Università
Ca'Foscari
Venezia

Corso di Dottorato di ricerca

in Informatica

ciclo XXXV

Tesi di Ricerca

Optimization of Steel Megastructures

SSD: INF/01

Coordinatore del Dottorato
ch. prof. Agostino Cortesi

Supervisore
ch. prof. Andrea Torsello

Dottorando
Alessia Cacchi
Matricola 956517

Contents

List of Figures v

List of Tables xii

Notations xvi

Reference Software xxii

Introduction 1

CHAPTER 1: Civil Engineering and Artificial Intelligence 1

1.1 Structural design overview 1

1.1.1 Structural design process 1

1.1.2 Integrated structural design 4

1.1.3 Importance of node design in steel structures 5

1.1.3.1 Structural node design with local models *9*

1.1.4 Cost of steel structures 10

1.1.5 Design tools 13

1.2 Artificial Intelligence overview 15

1.2.1 Artificial Intelligence (AI) 15

1.2.2 Machine Learning (ML) 16

1.3 Artificial Intelligence in Civil Engineering 18

1.4 Conclusion 20

CHAPTER 2: The search for the optimum **21**

2.1 Optimization problem 21

2.2 Gradient-based optimization 22

2.3 Meta-heuristic optimization 25

2.4 Optimization in structural design 27

2.5 Conclusion 28

CHAPTER 3: Surrogate models **29**

3.1 Design space approximation 31

3.2 Data-driven surrogate models: an overview 32
3.3 Neural Network (NN) 33
3.4 Graph Neural Network 37
3.4.1 Convolution operators 41
3.5 Conclusion 43
CHAPTER 4: Optimization of a steel multi-way node 44
4.1 Problem statement 44
4.2 Variables 45
4.2.1 Beam variables 46
4.2.2 Joint variables 47
4.3 Optimization framework 49
4.4 Structural checks 51
4.4.1 Beam checks 51
4.4.2 Joint Checks 54
4.5 Fitness function 54
4.6 Optimization algorithms 55
4.6.1 Genetic Algorithm 55
4.6.2 Gradient-based Algorithm 57
4.6.3 Grouping Algorithm 60
4.7 Case study 62
4.7.1 Calculation model 66
4.7.2 Multi-way node optimization 68
4.7.2.1 Genetic algorithm otpimization 71
4.7.2.2 Zeroth-order block coordinate descent optimization 77
4.7.2.3 Model comparison 84
4.8 Conclusion 87
CHAPTER 5: Optimization of a steel frame 90
5.1 Proposal workflow 91
5.2 *LoadsToForces* module 92
5.2.1 Representation of model loads and forces on beams 93

5.2.2 Case studies 99

5.2.3 *LoadsToForces* module creation approaches 100

5.2.3.1 Low-rank stiffness matrix *100*

5.2.3.2 MLP model *104*

5.2.3.3 GNN model *106*

5.2.3.4 GNN model with heterogeneous graph *107*

5.2.3.5 GNN model with twin approach *114*

5.2.4 Comparing approaches 122

5.3 *MLPsectionOpt* module 122

5.3.1 Database 123

5.3.2 Training 125

5.3.3 Optimization results 129

5.3.3.1 Two-dimensional Frame *130*

5.3.3.2 Three-dimensional Frame *133*

5.3.4 Additional training 144

5.3.4.1 Approach A *145*

5.3.4.2 Approach B *146*

5.3.5 Optimization results with additional training 147

5.3.5.1 Two-dimensional Frame *148*

5.4 Conclusion 164

Conclusions 166

References 175

List of Figures

Figure 1. Relationship between design freedom and design knowledge in building design projects (3) 3
Figure 2. Iteration types occurring in structural design stages (4) 4
Figure 3. The traditional design of civil structures, often based on a “trial and error” approach.......... 4
Figure 4. Collapsed zone of Terminal 2E at Charles de Gaulle Airport in Paris (5) 5
Figure 5. Relationships between the parts involved in the design of steel structures using the traditional approach (a) and by introducing tools that allow more correct calculation of joints (b) (6).......... 6
Figure 6. Geometric parameters of beam-column connection (8) 8
Figure 7. Example of FEM model of a multi-way node 8
Figure 8. Example of design framework for a steel megastructure.......... 10
Figure 9. Table defining the execution class (EXC) (9) 12
Figure 10. Types of structural design tools evolution (3) 15
Figure 11. Example of integration between BIM (Build Information Model) and structural calculation: through the Grasshopper platform (left window) the BIM software Tekla is connected to the FEM software SOFiSTiK so as to facilitate the exchange of information between the two software (10) 15
Figure 12. Comparison of traditional and Artificial Intelligence algorithms (13) 17
Figure 13. Common Machine Learning algorithms (12) 17
Figure 14. Research publications on the use of AI branches in civil engineering (12) 19
Figure 15. Kuhn–Tucker condition at a constrained optimum (19).......... 24
Figure 16. Gradient descent stuck at local minima (20) 24
Figure 50. The actual structure is translated into a model suitable for evaluating its structural behavior. The calculation model can be used for FEM analysis or to derive surrogate models. 30
Figure 18. Different levels of detail of the FEM model of a framed building (3) 31
Figure 19. Surrogate modeling procedure. (23).......... 33
Figure 20. Diagram of input processing at the end of the message center (13) 34

Figure 21. Neural Network structure, in which nodes represent the units of each layer while links represent the connections between units of successive layers, each of which is associated with a weight. The biases are represented by x_0 and z_0 (24) 35
Figure 22. Backward propagation of error information (24) 37
Figure 23. Skip connection example (25) 37
Figure 24. Examples of application fields of graphs (26) 38
Figure 25. Architecture of a Graph Neural Network (26).......... 40
Figure 26. Example of Convolution Neural Network architecture (27).......... 40
Figure 27. Example of Recurrent Neural Network architecture (28).......... 41
Figure 28. Message-Passing Neural Network (MPNN) (29) 42
Figure 29. Beams with CHS section before installation.......... 46
Figure 30. Section CHS.......... 46
Figure 31. Example of a multi-way steel connection with single plate connection joints 47
Figure 32. Plates that make up a single plate connection joint 48
Figure 33. Single plate connection.......... 48
Figure 34. Optimization framework.......... 50
Figure 35. Cross section classification for circular hollow sections 51
Figure 24. Genetic algorithm.......... 56
Figure 37. Example of a side view of a single arch 62
Figure 38. Side view of a facade.......... 62
Figure 39. View of the cover structure.......... 63
Figure 40. Example of joint with single plate connection.......... 63
Figure 41. Example of joint with single plate connection and split joint covers 64
Figure 42. Example of joint for cruciform plate.......... 64
Figure 43. Example of bolted flange joint 64
Figure 44. Example of welded flange joint 65
Figure 45. Example of pin connection 65
Figure 46. Calcutaion model shell.......... 66
Figure 47. Calculation model beams.......... 66
Figure 36. Multi-way node under study 68
Figure 49. Beam numbering.......... 69
Figure 50. Influence of mutation probability p on the performance of the Genetic Algorithm 72

Figure 51. Influence of generation size on the performance of the Genetic Algorithm 72
Figure 52. Influence of proportion of the best candidates and random candidates selected for the creation of the new generation on the performance of the Genetic Algorithm 73
Figure 53. Model of multi-way node optimized with the Genetic Algorithm 76
Figure 54. Model of multi-way node optimized with the Genetic Algorithm 77
Figure 55. Trend of the number of variables per block as function of the number of blocks J, with degree of sparsity s equal to 20 78
Figure 56. Trend of the number of directions sampled n_{dir} as function of the number of blocks J, with degree of sparsity s being 20 78
Figure 57. Trend of the number of directions sampled n_{dir} as function of the level of sparsity s, with the number of blocks J being 5 78
Figure 58. Influence of sparsity level on the performance of the Gradient-based Algorithm 80
Figure 59. Influence of number of blocks of variables on the performance of the Gradient-based Algorithm 80
Figure 60. Model of multi-way node optimized with the Gradient-based Algorithm 83
Figure 61. Model of multi-way node optimized with the Gradient-based Algorithm 84
Figure 62. Proposal workflow for steel frame optimization 92
Figure 63. Beam end forces 94
Figure 64. Beam node forces 95
Figure 65. FEM model with beam local axis 97
Figure 66. Fixed constraints are applied to the starting model (a) at the nodes, and from the solution of this model, forces at the nodes are derived as reactions (b) and forces on the beams are obtained (c) 98
Figure 67. Two-dimensional frame used as a case of study 99
Figure 68. Spatial frame used as a case of study 99
Figure 69. Two-dimensional frame loading conditions in which the low-rank stiffness matrix result was compared with that given by the FEM 103
Figure 70. Loss trend during MLP training in the bidimensional frame 105
Figure 71. Error trend of MLP in validation set in the bidimensional frame 106
Figure 72. Heterogeneous graph for model in Figure 67 107
Figure 73. LeakyReLU function 108

Figure 74. Analysis of MAE and MSE as loss function in training and test for the bidimensional frame 108
Figure 75. Effect of scaling between errors of forces and moments using MAE as loss function in two-dimensional frame 109
Figure 76. Comparison of network composed of convolutional layers and nonlinear layers and composed of linear and nonlinear layers. In both cases, the hidden layers have dimension k=32 and there are 5 packets of inner layers following the initial convolution consisting of convolutive/linear layer + nonlinear layer. The results refer to the two-dimensional frame. 109
Figure 77. NodeGraph for two-dimensional frame in Figure 67....115
Figure 78. NodeGraph for spatial frame in Figure 68 115
Figure 79. BeamGraph for two-dimensional frame in Figure 55....117
Figure 80. BeamGraph for spatial frame in Figure 56....117
Figure 81. Trend of twin GNN loss for two-dimensional structure for different dimensions of space k 120
Figure 82. Error trend of twin GNN in the validation set for two-dimensional structure for different dimensions of space k....121
Figure 83. Trend of twin GNN loss for spatial frame with concentrated loads for k=16121
Figure 84. Error trend of twin GNN in the validation set for spatial frame with concentrated loads for k=16....121
Figure 85. Sampling of length-section pairs obtained with Uniform Distribution 124
Figure 86. Sampling of length-section pairs obtained with Latin Hypercube 124
Figure 87. Training performance for different network architectures created with Uniform Distribution sampling for two-dimensional frame optimization in Figure 67....126
Figure 88. Incorrect predictions in test set for different network architectures. Data are referred to Neural Networks created with Uniform Distribution sampling for two-dimensional frame optimization in Figure 67 126
Figure 89. Mean number of difference classes between target and network output for items of test set with incorrect predictions. Data are referred to Neural Networks created with Uniform Distribution sampling for two-dimensional frame optimization in Figure 67 126

Figure 90. Training performance for different number of hidden layers (n) with inner layers having k=256 nodes. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67 127

Figure 91. Training performance for different number of nodes in the inner layers (k) with the number of layers equal to 5. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67............127

Figure 92. Incorrect predictions in test set for different number of hidden layers (n) with inner layers having k=256 nodes. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67............127

Figure 93. Incorrect predictions in test set for different number of nodes in the inner layers (k) with the number of layers equal to 5. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67 128

Figure 94. Mean error in incorrect predictions in test set for different number of hidden layers (n) with inner layers having k=256 nodes. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67 128

Figure 95. Mean error in incorrect predictions in test set for amounts of nodes in the inner layers (k) with 5 pairs of linear-ReLU inner layers. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67 128

Figure 96. Iterative optimization workflow...130

Figure 97. Two-dimensional frame optimized with network obtained from database with Uniform Distribution sampling 131

Figure 98. Two-dimensional frame optimized with network obtained from database with Latin Hypercube sampling 132

Figure 99. Non-optimized three-dimensional frame (starting model) 133

Figure 100. Indices of the sections of the optimized three-dimensional model beams. 138

Figure 101. Area of the sections of the optimized three-dimensional model beams. 139

Figure 102. Workflow for the additional training of MLPsectionOpt 144

Figure 103. Model of two-dimensional frame optimized with additional training with Approach A (lr1=0.001 and lr2=0.0005) 149

Figure 104. Model of two-dimensional frame optimized with additional training with Approach B (lr1=0.001 and lr2=0.0005) 150

Figure 105. Model of two-dimensional frame optimized with additional training with Approach A and penalty coefficient equal to 10 (lr1=0.003 and lr2=0.005) 151
Figure 106. Model of two-dimensional frame optimized with additional training with Approach B and penalty coefficient equal to 10 (lr1=0.001 and lr2=0.0005) 152
Figure 107. Indices of the sections of the optimized three-dimensional model beams obtained using the additional training (Approach A) 153
Figure 108. Area of the sections of the optimized three-dimensional model beams obtained using the additional training 153
Figure 109. Indices of the sections of the optimized three-dimensional model beams obtained using the additional training 158
Figure 110. Area of the sections of the optimized three-dimensional model beams obtained using the additional training 159

List of Tables

Table 1. Type of variables for each beam and each joint 45

Table 2. Number of variables for each beam and each joint 45

Table 3. Variables of the joints 48

Table 4. Strength checks for sections of class 1 or 2 52

Table 5. Buckling checks for sections of class 1 or 2 52

Table 6. Strength checks for sections of class 3 53

Table 7. Buckling checks for sections of class 3 53

Table 8. Nomenclature used for the strength and buckling equations for the beams 53

Table 9. Beam attributes 69

Table 10. Joints attributes 70

Table 11. Beams costs 70

Table 12. Joints costs 70

Table 13. Summary of costs 71

Table 14. Beam attributes of better solution from Genetic Algorithm 74

Table 15. Joints attributes of better solution from Genetic Algorithm 74

Table 16. Beams costs of better solution from Genetic Algorithm 74

Table 17. Joints costs of better solution from Genetic Algorithm 75

Table 18. Summary of costs of better solution from Genetic Algorithm 75

Table 19. Variation of beam characteristics between the original model and the model optimized with the Genetic Algorithm 75

Table 20. Variation of joint characteristics between the original model and the model optimized with the Genetic Algorithm 76

Table 21. Beam attributes of better solution from Gradient-based Algorithm 81

Table 22. Joints attributes of better solution from Gradient-based Algorithm 81

Table 23. Beams costs of better solution from Gradient-based Algorithm 81

Table 24. Joints costs of better solution from Gradient-based Algorithm 82

Table 25. Summary of costs of better solution from Gradient-based Algorithm 82

Table 26. Variation of beam characteristics between the original model and the model optimized with the Gradient-based Algorithm 82

Table 27. Variation of joint characteristics between the original model and the model optimized with the Gradient-based Algorithm 83

Table 28. Comparison of beam cost between initial model and model optimized with Genetic Algorithm and with Gradient-based Algorithm 84

Table 29. Comparison of joint cost between initial model and model optimized with Genetic Algorithm and with Gradient-based Algorithm 85

Table 30. Comparison of total cost between initial model and model optimized with Genetic Algorithm and with Gradient-based Algorithm 85

Table 31. Error of the low-ran matrix for model in Figure 67 102

Table 32. Result of comparing the result of low-rank stiffness matrix and that given by FEM for the case in Figure 69 102

Table 33. Error of the low-rank matrix for model in Figure 68 with 1000 load cases in which the loads are applied as concentrated forces and moments at nodes 103

Table 34. Error of the low-rank matrix for model in Figure 68 with 1000 load cases applied as beams distributed loads. The loads were described with 3 parameters per beam representative of the value of the loads distributed in the 3 directions of the reference system. 104

Table 35. Error of the low-rank matrix for model in Figure 68 with 1000 load cases applied as beams distributed loads. The loads were described using the unique approach 104

Table 36. Entries of eij matrix 112

Table 37. Errors of different approaches for constructing the surrogate model for the bidimensional frame 122

Table 38. Beam starting sections and corresponding utilization rate of the two-dimensional frame. The parameter u is the target utilization rate, which is considered to be equal to 0.99 131

Table 39. Results of the optimization of the beams with the networks created with the two databases 132

Table 40. Cross-sections and utilization rate of the three-dimensional frame starting model. The deviation between the beam utilization rate and the target rate u, which is 0.99, is reported. The utilization rate for class 4 sections was not reported. 138

Table 41. Results of the optimization of the beams of the three-dimensional frame 143

Table 42. Overall deviation values between actual and target utilization rates of the optimized three-dimensional frame beams, excluding beams with utilization rates greater than 1.4 (i.e., beams number 3, 18, and 108) 143

Table 43. Two-dimensional frame optimized with additional training with approach A (lr1=0.001 and lr2=0.0005) 148

Table 44. Two-dimensional frame optimized with additional training with Approach B (lr1=0.001 and lr2=0.0005) 149

Table 45. Two-dimensional frame optimized with additional training with approach A and penalty coefficient equal to 10 (lr1=0.003 and lr2=0.005) 150

Table 46. Two-dimensional frame optimized with additional training with Approach B and penalty coefficient equal to 10 (lr1=0.001 and lr2=0.0005) 151

Table 47. Results of the optimization of the beams of the three-dimensional frame with additional training (Approach A) 158

Table 48. Overall deviation values between actual and target utilization rates of the beams of the optimized three-dimensional frame with additional training (Approach A), excluding the beam with utilization rate greater than 1.4 (i.e., beam number 18) 158

Table 49. Results of the optimization of the beams of the three-dimensional frame with additional training (Approach B). 163

Table 50. Overall deviation values between actual and target utilization rates of the beams of the optimized three-dimensional frame with additional training (Approach B), excluding the beams with utilization rates greater than 1.4 (i.e., beams number 3, 6, 18 and 162). 163

Notations

Numbers

0_{dxd}	Zero matrix of shape [d x d]

Latin lower-case letters

b	Number of beams
c_1	Cost of a beam
c_2	Cost of a joint
c_i	Load vector at i-th node
$c_{i,j}$	Load vector at j-th node/beam in i-th load case
d	Dimension size
d_b	Number of variables per beam
d_j	Number of variables per joint
e_{vu}	Features of edge between graph node v and graph node u
f	Fitness function/objective function
$\bar{f}$	Undersize penalty coefficient
f_i	Vector of node forces at end nodes of a beam i
g	Gradient of function
h	Activation function
i_1	Versor of beam local axis 1
i_2	Versor of beam local axis 2
i_3	Versor of beam local axis 3
l	Beam length
lr	Learning rate
h_v^t	Hidden state of graph node v at step t
j	Number of joints
k	Size of the hidden space
m	Number of graph nodes

$\overline{m}$	Matrix of versors of beam local reference system
$m_v{}^t$	Message of graph node v at step t
n	Number of nodes
n	Number of layers of neural network
n_{best}	Number of the best individuals in a generation using the genetic algorithm
n_{dir}	Number of directions analyzed
n_{LC}	Number of load cases
n_r	Number of individuals randomly selected from the previous generation with the genetic algorithm
n_s	Number of section classes
n_{tot}	Number of individuals randomly selected from the previous generation with the genetic algorithm
n_{units}	Number of nodes composing a neural network layer
o	Oversampling parameter
p_1	Penalty factor related to utilization ratio of a beam
p_2	Penalty factor related to utilization ratio of a joint
p_{mut}	Mutation probability
r	Number of variables per block
s	Gradient sparsity level
s_i	Parameters of the beam forces at a station of the i-th beam
$s_{i,j}$	Beam forces vector at j-th beam in i-th load case
s_{block}	Gradient sparsity level of the variable block
u	Utilization rate value/vector
$\bar{u}$	Target utilization rate value/vector
v	Vector of velocities
w	Neural network layer parameter
w_e	Weight of graph link
x	Vector of variables
x_v	Attribute vector of graph nodes
$\hat{y}$	Feature vector of the whole graph

z	Output of an inner layer of the neural network

Latin upper-case letters

A	Encoding matrix
Adj	Adjacency matrix
AX	Axial force
B	Decoding matrix
BM_1	Bending moment around the direction 1
BM_2	Bending moment around the direction 2
C	Load matrix for a single load condition
$\bar{C}$	Load matrix of all load conditions
$\bar{C}^+$	Pseudoinverse of matrix $\bar{\mathrm{C}}$
E	Transformation matrix
E_g	Set of graph edges
F_1	Node force in direction 1
F_2	Node force in direction 2
F_3	Node force in direction 3
F_x	Node force in the global direction X
F_y	Node force in the global direction Y
F_z	Node force in the global direction Z
G	Graph
$I_{d\,x\,d}$	Identity matrix of shape [d x d]
J	Number of blocks of variables
M	Message between graph nodes
M_1	Moment of the node around direction 1
M_2	Moment of the node around direction 2
M_3	Moment of the node around direction 3
M_x	Moment of the node around the global direction X
M_y	Moment of the node around the global direction Y
M_z	Moment of the node around the global direction Z

$\overline{M}$	Matrix of versors of beam local reference system
M_t	Message-passing function at step t
N_v	Neighborhood of node v
Q	Mask matrix
R	Rotation matrix
$\tilde{R}$	Readout function
$\bar{R}$	Low-rank stiffness matrix
S	Beam forces matrix for a single load condition
$\bar{S}$	Beam forces matrix of all load conditions
SF_1	Shear forces along direction 1
SF_2	Shear forces along direction 2
T	Total number of steps
T	Transformation matrix between beam forces and node forces
TQ	Torque moment
U	Eigenvector matrix
U_t	Update function at step t
V	Set of graph vertices
W	Weight matrix
W_e	Set of weight of graph link
X	Input matrix of the neural network layer/model
X_v	Set of vectors of attributes of nodes in the graph
Y	Output matrix of the neural network layer/model
$\bar{Y}$	Output matrix of the neural network for section classification after using the softmax function for row
Z	Direction matrix with -1 or +1 entries with equal probability

Greek lower-case letters

α	Variable update modulation parameter based on gradient
δ	Variable steps

Script letters

$\mathcal{A}$	Section area
$\mathcal{F}$	Fourier transform

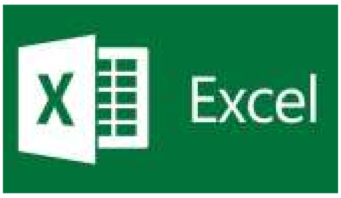
Double-struck letters

$\mathbb{E}$	Error function
$\mathbb{R}$	Set of real numbers

Reference Software

The software used for the development of this document are reported hereunder:

- Excel



- Visual Studio

- Straus7

- Rhinoceros 7.0

Introduction

The design of steel structures involves a large number of equations and relationships related to different aspects of the project, such as materials, geometries and costs. Traditional design methods generally require subsequent design revisions and relying on a trial-and-error approach does not lead to optimized solutions in a reasonable timeframe. This problem is particularly important for steel megastructures, such as stadiums and bridges, where there are high numbers of variables, high computational costs for evaluating structural performance, and various complex phenomena that require the development of sophisticated local models. The search for optimized design solutions requires the development of an innovative approach to the design, in which the experience of designers and traditional numerical methods are combined with the techniques offered by Artificial Intelligence (AI). This research project aims to develop tools for the design of steel megastructures that provide more optimized solutions from the earliest design stages, resulting in reduced costs and environmental impact, and requiring limited subsequent revisions. Several approaches provided by Artificial Intelligence, both stochastic and deterministic in nature, are investigated to achieve this.

The initial part of the first chapter of this thesis aims to provide the reader with definitions and an overview of structural design. In particular, the complexity of the design process related to the complexity of the physical phenomena that control the problem, the number of variables that need to be defined by design and that are interconnected, and the amount of different professionals involved in the design process are highlighted. The chapter continues by providing definitions of Artificial Intelligence and Machine Learning and investigating the state of the art on the role they play in the field of Civil Engineering. Chapters 2 and 3 contain technical notions about Artificial Intelligence methods that have been studied in this research for the development of useful tools to support structural design activities, facilitating the achievement of optimized results more quickly. In particular, the second chapter of the thesis is devoted to optimization problems, which are a popular type of problem in many disciplines. The third chapter, on the other hand, analyzes techniques for developing surrogate models, which aim to reduce the burden associated with analyzing and evaluating different candidate solutions to the problem.

Particular emphasis is placed on models implemented using Neural Networks. Chapters 4 and 5 are devoted to the development of AI-based structural design tools. Specifically, in Chapter 4 algorithms are developed to optimize a portion of the structure considering both beams and joints. Instead, in Chapter 5, Artificial Intelligence finds application for the creation of a tool for optimizing the beams of a frame and for the development of surrogate models that speed up the analysis of candidate solutions. In both chapters, a practical application of the proposed methods and a critical analysis of the results were carried out. The concluding chapter of the thesis, on the other hand, draws a summary of what has been observed and analyzed during the course of the thesis, both on the state of the art of structural design and research on the application of Artificial Intelligence methods in it, and on the results obtained through the methods developed in the course of the research.

CHAPTER 1: Civil Engineering and Artificial Intelligence

Civil engineering is the branch of engineering that deals with the design and construction of civil works, such as buildings, bridges, infrastructures, hydraulic works. This discipline has a significant economic impact: according to a report by Horta et al., the construction industry constitutes about 9% of world GDP. (1) Furthermore, it also has a major impact on resource exploitation and environmental pollution. According to (2), in 2017 the building sector was the second largest consumer of energy in China, consuming about 20% of the total energy, about 23% of electricity and about 30% of the total CO_2 emissions. Despite the economic and environmental importance of civil engineering, the traditional design process generally leads to low-efficiency structures. A recent study conducted by Moynihan and Allwood on more than 10,000 beams found an average reserve capacity of more than 50%. These observations explain the growing interest in improving the performance of this sector, looking for innovative approaches that lead to a more efficient and sustainable structural design.

This chapter aims to provide the reader with information on structural design by briefly describing the traditional approach normally used in design activities and going on to define all aspects that should be considered when evaluating the design solution.

1.1 Structural design overview

1.1.1 Structural design process

The design process aims to fully define the design for a project in all its aspects, including structural member sizes, static scheme, materials, the equipment. It can be divided in four sequential phases (3):

1. Conceptual design, in which the idea behind the project and a plan for its implementation are established.

2. Schematic design, in which several sketches of design solutions are analyzed based on project objectives and desired outcomes.
3. Design development, in which the different aspects of the chosen design solution are specified in detail, defining how to implement it from a practical point of view and carrying out the necessary checks to ensure that the structural behavior is as required by the design and by the standards.
4. Construction Documents, in which the documents necessary to describe the design solution in minimum detail and containing information about the structural and behavior checks that have been carried out are produced.

In each of these phases, several iterations are carried out, leading to subsequent changes in the design product.

A variety of actors are involved in the various phases: clients, architects, designers, builders. Each has a different role and comes into play at different times in the process. The various professional figures deal with different aspects but are closely interconnected. Consequently, the design process should be the result of cooperation between professionals from different fields. This causes the aforementioned iterations to occur not only between professionals in the same field, but also between groups from different disciplines, as visible in Figure 2. Therefore, it is especially important to have a proper exchange of information between the various working groups. The interdisciplinary nature of the design process also brings out a difficulty related to trying to match the needs of all stakeholders. Actors entering the later stages will be constrained by what was decided in the earlier stages, and this will limit their options. Added to this are difficulties related to meeting the many constraints to which a civil structure design is subject: demands from clients, environmental and topographical constraints, regulatory constraints, and many others.

The transition from one phase to the next involves greater design definition, accompanied by a gradual reduction in design freedom and an increase in design knowledge. While the most significant design impact occurs during the early stages, particularly with conceptual design, structural considerations come into play at later stages, limiting the ability of structural engineers to make major contributions to the design process.

The structural design process involves many domains, such as resistance, deformation, fracture toughness, fatigue, durability, performance, different loading conditions such as static, dynamic, impact, and many stages such as service and construction. Each of the topics mentioned is highly specialized in nature and refers to different engineering fields. As a consequence, the designer often deals with the various aspects in series and not in system and design iterations are very time and cost intensive. Furthermore, the high computational cost and the time required to perform structural analysis makes it difficult to explore different solutions of the problem, limiting the usability of optimization methods.

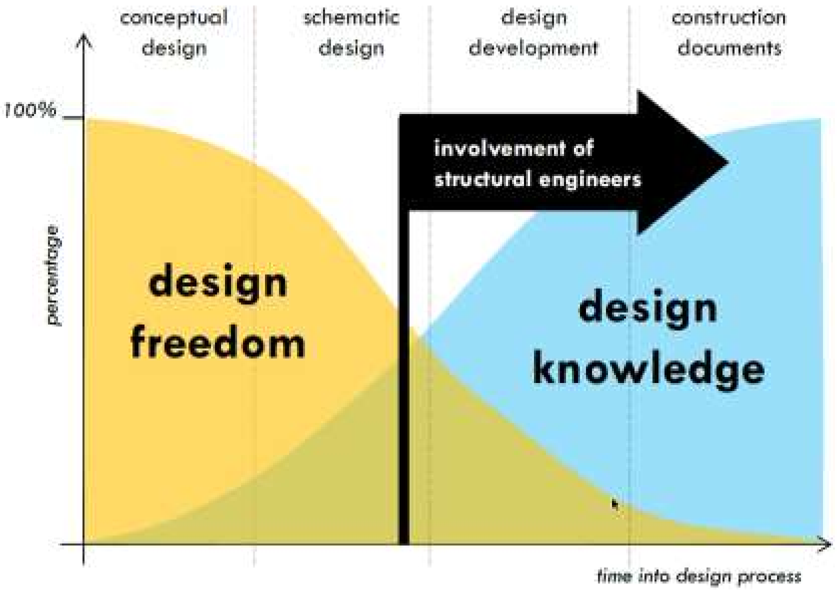


Figure 1. Relationship between design freedom and design knowledge in building design projects (3)

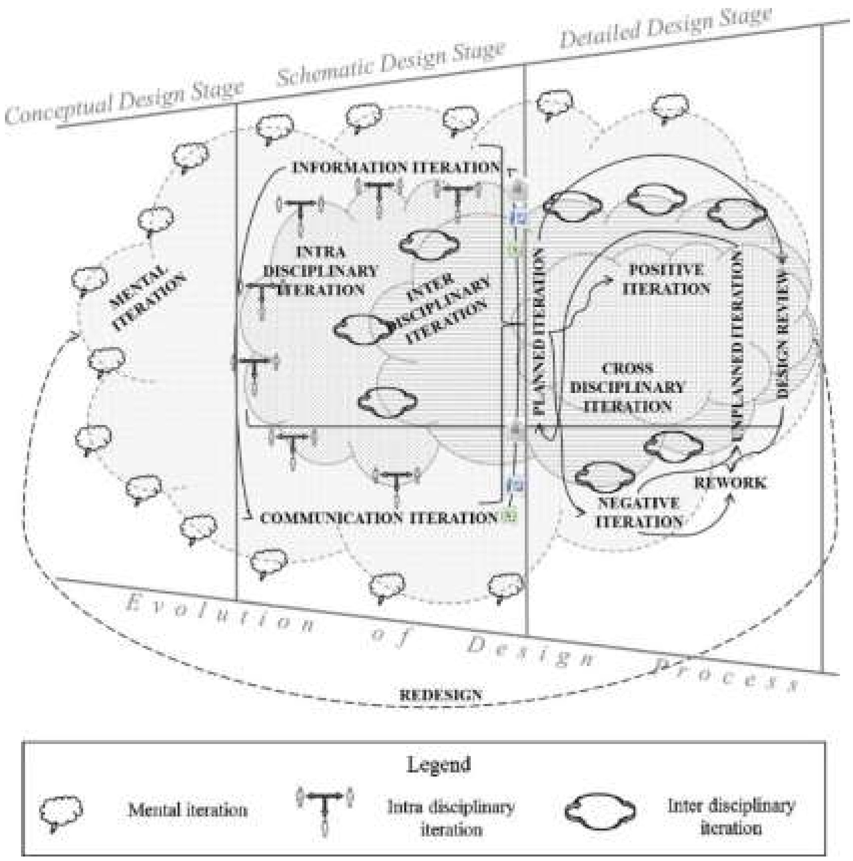


Figure 2. Iteration types occurring in structural design stages (4)

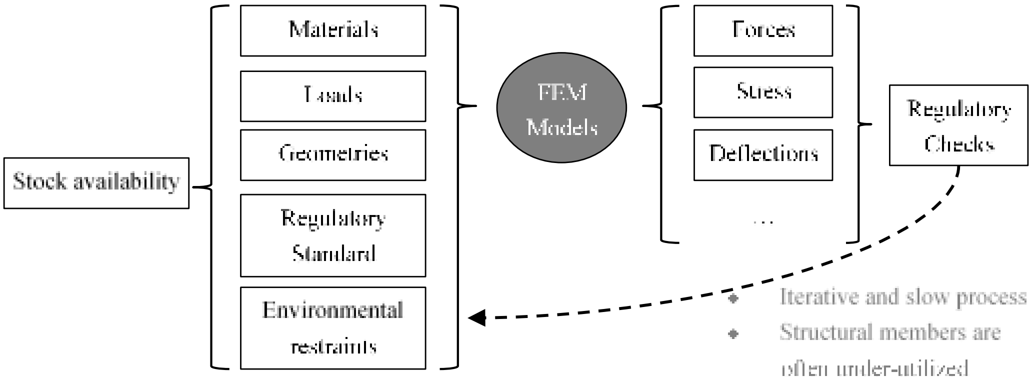


Figure 3. The traditional design of civil structures, often based on a "trial and error" approach

1.1.2 Integrated structural design

Greater involvement of structural aspects in the early stages of the design process, resulting in more opportunity for civil engineers to significantly influence the design, can lead to numerous benefits:

- Reduced construction cost
- Reduced environmental impact
- Aesthetic improvement

- Greater safety

Introducing structural reasoning and notions as early as the conceptual design stage can lead to the development of a design with a more efficient structural form that requires less material and reduced processing for its construction. Reducing the resources used easily translates into economic savings and lower environmental impact. The use of structural skills early in the design process therefore requires that the design solution be the result of a process aimed at harmony between aesthetic and technical goals. In addition to cost-saving and environmental sustainability motivations, the development of shapes based on integrated structural design also generally leads to safer structures due to a more appropriate distribution of internal stresses. There are numerous examples of structures that because they were not designed with structural aspects in mind as well, have turned out to be wasteful, requiring major maintenance and unsafe. One example is Terminal 2E at Charles de Gaulle Airport in Paris, whose shape contributed to the development of high internal forces and the need to use high-strength materials. The structure collapsed in 2004 less than a year after opening, killing four people, and costing 130 million in renovation and replacement costs. (3)

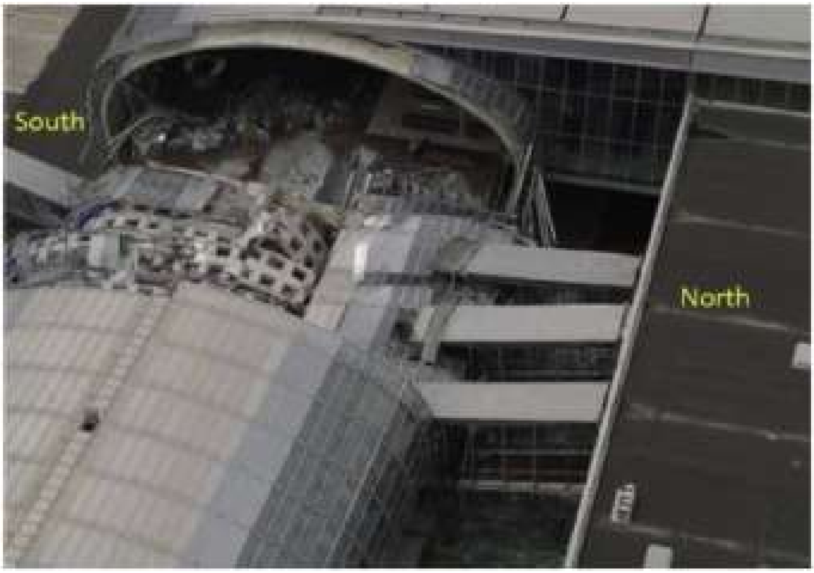
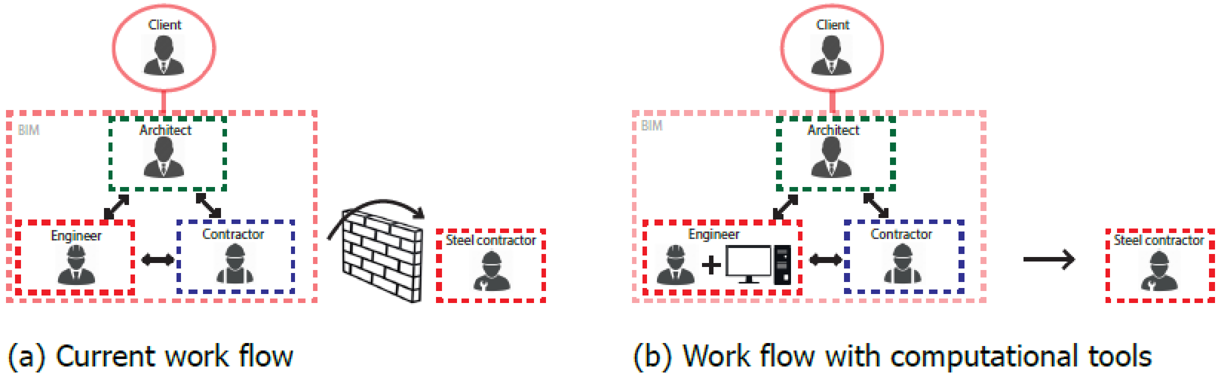


Figure 4. Collapsed zone of Terminal 2E at Charles de Gaulle Airport in Paris (5)

1.1.3 Importance of node design in steel structures

In the case of steel structures, among the various aspects that should be considered in the early stages of design are nodes and joints, which represent the connecting parts between the members that make up a frame. In the common design process, in fact, there is a tendency to view joints and nodes as the business of the steel contractor alone and thus to consider them only roughly in the preliminary design stages. Therefore, in the early stages of design, the structural engineer will focus on designing only the main steel members, in accordance with the constraints imposed by the client, the architect, the regulations, the environment, etc., offering a cost optimization solution that, however, only superficially considers joints and nodes and their influence. The connections are designed in detail later by the steel contractor's engineers, who will try to design them in the best way for their production process.



(a) Current work flow

(b) Work flow with computational tools

Figure 5. Relationships between the parts involved in the design of steel structures using the traditional approach (a) and by introducing tools that allow more correct calculation of joints (b) (6)

The deep influence that the design choices made on the rest of the structure have on the design of joints and nodes by the steel contractor means that designing the structure without adequate consideration of these entities often results in poor choices that can lead to significantly increased costs and/or reduced safety. For example, the pursuit of structural weight optimization can lead to the design of structures with slender beams, which may require reinforcing elements such as plates and stiffeners to achieve adequate structural joint strength, resulting in increased costs. Another solution is to change the cross-section of the beam, choosing a larger one: however, this solution may create problems of conflict with the other designed elements of the structure. Knowledge of the joint type as early as during the design of the entire structure can also lead to other advantages: for example, in the case of semi-rigid connections it is possible to consider the rotational stiffness of the joint, which by influencing the distribution of the bending moment can lead to further cost savings.

Another aspect to consider is the impact of joint costs on the overall cost of the structure. Among the main cost items in a steel structure are manufacturing costs, which include labor and processing costs. The large number of elements that make up the joints, such as plates, stiffeners, bolts, etc., make production costs for these parts of the structure particularly significant. The number of joints is influenced by the shape of the structure. In fact, limitations on the size of elements due to their transportability, or geometric parameters such as the size of spans or the height of a truss beam that determine the number of beams and diagonals, affect the overall number of joints on the structure. Incorrect assessment of the type of joint to be made can therefore lead to significant error in estimating the cost of the structure and thus in the comparative analysis between different design solutions.

It follows from these considerations that it is crucial to include joint design early in the design of the structure in order to achieve a more correct optimization of the solution. Sizing the joints at the same time as the rest of the structure and no longer downstream results in an increase in the variables involved in the design and the creation of more complex computational models. The search for the optimal design solution therefore requires not only adequate computational resources, but also support from appropriate tools so that the result can be achieved in a reasonable time. Artificial intelligence offers methods that lend themselves well to the creation of such tools.

Little research on structural node optimization can be found in the literature, but it is mostly limited to very simple nodes where few elements converge. Among them, R. da S. Hortencio and G. A. S. Falcón (7) used the genetic algorithm to optimize a beam-column node using an objective function that considers both cost and compliance with the performance and dimensional requirements of the standard. C. Diaz et al. (8) also used the genetic algorithm to optimize the same type of node, with the addition of creating metamodels to quickly evaluate cost, resistant moment, and rotational stiffness without having to build finite element models. The node used for this research is given in Figure 6. However, beam-column nodes are extremely simple compared to the multiway nodes that can be encountered in a complex framed structure characterized by many beams converging at the same node. An example of this type of node is shown in Figure 7. It is clear that the number of parameters required to characterize multiway nodes is much greater than those required for a simple beam-column node. The increase in the size of the solution search space therefore makes the search for the optimal solution more complex and onerous, and optimization studies featuring this type of node are not found in the literature. In addition, the state-of-the-art analysis revealed the absence of sizing and optimization tools involving the entire structure, that is, both the beams and the structural details that make up the connecting elements between them. (6)

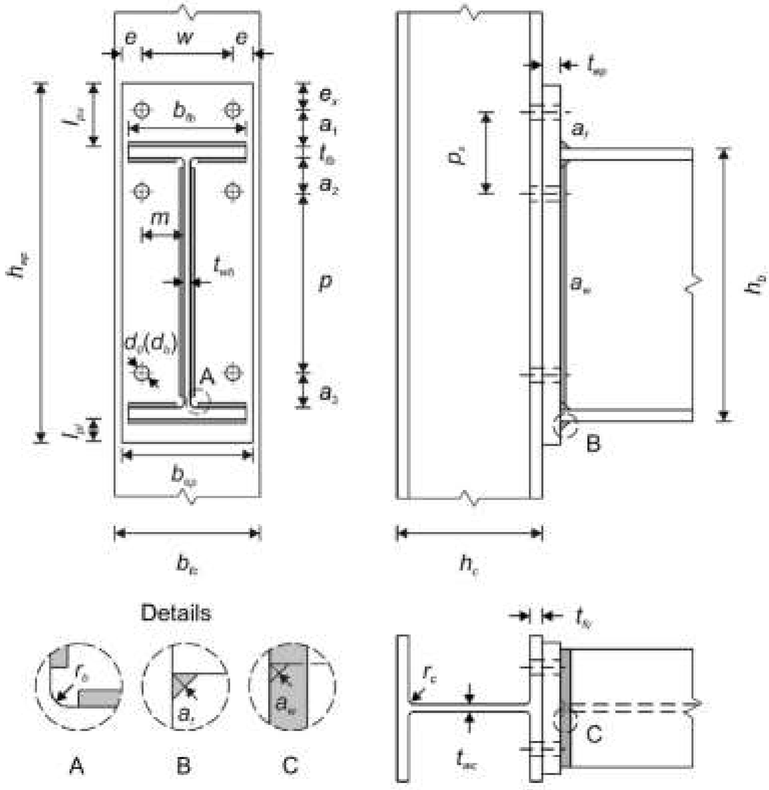


Figure 6. Geometric parameters of beam-column connection (8)

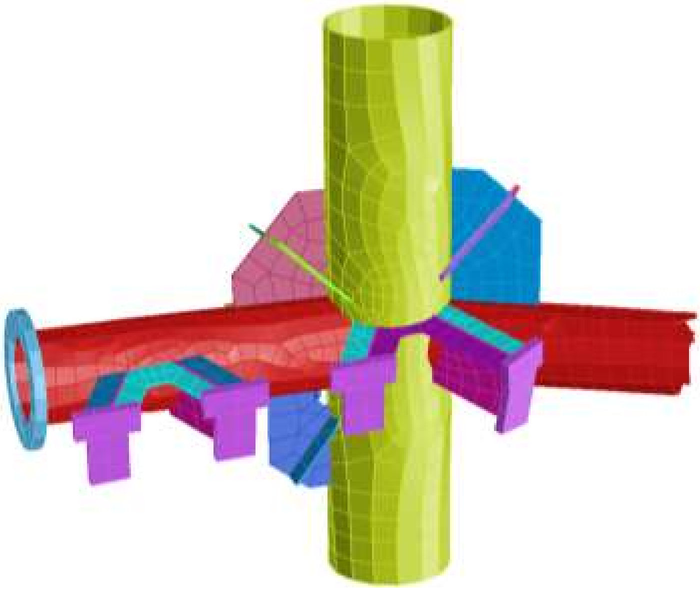


Figure 7. Example of FEM model of a multi-way node

1.1.3.1 Structural node design with local models

Nodes are composed of numerous elements: plates, bolts, welds, etc. All these elements must undergo the structural checks required by the standards. Calculation of the distribution of the input forces at the node among the various elements under different loading conditions requires detailed modeling of the node, using two-dimensional FEM elements. Such kind of details cannot be included in the global model of a steel megastructure, which for reasons of computational cost represents a very simplified version of the real structure, including almost exclusively the beams as one-dimensional elements. Thus, the study of nodes requires the creation of a detailed local model in which to represent all the elements of the node in order to correctly assess its structural behavior and force distribution and to perform structural checks. The loads applied to that local model results from the global model of the megastructure. However, because they are not part of a single model, the connection between the global model and local models is not direct and managing the design process requires special attention. The mutual dependence between global model and local models therefore makes the structural analysis process iterative, with changes to a local model resulting in changes to the global model and vice versa. In fact, any change to the global model after node checks also requires an update of local model loads and the performance of new structural analysis. Conversely, if the node checks lead to the conclusion that it is necessary to change the section of a beam, the global model will have to be modified and recalculated and the local models updated.

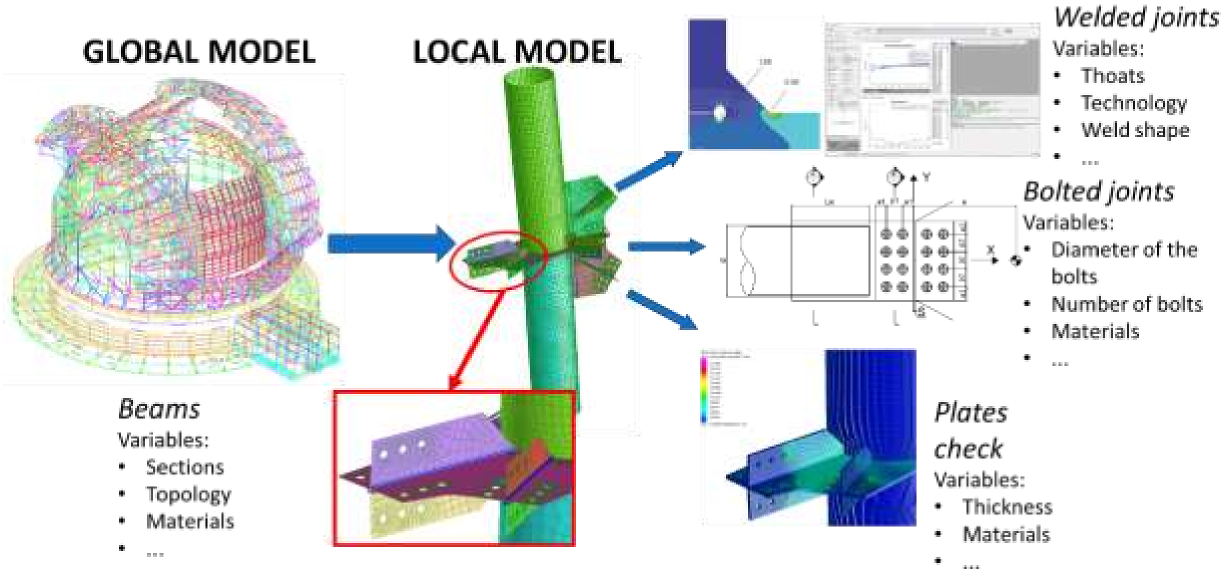


Figure 8. Example of design framework for a steel megastructure

1.1.4 Cost of steel structures

The goal of design should be to seek the solution that meets the safety and performance requirements of the customer at the lowest cost. These aspects must be considered simultaneously during the investigation among candidate solutions. In fact, considering only safety could lead to overly large and heavy structures and/or difficult to execute. On the other hand, considering only cost could lead to a structure that is not robust and therefore not sufficiently secure. The outcome of the design process must therefore be a compromise between these aspects. The engineering decision-making process to find the design solution among those that meet performance and safety criteria must be guided by the goal of minimizing the overall cost and not the total weight of the structure. In fact, the presence of structural nodes, with their high number of elements and the large amount of labor and workmanship they require, makes the relationship between the weight of the structure and its cost not necessarily monotonically increasing. Consequently, it is necessary to identify all the cost items that come into play in the design and construction of a structure with the goal of creating a cost function to guide the engineer during the optimization process. Identifying all cost items, as well as estimating them, is very complex, particularly in the early stages of design where knowledge of the final structure is more limited, especially of aspects related to its execution.

Costs can be divided into two macro-categories: direct costs and indirect costs. The first group includes cost items that are easily attributable to a cost objects, such as the cost of materials, workers' wages, transportation costs, and cutting and painting costs. Costs, on the other hand, not directly attributable to cost objects are part of indirect costs. These include energy consumption, machinery maintenance costs, depreciation of factory and machinery, and wages of factory managers.

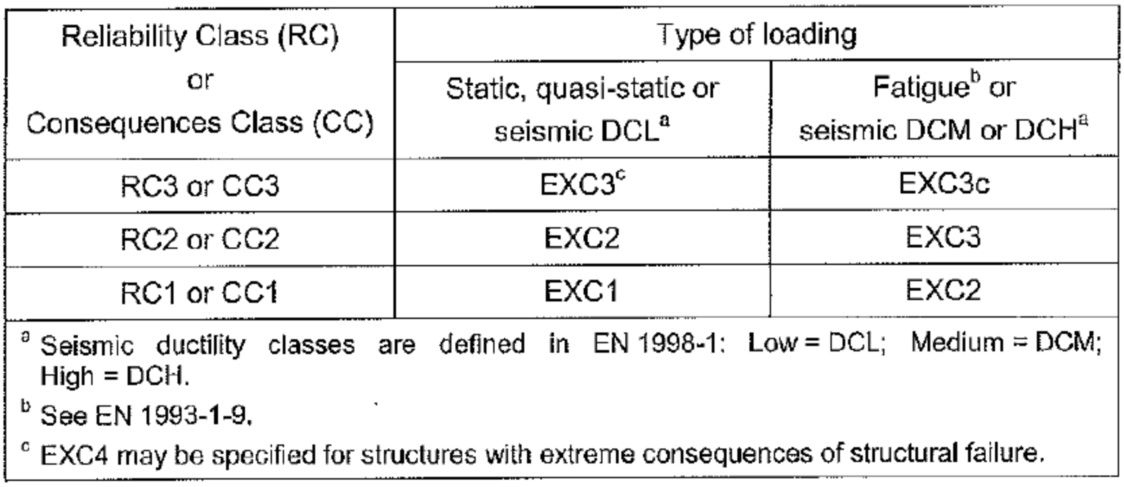
Material costs depend on the size of the elements to be made and the price of the material. The latter is a function of the price of raw material, which is defined by steel markets, and the volume of material to be purchased.

Direct cost items include labor cost, which depends on the number of man-hours and hourly wages. The latter varies by geographical area, especially a strong dependence is observed with the economic condition of the area. In particular, a better economic situation leads to higher wages. This correlation also led to an increase in the cost of hourly wages over time in the aftermath of World War II due to improved economic conditions. Variation in the cost of workers' wages between geographic areas also leads to a difference among countries in the ratio of labor costs to material costs. As a result, depending on the geographical location, it may be convenient to use more of the material to avoid extra labor or vice versa.

Speaking of the costs of building a structure, a classification can be carried out on a chronological basis: we have manufacturing costs, transportation costs, and on-site assembly costs. In all three cost categories we can identify items dependent on the sizing of the structural elements. Fabrication costs are a function of the geometric characteristics of the structure. In fact, they determine the number and extent of machining operations to be performed, with a consequent impact on the amount of resources to be used for their realization, such as labor, the number of machines and the amount of material needed to make them. The size and shape of the elements that make up the structure affect the type and number of means of transportation to be used to carry them from the workshop where the individual parts are made to the place of assembly. The geographic location of the workshop and the assembly site can also influence the choice of means of transportation, and, consequently, this should be taken into consideration when designing the structure. Finally, assembly costs depend mainly on the time required to assemble the structure and the means required to carry out the assembly. The storage and assembly speed of individual members is generally not proportional to their size. In addition, the shape and size of the elements can influence the type of maneuvers to be performed, the means to be used, and the time required to perform them. These are additional aspects that should be considered when estimating the cost of the overall structure.

Other aspects to be considered in estimating the overall costs of a structure concerns the number of repetitions of the same operation: in fact, repetitive execution of an operation results in greater efficiency of the process and, therefore, in reduced risk of errors.

Where welds are present, the overall cost of a structure also depends on its class of execution, in accordance with EN 1993-1-1:2005+A1:2014 (9). This standard has four execution classes, from EXC1 to EXC4, with which there is an associated increasing demand for weld quality. The execution class associated with a structure or part of a structure depends on the class of consequences and the type of loads to which it will be subjected. (6)



Reliability Class (RC) or Consequences Class (CC)	Type of loading	
	Static, quasi-static or seismic DCL[a]	Fatigue[b] or seismic DCM or DCH[a]
RC3 or CC3	EXC3[c]	EXC3c
RC2 or CC2	EXC2	EXC3
RC1 or CC1	EXC1	EXC2

[a] Seismic ductility classes are defined in EN 1998-1: Low = DCL; Medium = DCM; High = DCH.

[b] See EN 1993-1-9.

[c] EXC4 may be specified for structures with extreme consequences of structural failure.

Figure 9. Table defining the execution class (EXC) (9)

The large number of factors influencing the overall cost of a structure makes it difficult to develop a cost function for use in the design phase when looking for the cost-optimized design solution.

1.1.5 Design tools

In the design process, designers use software as tools to support their experience and knowledge. This software can be divided into two categories according to their function:

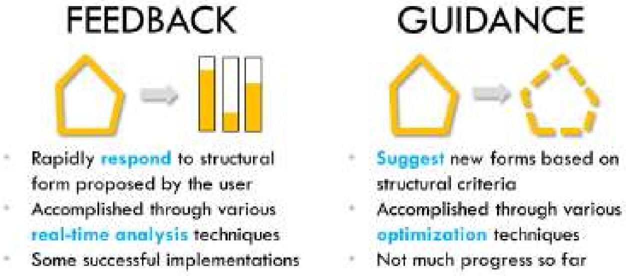
- Geometry software
- Structural software

Geometry software are tools that allow 2D or 3D visualization of the geometry of the structure, enabling simulations of design ideas to easily evaluate aspects of aesthetics and feasibility. They have gradually replaced manual drawings and allow the measurement of quantities and overall dimensions quickly. There are several types of geometric software on the market. They can be distinguished according to the level of detail of information they contain: in the simplest ones, a schematization of the actual structure is represented, while the more detailed ones contain all the information that enables the execution of the structure. In addition, this software can also be distinguished according to whether it gives those involved in building the model the opportunity to work on the same model simultaneously through real-time updates of any changes. Some geometry software also allows parametric modeling, based on the relationship of parts/components to each other and/or to parameters. This type of modeling promotes automation of the process and allows the designer to simultaneously evaluate several possible solutions and easily turn any changes. An example of a parametric modeling environment is given by *Grasshopper*, an internal platform for *Rhinoceros*, a 3D CAD software.

Structural software, on the other hand, arises with a different purpose than geometric software, namely to investigate the performance of the structure to assess compliance with the requirements of the standards and other project requirements. Generally, these software programs are based on the Finite Element Method (FEM), a numerical technique for finding approximate solutions to problems described by partial differential equations. There are several types of these software on the market as well, but they all have in common in their use the presence of two basic steps: modeling and discretization. During modeling, reality is filtered in order to obtain a mathematical model with a limited number of variables. The level of detail and the type of elements used in modeling depend strongly on the purpose of the analyses and can heavily affect the results obtained. This model is discretized to obtain a finite number of degrees of freedom. A mesh is thus obtained consisting of elements for each of which the solution is expressed as a linear combination of shape functions. The quantities obtained through the FEM, such as stresses and strains, can then be used for structural checks. It is clear that the creation of a structural model requires deep engineering knowledge of the structure to be analyzed and of the relevant standards.

Both types of software play a key role in the design process, enabling the user to achieve better results and in less time. Most software is either geometry-driven or analysis-driven. However, the mutual dependence between geometric and structural aspects is leading to the need to try to bridge this gap. Some tools have already been developed with this purpose in recent years. Many of these are internal plug-ins to existing geometric software for performing structural analyses directly in the geometric model. An example is given by *GeometryGym* in *Rhinoceros* or *Robot* in *Revit*. This approach brings with it important issues. From a practical point of view, they can only be limited to users of the parent geometric software. From a theoretical point of view, the difference in objectives between structural and geometric models makes the translation of reality into these two models different, so it is generally problematic to confuse a geometric model with a structural one. A different approach, however, involves the creation of tools that enable a link between geometric and structural software so as to make it easier the exchange of information between the two models, thereby facilitating alignment between the models. For example, *Grasshopper* has developed internal tools to link *SOFiSTiK* structural software with *Tekla* BIM software. This makes it possible to import the Tekla model with all properties into Grasshopper and use this information to define already in the CAD environment the attributes needed for structural calculations, such as materials, beam sections and loads, to be transferred directly into a project in *SOFiSTiK*. In addition, it is possible to take advantage of the "parametric" approach to associate structural attributes with geometric entities so that structural analyses can be updated more quickly in the event of changes to design geometry.

All of these new tools allow faster feedback to the designer, but they do not change the current design process. In fact, with this software the sequence of defining the geometry and performing the structural analysis remains the same, resulting in a "try and error" approach. A turning point in the design process is the creation of software that guides the designer according to structural principles, resulting from the early stages in a design solution that is closer to the final solution and thus reducing the iterations that the design process normally involves. (3)



FEEDBACK

- Rapidly respond to structural form proposed by the user
- Accomplished through various real-time analysis techniques
- Some successful implementations

GUIDANCE

- Suggest new forms based on structural criteria
- Accomplished through various optimization techniques
- Not much progress so far

Figure 10. Types of structural design tools evolution (3)

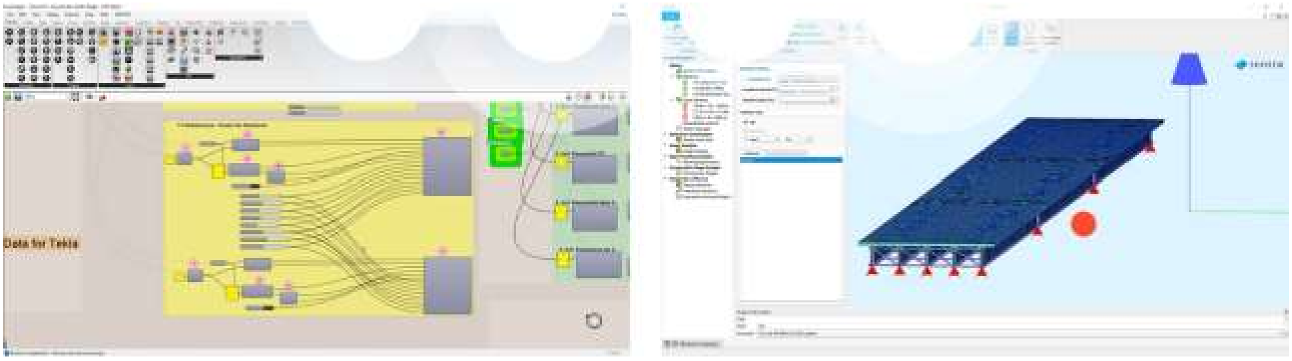


Figure 11. Example of integration between BIM (Build Information Model) and structural calculation: through the Grasshopper platform (left window) the BIM software Tekla is connected to the FEM software SOFiSTiK so as to facilitate the exchange of information between the two software (10)

1.2 Artificial Intelligence overview

This section aims to provide the definitions of Artificial Intelligence and Machine Learning.

