
global ALLREADY

% from gen_analysis (generated at the end of each generation)
global POP
global STATES
global MINS

% Parameters for the functions
global funcopts
global model

% Preliminaryy analysis
ALLREADY = 0;
```

```
%%%
%%% Input
%%%

%%% Structure
% Number of bays and storeys
% (max 9 for node's name troubles)
model.xbay = 3;
model.zbay = 2;
model.storey = 5;

% Dimension of the structure
% Bay width
model.width1 = 6000;
% Bay lenght
model.length1 = 6000;
% Bay height
model.height1 = 4000;
model.height2 = 3000;
% Infills (see "modelcreator" to verify the position)
model.yesno_infills = 1;
model.softstory = 0;


% Parameters
% Pushover maximum displacement
model.maxU = 300;
% Analysis step of the pushover
model.dU = 5;
% Degree of freedom of the pushover (1=>X, 2=>Y, 3=>Z)
model.dof = 3;
% Force distribution
model.H = 1;




% Number of columns where the retrofitting could
% be placed
n_col = 3*4*2; % 3 along z, 4 along x, 2 storeys

% Maximum and minimum of battens spacings (mm)
funcopts.battens.max_step = 300;
funcopts.battens.min_step = 150;

% Step variation of battens spacings (mm)
```

```
funcopts.battens.step_analysis = 50;

% Population size
opts.PopulationSize = 80;

% Number of generations
opts.MaxGenerations = 20;

% Display option
opts.Display = 'iter';

%%%
%%% End of inputs
%%%


% Log file
diary log.out
model.output{1} = 'log.out';

%
% Initial operations
%

% numb. of cases of battens step
funcopts.battens.numb_cases = ...
((funcopts.battens.max_step-funcopts.battens.min_step)/...
funcopts.battens.step_analysis)+1;

% Initial checks for battens parameters
%   (if the number of cases is an integer, if max > min)
if funcopts.battens.max_step < funcopts.battens.min_step
    msgbox('Battens max < Battens min',...
    'GA:creationfunction','error');
end
if mod(funcopts.battens.numb_cases,1) ~= 0
    msgbox('Number of cases of battens step not an integer', ...
    'GA:creationfunction','error');
end


%%% Design vector upper limit
x_max = ones(1,n_col+1);
x_max(1) = funcopts.battens.max_step;

% Design vector lower limit
```

```
x_min = zeros(1,n_col+1);
x_min(1) = funcopts.battens.min_step;

% Maximum cost of intervention vector
x_cmax = x_max;
x_cmax(1) = funcopts.battens.min_step;



%%% GA Parameters
% Popolation type
    opts.PopulationType='custom';
% Creation
    opts.CreationFcn = @gacreation;
% Crossover
    opts.CrossoverFcn = @gacrossover;
% Mutation
    opts.MutationFcn = @gamutation;
    % Mutation rate [0,1]
    funcopts.mutation.rate_pos = 0.1;
    funcopts.mutation.rate_step = 0.1;
% Elitism
    opts.EliteCount = 4;
% Function for the analysis of each generation
    opts.OutputFcn = @gaanalysis;

% Initial population
    dim_init = 40;
    x_init = zeros(dim_init,n_col+1);

    for i = 1:dim_init
        x_init(i,:) = randomDVGenerator(n_col+1,90);
    end
    opts.InitialPopulation = x_init;

%
% Genetic algorithm analysis
%


% Start-up analysis - parameters from the model creation
%    and modal analysis
if StartUpFunction(x_min)
    % Maximum cost
    COSTO_MAX = CostFunction(x_cmax);
```

```
    ALLREADY = 1;

    fprintf('Genetic algorithm analysis\n')

    % Control random number generator for reproducibility
    % rng default

    tic
    [X,res,exitflag,output,final_population,final_scores]= ...
    ga(@CostFunction,length(x_max),[],[],[],[],[],[],[],[],opts);
else
    error('Start-up analysis failed!');
end





%
% Final analysis
%

% Time analyis
time_tictoc = toc/3600;
fprintf('Time to perform the analysis: %.3f hours',...
    time_tictoc);

% Minimum solution analysis
[mins, min_ass] = finalAnalysis(COSTO_OPT,DISTRIBUTION);

% Log file
diary off

% Raw datas saving
save rawdata.mat
model.output{end+1} = 'rawdata.mat';

% Curves
curve

% Structure 3D
for i = 1:size(min_ass,2)
    structure3D(min_ass(3:end-2,i),min_ass(2,i),i)
end
```

## 7.2 CostFunction.m

```
 %
% Summary: Determine the cost of the retrofitting work
% Parameters: Design vector of battens step and position
% Return: Cost of retrofitting work in euro
% Author: Antonio Pio Sberna (fork of code by Marzia Malavisi)
%        antoniopio DOT sberna AT studenti DOT polito DOT it

function cost = CostFunction(X)

global COSTO_OPT
global COSTO_MAX
global DISTRIBUTION
global DUTT_R
global DUTT_D
global POPOLATION
global ALLREADY

global model
global V_Rd
global V_Ed
global N_Ed
global N
global V_inf
global R_abs
global D_abs


INPUT = (X);

if ALLREADY
    n_ind = size(COSTO_OPT,2)+1;
    fprintf('Individuo %d\n',n_ind)
else
   fprintf('Analisi costo massimo');
end

% GA input variables
n_col = sum(X(2:end)); % numb. of retrofitted columns

disp(X)



%%%
```

```
%%% Structural analysis
%%%

% Input file for Opensees
input_file = fopen('Input.txt','w');
fprintf(input_file,'%f\n',X);
fclose(input_file);

% Opening opensees file
ModelCreator(X,0,0);
!OpenSees.exe "Frame_analysis".tcl

% Reading Opensees outputs
reaction_file = fopen('R.out','r');
A = fscanf(reaction_file,'%f',...
[model.colum.numb_fundation,Inf]);
fclose(reaction_file);

displ_file = fopen('D.out','r');
formatSpec = '%f ';
B = textscan(displ_file,formatSpec);
fclose(displ_file);

shears = fopen('V.out','r');
E = fscanf(shears,'%f',[(length(X)-1)*12,Inf]);
fclose(shears);

if model.yesno_infills
    axfo_infill = fopen('N.out','r');
    N = fscanf(axfo_infill,'%f',...
    [(size(model.infills,1)),Inf]);
    fclose(axfo_infill);
end

% R = sum(cell2mat(A),2);
R = sum(A,1);
R_abs = abs(R);

D = cell2mat(B(1));
D_abs = abs(D);


% Import columns lenght
L = zeros(length(X)-1,1);
for i = 1:length(X)-1
    if i <= model.colum.numb_fundation % first floor columns
```

```
        L(i) = model.height1;
    else
        L(i) = model.height2;
    end
end




%%%
%%% Analysis output for shear verification
%%%

% Shear and compression values position into the file
pos_shear = 2;
pos_compr = 1;
% Variable allocation
V_Ed = zeros(length(X)-1,size(E,2));
N_Ed = zeros(length(X)-1,size(E,2));
% Picking of variables
V_Ed = abs(E(pos_shear,:));
N_Ed = abs(E(pos_compr,:));
for i = 1:length(X)-2
    V_Ed = [V_Ed;abs(E(i*12+pos_shear,:))];
    N_Ed = [N_Ed;abs(E(i*12+pos_compr,:))];
end


if model.yesno_infills
    % Shear component of the infill compression
    % Friction coefficient
    mhu = 0.7;

    V_inf = zeros(size(V_Ed));
    for i = 1:size(model.infills,1)
        % Position into model.nodes of infills nodes
        pos = find(model.nodes(:,1)==model.infills(i,2));
        % if the infills is defined along z
        if model.infills(i,5) == 1 && pos <= size(V_Ed,1)
            % for all load step
            for j = 1:size(V_Ed,2)
              % Di Trapani & Malavisi (2018)
            V_inf(pos,j) = N(i,j)*(cos(model.infills(i,4))-...
            mhu*sin(model.infills(i,4)));
            end
        end
```

```
    end
    V_Ed = abs(V_Ed) + abs(V_inf);
end




%%%
%%% Volumes analysis
%%%

% Input which do not change
gamma = 7850; %[kg/m^3] steel density

n_col_retr_1f = sum(X(2:model.colum.numb_fundation+1));
n_col_retr_2f = sum(X(model.colum.numb_fundation+2:end));


% Dimensions of columns
col_dim = zeros(2,size(X,2)-1)+500;


% Steel angles volume
V_mont = (n_col_retr_1f*model.height1+n_col_retr_2f*...
model.height2)*8*model.angular.height*model.angular.thickness;

% Steel battens volume
V_battens = 0;
for i = 1:size(col_dim,2) % for each element of the array
    if X(i+1) == 1          % if it is a retrofitting system
    V_battens = V_battens + floor(L(i)/X(1))*...
    (col_dim(1,i)+col_dim(2,i))*2*model.battens.height*...
    model.battens.thickness;
    end
end

% Total volume
V_tot = V_battens + V_mont;      % [mm^3]

% Total weight
W_tot = (V_tot*1e-9)*gamma;     % [kg]



% Check displacements
% fprintf("d_max,analisi = %.3f \n",D_abs(end))
```

```
% Shear verification of columns
V_Rd =zeros(size(V_Ed));
for j = 1:size(V_Ed,2)
 for i = 1:size(V_Ed,1)
 %ShearStrengthBiskinis(b, h, c, L, N, s, n_s, Phi_s, n_l,
 %                      Phi_l, flag_batt)
  V_Rd(i,j) = ShearStrengthBiskinis(model.section.columns.height(i), ...
  model.section.columns.width(i),model.section.columns.concr_cover(i),...
  L(i), N_Ed(i,j), model.stirrups.step(i), model.stirrups.branch(i),...
  model.stirrups.diam(i), model.long_bar.numb(i), ...
  model.long_bar.diam(i), X(i+1), X(1));
 end
end

% Cutoff capacity curve if the shear strength test is not verified
% Shear collapse flag
flag = 0;
for j = 1:size(V_Ed,2)
    for i = 1:size(V_Ed,1)
        if V_Rd(i,j) < V_Ed(i,j)
            flag = j-1;
            break
        end
    end
    if flag > 0
        break
    end
end

% Capacity curve cut off
if flag ~= 0 && flag < length(R_abs)
    R_abs = R_abs(1:flag);
    D_abs = D_abs(1:flag);
    % Indicazione a video
    fprintf('Rottura a taglio della colonna %d,...
    a %d mm \n',i, flag*5)
     if X(i+1) == 1
    fprintf('ATTENZIONE! Retr. column collapse!')
    end
end



% Check if the model verifies limit conditions
```

```
check_out = duttility_check(R_abs , D_abs);

%%%
%%% Cost analysis
%%%


Fixed_cost = 2000; %[euro] fixed cost per columns
p_unit = 4.5; %[euro/kg] cost of steel


% Penalty function
if check_out == 0
    if ALLREADY
        Cmax = COSTO_MAX;
        penalty_func = Cmax*(DUTT_R(end)/DUTT_D(end))^3;
    else
        errordlg('Soluzione con costo massimo non verificata',...
        'Errore inizializzazione');
    end
else
    penalty_func = 0;
end


cost = Fixed_cost*n_col+W_tot*p_unit+penalty_func;



% fine dell'individuo
if check_out==0
    fprintf("Non Verificato! %.3f x 1000 Euro \n\n\n",cost/fatt)
else
    fprintf("Verificato! %.3f x 1000 Euro\n\n\n",cost/fatt)
end

end
```

## 7.3 DuctilityCheck.m

```
function verification = duttility_check(F, D)

global DUTT_R
global DUTT_D
global D_max
global model
global DISTRIBUTION
global ALLREADY

% Spectrum parameters --> COSENZA (Vn = 100)
ag = 0.359; %[g]
F0 = 2.463;
Tb = 0.179; %[s]
Tc = 0.576; %[s]
Td = 3.037; %[s]
S = 1.169;
eta = 1;


% Peak force
[Fmax, idx_max] = max(F);

F_MDOF = []; %Forces [N]
D_MDOF = []; %Displacements [mm]


% End of capacity curve before 85% of the peak force
for i=1:length(F)
    if i<=idx_max || F(i)>Fmax *0.85
      F_MDOF = [F_MDOF, F(i)];
      D_MDOF = [D_MDOF, D(i)];
    else
        break
    end
end


% From MDOF to SDOF
F_SDOF = F_MDOF/model.modal_analysis.modalcoeff;
D_SDOF = D_MDOF/model.modal_analysis.modalcoeff;
```

```
% Equivalent bilinear curve
Fbu = max(F_SDOF);
du = D_SDOF(end);
start_point = Fbu*0.6;
[val, index] = min(abs(F_SDOF-start_point));
dy = D_SDOF(index);
m = start_point/dy; % line slope [N/mm]


spost = dy; %[mm]
Fy = start_point; %[N]

area = trapz(D_SDOF,F_SDOF);
area_bili = (spost*Fy/2)+(du-spost)*Fy;
lunghezza = du/0.001;
spost = D_SDOF(1);


% Area equivalence
delta = 0.001;
for j=1:du/0.001
    if area_bili<area
        spost = spost+delta;
        Fy = m*spost;
        area_bili = (spost*Fy/2)+(du-spost)*Fy;
    else
        break;
    end
end

dy = spost;


% Ductility of the structure
mu_d = du/dy;
% Equivalent stiffness
k = Fy/dy; % [N/mm]
% First structural period
T = 2*3.141592*sqrt(model.modal_analysis.modalmass/k); % [s]


% Elastic spectrum
if T<Tb
    Se_T = ag*S*eta*F0*((T/Tb)+((1/F0/eta)*(1-T/Tb)));
elseif (T>Tb) && (T<Tc)
    Se_T = ag*S*eta*F0;
```

```
elseif (T>Tc) && (T<Td)
    Se_T = ag*S*eta*F0*(Tc/T);
elseif T>Td
    Se_T = ag*S*eta*F0*(Tc*Td/T^2);
end

% Reduction facto
q = Se_T*model.modal_analysis.modalmass*10000/Fy;

% Ductility demand
if T<Tc
    mu_r = (q-1)*(Tc/T)+1;
else
    mu_r = q;
end

if ALLREADY
    x = [0, dy, du];
    y = [0, Fy, Fy];

    % Salvataggio dati
    DUTT_R = [DUTT_R, mu_r];
    DUTT_D = [DUTT_D, mu_d];
    D_max = [D_max,D_MDOF(end)];
end


global SDOF
global MDOF
global equiv
if ALLREADY
    MDOF = [D_MDOF',F_MDOF'];
    SDOF = [D_SDOF',F_SDOF'];
    equiv = [0, dy, du;0, Fy, Fy]';
end

% Ductility verification
if mu_r<mu_d
    verification = 1;
else
    verification = 0;
end

end
```

## 7.4 ModelCreator.m

```
%
% Summary: Create the fiber section model for push over analysis
% Parameters:
%  X -> Design vector with the first position the battens step
%  firstAnalysis -> =1 it's the first analysis, generate also
%    node and  analysis parameters
%  modalAnalysis -> =0 no modal analysis parameters
%                   =1 modal analysis par.
% Return: Four .tcl file with nodes, elements, loads and pushover
%       definitions and parameters
% Author: Antonio Pio Sberna
%           antoniopio DOT sberna AT studenti DOT polito DOT it


function mcOutput = ModelCreator(X,firstAnalysis,modalAnalysis)
mcOutput = 0;

global model

% Elements
% Number of integration points along length of element
np = 5;
% Id sezione per colonne non confinate
secIDNR = 10;
% Id sezione per colonne confinate
secIDR = 20;
% Id materiale per gli infills
infillSecTag = 5;
% Id sezione del trave
secID_travi = 40;
% Type of element
eleType2 = 'nonlinearBeamColumn';

% Numb. of nodes for each floor
n_piano = (model.xbay+1)*(model.zbay+1);

% Numb. of fundation nodes
model.colum.numb_fundation = (model.xbay+1)*(model.zbay+1);


% Control of the design vector
if (length(X)-1) ~= (n_piano*2)
  msgbox('The dimension of design vector \n is not compatible ...
  with the structure', 'GA:creationfunction','error');
```

```
end


if firstAnalysis
%% Nodes

model.nodes = zeros(n_piano*(model.storey+1),4);
temp = 1;
for j=0:model.storey
  for k=0:model.zbay
    for i=0:model.xbay
      node_name = 1011+j*100+i*10+k;
      if j == 0
        model.nodes(temp,:) = [node_name,model.width1*i,0,...
           model.length1*k];
      else
          model.nodes(temp,:) =
          [node_name-1000,model.width1*i,...
            model.height1+(j-1)*model.height2,model.length1*k];

      end
      temp = temp+1;
    end
  end
end

nodeFile = fopen('nodeGeometry.tcl','w');


for i=1:size(model.nodes,1)
   fprintf(nodeFile,'node %i %f %f %f \n',model.nodes(i,1),...
   model.nodes(i,2),model.nodes(i,3),model.nodes(i,4));
end



% Degree of freedom on foundation
for i=1:n_piano
    fprintf(nodeFile,'fix %i 1 1 1 1 1 1 \n',model.nodes(i,1));
end


% Diaphragmatic behaviour
for i=1:model.storey
  fprintf(nodeFile,'rigidDiaphragm 2 ');
  central_node = n_piano*i+1+ceil(model.zbay/2)*(model.xbay+1)+...
```

```
    ceil(model.xbay/2);
    fprintf(nodeFile,' %i', model.nodes(central_node));

    for j=1:n_piano
        node_name = n_piano*i+j;
        if model.nodes(node_name)~= model.nodes(central_node)
            fprintf(nodeFile,' %i',model.nodes(node_name,1));
        end
    end
    fprintf(nodeFile,'\n');
end
fclose(nodeFile);

end


%% Elements

elementFile = fopen('elementGeometry.tcl','w');

% Transformation
fprintf(elementFile,'geomTransf Linear 1 1 0 0\n');