1.2.1 Artificial Intelligence (AI)

Artificial Intelligence (AI) is a branch of computer science that designs, develops, and builds systems capable of imitating characteristics and skills typical of human beings, enabling them to make decisions independently. The term *Artificial Intelligence* was used for the first time at a meeting in Dartmouth College. However, the emergence of the expert system led to a considerable growth of interest in Artificial Intelligence from the late 1960s to the 1970s. Subsequently in the 1980s the development of fifth generation computers caused a new increase in research on AI. In the 1990s the development of network technologies led to a new surge in AI research and its use in a network environment. Furthermore, there has been a growing interest in applying AI techniques in new fields. (11)

1.2.2 Machine Learning (ML)

Machine learning (ML) is a branch of AI with the aim of developing tools capable of learning from data and providing predictions based on the knowledge learned without being explicitly programmed. In traditional algorithms, it is the coder who writes the rules that the software must follow, and it is therefore his responsibility to consider all possible scenarios. In ML techniques, on the other hand, it is the algorithm itself that creates the rules, leaving the coder "only" responsible for creating the architecture and providing the data from which the algorithms can learn. The comparison between the philosophy of traditional and ML software is shown in the Figure 12.

ML can be classified into:

- supervised learning
- unsupervised learning
- reinforcement learning

Supervised learning aims to build classification or regression models based on a data set for which the label is known. Instead, the unsupervised learning aims to subdivide a dataset into clusters without their label being available. In reinforcement learning the data labels are not available and the creation of the models is achieved through the use of an agent that can perform actions based on the current state of the environment and which themselves modify the environment. (12)

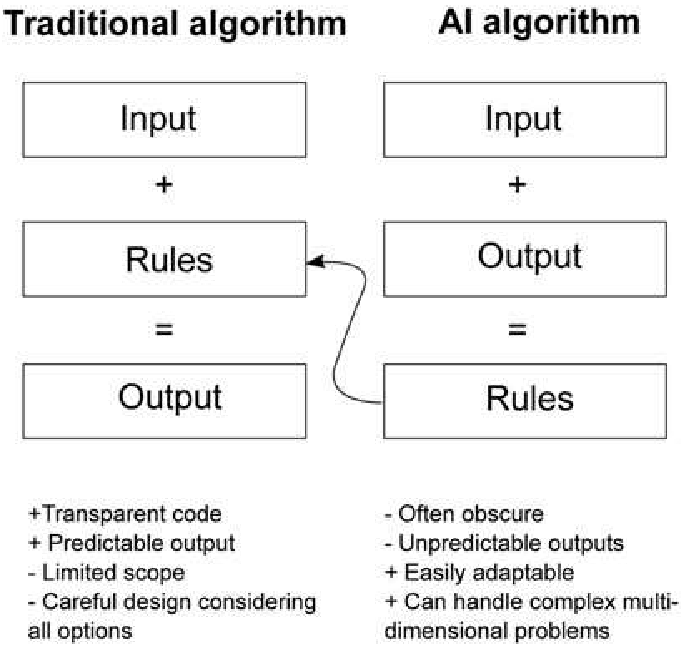


Figure 12. Comparison of traditional and Artificial Intelligence algorithms (13)

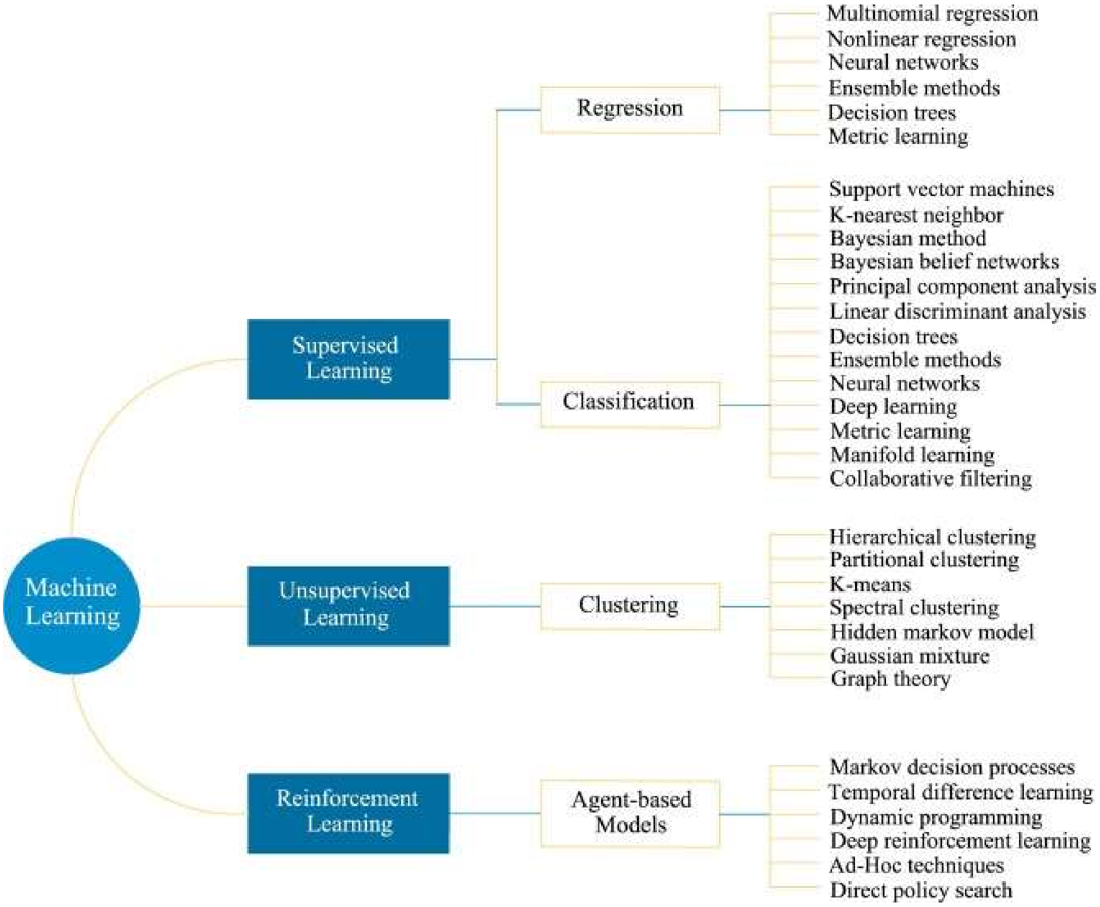


Figure 13. Common Machine Learning algorithms (12)

1.3 Artificial Intelligence in Civil Engineering

As in other fields, AI techniques have also attracted interest in civil engineering. In the last 30 years, a lot of research has been carried out in this topic and several applications have been realized, for example in the evaluation of the characteristics of materials. For example, Bassuoni and Nehdi have developed a neuro-fuzzy system capable of predicting the durability of self-consolidating concrete at various exposures to sodium sulphate. (14) Prasad et al. have developed an artificial intelligence network (ANN) that makes it possible to evaluate the resistance to complexion at 28 days of normal and high-strength self-compacting concrete and high-performance concrete with high volume fly ash. (15) However, the use of Artificial Intelligence in structural engineering as an optimization tool is very limited, especially compared to other engineering fields.

The research conducted to date has led to the development of optimization methods limited to simple cases and that do not include all aspects involved in structural design, such as those carried out by Farkas and Jàrmai (16). Dillen et al. (17) developed a nested optimization approach for a steel building that consists of two separate algorithms for discrete and continuous variables: a meta-heuristic method is used for the former while a gradient-based algorithm is used for the latter. Another interesting work is the one developed by Chang and Cheng (18), in which with the goal of optimizing a framed building, two graph neural networks were used, respectively, one providing the optimized sections of the members while the other performing the function of a surrogate model, updating the displacements of the structure without the aid of a FEM model. The variables considered by this research are limited to beam cross-section parameters, the number of braces in each plane, and topological data. In contrast, joint and node parameters are not part of the variables, and dynamic responses and fatigue verifications are not considered. Another example is provided by the research of Dìaz et al (8) who developed a method of node optimization based on the use of metamodels but limited to very simple cases. These methods therefore do not appear to be sufficiently general to be applied to the design of a megastructure. An optimization method that allows a structure to be fully optimized, considering simultaneously the different elements that compose it and the various phenomena that affect its performance is currently missing. The large number of variables and equations involved, combined with the Standards constraints, make the objectives of the optimization problem difficult to express in mathematical terms. It follows that nowadays the use of optimization techniques in civil engineering appears much more limited than in other engineering disciplines and human intuition still plays a crucial role in the design process.

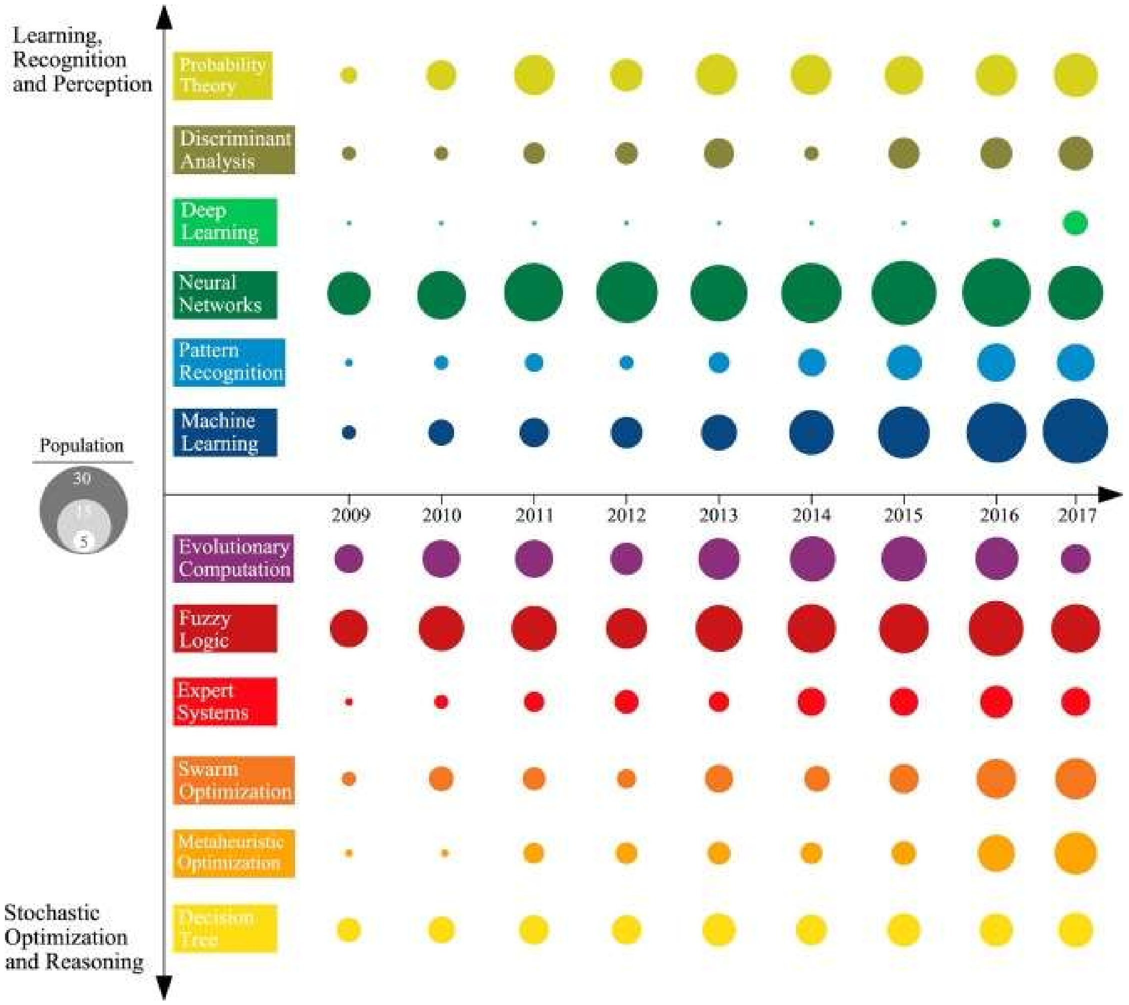


Figure 14. Research publications on the use of AI branches in civil engineering (12)

1.4 Conclusion

The analysis of the structural design problem brought out the large number of variables that come into play and should be considered to achieve an optimized, effective and efficient design solution. The large size of the problem, coupled with the variety of skills and professional figures involved, does not allow the traditional design approach to consider all the variables of the problem simultaneously, thus involving numerous design iterations and revisits to arrive at the final solution. The following chapters will investigate the possibility of developing tools that reduce these iterations, modifying the design approach so as to have a solution closer to the final solution at an early stage. The need to quickly find the solution to the design problem clashes with the onerousness of finite element analysis. This has prompted consideration of the possibility of creating surrogate models that provide an estimate of results more quickly than the FEM solver. An overview of surrogate models is presented in Chapter 3, and different approaches to creating them are studied and applied in the Chapter 5.

As pointed out in the section 1.3, the research and development of AI in civil engineering is only at the beginning, but the growing interest in it and the potential that resides in the tools it can offer suggest that it will play an important role in the future of this field. Not only the technology offered by AI can be a support to inexperienced designers to solve engineering problems, but also for the most experienced users it can represent a valid help to increase work performance.

CHAPTER 2: The search for the optimum

Optimization problems, in which the goal is to find the best solution to a given problem, can be found in all scientific fields. For example, in economics there is the problem of defining the best price of an item to be introduced into the market, in finance there is a need for evaluation of the composition of a portfolio to minimize risk while maximizing profit, in transportation engineering there is the problem of defining the optimal travel route for a bus or train or plane, etc., in structural engineering you want to find sections for beams that reduce costs while keeping structural requirements satisfied. The high prevalence of this type of problem has prompted research to find tools that facilitate finding the best solution among various alternatives.

After providing a description of the optimization problem in general, this chapter gives an overview of the techniques usually used to solve this type of problem. The chapter concludes by discussing the current application of optimization techniques to structural design problems.

2.1 Optimization problem

The optimization is a numerical problem that aims to find the best solution in compliance with the requirements expressed in mathematical form (*objective function*) and respecting the mathematically formulated constraints. The search for the optimized solution results in finding the configuration of a set of parameters, called *design variables*, which allows to solve the mathematical problem. From a mathematical point of view, the optimization problem can be defined as follows:

$$\begin{gathered} \min f(x) \\ \text{subjected to} \quad g(x) \leq 0 \qquad h(x) = 0 \qquad x_{i,b} \leq x_i \leq x_{i,ub} \end{gathered} \tag{2.1}$$

where $f(x)$ is the objective function, x is the vector of design variables, $g(x)$ and $h(x)$ are the constraints and $x_{i,b}$ and $x_{i,ub}$ are respectively the lower and upper limits of the variables.

Objective functions *f(x)* are the relationships between the design variables that describe the mathematical problem to be minimized. This mathematical definition also includes case in which the configuration of variables leading to a maximization of the function is sought. In fact, in this case the negative of the function can be considered as the objective function.

The high number of problems that can be translated into an optimization makes the mathematical formulations of optimization problems very different from each other. The type of objective function and design variables influences the type of solving techniques that can be used. In some cases, the objective function can be expressed in closed form, and if it has continuity and derivability characteristics, information about its gradient can be exploited to solve the problem. In other cases, the objective function is unknown (black-box function) and stochastic approaches should be used. The vector of design variables can be populated by continuous or discrete variables. In the latter case we speak of a *combinatorial optimization problem*, in which the solution is sought among a finite or infinite numerable set of candidate solutions. Even in the case of continuous variables, the value they can take can be limited within a range. Some optimization problems require the objective function to take values that satisfy mathematical equations or disequations. The presence of these constraints limits the range of the solution search. This type of problem is called *constrained optimization*, as opposed to *unconstrained optimization* in which these constraints are not present. An easy way to trace back an unconstrained problem from a constrained one involves using penalty coefficients to be applied to the objective function.

2.2 Gradient-based optimization

Gradient-based algorithms exploit information derived from the gradient of the objective function in the search for the optimal solution. As a result, these techniques lend themselves well to the solution of problems in which the objective function has characteristics of continuity and derivability. For functions without such characteristics, an approximate value of the gradient can be obtained by the finite difference method. Using the gradient to calculate the new position of the candidate solution in the space of admissible solutions can be done by several techniques. The choice is strongly related to the type of problem addressed. For example, in the case of unconstrained problems it is possible to use the quasi-Newton method for updating the variable.

For each iteration, updating the solution requires calculating or estimating the partial derivatives of the objective function for each variable parameter. These derivatives can be written in a vector with a number of entries equal to the number of variable parameters and representing the gradient of the objective function relative to the current position $\bar{x}$:

$$\nabla f(\bar{x}) = \left\{ \begin{matrix} \frac{\partial f}{\partial x_1} \\ \cdots \\ \frac{\partial f}{\partial x_n} \end{matrix} \right\} \qquad \text{with } \bar{x} = (x_1, \dots, x_n) \tag{2.2}$$

To approximate the gradient calculation using the finite difference method, this expression becomes the following:

$$\nabla f(\bar{x}) \cong \left\{ \begin{matrix} \frac{f(\bar{x} + \Delta x_1) - f(\bar{x})}{\Delta x_1} \\ \cdots \\ \frac{f(\bar{x} + \Delta x_n) - f(\bar{x})}{\Delta x_n} \end{matrix} \right\} \tag{2.3}$$

The position is updated in the opposite direction to that expressed by the gradient. At step i-th we can write it as follows:

$$x^{i+1} = x^i - \alpha * \nabla f(x^i) \tag{2.4}$$

where α is a parameter that modulates the update of variables according to the gradient. This is also done for subsequent iterations until the optimal solution is reached, at which it is no longer possible to define a direction of change of variables that leads to an improvement in the value of the function *f*. (19)

The calculation of partial derivatives for each variable makes the computational cost of this method closely related to the number of variables in the problem. To reduce this cost, it is possible to decrease the number of locations close to the current one at which to calculate the value of the objective function by considering block of variables. This requires the use of appropriate algorithms to derive the value of the partial derivatives for each variable so as to define the update value for each. The estimation of these derivatives can be improved by investigating the value of the function along different directions, obtained by changing the composition of the blocks of variables considered together and/or the direction of change in the value of each variable.

One problem with gradient-dependent methods is the possibility of running into local minima, as can be seen in Figure 16. In fact, unless concave functions, these techniques do not guarantee the achievement of a global optimum but may lead to local minima. To try to limit this problem, it is necessary to perform a few tricks such as using different starting points or act in the variable update algorithm.

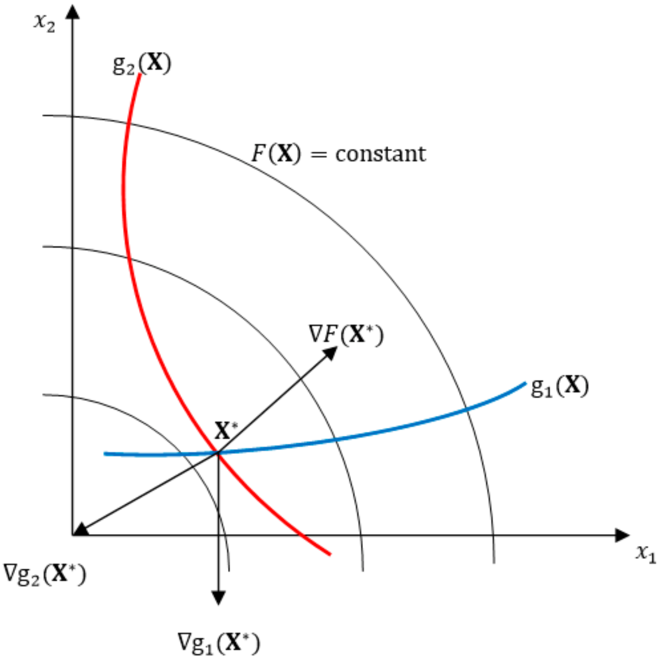

x_2
$g_2(\mathbf{X})$
$F(\mathbf{X}) = \text{constant}$
$\nabla F(\mathbf{X}^*)$
$g_1(\mathbf{X})$
$\mathbf{X}^*$
x_1
$\nabla g_2(\mathbf{X}^*)$
$\nabla g_1(\mathbf{X}^*)$

Figure 15. Kuhn–Tucker condition at a constrained optimum (19)

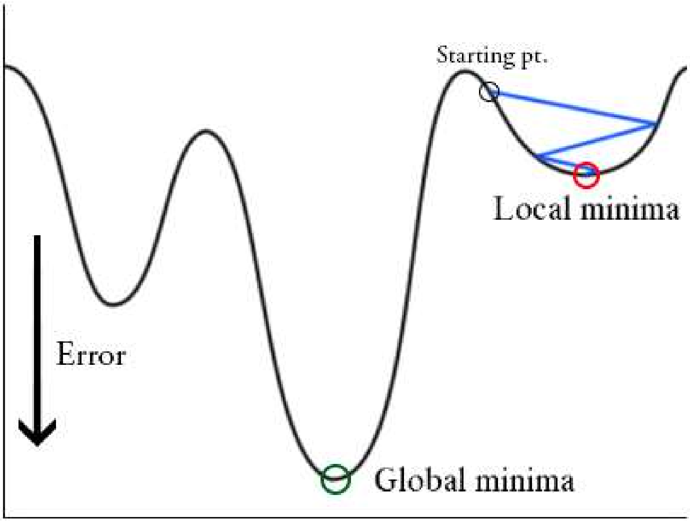


Figure 16. Gradient descent stuck at local minima (20)

The use of gradient-based methods for structural optimization is almost absent. One of the few examples is the research of Dillen et al. in (21), which used the gradient-based technique only for continuous variables while for discrete variables they relied on meta-heuristics.

2.3 Meta-heuristic optimization

Heuristic methods explore the search space with the support of stochastic operators in a pseudo-random manner. Some of these algorithms take inspiration from biological evolution, the most famous of which is the *Genetic Algorithm (GA)*. The starting point for these algorithms is a population of possible candidates for the solution of the problem, usually generated at random. Such individuals are evaluated based on a fitness function and the best ones are used as the basis for a new population. Evolution operators, generally of the stochastic type, such as crossover and mutation, are used to generate a new population of candidates that will be evaluated on the basis of the fitness function. The process continues iteratively until the end criterion is met. (11) Examples of using this algorithm for structural optimization can be found in the literature but limited to simple unrealistic problems. An attempt to use of this approach in a large steel frame structure was carried out by Dillen et al. (17). Their goal was to minimize the volume of the beams of the Market Hall in Ghent by imposing as constraints that the utilization rates of the various elements be less than the desired limits. To carry out the analysis, however, the solution search space was reduced by dividing all the rods into 13 groups and imposing that all elements belonging to the same group have the same cross section. The genetic algorithm has also been used for the optimization of simple beam-column nodes in the research by R. da S. Hortencio and G. A. S. Falcón (7) and in the research by C. Diaz et al. (8) mentioned in section 1.1.3

Other algorithms are based on populations of particles and interactions between them and their surroundings. These include *Particle Swarm Optimization (PSO)* and *Ant Colony Optimization (ACO)*.

Particle Swarm Optimization (PSO) is a population-based global optimization technique which draws inspiration from the movement of swarms. This technique involves the presence of particles, representing potential solutions, which move in the problem space following the current optimal particle. Each particle is associated with a position vector and a velocity vector, which are modified at each iteration on the basis of the best current local solution found by the particle itself and on the basis of the best current global solution found by the entire swarm. R. Yanzhi and L. Sanyang (22) have presented a Modified Particle Swarm Optimization algorithm (MPSO) to solve structural optimization problems. In this algorithm, a teaching mechanism is used to guide population evolution in the feasible region, and adaptive hyperparameters are adopted to balance exploration and exploitation. Finally, a particle zeroing strategy is used to avoid entry into a local optimum. Again, the method was applied in a small case study with only four variables involved.

Ant colony optimization (ACO) is an algorithm that takes inspiration from the foraging of ants in a colony. In each iteration, an ant travels from the nest to food and returns to the nest. During its journey it lays a substance called pheromone, the concentration of which depends on the distance of the path between nest and food and on the quality of the food. The higher the concentration of pheromone along a path, the more likely it is to be chosen by an ant. Each ant selects the path to take probabilistically based on the concentrations of pheromones and a heuristic value such as the value of the objective function.

The high level of abstraction of meta-heuristic algorithms allows them to adapt to a wide range of problems. The ease of implementation and the applicability to combinatorial optimization problems have favored the spread of these techniques, which have found application in numerous optimization problems, including some belonging to civil engineering. However, heuristic methods, unlike gradient-based methods, do not guarantee the achievement of the optimal solution and they appear inadequate for high-dimensional problems, for which it would take a long time to reach convergence.

2.4 Optimization in structural design

In civil engineering field, the objective function is often represented by the weight of the structure or its cost, but it could also be other quantities such as the strain energy.

Currently, their applicability is limited to rather delimited fields and relatively simple problems.

Three types of optimizations:

- Size optimization, in which the variables are the parameters of the cross sections of the elements
- Shape optimization, in which the variables are the nodal coordinates
- Topology optimization, in which the optimized solution is achieved by adding or eliminating elements connecting the nodes of the structure
- Multi-objective optimization, in which more than one of the above optimizations are performed simultaneously

Optimization was initially applied in component design in the automotive and aerospace industries, where the optimized component was replicated hundreds of times resulting in significant cost savings. It is currently an established and widely used method for high performance engineering design. However, in civil engineering the involvement in the design of numerous aspects, such as characteristics of materials, costs, safety, aspects related to constructability and assembly, combined with the constraints imposed by the standards, make the objectives of the optimization problem difficult to express in mathematical terms. In addition, the design space increases exponentially as the number of structural elements that make up a structure increases, making common optimization techniques impractical. It follows that nowadays the use of optimization techniques in civil engineering is much more limited than in other engineering disciplines and human intuition still plays a crucial role in the design process.

The high computational cost and the time required for carrying out the analyzes and simulations required in the design of civil structures makes it difficult to explore different solutions to the problem. This aspect represents a significant limitation in the search for the optimal solution through the traditional approach to design. Speed is a key aspect for solving optimization problems, both to promote the designer's competitiveness and to allow adequate exposure in the research field of the solution. It can be achieved in different ways such as:

- increase of computational resources and adequate configuration of the same, such as with a resolution in parallel
- correct writing of the optimization algorithm
- use of surrogate models for a more rapid evaluation of possible candidates for the solution of the problem

2.5 Conclusion

The optimization problem lends itself well to describing problems in numerous fields, including the structural field. The complexity of the structural design problem, already highlighted in the previous chapter, has so far severely limited the application of these tools in the design field. It is precisely to try to develop useful optimization techniques for the optimization of real steel structures that CHAPTER 4: and CHAPTER 5: are aimed at.

CHAPTER 3: Surrogate models

The problems faced by structural engineering often require computationally expensive simulations to be performed. In the case of megastructures, the high cost is due to the large number of elements that make up the model, as well as the high number of load cases to be analyzed, both related to the operational and assembly phases. The resulting high computation time is an obstacle to considering different design solutions and, consequently, to pursuing the goal of optimized design. The problem of high computational cost, however, is not only found in global models of large structures. In the study of particular phenomena controlled by nonlinear laws, such as buckling or plasticity, it is necessary to adopt nonlinear solvers, which, in addition to requiring special care in modeling and parameter setting, need high computation times to achieve convergence. To overcome the problems arising from the onerousness of structural simulations, efforts have been made in recent years to create surrogate models that predict structural performance while limiting or even avoiding the use of FEM solvers.

This chapter begins by discussing the various possible approaches that can be applied in the structural field to simplify the description of the real problem and reduce the computational burden, while maintaining an adequate level of accuracy of the analysis results. Next, an overview of surrogate models and their creation is provided. Among these, *Neural Networks* are illustrated in particular, with emphasis on *Graph Neural Networks*, which are particularly suitable for describing frame structures.

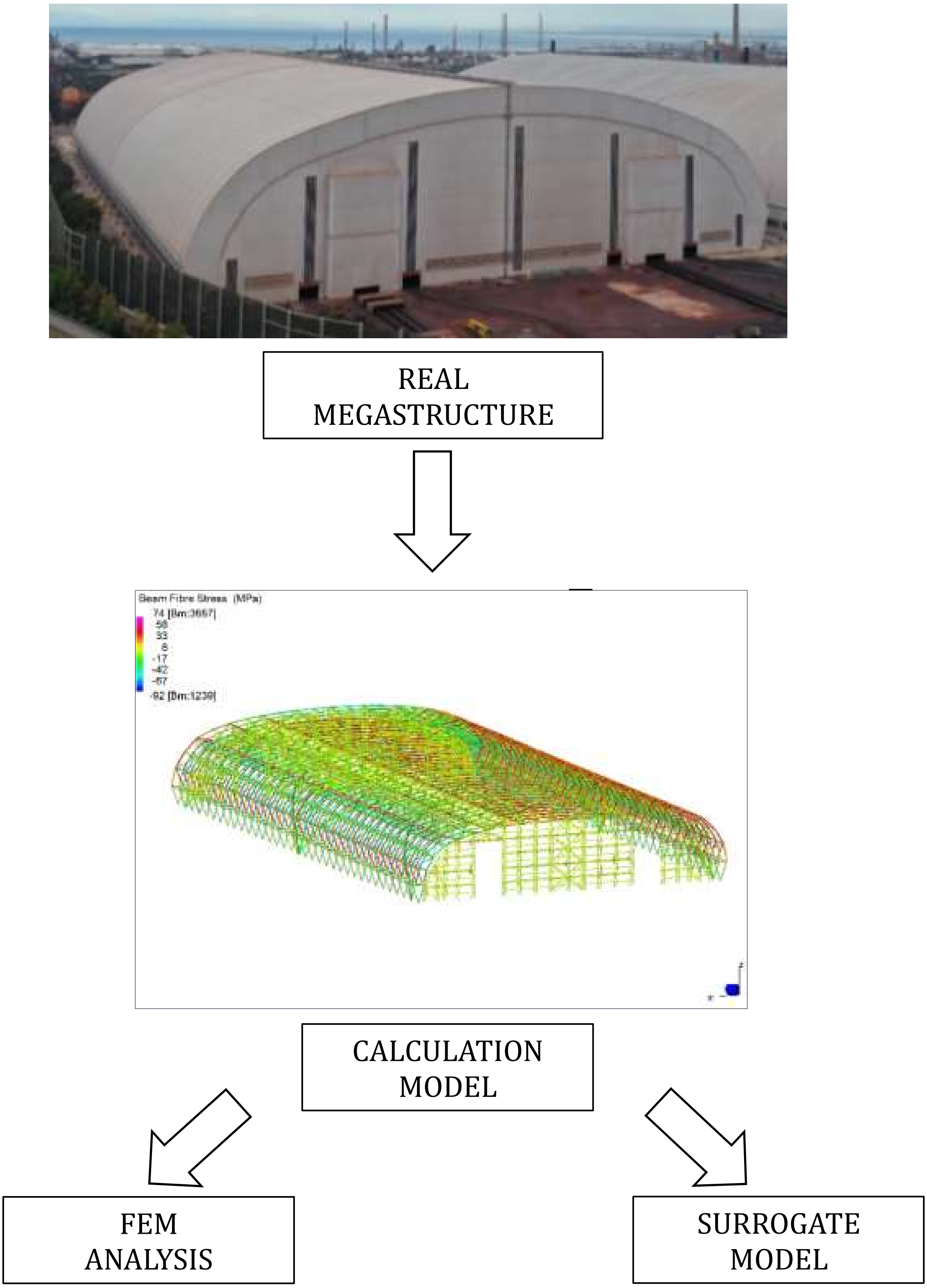


Figure 17. The actual structure is translated into a model suitable for evaluating its structural behavior. The calculation model can be used for FEM analysis or to derive surrogate models.

3.1 Design space approximation

The surrogate model technique is based on reducing the space of the starting problem to obtain a simplified model of the real problem, which is at the same time sufficiently accurate in imitating the phenomenon and allows a reduction in computational load. Thus, the creation of the surrogate model requires that an appropriate method be chosen to simplify the design space. There are numerous techniques in this regard, which follow very different philosophies. An initial classification can be made between techniques that take the physical phenomenon into account and those that do not. For example, the first category includes the hierarchical creation of FEM computational models, whereby a rough model is made in the early stages of design, and then gradually more complex and detailed models are made as the design process progresses. An example is shown in Figure 18, where it can be seen that for a framed building study there are various levels of detailing of the FEM model, going from a model of a simple cantilever beam to a detailed model of a node in the structure. Also belonging to the category of models related to physical phenomena are techniques by which the design space is reduced by decreasing the number of variables by establishing dependencies among them. These include adaptive finite element techniques, in which the mesh density is adjusted to achieve a good compromise between accuracy and computational cost. These methods related to the physical problem are attractive because they are linked to physical principles, but they have numerous drawbacks. First, they require extensive knowledge of the nature of the problem and the underlying physical phenomenon. In addition, they are difficult to automate and generalize for use over a wide range of problems. (3)

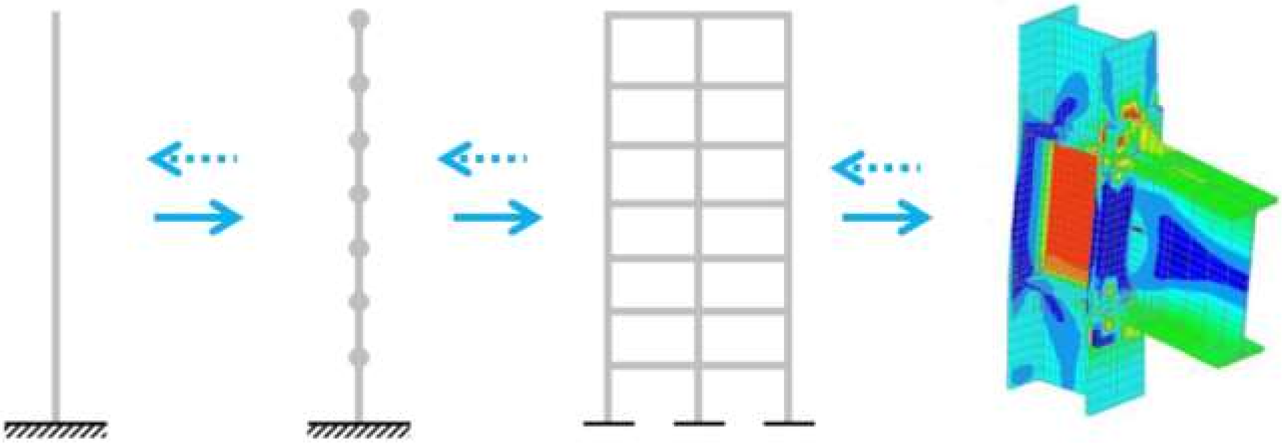


Figure 18. Different levels of detail of the FEM model of a framed building (3)

In opposition to techniques based on the physical phenomenon are data-driven techniques, in which the construction of the model is entirely based on data, regardless of the underlying physical phenomenon. This allows these techniques better ability to generalize and be implemented in a systematic way. In addition, generally data-driven methods have a prediction speed that does not depend on their accuracy. In this thesis, the focus is on these types of surrogate models. (3)

3.2 Data-driven surrogate models: an overview

Surrogate models, also called *metamodels*, are approximate models that mimic the behavior of a starting model, compared to which they have reduced computational cost, thus enabling assessments and judgments to be made more quickly. The construction of a surrogate model can also be seen as an optimization problem, with the aim of defining the configuration of the metamodel that best approximates the behavior of the initial model with the support of a suitable error measure. Numerous methods have been developed for making surrogate models, such as Response Surface Methodology (RSM), Kriging (KRIG), Artificial Neural Network (ANN), Multivariate Adaptive Regression Splines (MARS). In all methods, the creation of metamodels can be divided into three phases:

1. Training, in which models are built based on a set of data, called *training set*
2. Validation, in which the models trained in the previous phase are used on a data set other than the training one, the *validation set*, and the error between the exact value and the output provided by the models is calculated
3. Testing, in which the model that provided the least error in the validation phase is used on a third set of data, the *test set*, to evaluate whether it has reached the desired level of performance

The first two phases are usually repeated several times by changing the model parameters to achieve at the configuration that allows to obtain the model with the least error. The construction of the surrogate model requires that a data set be available to be used in the three phases, or that it be generated with an adequate sampling procedure. The construction of metamodels also requires the choice of the error measure to be used to evaluate the performance of the models based on the output of the model itself and the expected value. Examples of these metrics are Mean Squared Error (MSE), Root Mean

Squared Error (RMSE), Maximum Absolute Error (MAE). Another important aspect concerns the robustness of the model obtained, that is, its ability to provide sufficiently good results with as input a set of data that the model has never encountered. Surrogate models, in fact, should be a good compromise between adaptation to the data used for training and generalization. (23)

Data-driven surrogate models already find applications in numerous research fields, such as chemical, engineering, material science. These include the work done by Chang and Cheng (18), whose research aimed to optimize the beam and column cross-sections of a multi-story frame building. In this case, the surrogate model was used to predict the displacements at various stories of the candidate solutions, so that the displacement difference between floors could be estimated and this information used in the calculation of the objective function.

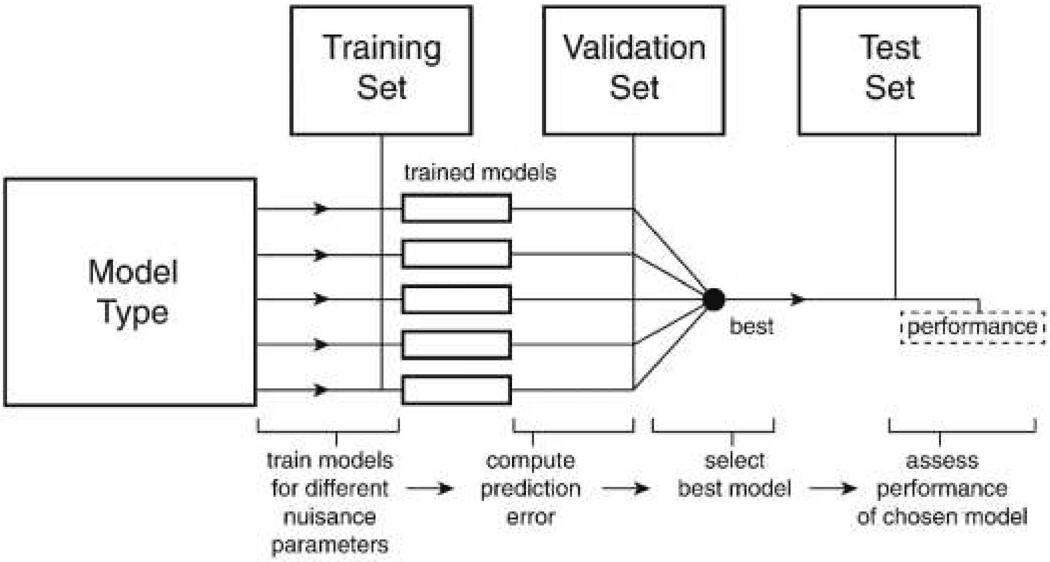


Figure 19. Surrogate modeling procedure. (23)

3.3 Neural Network (NN)

Neural Networks take their inspiration from the structure of the human brain, which consists of billions of interconnected neurons. Figure 20 shows a schematic of learning in the brain hypothesized by Rumelhart. (13) Each neuron receives inputs from the neurons connected to it through synapses, and if the sum of these impulses exceeds a certain threshold, activation of the neuron occurs, which will then also transmit the message to other neurons. This architecture is artificially implemented in Neural Networks in order to create a mathematical model that is able to learn based on data belonging to a dataset.

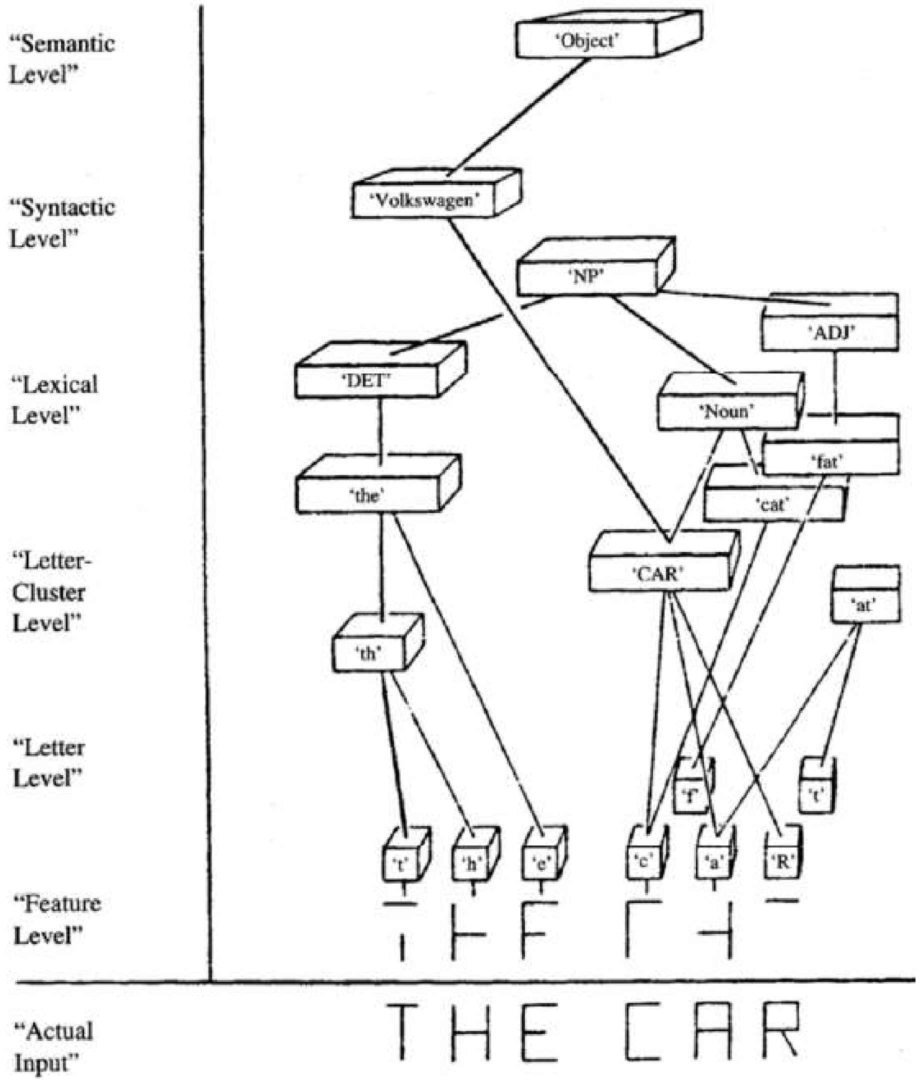


Figure 20. Diagram of input processing at the end of the message center (13)

For the generic unit i of the layer m, a linear combination is constructed with the outputs from the previous layer or, in the case of the first layer, with the input data of the problem. The coefficients used in the combination are called *weights*. Another term, representing the *bias*, can be added to this combination. The latter, along with the weights, are the parameters that must be learned during network training so as to minimize the error between the network's calculated outputs and the actual outputs. The result of the linear combination of the various unit inputs is transformed by a nonlinear function, called the *activation function*, thus obtaining the output of unit i of layer m. (24)

$$z_i^m = h(a_i) = h\left(\sum_{j=1}^{n_{units}^{m-1}} w_{ji}^{m-1} * z_j^{m-1} + w_{0i}^m\right) \tag{3.1}$$

where

z_i^m is the output of the unit i of the layer m

z_j^{m-1} is the output of the unit j of the previous layer m-1

w_{ji}^{m-1} is the weight relating to the connection between the unit j of the layer m-1 and the unit i of the layer m

n_{units}^{m-1} is the number of units in the layer m-1

w_{0i}^m is the bais of the unit i of the layer m

h is the activation function

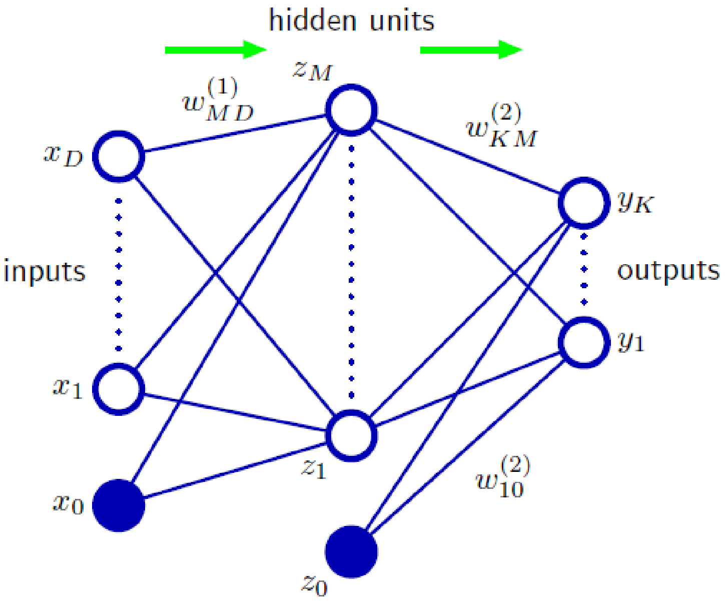


Figure 21. Neural Network structure, in which nodes represent the units of each layer while links represent the connections between units of successive layers, each of which is associated with a weight. The biases are represented by x_0 and z_0 (24)

In cases where nonlinear activation functions are used in the intermediate layers or if the network architecture has a small number of units in the intermediate layers compared to the size of the input and output layers, the model cannot be replaced with an equivalent one without intermediate layers.

Since the flow of information is in a single direction, that is, from the input layer to the output layer via intermediate layers, it is called a *feed-forward* architecture.

Neural Networks are parametric models that require a training phase to compute parameter values to better describe the phenomenon of interest. The goal is to find the parameter configuration that minimizes the error E between model outputs and the exact ones. The number of parameters depends on the architecture of the network, which in turn is related to the specific problem. More complex phenomena require more parameters to describe it and, consequently, a higher computational cost.

During the training phase, the network weights are updated to find the configuration that minimizes the error between model-calculated output and actual output. Correct definition of the error measure is critical for proper training of the network. The technique used is *backward propagation*. This method involves calculating the gradient of the error function with respect to each weight using the chain rule, starting from the last level and working backward. Referring to Figure 22, the derivative of the error $\mathbb{E}$ with respect to weight w_{ij} can be calculated as follows (24):

$$\frac{\partial \mathbb{E}}{\partial w_{ji}} = \frac{\partial \mathbb{E}}{\partial a_j}\frac{\partial a_j}{\partial w_{ji}} \tag{3.2}$$

$$\delta_j = \frac{\partial \mathbb{E}}{\partial a_j} = \sum_k \frac{\partial \mathbb{E}}{\partial a_k}\frac{\partial a_k}{\partial a_j} \tag{3.3}$$

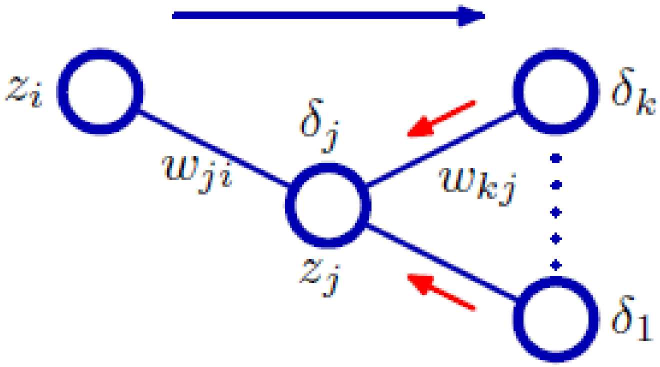


Figure 22. Backward propagation of error information (24)

Back propagation uses a multiplication of partial derivatives, which often have values less than unity. This can lead to vanishing gradient problems. One of the techniques to limit this issue is the *skip connection*, in which the output of one layer is not only used as input for the next layer but, by skipping connections, can also be used for layers further on. This provides alternative paths in the gradient calculation with beneficial effects in model convergence. Its use has also led to improvements in some surrogate models developed in this thesis and illustrated in CHAPTER 5: .

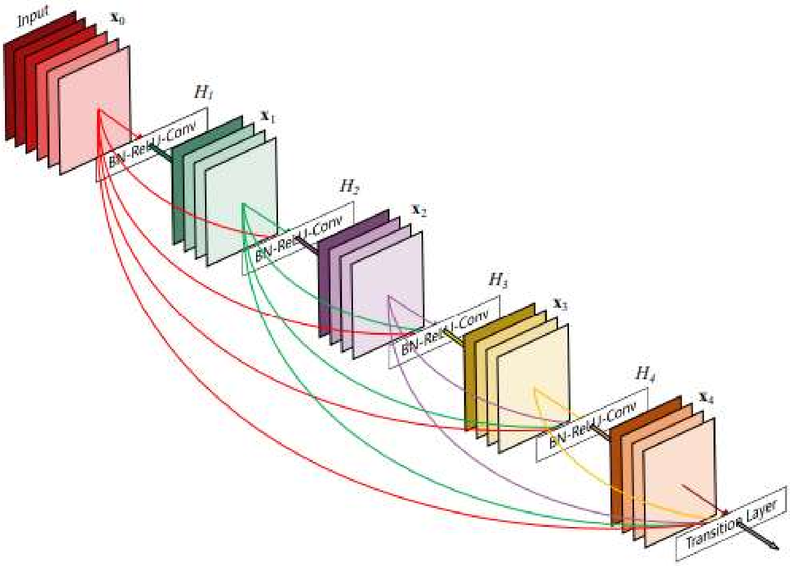


Figure 23. Skip connection example (25)

3.4 Graph Neural Network

Graphs are mathematical structures used to describe and study phenomena characterized by a set of objects in relation to each other. Recently they have attracted great interest because of their ability to describe numerous phenomena. A graph G consists of a set of nodes or vertices V connected to each other by link called edges (E). Each node can be associated with attributes x_v while each edge can be given a weight w_e.

$$G = \left(V, E_g, X_v, W_e\right) \tag{3.4}$$

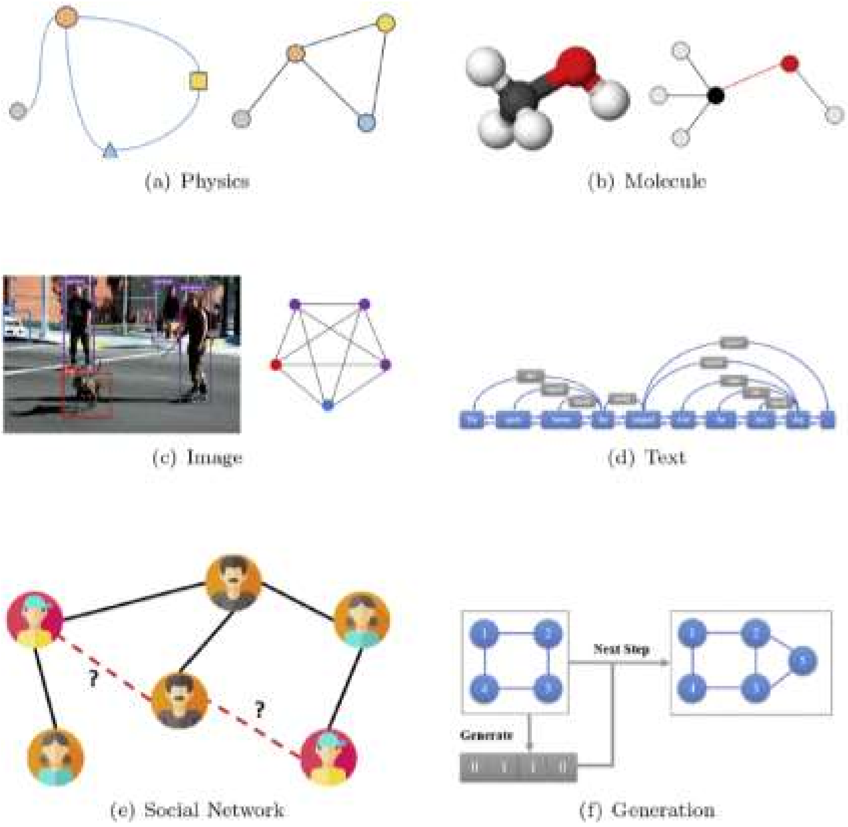
$V = \{v_1,\ v_2,\ \ldots, v_n\}$ set of nodes

$E_g = \{e_1,\ e_2,\ \ldots, e_m\}$ set of edges

$X_v = \{x_{v1,} x_{v2},\ \ldots,\ x_{vn}\}$ set of features vector $\forall v_i \in V$

$W_e = \{w_{e1,} w_{e2},\ \ldots,\ w_{em}\}$ set of features vector $\forall e_i \in E_g$

There are several classifications of graphs. If the relationships between the nodes in the graph are symmetrical then the graph is called *directed*, otherwise it is *undirected*. In addition, graphs can be classified into *homogeneous* or *heterogeneous* depending on whether or not the nodes are of the same type. If the input data or topology of the graph is a function of time, it is called a *dynamic graph*, otherwise a *static graph*.



(a) Physics

(b) Molecule

(c) Image

(d) Text

(e) Social Network

(f) Generation

Figure 24. Examples of application fields of graphs (26)

The representation of a problem by graph is used in a particular type of Neural Networks, called *Graph Neural Network (GNN)*. These models can be used in classification or regression problems at the node, arc or graph level. Like classical neural networks, GNNs have a layered configuration. An example of GNN's architecture is shown in Figure 25. The various layers that typically go into the structure of a GNN can be classified into 3 groups according to their function:

- *Propagation layers*, whose task is to transmit information between neighbors
- *Sampling layers,* which allow the message to be propagated through the graph while keeping the neighborhood size contained, and are therefore useful for large graphs
- *Pooling layers,* that offer a representation of a subgraph or the whole graph, which is particularly useful for node- or graph-level classification problems.

Propagation modules aim to transmit information within the graph. The operator used to perform this function can be of different types:

- *Convolution operators*, that perform convolutions (Figure 26)
- *Recurrent operators,* in which sequential data or data from time series are processed (Figure 27)
- *Skip connection operators*, which, as explained in the previous section, allow information to be gathered from historical representations of nodes

The structural problems addressed in this thesis are not dynamic in nature and therefore do not require sequential treatment of the data. Consequently, the following sections will focus on convolution operators.

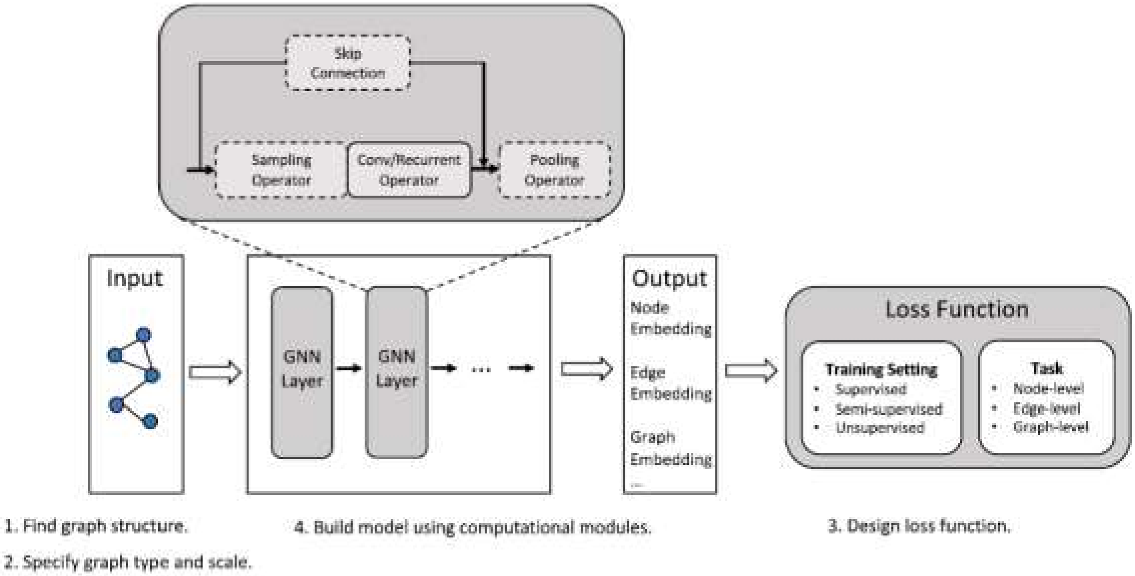


Figure 25. Architecture of a Graph Neural Network (26)

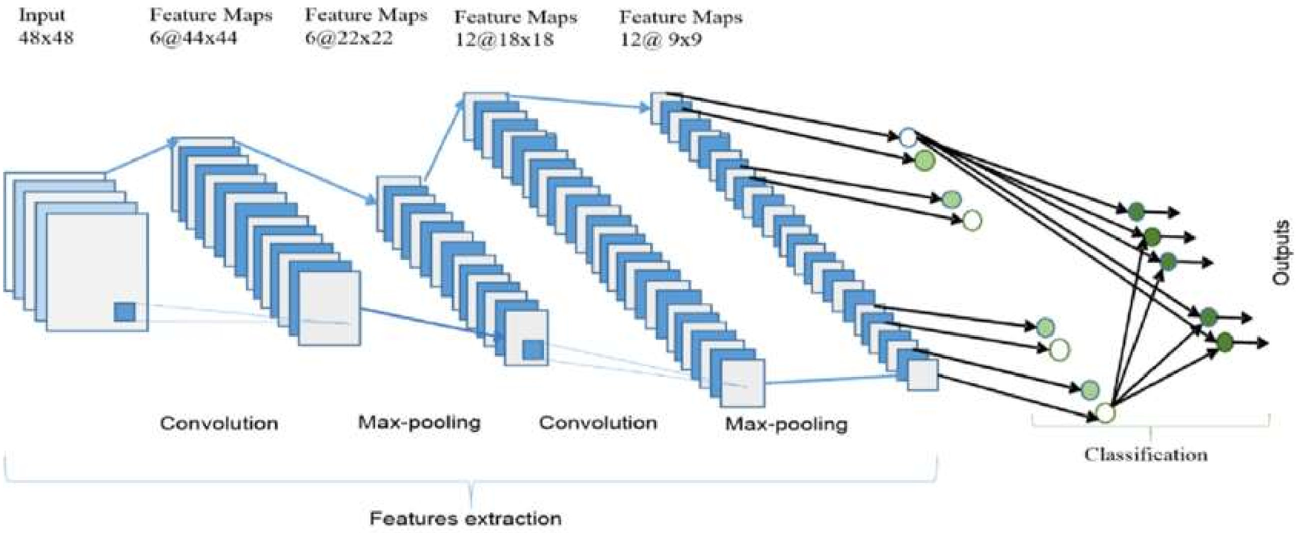


Figure 26. Example of Convolution Neural Network architecture (27)

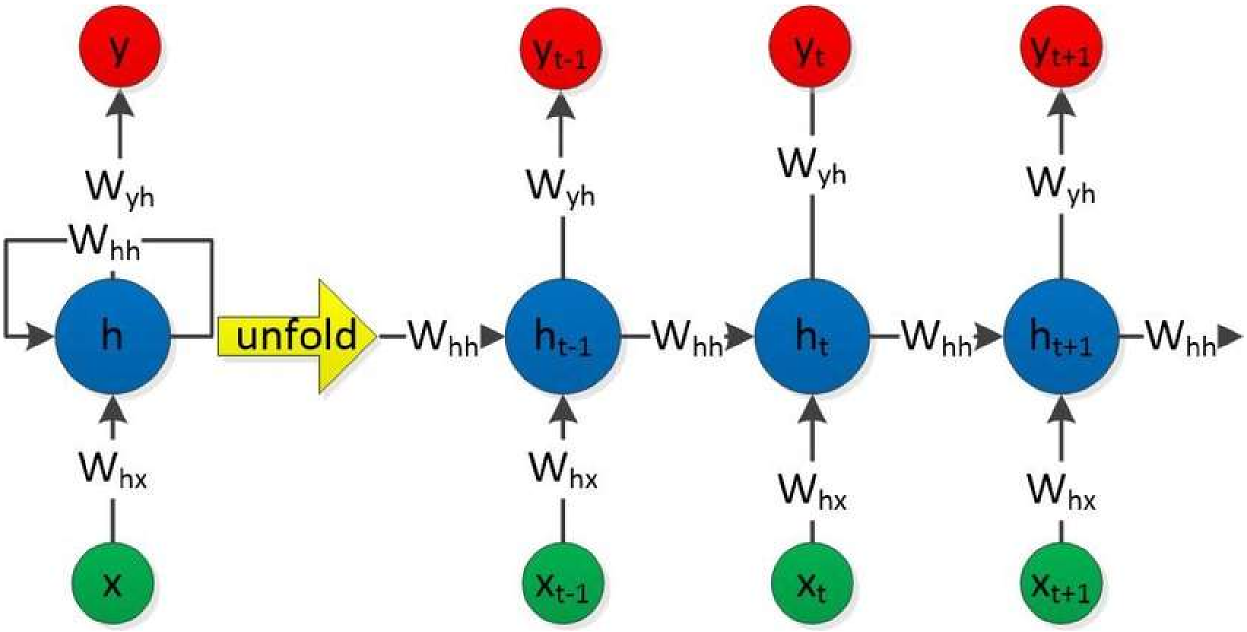


Figure 27. Example of Recurrent Neural Network architecture (28)

3.4.1 Convolution operators

Convolution operators allow convolutions to be performed on graphs. A representation of the architecture of a convolution-based GNN is in Figure 26. These operators can be divided into two categories, depending on whether a spectral or spatial approach is used.