% Columns
model.columns = zeros(size(model.nodes,1)-n_piano,4);
for i=1:size(model.nodes,1)-n_piano
    if i <= (length(X)-1) && X(i+1)== 1
        sec_prop = secIDR;
        model.columns(i,4) = 1;
    else
        sec_prop = secIDNR;
        model.columns(i,4) = 0;
    end
    fprintf(elementFile,'element %s %i%i %i %i %i %i 1 \n',...
       eleType2,model.nodes(i,1), model.nodes(i+n_piano,1),...
       model.nodes(i,1),model.nodes(i+n_piano,1),np,sec_prop);

    % Saving informations
    model.columns(i,1) = str2num(sprintf('%i%i', ...
       model.nodes(i,1),model.nodes(i+n_piano,1)));
    model.columns(i,2) = model.nodes(i,1);
    model.columns(i,3) = model.nodes(i+n_piano,1);
end
```

```
% Beams
n_beams = model.storey*((model.zbay+1)*model.xbay+...
   (model.xbay+1)*model.zbay);
model.beams = zeros(n_beams,4);
% Coordinate transformation
traviX = 2;
traviZ = 3;
fprintf(elementFile,...
  'geomTransf Linear %i 0 1 0\ngeomTransf Linear %i 1 0 0\n',...
   traviX,traviZ);

temp = 1;

for j=1:model.storey
  % Along X
  for k=1:model.zbay+1
    for i=1:model.xbay
      node_i = j*n_piano+i+(k-1)*(model.xbay+1);
      node_j = node_i+1;
      fprintf(elementFile,'element %s %i%i %i %i %i %i %i\n', ...
         eleType2,model.nodes(node_i), model.nodes(node_j,1), ...
         model.nodes(node_i,1), model.nodes(node_j,1), np,...
         secID_travi, traviX);
      % Salvataggio dati
      model.beams(temp,1) = str2num(sprintf(...
          '%i%i',model.nodes(node_i),model.nodes(node_j,1)));
      model.beams(temp,2) = model.nodes(node_i,1);
      model.beams(temp,3) = model.nodes(node_j,1);
      model.beams(temp,4) = 'x';
      temp = temp+1;
    end
  end
  % Along Z
  for i = 1:model.xbay+1
    for k =1:model.zbay
      node_i = j*n_piano+k+(i-1)*(model.zbay);
      node_j = node_i + model.xbay+1;
      fprintf(elementFile,'element %s %i%i %i %i %i %i %i\n',...
        eleType2,model.nodes(node_i), model.nodes(node_j,1),...
        model.nodes(node_i,1),model.nodes(node_j,1), np,...
        secID_travi, traviZ);
      % Salvataggio dati
      model.beams(temp,1) = str2num(sprintf(...
          '%i%i',model.nodes(node_i), model.nodes(node_j,1)));
```

```
        model.beams(temp,2) = model.nodes(node_i,1);
        model.beams(temp,3) = model.nodes(node_j,1);
        model.beams(temp,4) = 'z';
        temp = temp+1;
      end
    end

end

% Infills
% initial tag
init_tag = 50;

if model.yesno_infills
    model.infills = zeros((model.storey-model.softstory)*model.zbay,4);

% third columns:
% 0 if it is along x
% 1 if it is along z


% Nodes
if model.softstory
    iniz = 2;
else
    iniz = 1;
end

temp=1 ;
for k = iniz:model.storey
    for i=1:model.zbay
        % Right frame (nodes XendX)
        n_elem = init_tag+temp;
        node_i = (i+1)*(model.xbay+1)+(k-1)*n_piano;
        node_j = node_i +n_piano - (model.xbay+1);

        %%% Saving
        model.infills(temp,1) = n_elem;
        model.infills(temp,2) = node_i;
        model.infills(temp,3) = node_j;
        temp = temp+1;

        if ~model.infills_asym
            % Left frame (serie X1X)
            n_elem = init_tag+temp;
            node_i = ((i*(model.xbay+1))+1)+(k-1)*n_piano;
```

```
            node_j = node_i +n_piano - (model.xbay+1);

            %%% Saving
            model.infills(temp,1) = n_elem;
            model.infills(temp,2) = node_i;
            model.infills(temp,3) = node_j;
            temp = temp+1;
        end
    end
end



% Elements
temp = 1;
for i =1:size(model.infills,1)
  fprintf(elementFile,'element trussSection %i %i %i %i\n', ...
  model.infills(temp,1), model.nodes(model.infills(i,2),1),...
  model.nodes(model.infills(i,3),1), infillSecTag);

  %%% Infills inclination
  if model.nodes(model.infills(i,2),3) == ...
  model.nodes(model.infills(i,3),3)
      adiac = abs(model.nodes(model.infills(i,3),2) -...
         model.nodes(model.infills(i,2),2));
      model.infills(temp,5) = 0; % along X
  else
      adiac = abs(model.nodes(model.infills(i,3),4) -...
          model.nodes(model.infills(i,2),4));
      model.infills(temp,5) = 1; % along Z
  end

  opp = abs(model.nodes(model.infills(i,3),3) - ...
      model.nodes(model.infills(i,2),3));
  theta = atan(opp/adiac);
  % Saving
  model.infills(temp,4) = theta;
  model.infills(temp,2) = model.nodes(model.infills(temp,2),1);
  model.infills(temp,3) = model.nodes(model.infills(temp,3),1);
  temp = temp+1;
end
end


fclose(elementFile);
```

```

if firstAnalysis
%% Loads

% floor weight
pmq = 0.01;
% Floor area
AP = model.width1*model.xbay+model.length1*model.zbay;
WP = pmq*AP;
g = 9800;

loadFile = fopen('loadGeometry.tcl','w');

% Loads (for static analysis)
fprintf(loadFile,'pattern Plain 1 "Linear" {\n');

cornerLoad = -WP*(model.width1/2*model.length1/2)/AP;
xedgeLoad = -WP*(model.width1*model.length1/2)/AP;
zedgeLoad = -WP*(model.width1/2*model.length1)/AP;
innerLoad = -WP*(model.width1*model.length1)/AP;

for j = 1:model.storey
  for temp = 0:1
    nodeName = j*n_piano+1+temp*(n_piano-(model.xbay+1));
    fprintf(loadFile,'load %i 0.0 %f 0.0 0.0 0.0 0.0\n',...
    model.nodes(nodeName),cornerLoad);
    for i = 1:model.xbay-1
        nodeName = nodeName+1;
        fprintf(loadFile,'load %i 0.0 %f 0.0 0.0 0.0 0.0\n',...
        model.nodes(nodeName),xedgeLoad);
    end
    nodeName = nodeName+1;
    fprintf(loadFile,'load %i 0.0 %f 0.0 0.0 0.0 0.0\n',...
    model.nodes(nodeName),cornerLoad);
  end

  for temp = 1:model.zbay-1
    nodeName = j*n_piano+1+model.xbay+1+(temp-1)*(model.xbay+1);
    fprintf(loadFile,'load %i 0.0 %f 0.0 0.0 0.0 0.0\n',...
    model.nodes(nodeName),zedgeLoad);
    for i = 1:model.xbay-1
        nodeName = nodeName+1;
        fprintf(loadFile,'load %i 0.0 %f 0.0 0.0 0.0 0.0\n',...
        model.nodes(nodeName),innerLoad);
    end
    nodeName = nodeName+1;
```

```
    fprintf(loadFile,'load %i 0.0 %f 0.0 0.0 0.0 0.0\n',...
    model.nodes(nodeName),zedgeLoad);
  end

end
fprintf(loadFile,'}\n');


% Masses (for modal analysis)
if modalAnalysis == 1
  cornerMass = cornerLoad/-g;
  xedgeMass = xedgeLoad/-g;
  zedgeMass = zedgeLoad/-g;
  innerMass = innerLoad/-g;

  for j = 1:model.storey
    mass = 0;
    % primo e ultimo telaio nel piano xy
    for temp = 0:1
      nodeName = j*n_piano+1+temp*(n_piano-(model.xbay+1));
      fprintf(loadFile,'mass %i %f %f %f 0.0 0.0 0.0\n',...
      model.nodes(nodeName),cornerMass,cornerMass,cornerMass);
      mass = mass+cornerMass;
      for i = 1:model.xbay-1
        nodeName = nodeName+1;
        fprintf(loadFile,'mass %i %f %f %f 0.0 0.0 0.0\n',...
        model.nodes(nodeName),xedgeMass,xedgeMass,xedgeMass);
        mass = mass+xedgeMass;
      end
      nodeName = nodeName+1;
      fprintf(loadFile,'mass %i %f %f %f 0.0 0.0 0.0\n',...
      model.nodes(nodeName),cornerMass,cornerMass,cornerMass);
      mass = mass+cornerMass;
    end

    for temp = 1:model.zbay-1
       nodeName = j*n_piano+1+model.xbay+1+(temp-1)*(model.xbay+1);
       fprintf(loadFile,'mass %i %f %f %f 0.0 0.0 0.0\n',...
       model.nodes(nodeName),zedgeMass,zedgeMass,zedgeMass);
       mass = mass+zedgeMass;
       for i = 1:model.xbay-1
         nodeName = nodeName+1;
         fprintf(loadFile,'mass %i %f %f %f 0.0 0.0 0.0\n',...
         model.nodes(nodeName),innerMass,innerMass,innerMass);
         mass = mass+innerMass;
       end
```

```
        nodeName = nodeName+1;
        fprintf(loadFile,'mass %i %f %f %f 0.0 0.0 0.0\n',...
        model.nodes(nodeName),zedgeMass,zedgeMass,zedgeMass);
        mass = mass+zedgeMass;
     end
     model.modal_analysis.masses(j) = mass;
  end
end

fclose(loadFile);
end

if firstAnalysis

%% Push over

pushparFile = fopen('pushparFrame_analysis.tcl','w');


fprintf(pushparFile,'pattern Plain 3 "Linear" { \n');
for  y = 1:model.storey
  for  x = 1:model.xbay-1
    central_node = n_piano*y+1+ceil(model.zbay/2)*(model.xbay+1)+x;
    if model.dof == 3
     fprintf(pushparFile,'load %i 0.0 0.0 %f 0.0 0.0 0.0\n', ...
         model.nodes(central_node),model.H);
    elseif model.dof == 1
     fprintf(pushparFile,'load %i %f 0.0 0.0 0.0 0.0 0.0\n', ...
          model.nodes(central_node),model.H);
    elseif model.dof == 2
     fprintf(pushparFile,'load %i 0.0 %f 0.0 0.0 0.0 0.0\n',...
          model.nodes(central_node),model.H);
    else
      errmsg = sprintf('dof not compatible (1=x 2=y 3=z)');
      errordlg(errmsg,'Fatal error !');
    end
  end
end
model.head_node = central_node;
fprintf(pushparFile,'}\n');

fprintf(pushparFile,...
   'integrator DisplacementControl %i %i %i 1 %i %i\n',...
   model.nodes(model.head_node), model.dof, model.dU, model.dU,...
   model.dU);
```

```
fprintf(pushparFile,'set maxU %i\n', model.maxU);


% Recorders
% Reactions
fprintf(pushparFile,'recorder Node -file R.out -node ');
for i=1:n_piano
   fprintf(pushparFile,'%i ',model.nodes(i));
end
fprintf(pushparFile,'-dof %i reaction \n',model.dof);

% Displacements
fprintf(pushparFile,...
    'recorder Node -file D.out  -node %i -dof %i disp\n',...
    model.nodes(model.head_node),model.dof);


% Nodes displacements (for the deformed shape drawing)
% along Z
fprintf(pushparFile,'recorder Node -file DISPALL_Z.out -node ');
for temp = 1:size(model.nodes,1)
    fprintf(pushparFile,'%i ', model.nodes(temp,1));
end
fprintf(pushparFile,'-dof %i disp\n', 3);

% along X
fprintf(pushparFile,'recorder Node -file DISPALL_X.out -node ');
for temp = 1:size(model.nodes,1)
    fprintf(pushparFile,'%i ', model.nodes(temp,1));
end
fprintf(pushparFile,'-dof %i disp\n', 1);



% Columns shear
fprintf(pushparFile,'recorder Element -file V.out -ele ');
for piani_rinf = 0:1
   for i=1:n_piano
       node_i = piani_rinf*n_piano + i;
       node_j = node_i + n_piano;
       fprintf(pushparFile,'%i%i ', model.nodes(node_i),...
           model.nodes(node_j));
   end
end
fprintf(pushparFile,'localForce\n');
```

```
% Infills compressive value
if model.yesno_infills
    fprintf(pushparFile,'recorder Element -file N.out -ele ');
    for i = 1:size(model.infills,1)
        fprintf(pushparFile,'%i ', model.infills(i,1));
    end
    fprintf(pushparFile,'localForce\n');
end



% Pushover parameters
fprintf(pushparFile,'set nodo %i\n',model.nodes(model.head_node));
fprintf(pushparFile,'set dof %i\n',model.dof);

fclose(pushparFile);

end




%% Modal analysis
if modalAnalysis == 1

    % number of modes analysed
    %numModes = 3;

    modalFile = fopen('parameters_modal.tcl','w');

    % Recorder parameters
    fprintf(modalFile,'recorder Node -file modal_output.out -node ');
    for j = 1:model.storey
        node_name=size(model.nodes,1)-j*n_piano+1;
        fprintf(modalFile,'%i ',model.nodes(node_name));
    end
    fprintf(modalFile,'-dof 1 \"eigen 1\"\n');

    fclose(modalFile);
end



%% Output della funzione
mcOutput = 1;

end
```

## 7.5 StartUpFunction.m

```
%
% Summary: Start-up function for GA analysis of  optimization
%       of steel battens retrofitting
% Parameters: Design vector containing the battens spacing
%                         and position of retrofitted columns
% Return:
%       3) height columns section [mm]
%       4) width columns section [mm]
%       5) concrete cover width [mm]
%       6) stirrups step [mm]
%       7) numb. of stirrups branch [-]
%       8) stirrups diameters [mm]
%       9) numb. longitudinal bars [-]
%       10) diameter longitudinal bars [mm]
%       11) height battens [mm]
%       12) width battens [mm]
%       13) height angular [mm]
%       14) wiidth angular [mm]
%       15) concrete compressive strenght [MPa]
%       16) bars yielding stress [MPa]
%       17) numb. of columns on the foundation [-]

% Author: Antonio Pio Sberna
%           antoniopio DOT sberna AT studenti DOT polito DOT it


function SUFOutput = StartUpFunction(X)
SUFOutput = 0;

global model
fprintf('Start-up analysis ')

%% Analysis

% Flag up
flagFile = fopen('StartUpFlag.txt','w');
fprintf(flagFile,'%f',1);
fclose(flagFile);

% Design vector for opensees
input_file = fopen('Input.txt','w');
fprintf(input_file,'%f\n',X);
fclose(input_file);
```

```
% Opening opensees file
modelflag = ModelCreator(X,1,1);
if modelflag
!OpenSees.exe "modal".tcl
else
    msgbox('Fatal error during the model creation',...
    'Startup function','error');
end

% Flag down
flagFile = fopen('StartUpFlag.txt','w');
fprintf(flagFile,'%f',0);
fclose(flagFile);

% Section parameters
startUpInput = fopen('StartUpInput.txt','r');
OSOutput = fscanf(startUpInput,'%f');
fclose(startUpInput);

% Constant HP for all columns
for i = 1:size(model.columns,1)
    % Section
    % height columns section [mm]
    model.section.columns.height(i) = OSOutput(1);
    % width columns section [mm]
    model.section.columns.width(i) = OSOutput(2);
    % concrete cover width [mm]
    model.section.columns.concr_cover(i) = OSOutput(3);


    % Stirrups
    % stirrups step [mm]
    model.stirrups.step(i) = OSOutput(4);
    % numb. of stirrups branch [-]
    model.stirrups.branch(i) = OSOutput(5);
    % stirrups diameters [mm]
    model.stirrups.diam(i) = OSOutput(6);

    % Longitudinal bars
    % numb. longitudinal bars [-]
    model.long_bar.numb(i) = OSOutput(7);
    % diameter longitudinal bars [mm]
    model.long_bar.diam(i) = OSOutput(8);
end
```

```
% Beams
model.section.beams.height = OSOutput(16);
model.section.beams.width = OSOutput(17);

% Battens
% battens height[mm]
model.battens.height = OSOutput(9);
% battens thickness [mm]
model.battens.thickness = OSOutput(10);

% Angles
% angles height [mm]
model.angular.height = OSOutput(11);
% angles thickness [mm]
model.angular.thickness = OSOutput(12);

% Materials
% concrete compressive strenght [MPa]
model.material.concr_strenght = OSOutput(13);
% bars yielding stress [MPa]
model.material.bar_yield = OSOutput(14);
% battens yielding stress [MPa]
model.material.battens_yield = OSOutput(15);

% Modal analysis (parameters for N2 analysis)
% Import eigenvector
startUpInput = fopen('modal_output.out','r');
eigenVector = fscanf(startUpInput,'%f',...
    [length(model.modal_analysis.masses),Inf]);
fclose(startUpInput);

% Normalization of the eigenvector
eigenVector = eigenVector/eigenVector(1);

% Modal masses
model.modal_analysis.modalmass = ...
dot(model.modal_analysis.masses,eigenVector);

% Modal participation coefficient
model.modal_analysis.modalcoeff=0;
for i =1:length(model.modal_analysis.masses)
  model.modal_analysis.modalcoeff = ...
     model.modal_analysis.modalcoeff + ...
    (model.modal_analysis.masses(i)*eigenVector(i)^2);
end
model.modal_analysis.modalcoeff = model.modal_analysis.modalmass/...
```

```
    model.modal_analysis.modalcoeff;

SUFOutput = 1;
end
```

## 7.6 ModalAnalysis

```
# Define Geometry
wipe
source Geometry.tcl

set lambda [eigen 1]
source parameters_modal.tcl

integrator LoadControl 0 1 0 0

# Convergence test
test EnergyIncr             1.0e-6    100         0

# Solution algorithm
algorithm Newton

# DOF numberer
numberer RCM

# Constraint handler
constraints Transformation


# System of equations solver
system ProfileSPD

analysis Static
set res [analyze 1]
if {$res < 0} {
    puts "Modal analysis failed"
}
```

## 7.7 ShearStrengthBiskinis.m