In the case of spectral approaches, before convolution operations are performed, the graph signal x is transformed to the spectral domain by graph Fourier transform $\mathcal{F}$. What results from the convolution is transformed to the source domain through the inverse of the graph Fourier transform ($\mathcal{F}^{-1}$). (26)

$$\mathcal{F}(x) = U^T x \tag{3.5}$$

$$\mathcal{F}^{-1}(x) = Ux \tag{3.6}$$

where U is the matrix of eigenvectors of the normalized graph Laplacian.

In the spatial approach, however, operators perform the convolutional action based on the topology of the graph. They are constructed so that they can act regardless of the size of the neighborhood and maintaining the property of local invariance. Among the different types of Neural Networks with space-based convolutions are the Message-

Passing Neural Network (MPNN), in which information is passed from one node to another directly along the edges. It consists of two phases: (26)

a) *Message-passing phase*, in which the message of node *v* ($m_v{}^t$) is aggregated from the set of neighborhood nodes N_v through the message-passing function M_t and then the hidden state of node *v* ($h_v{}^t$) is updated with the update function U_t

$$m_v^{t+1} = \sum_{u \epsilon N_v} M_t(h_v^t, h_u^t, e_{vu}) \tag{3.7}$$

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1}) \tag{3.8}$$

where e_{vu} are the features of undirect edge between node *v* and node *u*.

b) *Readout phase* in which we obtain a representation of the entire graph based on the new hidden state of the nodes

$$\hat{y} = \tilde{R}(\{h_v^T | v \in G\}) \tag{3.9}$$

where *G* indicates the graph, $\tilde{R}$ is the readout function and *T* is the total number of steps in which the message has been propagated.

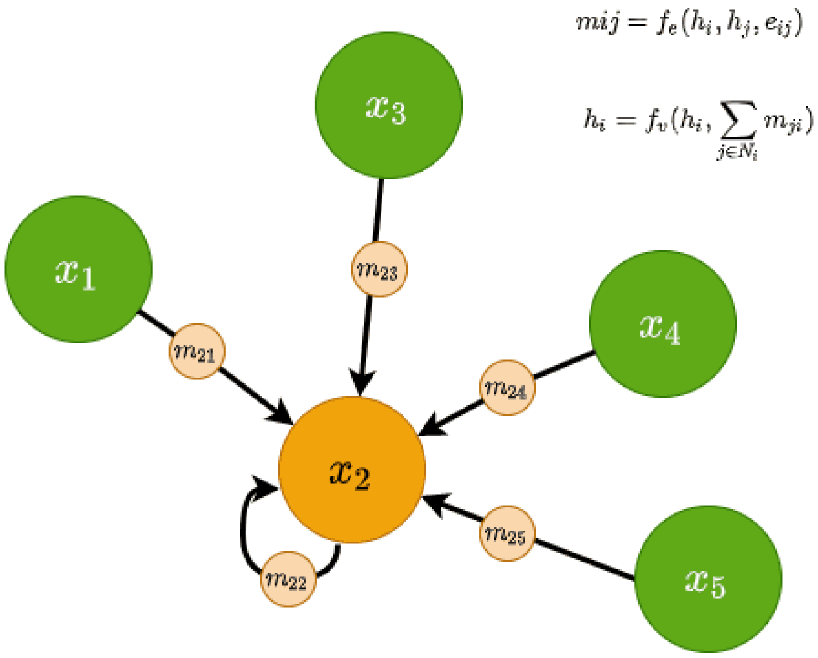

$mij = f_e(h_i, h_j, e_{ij})$
$h_i = f_v(h_i, \sum_{j \in N_i} m_{ji})$
x_3
x_1
m_{23}
x_4
m_{21}
m_{24}
x_2
m_{25}
m_{22}
x_5

Figure 28. Message-Passing Neural Network (MPNN) (29)

3.5 Conclusion

The increased ease of generalization and automation has made data-driven surrogate models more attractive than those based on the physicality of the problem for use in a context such as the structural context, where the phenomena driving the problem are very complex and where the candidate solutions to be evaluated are potentially numerous and varied. The next chapter investigated the possibility of creating a surrogate model that, having loads as input, provides the stresses on the beams, based on MLP and Graph Neural Networks. In addition to these approaches, another was tested based on creating a low-rank stiffness matrix to relate loads and stresses on the model beams.

CHAPTER 4: Optimization of a steel multi-way node

This chapter aims to propose optimization algorithms suitable for the structural problem. It starts with a description of the problem, in which the variables involved and the objective function guiding the search for the optimum are highlighted. The tools needed to compute the objective function for candidate solutions and how these enter the workflow with the optimization algorithm are shown. Three different optimization algorithms are proposed: a genetic algorithm, one based on gradients and one that seeks to exploit similarities between different elements in the search for the optimum. The first two of these algorithms are applied to a case study, consisting of a portion of a frame in which several beams converge to a node. The chapter ends with a comparison of the results obtained from the two different approaches.

4.1 Problem statement

The goal is to define an algorithm that provides an optimized design solution for a steel frame model. This is a minimization problem in which the objective function is the total cost of the structure, obtained as the sum of the costs of material and processing for beam and joints, and subject to the constraint of compliance with regulatory standards.

Structural verifications and regulatory standards dictated by the Eurocodes are considered. They represent a constraining condition in the search for the optimal solution within the solution space. These checks were implemented within a specially created verification software. This tool may be expanded in the future by also considering requests from other normative sources.

Different algorithm solutions have been analyzed to find the one that is best suited to the structural optimization problem, providing better performance in terms of computation times and quality of the solution.

The optimization algorithm is embedded in software created with the python programming language. This software is linked with Straus7 Finite Element software and Rhinoceros 3D CAD design software. In addition to the hyperparameters required by the optimization algorithm, and which depend on the chosen method, the software also asks for the FEM model of the frame as input. The optimization of the model is pursued by looking for the configuration of the variables of members and joints that minimize the objective function. These variables are described in the paragraph 4.2 .

4.2 Variables

The optimization problem involves variables relating to the cross section of the beams and variables that define the configuration of the joints located at the ends of the members. The size of the optimization problem is related to the number of beams in the portion of the structure to be optimized and the relative position they have between them. In fact, the number of beams affects the number of cross-sections to be optimized and the amount of joints included in the optimization problem. For each joint, the number of variables depends on the type of the joint itself, which is a function of the type of cross-section of the beams and the relative angles between the various beams. Only one type of joint was considered in this case study for simplicity.

Variables	
Beams	Section (Circular Hollow Section)
Joints	Bolt diameter Number of bolts Thickness of the plates (gusset, connection and covers) Welding throat Length of the welds

Table 1. Type of variables for each beam and each joint

Number of variables			
Entities	**Number of entities**	**Number of variables for one entity**	**Number of total variables**
Beams	9	1 (or 2)	9 (or 18)
Joints	14	5	70
	25	6 (or 7)	79 (or 88)

Table 2. Number of variables for each beam and each joint

4.2.1 Beam variables

Included in the set of variables of the optimization problem are the variables that define the geometry of the cross sections of the beams. They can be of two types:

- integer variables representing the section index within a database
- floating point variables which define the geometric parameters of the sections

In the first case, the number of variables corresponds to the number of beams, and the set of possible values each variable can take is finite and discrete. In the second case, however, the variables can be discrete or continuous depending on the approach chosen to solve the optimization problem. The number of variables depends on the type of cross section chosen for the beams. In fact, the type and amount of geometric parameters needed to define it depend on it. Moreover, for the same type of section, the choice of section parameters is not unique. In the case study presented in this chapter, Hollow Circular Sections (CHS) of class 1, 2 and 3 were considered. The structural checks from Eurocode implemented are therefore those for this category of profiles. The geometric parameters chosen to define these sections are two: the diameter (Diam$_{CHS}$) and thickness (Thk$_{CHS}$).

Figure 29. Beams with CHS section before installation

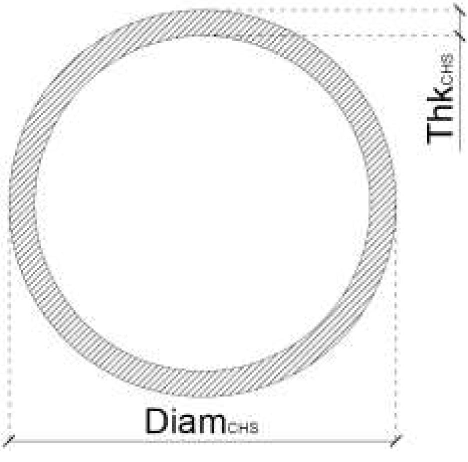


Figure 30. Section CHS

4.2.2 Joint variables

As mentioned in previous chapters, joints are generally composed of numerous elements, such as plates, bolts, and welds, whose geometric configuration can be subject to optimization. Therefore, the variables that define the joint will be the set of geometric parameters of the various elements that make up the joint. The type and amount of variables representing the joint will therefore depend on the type of the joint itself and the type of elements that compose it. In this case of study, it has been assumed that all joints are single plate connections, like the one in the Figure 33. It consists of a plate, called *gusset plate*, connected on one side to the beam by four welds and on the other side to another plate, called *connection plate*, by bolting. The bolted connection is made with two *cover plates*. The connection plate represents the connecting element of the joint to the actual node.

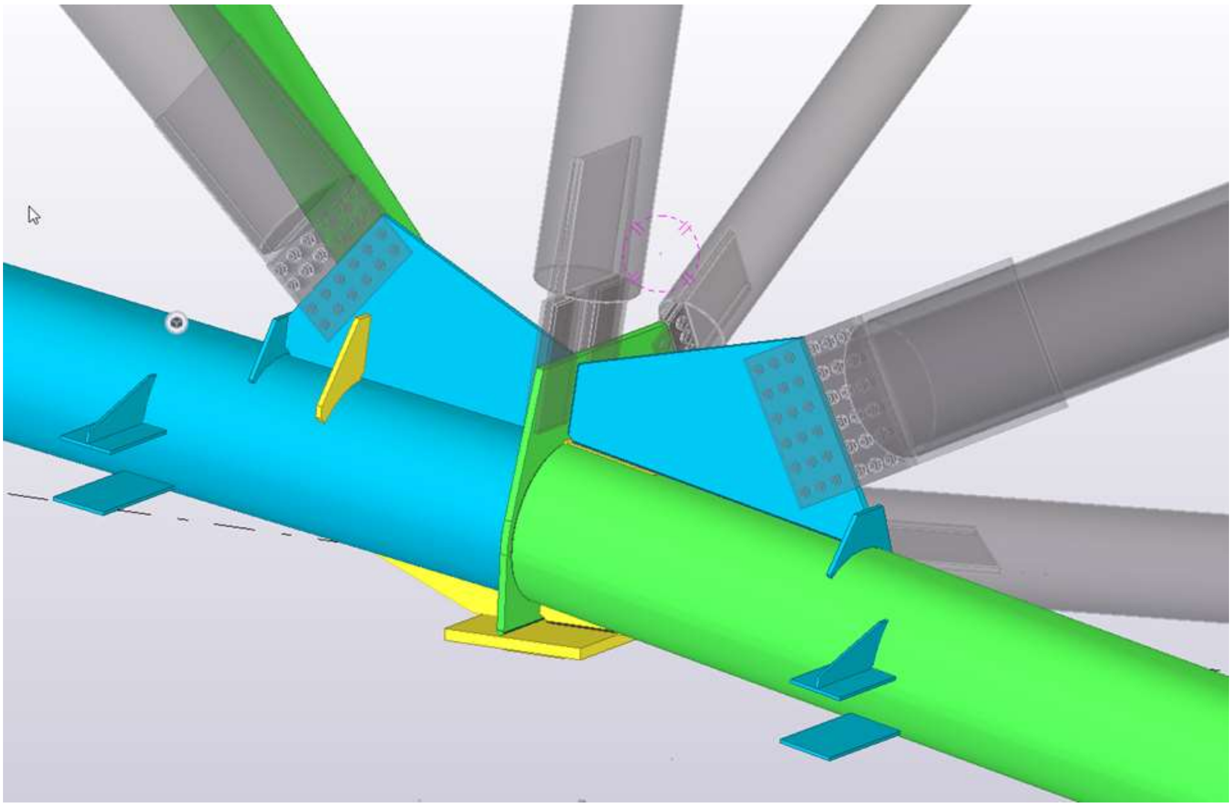


Figure 31. Example of a multi-way steel connection with single plate connection joints

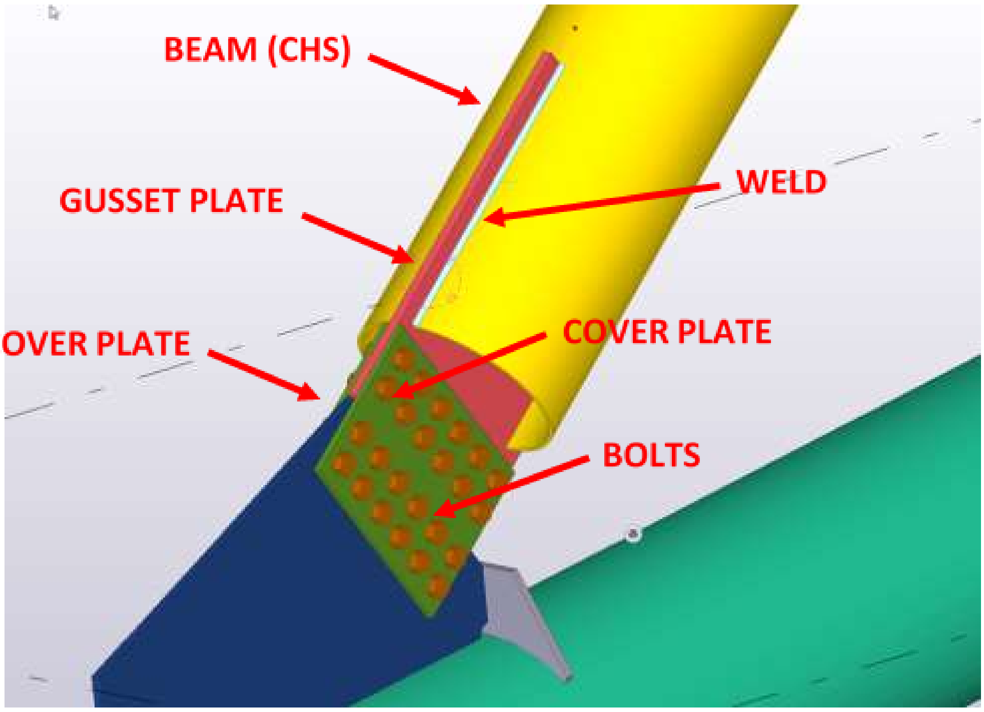


Figure 32. Plates that make up a single plate connection joint

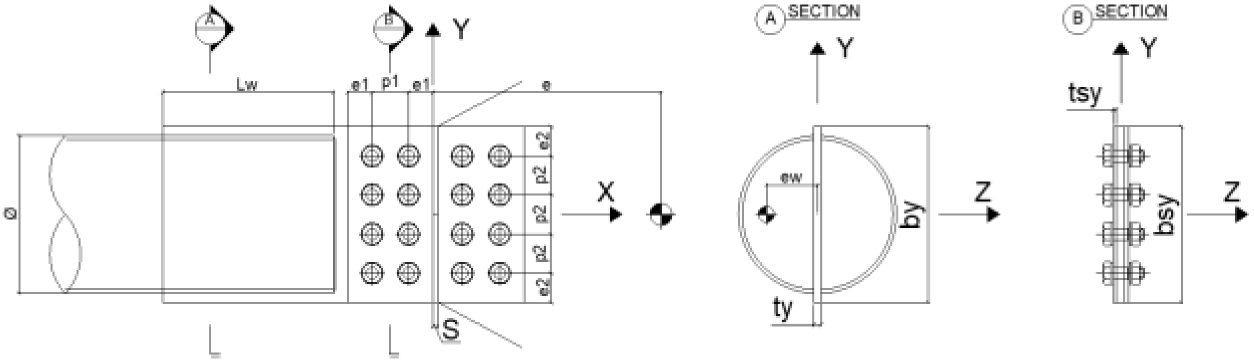


Figure 33. Single plate connection

The variables of the joints considered in the optimization problem are summarized in Table 3.

Variable	Type	Description
Thk	Float	Thickness of the gusset plate. The thickness of the cover plates is instead assumed to be equal to half of Thk.
D_b	Float	Bolt diameter
n_b	Integer	Number of bolts
a_w	Float	Throat of the weld
L_w	Float	Length of the weld

Table 3. Variables of the joints

The connection weld between beam and gusset plate is assumed to be a fillet weld.

The configuration of the bolts in the joint in terms of the number of rows and columns of bolts is researched using an algorithm specifically created to minimize the number of columns of bolts respecting the standards required by EN 1993-1-8 regarding the spacing between the bolts and the distance between bolts and free edge.

The thickness of the plates and the diameter of the bolts is constrained by the commercial availability of these elements. Therefore, these values can be treated either as an index in a database of available elements or as continuous variables and subsequently traced back to the closest element available commercially. However, for all variables a range can be defined within which the solution can be searched. In this way, the search space for solutions is reduced by discarding non-feasible solutions a priori.

4.3 Optimization framework

This section discusses the proposed workflow for optimizing a steel structure (Figure 34).

The input data is the starting FEM model containing the beams to be optimized and including all load conditions to be considered. This model will then be used in the optimization process to evaluate the structural behavior of the candidate solutions.

At the heart of the entire process is the optimization algorithm, which, by proposing the different candidate solutions, allows the solution search space to be explored. Each time the algorithm proposes a new solution, the fitness function must be calculated for it. The fitness function consists of a function that allows the estimation of the cost of the solution, to which factors are applied to consider whether or not the structural checks are satisfied.

For each load condition, resistance and instability checks are carried out for each beam and each joint according to the Eurocode. The results of these checks make it possible to assign a utilization rate to each element and contribute to the result of the objective function of the model by means of any penalty factors.

Carrying out the strength checks requires that the stresses on the beams at various stations located along the length of the element be known. These amounts are obtained using a FEM software, Straus7. The connection via API of the optimization software with Straus7 allows for each candidate solution to obtain the corresponding finite element model from which the stresses acting on the beams are derived for the different load conditions. The forces at the ends of the beams are also used for joint checks. The buckling check of the beams also requires that the effective length of the beams be defined, net of the encumbrance of the elements that make up the nodes and joints at the ends of the beam. This reduction in length of the beam is usually significant and, considering that often in compressed beams the dimensioning check is that of buckling, not taking it into consideration would be an excessively precautionary choice. In addition, the actual dimensions of beams and plates are necessary for estimating element costs. The evaluation of the length of beams and the size of joint elements is analytically complicated. For these reasons it was decided to use a 3D CAD software, Rhinoceros, connected via API to the optimization software, which through its geometric libraries allows you to perform these operations easily. It also allows you to create a 3D graphic model of the solution obtained by the optimizer allowing the user to better interact with the solution. The objective function values of candidate solutions are used by the optimization algorithm to propose new solutions. The process continues iteratively until convergence.

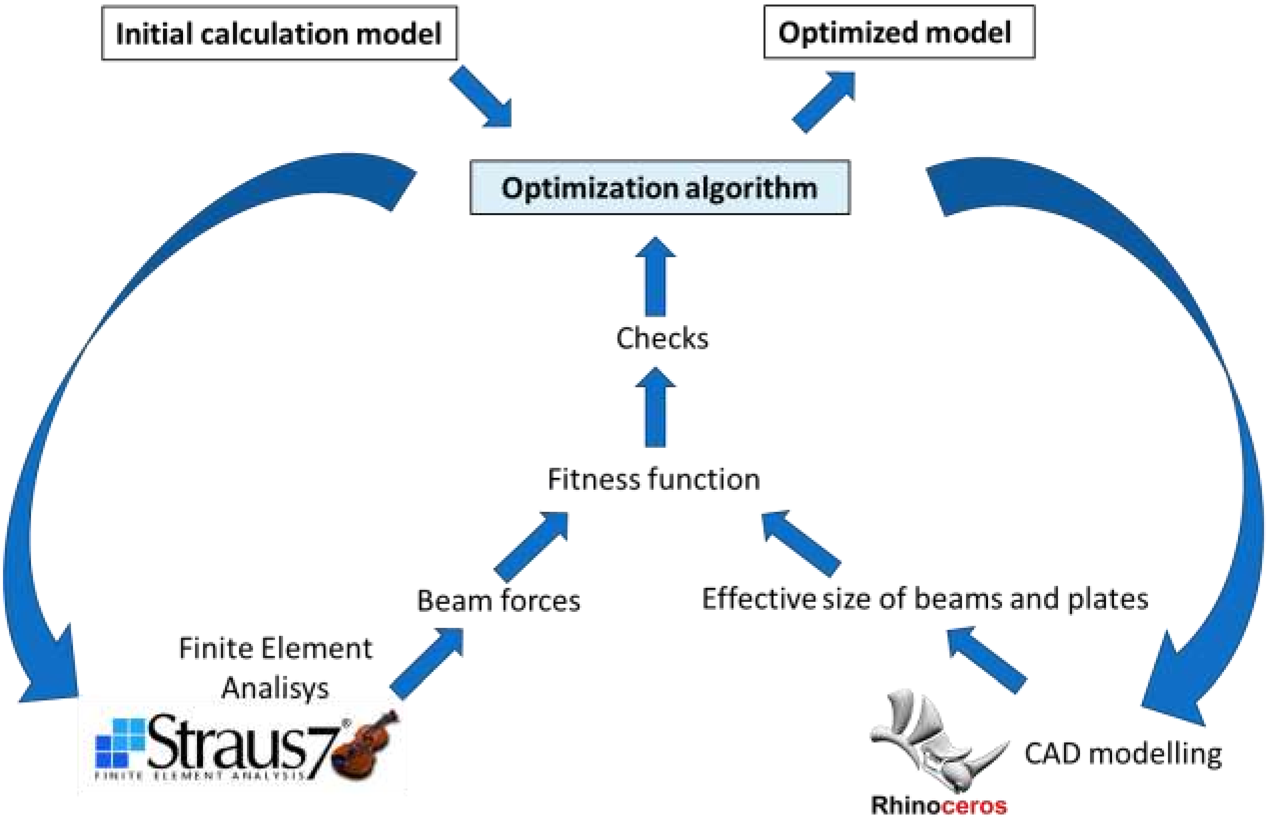
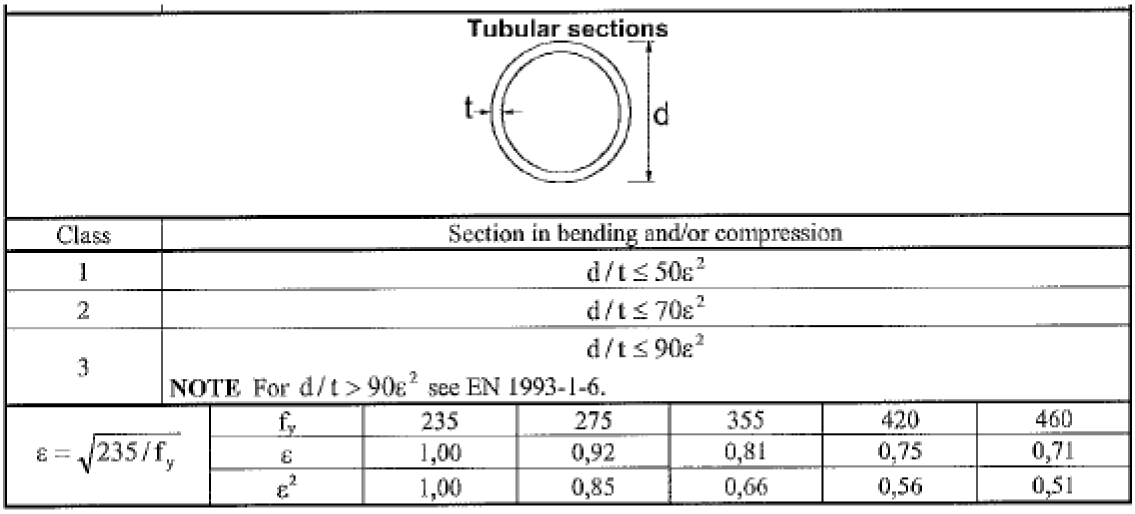


Figure 34. Optimization framework

4.4 Structural checks

4.4.1 Beam checks

The cross section is classified according to EN 1993-1-1-2005, who for the CHS sections is illustrated in Figure 35. This classification influences the checks to be carried out for the beam, which are summarized in Table 4, Table 5, Table 6 and Table 7.



Tubular sections

Class	Section in bending and/or compression
1	$d/t \leq 50\varepsilon^2$
2	$d/t \leq 70\varepsilon^2$
3	$d/t \leq 90\varepsilon^2$ NOTE For $d/t > 90\varepsilon^2$ see EN 1993-1-6.

$\varepsilon = \sqrt{235/f_y}$	f_y	235	275	355	420	460
	ε	1,00	0,92	0,81	0,75	0,71
	ε^2	1,00	0,85	0,66	0,56	0,51

Figure 35. Cross section classification for circular hollow sections

Strength checks for sections of class 1 or 2		
Check name	Equations	Forces involved
Tensile/compression	$\frac{N_{Ed}}{N_{pl,Rd}} \leq 1.0$ with $N_{pl,Rd} = \frac{A \cdot f_y}{\gamma_{M0}}$	Axial force
Bending	$\frac{M_{Ed}}{M_{pl,i,Rd}} \leq 1.0 \quad i=1,2$ with $M_{N,i,Rd} = M_{pl,I,Rd} * \rho_N * \rho_V$ $M_{pl,i,Rd} = \frac{W_{pl} \cdot f_y}{\gamma_{M0}} = \frac{2 * S * f_y}{\gamma_{M0}}$ $S = \text{static moment} = \frac{A}{2} * \frac{4}{3 * \pi} * \frac{R^3 - r^3}{R^2 - r^2}$ $J = \text{moment of inertia} = \frac{\pi}{64} * (D^4 - d^4)$ $\rho_N = \text{reduction of strength for tensile force}$ $= 1 - \left(\frac{N_{Ed}}{N_{pl,Rd}}\right)^{1.7}$ $\rho_V = \text{reduction of strength for shear force}$ $= \begin{cases} 1 \ \text{se} \ \frac{V_{Ed}}{V_{pl,Rd}} < 0.5 \\ (2 * \frac{V_{Ed}}{V_{pl,Rd}} - 1)^2 \ \text{se} \ \frac{V_{Ed}}{V_{pl,Rd}} \geq 0.5 \end{cases}$	Bending moment

Bi-axial bending	$\left[\frac{M_{1,Ed}}{M_{N,1,Rd}}\right]^{\alpha} + \left[\frac{M_{2,Ed}}{M_{N,2,Rd}}\right]^{\beta} \leq 1 \quad \text{with } \alpha = \beta = 2$	Bending moment around the axis 1 and 2
Plastic shear check	$\left\lvert\frac{V_{Ed,1}}{V_{pl,Rd,1}}\right\rvert + \left\lvert\frac{V_{Ed,2}}{V_{pl,Rd,2}}\right\rvert \leq 1.0$ with $V_{pl,Rd} = V_{Rd} * \rho_T$ $V_{pl,Rd} = \frac{A_v \cdot f_y}{\gamma_{M0} \cdot \sqrt{3}}$ $A_v =$ shear area $= \frac{2A}{\pi}$ $\rho_T =$ reduction of strength for torsional moment $= 1 - \frac{\tau_{t,Ed}}{(f_y/\sqrt{3})/\gamma_{M0}}$ $\tau_{t,Ed} =$ shear stresses for St. Venant torsion $= \frac{T_{Ed}}{2 * \Omega * t}$	Shear
Elastic shear check	$\frac{\tau_{Ed}}{f_y/(\sqrt{3} \cdot \gamma_{M0})} \leq 1.0$ with $\tau_{Ed} = \tau_{v,Ed} + \tau_{t,Ed}$ $\tau_{v,Ed}$shear stresses $= \frac{V_{Ed} * S}{J * t}$ $V_{Ed} = \sqrt{{V_{Ed,1}}^2 + {V_{Ed,2}}^2}$	Shear

Table 4. Strength checks for sections of class 1 or 2

Buckling checks for sections of class 1 or 2		
Check name	Equations	Forces involved
Compression	$\frac{N_{Ed}}{N_{b,Rd}} \leq 1$ con $N_{b,Rd} = \chi \cdot \frac{A \cdot f_y}{\gamma_{M1}}$ $\chi = \frac{1}{\Phi + \sqrt{\Phi^2 - \bar{\lambda}^2}}$ $\Phi = 0{,}5 \cdot \left[1 + \alpha \cdot \left(\bar{\lambda} - 0{,}2\right) + \bar{\lambda}^2\right]$ $\bar{\lambda} = \sqrt{\frac{A \cdot f_y}{N_{cr}}}$ $N_{cr} = \pi^2 \frac{E * J}{{L_0}^2} = \pi^2 \frac{E * J}{(L * \beta)^2}$	Compression axial force
Bending and axial compression	$\frac{N_{Ed}}{N_{b,y,Rd}} + k_{yy} \cdot \frac{M_{y,Ed}}{M_{b,y,Rd}} + k_{yz} \cdot \frac{M_{z,Ed}}{M_{b,z,Rd}} \leq 1$ $\frac{N_{Ed}}{N_{b,z,Rd}} + k_{zy} \cdot \frac{M_{y,Ed}}{M_{b,y,Rd}} + k_{zz} \cdot \frac{M_{z,Ed}}{M_{z,Rd}} \leq 1$ $M_{b,Rd} = \chi_{LT} \cdot \frac{W_{pl} \cdot f_y}{\gamma_{M1}} \quad$ con $\chi_{LT} = 1$	Compression axial force and bending moment

Table 5. Buckling checks for sections of class 1 or 2

Strength checks for sections of class 3		
Check name	Equations	Forces involved
Elastic strength check	$\frac{\sigma_{VM}}{f_y/\gamma_{m0}} \leq 1.0$ with $\sigma_{VM} = \sqrt{\sigma_{pc}{}^2 + 3 * \tau_{Ed}{}^2}$ $\tau_{Ed} = \tau_{t,Ed} + \tau_{v,Ed}$ σ_{PC} = stress for bending and axial force	Axial force, bending moment and shear force

Table 6. Strength checks for sections of class 3

Buckling checks for sections of class 3		
Check name	Equations	Forces involved
Compression	$\frac{N_{Ed}}{N_{b,Rd}} \leq 1$ con $N_{b,Rd} = \chi \cdot \frac{A \cdot f_y}{\gamma_{M1}}$ $\chi = \frac{1}{\Phi + \sqrt{\Phi^2 - \bar{\lambda}^2}}$ $\Phi = 0{,}5 \cdot \left[1 + \alpha \cdot \left(\bar{\lambda} - 0{,}2\right) + \bar{\lambda}^2\right]$ $\bar{\lambda} = \sqrt{\frac{A \cdot f_y}{N_{cr}}}$ $N_{cr} = \pi^2 \frac{E * J}{L_0{}^2} = \pi^2 \frac{E * J}{(L * \beta)^2}$	Compression axial force
Bending and axial compression	$\frac{N_{Ed}}{N_{b,y,Rd}} + k_{yy} \cdot \frac{M_{y,Ed}}{M_{b,y,Rd}} + k_{yz} \cdot \frac{M_{z,Ed}}{M_{b,z,Rd}} \leq 1$ $\frac{N_{Ed}}{N_{b,z,Rd}} + k_{zy} \cdot \frac{M_{y,Ed}}{M_{b,y,Rd}} + k_{zz} \cdot \frac{M_{z,Ed}}{M_{z,Rd}} \leq 1$ $M_{b,Rd} = \chi_{LT} \cdot \frac{W_{el} \cdot f_y}{\gamma_{M1}}$ with $\chi_{LT} = 1$	Compression axial force and bending moment

Table 7. Buckling checks for sections of class 3

Notations	
A	Section area
D	External diameter of the CHS beam section
d	Inner diameter of the CHS beam section
R	External radius
r	Inner radius
Ω	Area enclosed by the cross-sectional center line
t	Section thickness
N_{Ed}	Axial force
$V_{Ed,i}$ con i=1,2	Shear force along i axis
$M_{Ed,i}$ con i=1,2	Bending moment due to the shear force along i axis
T_{Ed}	Torsion force
f_y	Material strength
E	Young's modulus of steel
β	Buckling factor
L	Effective beam length

Table 8. Nomenclature used for the strength and buckling equations for the beams

4.4.2 Joint Checks

Joint verifications are performed using the stress parameters read at the corresponding beam end. The following checks are carried out for each joint:

- Bolt strength check, to assess whether the number and size of bolts offer sufficient resistance to the input forces
- Weld checks, which assesses whether the chosen throat is suitable for the stresses
- Strength checks of plates, to be carried out in both gross and net-of-hole sections
- Shear checks of plates to assess their shear behavior
- Plate buckling checks
- Bearing checks

4.5 Fitness function

The optimization process is focused on finding the solution that minimizes the objective function, also called the fitness function. Its correct definition is therefore crucial to obtaining the desired result from the optimization problem. In the present case, the function was constructed with the goal of obtaining a solution that would minimize the overall cost and, at the same time, satisfy structural checks. Therefore, it was defined as a weighted summation of the costs of each beam and each joint, where the weights are factors that depend on the maximum utilization rate of the element. For beams, only material-related costs were considered, which is a function of the cross section of the beam and its effective length. For joints, on the other hand, in addition to the material cost, cost items related to the following processing are also estimated:

- Cost of bolting, which depends on the number of bolts
- Cost of drilling, which is a function of the diameter and number of bolts as well as the thickness of the gusset plate and cover plates
- Cost of welding, which depends on the groove, the throat and length of the weld seams
- Cost of painting, function of plate area
- Cost of cuts, which depends on the size of the plates to be cut

The cost items for this processing were estimated in accordance with Díaz et al. (8).

Mathematically, the fitness function can be written as follows:

$$f = fitness\ function = \sum_{i=1}^{b} c_{1,i} * p_{1,i} + \sum_{k=1}^{j} c_{2,k} * p_{2,k} \tag{4.1}$$

where b and j are the number of beams and joints respectively, c_1 and c_2 are the costs of beams and joints while p_1 and p_2 are the penalty factors:

$$p_{1,i} = penalty\ factor\ of\ the\ i - th\ beam = \begin{cases} 1.0 & if\ \ u_{max} \leq 1.0 \\ 100 * u_{max} & if\ \ u_{max} > 1.0 \end{cases} \tag{4.2}$$

$$p_{2,k} = penalty\ factor\ of\ the\ k - th\ joint = \begin{cases} 1.0 & if\ \ u_{max} \leq 1.0 \\ 100 * u_{max} & if\ \ u_{max} > 1.0 \end{cases} \tag{4.3}$$

where u_{max} is the maximum utilization ratio for the entity.

4.6 Optimization algorithms

This section explains the optimization algorithms implemented to solve the structural optimization problem. These algorithms are based on different optimization approaches in order to identify the most appropriate one for the matter in hand.

The algorithms developed are as follows:

1. Genetic algorithm
2. Gradient-based algorithm
3. Grouping algorithm

4.6.1 Genetic Algorithm

The genetic algorithm represents one of the most popular meta-heuristic evolutionary methods used for solving optimization problems.

The algorithm starts by creating an initial random population consisting of N solutions. For each of them, a FEM calculation model is generated for calculating the forces acting on the beams and a CAD model is created for correctly estimating the dimensions of the joint plates and the length of the members. These information are used to carry out the structural checks and to evaluate the costs associated with the model in order to determine the value of the fitness function. If the optimization criterion is met, i.e., if the maximum limit of iterations is reached or if the minimum between the fitness function values of the analyzed models is less than the acceptability threshold, the best model is returned and the algorithm ends, otherwise a new model generation is created. The new generation, based on the previous population, has the following composition:

- Best n_{best} individuals from the preceding population
- n_r random individuals extract from the solutions of the previous population not belonging to the group of best
- The remaining (n_{tot}- n_{best} - n_r) individuals are created from the L best individuals of the previous population through crossover operation and mutation operation with probability p_{mut}

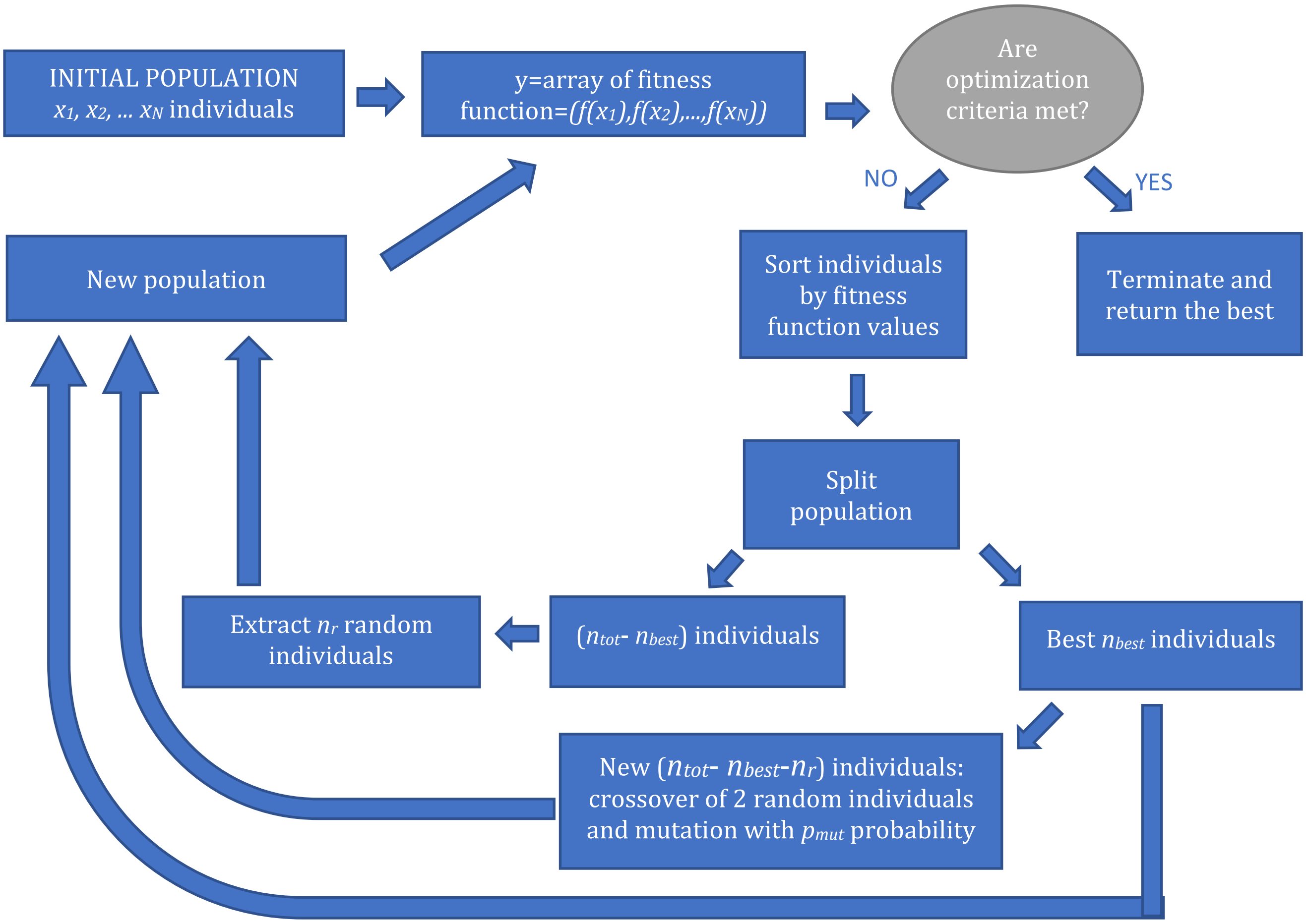

INITIAL POPULATION
$x_1, x_2, \dots x_N$ individuals
y=array of fitness function=$(f(x_1),f(x_2),\dots,f(x_N))$
Are optimization criteria met?
NO
YES
Sort individuals by fitness function values
Terminate and return the best
New population
Split population
Extract n_r random individuals
(n_{tot}- n_{best}) individuals
Best n_{best} individuals
New (n_{tot}- n_{best}-n_r) individuals: crossover of 2 random individuals and mutation with p_{mut} probability

Figure 36. Genetic algorithm

4.6.2 Gradient-based Algorithm

Gradient-based approaches involve using information from the gradient of the objective function when updating variables. These methods are suitable for problems with derivable objective functions and continuous variables. The presence of a black-box objective function and of both continuous and discrete variables with different scales requires that special measures are taken in order to be able to use these techniques for this case as well. These are as follows:

- fixed variable step different for each variable
- estimator approach
- use a variation and not a gradient

The algorithm was developed on the basis of that used in (30).

We start by assigning an initial set of random values to the *d* variables:

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix} \in \mathbb{R}^d \tag{4.4}$$

For each variable, we define a step size according to its type:

$$\delta = \begin{pmatrix} \delta_1 \\ \delta_2 \\ \dots \\ \delta_d \end{pmatrix} \in \mathbb{R}^d \tag{4.5}$$

The *d* variables are divided into *J* blocks of sizes $d^{(1)}, d^{(2)}, \dots, d^{(J)}$.

Assuming the existence of a permutation matrix $U \in \mathbb{R}^{dxd}$ and that this can be subdivided into *J* submatrices $[U^{(1)}, U^{(2)}, \dots, U^{(J)}]$, one for each block, it is possible to rewrite the *d* variables as follows:

$$x = \sum_{j=1}^{J} U^{(j)} x^{(j)} = \left(x^{(1)}, x^{(2)}, \dots, x^{(J)}\right) \tag{4.6}$$

$$U^{(j)} \in \mathbb{R}^{dxd^{(j)}} \qquad x^{(j)} \in \mathbb{R}^{d^{(j)}}$$

We introduce the sparse gradient hypothesis, which allows us to define an integer s_{exact} such that:

$$\|g(x)\|_0 \coloneqq |\{i: g_i(x) \neq 0\}| \leq s_{exact} \qquad 0 < s_{exact} < d \qquad \forall x \in \mathbb{R}^d \tag{4.7}$$

where $g(x) := \nabla f(x)$.

This assumption is exploited for the estimation of the gradient.

The number of variables belonging to each block is calculated by dividing the total number of variables by the number of blocks:

$$r = ceil(d/J) \tag{4.8}$$

For each iteration a random block is selected and n_{dir} directions of variation of the variables belonging to the selected block are investigated. The n_{dir} sampled directions are defined by a random matrix $Z \in \mathbb{R}^{n_{dir}\text{x}r}$ with -1 or +1 entries.

The degree of sparsity of the block is defined as follows:

$$s_{block} = o * s/J \tag{4.9}$$

where *s* is the gradient sparsity level while *o* is an oversampling parameter, assumed 1.1.

The number of sampled directions n_{dir} is calculated as a function of s_{block} and the size of each block *r*:

$$n_{dir} = o * s_{block} * r = o * s_{block} * \ln\frac{d}{J} \tag{4.10}$$

At the k-th iteration, the j-th block is randomly chosen, to which the variables $x^{(j)} \in \mathbb{R}^r$ belong and that have variable steps indicated in the vector $\delta^{(j)} \in \mathbb{R}^r$. The update of these variables along the n_{dir} directions is calculated as follows:

$$t_l = x_{k-1}^{(j)} + Z[l,:] * \delta^{(j)} \qquad 0 \leq l < n_{dir} \tag{4.11}$$

For the l-th direction the vector of variables $\bar{x}_l \in \mathbb{R}^d$ is constructed using for the variables belonging to block j the corresponding value contained in t_l while the others retain the value at iteration k-1.

The variation of the fitness function f(x) is calculated as follows:

$$h = \begin{pmatrix} h_0 \\ \cdots \\ h_{n_{dir}-1} \end{pmatrix} \epsilon \mathbb{R}^{n_{dir}} \qquad with\ h_l = \frac{f(\bar{x}_l) - f(x_{k-1})}{\sqrt{n_{dir}}} \tag{4.12}$$

The objective is to evaluate the influence of changes in variables on the fitness function by knowing the value taken by the fitness function in the n_{dir} directions explored and the changes in the variables that defined those directions. In mathematical terms this translates into solving the following problem:

$$g_k = argmin_{v \in R^r} \left\| \left(\frac{Z}{\sqrt{m}}\right) v - h \right\|_2 \qquad \text{s.t. } \|v\|_0 \leq s \tag{4.13}$$

Exploiting the gradient sparsity hypothesis, the problem is solved using CoSaMP, a signal reconstruction algorithm introduced in (31). The value of the variables is updated according to the vector g resulting from the CoSaMP algorithm:

$$x_k \epsilon \mathbb{R}^{dx1} \qquad x_k[i] = \begin{cases} x_{k-1}[i] & if\ x[i] \notin j - th\ block \\ x_{k-1}[i] + \delta[i] * \beta(g_k) & if\ x[i] \in j - th\ block \end{cases} \tag{4.14}$$

Each variable is limited in a range of allowable values.

$$x_{min}[i] \leq x_k[i] \leq x_{max}[i] \tag{4.15}$$

where x_{min} and x_{max} are the vectors containing the minimum and maximum values that the variables can take.

The function β should be chosen in a way that facilitates reaching the solution without incurring a local minimum and without the value of the variables being of such magnitude that convergence is not possible. The presence of variables of very different order of magnitude, combined with the strong nonlinearity of the objective function that occurs in the case of structurally unchecked elements, makes the evaluation of this function difficult. For this reason, it was set equal to the gradient sign function.

$$\beta(g_k) = -sign(g_k) \tag{4.16}$$

In this way, the information of the estimated gradient is exploited to evaluate the direction in which to increase the value of the variables. In the future, it would be interesting to be able to define a function for β that allows the value of the variables to be updated as a function of the magnitude of the gradient as well, so that convergence to the solution can be achieved more quickly.

4.6.3 Grouping Algorithm

The Grouping Algorithm aims to exploit the presence in the structure of beams and joints subjected to similar forces and which will therefore have a similar optimized geometry. The development of this algorithm was inspired by the Particle Swarm Algorithm (PSA), which attempts to reproduce the behavior of groups of animals in their search for the best place to rest, based on the availability of food, water, space, etc... It is based on the idea that each individual can represent the solution to the problem and that the measure of the goodness of his position depends both on what the particle itself has explored and what the rest of the group has discovered. Each individual has a memory of its previous states and this, together with the positions of the rest of the population and a stochastic factor, contributes in updating the position of the particle itself. (32)

Taking inspiration from PSA, in the Grouping Algorithm n_{tot} random models are generated, in each of which the beams and joints are divided into groups based on their stress state. At each iteration, joint and beam variables in each model are updated based on 3 contributions:

- value of the variable at the previous iteration
- the best beam/joint in its own group
- the best solution ever found among the various models

So, the part of variable updating that in the PSA is related to the best solution in the neighborhood, in the Grouping Algorithm is due to the best member of the beam/joint group membership.

Each beam has *d_b* number of variables while each joint has *d_j* number of variables. Denoting *b* as the number of beams and *j* as the number of joints, the total number of variables is as follows:

$$d = b * d_b + j * d_j \tag{4.17}$$

The variables that make up the h-th model are as follows:

$$x_h = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_d \end{pmatrix} \in \mathbb{R}^d \tag{4.18}$$

At k-th iteration the model h is updated by adding to the current values the product between the velocity vector *v* and the factor *K*:

$$x_h^{k+1} = x_h^k + v_h^{k+1} * K \tag{4.19}$$

The velocity vector $v_h^{k+1} \epsilon \mathbb{R}^d$ is obtained by adding two contributions to the current value of the velocity vector to consider the best solution found so far and the current value assumed by the variables of the best beam/joint in the group membership.

$$v_h^{k+1} = v_h^k + c_1 * r_p * (x_{h,best}^k - x_h^k) \; + c_2 * r_g * (x_{BEST} - x_h^k) \tag{4.20}$$

where c_1 and $c_2$are adjustable parameters with value 2.8 and 1.3, respectively, r_p and r_g are random value between 0 and 1, x_{BEST} is the best solution ever found among the various models and $x_{h,best}^k$ is the vector with the values of the variables taken by the best joint/beam for each group at iteration k.

The velocity weighting factor *K* is given by the following expression:

$$K = velocity\ weighting\ factor = \frac{2}{\left|2 - c_3 - \sqrt{{c_3}^2 - 4 * c_3}\right|} \tag{4.21}$$

with

$$c_3 = c_1 + c_2 \tag{4.22}$$

This algorithm is still under study and therefore has not been applied to the case study.

4.7 Case study

The case study examined for the development of the optimization tool is presented below. It is a steel cover consisting of 36 single-span truss arches made of tubular elements connected by bolts and welds. Each of these arches is 254 m wide and 77 m high. The Figure 37 shows an example of one of the arches making up the structure. The overall length of the structure is 476 m, as can be seen from the Figure 39. The structure is completed by facades at both ends of the cover, made up of a steel grid structure and connected to the main arches by expansion joints.