```
% Summary: Shear strength Biskinis E. et all (2004)
% Parameters:
%        b          -> height of the section [mm]
%        h          -> width of the section [mm]
%        c          -> concrete cover length [mm]
%        L          -> length of the beam  analysed[mm]
%        N          -> compressive force acting on the beam [mm]
%        s          -> stirrups step [mm]
%        n_s        -> numb. of stirrups branch [-]
%        Phi_s      -> stirrups diameters [mm]
%        n_l        -> longitudinal bars [-]
%        Phi_l      -> diameter longitudinal bars [mm]
%        flag_batt -> 1 if there are the battens [mm]
%        s_b        -> battens step [mm]
%            from model
%        t_b        -> thickness of the battens [mm]
%        b_b        -> height of the battens [mm]
%        f_c        -> concrete compressive strenght [MPa]
%        f_y        -> bar yielding stress [MPa]
%        f_yb       -> battems yielding stress [MPa]
%
% Return: V_tot    -> shear strenght of the column
%
% Author: Antonio Pio Sberna
%            antoniopio DOT sberna AT studenti DOT polito DOT it

function V_tot = ShearStrengthBiskinis(b, h, c, L, N, s, n_s, ...
  Phi_s, n_l, Phi_l, flag_batt, s_b)

global model

% Materials
f_c = model.material.concr_strenght;
f_y = model.material.bar_yield;
f_yb = model.material.battens_yield;

% Young modulus
gamma_el = 1.15;

% Height
d = h-c;
z = 0.9*d;
% Concrete area
```

```
A_c = b*h;

% Shear lenght
L_v = L/2;

% Longitudinal bars
A_sl = n_l*Phi_l^2*3.1415926535/4;
rho_tot = A_sl/A_c;
% Shear reinforcement
A_sw = n_s*Phi_s^2*3.1415926535/4;
rho_sx = A_sw/s/b;

% Ductility contribution
mhu_d = 4;
mhu_d_pl = mhu_d -1;
beta = 1-0.05*min([5,mhu_d_pl]);

% Compressed area height
x = h*min([1,(0.25+0.85*N/A_c/f_c)]);

% Shear strenght
V_1 = (h-x)/2/L_v * min([N,0.55*A_c*f_c]);
V_2 = 0.16*max([0.5,100*rho_tot])*...
         (1-0.16*min([5,L_v/h]))*A_c*sqrt(f_c);
V_w = rho_sx*b*z*f_y;


% Steel-jacketing
% Battens parameters
t_b = model.battens.thickness;
b_b = model.battens.height;

% Circ. NTC18 C.8.7.4.5 (pag 290)
if flag_batt % if there is battens
    V_j = t_b*b_b*f_yb*0.9*d/s_b;
else
    V_j = 0;
end

% Total shear strength
V_tot = (V_1 + beta*(V_2 + V_w))/gamma_el + V_j;

end
```

## 7.8 randomDVGenerator.m

```
%
% Summary: Random design vector generator
% Parameters: Dimention of DV (integer),
%     probability to have a 1 (integer, percentage)
% Return: The design vector
% Author: Antonio Pio Sberna
%         antoniopio DOT sberna AT studenti DOT polito DOT it
%

function [X] = randomDVGenerator(dim,probability)
  global funcopts


  % Variable allocation
  X = zeros(dim,1);

  % Battens spacing (casual)
  X(1) = funcopts.battens.min_step + ...
  funcopts.battens.step_analysis * ...
  /rando(funcopts.battens.numb_cases)-1);

  % Retrofitting system location
  for i = 2:dim
      temp = randi(100);
        if temp <= probability
          X(i) = 1;
      else
          X(i) = 0;
      end

  end

end
```

## 7.9 GASelection.m

```
function parents = selectionstochunif(expectation,nParents,options)
% SELECTIONSTOCHUNIF Choose parents using stochastic universal
% sampling (SUS). PARENTS = SELECTIONSTOCHUNIF(EXPECTATION,
% NPARENTS,OPTIONS) chooses the PARENTS using roulette wheel and
% uniform sampling, based on EXPECTATION and numb of parents NPARENTS.

% Copyright 2003-2015 The MathWorks, Inc.

expectation = expectation(:,1);
wheel = cumsum(expectation) / nParents;

parents = zeros(1,nParents);

% we will step through the wheel in even steps.
stepSize = 1/nParents;

% we will start at a random position less that one full step
position = rand * stepSize;

% a speed optimization. Position is monotonically rising.
lowest = 1;

for i = 1:nParents % for each parent needed,
    for j = lowest:length(wheel) % find the wheel position
        if(position < wheel(j)) % that this step falls in.
            parents(i) = j;
            lowest = j;
            break;
        end
    end
    position = position + stepSize; % take the next step.
end
```

## 7.10 GACrossover.m

```
function xoverKids  = gacrossover(parents,options,GenomeLength,...
  ~,~,thisPopulation)
% CROSSOVERSCATTERED Position independent crossover function.
% XOVERKIDS = CROSSOVERSCATTERED(PARENTS,OPTIONS,GENOMELENGTH,
% FITNESSFCN,SCORES,THISPOPULATION) creates the children
% XOVERKIDS of the population THISPOPULATION using PARENTS.
% Each gene has an equal chance of coming from either parent.
% Copyright 2003-2015 The MathWorks, Inc.
% forked by Antonio Pio Sberna
%       antoniopio DOT sberna AT studenti DOT polito DOT it

% Number of children to produce
nKids = length(parents)/2;
% Allocate space for the kids
xoverKids = zeros(nKids,GenomeLength);

% To move through the parents twice as fast as thekids are
% being produced, a separate index for the parents is needed
index = 1;
% for each kid...
for i=1:nKids
    % get parents
    r1 = parents(index);
    index = index + 1;
    r2 = parents(index);
    index = index + 1;
    % Randomly select half of the genes from each parent
    % This loop may seem like brute force, but it is twice
    % as fast as the vectorized version.
    for j = 1:GenomeLength
        if(rand > 0.5)
            xoverKids(i,j) = thisPopulation(r1,j);
        else
            xoverKids(i,j) = thisPopulation(r2,j);
        end
    end
end
end
```

## 7.11 GAMutation.m

```
%
% The function returns mutationChildren the mutated offspring
% as a matrix  where rows correspond to the children.
% The number of columns of the matrix is Number of variables.

% The mutation for the position of the retrofitting system
% is a uniform, for the step is a adjacent mutation
%   Copyright 2003-2015 The MathWorks, Inc.
%   forked by Antonio Pio Sberna
%                 antoniopio DOT sberna AT studenti DOT polito DOT it

function mutationChildren = gamutation(parents, options,...
  GenomeLength,FitnessFcn, state, thisScore, thisPopulation)
global funcopts

% Control of the mutation rates
if funcopts.mutation.rate_pos <0 || funcopts.mutation.rate_pos>1 ||...
funcopts.mutation.rate_step <0 || funcopts.mutation.rate_step >1
msgbox('Mutation rates must be major of 0 and minor of 1', ...
'GA:mutationChildren','error');
end

% Allocation of the space
mutationChildren = zeros(length(parents),GenomeLength);

for i=1:length(parents)
    child = thisPopulation(parents(i),:);
    % For the battens
    if rand < funcopts.mutation.rate_step
        if rand < 0.5
            child(1) = child(1) + funcopts.battens.step_analysis;
        else
            child(1) = child(1) - funcopts.battens.step_analysis;
        end
    end
    % For the position
    mutationPoints = find(rand(1,length(child)-1) <...
        funcopts.mutation.rate_pos);
    child(mutationPoints+1) = ~child(mutationPoints+1);
    mutationChildren(i,:) = child;
end
end
```

## 7.12 Geometry.tcl

```
wipe

model basic -ndm 3 -ndf 6
source library.tcl

# GEOMETRY

# ################ OPTIMIZATION ##############################
###
###  Import input file
###

set fp [open "Input.txt" r]
set file_data [read $fp]
close $fp

#Input data
set Input_data [split $file_data "\n"]
set s_batt [lindex $Input_data 0]


# ##############################################################
###
###  Definition of the nodes
###

source nodeGeometry.tcl

# ##############################################################
###
###  MATERIALS AND SECTIONS PARAMETERS
###

# ##############################################################
# Set parameters for the number of fiber

set i_col 40
set j_col 4
set i_tra 25
set j_tra 4

set i_inf 1
set j_inf 1
```

```


################################################################
# Geometry of the section
# Columns
set colB 500.
set colH 500.
set cover 35.

# Beams
set beaH 500.
set beaB 400.

# Infills
set wi 1000.
set ti 250.


# #############################################################
# Concrete
set fc 20.     ;# Average strength of concrete


# #############################################################
# Steel
set fy 455.    ;# Yielding stress
set Es 210000. ;# Young's modulus
# Battens steel
set fyb 275.   ;# Yield stress



# #############################################################
# Reinforcement

# Columns
# Longitudinal
set nlx 4.     ;# number of bars along x
set nly 4.     ;# number of bars along y
set Phi_lspi 18.;# diameter of the bars in the corner
set Phi_lpar 18.;# diameter of the bars in the edge
# area of 1 long. bar
set As   [expr pow($Phi_lspi,2)*3.141592/4.0] ;

# Stirrups
set nsx 2.     ;# number of stirrups arms along x
```

```
set nsy 2.      ;# number of stirrups arms along y
set Phis 6.     ;# diameter of stirrups
set ss 180.     ;# stirrups spacing

# Battens
set tb     5.   ;# Concrete thickness
set ab     50.  ;# Concrete height
set ta     5.   ;# Concrete thickness
set la     100. ;# Concrete height


# Beams
set Phi_lbeam 18.;# diameter bars of beams
set Ast [expr (3.141592*pow($Phi_lbeam,2)/4.)] ;# area of 1 bar

# ##############################################################
###
###  DEFINITION OF MATERIALS
###

# ##############################################################
# Concrete

# Concrete confined with stirrups
set id 1
set eps_cc [ConfinedConcreteSR $id $colB $colH $cover $ss $fy $fc
  $nsx $nsy $Phis $nlx $nly $Phi_lspi $Phi_lpar $Phi_lpar]


# Concrete confined with stirrups and battens
set id 6
ConfinedConcreteBattens $id $colB $colH $cover $ss $Phis $nsx $nsy
  $s_batt $tb $ab $la $ta $fc $fy $fyb $eps_cc



# ##############################################################
# Steel

# Rebars steel
#                         tag     fy    E0      b      R0     cR1
CR1
uniaxialMaterial Steel02   3     $fy   $Es    0.01    15.    0.925
0.15
#uniaxialMaterial MinMax   3      33   -min   -0.2 -max   0.2
```

```
# Battens steel
#                           tag     fy    E0      b       R0     cR1
CR1
uniaxialMaterial Steel02    4     $fyb    $Es    0.000     15.    0.925
0.15
#uniaxialMaterial MinMax     4      34     -min -0.075    -max  0.02


# ##############################################################
# Infills

# Homogeneous material for all floors
uniaxialMaterial Concrete02 29 -1.88 -0.0013 -0.857 -0.0073 0.12 0. 0.
uniaxialMaterial MinMax 9 29  -min -0.015 -max 1

# ##############################################################
###
###   DEFINITION OF SECTIONS
###

# ##############################################################
# Columns

set y1          [expr $colH/2.0]
set z1          [expr $colB/2.0]


# Torsion shear material values for all materials

set Gc 25000000
set C250 10

set GJcol [expr $Gc*$C250*$colB*pow($colH,3)]
set GAcol [expr $Gc*$colB*$colH*5/6]

uniaxialMaterial Elastic 50 $GJcol
uniaxialMaterial Elastic 51 $GAcol


# NO RETROFITTING COLUMN
section Fiber 1 {
 # Create the concrete core fibers
 patch rect 1 $i_col $j_col [expr -$y1] [expr -$z1] [expr $y1] [expr $z1]

 # Create the reinforcing fibers (left, middle, right)
```

```
 layer straight 3 4 $As [expr -$y1+$cover] [expr $z1-$cover]
             [expr $y1-$cover] [expr $z1-$cover]
 layer straight 3 2 $As [expr -$y1+$cover] 0.0 [expr $y1-$cover] 0.0
 layer straight 3 4 $As [expr -$y1+$cover] [expr -$z1+$cover]
             [expr $y1-$cover] [expr -$z1+$cover]
}

# Attach torsion to the RC beam section
#section Aggregator  $secTag $matTag1   $dof1   $matTag2   $dof2
.......   <-section $sectionTag>
section Aggregator     10        51        Vy         51         Vz
50   T          -section 1



# REINFORCED COLUMN
section Fiber 2 {

    # Create the concrete core fibers
    patch rect 6 $i_col $j_col -$y1 -$z1  $y1  $z1

    # Create the reinforcing fibers (left, middle, right)
    layer straight 3 4 $As [expr -$y1+$cover] [expr $z1-$cover]
             [expr $y1-$cover] [expr $z1-$cover]
    layer straight 3 2 $As [expr -$y1+$cover] 0.0 [expr $y1-$cover] 0.0
    layer straight 3 4 $As [expr -$y1+$cover] [expr -$z1+$cover]
             [expr $y1-$cover] [expr -$z1+$cover]
}

# Attach torsion to the RC beam section
#section Aggregator  $secTag $matTag1 $string1  $matTag2  $string2
.......   <-section $sectionTag>
section Aggregator     20        51        Vy         51         Vz
50   T          -section 2


# ###############################################################
# Beams

set yb1 [expr $beaH/2.0]
set zb1 [expr $beaB/2.0]

section Fiber 4 {

    # Create the concrete core fibers
    patch rect 1 $i_tra $j_tra [expr -$yb1] [expr -$zb1] [expr $yb1] [expr
```

```
    # Create the reinforcing fibers (left, middle, right)
    layer straight 3 4 $Ast [expr -$yb1+$cover] [expr $zb1-$cover]
          [expr $yb1-$cover] [expr $zb1-$cover]
    layer straight 3 4 $Ast [expr -$yb1+$cover] [expr -$zb1+$cover]
          [expr $yb1-$cover] [expr -$zb1+$cover]
}

set GJbea [expr $Gc*$C250*$beaB*pow($beaH,3)]
set GAbea [expr $Gc*$beaH*$beaB*5/6]

uniaxialMaterial Elastic 54 $GJbea
uniaxialMaterial Elastic 55 $GAbea


# Attach torsion to the RC beam section
# section Aggregator  $secTag $matTag1 $string1  $matTag2  $string2
.......    <-section $sectionTag>
section Aggregator     40        55        Vy         55         Vz
54     T          -section 4


# ################################################################
# Infills

set yi1 [expr $wi/2.0]
set zi1 [expr $ti/2.0]

section Fiber 5 {
 # Create the concrete core fibers
 patch rect 9 $i_inf $j_inf [expr -$yi1] [expr -$zi1]
       [expr $yi1] [expr $zi1]
}


# ################################################################
###
### ELEMENTS
###

source elementGeometry.tcl


# ################################################################
###
###  GRAVITY LOAD
```

```
###

source loadGeometry.tcl

# set finalmodelclock [clock milliseconds]


# ------------------------------
# End of model generation
# ------------------------------


# ------------------------------
# Start of analysis generation
# ------------------------------

# Create the system of equation
# a sparse solver with partial pivoting
system BandGeneral

# Create the constraint handler, the transformation method
constraints Transformation

# Create the DOF numberer, the reverse Cuthill-McKee algorithm
numberer RCM

# Create the convergence test, the norm of the residual with
# a tolerance of 1e-12 and a max number of iterations of 10
test NormDispIncr 1.0e-12  1000 3

# Create the solution algorithm, a Newton-Raphson algorithm
algorithm Newton

# Create the integration scheme
# the LoadControl scheme using steps of 0.1
integrator LoadControl 0.1

# Create the analysis object
analysis Static

# ------------------------------
# End of analysis generation
# ------------------------------


# ------------------------------
```

```
# Finally perform the analysis
# ------------------------------

# perform the gravity load analysis
# requires 10 steps to reach the load level
analyze 10


# ############################################################
###
### Output for the initialization
###

## Initialization flags

# Import input file
set Su [open "StartUpFlag.txt" r]
set SuFlag [read $Su]
close $Su

# Input data
# set SuFlag [split $SuData "\n"]


## Output

if {$SuFlag == 1} {

### Output geometric information for CostFunction.m

set SuOutput [open StartUpInput.txt w]
## floors height
# puts $SuOutput $heigth1
# puts $SuOutput $heigth2
# column section size
puts $SuOutput $colB
puts $SuOutput $colH
# concrete cover
puts $SuOutput $cover
# spacing of stirrups
puts $SuOutput $ss
# number of stirrups arms
puts $SuOutput $nsx
# diameter of stirrups
puts $SuOutput $Phis
# n. bars long
```

```
puts $SuOutput [expr 4*$nlx-4]
# diameter bars long
puts $SuOutput $Phi_lspi
# battens height
puts $SuOutput $ab
# battens thickness
puts $SuOutput $tb
# steel angle height
puts $SuOutput $la
# steel angle thickness
puts $SuOutput $ta
# concrete compression strenght
puts $SuOutput $fc
# steel yielding strenght
puts $SuOutput $fy
# yielding strenght for steel jacketing
puts $SuOutput $fyb
# beams height
puts $SuOutput $beaH
# beams thickness
puts $SuOutput $beaB
close $SuOutput
}
```

## 7.13 PushOver.tcl

```

# ----------------------------------------------------
# Start of Model Generation & Initial Gravity Analysis
# ----------------------------------------------------

# Do operations of Example3.1 by sourcing in the tcl file

source Geometry.tcl

loadConst -time 0.0

# ----------------------------------------------------
# End of Model Generation & Initial Gravity Analysis
# ----------------------------------------------------


# Loading the file from ModelCreator

source pushparFrame_analysis.tcl
```

```
# ------------------------------
# Finally perform the analysis
# ------------------------------

# Set some parameters
set currentDisp 0.0;
set ok 0


while {$ok == 0 && $currentDisp < $maxU} {

    set ok [analyze 1]