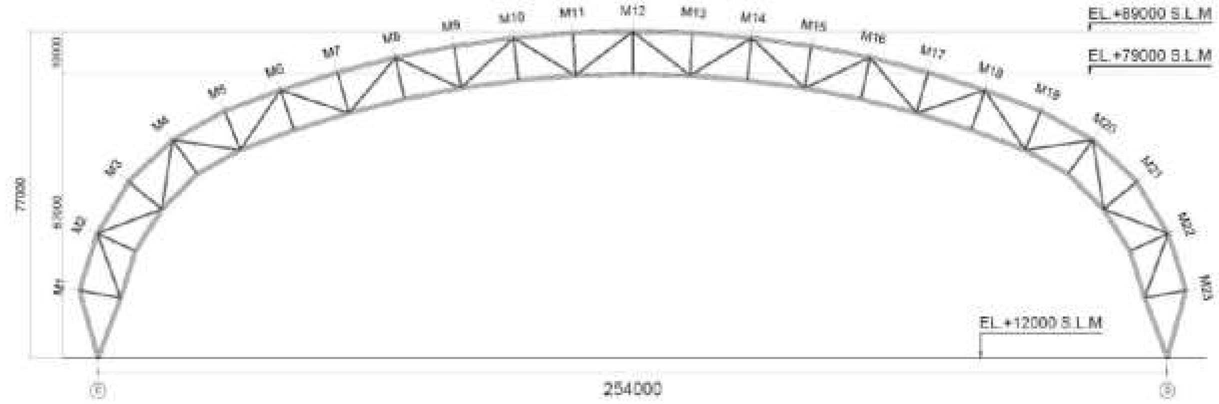


Figure 37. Example of a side view of a single arch

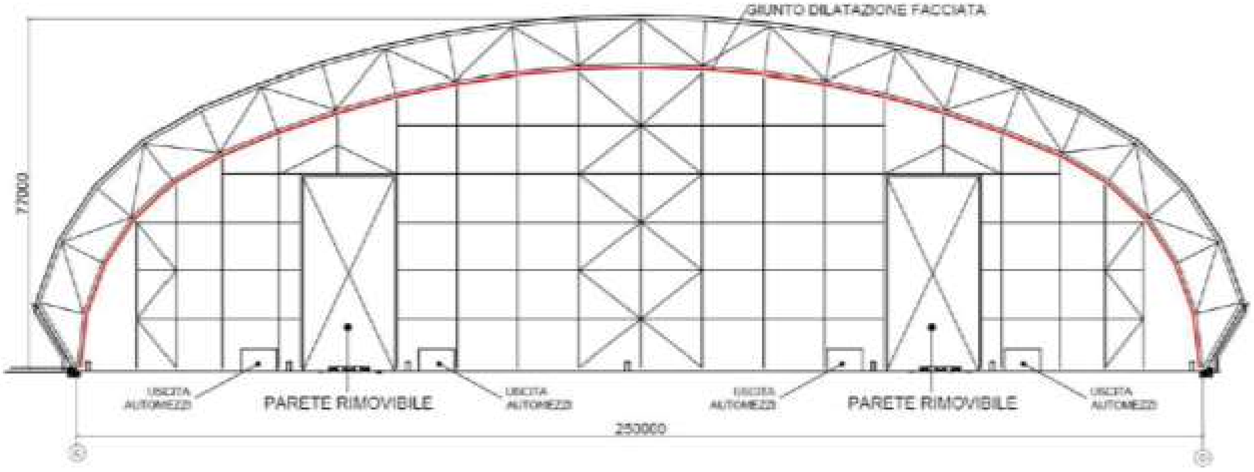


Figure 38. Side view of a facade

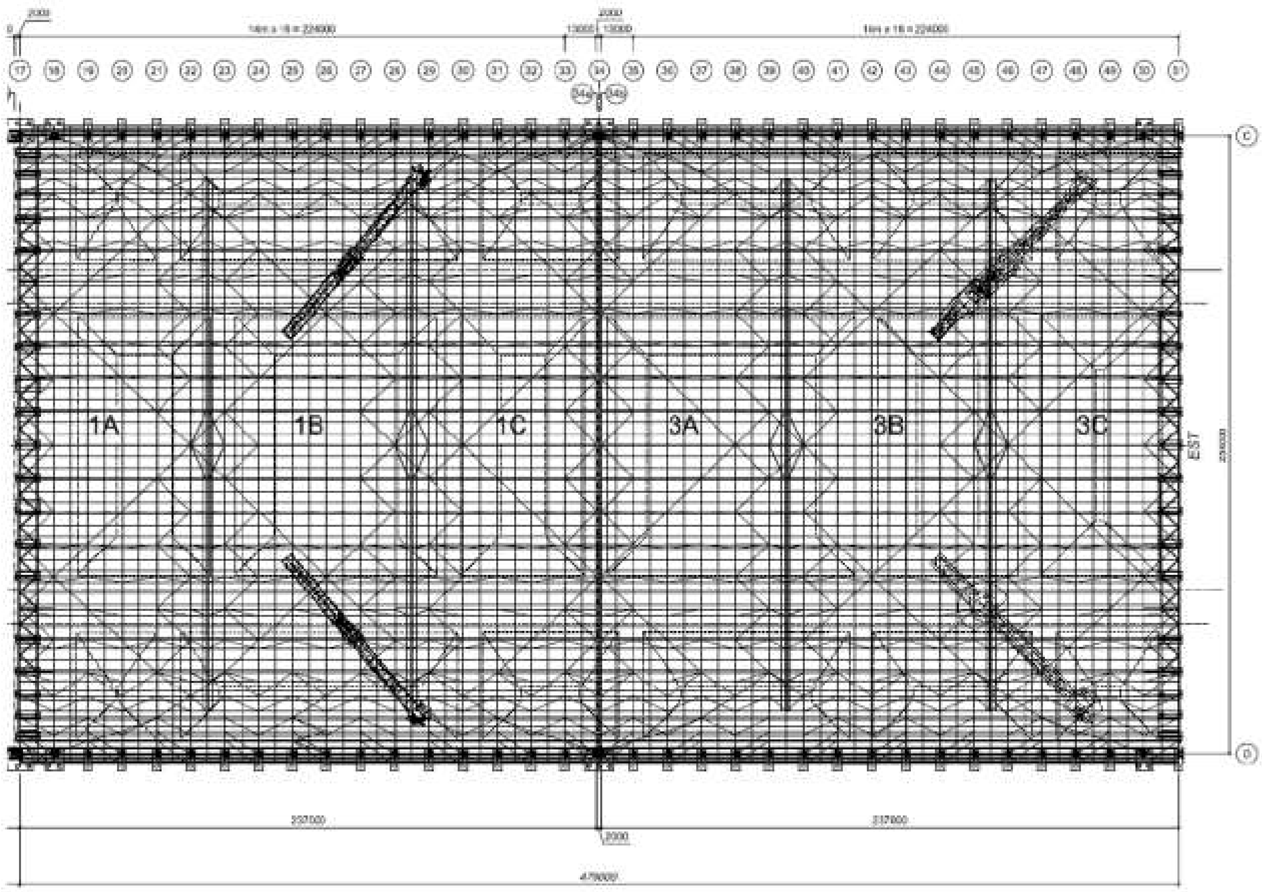


Figure 39. View of the cover structure

The connections between the main beams are:

- Joints with single plate connection, used for diagonals, bracing and upper longitudinal beams

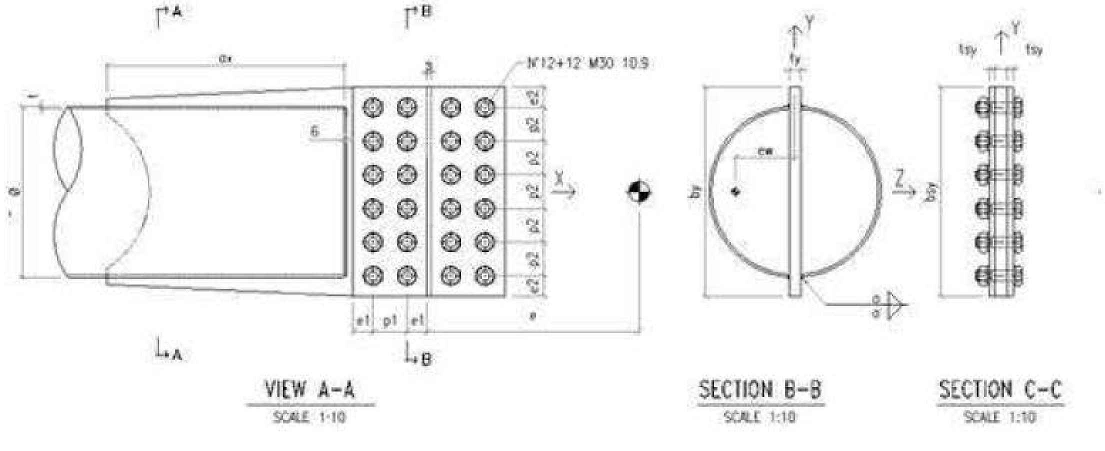


Figure 40. Example of joint with single plate connection

- Joints with single plate connection and split joint covers, used for struts, bracings and upper longitudinal beams

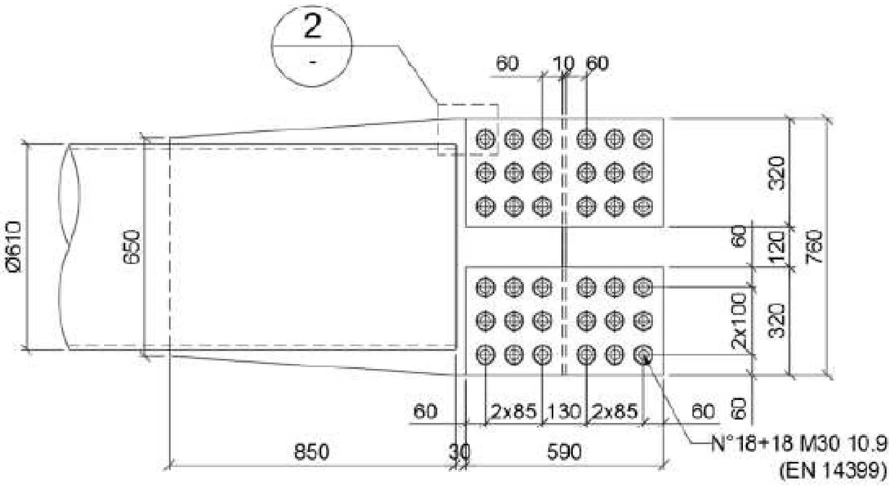


Figure 41. Example of joint with single plate connection and split joint covers

- Joints for cruciform plates, used for some diagonals of the arcs

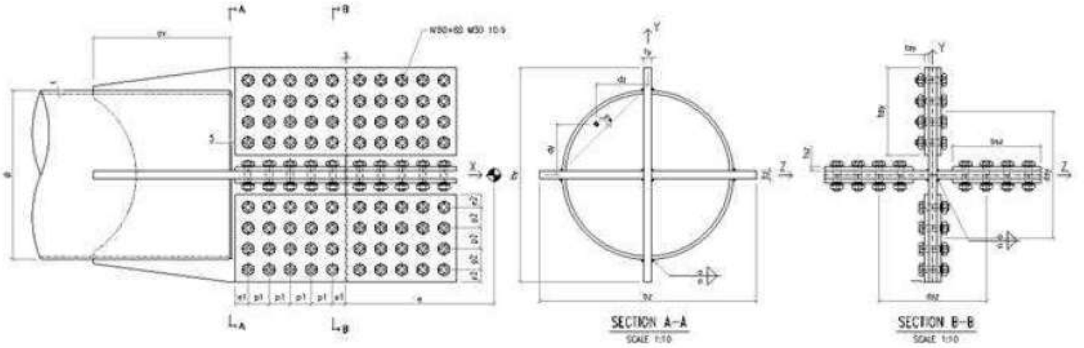


Figure 42. Example of joint for cruciform plate

- Bolted flange joints, used for upper and lower beams

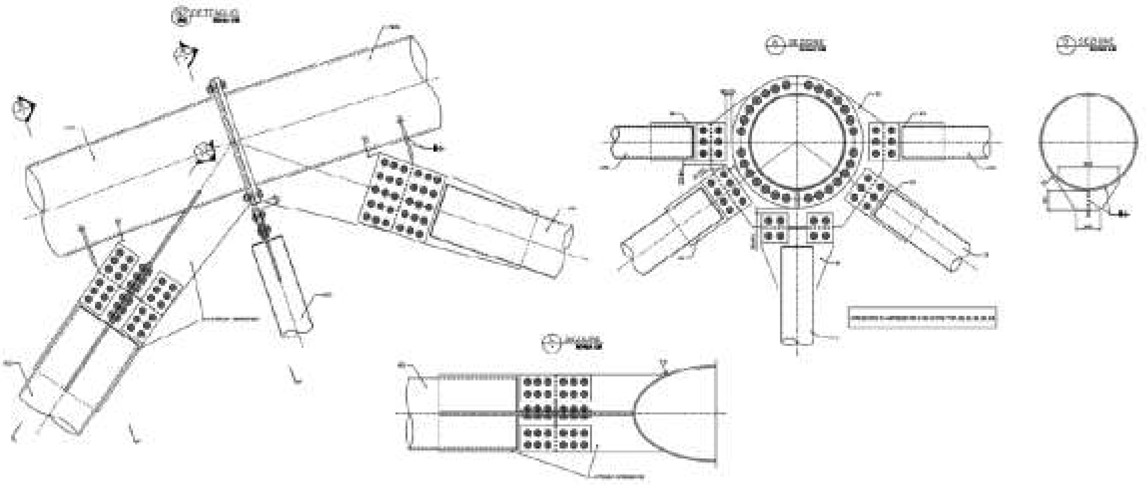


Figure 43. Example of bolted flange joint

- Welded flange joints, used for upper and lower beams

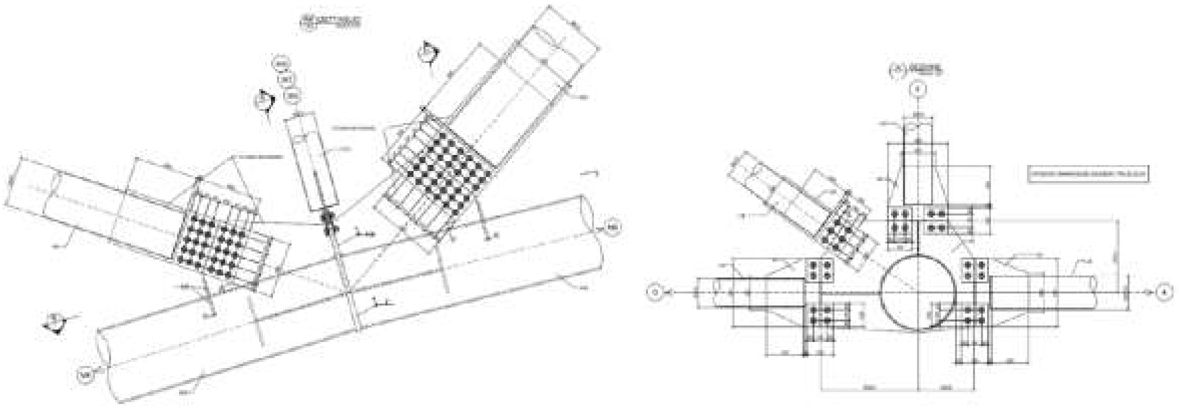


Figure 44. Example of welded flange joint

- Pin connection, used in expansion joints and for some diagonals on assembly joints

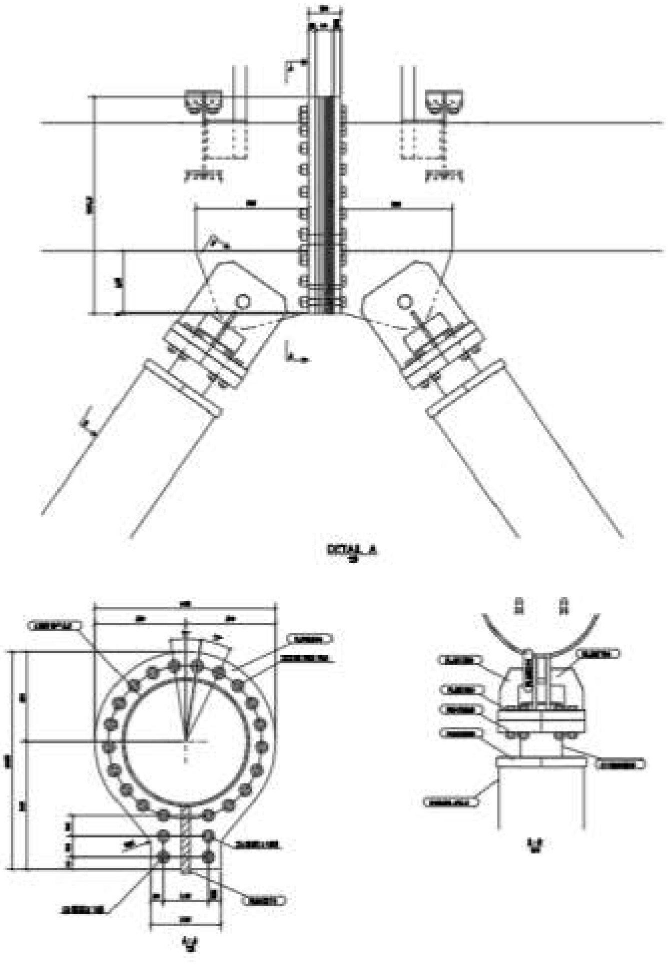


Figure 45. Example of pin connection

As can be seen from the images above, the connections between the main elements of the structure are characterised by a considerable amount of plates, bolts and welds and their realisation requires numerous processes. The number of such connections in the arches alone can be estimated at 1728, to which the connections on the facades and those forming expansion joints must also be added. These considerations show the crucial role that connections play in the cost of the whole structure and the importance of acting on these elements to achieve a correctly optimised design solution.

4.7.1 Calculation model

The evaluation of the structural performance of the structure requires the elaboration of a finite element calculation model. It is composed of beam elements representing the tubular elements that make up the cover, while the assignment of loads is carried out by introducing plate elements.

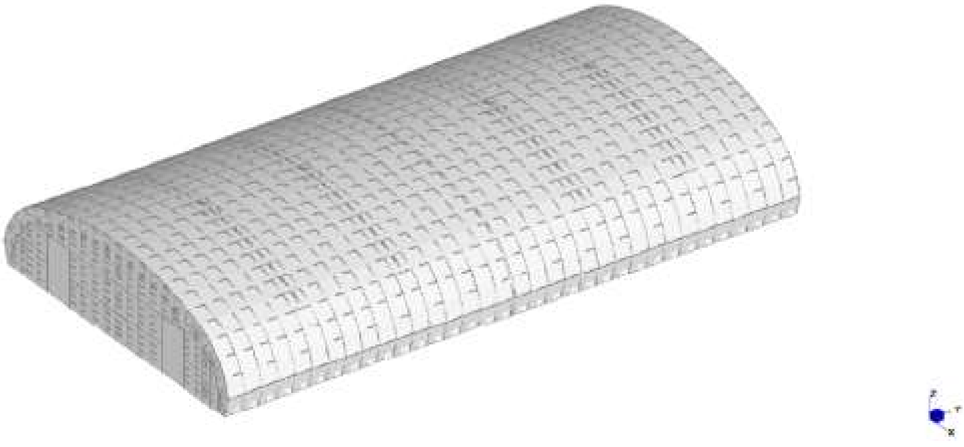


Figure 46. Calcutaion model shell

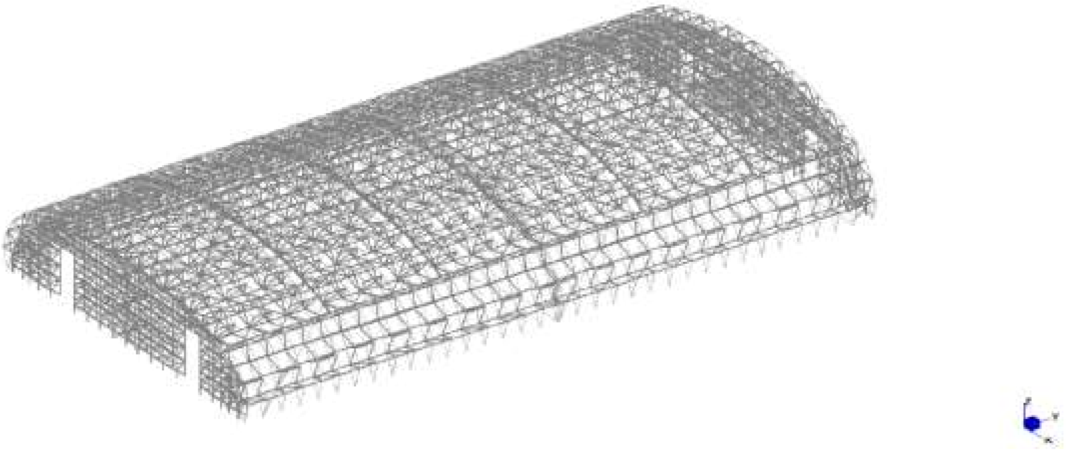


Figure 47. Calculation model beams

A total of 255 load cases of the following types were included in the calculation model:

- Self-weight
- Weights of non-structural elements and installations
- Snow load
- Wind load
- Thermal load
- Seismic load
- Foundation subsidence

These load cases are used to create combinations that represent the behavior of the structure during the operative phase, in the case of an earthquake and at the final condition in accordance with the regulations. A total of 3716 load combinations were carried out. The large number of load combinations, together with the large number of elements that make up the calculation model, mean that the analysis of the model requires a high computational cost.

4.7.2 Multi-way node optimization

The performance of the optimization approaches explained in the previous sections was evaluated by applying them to the optimization problem of the multi-way node in Figure 48. It consists of 9 beams and 14 joints.

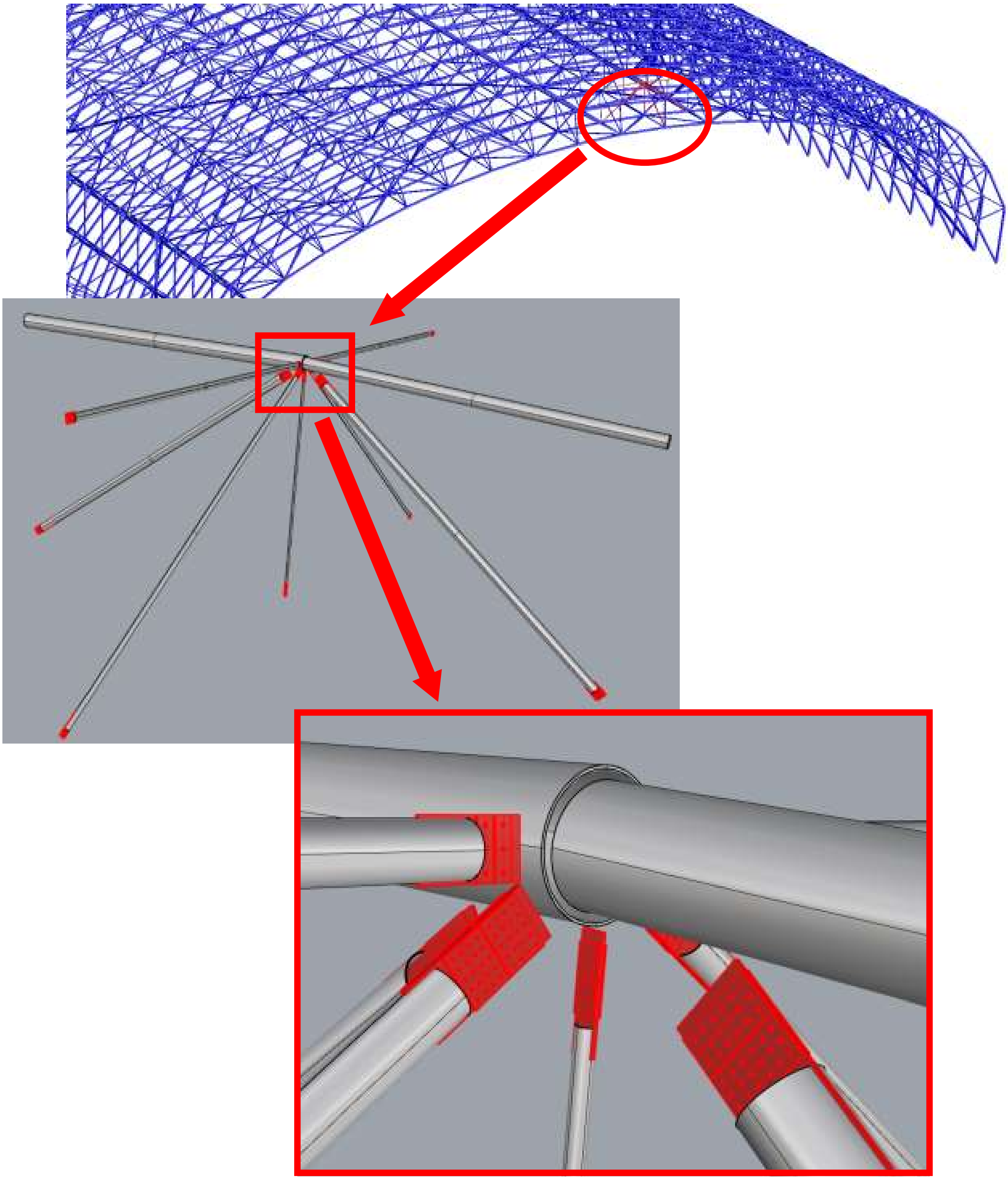


Figure 48. Multi-way node under study

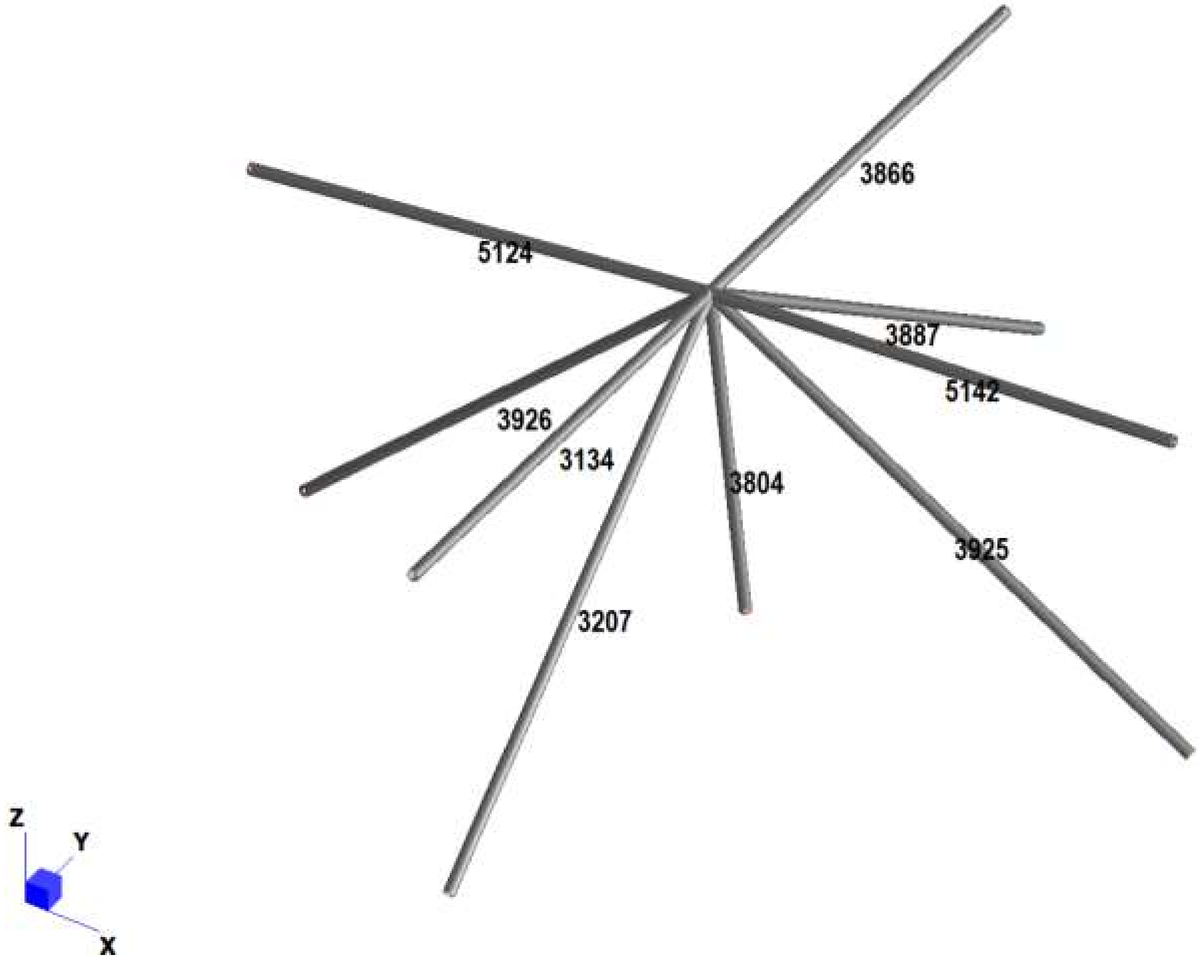


Figure 49. Beam numbering

The following tables show the initial characteristics of the beams and joints belonging to the multi-way node. The cost estimate is based on the indications given in (8).

BEAM	DIAM [mm]	THK [mm]	LENGTH [mm]
3134	355.6	6.3	11912.4
3207	406.4	8.0	14423.9
3804	273.0	5.0	8382.8
3866	323.9	5.6	12114.4
3887	323.9	5.6	15143.8
3925	610.0	11.0	13422.5
3926	508.0	8.0	13628.4
5124	813.0	16.0	14184.0
5142	813.0	16.0	14186.4

Table 9. Beam attributes

BEAM	Joint End1								Joint End2							
	NR	Diam bolt [mm]	Nr rows	Nr col	Thk gusset [mm]	Thk covers [mm]	Throat [mm]	Lw [mm]	NR	Diam bolt [mm]	Nr rows	Nr col	Thk gusset [mm]	Thk covers [mm]	Throat [mm]	Lw [mm]
3134	0	30	4	2	25	15	8	480	1	30	4	2	25	15	8	480
3207	2	30	5	2	25	15	5	550	3	30	5	2	25	15	5	550
3804	4	30	4	1	20	15	5	350	5	30	4	1	20	15	5	350
3866	6	30	4	2	25	15	8	480	7	30	4	2	25	15	8	480
3887	8	30	4	1	20	10	5	400	9	30	4	1	20	10	5	400
3925	10	30	6	3	35	18	8	800	11	30	6	3	35	18	8	800
3926	12	30	6	3	25	15	7	750	13	30	6	3	25	15	7	750
5124	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
5142	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 10. Joints attributes

NR	ID	SECTION AREA [mm2]	VOLUME [mm3]	WEIGTH [kg]	COST [euro]
3134	51510	6913.36	82354678.42	644.01	**257.61**
3207	51610	10012.88	144424839.10	1129.40	**451.76**
3804	52310	4209.73	35289359.48	275.96	**110.39**
3866	52510	5599.83	67838532.98	530.50	**212.20**
3887	52610	5599.83	84802646.09	663.16	**265.26**
3925	52411	20699.95	277845132.50	2172.75	**869.10**
3926	52410	12566.37	171259525.30	1339.25	**535.70**
5124	52210	40061.59	568233585.70	4443.59	**1777.43**
5142	52211	40061.59	568329733.60	4444.34	**1777.74**
TOTAL					***6257.18***

Table 11. Beams costs

NR	BEAM NR	STEEL GUSSET PLATE [euro]	STEEL COVER PLATES [euro]	BOLT [euro]	CUT GUSSET PLATE [euro]	CUT COVER PLATES [euro]	HOLE [euro]	PAINTING GUSSET PLATE [euro]	PAINTING COVER PLATES [euro]	WELD [euro]	COST [euro]
0	3134	23.48	16.55	162.96	2.93	3.92	6.58	2.15	2.53	1.07	**222.17**
1	3134	23.48	16.55	162.96	2.93	3.92	6.58	2.15	2.53	1.07	**222.17**
2	3207	31.06	20.49	203.70	3.21	4.24	7.26	2.84	3.13	0.51	**276.45**
3	3207	31.06	20.49	203.70	3.21	4.24	7.26	2.84	3.13	0.51	**276.45**
4	3804	14.89	10.32	81.48	2.36	4.24	5.09	1.70	1.58	0.34	**122.00**
5	3804	14.89	10.32	81.48	2.36	4.24	5.09	1.70	1.58	0.34	**122.00**
6	3866	21.81	15.37	162.96	2.14	3.82	6.58	2.00	2.35	1.07	**218.09**
7	3866	21.81	15.37	162.96	2.14	3.82	6.58	2.00	2.35	1.07	**218.09**
8	3887	13.42	6.10	81.48	2.33	2.99	4.84	1.54	1.40	0.38	**114.47**
9	3887	13.42	6.10	81.48	2.33	2.99	4.84	1.54	1.40	0.38	**114.47**
10	3925	82.15	44.51	366.66	5.05	5.60	11.79	5.37	5.66	1.76	**528.55**
11	3925	82.15	44.51	366.66	5.05	5.60	11.79	5.37	5.66	1.76	**528.55**
12	3926	46.22	23.65	366.66	3.85	4.50	10.00	4.23	3.61	1.29	**464.00**
13	3926	46.22	23.65	366.66	3.85	4.50	10.00	4.23	3.61	1.29	**464.00**
TOTAL		**466.06**	**274.01**	**2851.80**	**43.75**	**58.61**	**104.26**	**39.67**	**40.48**	**12.86**	***3891.49***

Table 12. Joints costs

SUMMARY		
BEAMS	6257.18	euro
JOINTS	3891.49	euro
TOTAL	**10148.67**	**euro**

Table 13. Summary of costs

4.7.2.1 Genetic algorithm otpimization

The structure of the Genetic Algorithm leads to identifying at each generation the individuals that minimize the objective function and placing them in the next generation. As a result, the objective function curve obtained by this algorithm has a monotonically decreasing trend with the number of generations, as can be seen in Figure 50, Figure 51 and Figure 52. Analyzes were stopped when solution stagnation occurred, that is, there was no significant improvement in objective function for more than a set number of generations.

The speed of reaching the optimized solution depends not only on the number of variables involved in the problem, which thus affect the size of the search space, but also on the hyperparameters chosen for setting the algorithm. These include the mutation probability p_{mut}, which represents the probability with which the value of a gene is changed from one generation to the next. It is precisely the process of mutation, together with crossover, that allows for the exploration of the solution-finding space. High values of p_{mut} introduce greater changes between individuals of two successive generations. This allows for a faster rate of decrease in the fitness function in early generations, where one is farther away from the optimized solution and thus the introduction of a mutation is more likely to result in an improvement for the individual. Conversely, as one gets closer to the optimized solution, the changes are more unlikely to result in a benefit. Having a high mutation probability leads more easily to having multiple mutations in the same individual. In the case where this is already close to the optimized solution, only a small proportion of them are likely to constitute an improvement. The presence of both negative and beneficial mutations in the same individual thus leads to more difficult progress for the generation. This behavior can be seen in the graph in Figure 50. Thus, high values of p_{mut} are useful for "coarse" exploration of the solution space while they are not useful for fine improvements of a solution already with low values of the objective function. One might therefore consider adopting varying values of the mutation rate during the analysis, higher initially and decreasing as the generations proceed. This possibility was not analyzed in this thesis.

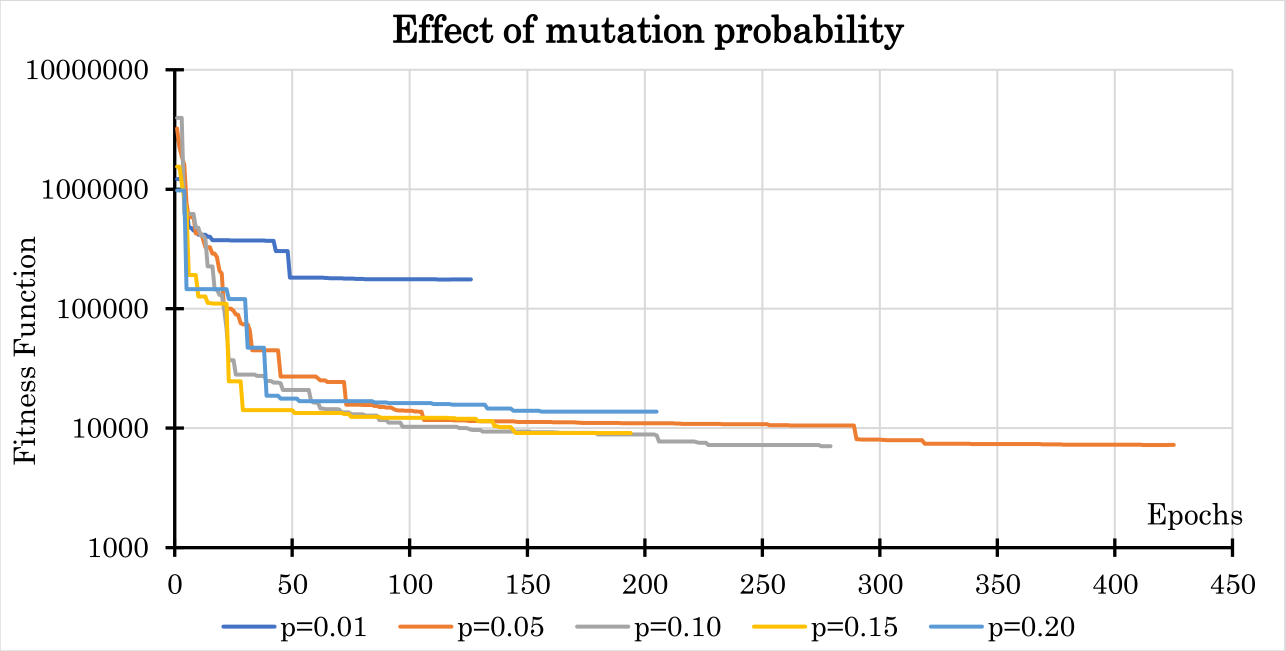


Figure 50. Influence of mutation probability p on the performance of the Genetic Algorithm

The number of individuals making up a generation also influences the rate of decrease of the objective function, as can be seen in Figure 51. Larger populations allow to better explore the solution space and therefore generally to obtain individuals with a smaller objective function for the same generation number. However, larger populations also imply a higher number of individuals to analyze and therefore greater computational cost and more time to solve a generation.

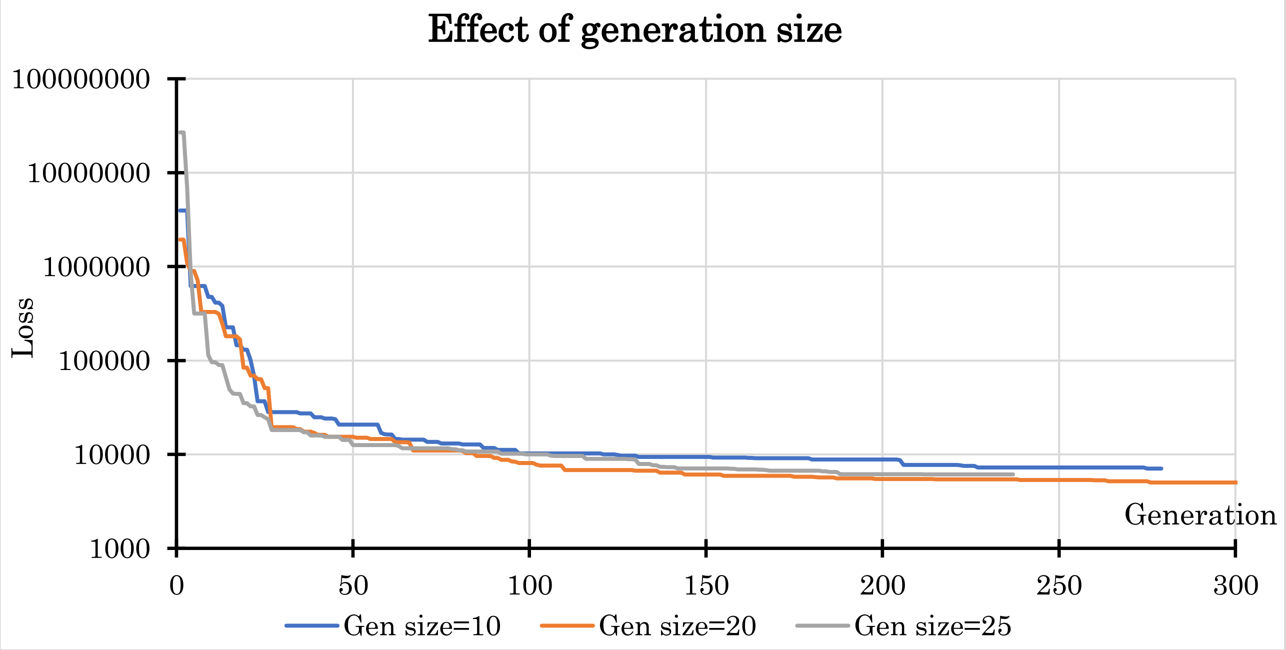


Figure 51. Influence of generation size on the performance of the Genetic Algorithm

Another aspect to consider is the composition of each generation, i.e., the percentage of best individuals and the percentage of random individuals taken from the previous generation. The remainder of the new generation will be created by mutation and crossover of pairs of individuals randomly selected from the previous generation. The Figure 52 shows three different generation compositions. The effects of generation composition are similar to those reported for the mutation probability rate p_{mut}. In fact, a lower overall percentage of individuals inherited from the previous generation (sum of the best ones and those chosen randomly) implies a greater exploration of the solution space and therefore a higher rate of decrease of the objective function in the first generations, similar to what was found for values of p_{mut} high. The increase in the percentage of new individuals in a generation has positive effects in the first generations but also implies a greater computational cost as the number of individuals for which analyzes have not yet been carried out increases.

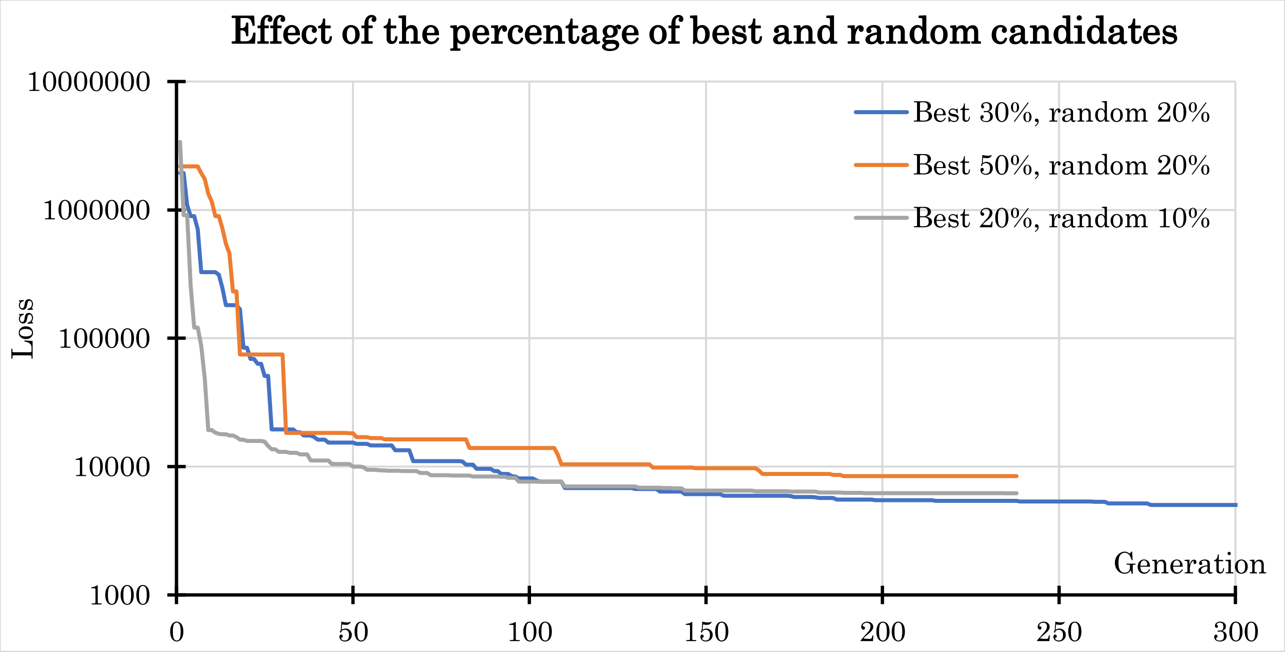


Figure 52. Influence of proportion of the best candidates and random candidates selected for the creation of the new generation on the performance of the Genetic Algorithm

If on the one hand the Genetic Algorithm has the advantage of offering each generation a solution equal to or better, but never worse than the previous generation, on the other hand it presents problems of local minimum, i.e., the solution found may not actually be a global minimum point in the solution space, and also presents problems of stagnation or of refining the solution. Furthermore, in the case where multiple mutations are carried out on the same individual, it does not allow to distinguish the effects of each of them in order to possibly keep only those that produce benefits.

Below are reported the results of the optimization of the multi-way node with the Genetic Algorithm and the comparison with the non-optimized solution.

BEAM	**DIAM [mm]**	**THK [mm]**	**LENGTH [mm]**
3134	273.00	12.00	12536.99
3207	273.00	6.00	15384.14
3804	323.90	6.30	7975.15
3866	177.80	10.00	12904.58
3887	273.00	6.30	15868.44
3925	355.60	6.00	15819.27
3926	355.60	6.00	15333.54
5124	508.00	12.00	14200.00
5142	457.00	16.00	14200.00

Table 14. Beam attributes of better solution from Genetic Algorithm

BEAM	**Joint End1**								**Joint End2**							
	NR	**Diam bolt [mm]**	**Nr rows**	**Nr col**	**Thk gusset [mm]**	**Thk covers [mm]**	**Throat [mm]**	**Lw [mm]**	**NR**	**Diam bolt [mm]**	**Nr rows**	**Nr col**	**Thk gusset [mm]**	**Thk covers [mm]**	**Throat [mm]**	**Lw [mm]**
3134	0	12	6	6	40	20	10	218	1	12	6	2	20	10	5	876
3207	2	12	6	4	50	25	6	613	3	12	5	8	40	20	10	949
3804	4	20	4	5	20	10	5	486	5	24	3	4	60	30	9	666
3866	6	16	4	1	15	10	8	789	7	16	4	1	55	30	10	173
3887	8	12	6	4	55	30	6	236	9	12	11	2	45	25	8	287
3925	10	16	6	2	25	15	3	339	11	20	5	1	25	15	4	822
3926	12	12	6	6	30	15	6	404	13	20	4	2	35	20	3	498
5124	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
5142	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 15. Joints attributes of better solution from Genetic Algorithm

NR	**ID**	**SECTION AREA [mm2]**	**VOLUME [mm3]**	**WEIGTH [kg]**	**UTIL. RATIO**	**COST [euro]**
3134	51510	9839.47	123357350.66	964.65	0.1831	**385.86**
3207	51610	5032.83	77425764.03	605.47	0.9712	**242.19**
3804	52310	6285.95	50131401.90	392.03	0.5214	**156.81**
3866	52510	5271.59	68027677.88	531.98	0.5622	**212.79**
3887	52610	5278.54	83762105.35	655.02	0.8509	**262.01**
3925	52411	6589.80	104245928.00	815.20	0.4277	**326.08**
3926	52410	6589.80	101045054.70	790.17	0.7510	**316.07**
5124	52210	18698.76	265522384.48	2076.39	0.9759	**830.55**
5142	52211	22167.08	314772504.29	2461.52	0.9666	**984.61**
TOTAL						***3716.97***

Table 16. Beams costs of better solution from Genetic Algorithm

NR	BEAM NR	STEEL GUSSET PLATE [euro]	STEEL COVER PLATES [euro]	BOLT [euro]	CUT GUSSET PLATE [euro]	CUT COVER PLATES [euro]	HOLE [euro]	PAINTING GUSSET PLATE [euro]	PAINTING COVER PLATES [euro]	WELD [euro]	UTIL. RATIO	COST [euro]
0	3134	19.91	21.26	14.53	2.86	4.41	16.72	1.14	2.43	0.75	0.230	**84.02**
1	3134	18.72	4.03	4.84	3.01	2.64	5.99	2.14	0.92	0.80	0.536	**43.10**
2	3207	38.86	18.33	9.69	4.16	4.06	14.58	1.78	1.68	0.80	0.948	**93.93**
3	3207	50.00	27.86	16.15	5.07	5.07	18.15	2.86	3.19	3.13	0.466	**131.48**
4	3804	19.24	16.05	87.12	2.87	4.23	8.13	2.20	3.68	0.46	0.542	**143.98**
5	3804	68.24	45.97	111.42	5.55	6.00	12.24	2.60	3.51	1.82	0.464	**257.35**
6	3866	10.53	2.51	5.58	2.46	2.36	4.53	1.61	0.58	1.74	0.720	**31.89**
7	3866	12.33	7.54	5.58	2.40	3.08	6.11	0.51	0.58	0.60	0.648	**38.72**
8	3887	23.74	22.00	9.69	3.15	4.36	16.19	0.99	1.68	0.33	0.811	**82.12**
9	3887	31.20	17.65	8.88	3.36	4.22	13.19	1.59	1.62	0.65	0.732	**82.35**
10	3925	15.05	9.74	16.75	2.43	3.32	7.09	1.38	1.49	0.13	0.947	**57.36**
11	3925	28.05	6.85	21.78	3.33	3.05	5.32	2.57	1.05	0.48	0.877	**72.47**
12	3926	25.72	20.47	14.53	3.11	4.29	13.50	1.96	3.12	0.54	0.531	**87.25**
13	3926	28.77	15.62	34.85	3.28	3.81	7.06	1.88	1.79	0.17	0.829	**97.22**
TOTAL		**390.37**	**235.88**	**361.39**	**47.03**	**54.91**	**148.78**	**25.21**	**27.29**	**12.38**		***1303.24***

Table 17. Joints costs of better solution from Genetic Algorithm

SUMMARY		
BEAMS	3716.97	euro
JOINTS	1303.24	euro
TOTAL	**5020.21**	**euro**

Table 18. Summary of costs of better solution from Genetic Algorithm

BEAM	Diam [mm]	Thk [mm]	Weigth [kg]	COST [euro]
3134	-23%	+90%	+50%	+50%
3207	-33%	-25%	-46%	-46%
3804	+19%	+26%	+42%	+42%
3866	-45%	+79%	0%	+0%
3887	-16%	+13%	-1%	-1%
3925	-42%	-45%	-62%	-62%
3926	-30%	-25%	-41%	-41%
5124	-38%	-25%	-53%	-53%
5142	-44%	0%	-45%	-45%

Table 19. Variation of beam characteristics between the original model and the model optimized with the Genetic Algorithm

JOINT	Diam bolt [mm]	Nr rows	Nr col	Thk gusset [mm]	Thk covers [mm]	Throat [mm]	Lw [mm]	COST [euro]
0	-60%	+50%	+200%	+60%	+33%	+25%	-55%	-62%
1	-60%	+50%	0%	-20%	-33%	-38%	+83%	-81%
2	-60%	+20%	+100%	+100%	+67%	+20%	+11%	-66%
3	-60%	0%	+300%	+60%	+33%	+100%	+73%	-52%
4	-33%	0%	+400%	0%	-33%	0%	+39%	+18%
5	-20%	-25%	+300%	+200%	+100%	+80%	+90%	+111%
6	-47%	0%	-50%	-40%	-33%	0%	+64%	-85%
7	-47%	0%	-50%	+120%	+100%	+25%	-64%	-82%
8	-60%	+50%	+300%	+175%	+200%	+20%	-41%	-28%
9	-60%	175%	+100%	+125%	+150%	+60%	-28%	-28%
10	-47%	0%	-33%	-29%	-17%	-63%	-58%	-89%
11	-33%	-17%	-67%	-29%	-17%	-50%	+3%	-86%
12	-60%	0%	+100%	+20%	0%	-14%	-46%	-81%
13	-33%	-33%	-33%	+40%	+33%	-57%	-34%	-79%

Table 20. Variation of joint characteristics between the original model and the model optimized with the Genetic Algorithm

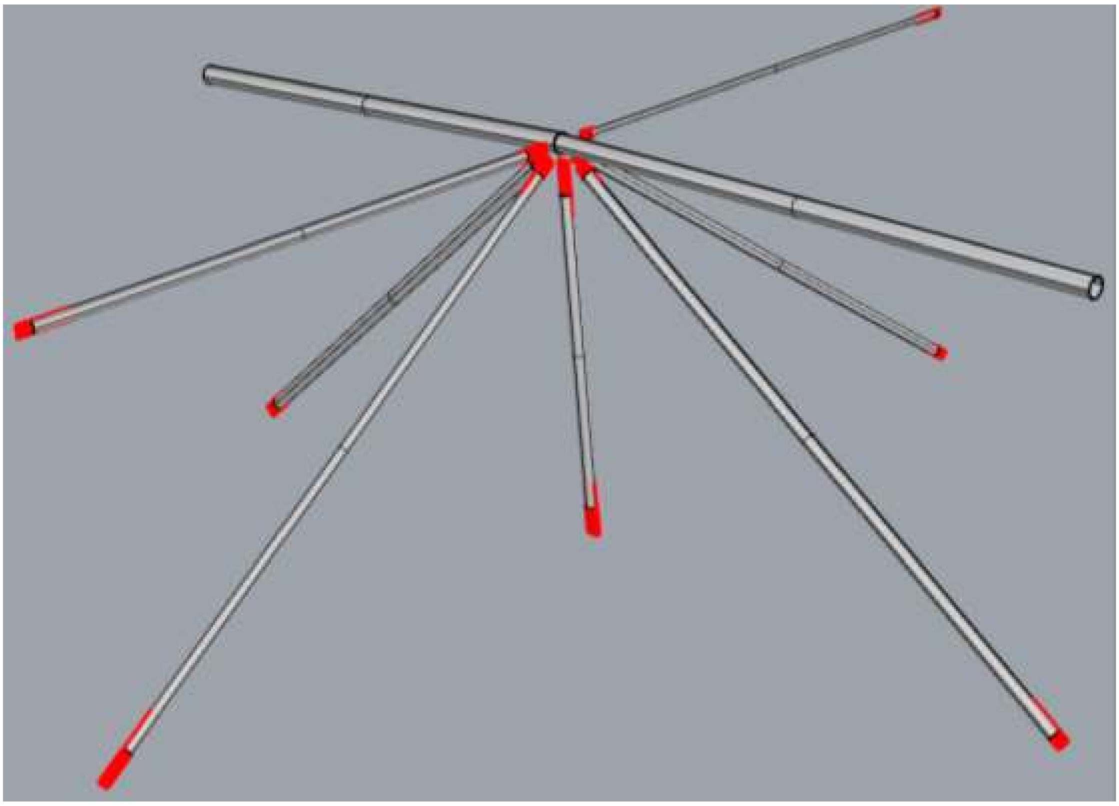


Figure 53. Model of multi-way node optimized with the Genetic Algorithm

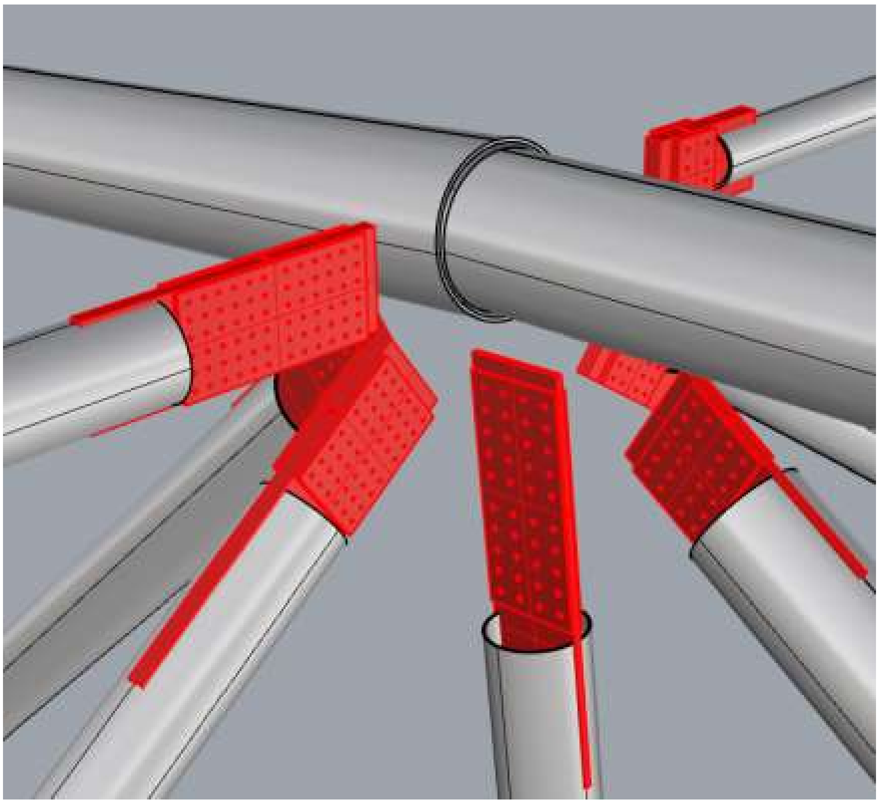


Figure 54. Model of multi-way node optimized with the Genetic Algorithm

4.7.2.2 Zeroth-order block coordinate descent optimization

The ability to explore the solution field, as well as the computational cost required by the algorithm, are a function of the hyperparameters used to set up the algorithm. Increasing the number of blocks into which the variables are divided, in addition to implying fewer variables in each block, results in a reduction in the number of directions sampled. The latter behavior can be attributed to the fact that by reducing the number of concurrently changed variables, it is sufficient to investigate fewer directions to assess which of the changed variables have beneficial effects and which have negative effects on the value of the fitness function. Instead, increasing the level of sparsity *s*, corresponding to greater dependence among the variables involved, results in an increase in the number of directions analyzed n_{dir}.

The computational cost of the algorithm is a function of the number of solutions analyzed. Consequently, it turns out to be dependent on the number of directions sampled n_{dir} while there is no dependence on the number of variables in each block.

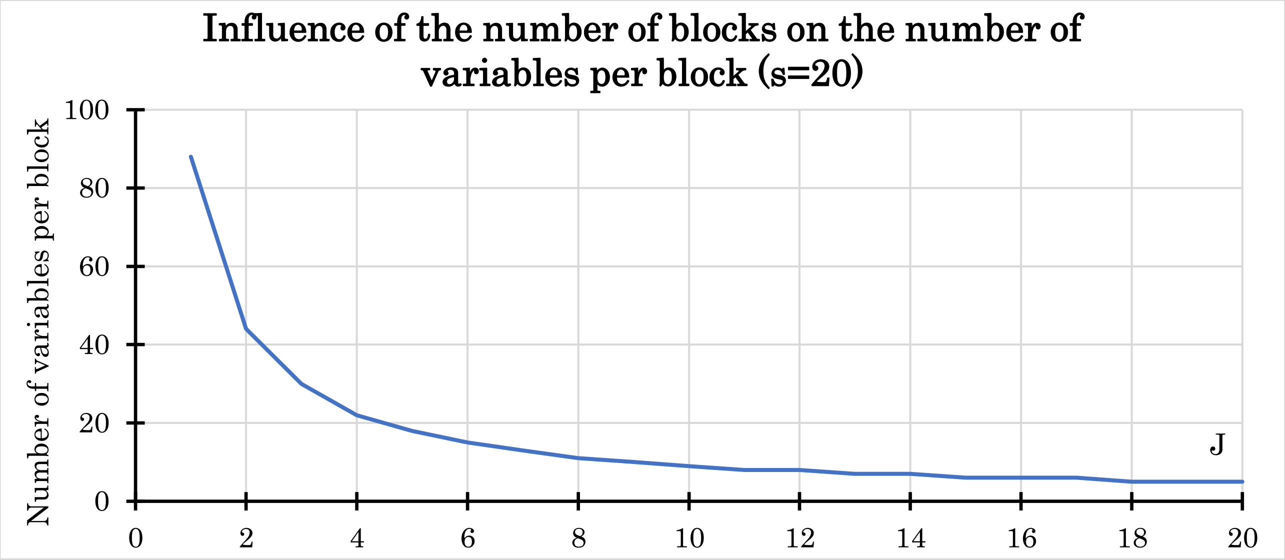


Figure 55. Trend of the number of variables per block as function of the number of blocks J, with degree of sparsity s equal to 20

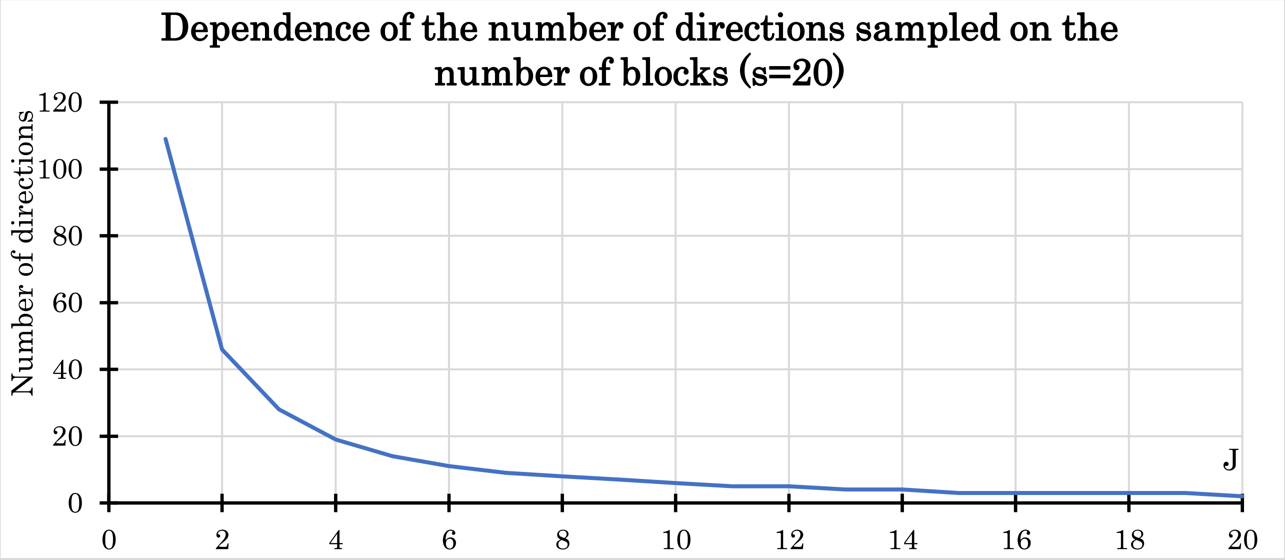


Figure 56. Trend of the number of directions sampled n_{dir} as function of the number of blocks J, with degree of sparsity s being 20

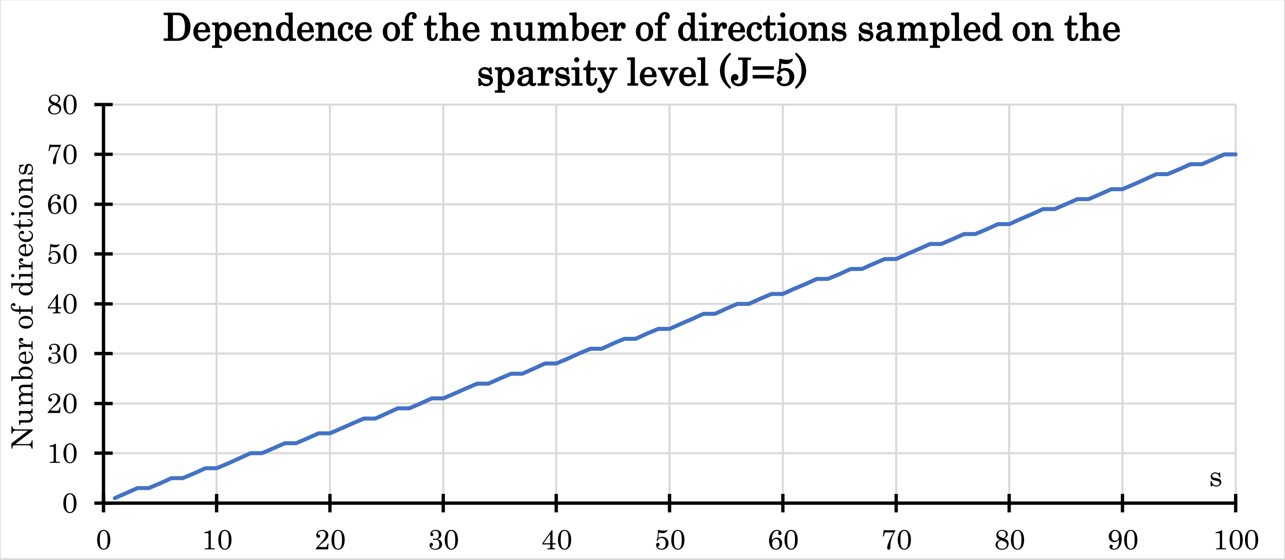


Figure 57. Trend of the number of directions sampled n_{dir} as function of the level of sparsity s, with the number of blocks J being 5

Numerous tests were conducted to find the best hyperparameter configuration for the case of study. The outcomes of these tests can be seen in the graphs in Figure 58 and Figure 59. The lower rate of decrease in the fitness function that occurs for lower levels of sparsity, and found especially after the first few iterations, show that more sampling directions are needed to indicate which changes to apply to the variables belonging to the chosen block have beneficial effects. However, too high values of sparsity degrees may lead to confusion in the gradient estimation algorithm and thus not translate into a benefit in exploring the field of finding the optimal solution. In addition, an increase in the number of directions investigated results in more individuals to be analyzed and thus an increase in computational cost. It follows that the level of sparsity used for the analysis should result from a compromise between these different aspects. Using fewer blocks, and thus more variables belonging to the same block, allows the algorithm to converge more quickly to lower objective function values. However, the presence of a larger number of variables per block implies a larger number of directions analyzed to distinguish the effects of each variable belonging to the selected block and thus a higher computational cost.

From the graphs in Figure 58 and Figure 59, it can be seen that compared with the Genetic Algorithm, the behavior of the objective function curve is not monotonically decreasing with the number of iterations. This behavior can be attributed from the possibility of failure by the gradient estimation algorithm that results in a change in the value of variables between the current and next iteration with increase in the value of the objective function. Consequently, more attention should be paid to parameter setting with this type of algorithm. It also implies that the best result does not necessarily occur at the last iteration performed.

Another difference with the Genetic Algorithm concerns the fact that information obtained from the gradient is exploited, and thus if even the change in the value of a variable has a negative effect on the objective function, that variable will undergo a change of the opposite sign during the creation of the ultimate individual of that iteration.