    # if the analysis fails try initial tangent iteration
    if {$ok != 0} {
        puts "regular newton failed"
        test NormDispIncr 1.0e-2  2000
     # test RelativeNormUnbalance 1.0e-3  2000
     # test EnergyIncr 1.0e-3  2000
        # algorithm ModifiedNewton
     # -initial
        set ok [analyze 1]
        if {$ok == 0} {puts "that worked .. back to regular newton"}
        test NormDispIncr 1.0e-3  2000
        algorithm Newton
    }

  set currentDisp [nodeDisp $nodo $dof]
  # puts [expr int($currentDisp)]
}


if {$ok == 0} {
  puts "Pushover analysis completed SUCCESSFULLY";
} else {
  puts "Pushover analysis FAILED";
}
```

## 7.14 ConfinedConcreteSR.tcl

```
#################################################################
###                                                           ###
###  Confined concrete according Saatcioglu - Razvi (1992)   ###
###                                                           ###
#################################################################


## Parameters (mm MPa):
#  id  -> number of the material
#  b   -> width of the section
#  h   -> height of the section
#  c   -> reinforcement cover width
#  s   -> spacing of the stirrups
#  fy  -> yielding stress of the steel used for stirrups
#  fc  -> compressive strength of concrete
#  ny  -> numb. of arms in y direction (along b)
#  nz  -> numb. of arms in z direction (along h)
#  Os  -> diameter of the stirrups (supposed equal in 2 dir.)
#  nly -> numb. of longitudinal bars along y direction
#  nlz -> numb. of longitudinal bars along z direction
#  Ol  -> diam. of the longitudinal bars on the corner
#  Oy  -> diam. of the long. bars on the edge along y (if present)
#  Oz  -> diam. of the long. bars on the edge along z (if present)
#


#       ^
#     z |
#
#       |---------------------|   |
#       | o------- o---------o |   |
#       | |        |         | |   |
#       | |        |         | |   |  h
#       | |        |         | |   |
#       | o--------o---------o |   |
#       |_____________________|   |    --->
#                                    y
#       |---------------------|
#                 b
```

```
#proc ConfinedConcreteCutOff {id tempId e_ccu e_cc} { }


proc ConfinedConcreteSR {id b h c s fy fc ny nz Os nly nlz
    Ol {Oy 0} {Oz 0}} {

set pi 3.141592654

# Dimension of the section without covers
set b0 [expr ($b-2*$c-$Os)]
set h0 [expr ($h-2*$c-$Os)]

# Distance of the longitudinal bars
set sly [expr (($b-2*$c-2*$Os-2*$Ol-($ny-2)*$Oy)/($ny-1))]
set slz [expr (($h-2*$c-2*$Os-2*$Ol-($nz-2)*$Oz)/($nz-1))]

# Area of one stirrup
set Ass [expr ($pi*pow($Os,2)/4.)]

# Confining pressures
set fly [expr ($nz*$Ass*$fy/($h0*$s))]
set flz [expr ($ny*$Ass*$fy/($b0*$s))]

set k2y [expr (0.26*sqrt(pow($h0,2)/$s/$slz/$fly))]
set k2z [expr (0.26*sqrt(pow($b0,2)/$s/$sly/$flz))]

# Effective confining pressures
set fley [expr $fly*$k2y]
set flez [expr $flz*$k2z]

set fle [expr (($fley*$h0+$flez*$b0)/($h0+$b0))]

# Corrective coefficient
set k1 [expr (6.7*pow($fle,-0.17))]

# peak strengt of concrete confined
set fcc [expr ($fc+($k1*$fle))]

set K [expr ($k1*$fle/$fc)]

set e_c0 0.002
set e_c085 0.0038

# strain at the peak (confined concrete)
set e_cc [expr ($e_c0*(1+5*$K))]
```

```
# strain at %85 fcc (for the slope of softening branch)
set rho [expr ((($ny+$nz)*$Ass)/(($b0+$h0)*$s))]
set e_cc85 [expr ($e_c085+(260*$rho*$e_cc))]

# strength and strain at the end
set fccu [expr (0.2*$fcc)]
set e_ccu [expr ($e_cc85*16/3-$e_cc*13/3)]

# temporary id for Concrete02 material
set tempId 1805
if {$tempId == $id} {incr tempId 42}

#                                Tag        fc        eps0     fpcu
epscu      lambda     ft      Et
uniaxialMaterial Concrete02   $tempId      -$fcc      -$e_cc   -$fccu
-$e_ccu      0.12      3.0    1500
#ConfinedConcreteCutOff $id $tempId $e_ccu $e_cc

# perc. of peak strength where concrete crush
set alfa 0.7

set max 5

# strain at the cut off (crush of the concrete)
set e_co [expr ((1-$alfa)*$e_ccu + ($alfa-0.2)*$e_cc)/0.8]

uniaxialMaterial MinMax $id $tempId -min -$e_co  -max $max
# puts "e_co = $e_co"
return $e_cc
# return[puts "done!"]
}
```

## 7.15 ConfinedConcreteBattens.tcl

```
#################################################################
###                                                           ###
###        Reinforced Concrete with hoops and battens        ###
###                                                           ###
#################################################################


## Parameters (mm MPa):
# gid   -> number of the material
# b     -> width of the section
# h     -> height of the section
# cvr   -> reinforcement cover width
# ss    -> spacing of the stirrups
# diast-> diameter of the stirrups (supposed equal on 2 dir.)
# nstz -> numb. of arms in z direction (along h)
# nsty -> numb. of arms in y direction (along b)
# sb    -> spacing of the battens
# tb    -> width of the battens
# ab    -> heigth of the battens
# la    -> heigth of the angular
# ta    -> width of the angular
# fc    -> compressive strength of concrete
# fyb   -> yielding stress of the steel used for stirrups
# eps0 -> strain at the peak (confined concrete)


#       ^
#     z |
#
#       |---------------------|   |
#       | o-------- o---------o |   |
#       | |         |         | |   |
#       | |         |         | |   |   h
#       | |         |         | |   |
#       | o---------o---------o |   |
#       |__________________ __|   |    --->
#                                y
#       |---------------------|
#                 b
```

```
proc ConfinedConcreteBattens {id b h cvr ss diast nstz nsty sb tb
    ab la ta fc fy fyb eps0} {

set Ast      [expr 3.14*($diast/2.0)*($diast/2.0)]
set Astz     [expr $nstz*$Ast]
set Asty     [expr $nsty*$Ast]

set b0       [expr ($b-2*$cvr)]
set h0       [expr ($h-2*$cvr)]
set sm       [expr ($ss+$sb)/2] ; # s tilde

set ke  [expr (1.-($ss-$diast)/(2.*$b0))*(1.-($ss-$diast)/(2.*$h0))]
set Asbe     [expr $tb*$ab*$fyb/$fy]

set roz        [expr $Astz/($ss*$h0)+2.*$Asbe/($sb*$h)]
set roy        [expr $Asty/($ss*$b0)+2.*$Asbe/($sb*$b)]
set ros        [expr ($Astz+$Asty+4.*$Asbe)/($sm*($b0+$h0))]

set flez     [expr $ke*$roz*$fy]
set fley     [expr $ke*$roy*$fy]
set fle        [expr ($flez*$b0+$fley*$h0)/($b0+$h0)]

set k1         [expr 6.7*pow($fle,-0.17)]
set K          [expr $k1*$fle/$fc]

set fcc        [expr $fc+$k1*$fle]
set epscc    [expr $eps0*(1.+5.*$K)]

set epscc85 [expr 0.0036+260.*$ros*$epscc]
set fcc20    [expr 0.2*$fcc]
set epscc20 [expr $epscc+(1.-0.2)*($epscc85-$epscc)/0.15]

set alfaU    0.7;
set epsccu   [expr $epscc+(1.-$alfaU)*($epscc85-$epscc)/0.15]