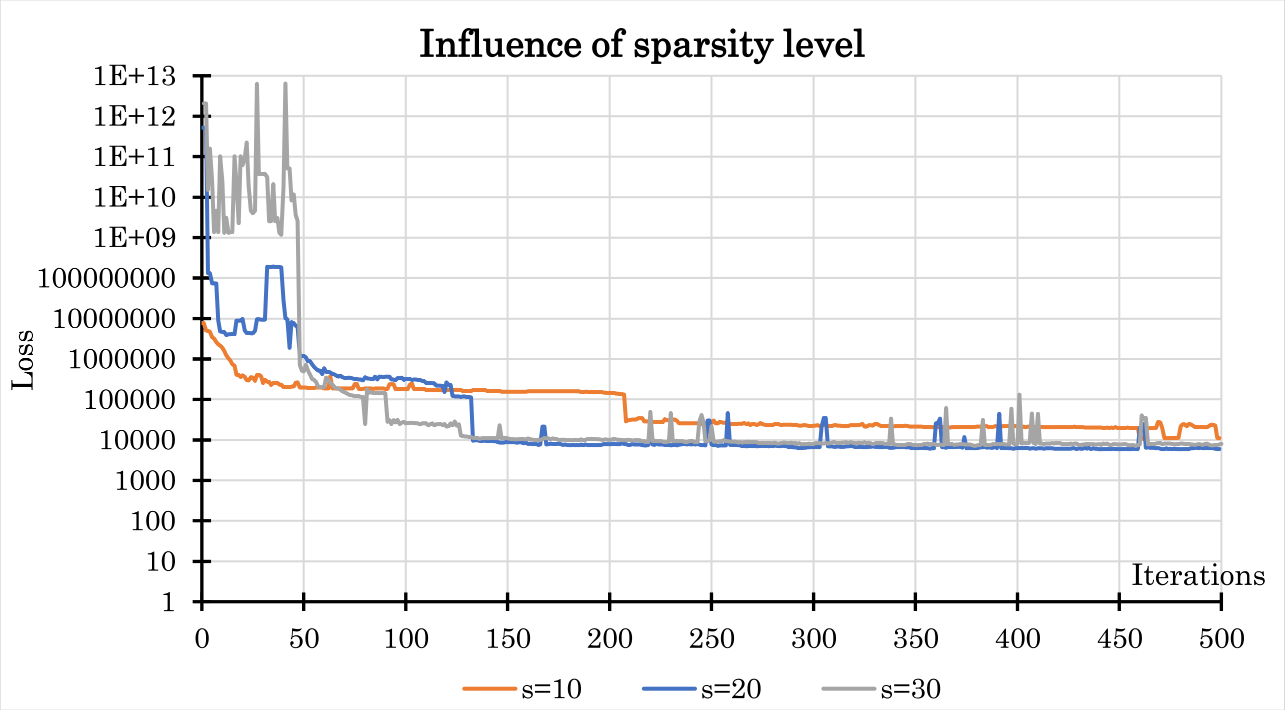


Figure 58. Influence of sparsity level on the performance of the Gradient-based Algorithm

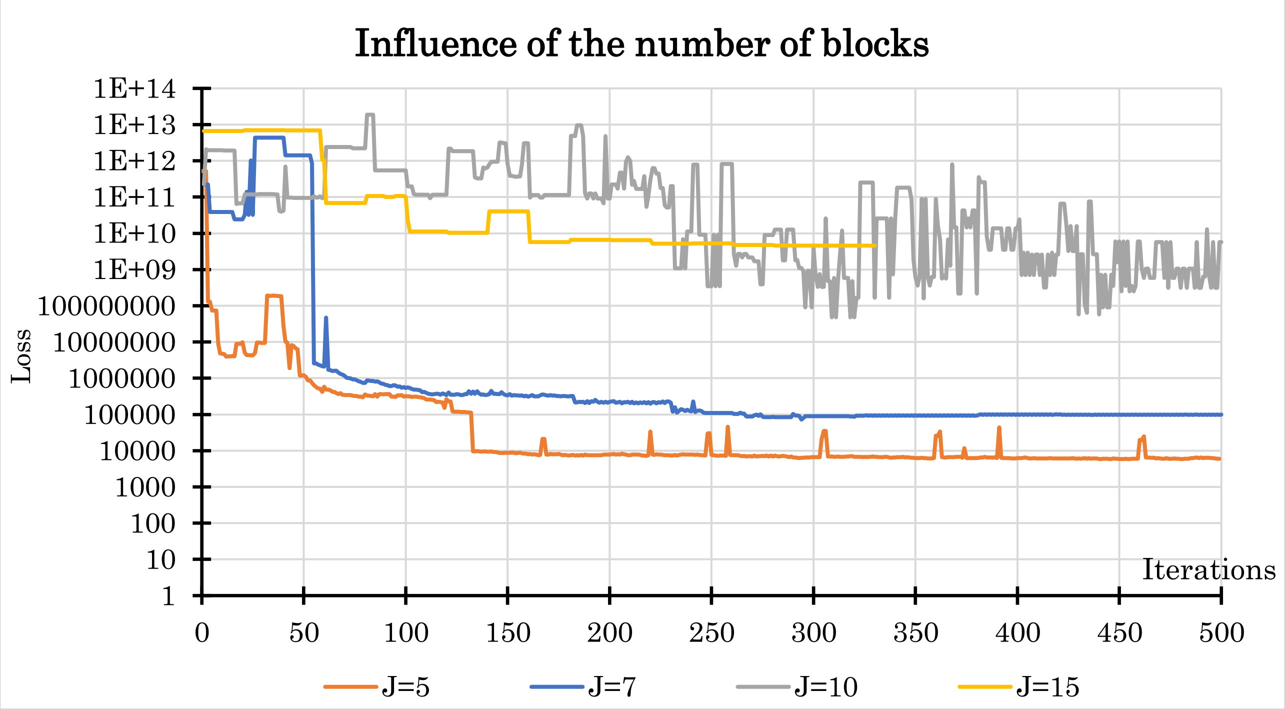


Figure 59. Influence of number of blocks of variables on the performance of the Gradient-based Algorithm

Below are reported the results of the optimization of the multi-way node with the Zeroth-Order Block Coordinate Descendent Algorithm and the comparison with the non-optimized solution.

BEAM	DIAM [mm]	THK [mm]	LENGTH [mm]
3134	236.10	8.00	12385.40
3207	166.40	9.00	15108.93
3804	139.70	6.19	7916.08
3866	203.90	10.00	12173.10
3887	228.30	5.50	15522.93
3925	388.90	7.03	15145.26
3926	470.00	7.89	15149.16
5124	579.10	13.24	14200.00
5142	696.10	12.02	14200.00

Table 21. Beam attributes of better solution from Gradient-based Algorithm

BEAM	Joint End1								Joint End2							
	NR	Diam bolt [mm]	Nr rows	Nr col	Thk gusset [mm]	Thk covers [mm]	Throat [mm]	Lw [mm]	NR	Diam bolt [mm]	Nr rows	Nr col	Thk gusset [mm]	Thk covers [mm]	Throat [mm]	Lw [mm]
3134	0	19	5	2	20	10	8	934	1	12	5	8	65	33	5	458
3207	2	12	4	7	25	13	10	293	3	20	2	6	25	13	9	598
3804	4	12	4	7	50	25	6	797	5	12	3	10	75	38	5	867
3866	6	12	5	6	60	30	7	228	7	12	5	8	25	13	7	1037
3887	8	20	3	2	80	40	4	640	9	12	4	9	30	15	9	479
3925	10	12	8	5	80	40	7	904	11	12	8	5	75	38	10	706
3926	12	12	10	4	35	18	5	1100	13	20	7	4	40	20	9	711
5124	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
5142	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 22. Joints attributes of better solution from Gradient-based Algorithm

NR	ID	SECTION AREA [mm2]	VOLUME [mm3]	WEIGTH [kg]	UTIL. RATIO	COST [euro]
3134	51510	5732.78	71002778.40	555.24	0.6313	**222.10**
3207	51610	4450.38	67240481.90	525.82	0.5352	**210.33**
3804	52310	2596.30	20552485.15	160.72	0.3540	**64.29**
3866	52510	6091.55	74153015.44	579.88	0.4321	**231.95**
3887	52610	3849.71	59758729.44	467.31	0.5690	**186.93**
3925	52411	8433.75	127731378.15	998.86	0.6214	**399.54**
3926	52410	11454.40	173524538.94	1356.96	0.6000	**542.78**
5124	52210	23536.77	334222125.92	2613.62	0.9078	**1045.45**
5142	52211	25832.19	366817104.36	2868.51	0.9501	**1147.40**
TOTAL						***4050.77***

Table 23. Beams costs of better solution from Gradient-based Algorithm

NR	BEAM NR	STEEL GUSSET PLATE [euro]	STEEL COVER PLATES [euro]	BOLT [euro]	CUT GUSSET PLATE [euro]	CUT COVER PLATES [euro]	HOLE [euro]	PAINTING GUSSET PLATE [euro]	PAINTING COVER PLATES [euro]	WELD [euro]	UTIL. RATIO	COST [euro]
0	3134	23.63	6.41	34.31	3.27	2.99	5.94	2.71	1.47	2.05	0.679	**82.78**
1	3134	44.31	37.10	16.15	4.73	5.81	27.10	1.56	2.61	0.43	0.265	**139.80**
2	3207	9.37	9.15	11.30	2.31	3.65	10.10	0.86	1.68	0.99	0.483	**49.42**
3	3207	15.44	12.37	52.27	3.15	4.31	7.06	1.41	2.27	1.64	0.454	**99.92**
4	3804	34.03	18.60	11.30	4.77	4.56	16.36	1.56	1.70	1.03	0.384	**93.91**
5	3804	50.34	32.96	12.11	6.80	6.59	23.97	1.54	2.01	0.79	0.461	**137.10**
6	3866	22.45	22.78	12.11	3.28	4.69	19.94	0.86	1.74	0.42	0.442	**88.27**
7	3866	25.00	12.44	16.15	3.99	4.01	12.79	2.29	2.28	1.77	0.348	**80.70**
8	3887	51.97	19.82	26.14	5.29	4.28	8.98	1.49	1.13	0.38	0.588	**119.47**
9	3887	21.29	18.55	14.53	3.24	4.57	13.50	1.62	2.83	1.32	0.180	**81.45**
10	3925	119.54	46.88	16.15	7.42	5.89	32.46	3.42	2.68	1.55	0.907	**235.99**
11	3925	93.07	43.95	16.15	6.30	5.71	30.67	2.84	2.68	2.34	0.883	**203.70**
12	3926	70.92	20.06	16.15	5.10	4.19	16.36	4.64	2.62	0.99	0.995	**141.04**
13	3926	63.58	35.68	121.97	4.72	5.14	15.83	3.64	4.08	1.94	0.531	**256.58**
TOTAL		**644.93**	**336.75**	**376.79**	**64.36**	**66.38**	**241.07**	**30.43**	**31.79**	**17.64**		***1810.13***

Table 24. Joints costs of better solution from Gradient-based Algorithm

SUMMARY		
BEAMS	4050.77	euro
JOINTS	1810.13	euro
TOTAL	**5860.90**	**euro**

Table 25. Summary of costs of better solution from Gradient-based Algorithm

BEAM	Diam [mm]	Thk [mm]	Weigth [kg]	COST [euro]
3134	-34%	+27%	-14%	-14%
3207	-59%	+13%	-53%	-53%
3804	-49%	+24%	-42%	-42%
3866	-37%	+79%	+9%	+9%
3887	-30%	-2%	-30%	-30%
3925	-36%	-36%	-54%	-54%
3926	-7%	-1%	+1%	+1%
5124	-29%	-17%	-41%	-41%
5142	-14%	-25%	-35%	-35%

Table 26. Variation of beam characteristics between the original model and the model optimized with the Gradient-based Algorithm

JOINT	Diam bolt [mm]	Nr rows	Nr col	Thk gusset [mm]	Thk covers [mm]	Throat [mm]	Lw [mm]	COST [euro]
0	-37%	+25%	0%	-20%	-33%	0%	+95%	-63%
1	-60%	+25%	+300%	+160%	+117%	-38%	-5%	-37%
2	-60%	-20%	+250%	0%	-17%	+100%	-47%	-82%
3	-33%	-60%	+200%	0%	-17%	+80%	+9%	-64%
4	-60%	0%	+600%	+150%	+67%	+20%	+128%	-23%
5	-60%	-25%	+900%	+275%	+150%	0%	+148%	+12%
6	-60%	+25%	+200%	+140%	+100%	-13%	-53%	-60%
7	-60%	+25%	+300%	0%	-17%	-13%	+116%	-63%
8	-33%	-25%	+100%	+300%	+300%	-20%	+60%	+4%
9	-60%	0%	+800%	+50%	+50%	+80%	+20%	-29%
10	-60%	+33%	+67%	+129%	+122%	-13%	+13%	-55%
11	-60%	+33%	+67%	+114%	+108%	+25%	-12%	-61%
12	-60%	+67%	+33%	+40%	+17%	-29%	+47%	-70%
13	-33%	+17%	+33%	+60%	+33%	+29%	-5%	-45%

Table 27. Variation of joint characteristics between the original model and the model optimized with the Gradient-based Algorithm

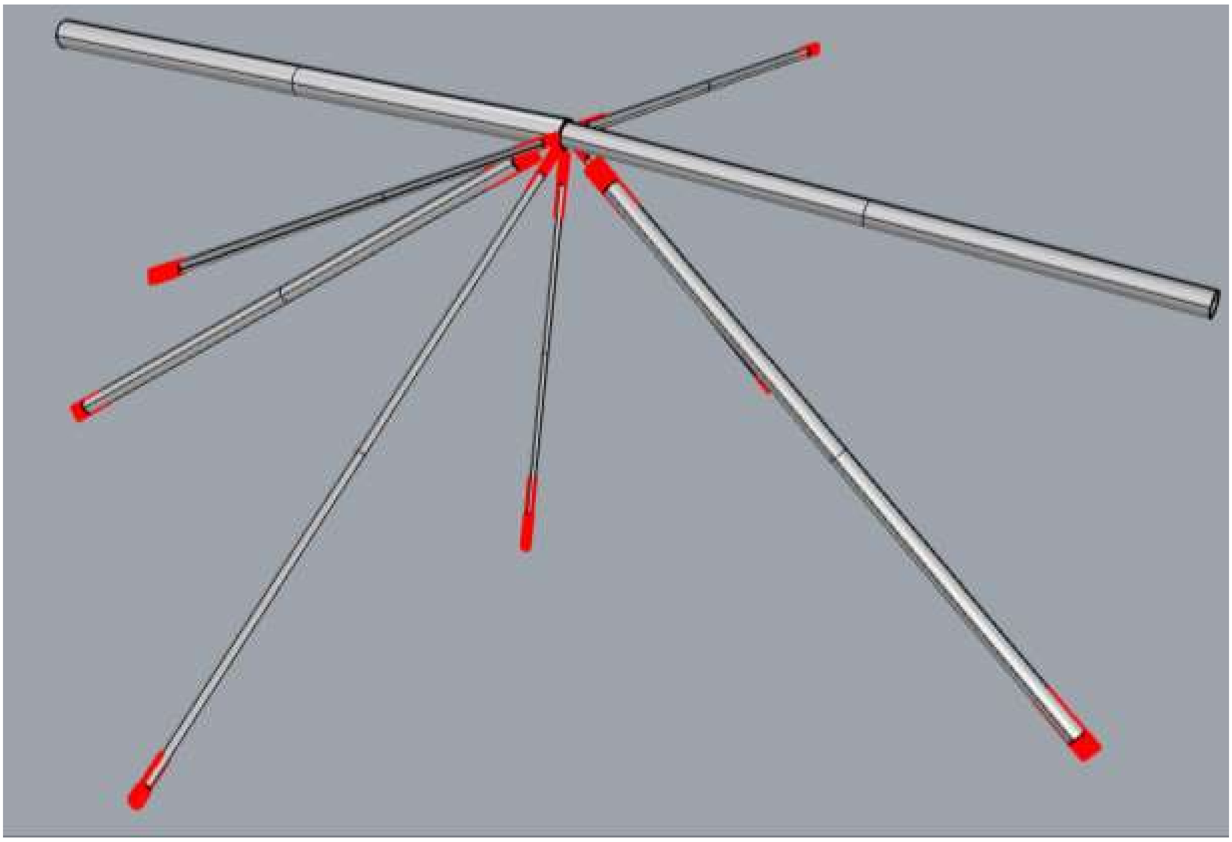


Figure 60. Model of multi-way node optimized with the Gradient-based Algorithm

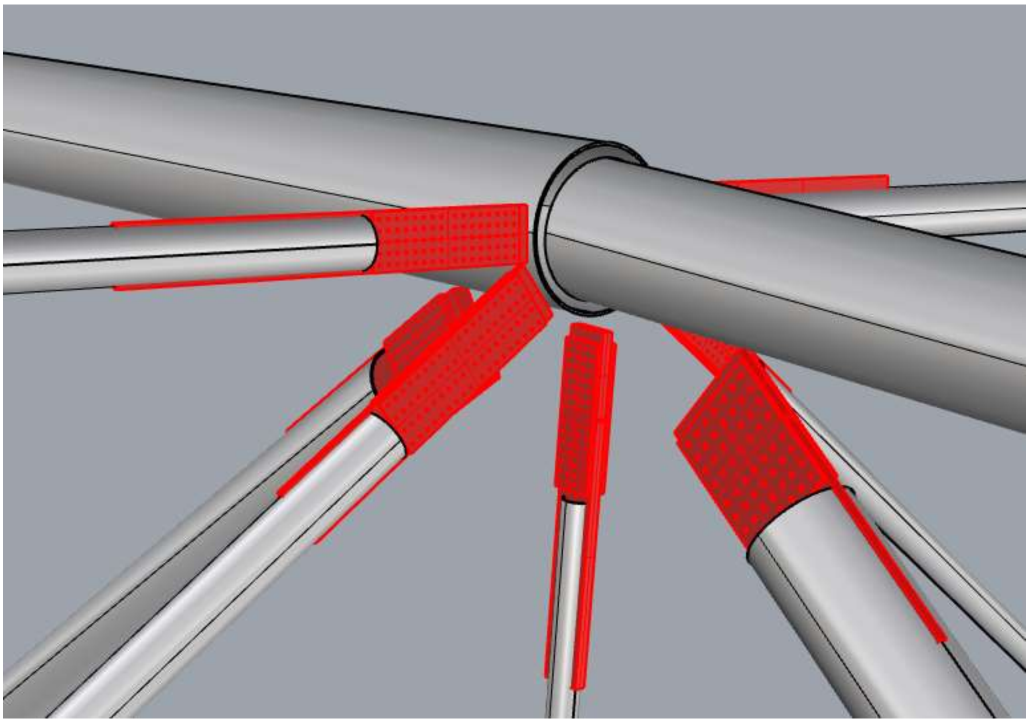


Figure 61. Model of multi-way node optimized with the Gradient-based Algorithm

4.7.2.3 Model comparison

The percentage change in cost between the best results obtained with the Genetic Algorithm and with the Gradient-based Algorithm and the initial model is shown below.

NR	**BEAM COST [euro]**				
	Initial model	**GA Algorithm**		**Gradient-based Algorithm**	
3134	257.61	385.86	+50%	222.10	-14%
3207	451.76	242.19	-46%	210.33	-53%
3804	110.39	156.81	+42%	64.29	-42%
3866	212.20	212.79	+0%	231.95	+9%
3887	265.26	262.01	-1%	186.93	-30%
3925	869.10	326.08	-62%	399.54	-54%
3926	535.70	316.07	-41%	542.78	+1%
5124	1777.43	830.55	-53%	1045.45	-41%
5142	1777.74	984.61	-45%	1147.40	-35%
TOTAL	**6257.18**	**3716.97**	**-40%**	**4050.77**	**-35%**

Table 28. Comparison of beam cost between initial model and model optimized with Genetic Algorithm and with Gradient-based Algorithm

NR	JOINT COST [euro]				
	Initial model	GA Algorithm		Gradient-based Algorithm	
0	222.17	84.02	-62%	82.78	-63%
1	222.17	43.10	-81%	139.80	-37%
2	276.45	93.93	-66%	49.42	-82%
3	276.45	131.48	-52%	99.92	-64%
4	122.00	143.98	+18%	93.91	-23%
5	122.00	257.35	+111%	137.10	+12%
6	218.09	31.89	-85%	88.27	-60%
7	218.09	38.72	-82%	80.70	-63%
8	114.47	82.12	-28%	119.47	+4%
9	114.47	82.35	-28%	81.45	-29%
10	528.55	57.36	-89%	235.99	-55%
11	528.55	72.47	-86%	203.70	-61%
12	464.00	87.25	-81%	141.04	-70%
13	464.00	97.22	-79%	256.58	-45%
TOTAL	**3891.49**	**1303.24**	**-67%**	**1810.13**	**-53%**

Table 29. Comparison of joint cost between initial model and model optimized with Genetic Algorithm and with Gradient-based Algorithm

NR	TOTAL COST [euro]				
	Initial model	GA Algorithm		Gradient-based Algorithm	
Beam cost	6257.18	3716.97	-40%	4050.77	-35%
Joint cost	3891.49	1303.24	-67%	1810.13	-53%
TOTAL	**10148.67**	**5020.21**	**-51%**	**5860.90**	**-42%**

Table 30. Comparison of total cost between initial model and model optimized with Genetic Algorithm and with Gradient-based Algorithm

The two optimization algorithms resulted in different solutions to each other, both with lower costs than the starting model. Despite the difference in results, similarities can be observed between the two solutions. For example, the tendency to decrease the diameter of the beams can be observed. This indicates that the beams in the starting model were oversized. Also, it can be observed that in both solutions in all joints the size of the bolt diameters was reduced and in most cases the number of bolts was increased. The tendency to reduce the diameter, preferring an increase in the quantity of bolts, is due to the quadratic dependence of bolt cost on bolt diameter within the cost function.

The computational cost is mainly due to the analyzes carried out on a solution to calculate the fitness function. The operations for calculating the objective function on a solution are the same for the two algorithms, so the computational cost is related only to the number of solutions to be analyzed.

With the Genetic Algorithm, the best result was achieved with a population of 20 individuals made up of 30% of the best individuals of the previous generation, 20% of random individuals of the previous generation while the remaining 50% consists of new individuals obtained by crossover and mutations. The latter are the population component that must be analyzed. The number of iterations needed to reach the best solution was 276.

With the Gradient-based Algorithm the best results have been achieved with a degree of sparsity s equal to 20 and 5 blocks of variables. Consequently, the number of directions investigated and therefore of solutions analyzed is equal to 14, to which is added the solution calculated at the end of the iteration and obtained by changing the variables defined on the basis of the gradient estimate. The best solution was obtained after 455 iterations.

From the above results it appears that the Genetic Algorithm, in addition to being easier to use and set up, leads to better results and with a lower computational cost than the Gradient-based Algorithm. However, the identification of an update function of the single variables that can better exploit the value of the estimated gradient could lead in the future to an important reduction in the number of iterations necessary to reach the optimized solution.

4.8 Conclusion

Using a multi-way node as a case study, this chapter highlights the large number of variables involved in the design problem, as already highlighted in the initial chapter of the thesis, and how it grows rapidly with the number of elements that make up a structure. The formulation of the design problem as an optimization problem requires preliminary investigations of the type of beams to be adopted and the type of joints to be made, based on which the variables are defined to describe the problem and the verifications required by the standard to evaluate structural performance are specified. Investigations of stock availability allow the narrowing of the search for the solution by promoting the convergence of the optimization algorithm. In this chapter, two optimization approaches, one based on metaheuristics (Genetic Algorithm) and one based on gradient (Gradient-based Algorithm), were presented and applied. Both resulted in optimized solutions, but the first one was more performant, both in terms of results and computational resources. The gradient-based approach was more complicated to apply because of the non-derivability of the objective function and the presence of variables of different types and different scales, which made it difficult to find a solution update function at each iteration based on the gradient. Future research could lead to the development of better performing update functions that would improve the performance of the method. A third optimization approach has only been described and not applied because it is still subject to study. This last method aims to introduce groupings between similar beams and joints to speed up the achievement of the optimized solution.

CHAPTER 5: Optimization of a steel frame

The goal is to create a workflow that can quickly return an optimized dimensioning of a framed structure. In particular, the aim is to define the section of the beams belonging to the structure. The choice of these sections should be such as to minimize costs, but at the same time meet the structural requirements dictated by the standards. For each beam that makes up the structure, the structural checks depend on the stresses affecting that element, which are traditionally calculated using FEM software, and depend on other geometric attributes, such as section and beam length. The high computational cost of solving FEM models is a major limitation in the number of analyses that can be performed. From these considerations, a picture emerges in which rapid optimization of the sections of the members of a framed structure requires the creation of tools that from the stresses and possibly other geometric characteristics of the member (such as length) return the section that provides the chosen utilization rate, and a tool that allows the beam forces to be updated as the applied loads change so that the change in self-weight due to the change in sections can also be considered in an approximate way.

In Section 5.1 , the proposed workflow for optimizing the beams of a frame structure is presented, and the modules that comprise it and how they are interconnected are shown. Section 5.2 aims to create a surrogate model to be used in the optimization tool to quickly update the forces on the beams as loads change, so as to accelerate the optimizer's convergence. The next section, on the other hand, describes the creation and training of a neural network that provides for each beam the cross section that, based on the acting forces and the length of the element, will result in the element having the desired utilization rate. The performance of the optimizer and the various specially created modules was evaluated by application to a two-dimensional frame structure and to a three-dimensional frame structure.

5.1 Proposal workflow

As mentioned above, the choice of section to be assigned to each beam should be such as to minimize the cost of the structure, consistent with meeting structural checks. Unlike in CHAPTER 4: , joints are not considered in this study. Consequently, since the processing and various components required to make the joints are not involved, cost optimization can be considered as a minimization of the weights of the beams, that is, a search for the section with minimum area that can be attributed to each beam. The search for the minimum beam cross-sections that meet the requirements of structural verifications can be translated into the search for cross-sections that lead to the maximum allowable utilization rate in each element. In this way, all beams in the frame are well utilized. It is therefore necessary to create a tool that, given as input the beam forces, length and desired utilization rate, outputs the beam cross section. To build such a model, a *Multi-Layer Perceptrons* (MLP) Neural Network was chosen, which can be trained using a database of already known cases or a purpose-built database. Updating the self-weight of the beams due to the modification of the beam cross sections by using MLP leads to a change in the forces on the beams. The new beam forces can be calculated by updating the starting FEM model with the new cross sections and solving it again or by constructing a surrogate model. The latter approach is proposed to provide an estimate of the new stresses in much less time than the solution of the updated FEM model would require.

The surrogate models made for this workflow are structure-specific: generalizations to apply the models to structures other than the one they were trained with are the subject of future development.

The proposed workflow is summarized in the diagram in the Figure 62. The input to the process is the FEM model of the frame to be optimized and its solution, which provides information on the tensional state of the beams for the various loading conditions to which they are subjected. From this model, information on the loads and forces on the beams is derived for use both in the MLP that provides the optimized section and for the construction of the surrogate model. Using the tensional state of the starting model solution and other geometric characteristics of the beams, initial proposed sections for each beam are obtained from the MLP. The change in load correlated with the change in the weight of the beams results in a change in the forces on the beams, which is estimated using a surrogate model or calculated accurately by updating the FEM model and solving it. The new set of forces on the beams is used to obtain a new section proposal from the MLP, which will result in an additional load change. The process continues in this way until convergence.

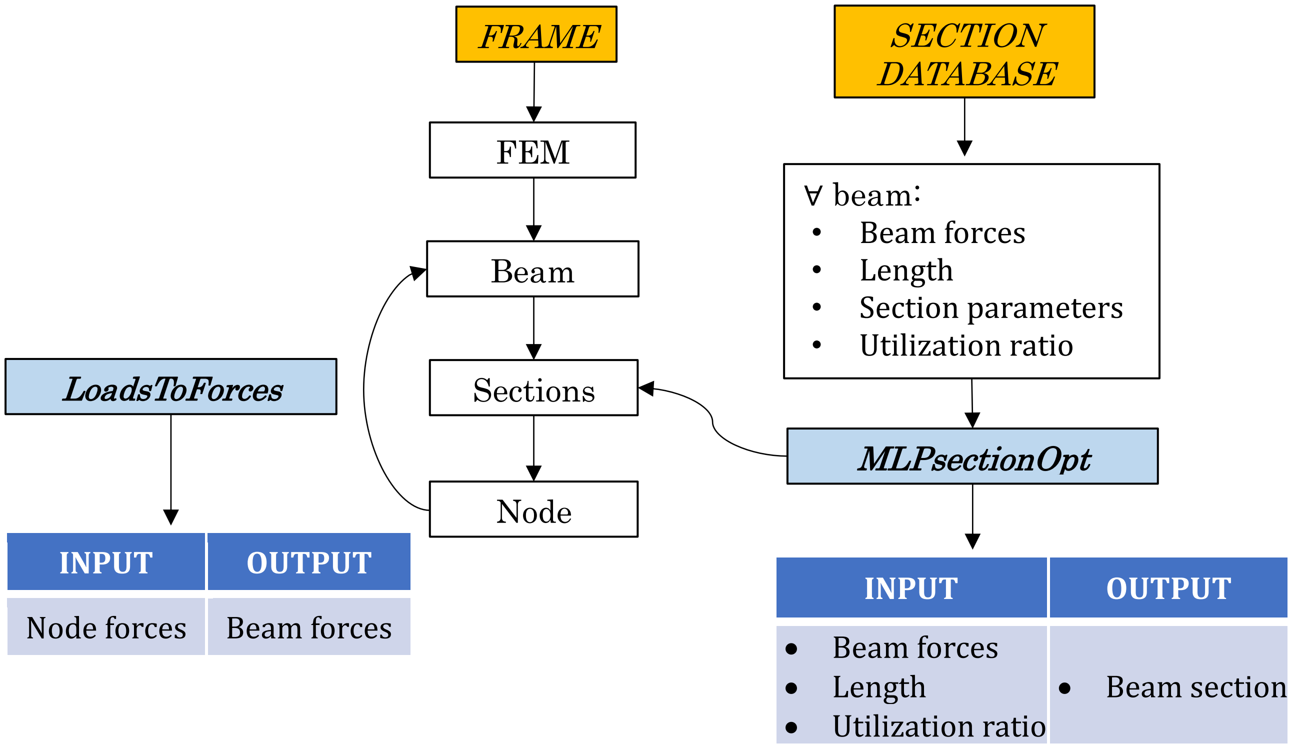


Figure 62. Proposal workflow for steel frame optimization

5.2 *LoadsToForces* module

LoadsToForces identifies the tool for updating forces on beams as loads change. It can consist of a module that updates the FEM model with the new load state and performs the subsequent recalculation, or it can be a surrogate model. The high computational cost associated with solving large FEM models makes the latter option attractive. The construction of a surrogate model makes it necessary to identify a model that can capture the link between loads and forces at the beams in a subspace of lower dimension than that of the real problem. Several alternatives were evaluated:

- Low-rank stiffness matrix
- MLP model
- GNN model

The following shows how the tensional state of the beams and the load acting on the structure are represented for surrogate model creation. Subsequently, the investigated approaches are explained in detail.

5.2.1 Representation of model loads and forces on beams

The stress state at any point of the beam can be described by 6 parameters:

$$s = \begin{pmatrix} SF_1 \\ BM_1 \\ SF_2 \\ BM_2 \\ AX \\ TQ \end{pmatrix} \tag{5.1}$$

where *SF1* and *SF2* are the shear forces, *BM1* and *BM2* are the bending moments, *AX* is the axial force while *TQ* is the torque moment, all expressed in the principal reference system of the i-th beam. Through these parameters, the strength utilization rate of the element can be assessed.

The 6 stress parameters at the two ends of the i-th beam, shown in the Figure 63, can be written in the following vector:

$$s_i = \begin{pmatrix} SF_{1,i}^{end1} \\ BM_{1,i}^{end1} \\ SF_{2,i}^{end1} \\ BM_{2,i}^{end1} \\ AX_i^{end1} \\ TQ_i^{end1} \\ SF_{1,i}^{end2} \\ BM_{1,i}^{end2} \\ SF_{2,i}^{end2} \\ BM_{2,i}^{end2} \\ AX_i^{end2} \\ TQ_i^{end2} \end{pmatrix} \qquad \text{with i=1, ..., b} \tag{5.2}$$

For a better description of the trend of the stress state along the development of the beam, it is possible to increase the size of the vector by also including stress parameters related to other positions along the beam. The stress vectors of all the beams in the model can be collected into a single matrix [*b* x 12]:

$$S = \begin{bmatrix} s_1^T \\ \dots \\ s_b^T \end{bmatrix} \tag{5.3}$$

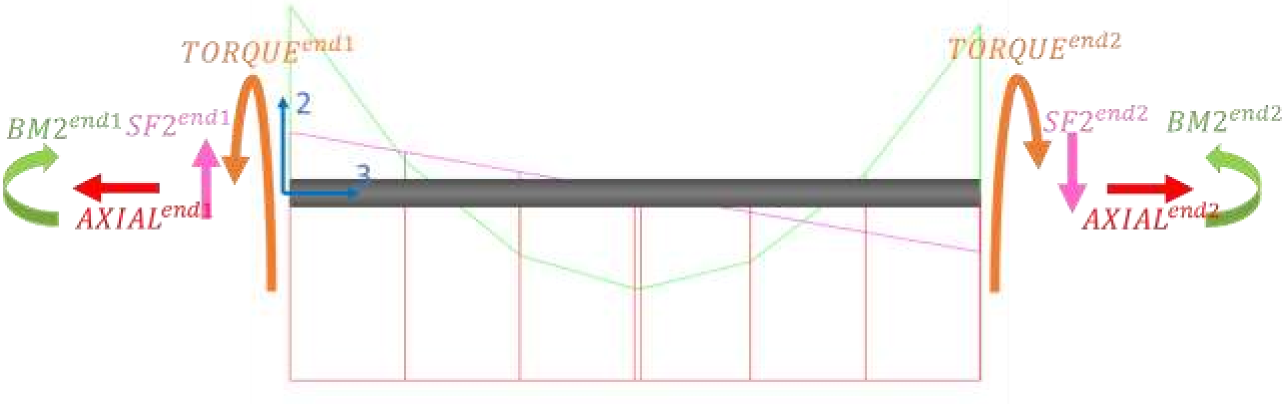


Figure 63. Beam end forces

The stress parameters at the beam ends are related to the node forces, that is, the forces and moments acting on the beam nodes.

$$F_1^{end1} = SF_1^{end1} \tag{5.4}$$

$$M_1^{end1} = BM_2^{end1} \tag{5.5}$$

$$F_2^{end1} = SF_2^{end1} \tag{5.6}$$

$$M_2^{end1} = -BM_1^{end1} \tag{5.7}$$

$$F_3^{end1} = -AXIAL^{end1} \tag{5.8}$$

$$M_3^{end1} = -TORQUE^{end1} \tag{5.9}$$

$$F_1^{end2} = -SF_1^{end2} \tag{5.10}$$

$$M_1^{end2} = -BM_2^{end2} \tag{5.11}$$

$$F_2^{end2} = -SF_2^{end2} \tag{5.12}$$

$$M_2^{end2} = BM_1^{end2} \tag{5.13}$$

$$F_3^{end2} = AXIAL^{end2} \tag{5.14}$$

$$M_3^{end2} = TORQUE^{end2} \tag{5.15}$$

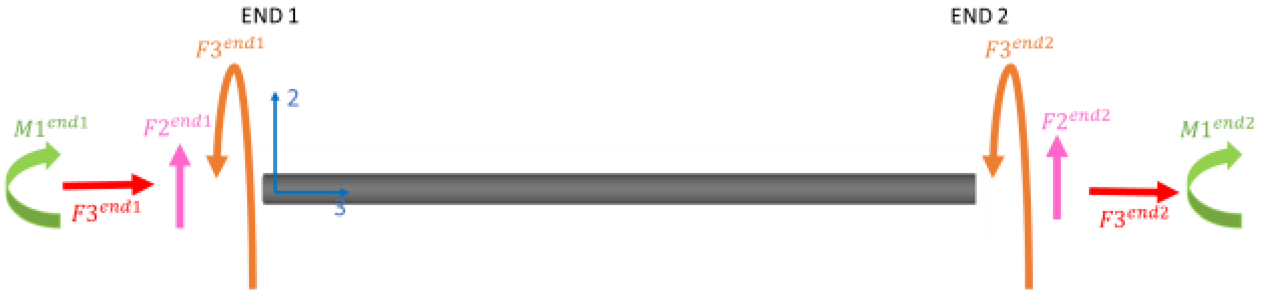


Figure 64. Beam node forces

The nodal forces at the end nodes of the i-th beam, shown in the Figure 64, can be written in the following vector:

$$f_i = \begin{pmatrix} F_{1,i}^{end1} \\ F_{2,i}^{end1} \\ F_{3,i}^{end1} \\ M_{1,i}^{end1} \\ M_{2,i}^{end1} \\ M_{3,i}^{end1} \\ F_{1,i}^{end2} \\ F_{2,i}^{end2} \\ F_{3,i}^{end2} \\ M_{1,i}^{end2} \\ M_{2,i}^{end2} \\ M_{3,i}^{end2} \end{pmatrix} \qquad \text{with i=1, ..., b} \tag{5.16}$$

The node forces vectors of all the beams in the model can be collected into a matrix [*b* x 12]:

$$F = \begin{bmatrix} f_1^T \\ \dots \\ f_b^T \end{bmatrix} \tag{5.17}$$

The relationship between the nodal forces acting on the end nodes of the i-th beam f_i and the beam forces generated by them s_i can be described using the following matrix:

$$T * s_i = f_i \tag{5.18}$$

$$T^{-1} * f_i = s_i \tag{5.19}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} SF_1^{end1} \\ BM_1^{end1} \\ SF_2^{end1} \\ BM_2^{end1} \\ AXIAL^{end1} \\ TORQUE^{end1} \\ SF_1^{end2} \\ BM_1^{end2} \\ SF_2^{end2} \\ BM_2^{end2} \\ AXIAL^{end2} \\ TORQUE^{end2} \end{pmatrix} = \begin{pmatrix} F_1^{end1} \\ F_2^{end1} \\ F_3^{end1} \\ M_1^{end1} \\ M_2^{end1} \\ M_3^{end1} \\ F_1^{end2} \\ F_2^{end2} \\ F_3^{end2} \\ M_1^{end2} \\ M_2^{end2} \\ M_3^{end2} \end{pmatrix} \tag{5.20}$$

The forces on the beams described above refer to the local reference system of the beams, which is specific to each of them. These local reference systems are defined for each beam by a tern of versors. These vectors can be collected in a matrix $\overline{M}_i$:

$$\overline{M}_i = [6\ x\ 6] = \begin{pmatrix} \overline{m}_i & 0 \\ 0 & \overline{m}_i \end{pmatrix} \qquad \text{with} \qquad \overline{m}_i = \begin{pmatrix} i_{1x} & i_{2x} & i_{3x} \\ i_{1y} & i_{2y} & i_{3y} \\ i_{1z} & i_{2z} & i_{3z} \end{pmatrix} \tag{5.21}$$

where i$_1$, i$_2$ and i$_3$ are the beam local axis. $\overline{M}_i$ matrix allows the conversion of forces and moments from the local to the global reference system and vice versa.

$$\overline{M}_i * F_{i,loc} = F_{i,glob} \tag{5.22}$$

where $F_{i,loc}$ and $F_{i,glob}$ are node forces in local reference system and global reference system, respectively.

These matrices can be used to convert the forces and moments expressed in the reference system of one beam into the reference system of a different beam. They therefore prove useful in convolution operations between nodes and beams and between beams.

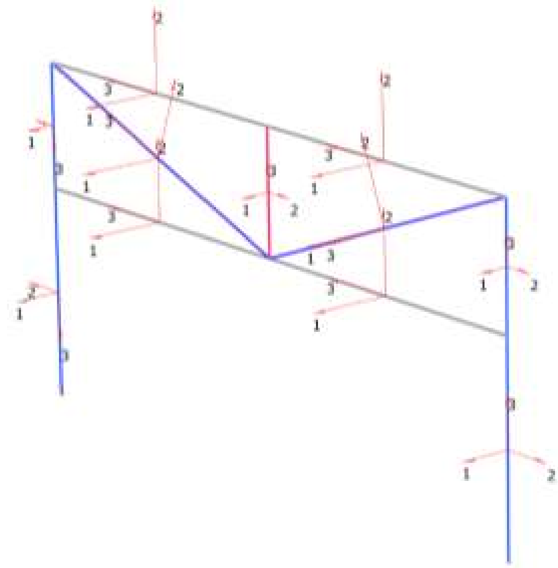


Figure 65. FEM model with beam local axis

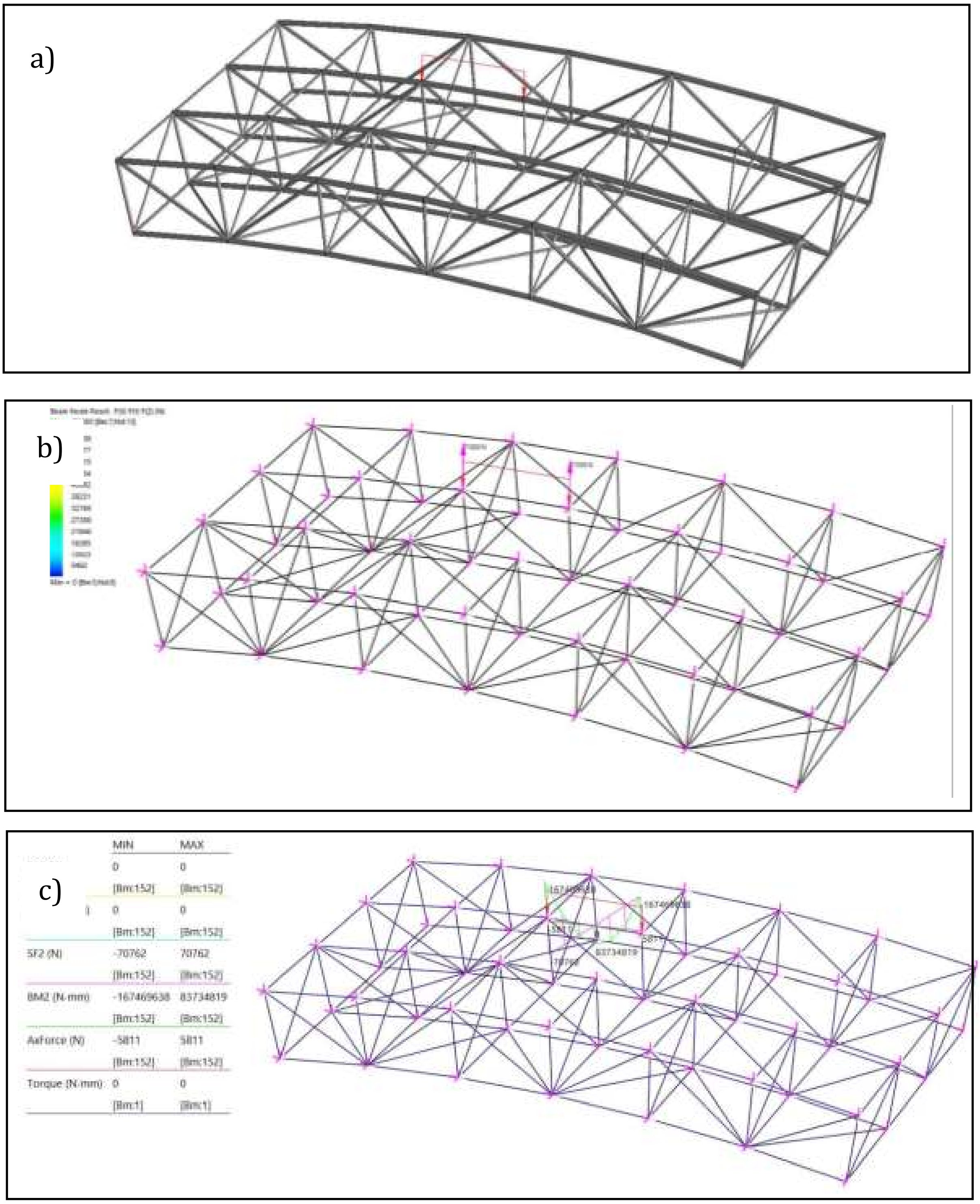
In a structural model, the ways in which loads are applied to the beams are different. For example, loads can be applied distributed along the development of the beam, or forces concentrated at its end, or indirectly through the introduction of two-dimensional loaded elements, or many others. The type of load applied influences how to describe it mathematically. For example, in the case of loads acting on nodes, a 6-entry vector can be used for each node:

$$c_i = \begin{pmatrix} F_{x,i} \\ F_{y,i} \\ F_{z,i} \\ M_{x,i} \\ M_{y,i} \\ M_{z,i} \end{pmatrix} \qquad \text{with i=1, ..., n} \qquad (5.23)$$

where *n* is the number of nodes, F_x, F_y and F_z are the translational forces in X, Y and Z directions while M_x, M_y and M_z are the moments around the X, Y and Z axis. Overall, the loads of a single condition for the global model can be described as a matrix [*n* x 6]:

$$C = \begin{bmatrix} c_1^T \\ \dots \\ c_n^T \end{bmatrix} \qquad (5.24)$$

If more than one type of load has been applied to the same model, it is necessary to identify a unique way to describe the load state that is valid for all load types. The idea is to express loads as a set of force vectors at the nodes and beam forces vectors on the beams. The vectors at the nodes represent the loads applied directly to the nodes or, in the case where the load is applied along the beams, the forces and moments transmitted from the loaded beam to the neighborhood. Beam forces vectors, on the other hand, allow us to describe the effect that a load distributed on a beam has on the beam itself. Both node and beam vectors are derived by solving the model in which fixed constraints have been applied at each node.



	MIN	MAX
	0	0
	[Bm:152]	[Bm:152]
	0	0
	[Bm:152]	[Bm:152]
SF2 (N)	-70762	70762
	[Bm:152]	[Bm:152]
BM2 (N-mm)	-167469638	83734819
	[Bm:152]	[Bm:152]
Axforce (N)	-5811	5811
	[Bm:152]	[Bm:152]
Torque (N-mm)	0	0
	[Bm:1]	[Bm:1]

Figure 66. Fixed constraints are applied to the starting model (a) at the nodes, and from the solution of this model, forces at the nodes are derived as reactions (b) and forces on the beams are obtained (c)

5.2.2 Case studies

The different approaches analyzed for constructing the surrogate model were tested on two different models: the two-dimensional frame in Figure 67 and the spatial frame in Figure 68.

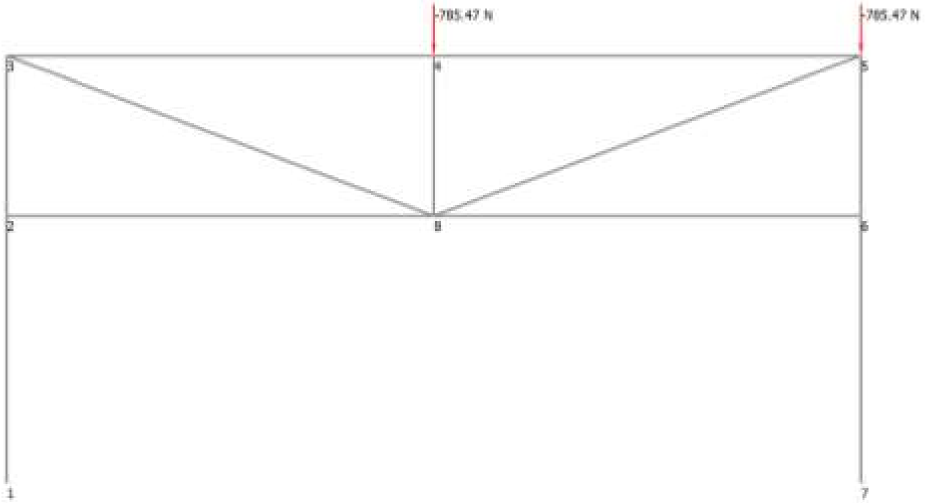


Figure 67. Two-dimensional frame used as a case of study

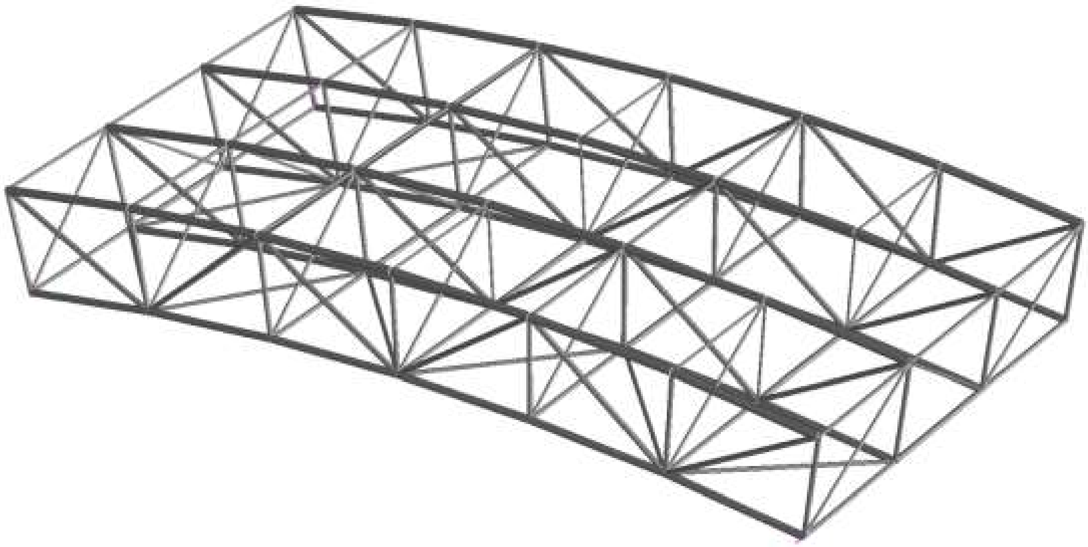


Figure 68. Spatial frame used as a case of study

For both models, 80% of the load cases were used to create the surrogate model while the remaining 20% were used as validation tests. The error is calculated as the Mean Absolute Error (MAE) between the components of the forces at the beam ends predicted with the surrogate model and those calculated with the FEM.

The two-dimensional frame in Figure 67 consists of a few elements, 8 nodes and 11 beams. The simplicity of the model made it possible to generate 1000 random load cases and thus explore well the possible load condition cases without computational cost issues. Loads were applied as nodal forces and/or moments. The lack of stiffness in the third dimension led to loading this model only with forces or moments causing displacements in the plane of the frame. Consequently, for each node the applied load will be defined with a 6-entry vector (3 forces and 3 moments) of which only 3 can be non-zero.

The frame in Figure 68, compared with the previous case, represents a condition closer to a real situation that a designer may face. The structure consists of 56 nodes and 167 beams arranged spatially. The method was tested by considering both the structure subjected to 1000 load conditions applied as forces and moments at the nodes and 1000 load conditions as distributed forces on the beams.

5.2.3 *LoadsToForces* module creation approaches

The approaches analyzed for creating the *LoadsToForces* module are presented below.

5.2.3.1 Low-rank stiffness matrix

The first approach analyzed to update the forces on the beams as the loads change without the use of FEM involves estimating the stiffness matrix from the known FEM solution. Having used a linear-type solver to calculate the forces on the beams from the loads, the goal is to find a linear link that relates the loads to the tensional state of the beams. This function can then be used to estimate changes in the forces on the beams due to changes in the loads.

The calculation of the stiffness matrix requires the construction of a matrix of loads $\bar{C}$ and a matrix of forces on beams $\bar{S}$ containing information on known n_{LC} load cases. In the case where loads are applied to the model as forces and moments concentrated at the nodes, the load matrix can be written as follows:

$$\bar{C} = [n_{LC}\ x\ 6n] = \begin{bmatrix} c_{1,1}^T & \dots & c_{1,n}^T \\ \dots & \dots & \dots \\ c_{n_{LC},1}^T & \dots & c_{n_{LC},n}^T \end{bmatrix} \tag{5.25}$$

where $c_{i,j}$is the vector of loads [6 x 1] applied to node j in i-th load case. If, on the other hand, the load is applied as uniformly distributed on the beams, $\bar{C}$ became a matrix having $3b$ columns representing the value of the load on the beams in the 3 directions:

$$\bar{C} = [n_{LC}\ x\ 3b] = \begin{bmatrix} c_{1,1}^T & \dots & c_{1,b}^T \\ \dots & \dots & \dots \\ c_{n_{LC},1}^T & \dots & c_{n_{LC},b}^T \end{bmatrix} \tag{5.26}$$

where in this case $c_{i,j}$denote the load vector [3 x 1] at j-th beam in load case i.

If, on the other hand, loads are described using the unique approach presented in section 5.2 , the load matrix becomes as follows:

$$\bar{C} = [n_{LC}\ x\ (6n + 12b)] = \begin{bmatrix} c_{1,1}^T & \dots & c_{1,n+b}^T \\ \dots & \dots & \dots \\ c_{n_{LC},1}^T & \dots & c_{n_{LC},n+b}^T \end{bmatrix} \tag{5.27}$$

with $c_{i,j}$ is the load vector [6 x 1] of the j-th node if $1 \leq j \leq n$, otherwise for $n < j \leq n + b$ is the load vector [12 x 1] of the end of beam $j - n$, in i-th load case.

The matrix with information on the forces on the beams for the various load cases is as follows:

$$\bar{S} = [n_{LC}\ x\ 12b] = \begin{bmatrix} s_{1,1}^T & \dots & s_{1,b}^T \\ \dots & \dots & \dots \\ s_{n_{LC},b}^T & \dots & s_{n_{LC},b}^T \end{bmatrix} \tag{5.28}$$

where $s_{i,j}$ is the vector of forces at ends of beam j in the load case i, as defined in the previous paragraph.

Through decomposition to singular values, the pseudoinverse of the matrix of loads $\bar{C}^+$ is calculated. Therefore, the low-rank stiffness matrix can be calculated as follows:

$$\bar{R} = [12b\ x\ 6n] = \bar{S}^T * \bar{C}^+ \tag{5.29}$$

Results

The performance offered by this approach was analyzed on the models in Figure 67 and in Figure 68. For both models, 80% of the load cases were used to calculate the low-rank stiffness matrix while the remaining 20% were used as validation tests. The error is calculated as the *Mean Absolute Error* (MAE) between the components of the forces at the beam ends predicted with the stiffness matrix and those calculated with the FEM.

The low-rank stiffness matrix for the two-dimensional model leads to the errors in the training set and validation set shown in Table 31. Since no distinction has been made between forces and moments in calculating the error, its unit can be N or N*mm: in either case the error can be considered negligible. As an example, the results provided by the FEM and calculated with the low-rank stiffness matrix for the case in Figure 69, which belongs to neither the training set nor the validation set and in which two forces were applied concentrated at nodes 4 and 5, are shown in Table 32.

	Training set	**Validation set**
MAE	2.2194	1.8872

Table 31. Error of the low-ran matrix for model in Figure 67

BEAM	**END**	**Low-rank stiffness matrix**						**FEM**						**Absolute error**					
		SF1 [N]	**BM1 [N*mm]**	**SF2 [N]**	**BM2 [N*mm]**	**AX [N]**	**TQ [N*mm]**	**SF1 [N]**	**BM1 [N*mm]**	**SF2 [N]**	**BM2 [N*mm]**	**AX [N]**	**TQ [N*mm]**	**SF1 [N]**	**BM1 [N*mm]**	**SF2 [N]**	**BM2 [N*mm]**	**AX [N]**	**TQ [N*mm]**
1	1	0.00	0.00	-20.22	16149.54	-392.76	0.00	0.00	0.00	-20.23	16160.70	-392.99	0.00	0.00	0.00	0.01	11.16	0.23	0.00
	2	0.00	0.00	-20.22	-34402.90	-392.76	0.00	0.00	0.00	-20.23	-34424.50	-392.99	0.00	0.00	0.00	0.01	21.60	0.23	0.00
2	1	0.00	0.00	-48.03	19619.34	-363.27	0.00	0.00	0.00	-48.06	19630.50	-363.48	0.00	0.00	0.00	0.03	11.16	0.21	0.00
	2	0.00	0.00	-48.03	-52430.74	-363.27	0.00	0.00	0.00	-48.06	-52462.00	-363.48	0.00	0.00	0.00	0.03	31.26	0.21	0.00
3	1	0.00	0.00	21.00	-31031.24	-918.34	0.00	0.00	0.00	21.01	-31049.80	-918.89	0.00	0.00	0.00	0.01	18.56	0.55	0.00
	2	0.00	0.00	21.00	52958.96	-918.34	0.00	0.00	0.00	21.01	52990.60	-918.89	0.00	0.00	0.00	0.01	31.64	0.55	0.00
4	1	0.00	0.00	-21.22	53297.42	-917.36	0.00	0.00	0.00	-21.23	53329.20	-917.91	0.00	0.00	0.00	0.01	31.78	0.55	0.00
	2	0.00	0.00	-21.22	-31563.43	-917.36	0.00	0.00	0.00	-21.23	-31582.30	-917.91	0.00	0.00	0.00	0.01	18.87	0.55	0.00
5	1	0.00	0.00	50.77	-52918.90	-1146.88	0.00	0.00	0.00	50.80	-52950.60	-1147.57	0.00	0.00	0.00	0.03	31.70	0.69	0.00
	2	0.00	0.00	50.77	23234.96	-1146.88	0.00	0.00	0.00	50.80	23249.50	-1147.57	0.00	0.00	0.00	0.03	14.54	0.69	0.00
6	1	0.00	0.00	20.22	-32332.88	-1177.24	0.00	0.00	0.00	20.23	-32351.20	-1177.95	0.00	0.00	0.00	0.01	18.32	0.71	0.00
	2	0.00	0.00	20.22	18224.35	-1177.24	0.00	0.00	0.00	20.23	18234.00	-1177.95	0.00	0.00	0.00	0.01	9.65	0.71	0.00
7	1	0.00	0.00	29.49	-54022.33	27.81	0.00	0.00	0.00	29.51	-54055.00	27.83	0.00	0.00	0.00	0.02	32.67	0.01	0.00
	2	0.00	0.00	29.49	63945.65	27.81	0.00	0.00	0.00	29.51	63984.00	27.83	0.00	0.00	0.00	0.02	38.35	0.01	0.00
8	1	0.00	0.00	-30.36	65870.63	30.55	0.00	0.00	0.00	-30.38	65909.80	30.57	0.00	0.00	0.00	0.02	39.17	0.02	0.00
	2	0.00	0.00	-30.36	-55567.94	30.55	0.00	0.00	0.00	-30.38	-55600.80	30.57	0.00	0.00	0.00	0.02	32.86	0.02	0.00
9	1	0.00	0.00	0.98	-1138.07	-742.79	0.00	0.00	0.00	0.98	-1138.55	-743.23	0.00	0.00	0.00	0.00	0.48	0.45	0.00
	2	0.00	0.00	0.98	338.48	-742.79	0.00	0.00	0.00	0.98	338.61	-743.23	0.00	0.00	0.00	0.00	0.13	0.45	0.00
10	1	0.00	0.00	14.89	-21399.52	935.08	0.00	0.00	0.00	14.90	-21412.30	935.64	0.00	0.00	0.00	0.01	12.78	0.56	0.00
	2	0.00	0.00	14.89	42211.84	935.08	0.00	0.00	0.00	14.90	42237.00	935.64	0.00	0.00	0.00	0.01	25.16	0.56	0.00
11	1	0.00	0.00	-14.70	41424.91	931.03	0.00	0.00	0.00	-14.70	41449.70	931.59	0.00	0.00	0.00	0.01	24.79	0.55	0.00
	2	0.00	0.00	-14.70	-21355.49	931.03	0.00	0.00	0.00	-14.70	-21368.30	931.59	0.00	0.00	0.00	0.01	12.81	0.55	0.00

Table 32. Result of comparing the result of low-rank stiffness matrix and that given by FEM for the case in Figure 69

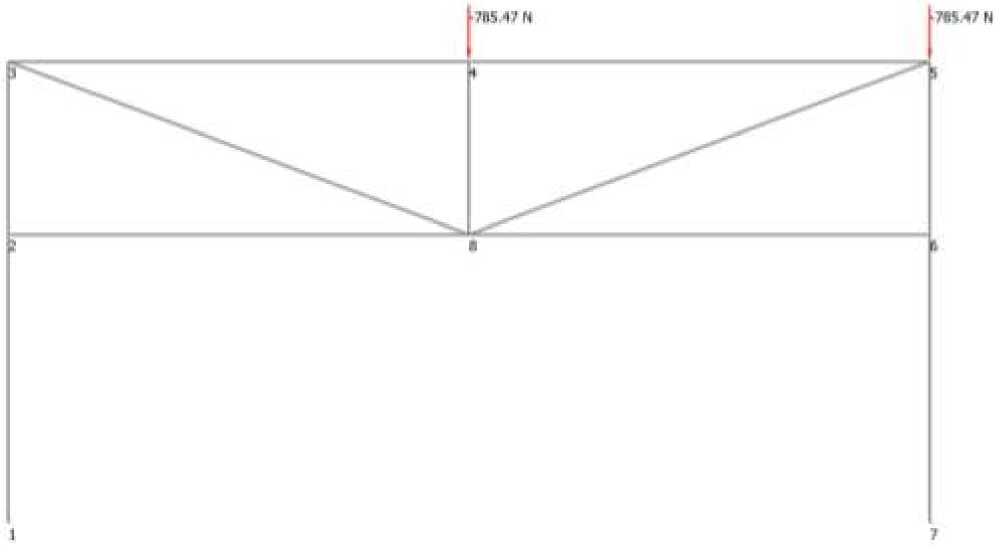


Figure 69. Two-dimensional frame loading conditions in which the low-rank stiffness matrix result was compared with that given by the FEM

For the spatial frame loaded with forces at the nodes, the stiffness matrix was derived by describing the loads as only forces at the nodes. The errors on the training set and validation set are given in Table 33 for different *rcond*, corresponding to the tolerance values of the singular values used in the calculation of the pseudoinverse of the load matrix.

	MAE	
rcond	**TRAINING**	**TEST**
10^(-15)	25301.53	30439.57
10^(-11)	25301.53	30439.57
10^(-10)	754.2961	846.31
10^(-9)	754.2961	846.31
10^(-8)	503.3905	579.3168
10^(-7)	449.7696	519.4872
10^(-6)	446.1117	518.8887

Table 33. Error of the low-rank matrix for model in Figure 68 with 1000 load cases in which the loads are applied as concentrated forces and moments at nodes

In the case where the spatial model is loaded with distributed loads, the calculation of the low-rank stiffness matrix was tried using two types of load matrices. In one case, the load matrix was built by assigning to each beam 3 parameters describing the value of the distributed load acting on them in the three directions of the reference system. The use of this load matrix led to results reported in Table 34. In the other case, the unique approach was used to describe the loads, and the results are shown in Table 35.

The two types of load matrices led to the definition of a low-rank stiffness matrix with acceptable error values. By using the distributed load parameters directly, the error is lower. However, the unique approach has also led to acceptable errors and offers the advantage that it can be applied whatever the load condition.

	MAE	
rcond	**TRAINING**	**TEST**
10^(-15)	34.9309	46.7875
10^(-7)	34.9309	46.7875
10^(-5)	34.9309	46.7875

Table 34. Error of the low-rank matrix for model in Figure 68 with 1000 load cases applied as beams distributed loads. The loads were described with 3 parameters per beam representative of the value of the loads distributed in the 3 directions of the reference system.

	MAE	
rcond	**TRAINING**	**TEST**
10^(-15)	32366736.0	31352478.0
10^(-7)	438892.8	421826.2
10^(-6)	3848.0	7760.8
10^(-5)	3848.0	7760.8
10^(-4)	870459.4	2283232.8

Table 35. Error of the low-rank matrix for model in Figure 68 with 1000 load cases applied as beams distributed loads. The loads were described using the unique approach.

5.2.3.2 MLP model

The second approach to creating surrogate model involves the development of an MLP-type Neural Network that takes input loads at the nodes and returns output forces at the ends of the beams. The input data for the MLP is the load matrix, having for each row the information about a single load case. The number of columns in the matrix, however, depends on the mode chosen for describing the loads. The output matrix, on the other hand, contains information on the forces at the ends of the beams for each loading condition and has size [n_{LC} x 12b], where b is the number of beams and n_{LC} is the number of load cases.

Different network architectures were analyzed. In the case where only one hidden layer of linear type is used, the relationship described by the MLP model between nodal loads and forces on the beams is of linear type, such as that sought through using the low-rank stiffness matrix. Comparison with architectures having multiple intermediate layers of linear type interspersed with nonlinear layers allows the linearity of the problem to be evaluated.

The use of the MLP network was tested with the two-dimensional frame. The load was described as vectors of forces and moments at the nodes, and thus the input matrix is $[n_{LC} \text{ x } 6n]$. Figure 70 and Figure 71 show the results for two networks having 48 intermediate nodes, which corresponds to the number of variables used to describe a model load condition ($6n$). The two networks differ in the presence or absence of nonlinearities in the inner layers. It is observed that the introduction of nonlinearities increases the computational burden without benefiting the network in learning. This is in agreement with the fact that the tensional state on the beams was calculated by a linear type analysis performed on a FEM model. The errors obtained with the MLP method are acceptable but higher than those obtained with the low-rank stiffness matrix. In addition, training the network requires much greater computational resources than those required for estimating the low-rank stiffness matrix. For these reasons, this approach was discarded.

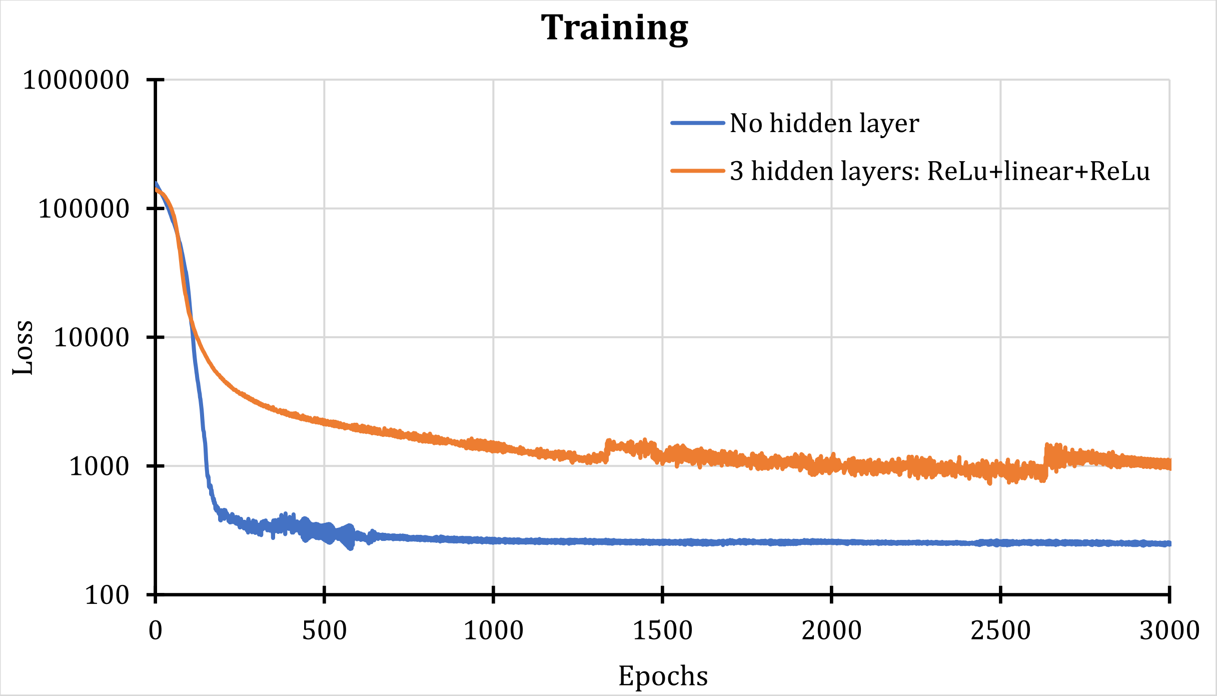


Figure 70. Loss trend during MLP training in the bidimensional frame

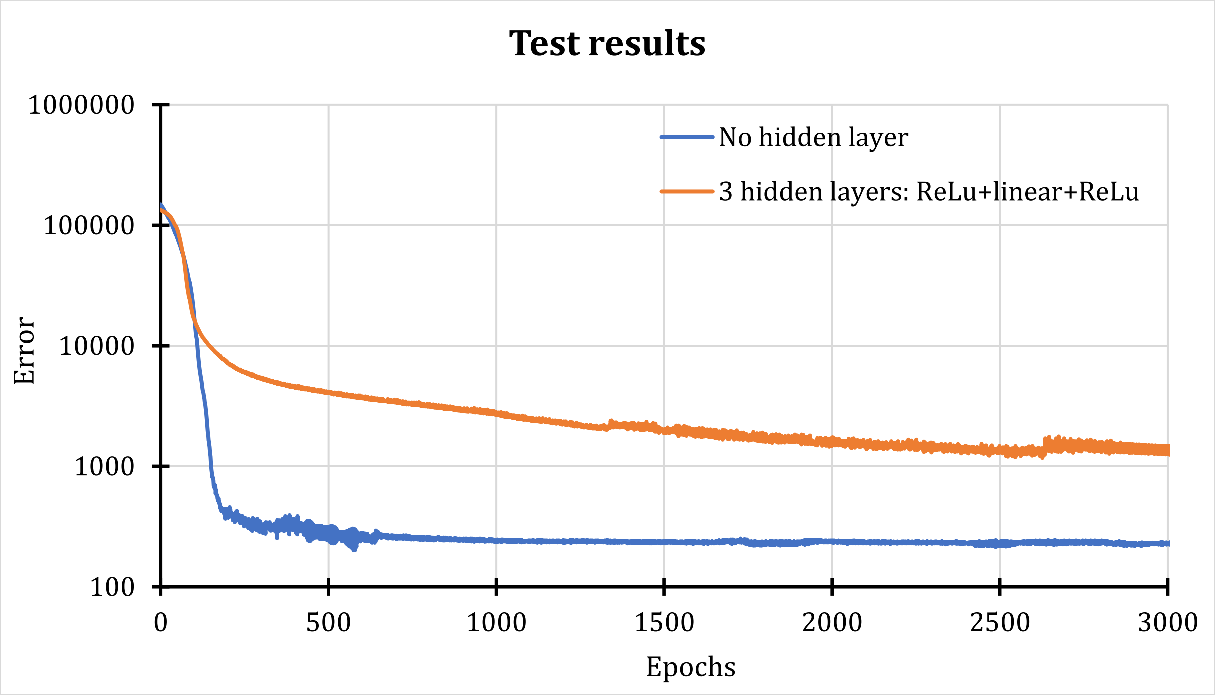


Figure 71. Error trend of MLP in validation set in the bidimensional frame

5.2.3.3 GNN model

Frame models lend themselves well to being represented with a graph structure, in which nodes indicate structural entities while edges represent the relationships between them. This makes it possible to evaluate the construction of a GNN-type model, i.e., a network in which input and output quantities are attributes associated with the nodes in the network, and which also exploits the relationships existing between the nodes for the construction of the map between inputs and outputs.

The model to be built represents a relationship between the loads on the FEM nodes and the forces on the beams. We consider the case where the loads are applied as concentrated forces and moments at the FEM nodes. Therefore, the load state of the entire model in a load case can be described as a vector of dimension $6n$, where n is the number of FEM nodes in the entire model. The output of the network, on the other hand, represents the forces at the ends of the beams and thus can be written as a vector of dimension $12b$, where b is the number of beams in the model.

Two possible approaches were evaluated: construction of a single model based on a heterogeneous graph in which the nodes are both FEM nodes and beams, and construction of two models based on two separate graphs, one for FEM nodes and one for beams, to compute the encoding matrix and the decoding matrix.

5.2.3.4 GNN model with heterogeneous graph

The neural network for mapping nodal forces to stresses was named *GNNLoadsToForces*. Different architectures for building this network have been analyzed. This study was conducted on the two-dimensional frame in Figure 67.

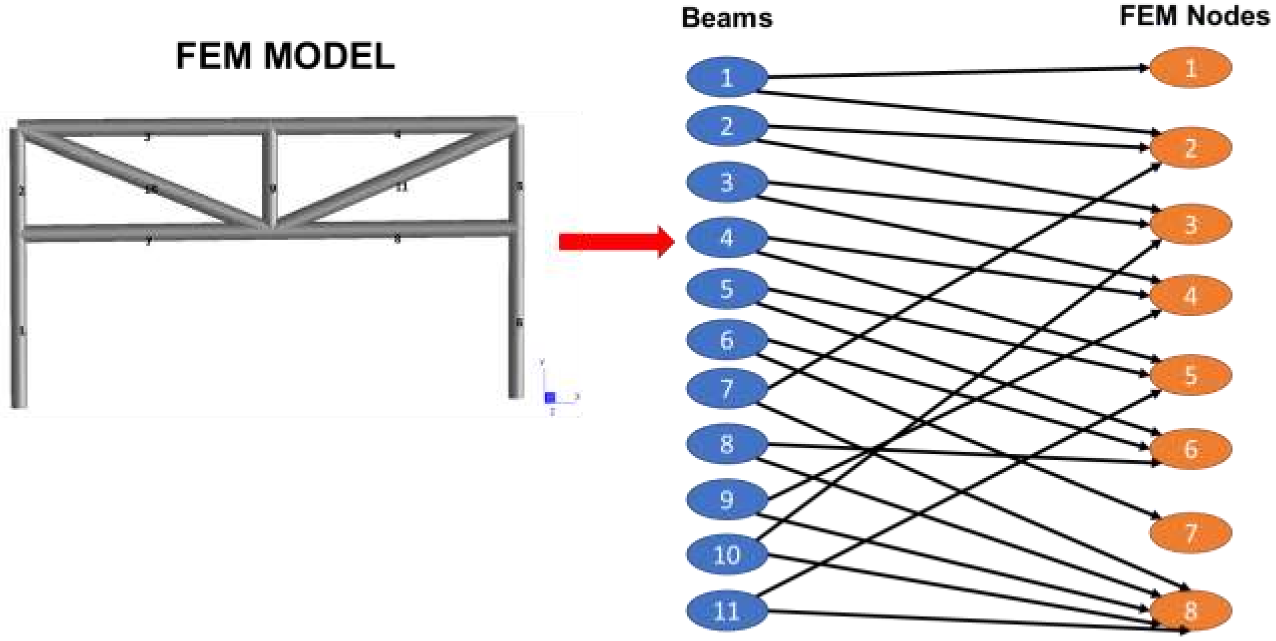


Figure 72. Heterogeneous graph for model in Figure 67

The network is constructed based on the heterogeneous graph in which there are two categories of nodes:

- FEM nodes, to which loads are assigned
- beams, to which stresses are associated at various points along their length.

To facilitate network learning, input and output data were scaled and normalized by type.

The information contained in each FEM node is transmitted to the beams terminating at that node. The first layer of the network is thus a convolution that converts the loads from the FEM nodes to the beams connected to them. The second layer is a convolution between beams neighboring. Two possible configurations were evaluated for subsequent layers: an alternation of linear and nonlinear layers or an alternation of convolutional layers with nonlinear layers. The nonlinear layers are *LeakyReLU* functions (Figure 73), which are characterized by two linear segments with different slopes between positive and negative values. Thus, compared with the *ReLU* function, cancellation of negative input values does not occur.

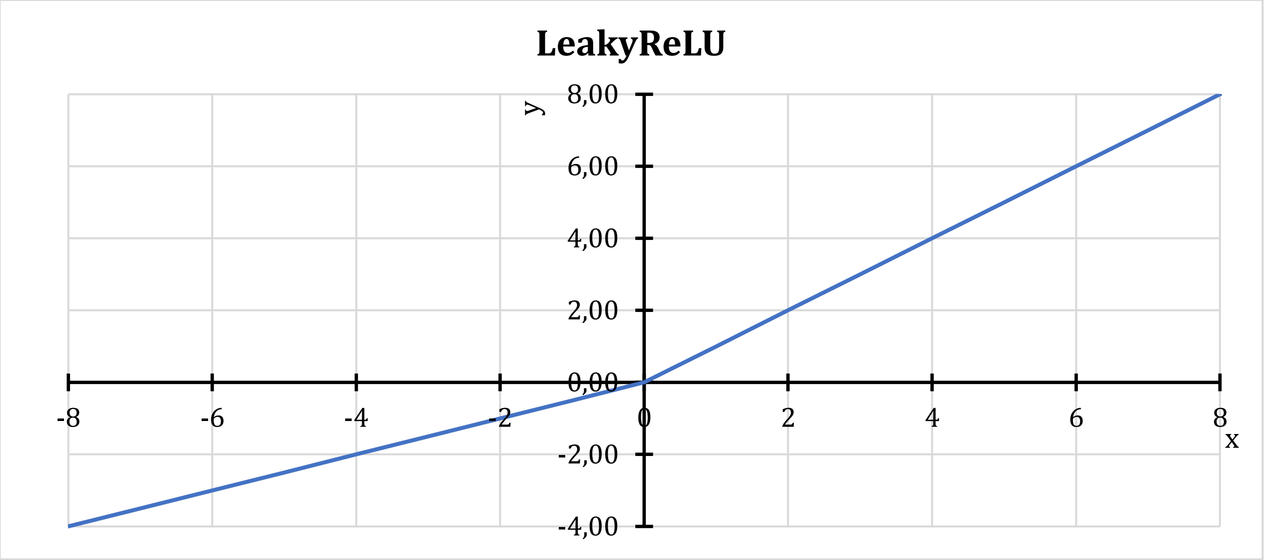


Figure 73. LeakyReLU function

Two possible functions were considered as loss: *Mean Absolute Error* (MAE) and *Mean Square Error* (MSE). From the graph in Figure 74, a better performance of network learning and generalization of results is observed using MAE as a loss function than MSE. Therefore, it was chosen to use MAE as the function to lead the network learning. This function contains within it the differences between the forces and moments calculated using the network and the exact ones. Since forces and moments are two quantities of different types, it was considered to introduce a scaling factor to be applied to the estimated and exact moments to make the error of moments comparable with that of forces for loss calculation. An 80% reduction was chosen to be applied to the moments. The effects of this scaling can be seen in Figure 75: a reduction in MAE (calculated in both cases without scaling) is observed in both training and testing when the network is trained with the loss function containing the reductive factor for moments.

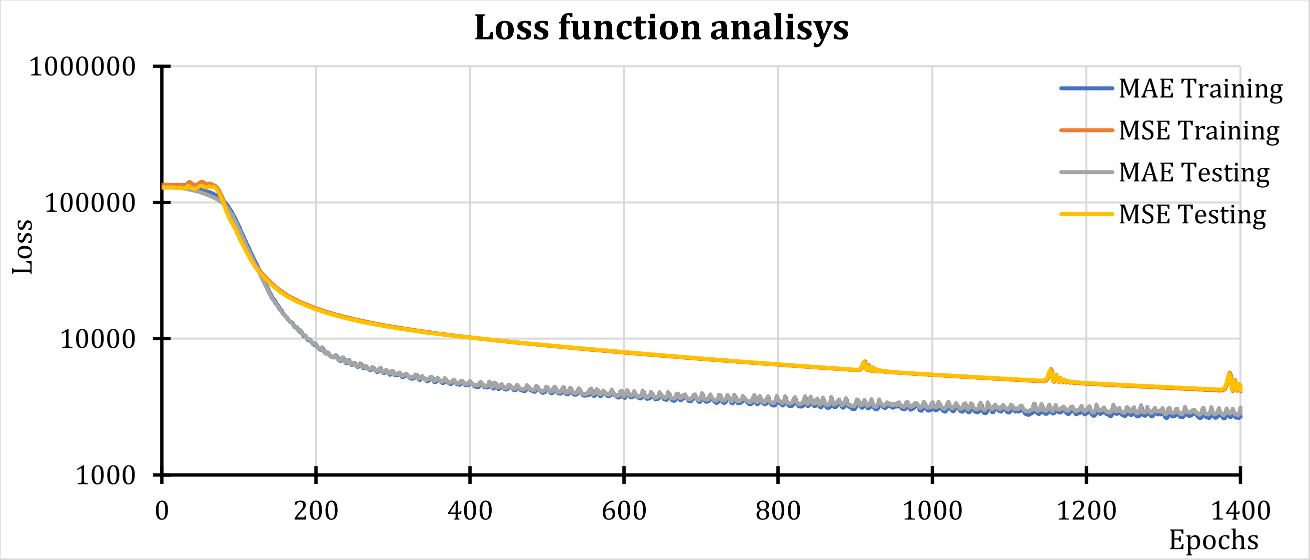


Figure 74. Analysis of MAE and MSE as loss function in training and test for the bidimensional frame

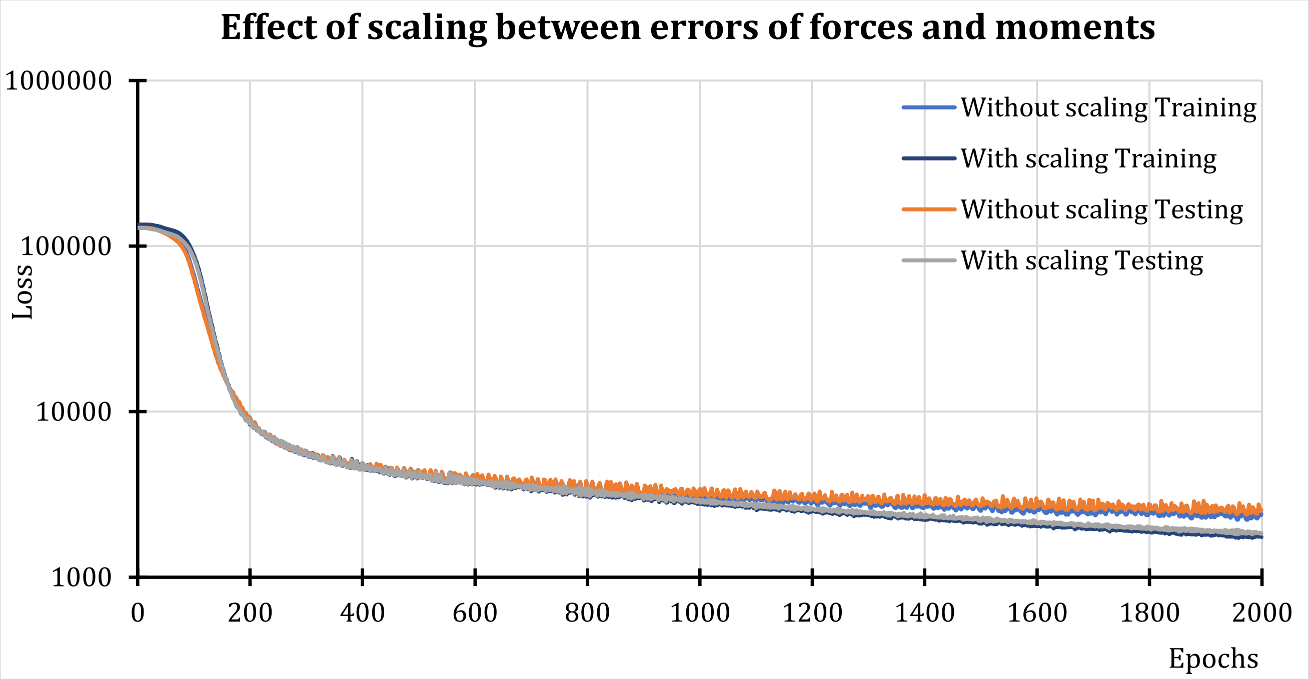


Figure 75. Effect of scaling between errors of forces and moments using MAE as loss function in two-dimensional frame

The effects of the type of intermediate layers used in the network architecture can be seen in Figure 76. From the graph, greater difficulties are observed in the case where convolutions have not been included in the inner layers.

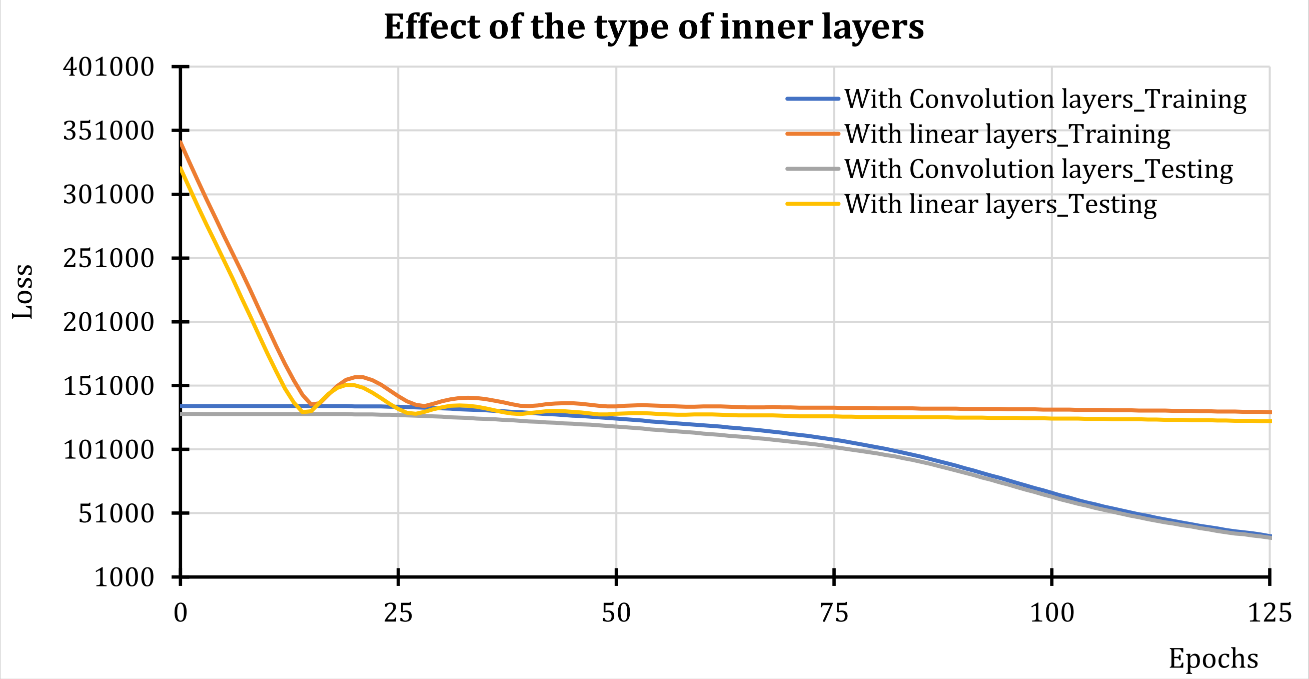


Figure 76. Comparison of network composed of convolutional layers and nonlinear layers and composed of linear and nonlinear layers. In both cases, the hidden layers have dimension k=32 and there are 5 packets of inner layers following the initial convolution consisting of convolutive/linear layer + nonlinear layer. The results refer to the two-dimensional frame.

Nodes-beams convolution

The first layer of the network describes a convolution in which the message is transmitted from the FEM nodes to the beams converging on it. For each loading condition, the input of the layer is the matrix X [$6n$ x 1], which represents the load applied to the structure in terms of concentrated forces and moments acting on the n FEM nodes. Instead, the output of the layer is a matrix Y of size [$12b$ x k], where b is the number of beams while k is the dimension of the subspace used for the problem description. The convolution is based on an adjacency matrix [$12b$ x $6n$] that expresses for each beam what the end nodes are and also allows the transformation of forces from the global to the local reference system:

$$Adj = [12b\ x\ 6n] = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \cdots & \cdots & \cdots \\ a_{b1} & \cdots & a_{bn} \end{pmatrix} \tag{5.30}$$

$$a_{ij} = [12\ x\ 6] = \begin{pmatrix} a_{ij,1} \\ a_{ij,2} \end{pmatrix} \tag{5.31}$$

$$a_{ij,1} = [6\ x\ 6] = \begin{cases} 0_{6x6} & if\ node\ j\ isn't\ the\ node\ at\ end\ 1\ of\ beam\ i \\ I_{2x2} \otimes \overline{m}_i^{-1} & if\ node\ j\ is\ the\ node\ at\ end\ 1\ of\ beam\ i \end{cases} \tag{5.32}$$

$$a_{ij,2} = [6\ x\ 6] = \begin{cases} 0_{6x6} & if\ node\ j\ isn't\ the\ node\ at\ end\ 2\ of\ beam\ i \\ I_{2x2} \otimes \overline{m}_i^{-1} & if\ node\ j\ is\ the\ node\ at\ end\ 2\ of\ beam\ i \end{cases} \tag{5.33}$$

where $\overline{m}_i$ is the [3 x 3] matrix to convert forces and moments from local reference system of beam i to global reference system.

The parameters to be learned during network training are contained in mask matrix Q [12 x 6] and weight matrix W [1 x k]. The nodal forces and moments that result from the convolution between nodes and beams are converted to beam forces through the T matrix defined in the section 5.2.1 . Overall, the convolutional layer can be expressed in mathematical terms as follows:

$$Y = [12b\ x\ k] = (I_{b\ x\ b} \otimes T^{-1}) * ((I_{b\ x\ b} \otimes Q)Adj * X * W) \tag{5.34}$$

Beams-beams convolution

In these convolutional layers, each beam transmits the message to other beams belonging to their neighborhood. The transmission of forces requires that they undergo a change of reference system between a starting beam and the finishing beam. It can be carried out using the matrix M_i, which describes the conversion between local reference systems of beam *i* and global reference systems. Passing through the global reference system, it is possible to express the components of forces and moments transmitted from beam *j*, and thus initially expressed in its reference system, to beam *i* in the latter's reference system.

$$(\bar{M}_i^{-1}\bar{M}_j)f_j = -f_{i\_j} \tag{5.35}$$

$$\bar{M}_{ji} = \bar{M}_i^{-1}\bar{M}_j \tag{5.36}$$

$$\bar{M}_{ji} * f_j = -f_{i\_j} \tag{5.37}$$

where $\bar{M}_i$ and $\bar{M}_j$ are respectively the reference system conversion matrices of beam *i* and beam *j*, while f_j represents the forces and moments transmitted from beam *j* to beam *i* expressed in the local reference system of the beam *j*. In the equation (5.37), the matrix $\bar{M}_{ji}$ allows conversion from the local reference system of the *j*-th beam to that of the *i*-th beam while f_{i_j} is the vector of forces and moments transmitted from the *j*-th beam to the *i*-th beam in the local reference system of beam *i*.

By denoting by s_j the stresses on the *j*-th beam and using expressions (5.18) and (5.19) for the conversions between the forces at the nodes and the beam forces, it is possible to express in mathematical terms the stresses s_{i_j} transmitted from beam *j* to beam *i*:

$$f_{i\_j} = -\left(M_i^{-1}M_j\right)f_j = -\left(M_i^{-1}M_j\right)(T * s_j) = T * s_{i\_j} \tag{5.38}$$

$$s_{i\_j} = -T^{-1}\left(M_i^{-1}M_j\right)\left(T * s_j\right) = H * s_j \tag{5.39}$$

$$H = -T^{-1}\left(M_i^{-1}M_j\right)T \tag{5.40}$$

These remarks are used in the definition of the sparse matrix transformation graph E [12b x 12b]:

$$E = \begin{pmatrix} e_{11} & \cdots & e_{1b} \\ \cdots & \cdots & \cdots \\ e_{b1} & \cdots & e_{bb} \end{pmatrix} \tag{5.41}$$

where e_{ij} are [12 x 12] matrix defining stress transmission between beam i and beam j. The correspondences between the entries [k, q] of the E matrix and the components of the beam forces are shown in the Table 36.

Row/column entries of e$_{ij}$ matrix	**Beam force**
0	SF_1^{end1}
1	BM_1^{end1}
2	SF_2^{end1}
3	BM_2^{end1}
4	$AXIAL^{end1}$
5	$TORQUE^{end1}$
6	SF_1^{end2}
7	BM_1^{end2}
8	SF_2^{end2}
9	BM_2^{end2}
10	$AXIAL^{end2}$
11	$TORQUE^{end2}$

Table 36. Entries of e_{ij} matrix

The definition of the submatrices e_{ij} is given in the equation (5.42).

$$e_{ij} = [12\,x\,12] = \begin{cases} \begin{pmatrix} T^{-1} * M_i^{-1} * M_j * T & 0_{6x6} \\ 0_{6x6} & 0_{6x6} \end{pmatrix} & from\ end\ 1\ of\ beam\ j\ to\ end\ 1\ of\ beam\ i \\ \begin{pmatrix} 0_{6x6} & 0_{6x6} \\ T^{-1} * M_i^{-1} * M_j * T & 0_{6x6} \end{pmatrix} & from\ end\ 1\ of\ beam\ j\ to\ end\ 2\ of\ beam\ i \\ \begin{pmatrix} 0_{6x6} & T^{-1} * M_i^{-1} * M_j * T \\ 0_{6x6} & 0_{6x6} \end{pmatrix} & from\ end\ 2\ of\ beam\ j\ to\ end\ 1\ of\ beam\ i \\ \begin{pmatrix} 0_{6x6} & 0_{6x6} \\ 0_{6x6} & T^{-1} * M_i^{-1} * M_j * T \end{pmatrix} & from\ end\ 2\ of\ beam\ j\ to\ end\ 2\ of\ beam\ i \end{cases} \tag{5.42}$$

The output X_{ij} from node *j* and directed to node *i* should be converted from the local reference system of node *j*-th to that of node *i*-th. The transformation matrix between these two reference systems is denoted by R_{ij}. The result from change of reference system is multiplied on the left by the Q matrix and on the right by the W matrix. While the latter is a matrix of weights of size [k x k], the Q matrix [12 x 12] represents the mask to be learned. Both the W matrix and the Q matrix are computed by GNN during training. Thus, the message M_{ij} transmitted between the two nodes is as follows:

$$M_{ij} = QR_{ij}X_{ij}W \tag{5.43}$$

Taking advantage of the property of Kronecker's product such that:

$$(M \otimes N)(C \otimes D) = (MC) \otimes (ND) \tag{5.44}$$

by setting M=$I_{b \text{ x } b}$, N=Q, C=Adj and D=R, we obtain:

$$(I_{b\,x\,b} \otimes Q)(Adj \otimes R) = Adj \otimes (QR) \tag{5.45}$$

The sum of QR_{ij} from all nodes in the neighborhood can be calculated as $(I \otimes Q)E$. The message transmitted from the whole neighborhood to the *i*-th node at layer *h* can be written in matrix form:

$$M^h_{neighborhood} = [12b\ x\ k] = (I_{b\,x\,b} \otimes Q)EXW \tag{5.46}$$

where we denote by *b* the number of beams. To this must be added the contribution of self-loops, for which the transformation matrix coincides with the identity matrix:

$$M^h_{self-loop} = [12b\ x\ k] = (I_{b\,x\,b} \otimes Q_{sl})XW_{sl} \tag{5.47}$$

where Q_{sl} and W_{sl}, respectively of dimensions [12 x 12] and [k x k], are the matrices of weights specific to self-loops.

The output of the h layer is then the sum of the messages coming from the neighborhood with those coming from the nodes themselves:

$$Y^h = [12b\ x\ k] = M^h_{neighborhood} + M^h_{self-loop} = (I_{b\ x\ b} \otimes Q)EXW - (I_{b\ x\ b} \otimes Q_{sl})XW_{sl} \qquad (5.48)$$

5.2.3.5 GNN model with twin approach

The mapping from load space to stress space requires the definition of an encoding matrix A describing the relationship between load space and space of size k and a decoding matrix B representing the function from space of size k to beam forces space.

Since the loads are associated with each node of the FEM model while the beam forces are related to the beams, the two matrices are derived on two separate graphs, one built on the FEM nodes (*NodeGraph*) while the other is built on the beams (*BeamGraph*). The matrices are calculated by training two GNNs in parallel, *NodeGNN* and *BeamGNN*, which are built on the two separate graphs and which use the same loss function. For i-th result case:

$$C_i * A = H_i \qquad (5.49)$$

$$S_i * B = F_i \qquad (5.50)$$

where C_i and S_i are the loads matrix and the beam forces matrix, respectively. The loss function for training the two graphs is as follows:

$$loss = MAE(B * A * C_i - S_i) \qquad (5.51)$$

Similar to the heterogeneous graph-based network in paragraph 5.2.3.4 , both GNNs are composed of a succession of convolution layers, in which contributions from the neighborhood are added together, and non-linear *LeakyReLU*-type layers. The transmission of messages between graph nodes is controlled by a transformation matrix E, which is the result of combining the adjacency matrix Adj and the transformation matrix R between local element reference systems.

NodeGNN

NodeGNN is the network used to calculate the encoding matrix. It is built on the graph *NodeGraph* with the FEM nodes of the structural calculation model as nodes. Edges connect nodes for which there is a beam having the two nodes as ends. Figure 77 and Figure 78 show, respectively, the graphs for the frames in Figure 67 and in Figure 68.

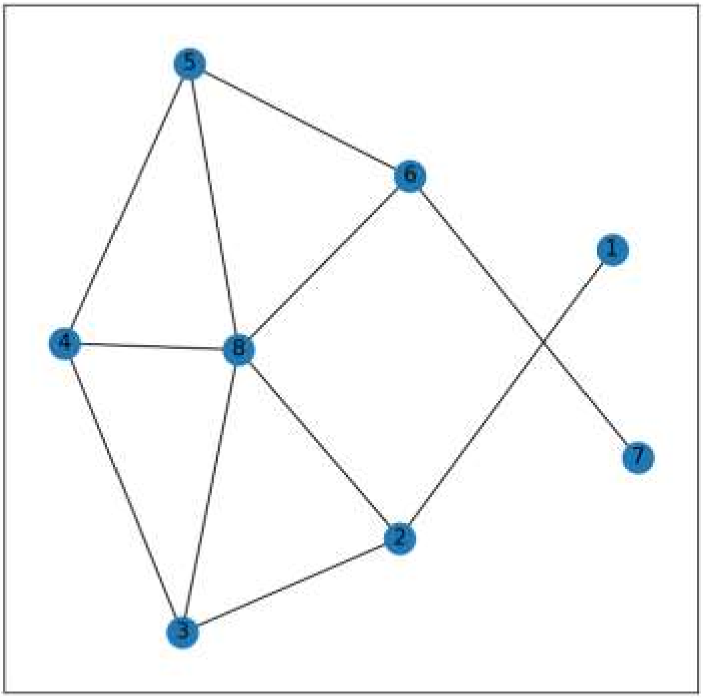


Figure 77. NodeGraph for two-dimensional frame in Figure 67

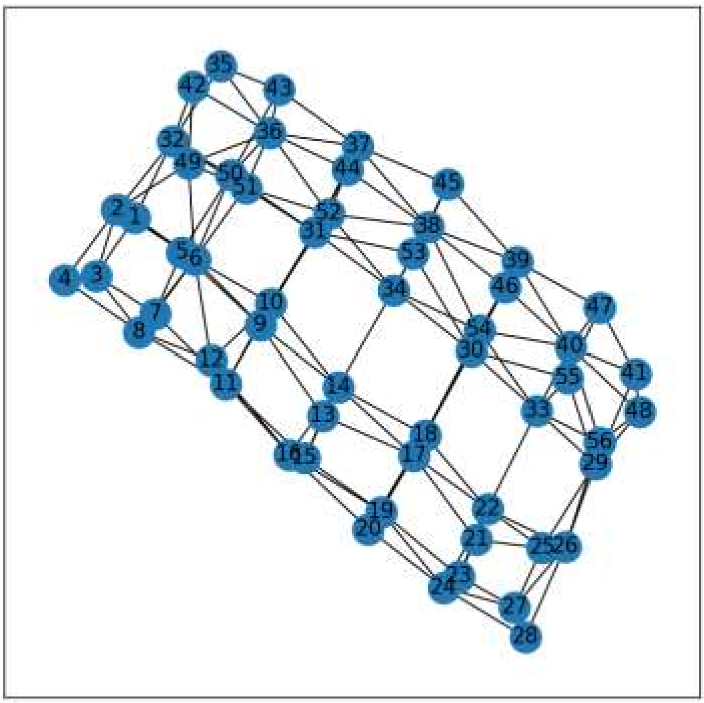


Figure 78. NodeGraph for spatial frame in Figure 68

The transformation matrix in this graph is obtained as a combination of the adjacency matrix *Adj*$^{(N)}$ of shape [*n* x *n*] with the transformation matrix between two connected nodes. Since the loads associated with each node in this graph are expressed in the global reference system of the model, the rotation matrix *R*$^{(N)}$ ([6 x 6]) turns out to be the same for any pair of nodes and is equal to an identity matrix. Therefore, the matrix $E^{(N)}$ ([6*n* x 6*n*]) can be calculated through the product of Kronecker:

$$E^{(N)} = [6n\ x\ 6n] = Adj^{(N)} \otimes R^{(N)} \tag{5.52}$$

with

$$Adj^{(N)}[k,p] = \begin{cases} 0 & if \nexists\ a\ beam\ with\ k\ and\ p\ as\ node\ ends \\ 1 & if \exists\ a\ beam\ with\ k\ and\ p\ as\ node\ ends \end{cases} \tag{5.53}$$

$$R^{(N)} = I_{6x6} \tag{5.54}$$

$E^{(N)}$ matrix can be re-written as follows:

$$E^{(N)} = \begin{pmatrix} e_{11}^{(N)} & \dots & e_{1n}^{(N)} \\ \dots & \dots & \dots \\ e_{n1}^{(N)} & \dots & e_{nn}^{(N)} \end{pmatrix} \tag{5.55}$$

with

$$e_{ij}^{(N)} = [6\ x\ 6] = \begin{cases} 0_{6x6} & if \nexists\ beam\ with\ i-th\ and\ j-th\ nodes\ at\ ends \\ I_{6x6} & if \exists\ beam\ with\ i-th\ and\ j-th\ nodes\ at\ ends \end{cases} \tag{5.56}$$

The eigenvectors associated with the 6 smallest eigenvalues of the Laplacian of adjacency matrix *Adj*$^{(N)}$ were used as input data:

$$X^{(N)} = [n\ x\ 6] \tag{5.57}$$

With a reshape, this input matrix can be written as a matrix of shape [6*n* x 1].

BeamGNN

BeamGNN is the network used to calculate the decoding matrix. It is built on the graph *BeamGraph* with the beams of the structural model as nodes. Figure 79 and Figure 80 show, respectively, the graphs for the frames in Figure 67 and in Figure 68.

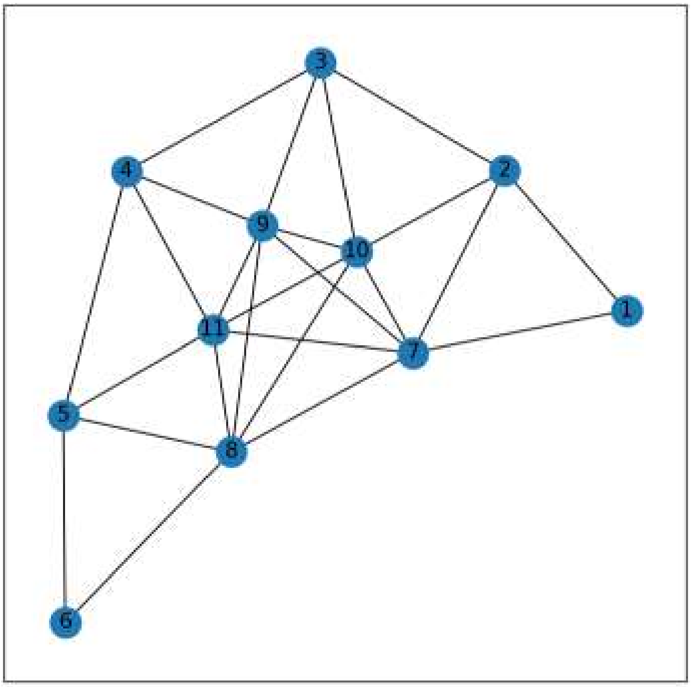


Figure 79. BeamGraph for two-dimensional frame in Figure 55

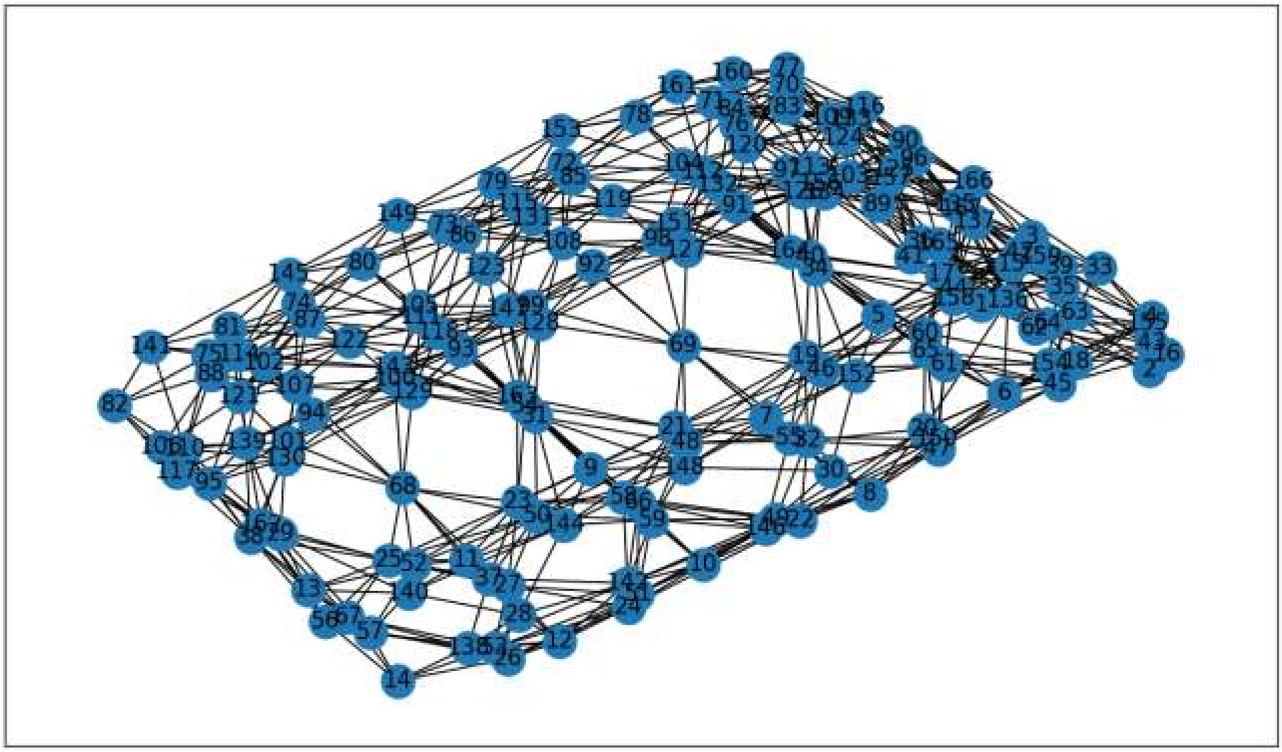


Figure 80. BeamGraph for spatial frame in Figure 56

As in the *NodeGraph* case, the eigenvectors associated with the smallest eigenvalues of the Laplacian of the graph adjacency matrix [b x b] were used as input data. The output features of each beam, on the other hand, express the tensional state at beam ends.

The transformation matrix to be used for convolution in this graph is the E matrix for beams-beams convolution presented in section 5.2.3.4 .

Network architecture

The network consists of a succession of convolution layers and nonlinear *LeakyReLU*-type layers. Problems of gradient vanishing necessitated the adoption of a *skip connection* type structure, in which the outputs of one convolutional layer are concatenated to the inputs of the previous layer to be used as inputs for the next convolution.

The input matrix of the *NodeGNN* is [n x 6] while that of the *BeamGNN* is [b x 12]. They are rewritten respectively in the form [6n x 1] and [12b x 1]. Since the input data are not referenced to the local reference system of each element, we can assume in the first layer:

$$E^0 = Adj \otimes I \tag{5.58}$$

where I is the identity matrix of dimension [n x n] for *NodeGNN* and [b x b] for *BeamGNN*.

The output of the first layer of the network can be written as follows:

$$\begin{aligned} Y^0 = M^0_{neighborhood} + M^0_{self-loop} &= (I \otimes Q^0)\left(A \otimes I_{p\,x\,p}\right)X^0W^0 - \left(I \otimes {Q_{sl}}^0\right)X^0{W_{sl}}^0 = \\ &= (Adj \otimes Q^0)X^0W^0 - \left(I_{b\,x\,b} \otimes {Q_{sl}}^0\right)X^0{W_{sl}}^0 \end{aligned} \tag{5.59}$$

The matrices of weights W^0 and W_{sl}^0 have shape [1 x k]. Instead, the matrices Q^0 and Q_{sl}^0 have size [6 x 6] and [12 x 12] for *NodeGNN* and *BeamGNN*, respectively. The parameter values contained in the matrices W^0, W_{sl}^0, Q^0 and Q_{sl}^0 must be learned during training. The output matrix obtained from this first layer will then have the shape [6n x 1] and [12b x 1] for *NodeGNN* and *BeamGNN*, respectively.

The convolutions of the inner layers, on the other hand, have a structure similar to that adopted for the convolution between beams in the heterogeneous graph.

$$Y^h = [p\ x\ 2^{h-1}] = M^h_{neighborhood} + M^h_{self-loop} = \left(I \otimes Q^h\right)EX^hW^h - (I \otimes {Q_{sl}}^h)X^h{W_{sl}}^h \qquad (5.60)$$

with *h*=1, 2, …. and *p*=6*n* for *NodeGNN* while *p*=12*b* for *BeamGNN*.

The size of the input matrix varies at each layer being the result of combining outputs from previous layers:

$$X^h = concat(X^{h-1}, O^{h-1}) = [p\ x\ 2^{h-1}] \qquad (5.61)$$

The matrices Q^h and $Q_{sl}{}^h$ have equal shape in each convolutional layer, equal to [6 x 6] and [12 x 12] for *NodeGNN* and *BeamGNN*, respectively. The size of the weight matrices W^h and $W_{sl}{}^h$, instead, increases with the depth of the layer and is equal to [2^{h-1} x 2^{h-1}].

The numerosity of the inner layers is such that a subspace of the chosen dimension *k* is reached.

The last convolution layer, on the other hand, has the task of transforming the final outputs of the network into the [6*n* x 1] and [12*b* x 1] forms. It has the following expression:

$$Y^{last} = [p\ x\ 1] = (I \otimes Q^{last})X^{last}W^{last} \qquad (5.62)$$

where Q^{last} have size [6 x 6] and [12 x 12] for *NodeGNN* and *BeamGNN*, respectively, while W^{last} is [*k* x 1].

Loss function

The two GNN networks, characterized by a sequence of convolution layers and nonlinear *LeakyReLU* layers, lead to the calculation of the encoding A [6*n* x *k*] and decoding B [12*b* x *k*] matrices. The former relates the space of loads to space of size *k* while the latter relates beam forces to space of size *k*. The two GNNs are trained in parallel but using the same loss for updating the weight matrices at each epoch. The loss is the MAE function that measures the error between the beam forces calculated by multiplying the loads with the encoding and decoding matrices and the actual beam forces S_i. For *i*-th result case we can write:

$$Loss = MAE(B * A^T * C_i - S_i) \qquad (5.63)$$

Results of Twin GNN

The twin GNN approach was tested on the two-dimensional frame for different dimensions of problem space *k*. The results of the analysis are shown in Figure 81 and in Figure 82. It is observed that after a sharp decline in the error in the first epochs, the value of the loss remains constant for several epochs and then undergoes another sharp decline until a new plateau is reached. The trend of the error detected in the validation set mimics that of the loss calculated in the training set. The change in the dimension of subspace *k* does not lead to important differences in the error values found in the two plateaus, while it influences the epoch at which starting from the first plateau the reduction of error begins until the second plateau is reached. More precisely, an increase in the dimension of subspace *k* leads to an increase in the number of epochs in which the first plateau persists. Despite the small size of the structure under consideration, the approach was computationally onerous, taking a long time to achieve convergence. This makes it difficult to use this approach with more complex models. Indeed, this approach was also tried with the spatial frame subjected to concentrated loads using *k*=16, and the results are shown in Figure 83 and Figure 84. Compared with the two-dimensional frame case, increasing the size of the graphs associated with the structure resulted in a significant increase in the computation time required for each epoch. Furthermore, in the first 1400 epochs there was only one major reduction in error during the first ten epochs, after which the error settled to a plateau, maintaining high and unacceptable values.

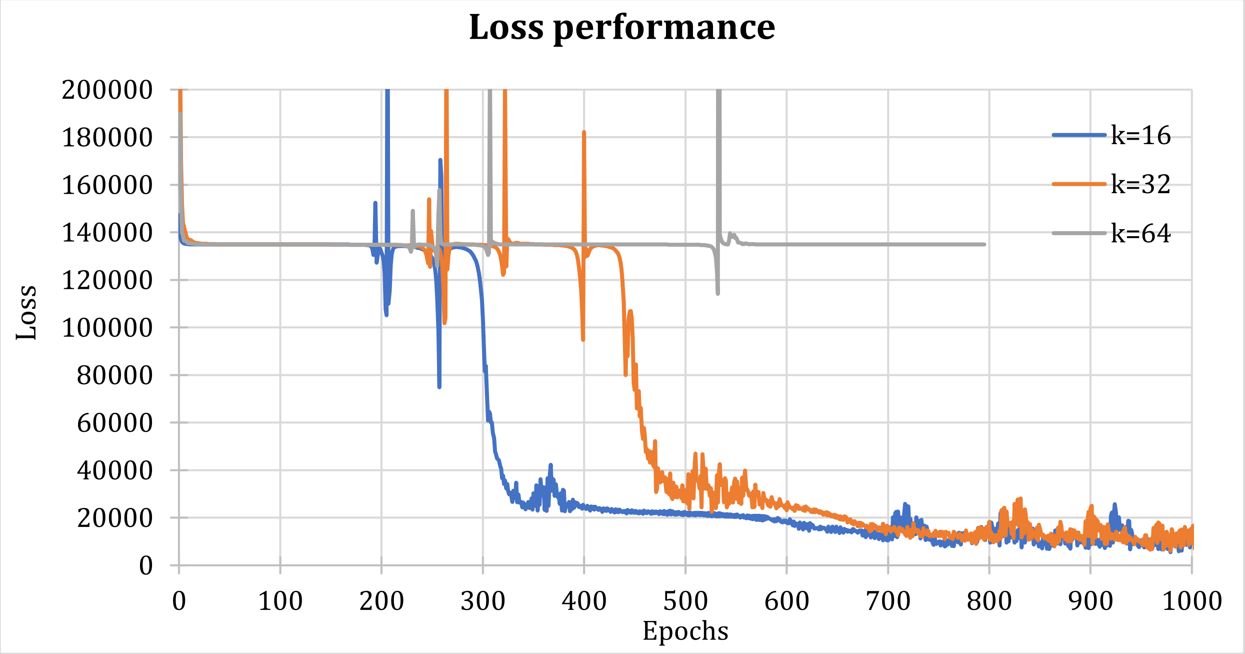


Figure 81. Trend of twin GNN loss for two-dimensional structure for different dimensions of space k

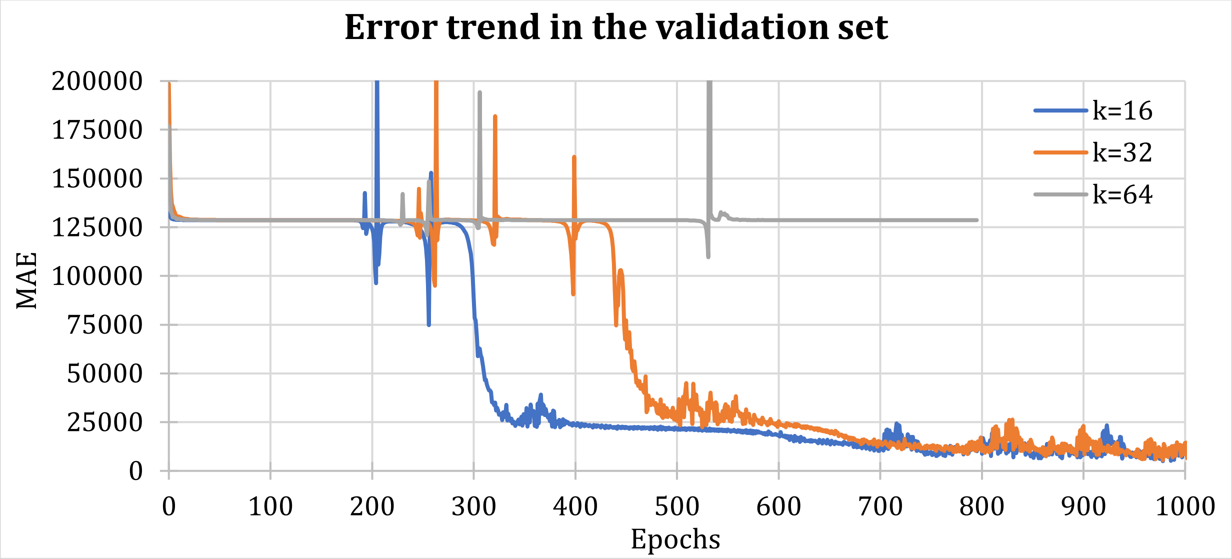


Figure 82. Error trend of twin GNN in the validation set for two-dimensional structure for different dimensions of space k

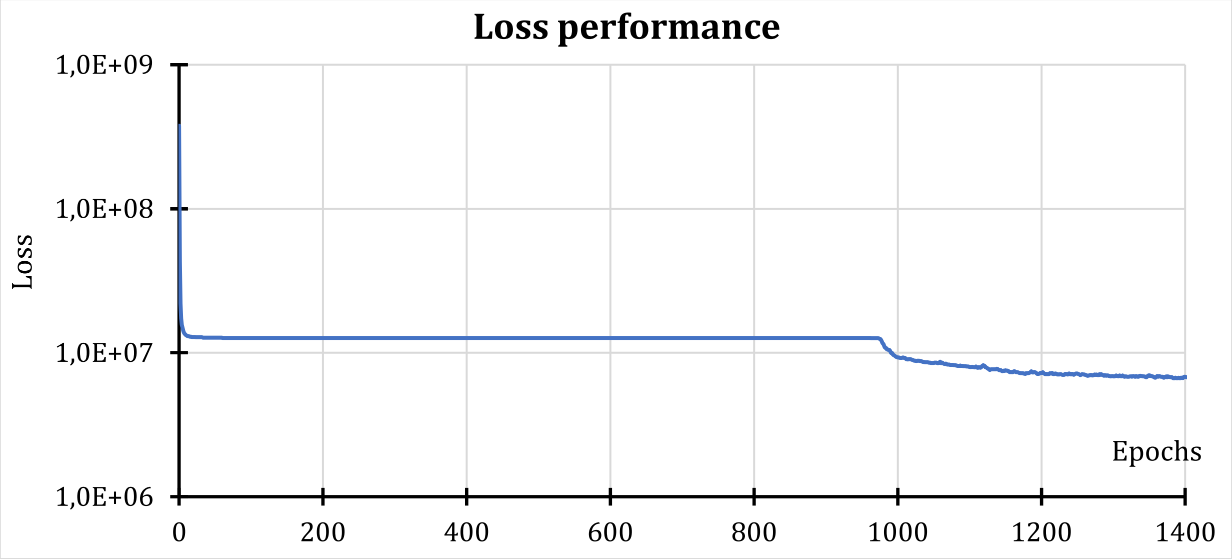


Figure 83. Trend of twin GNN loss for spatial frame with concentrated loads for k=16

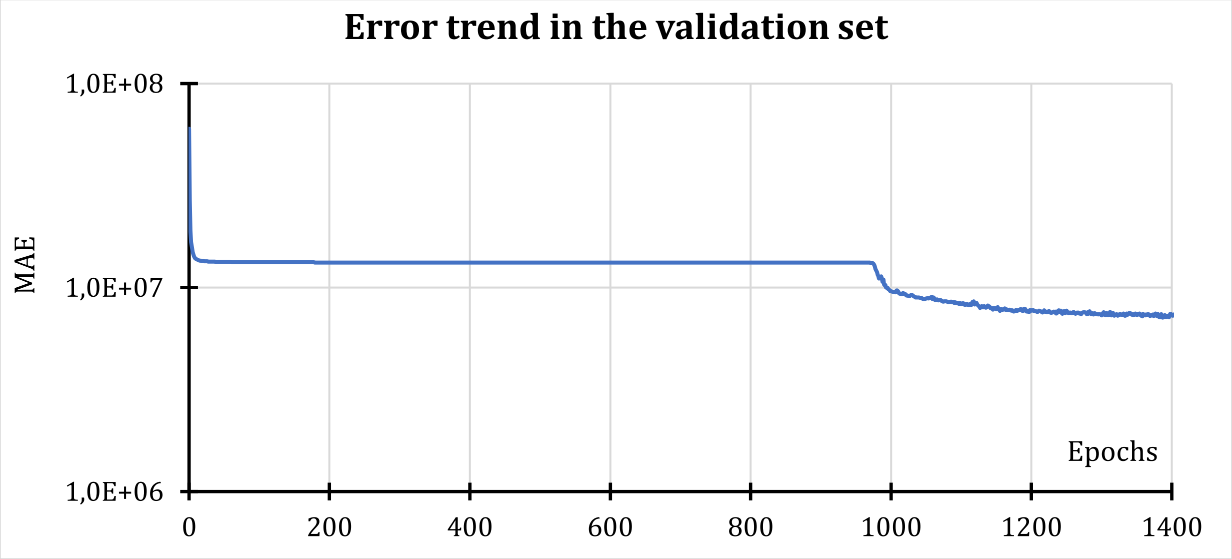


Figure 84. Error trend of twin GNN in the validation set for spatial frame with concentrated loads for k=16

5.2.4 Comparing approaches

Table 37 shows the errors found in the training set and validation set during the construction of surrogate models for the two-dimensional frame. The lowest error was detected with the low-rank stiffness matrix. This method is also the one that requires the lowest computational cost and construction time, since there is no need to perform an iterative process for training the model.