set tempId 1653
if {$tempId == $id} {incr tempId 42}

# Core concrete (confined)
uniaxialMaterial Concrete02 $tempId -$fcc -$epscc -$fcc20
-$epscc20 0.12 0.00 0.00

uniaxialMaterial MinMax  $id $tempId -min -$epsccu -max  1
}
```

# Bibliography


[1] D. Sharma and P. Chandra, "A comparative analysis of soft computing techniques in software fault prediction model development," *International Journal of Information Technology*, vol. 11, no. 1, pp. 37–46, 2019.

[2] R. Falcone, C. Lima, and E. Martinelli, "Soft computing techniques in structural and earthquake engineering: A literature review," *Engineering Structures*, vol. 207, p. 110269, 2020.

[3] R. Woodward and E. J. Kelleher, "Towards 'smart lasers': self-optimisation of an ultrafast pulse source using a genetic algorithm," *Scientific reports*, vol. 6, p. 37616, 2016.

[4] G. G. Deierlein, A. M. Reinhorn, and M. R. Willford, "Nonlinear structural analysis for seismic design," *NEHRP seismic design technical brief*, vol. 4, pp. 1–36, 2010.

[5] M. Saatcioglu and S. R. Razvi, "Strength and ductility of confined concrete," *Journal of Structural Engineering*, vol. 118, no. 6, pp. 1590–1607, 1992.

[6] C. W. Churchman, R. L. Ackoff, and E. L. Arnoff, "Introduction to operations research.," 1957.

[7] Z. Michalewicz and D. B. Fogel, *How to solve it: modern heuristics.* Springer Science & Business Media, 2013.

[8] O. Bozorg-Haddad, M. Solgi, and H. A. Loáiciga, *Meta-heuristic and evolutionary algorithms for engineering optimization.* John Wiley & Sons, 2017.

[9] C. Darwin and W. F. Bynum, *The origin of species by means of natural selection: or, the preservation of favored races in the struggle for life.* AL Burt New York, 2009.

[10] G. Mendel, "Versuche uber pflanzen-hybriden," *Verhandlungen des naturforschenden Vereins in Brunn fur*, vol. 4, pp. 3–47, 1866.

[11] F. Glover, "Future paths for integer programming and links to ar tifi cial intelli g en ce," *Computers operations research*, vol. 13, no. 5, pp. 533–549, 1986.

[12] T. Stützle, "Local search algorithms for combinatorial problems-analysis, algorithms and new applications," *DISKI-Dissertationen zur Künstliken Intelligenz. In x, Sankt Augustin, Germany*, 1999.

[13] N. A. Barricelli *et al.*, "Esempi numerici di processi di evoluzione," *Methodos*, vol. 6, no. 21-22, pp. 45–68, 1954.

[14] N. A. Barricelli, *Symbiogenetic evolution processes realized by artificial methods.* 1957.

[15] J. H. Holland *et al.*, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* MIT press, 1992.

[16] D. E. Goldberg and M. P. Samtani, "Engineering optimization via genetic algorithm," in *Electronic computation*, pp. 471–482, ASCE, 1986.

[17] S. S. Rao, *Engineering optimization: theory and practice.* John Wiley & Sons, 2019.

[18] A. Eiben and C. Schippers, "On evolutionary exploration and exploitation," *Fundam. Inform.*, vol. 35, pp. 35–50, 08 1998.

[19] N. D. Lagaros, M. Papadrakakis, and G. Kokossalakis, "Structural optimization using evolutionary algorithms," *Computers & structures*, vol. 80, no. 7-8, pp. 571–589, 2002.

[20] H. Seo, J. Kim, and M. Kwon, "Optimal seismic retrofitted rc column distribution for an existing school building," *Engineering Structures*, vol. 168, pp. 399–404, 2018.

[21] R. Falcone, F. Carrabs, R. Cerulli, C. Lima, and E. Martinelli, "Seismic retrofitting of existing rc buildings: a rational selection procedure based on genetic algorithms," in *Structures*, vol. 22, pp. 310–326, Elsevier, 2019.

[22] G. Mahdavi, K. Nasrollahzadeh, and M. Hariri-Ardebili, "Optimal frp jacket placement in rc frame structures towards a resilient seismic design," *Sustainability*, vol. 11, no. 24, p. 6985, 2019.

[23] F. Di Trapani, M. Malavisi, G. C. Marano, A. P. Sberna, and R. Greco, "Optimal seismic retrofitting of reinforced concrete buildings by steel-jacketing using a genetic algorithm-based framework," *Engineering Structures*, vol. 219, p. 110864, 2020.

[24] F. McKenna, G. L. Fenves, M. H. Scott, *et al.*, "Open system for earthquake engineering simulation," *University of California, Berkeley, CA*, 2000.

[25] A. Lipowski and D. Lipowska, "Roulette-wheel selection via stochastic acceptance," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 6, pp. 2193–2196, 2012.

[26] T. Gwiazda, "Genetic algorithms reference vol. i. crossover for single-objective numerical optimization problems," 2006.

[27] D. A. Umbarkar and P. Sheth, "Crossover operators in genetic algorihms: a review," *ICTACT Journal on Soft Computing ( Volume: 6 , Issue: 1 )*, vol. 6, 10 2015.

[28] G. Syswerda, "Uniform crossover in genetic algorithms," in *ICGA*, 1989.

[29] H. S. Oey and C. J. Aldrete, "Simple method for upgrading an existing reinforced-concrete structure," *Practice Periodical on Structural Design and Construction*, vol. 1, no. 1, pp. 47–50, 1996.

[30] Y.-F. Wu, T. Liu, and L. Wang, "Experimental investigation on seismic

retrofitting of square rc columns by carbon frp sheet confinement combined with transverse short glass frp bars in bored holes," *Journal of Composites for Construction*, vol. 12, no. 1, pp. 53–60, 2008.

[31] V. Badalamenti, G. Campione, and M. L. Mangiavillano, "Simplified model for compressive behavior of concrete columns strengthened by steel angles and strips," *Journal of engineering mechanics*, vol. 136, no. 2, pp. 230–238, 2010.

[32] G. Campione, L. Cavaleri, F. Di Trapani, and M. Ferrotto, "Frictional effects in structural behavior of no-end-connected steel-jacketed rc columns: experimental results and new approaches to model numerical and analytical response," *Journal of Structural Engineering*, vol. 143, no. 8, p. 04017070, 2017.

[33] R. Montuori and V. Piluso, "Reinforced concrete columns strengthened with angles and battens subjected to eccentric load," *Engineering Structures*, vol. 31, no. 2, pp. 539 – 550, 2009.

[34] F. E. Richart, A. Brandtzaeg, and R. L. Brown, "A study of the failure of concrete under combined compressive stresses," *Bulletin No. 185 Engineering Experiment Station*, vol. 26, no. 12, pp. 7–92, 1928.

[35] G. Balmer, *Shearing Strength of Concrete Under High Triaxial Stress: Computation of Mohr's Envelope as a Curve.* U.S. Department of the Interior, Bureau of Reclamation, 1949.

[36] J. B. Mander, M. J. N. Priestley, and R. Park, "Theroretical stress-strain model for confined concrete," *Journal of Structural Engineering*, vol. 114, no. 8, pp. 1804–1826, 1989.

[37] E. Hognestad, *A Study of Combined Bending and Axial Load in Reinforced Concrete Members.* Bulletin (University of Illinois (Urbana-Champaign campus). Engineering Experiment Station)), University of Illinois, 1951.

[38] G. Campione, L. Cavaleri, F. D. Trapani, and M. F. Ferrotto, "Frictional effects in structural behavior of no-end-connected steel-jacketed rc columns: Experimental results and new approaches to model numerical and analytical response," *Journal of Structural Engineering*, vol. 143, no. 8, p. 04017070, 2017.

[39] F. Braga, R. Gigliotti, and M. Laterza, "Analytical stress–strain relationship for concrete confined by steel stirrups and/or frp jackets," *Journal of Structural Engineering*, vol. 132, no. 9, pp. 1402–1416, 2006.

[40] F. D. Trapani, M. Malavisi, G. C. Marano, A. P. Sberna, and R. Greco, "Optimal seismic retrofitting of reinforced concrete buildings by steel-jacketing using a genetic algorithm-based framework," *Engineering Structures*, vol. 219, p. 110864, 2020.

[41] F. Di Trapani, G. Bertagnoli, M. F. Ferrotto, and D. Gino, "Empirical equations for the direct definition of stress–strain laws for fiber-section-based macromodeling of infilled frames," *Journal of Engineering Mechanics*, vol. 144, no. 11, p. 04018101, 2018.

[42] P. Fajfar and P. Gašperšič, "The n2 method for the seismic damage analysis of rc buildings," *Earthquake engineering & structural dynamics*, vol. 25, no. 1,

pp. 31–46, 1996.
[43] " Eurocode 8: Design of structures for earthquake resistance - Part 1: General rules, seismic actions and rules for buildings ," 2004.
[44] "Norme tecniche per le costruzioni, decreto ministeriale 17 gennaio 2018," 2018.
[45] T. Vidic, P. Fayfar, and M. Fischinger, "Procedure for determining consistent inelastic design spectra," *Nonlinear Seismic Analysis and Design of Reinforced Concrete Buildings, Supplementaly Proc. of a worksholp held in Bled*, no. 103, pp. 9–32, 1992.
[46] D. E. Biskinis, G. K. Roupakias, and M. N. Fardis, "Degradation of shear strength of reinforced concrete members with inelastic cyclic displacements," *Structural Journal*, vol. 101, no. 6, pp. 773–783, 2004.
[47] "Eurocode 8: Design of structures for earthquake resistance-part 3: Assessment and retrofitting of buildings," 2005.
[48] "Istruzioni per l'applicazione dell'«aggiornamento delle "norme tecniche per le costruzioni"» di cui al decreto ministeriale 17 gennaio 2018," 2019.
[49] W. Ritter, "Die bauweise hennebique," *Schweizerische Bauzeitung*, vol. 33, no. 7, pp. 59–61, 1899.
[50] E. Mörsch, "Reinforced concrete: Theory and application," 1922.
[51] F. Di Trapani and M. Malavisi, "Seismic fragility assessment of infilled frames subject to mainshock/aftershock sequences using a double incremental dynamic analysis approach," *Bulletin of Earthquake Engineering*, vol. 17, no. 1, pp. 211–235, 2019.