APPROACH	ERROR TRAINING SET	ERROR VALIDATION SET
Low-rank stiffness matrix	2.219	1.887
MLP	226.272	304.000
Heterogeneous graph	1726.221	1855.000
Twin GNN	2578.184	6061.000

Table 37. Errors of different approaches for constructing the surrogate model for the bidimensional frame

5.3 *MLPsectionOpt* module

The choice of section to be assigned to each beam must be such that it can be used as much as possible. This results in finding the section with which the entity has a utilization rate at lower but close to unity. In this thesis, the phenomena of oligocyclic and polycyclic fatigue, plasticity and fracture toughness are left out. Therefore, we only consider strength-related checks derived from the standards. These verifications depend on the forces acting on the beam cross-section. In addition, in the case of compressed elements, the phenomenon of buckling also comes into play, which also depends on the length of the beam and the moment diagrams along the development of the beam.

As explained in the previous paragraphs, structural checks also depend on the type of section. In this thesis, Circular Hollow Sections (CHS) were considered, so the structural verifications are similar to those in the section 4.4.1

It is necessary to build a mathematical model that for a given set of forces acting on a beam, proposes the section that provides the desired utilization rate. A Multi-Layer Perceptron (MLP) Neural Network was chosen for this purpose. The network consists of a succession of linear and nonlinear *ReLU*-type layers. The type of problem to be solved is a classification, that is, the section obtained in output is chosen from within a database of available sections. The loss function with which the network is trained is the Cross-Entropy.

The network has 20 input data, 18 of which indicate the forces on the beams at the ends and mid-lengths, while the other two define the length of the beams and the utilization rate. To facilitate learning the network, the input data are scaled and normalized.

5.3.1 Database

The choice of database is particularly important to obtain good results from using the model. It should be large enough to cover all possible scenarios. It should also be suitable in relation to the structure to be optimized. In fact, the forces and the size of the beams of that structure, which will become the input data when using the model, should fall within the space analyzed by the network model. The database can be obtained using pre-existing designs: in this way it is possible to take advantage of the optimization reasoning carried out earlier. However, this requires that there be a sufficiently large number of preexisting designs from which to acquire data and that among these there are instances similar to the structure to be optimized. Alternatively, the database can be created ad hoc: in this case it is advisable to exploit techniques that improve sampling in order to obtain a better-quality database.

An ad hoc database was created for the two-dimensional frame analysis, using the forces on the beams of the starting model in the various load cases and calculating the utilization rate for various sections and member lengths. Sampling was then limited to the choice of m section-length element pairs for each beam for each load case. Two different statistical sampling methods were evaluated: Uniform Distribution and Latin Hypercube. The distribution obtained with the two types of sampling is shown in Figure 85 and Figure 86.

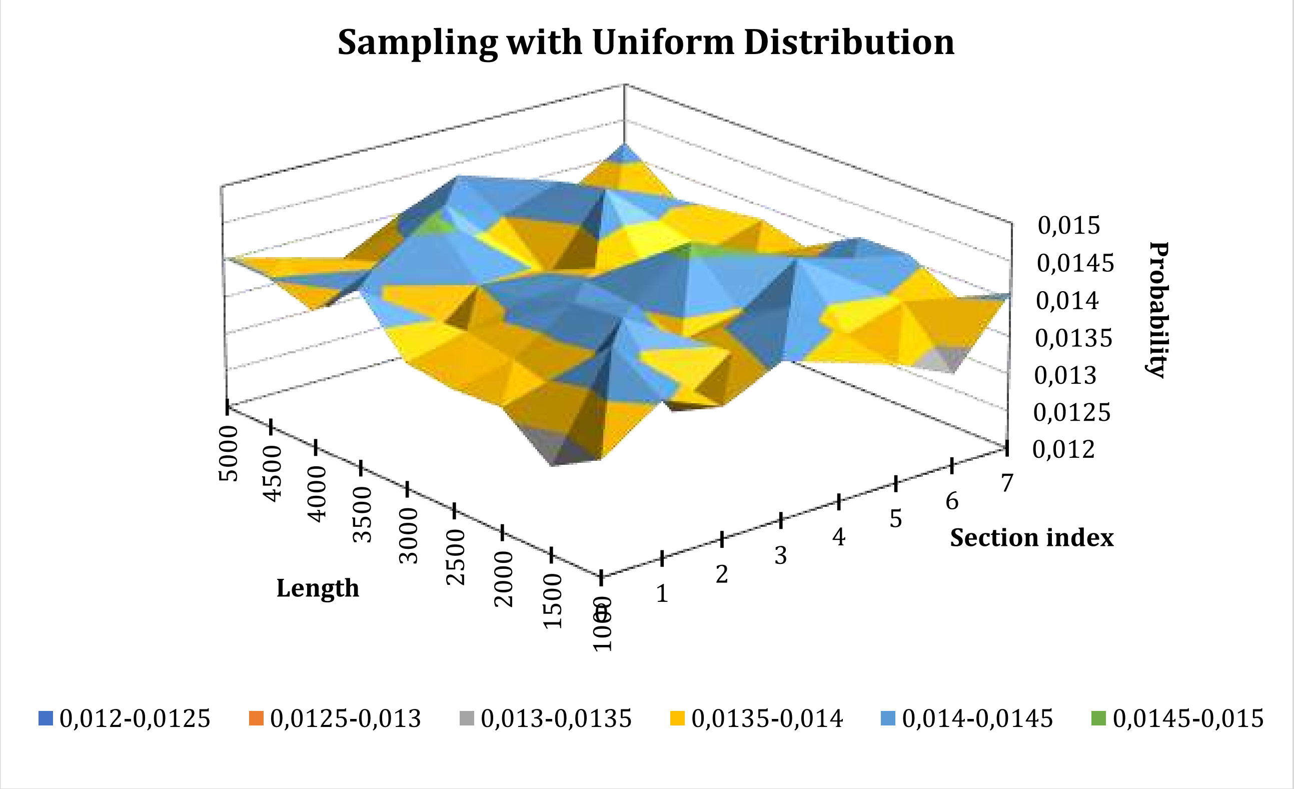


Figure 85. Sampling of length-section pairs obtained with Uniform Distribution

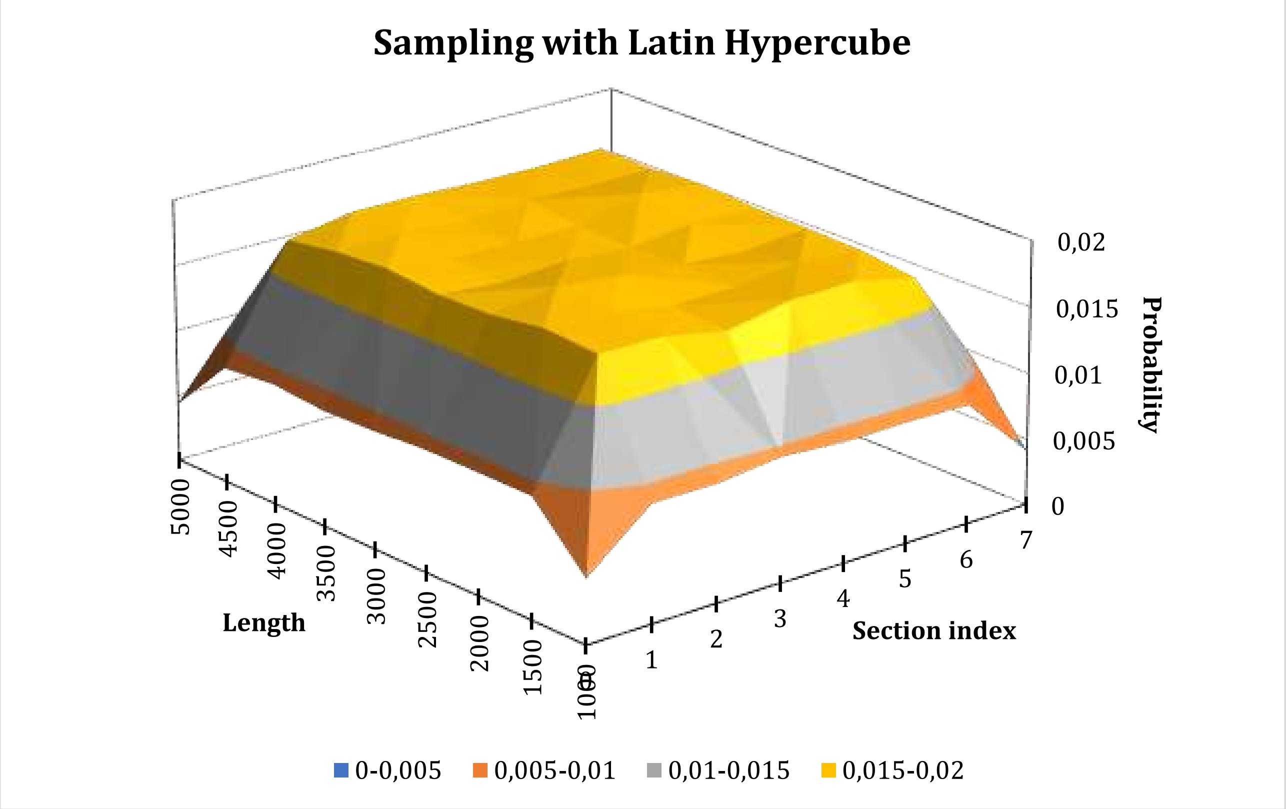


Figure 86. Sampling of length-section pairs obtained with Latin Hypercube

5.3.2 Training

The MLP network for defining the beam section is trained using a database of known cases or a database created ad hoc, as discussed in the previous section. The database was divided to use 80% as a training set and 20% as a test set. Different network architectures were tested during the training phase, obtained by changing the number of layers making up the network (n) and the size of the internal layers (k). For each configuration, the loss trend in the training set over epochs was analyzed. The loss function used is the Cross-Entropy. In the test set, the number of incorrect predictions is evaluated against the total size of the test set. In addition, for incorrect predictions, an average of how much the predicted class deviates from the correct class are calculated.

For the two-dimensional frame in Figure 67, the networks obtained using the database created with Uniform Distribution and that obtained with Latin Hypercube sampling were evaluated. The graphs in Figure 87, Figure 88 and Figure 89 show the error assessment metrics related to the database with uniform sampling. For Latin Hypercube sampling, on the other hand, the performances observed in the training are shown in the graphs in from the Figure 90 to Figure 95. It can be seen from the graphs that too few nodes in the hidden layers causes a slowdown in network learning. However, it is not advisable to increase the number of nodes in the inner layers too much because, in addition to leading to an increase in computational cost, beyond a certain size, no more reductions in the number of epochs needed for learning convergence are observed. Similarly, too few layers in the network leads to a slowdown in learning speed. However, increasing the number of network layers too many leads to more unstable learning as well as increased computational cost. With both databases, the optimal network architecture is similar: n=3 pairs of inner layers of linear and $ReLU$-type and k=256 size of inner layers. Metric to evaluate network performance during training also provides similar results for the two databases.

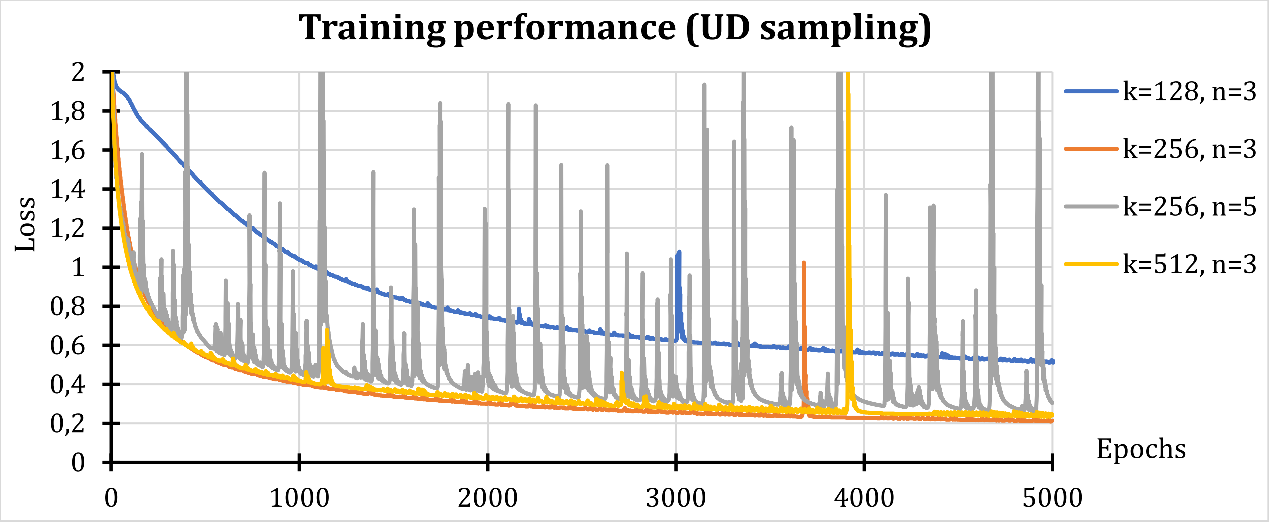


Figure 87. Training performance for different network architectures created with Uniform Distribution sampling for two-dimensional frame optimization in Figure 67

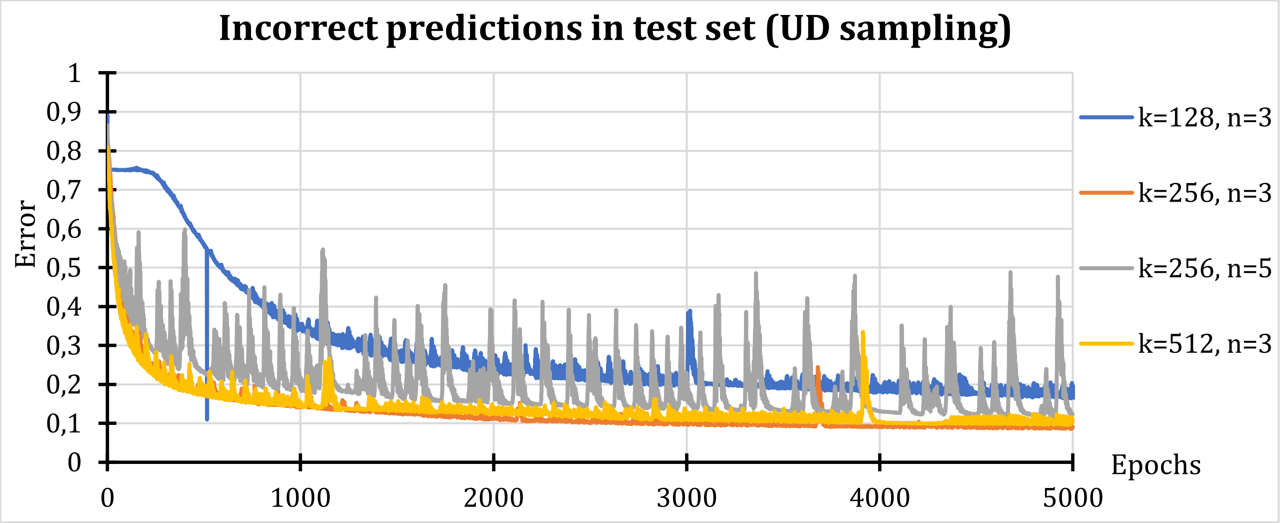


Figure 88. Incorrect predictions in test set for different network architectures. Data are referred to Neural Networks created with Uniform Distribution sampling for two-dimensional frame optimization in Figure 67

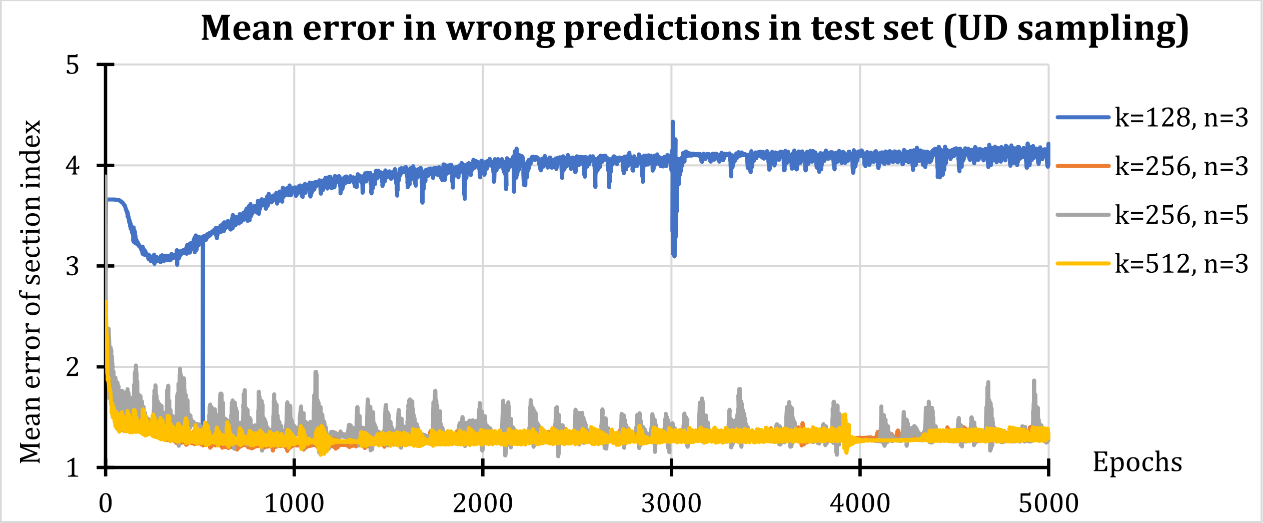


Figure 89. Mean number of difference classes between target and network output for items of test set with incorrect predictions. Data are referred to Neural Networks created with Uniform Distribution sampling for two-dimensional frame optimization in Figure 67

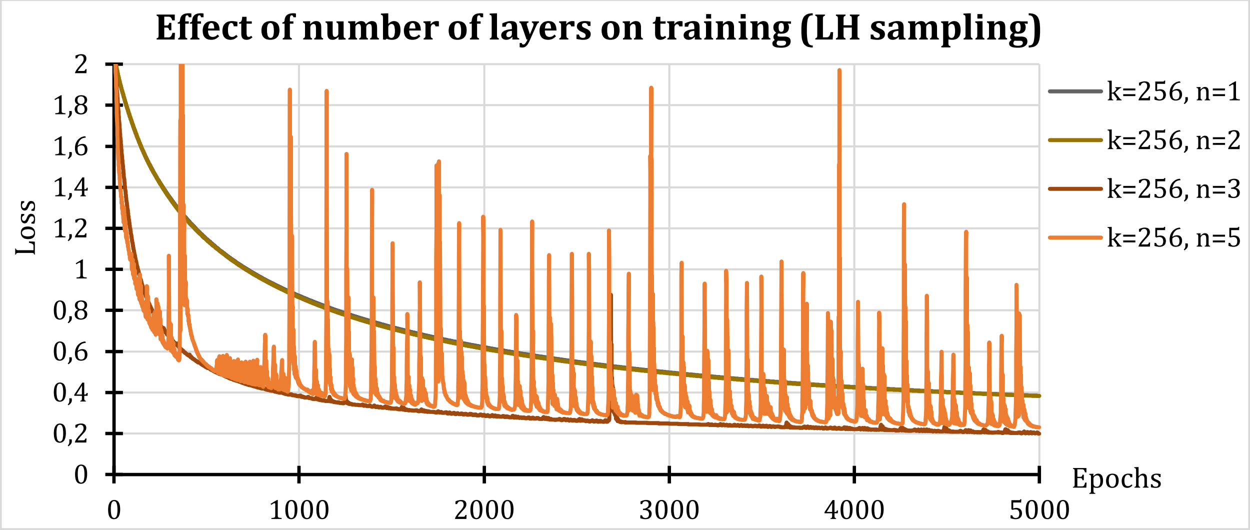


Figure 90. Training performance for different number of hidden layers (n) with inner layers having k=256 nodes. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67

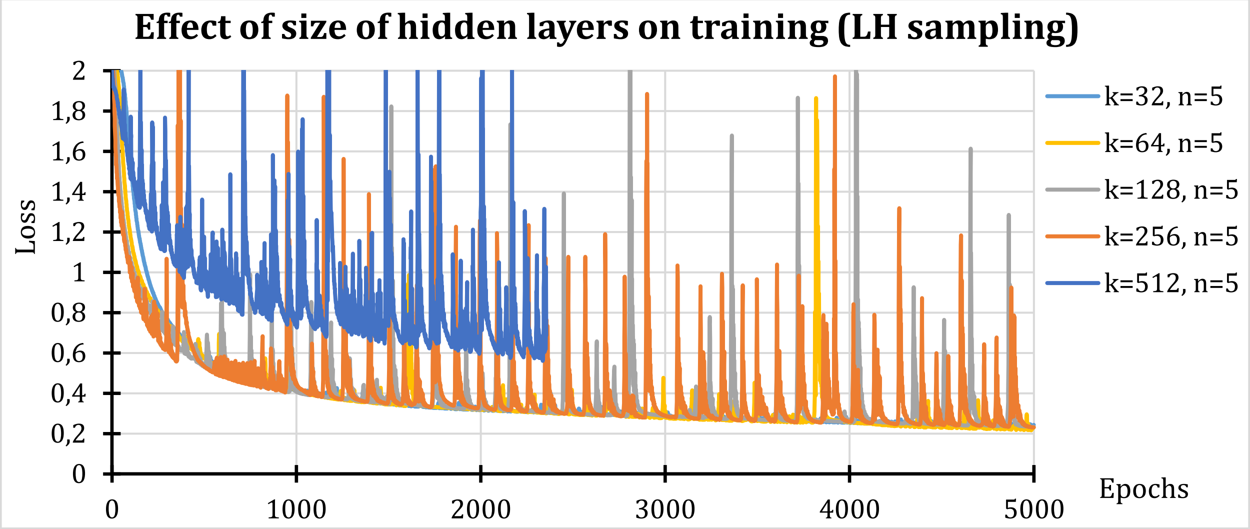


Figure 91. Training performance for different number of nodes in the inner layers (k) with the number of layers equal to 5. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67

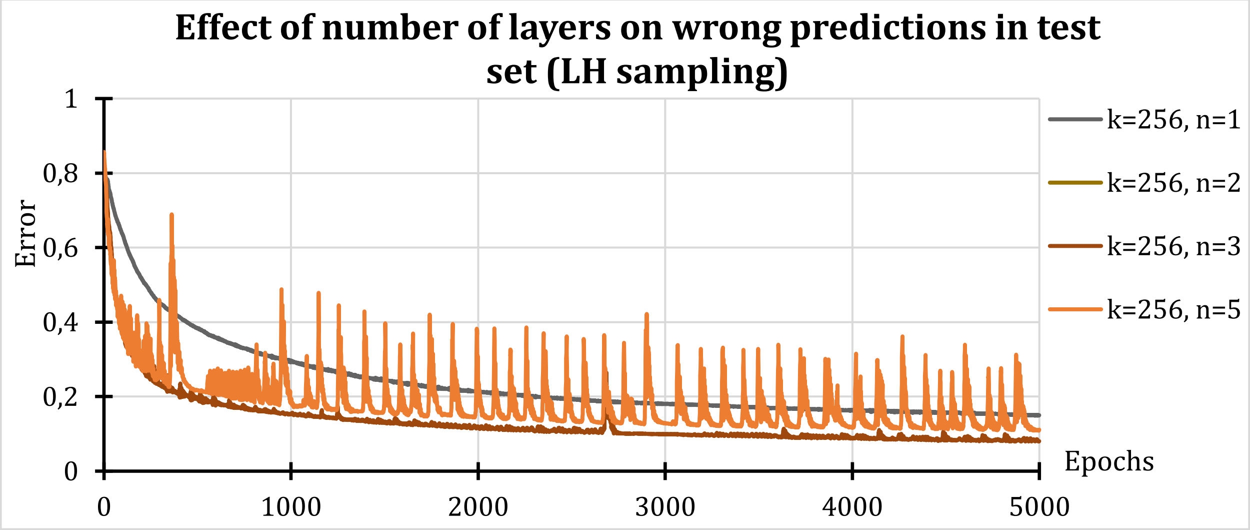


Figure 92. Incorrect predictions in test set for different number of hidden layers (n) with inner layers having k=256 nodes. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67

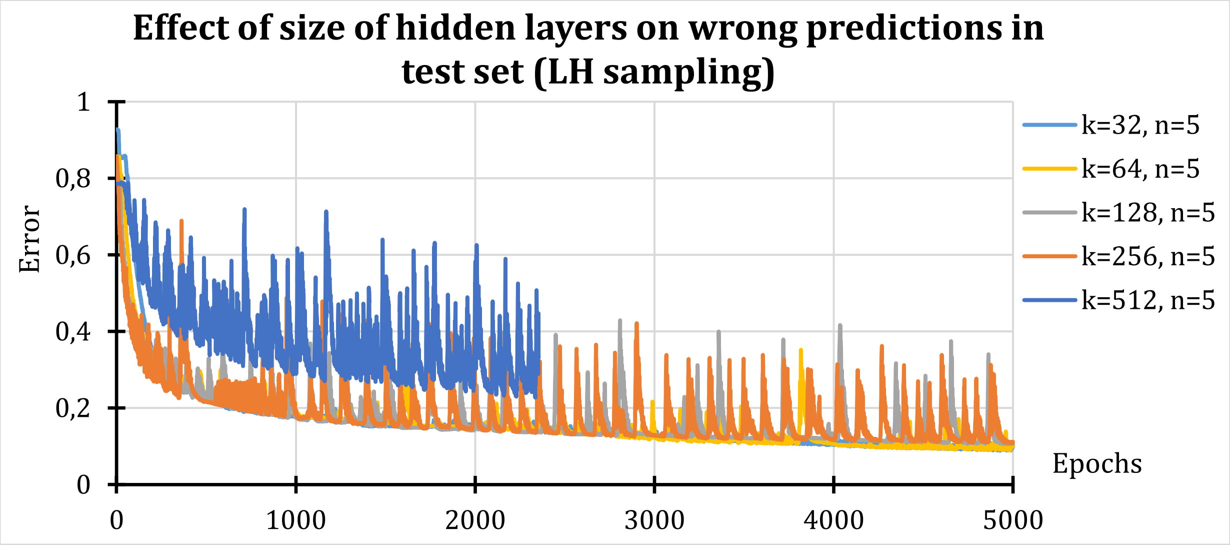


Figure 93. Incorrect predictions in test set for different number of nodes in the inner layers (k) with the number of layers equal to 5. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67

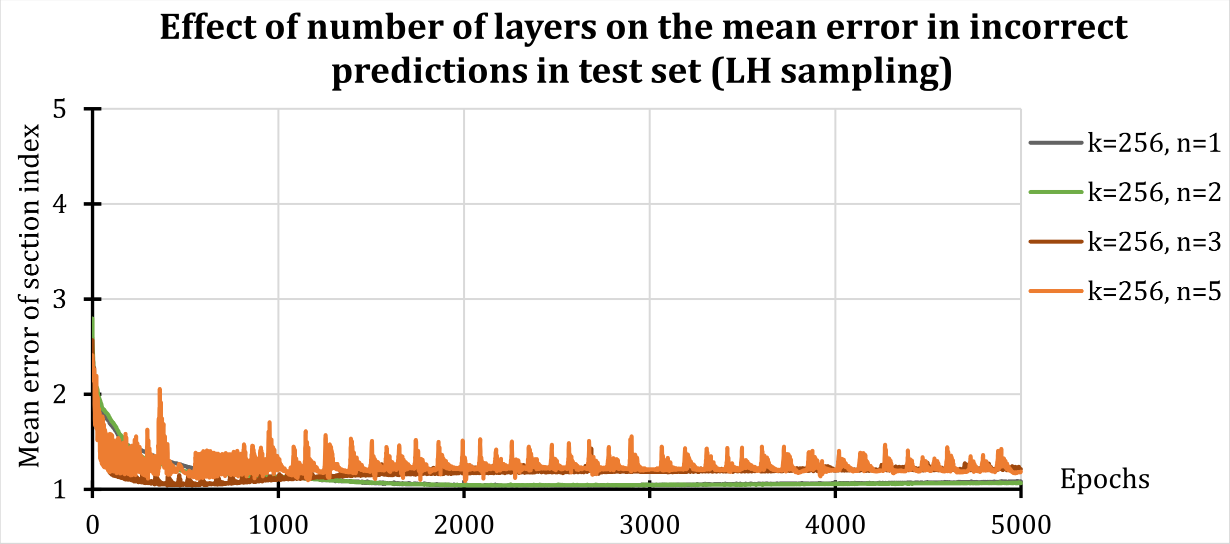


Figure 94. Mean error in incorrect predictions in test set for different number of hidden layers (n) with inner layers having k=256 nodes. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67

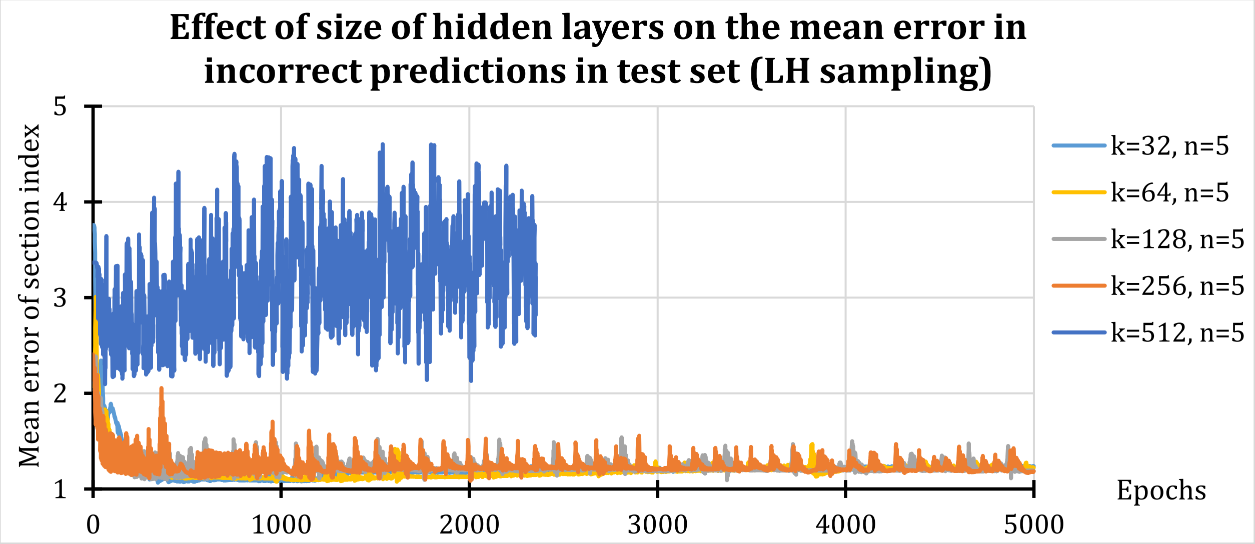


Figure 95. Mean error in incorrect predictions in test set for amounts of nodes in the inner layers (k) with 5 pairs of linear-ReLU inner layers. The database used is created with Latin Hypercube sampling for two-dimensional frame optimization in Figure 67

5.3.3 Optimization results

The *MLPsectionOpt* network enables optimization of the cross sections of beams that make up a frame structure. The proposal of new sections by the network results in a change in the loads acting on the structure and the distribution of stiffnesses of the structure, resulting in a change in the stresses on the beams. Consequently, the use of this Neural Network for optimization purposes should be placed within an iterative workflow (Figure 96). At iteration t, the matrix X^t contains information about the forces on the beams, their length and the desired utilization rate. For each load case, through the *MLPsectionOpt* network, we define the cross section of each beam that leads to having the desired utilization rate, which was set to 0.99. Each beam is associated with the largest cross section of those obtained with the different load cases (Y^t). The forces on the beams are then updated using the low-rank stiffness matrix in the *LoadsToForce* module. The process is repeated until convergence, after which the FEM model is updated with the new sections to calculate by finite elements more accurately the stresses on the beams with the new sections. A new low-rank matrix is obtained from the modified FEM model, and the *LoadsToForce* module is updated accordingly. Optimized sections are calculated again through the *MLPsectionOpt* network using the stresses on the beams obtained from the FEM solution. If the output of the network remains unchanged from the result provided before updating the FEM model, then the process is finished, otherwise iterations are continued after updating the stresses with the new sections provided by the network using the new low-rank stiffness matrix.

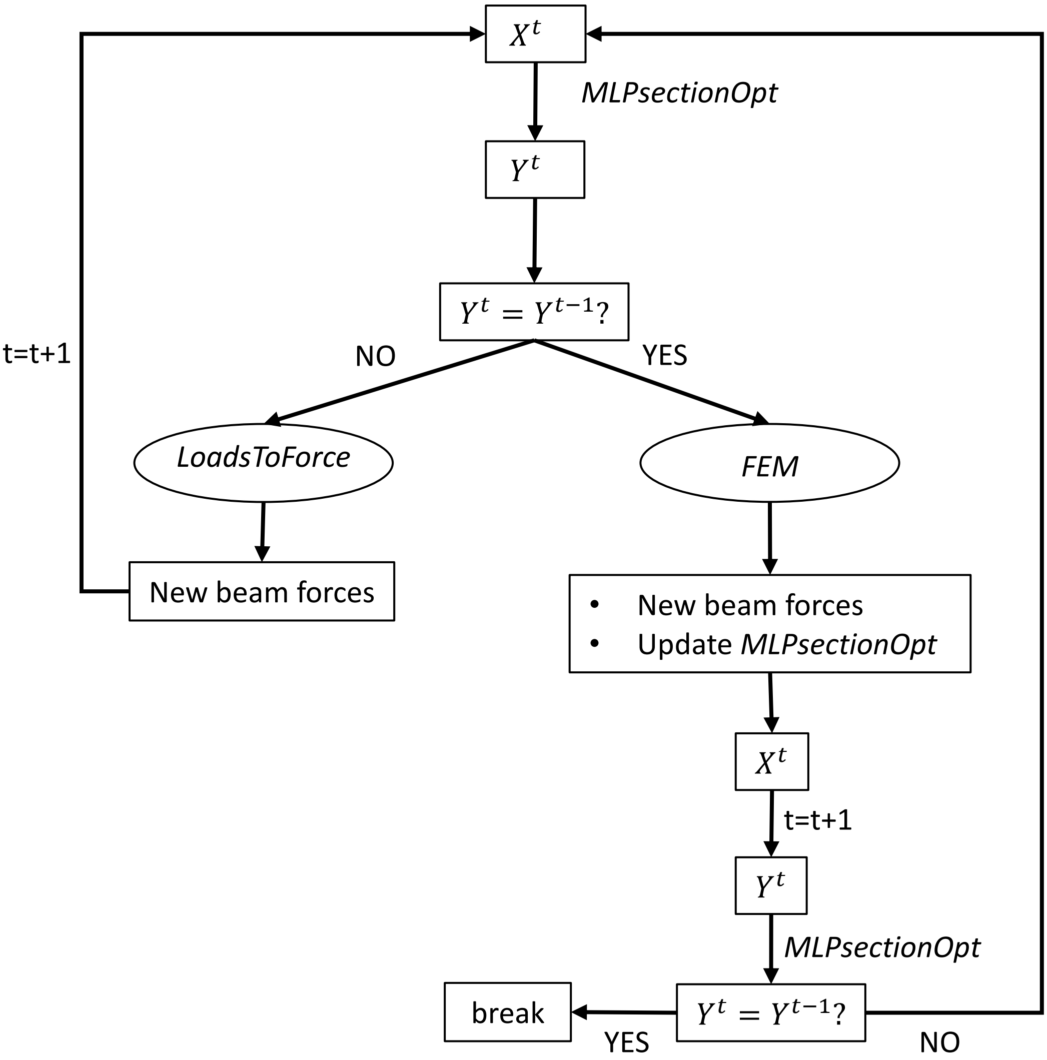


Figure 96. Iterative optimization workflow

5.3.3.1 Two-dimensional Frame

The starting two-dimensional frame model provides that all beams have a cross section with a diameter of 273.00 mm and a thickness of 5.0 mm. Performing the structural analysis, it is observed that two beams are undersized and three have a utilization rate of less than 50%. The results are shown in Table 38. The optimization workflow outlined in Section 5.3.3 was used to calculate beam sections that produce a utilization rate closer to the target rate than the initial model.

Beam	Length [mm]	ID Section	Diam [mm]	Thk [mm]	u_{max}	$u_{max}-\bar{u}$	$\lvert u_{max}-\bar{u}\rvert$
1	2500	6	273.0	5.0	1.414	0.424	0.424
2	1500	6	273.0	5.0	0.727	-0.263	0.263
3	4000	6	273.0	5.0	0.737	-0.253	0.253
4	4000	6	273.0	5.0	0.739	-0.251	0.251
5	1500	6	273.0	5.0	0.846	-0.144	0.144
6	2500	6	273.0	5.0	1.396	0.406	0.406
7	4000	6	273.0	5.0	0.474	-0.516	0.516
8	4000	6	273.0	5.0	0.462	-0.528	0.528
9	1500	6	273.0	5.0	0.325	-0.665	0.665
10	4272	6	273.0	5.0	0.673	-0.317	0.317
11	4272	6	273.0	5.0	0.666	-0.324	0.324
SUM						**-2.431**	**4.091**
MIN						**-0.665**	**0.144**
MAX						**0.424**	**0.665**
MEAN						**-0.221**	**0.372**
NUMBER OF UNDER-SIZED BEAMS						**2**	

Table 38. Beam starting sections and corresponding utilization rate of the two-dimensional frame. The parameter $\bar{u}$ is the target utilization rate, which is considered to be equal to 0.99

MLP networks with 256 nodes in the inner layers and 3 pairs of intermediate layers (linear+*ReLU*) were used to optimize the two-dimensional structure. The optimized models are shown in Figure 97 and Figure 98. Table 39 shows the sizes of the optimized sections and the maximum utilization ratios associated with each beam. It is observed that the networks obtained with the two databases lead to similar results between them.

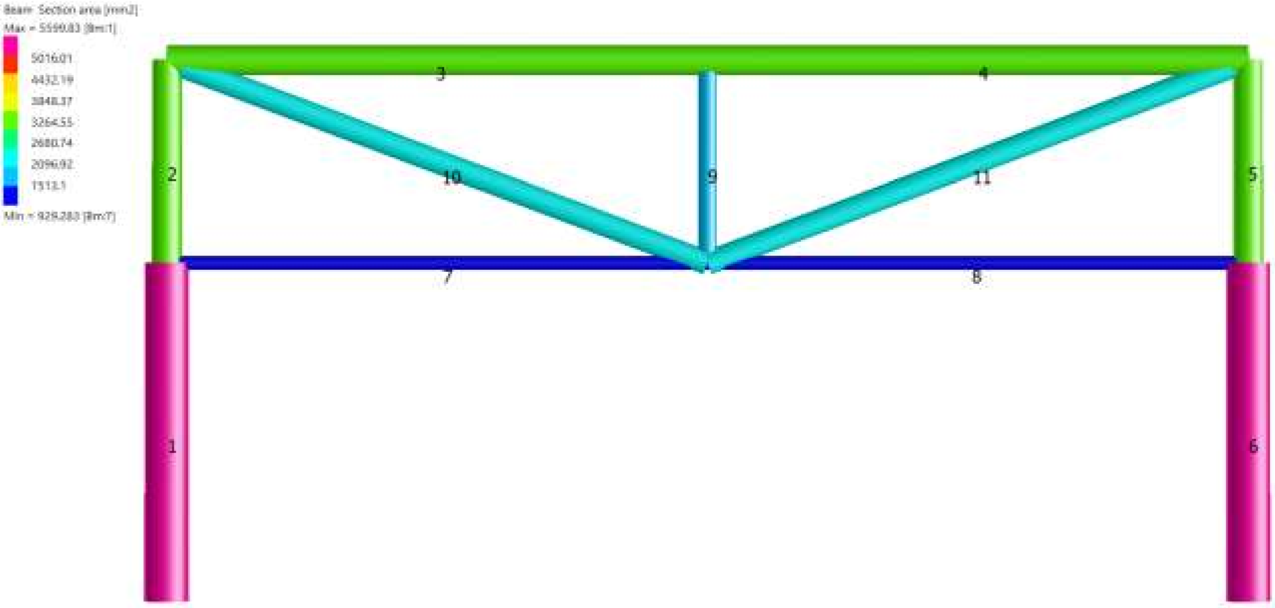


Figure 97. Two-dimensional frame optimized with network obtained from database with Uniform Distribution sampling

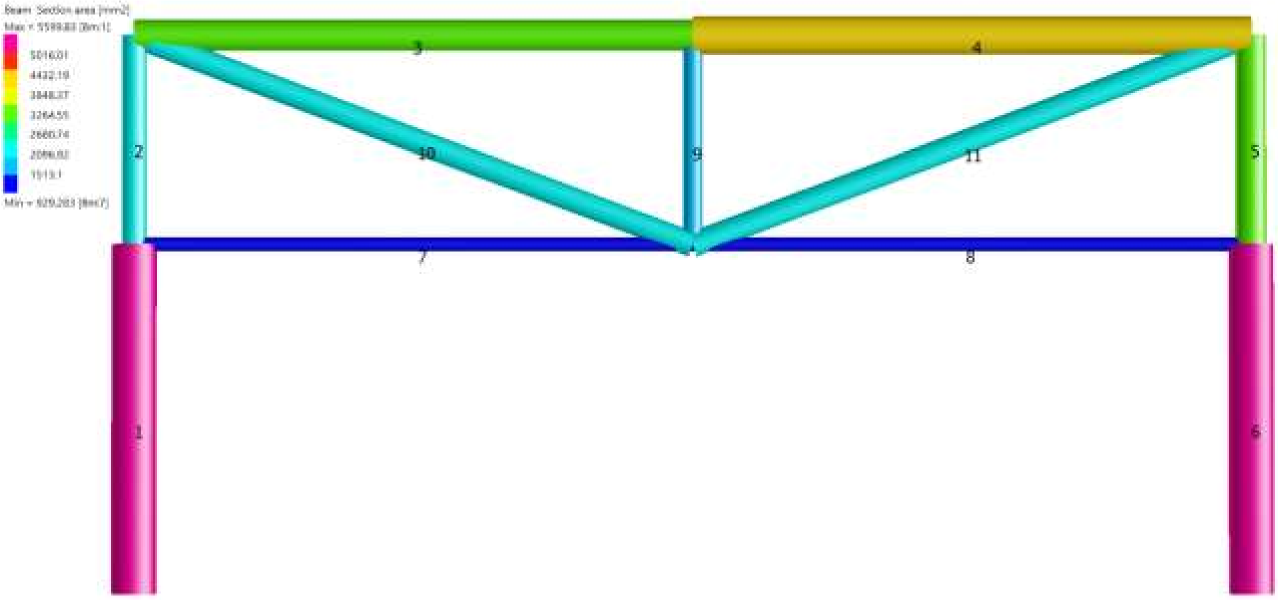


Figure 98. Two-dimensional frame optimized with network obtained from database with Latin Hypercube sampling

Beam	Length [mm]	Uniform sampling				Latin Hypercube sampling				Uniform sampling		Latin Hypercube sampling	
		ID Section	Diam [mm]	Thk [mm]	u_{max}	ID Section	Diam [mm]	Thk [mm]	u_{max}	$u_{max}-\bar{u}$	$\lvert u_{max}-\bar{u}\rvert$	$u_{max}-\bar{u}$	$\lvert u_{max}-\bar{u}\rvert$
1	2500	7	323.9	5.6	1.147	7	323.9	5.6	1.172	0.157	0.157	0.182	0.182
2	1500	5	219.1	5	0.767	4	177.8	4	0.916	-0.223	0.223	-0.074	0.074
3	4000	5	219.1	5	0.799	5	219.1	5	0.837	-0.191	0.191	-0.153	0.153
4	4000	5	219.1	5	0.799	6	273	5	0.834	-0.191	0.191	-0.156	0.156
5	1500	5	219.1	5	0.852	5	219.1	5	0.871	-0.138	0.138	-0.119	0.119
6	2500	7	323.9	5.6	1.176	7	323.9	5.6	1.256	0.186	0.186	0.266	0.266
7	4000	2	101.6	3	1.094	2	101.6	3	0.980	0.104	0.104	-0.010	0.010
8	4000	2	101.6	3	0.959	2	101.6	3	1.039	-0.031	0.031	0.049	0.049
9	1500	3	139.7	4	0.818	3	139.7	4	0.892	-0.172	0.172	-0.098	0.098
10	4272	4	177.8	4	1.029	4	177.8	4	0.861	0.039	0.039	-0.129	0.129
11	4272	4	177.8	4	1.014	4	177.8	4	0.894	0.024	0.024	-0.096	0.096
SUM										**-0.435**	**1.456**	**-0.338**	**1.331**
MIN										**-0.223**	**0.024**	**-0.156**	**0.010**
MAX										**0.186**	**0.223**	**0.266**	**0.266**
MEAN										**-0.040**	**0.132**	**-0.031**	**0.121**
NUMBER OF UNDER-SIZED BEAMS										**5**		**3**	

Table 39. Results of the optimization of the beams with the networks created with the two databases

5.3.3.2 Three-dimensional Frame

The starting model of the three-dimensional frame under study is shown in Figure 99. Information on beam sections and maximum utilization rates in this model is shown in Table 40. For some beams, the section used is class 4 according to (9), for which local buckling phenomena come into play before the section can express the elastic resistant moment, and the calculation of the section strength must be carried out not considering the whole section but only the part of it that is effective, i.e., not subject to the buckling problems. This type of section requires special iterative analysis for defining the utilization rate. The focus on this thesis has been on sections that can be fully utilized in the elastic field, namely those belonging to class 1, 2 and 3 according to the above classification. For these reasons, sections in class 4 were not considered in this thesis, and the utilization rates for the starting frame beams having sections belonging to this type are not shown in Table 40.

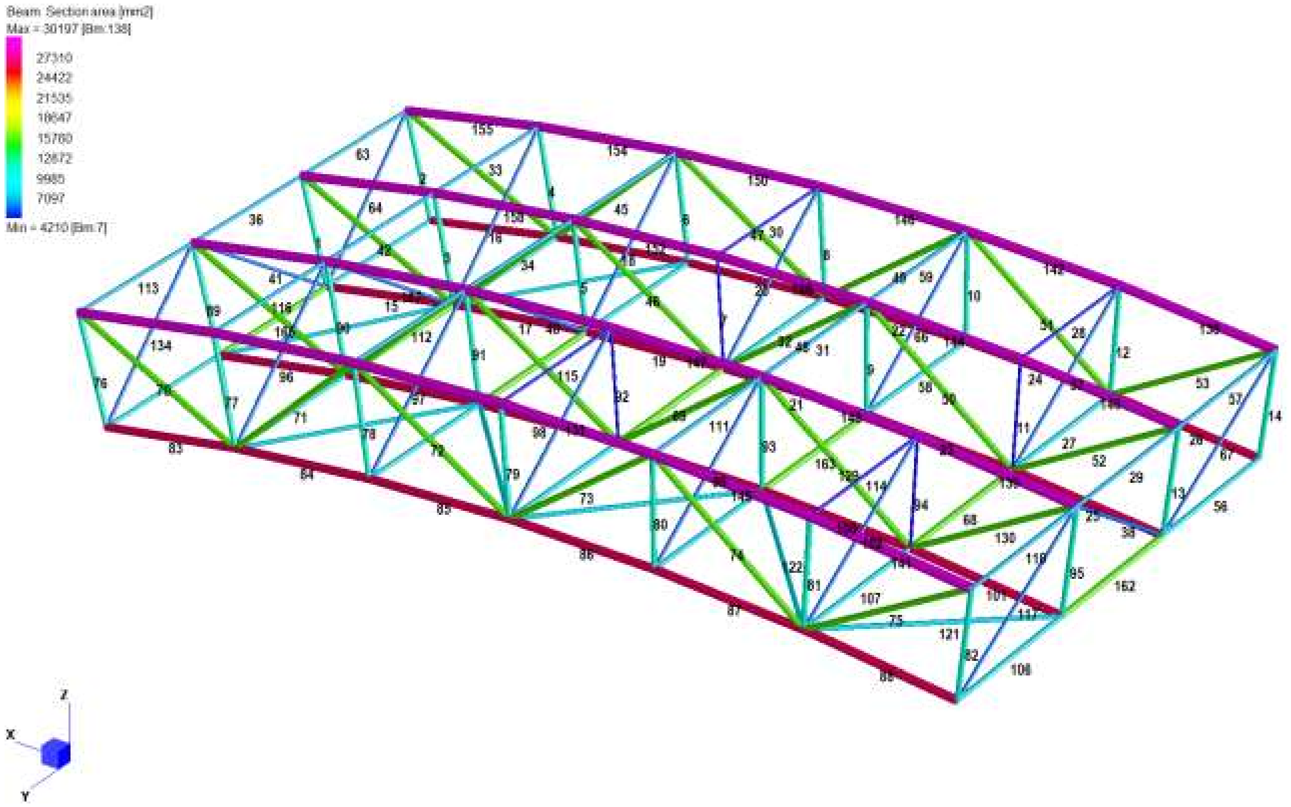


Figure 99. Non-optimized three-dimensional frame (starting model)

Beam	ID Section	Diam [mm]	Thk [mm]	Length [mm]	Result case max	u_{max}	$u_{max}-\bar{u}$	$\lvert u_{max}-\bar{u}\rvert$
1	9	457.0	8.0	10000.00	723	0.326	-0.664	0.664
2	9	457.0	8.0	10000.00	529	0.868	-0.122	0.122
3	9	457.0	8.0	10000.00	529	0.319	-0.671	0.671
4	9	457.0	8.0	10000.00	529	0.256	-0.734	0.734
5	9	457.0	8.0	10000.00	967	0.322	-0.668	0.668
6	9	457.0	8.0	10000.00	879	0.368	-0.622	0.622
7	1	273.0	5.0	10000.00	664	1.228	0.238	0.238
8	9	457.0	8.0	10000.00	636	0.422	-0.568	0.568
9	9	457.0	8.0	10000.21	530	0.373	-0.617	0.617
10	9	457.0	8.0	10000.21	786	0.377	-0.613	0.613
11	1	273.0	5.0	10000.00	819	1.312	0.322	0.322
12	9	457.0	8.0	10000.00	730	0.396	-0.594	0.594
13	9	457.0	8.0	10000.00	819	0.606	-0.384	0.384
14	9	457.0	8.0	10000.00	652	0.950	-0.040	0.040
15	20	711.0	12.0	13653.85	181	0.357	-0.633	0.633
16	20	711.0	12.0	13653.85	608	0.687	-0.303	0.303
17	20	711.0	12.0	13653.85	516	0.493	-0.497	0.497
18	20	711.0	12.0	13653.85	728	0.597	-0.393	0.393
19	20	711.0	12.0	13653.85	97	0.387	-0.603	0.603
20	20	711.0	12.0	13653.85	631	0.420	-0.570	0.570
21	20	711.0	12.0	13653.85	440	0.321	-0.669	0.669
22	20	711.0	12.0	13653.85	771	0.282	-0.708	0.708
23	20	711.0	12.0	13653.85	608	0.291	-0.699	0.699
24	20	711.0	12.0	13653.85	679	0.260	-0.730	0.730
25	20	711.0	12.0	13653.85	617	0.264	-0.726	0.726
26	20	711.0	12.0	13653.85	529	0.594	-0.396	0.396
27	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
28	1	273.0	5.0	14000.00	581	2.238	1.248	1.248
29	5	355.6	6.3	14000.00	608	2.173	1.183	1.183
30	1	273.0	5.0	14000.00	516	2.155	1.165	1.165
31	5	355.6	6.3	14000.00	819	1.573	0.583	0.583
32	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
33	5	355.6	6.3	14000.00	927	0.886	-0.104	0.104
34	5	355.6	6.3	14000.00	177	1.357	0.367	0.367
35	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
36	5	355.6	6.3	14000.00	529	1.523	0.533	0.533
37	3	323.9	5.6	17204.65	302	2.271	1.281	1.281
38	3	323.9	5.6	17204.65	790	2.276	1.286	1.286
39	3	323.9	5.6	17204.65	597	1.227	0.237	0.237

40	3	323.9	5.6	17204.65	836	2.678	1.688	1.688
41	3	323.9	5.6	17204.65	876	1.985	0.995	0.995
42	12	508.0	10.0	17143.06	624	0.338	-0.652	0.652
43	12	508.0	10.0	17143.06	529	0.580	-0.410	0.410
44	12	508.0	10.0	17143.06	600	0.416	-0.574	0.574
45	12	508.0	10.0	17143.06	935	0.783	-0.207	0.207
46	12	508.0	10.0	17143.06	443	0.414	-0.576	0.576
47	12	508.0	10.0	17143.06	756	0.508	-0.482	0.482
48	12	508.0	10.0	17143.19	250	0.436	-0.554	0.554
49	12	508.0	10.0	17143.19	714	0.499	-0.491	0.491
50	12	508.0	10.0	17143.19	444	0.457	-0.533	0.533
51	12	508.0	10.0	17143.19	712	0.746	-0.244	0.244
52	12	508.0	10.0	17143.06	245	0.410	-0.580	0.580
53	12	508.0	10.0	17143.06	728	0.569	-0.421	0.421
54	3	323.9	5.6	17204.77	246	2.454	1.464	1.464
55	3	323.9	5.6	17204.65	714	1.905	0.915	0.915
56	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
57	5	355.6	6.3	14000.00	978	1.917	0.927	0.927
58	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
59	5	355.6	6.3	14000.00	819	1.464	0.474	0.474
60	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
61	5	355.6	6.3	14000.00	918	1.036	0.046	0.046
62	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
63	5	355.6	6.3	14000.00	927	1.613	0.623	0.623
64	3	323.9	5.6	17204.65	771	1.738	0.748	0.748
65	3	323.9	5.6	17204.65	636	1.678	0.688	0.688
66	3	323.9	5.6	17204.77	338	2.083	1.093	1.093
67	3	323.9	5.6	17204.65	450	2.766	1.776	1.776
68	12	508.0	10.0	14000.00	566	0.749	-0.241	0.241
69	12	508.0	10.0	14000.00	181	0.557	-0.433	0.433
70	12	508.0	10.0	17143.06	813	0.607	-0.383	0.383
71	12	508.0	10.0	17143.06	778	0.674	-0.316	0.316
72	12	508.0	10.0	17143.06	385	0.750	-0.240	0.240
73	12	508.0	10.0	17143.06	638	0.623	-0.367	0.367
74	12	508.0	10.0	17143.06	741	0.749	-0.241	0.241
75	12	508.0	10.0	17143.06	782	0.753	-0.237	0.237
76	9	457.0	8.0	10000.00	685	1.275	0.285	0.285
77	9	457.0	8.0	10000.00	1002	0.332	-0.658	0.658
78	9	457.0	8.0	10000.00	232	2.305	1.315	1.315
79	9	457.0	8.0	10000.00	579	1.622	0.632	0.632

80	9	457.0	8.0	10000.00	638	0.466	-0.524	0.524
81	9	457.0	8.0	10000.00	350	15.210	14.220	14.220
82	9	457.0	8.0	10000.00	124	0.654	-0.336	0.336
83	20	711.0	12.0	13653.85	704	0.644	-0.346	0.346
84	20	711.0	12.0	13653.85	181	0.345	-0.645	0.645
85	20	711.0	12.0	13653.85	137	1.085	0.095	0.095
86	20	711.0	12.0	13653.85	712	0.362	-0.628	0.628
87	20	711.0	12.0	13653.85	608	0.405	-0.585	0.585
88	20	711.0	12.0	13653.85	987	0.566	-0.424	0.424
89	9	457.0	8.0	10000.00	87	0.346	-0.644	0.644
90	9	457.0	8.0	10000.00	875	0.264	-0.726	0.726
91	9	457.0	8.0	10000.00	826	0.324	-0.666	0.666
92	1	273.0	5.0	10000.00	832	1.453	0.463	0.463
93	9	457.0	8.0	10000.21	246	0.374	-0.616	0.616
94	1	273.0	5.0	10000.00	223	1.331	0.341	0.341
95	9	457.0	8.0	10000.00	199	0.366	-0.624	0.624
96	20	711.0	12.0	13653.85	181	0.392	-0.598	0.598
97	20	711.0	12.0	13653.85	181	0.573	-0.417	0.417
98	20	711.0	12.0	13653.85	246	0.539	-0.451	0.451
99	20	711.0	12.0	13653.85	246	0.508	-0.482	0.482
100	20	711.0	12.0	13653.85	802	0.352	-0.638	0.638
101	20	711.0	12.0	13653.85	617	0.440	-0.550	0.550
102	3	323.9	5.6	17204.65	987	3.665	2.675	2.675
103	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
104	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
105	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
106	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
107	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
108	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
109	7	406.4	6.3	14000.00	-	-	Class 4	Class 4
110	5	355.6	6.3	14000.00	500	1.025	0.035	0.035
111	5	355.6	6.3	14000.00	615	0.941	-0.049	0.049
112	5	355.6	6.3	14000.00	358	1.060	0.070	0.070
113	5	355.6	6.3	14000.00	387	0.931	-0.059	0.059
114	1	273.0	5.0	14000.00	26	2.133	1.143	1.143
115	1	273.0	5.0	14000.00	185	2.127	1.137	1.137
116	5	355.6	6.3	14000.00	179	0.910	-0.080	0.080
117	3	323.9	5.6	17204.65	267	4.427	3.437	3.437
118	3	323.9	5.6	17204.77	826	2.250	1.260	1.260
119	7	406.4	6.3	19555.75	-	-	Class 4	Class 4

120	7	406.4	6.3	19555.75	-	-	Class 4	Class 4
121	7	406.4	6.3	19555.75	-	-	Class 4	Class 4
122	7	406.4	6.3	19555.75	-	-	Class 4	Class 4
123	7	406.4	6.3	19555.75	-	-	Class 4	Class 4
124	7	406.4	6.3	19555.75	-	-	Class 4	Class 4
125	12	508.0	10.0	17143.06	818	0.387	-0.603	0.603
126	12	508.0	10.0	17143.06	689	0.506	-0.484	0.484
127	12	508.0	10.0	17143.06	304	0.430	-0.560	0.560
128	12	508.0	10.0	17143.19	962	0.560	-0.430	0.430
129	12	508.0	10.0	17143.19	987	0.639	-0.351	0.351
130	12	508.0	10.0	17143.06	616	0.490	-0.500	0.500
131	3	323.9	5.6	17204.65	962	2.163	1.173	1.173
132	3	323.9	5.6	17204.65	826	2.162	1.172	1.172
133	3	323.9	5.6	17204.65	878	2.051	1.061	1.061
134	3	323.9	5.6	17204.65	8	3.786	2.796	2.796
135	12	508.0	10.0	14000.00	631	0.754	-0.236	0.236
136	7	406.4	6.3	19555.75	-	-	Class 4	Class 4
137	7	406.4	6.3	19555.75	-	-	Class 4	Class 4
138	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
139	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
140	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
141	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
142	23	813.0	12.0	14200.01	-	-	Class 4	Class 4
143	23	813.0	12.0	14200.01	-	-	Class 4	Class 4
144	23	813.0	12.0	14200.01	-	-	Class 4	Class 4
145	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
146	23	813.0	12.0	14200.01	-	-	Class 4	Class 4
147	23	813.0	12.0	14200.01	-	-	Class 4	Class 4
148	23	813.0	12.0	14200.01	-	-	Class 4	Class 4
149	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
150	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
151	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
152	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
153	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
154	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
155	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
156	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
157	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
158	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
159	23	813.0	12.0	14200.00	-	-	Class 4	Class 4

160	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
161	23	813.0	12.0	14200.00	-	-	Class 4	Class 4
162	12	508.0	10.0	14000.00	944	0.635	-0.355	0.355
163	12	508.0	10.0	14000.00	608	0.759	-0.231	0.231
164	12	508.0	10.0	14000.00	771	0.751	-0.239	0.239
165	12	508.0	10.0	14000.00	728	0.459	-0.531	0.531
166	5	355.6	6.3	14000.00	529	0.804	-0.186	0.186
167	3	323.9	5.6	17204.65	858	2.038	1.048	1.048
SUM							**17.280**	**91.192**
MIN							**-0.734**	**0.035**
MAX							**14.220**	**14.220**
MEAN (excluding classes 4)							**0.143**	**0.754**

Table 40. Cross-sections and utilization rate of the three-dimensional frame starting model. The deviation between the beam utilization rate and the target rate ū, which is 0.99, is reported. The utilization rate for class 4 sections was not reported.

The optimization tool provided the model in Figure 100 and Figure 101 as output. Information on the size of the cross sections, their utilization rates, and deviation from the target is given in Table 41. It is observed that the average absolute difference between actual and target utilization rates is close to zero (0.092), and that the range of these differences (with sign) is from a minimum of -0.289 to a maximum of 1.544. Analyzing the utilization rates of the various beams shows that only three of them have utilization rates greater than 1.4, that is, they have a deviation greater than 0.4. Excluding these three, the average absolute deviation further approaches zero, that is, it becomes 0.078, while the minimum and maximum values with sign become 0.013 and 0.328, respectively.

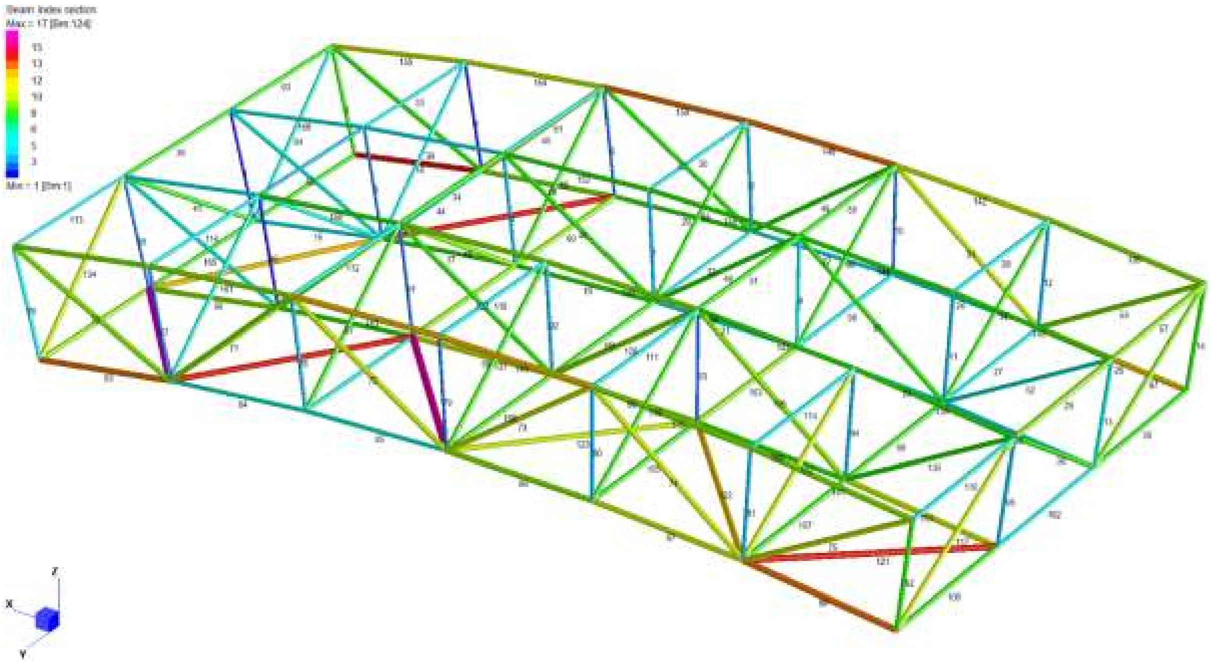


Figure 100. Indices of the sections of the optimized three-dimensional model beams.

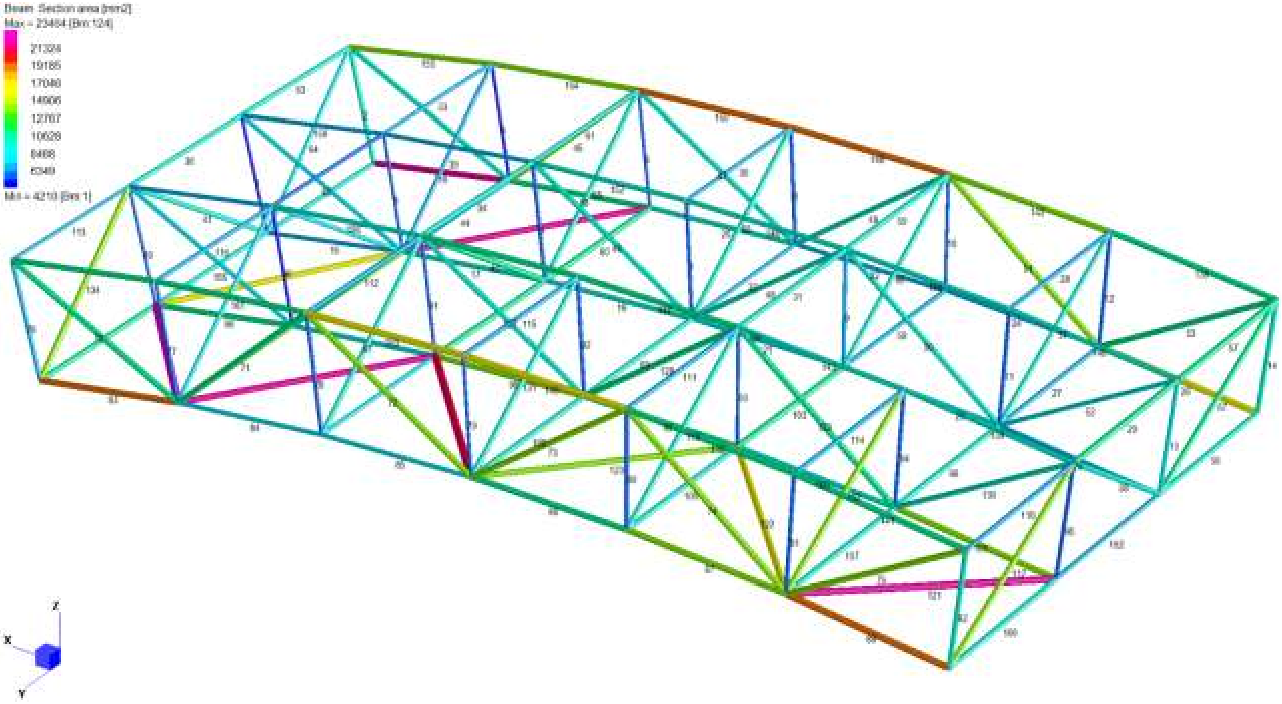


Figure 101. Area of the sections of the optimized three-dimensional model beams.

Beam	ID Section	Diam [mm]	Thk [mm]	Length [mm]	Result case max	u_{max}	$u_{max}-\bar{u}$	$\lvert u_{max}-\bar{u}\rvert$
1	1	273.0	5.0	10000.00	723	1.013	0.023	0.023
2	8	406.4	8.0	10000.00	529	0.963	-0.027	0.027
3	2	273.0	6.3	10000.00	36	2.067	1.077	1.077
4	1	273.0	5.0	10000.00	229	0.701	-0.289	0.289
5	3	323.9	5.6	10000.00	254	0.937	-0.053	0.053
6	2	273.0	6.3	10000.00	868	1.034	0.044	0.044
7	3	323.9	5.6	10000.00	436	0.868	-0.122	0.122
8	3	323.9	5.6	10000.00	366	0.949	-0.041	0.041
9	4	323.9	7.1	10000.21	530	0.879	-0.111	0.111
10	2	273.0	6.3	10000.21	471	1.050	0.060	0.060
11	3	323.9	5.6	10000.00	819	0.824	-0.166	0.166
12	3	323.9	5.6	10000.00	787	1.003	0.013	0.013
13	6	355.6	7.1	10000.00	819	0.862	-0.128	0.128
14	8	406.4	8.0	10000.00	652	1.033	0.043	0.043
15	5	355.6	6.3	13653.85	772	0.976	-0.014	0.014
16	14	508.0	14.2	13653.85	33	0.929	-0.061	0.061
17	8	406.4	8.0	13653.85	70	0.910	-0.080	0.080
18	9	457.0	8.0	13653.85	728	2.534	1.544	1.544
19	8	406.4	8.0	13653.85	798	0.803	-0.187	0.187
20	8	406.4	8.0	13653.85	903	0.906	-0.084	0.084
21	6	355.6	7.1	13653.85	212	1.123	0.133	0.133
22	5	355.6	6.3	13653.85	771	1.125	0.135	0.135

23	6	355.6	7.1	13653.85	798	0.992	0.002	0.002
24	4	323.9	7.1	13653.85	397	0.929	-0.061	0.061
25	4	323.9	7.1	13653.85	438	1.043	0.053	0.053
26	12	508.0	10.0	13653.85	33	1.060	0.070	0.070
27	5	355.6	6.3	14000.00	334	1.017	0.027	0.027
28	5	355.6	6.3	14000.00	399	0.989	-0.001	0.001
29	9	457.0	8.0	14000.00	978	0.880	-0.110	0.110
30	4	323.9	7.1	14000.00	246	1.009	0.019	0.019
31	8	406.4	8.0	14000.00	819	0.954	-0.036	0.036
32	4	323.9	7.1	14000.00	242	1.065	0.075	0.075
33	4	323.9	7.1	14000.00	181	0.828	-0.162	0.162
34	8	406.4	8.0	14000.00	177	0.858	-0.132	0.132
35	5	355.6	6.3	14000.00	503	0.976	-0.014	0.014
36	8	406.4	8.0	14000.00	297	0.883	-0.107	0.107
37	8	406.4	8.0	17204.65	302	1.114	0.124	0.124
38	8	406.4	8.0	17204.65	872	0.888	-0.102	0.102
39	5	355.6	6.3	17204.65	597	0.990	0.000	0.000
40	9	457.0	8.0	17204.65	836	1.005	0.015	0.015
41	8	406.4	8.0	17204.65	810	0.951	-0.039	0.039
42	6	355.6	7.1	17143.06	624	1.138	0.148	0.148
43	8	406.4	8.0	17143.06	339	1.170	0.180	0.180
44	6	355.6	7.1	17143.06	518	1.221	0.231	0.231
45	10	457.0	10.0	17143.06	33	1.047	0.057	0.057
46	8	406.4	8.0	17143.06	404	0.934	-0.056	0.056
47	8	406.4	8.0	17143.06	756	1.007	0.017	0.017
48	8	406.4	8.0	17143.19	250	1.015	0.025	0.025
49	8	406.4	8.0	17143.19	714	1.144	0.154	0.154
50	8	406.4	8.0	17143.19	444	1.049	0.059	0.059
51	10	457.0	10.0	17143.19	712	0.955	-0.035	0.035
52	6	355.6	7.1	17143.06	685	1.204	0.214	0.214
53	9	457.0	8.0	17143.06	380	1.061	0.071	0.071
54	8	406.4	8.0	17204.77	246	1.147	0.157	0.157
55	8	406.4	8.0	17204.65	865	1.024	0.034	0.034
56	8	406.4	8.0	14000.00	741	1.016	0.026	0.026
57	9	457.0	8.0	14000.00	978	0.897	-0.093	0.093
58	5	355.6	6.3	14000.00	771	0.990	0.000	0.000
59	8	406.4	8.0	14000.00	819	0.979	-0.011	0.011
60	9	457.0	8.0	14000.00	291	1.143	0.153	0.153
61	6	355.6	7.1	14000.00	622	1.056	0.066	0.066
62	8	406.4	8.0	14000.00	1002	0.997	0.007	0.007

63	8	406.4	8.0	14000.00	529	0.892	-0.098	0.098
64	6	355.6	7.1	17204.65	715	1.289	0.299	0.299
65	8	406.4	8.0	17204.65	636	0.889	-0.101	0.101
66	8	406.4	8.0	17204.77	868	1.101	0.111	0.111
67	9	457.0	8.0	17204.65	450	0.886	-0.104	0.104
68	8	406.4	8.0	14000.00	608	0.851	-0.139	0.139
69	8	406.4	8.0	14000.00	123	0.934	-0.056	0.056
70	9	457.0	8.0	17143.06	813	1.023	0.033	0.033
71	9	457.0	8.0	17143.06	1002	1.020	0.030	0.030
72	10	457.0	10.0	17143.06	385	0.925	-0.065	0.065
73	10	457.0	10.0	17143.06	638	0.927	-0.063	0.063
74	10	457.0	10.0	17143.06	741	0.920	-0.070	0.070
75	10	457.0	10.0	17143.06	782	0.970	-0.020	0.020
76	5	355.6	6.3	10000.00	114	1.104	0.114	0.114
77	2	273.0	6.3	10000.00	788	1.083	0.093	0.093
78	3	323.9	5.6	10000.00	108	0.977	-0.013	0.013
79	2	273.0	6.3	10000.00	347	0.947	-0.043	0.043
80	3	323.9	5.6	10000.00	638	0.897	-0.093	0.093
81	3	323.9	5.6	10000.00	168	0.827	-0.163	0.163
82	8	406.4	8.0	10000.00	124	0.744	-0.246	0.246
83	13	508.0	12.0	13653.85	704	0.959	-0.031	0.031
84	6	355.6	7.1	13653.85	88	0.997	0.007	0.007
85	6	355.6	7.1	13653.85	523	1.014	0.024	0.024
86	9	457.0	8.0	13653.85	608	0.908	-0.082	0.082
87	10	457.0	10.0	13653.85	608	0.870	-0.120	0.120
88	13	508.0	12.0	13653.85	987	0.943	-0.047	0.047
89	2	273.0	6.3	10000.00	87	1.043	0.053	0.053
90	1	273.0	5.0	10000.00	875	1.005	0.015	0.015
91	2	273.0	6.3	10000.00	28	1.039	0.049	0.049
92	3	323.9	5.6	10000.00	832	1.126	0.136	0.136
93	2	273.0	6.3	10000.21	147	0.955	-0.035	0.035
94	3	323.9	5.6	10000.00	223	0.984	-0.006	0.006
95	3	323.9	5.6	10000.00	199	0.839	-0.151	0.151
96	9	457.0	8.0	13653.85	689	0.902	-0.088	0.088
97	8	406.4	8.0	13653.85	298	0.954	-0.036	0.036
98	10	457.0	10.0	13653.85	338	0.978	-0.012	0.012
99	9	457.0	8.0	13653.85	29	0.963	-0.027	0.027
100	8	406.4	8.0	13653.85	888	0.932	-0.058	0.058
101	10	457.0	10.0	13653.85	29	0.860	-0.130	0.130
102	10	457.0	10.0	17204.65	987	1.019	0.029	0.029

103	9	457.0	8.0	14000.00	688	1.008	0.018	0.018
104	6	355.6	7.1	14000.00	523	1.071	0.081	0.081
105	8	406.4	8.0	14000.00	70	0.809	-0.181	0.181
106	8	406.4	8.0	14000.00	29	0.941	-0.049	0.049
107	8	406.4	8.0	14000.00	597	1.223	0.233	0.233
108	8	406.4	8.0	14000.00	123	1.342	0.352	0.352
109	10	457.0	10.0	14000.00	689	0.887	-0.103	0.103
110	5	355.6	6.3	14000.00	500	0.992	0.002	0.002
111	5	355.6	6.3	14000.00	101	0.977	-0.013	0.013
112	5	355.6	6.3	14000.00	358	1.026	0.036	0.036
113	4	323.9	7.1	14000.00	387	1.021	0.031	0.031
114	5	355.6	6.3	14000.00	26	0.997	0.007	0.007
115	5	355.6	6.3	14000.00	185	0.952	-0.038	0.038
116	5	355.6	6.3	14000.00	179	0.948	-0.042	0.042
117	10	457.0	10.0	17204.65	267	0.993	0.003	0.003
118	8	406.4	8.0	17204.77	385	1.102	0.112	0.112
119	16	610.0	11.0	19555.75	246	1.355	0.365	0.365
120	14	508.0	14.2	19555.75	720	0.984	-0.006	0.006
121	14	508.0	14.2	19555.75	617	0.990	0.000	0.000
122	12	508.0	10.0	19555.75	810	0.906	-0.084	0.084
123	10	457.0	10.0	19555.75	158	1.021	0.031	0.031
124	17	610.0	12.5	19555.75	728	1.032	0.042	0.042
125	6	355.6	7.1	17143.06	818	1.141	0.151	0.151
126	8	406.4	8.0	17143.06	689	0.942	-0.048	0.048
127	8	406.4	8.0	17143.06	385	0.950	-0.040	0.040
128	9	457.0	8.0	17143.19	798	1.014	0.024	0.024
129	9	457.0	8.0	17143.19	987	1.009	0.019	0.019
130	8	406.4	8.0	17143.06	616	0.997	0.007	0.007
131	8	406.4	8.0	17204.65	120	1.176	0.186	0.186
132	8	406.4	8.0	17204.65	826	1.021	0.031	0.031
133	6	355.6	7.1	17204.65	586	1.260	0.270	0.270
134	10	457.0	10.0	17204.65	8	0.992	0.002	0.002
135	10	457.0	10.0	14000.00	771	0.958	-0.032	0.032
136	14	508.0	14.2	19555.75	241	0.972	-0.018	0.018
137	12	508.0	10.0	19555.75	688	0.894	-0.096	0.096
138	9	457.0	8.0	14200.00	33	1.213	0.223	0.223
139	8	406.4	8.0	14200.00	1002	0.850	-0.140	0.140
140	8	406.4	8.0	14200.00	259	0.909	-0.081	0.081
141	9	457.0	8.0	14200.00	987	0.991	0.001	0.001
142	10	457.0	10.0	14200.01	821	0.888	-0.102	0.102

143	8	406.4	8.0	14200.01	126	1.017	0.027	0.027
144	8	406.4	8.0	14200.01	259	0.936	-0.054	0.054
145	9	457.0	8.0	14200.00	987	0.981	-0.009	0.009
146	13	508.0	12.0	14200.01	33	0.966	-0.024	0.024
147	9	457.0	8.0	14200.01	449	0.920	-0.070	0.070
148	8	406.4	8.0	14200.01	33	1.022	0.032	0.032
149	12	508.0	10.0	14200.00	962	1.051	0.061	0.061
150	13	508.0	12.0	14200.00	33	0.969	-0.021	0.021
151	9	457.0	8.0	14200.00	97	0.912	-0.078	0.078
152	8	406.4	8.0	14200.00	33	0.996	0.006	0.006
153	12	508.0	10.0	14200.00	741	0.927	-0.063	0.063
154	10	457.0	10.0	14200.00	529	0.911	-0.079	0.079
155	10	457.0	10.0	14200.00	529	0.941	-0.049	0.049
156	8	406.4	8.0	14200.00	15	0.832	-0.158	0.158
157	5	355.6	6.3	14200.00	737	0.999	0.009	0.009
158	5	355.6	6.3	14200.00	821	1.038	0.048	0.048
159	5	355.6	6.3	14200.00	723	0.970	-0.020	0.020
160	9	457.0	8.0	14200.00	878	0.934	-0.056	0.056
161	9	457.0	8.0	14200.00	741	0.958	-0.032	0.032
162	5	355.6	6.3	14000.00	21	1.648	0.658	0.658
163	9	457.0	8.0	14000.00	21	1.096	0.106	0.106
164	9	457.0	8.0	14000.00	631	1.058	0.068	0.068
165	8	406.4	8.0	14000.00	181	0.812	-0.178	0.178
166	3	323.9	5.6	14000.00	783	0.989	-0.001	0.001
167	6	355.6	7.1	17204.65	858	1.318	0.328	0.328
SUM							**3.238**	**15.998**
MIN							**-0.289**	**0.000**
MAX							**1.544**	**1.544**
MEAN							**0.019**	**0.096**

Table 41. Results of the optimization of the beams of the three-dimensional frame

	$u_{max}-\bar{u}$	$\lvert u_{max}-\bar{u}\rvert$
SUM	-0.041	12.719
MIN	-0.289	0.013
MAX	0.365	0.328
MEAN	0.000	0.078

Table 42. Overall deviation values between actual and target utilization rates of the optimized three-dimensional frame beams, excluding beams with utilization rates greater than 1.4 (i.e., beams number 3, 18, and 108)

5.3.4 Additional training

Analyzing the optimization results reported in the paragraph 5.3.3 , it is observed that the maximum utilization rate of the beams deviates from the desired target rate. To improve the results, it is proposed to continue training the network during the optimization process as well, adding a loss function that considers the difference between the utilization rate of the beams with the cross-sectional area provided as output by the network and the target utilization rate. Therefore, each time the *MLPsectionOpt* network is used during the optimization process, an update of the parameters is made using a loss function (*Loss1*) that evaluates the deviation between the beam utilization rate with the section output from the network and the target rate. After that, the network is further trained with the database previously used during the training phase and using the Cross Entropy loss (*Loss2*). The entire additional training process is illustrated in Figure 102.

Two different approaches for calculating the deviation loss function between the utilization rate and the target were evaluated. They are presented in the following sections.

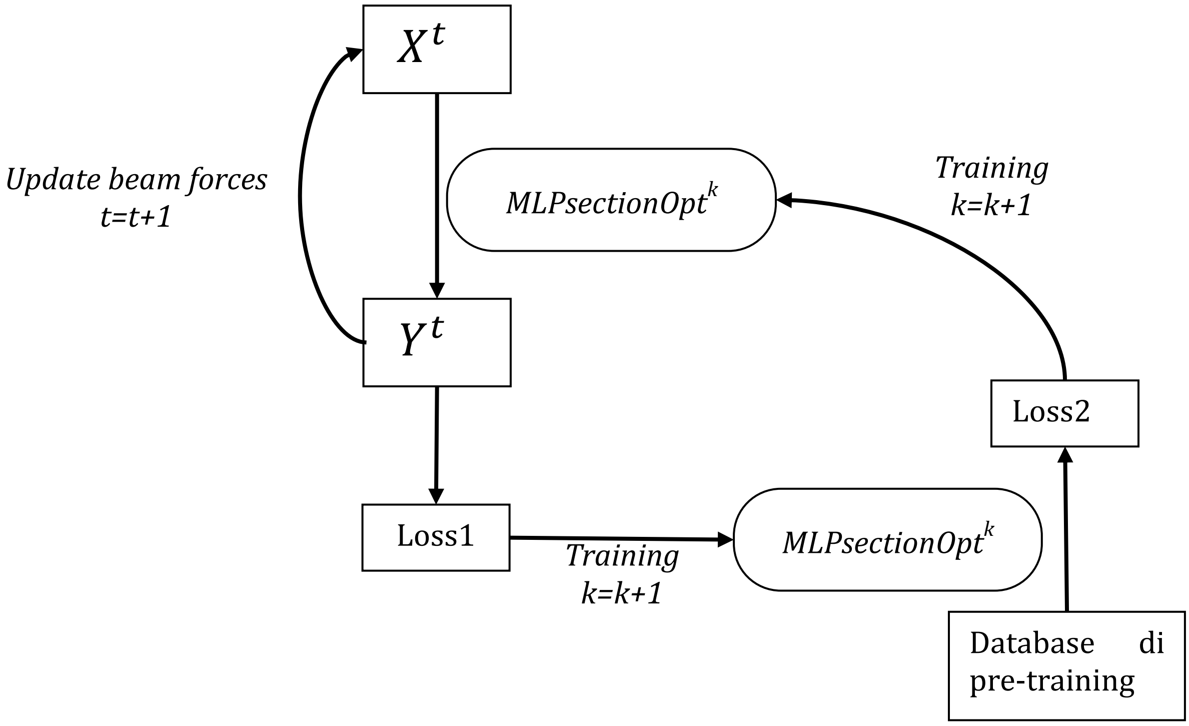


Figure 102. Workflow for the additional training of MLPsectionOpt

5.3.4.1 Approach A

For each load case, the network returns as output a matrix *Y* [*b* x *ns*], with *b* equal to the number of beams and *ns* the number of section classes. The softmax function is applied in each row of the matrix, resulting in a matrix $\bar{Y}$ in which in each row the values are between 0 and 1 and their sum is 1.

$$Y = [b\ x\ n_s] = \begin{bmatrix} y_{11} & \cdots & y_{1n_s} \\ \cdots & \cdots & \cdots \\ y_{b1} & \cdots & y_{bn_s} \end{bmatrix} \tag{5.64}$$

$$\bar{Y} = [b\ x\ n_s] = \begin{bmatrix} \bar{y}_{11} & \cdots & \bar{y}_{1n_s} \\ \cdots & \cdots & \cdots \\ \bar{y}_{b1} & \cdots & \bar{y}_{bn_s} \end{bmatrix} \qquad with\ \bar{y}_{ij} = \frac{e^{y_{ij}}}{\sum_{k=1}^{n_s} e^{y_{ik}}} \tag{5.65}$$

For each beam, the utilization rate that would occur with each section of the database is calculated, and the absolute difference between the obtained value and the target is used to compose the *V* matrix [*b* x *ns*].

$$V = [b\ x\ n_s] = \begin{bmatrix} v_{11} & \cdots & v_{1n_s} \\ \cdots & \cdots & \cdots \\ v_{b1} & \cdots & v_{bn_s} \end{bmatrix} \tag{5.66}$$

$$v_{ij} = (\max(u_{ij} - \bar{u}, 0)) * \bar{f} + \min(u_{ij} - \bar{u}, 0)\,(-1) \tag{5.67}$$

In the above equation, u_{ij} is denoted as the utilization rate of the *i*-th beam with *j*-th section, $\bar{u}$ is the target utilization rate, while $\bar{f}$ is a coefficient to penalize cases of under-sizing versus over-sizing.

The Hadamard product between the matrices $\bar{Y}$ and V is then calculated.

$$\bar{A} = [b\ x\ n_s] = \begin{bmatrix} \bar{a}_{11} & \cdots & \bar{a}_{1c} \\ \cdots & \cdots & \cdots \\ \bar{a}_{b1} & \cdots & \bar{a}_{bn_s} \end{bmatrix} = \bar{Y} \cdot V \tag{5.68}$$

The loss function is given by the summation of the elements of matrix $\bar{A}$:

$$loss = \sum_{ij} \bar{a}_{ij} \tag{5.69}$$

The gradient that allows the model parameters to be updated to reduce the deviation between the utilization rate with the section being output from the model and the target utilization rate corresponds to the gradient between the loss function and the matrix Y:

$$\left(\frac{d\ loss}{d\ y}\right)_{hk} = \sum_{j} v_{hj} * \frac{d\ \bar{y}_{hj}}{d\ y_{hk}} \tag{5.70}$$

Using this approach allows us to take advantage of the automatic differentiation operators of the library used to write the network without therefore having to create the ad hoc gradient function between matrices $\bar{A}$ and Y.

5.3.4.2 Approach B

The sections obtained in output from the model are used to calculate the utilization rate of each beam. The utilization rate can be amplified with a coefficient $\bar{f}$ to penalize under-sizing versus over-sizing. These values are entered into a vector u $[n_{LC}b \text{ x } 1]$, which is used in the loss calculation with the following function:

$$loss = MAE(u, \bar{u}) \tag{5.71}$$

where $\bar{u}$ is the vector with the target utilization rate, b is the number of beams and n_{LC} is the number of load cases.

It is necessary to define the gradient between utilization rates and model output in order to perform back-propagation and train the network. This gradient is expressed by the following matrix:

$$G = [b\ x\ n_{LC}] = \begin{pmatrix} g_{11} & \cdots & g_{1n_{LC}} \\ \cdots & \cdots & \cdots \\ g_{b1} & \cdots & g_{bn_{LC}} \end{pmatrix} \tag{5.72}$$

The entries of G matrix are calculated as follows:

$$g_{ij} = \frac{u\big(t_{\min\,(j+1,c)}, s_i, l_i\big) - u\big(t_{\max\,(j-1,1)}, s_i, l_i\big)}{\mathcal{A}\big(t_{\min(j+1,c)}\big) - \mathcal{A}\big(t_{\max(j-1,0)}\big)} * y_{ij} \tag{5.73}$$

where

t_k is the k-th section of the database, with $k = 1, \dots, c$

s_i are the forces on the i-th beam

l_i is the length of the i-th beam

$u(t_k, s_i, l_i)$ is the utilization rate calculated using the forces and the length of the i-th beam and the k-th section of the database

$\mathcal{A}(t_k)$ is the area of the k-th section, with $k = 1, \dots, c$

y_{ij} is the element at row i and column j of the matrix Y

5.3.5 Optimization results with additional training

Since multiple loss functions are involved, and moreover calculated on different databases, special attention must be paid to the choice and setting of optimizers with which to update model parameters during training. In fact, updating the network based on the gradients of one of the two loss functions will cause a shift in the values of the network parameters in favor of that loss, potentially leading it toward a worst-case configuration for the other loss function. The goal, therefore, is to identify optimizers that will allow the network to be trained toward a configuration that provides outputs closer to the target but at the same time without departing too far from the performance obtained with pretraining. Specifically, it was observed that in order to obtain benefits from additional training, higher learning rate values should be adopted with the Loss1 optimizer than with the Loss2 optimizer. The reason for this behavior is due to the larger database size used for Loss2 and the fact that the model was previously trained with that loss through that database.

5.3.5.1 Two-dimensional Frame

Through the Approach A, the best results were obtained with Adam's algorithm setting a learning rate of 0.001 for Loss1 and 0.0005 for Loss2. The results are shown in the Table 43. The mean absolute deviation between the utilization rate and the target decreased from the result achieved without additional training, from 0.132 to 0.120. The same learning rate values also provided the best results with Approach B. However, in this case the additional learning did not provide a better solution than without it. In fact, the mean deviation between the utilization rate and the target was 0.145 with Approach B versus 0.132 obtained without the additional training. The results for Approach B are shown in Table 44.

Beam	Length [mm]	ID Sect.	ID Sect. – ID Sect.$_{\text{NO TRAIN}}$	Diam [mm]	Thk [mm]	u_{max}	u_{max}-$\bar{u}$	|u_{max}-$\bar{u}$|
1	2500	7	0	323.9	5.6	1.092	0.102	0.102
2	1500	4	-1	177.8	4.0	0.863	-0.127	0.127
3	4000	5	0	219.1	5.0	0.799	-0.191	0.191
4	4000	5	0	219.1	5.0	0.801	-0.189	0.189
5	1500	6	1	273.0	5.0	0.960	-0.030	0.030
6	2500	7	0	323.9	5.6	1.228	0.238	0.238
7	4000	2	0	101.6	3.0	0.908	-0.082	0.082
8	4000	2	0	101.6	3.0	1.108	0.118	0.118
9	1500	3	0	139.7	4.0	0.843	-0.147	0.147
10	4272	4	0	177.8	4.0	0.966	-0.024	0.024
11	4272	4	0	177.8	4.0	1.061	0.071	0.071
SUM							**-0.259**	**1.320**
MIN							**-0.191**	**0.024**
MAX							**0.238**	**0.238**
MEAN							**-0.024**	**0.120**
NUMBER OF UNDER-SIZED BEAMS							**4**	

Table 43. Two-dimensional frame optimized with additional training with approach A (lr1=0.001 and lr2=0.0005)

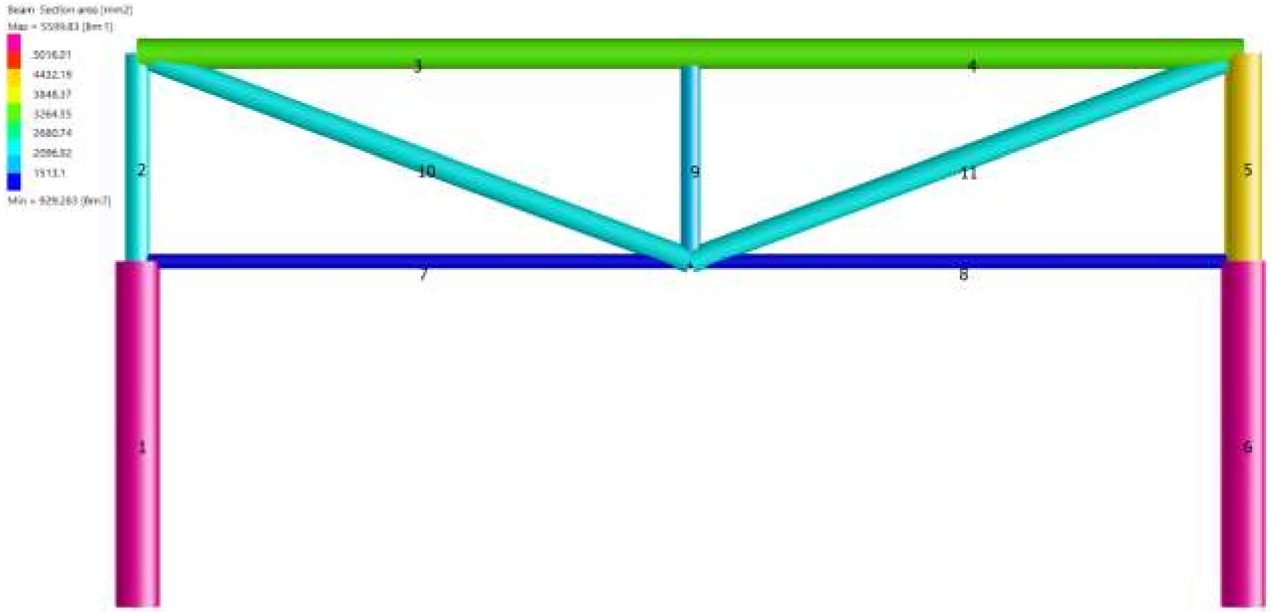


Figure 103. Model of two-dimensional frame optimized with additional training with Approach A (lr1=0.001 and lr2=0.0005)

Beam	Length [mm]	ID Sect.	ID Sect. – ID Sect.$_{\text{NO TRAIN}}$	Diam [mm]	Thk [mm]	u_{max}	u_{max}-$\bar{u}$	$\lvert u_{max}$-$\bar{u}\rvert$
1	2500	7	0	323.9	5.6	1.157	0.167	0.167
2	1500	4	-1	177.8	4.0	0.968	-0.022	0.022
3	4000	4	-1	177.8	4.0	1.128	0.138	0.138
4	4000	4	-1	177.8	4.0	1.118	0.128	0.128
5	1500	5	0	219.1	5.0	0.900	-0.090	0.090
6	2500	7	0	323.9	5.6	1.242	0.252	0.252
7	4000	2	0	101.6	3.0	1.006	0.016	0.016
8	4000	2	0	101.6	3.0	1.063	0.073	0.073
9	1500	3	0	139.7	4.0	0.858	-0.132	0.132
10	4272	7	3	323.9	5.6	0.703	-0.287	0.287
11	4272	7	3	323.9	5.6	0.705	-0.285	0.285
SUM							**-0.042**	**1.590**
MIN							**-0.287**	**0.016**
MAX							**0.252**	**0.287**
MEAN							**-0.004**	**0.145**
NUMBER OF UNDER-SIZED BEAMS							**6**	

Table 44. Two-dimensional frame optimized with additional training with Approach B (lr1=0.001 and lr2=0.0005)

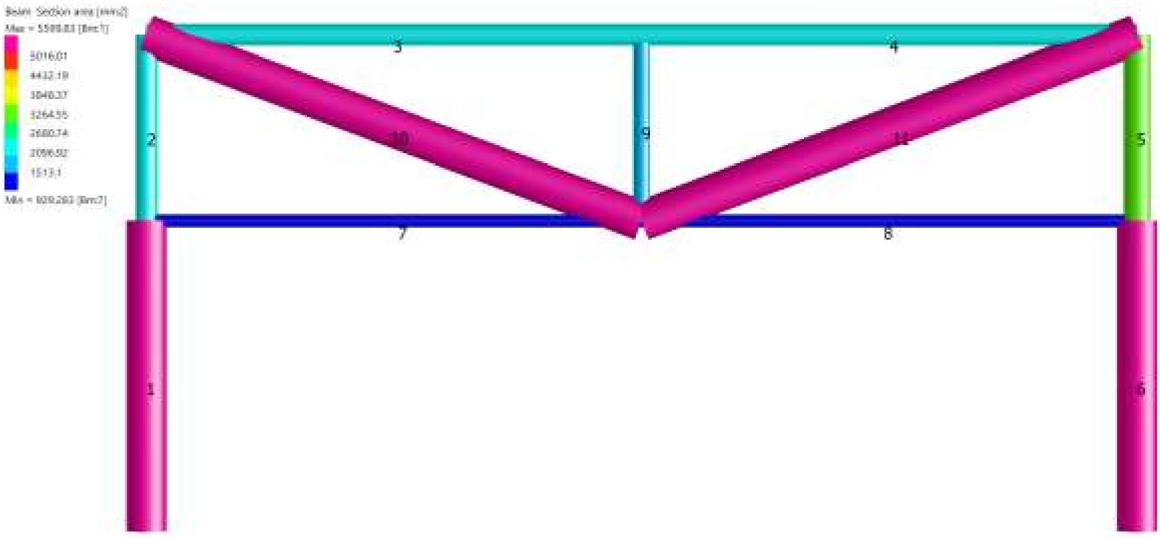


Figure 104. Model of two-dimensional frame optimized with additional training with Approach B (lr1=0.001 and lr2=0.0005)

The use of additional training has the advantage of reducing the probability of under-sizing, a condition obviously absolutely to be avoided, through the use of values greater than 1 for the penalty coefficient multiplying utilization rates above the target. The following tables and pictures show the results obtained by imposing the penalty coefficient of 10. It is observed that the number of under-sized beams is zero, but the average deviation between actual and target utilization rates has increased. For better results, the existence of an optimizer that guides additional training more accurately can be investigated.

Beam	Length [mm]	ID Sect.	ID Sect. – ID Sect.$_{\text{NO TRAIN}}$	Diam [mm]	Thk [mm]	u_{max}	u_{max}-ū	|u_{max}-ū|
1	2500	7	0	323.9	5.6	0.961	-0.029	0.029
2	1500	7	2	323.9	5.6	0.624	-0.366	0.366
3	4000	7	2	323.9	5.6	0.744	-0.246	0.246
4	4000	7	2	323.9	5.6	0.744	-0.246	0.246
5	1500	7	2	323.9	5.6	0.717	-0.273	0.273
6	2500	7	0	323.9	5.6	0.980	-0.010	0.010
7	4000	3	1	139.7	4.0	0.446	-0.544	0.544
8	4000	3	1	139.7	4.0	0.395	-0.595	0.595
9	1500	6	3	273.0	5.0	0.300	-0.690	0.690
10	4272	5	1	219.1	5.0	0.517	-0.473	0.473
11	4272	5	1	219.1	5.0	0.516	-0.474	0.474
SUM							**-3.946**	**3.946**
MIN							**-0.690**	**0.010**
MAX							**-0.010**	**0.690**
MEAN							**-0.359**	**0.359**
NUMBER OF UNDER-SIZED BEAMS							**0**	

Table 45. Two-dimensional frame optimized with additional training with approach A and penalty coefficient equal to 10 (lr1=0.003 and lr2=0.005)

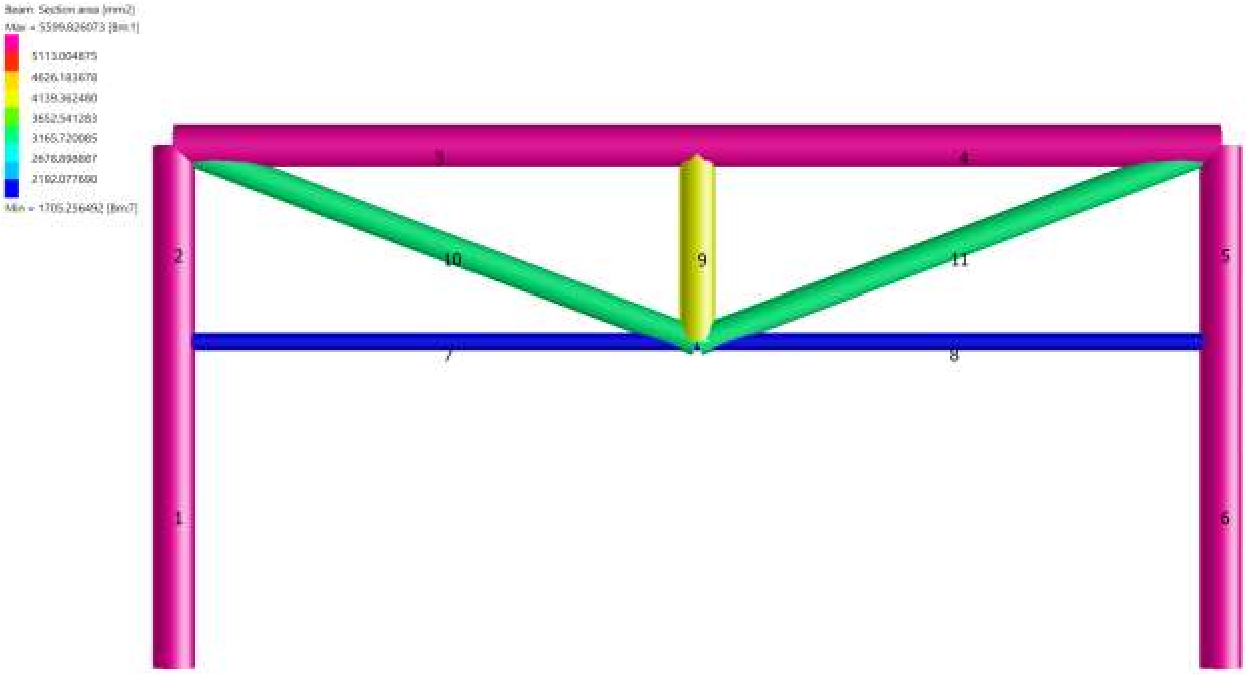


Figure 105. Model of two-dimensional frame optimized with additional training with Approach A and penalty coefficient equal to 10 (lr1=0.003 and lr2=0.005)

Beam	**Length [mm]**	**ID Sect.**	**ID Sect. – ID Sect.$_{\text{NO TRAIN}}$**	**Diam [mm]**	**Thk [mm]**	**u_{max}**	**u_{max}-$\bar{u}$**	**$\lvert u_{max}$-$\bar{u}\rvert$**
1	2500	7	0	323.9	5.6	0.954	-0.036	0.036
2	1500	7	+2	323.9	5.6	0.613	-0.377	0.377
3	4000	7	+2	323.9	5.6	0.683	-0.307	0.307
4	4000	7	+2	323.9	5.6	0.811	-0.179	0.179
5	1500	7	+2	323.9	5.6	0.802	-0.188	0.188
6	2500	7	0	323.9	5.6	0.966	-0.024	0.024
7	4000	4	+2	177.8	4.0	0.322	-0.668	0.668
8	4000	4	+2	177.8	4.0	0.341	-0.649	0.649
9	1500	7	+4	323.9	5.6	0.355	-0.635	0.635
10	4272	7	+3	323.9	5.6	0.488	-0.502	0.502
11	4272	4	0	177.8	4.0	0.754	-0.236	0.236
SUM							**-3.801**	**3.801**
MIN							**-0.668**	**0.024**
MAX							**-0.024**	**0.668**
MEAN							**-0.346**	**0.346**
NUMBER OF UNDER-SIZED BEAMS							**0**	

Table 46. Two-dimensional frame optimized with additional training with Approach B and penalty coefficient equal to 10 (lr1=0.001 and lr2=0.0005)

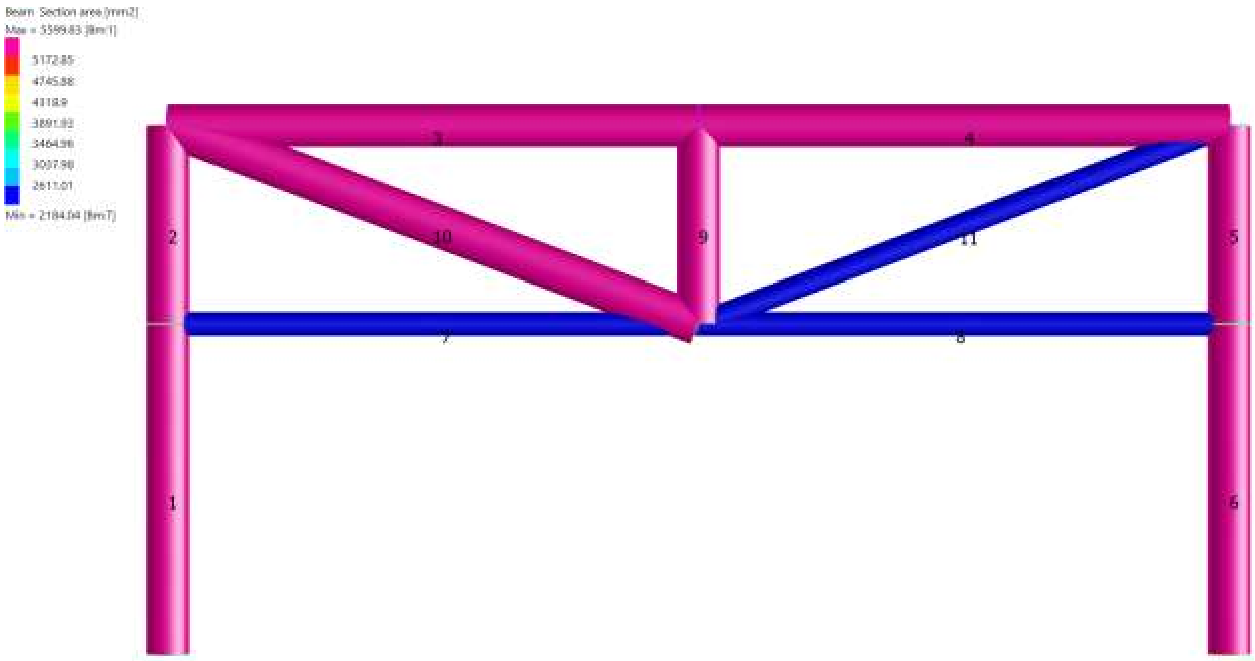


Figure 106. Model of two-dimensional frame optimized with additional training with Approach B and penalty coefficient equal to 10 (lr1=0.001 and lr2=0.0005)

Three-dimensional Frame

The additional training also had beneficial effects in the optimization of the three-dimensional model. Specifically, Approach A was applied without penalty for under-sizing and adopting the same learning rates used for two-dimensional model optimization (lr1=0.001 and lr2=0.0005). Analyzing the results, we observe that the average absolute deviation between the current utilization rate and the target has decreased from 0.096 to 0.082 and that the number of beams whose utilization rate exceeds 1.4 has decreased from three to one. Excluding this beam, the deviation takes on values between -0.287 and 0.388, with an absolute mean value of 0.073. The results are shown in Table 47 and Table 48 while the distributions of sections and corresponding areas in the output model are in Figure 107 and Figure 108.

Additional training using Approach B, on the other hand, led to worse results than the previous approach. As can be seen from Table 49, in which the results are shown, there are some entities with high utilization rates. Specifically, four beams have a utilization rate greater than 2. Excluding these elements, the results provided by Approach B are similar to those of Approach A. The presence of elements with very high utilization rate compared to the other approach may indicate a greater difficulty on the part of Approach B in taking into account nonlinearities between utilization rate and section area. The distribution in the structure of beam sections and corresponding areas obtained in output from the model with additional training B are shown in Figure 109 and in Figure 110.

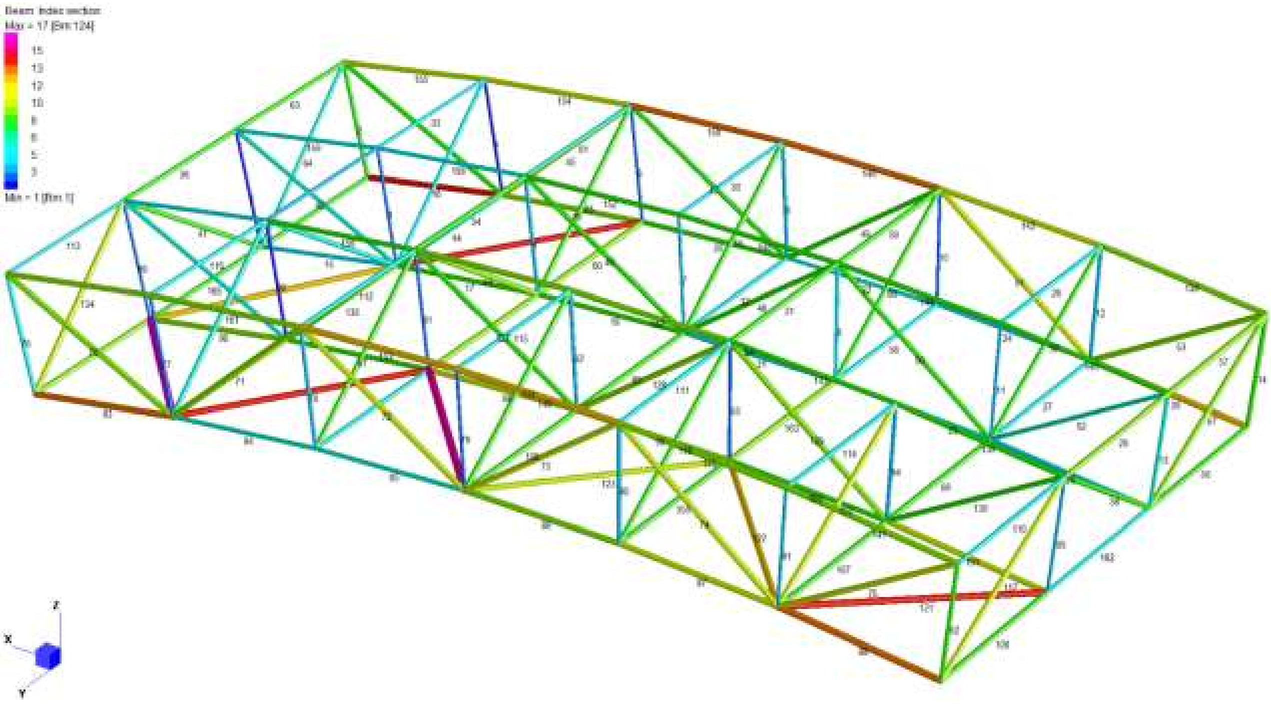


Figure 107. Indices of the sections of the optimized three-dimensional model beams obtained using the additional training (Approach A)

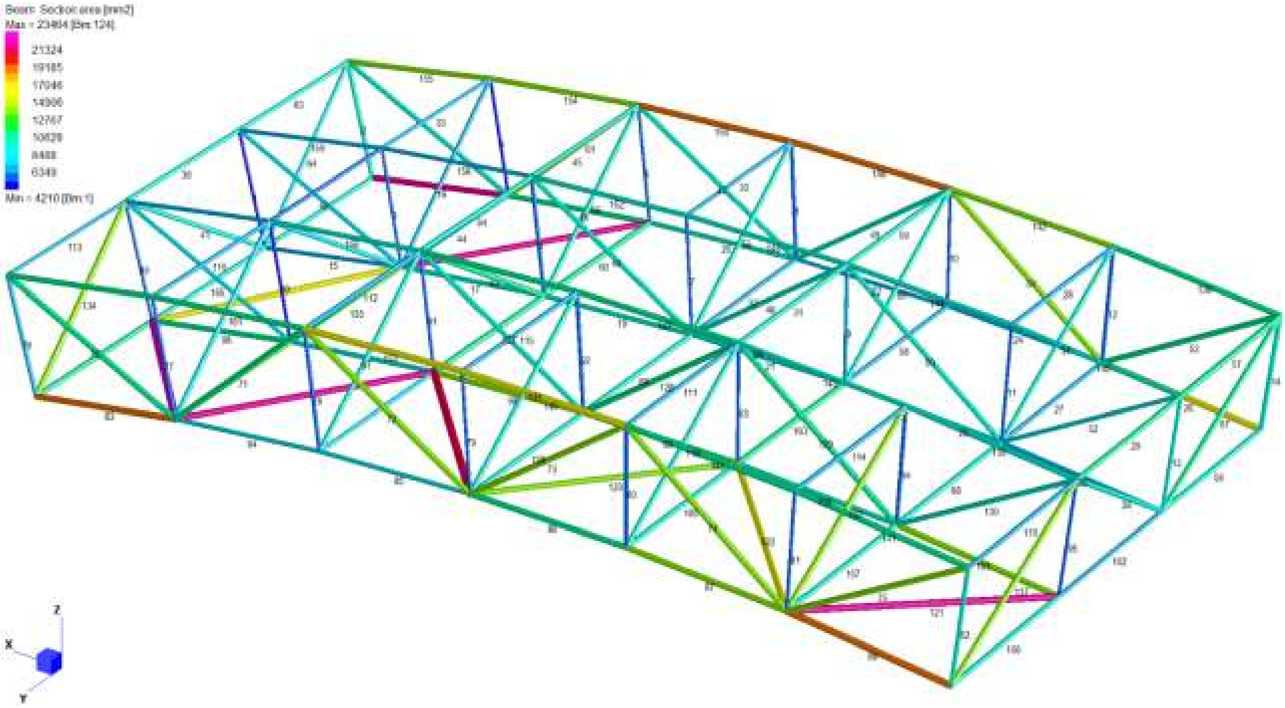


Figure 108. Area of the sections of the optimized three-dimensional model beams obtained using the additional training (Approach A)

Beam	ID Sect.	ID Sect. – ID Sect.$_{NO\ TRAIN}$	Diam [mm]	Thk [mm]	Length [mm]	Result case max	u_{max}	$u_{max}-\bar{u}$	$\lvert u_{max}-\bar{u}\rvert$
1	1	0	273.0	5.0	10000	723	0.998	0.008	0.008
2	8	0	406.4	8.0	10000	529	0.964	-0.026	0.026
3	2	0	273.0	6.3	10000	529	1.013	0.023	0.023
4	1	0	273.0	5.0	10000	229	0.703	-0.287	0.287
5	3	0	323.9	5.6	10000	254	0.939	-0.051	0.051
6	2	0	273.0	6.3	10000	868	1.033	0.043	0.043
7	2	-1	273.0	6.3	10000	436	1.088	0.098	0.098
8	3	0	323.9	5.6	10000	868	0.947	-0.043	0.043
9	4	0	323.9	7.1	10000	530	0.870	-0.120	0.120
10	2	0	273.0	6.3	10000	471	1.051	0.061	0.061
11	2	-1	273.0	6.3	10000	819	1.028	0.038	0.038
12	3	0	323.9	5.6	10000	787	0.999	0.009	0.009
13	4	-2	323.9	7.1	10000	819	1.012	0.022	0.022
14	8	0	406.4	8.0	10000	652	1.037	0.047	0.047
15	5	0	355.6	6.3	13654	772	0.974	-0.016	0.016
16	14	0	508.0	14.2	13654	33	0.926	-0.064	0.064
17	8	0	406.4	8.0	13654	70	0.915	-0.075	0.075
18	9	0	457.0	8.0	13654	728	2.611	1.621	1.621
19	8	0	406.4	8.0	13654	798	0.808	-0.182	0.182
20	8	0	406.4	8.0	13654	903	0.900	-0.090	0.090
21	8	+2	406.4	8.0	13654	212	0.765	-0.225	0.225
22	4	-1	323.9	7.1	13654	771	1.192	0.202	0.202
23	6	0	355.6	7.1	13654	798	1.018	0.028	0.028
24	4	0	323.9	7.1	13654	397	0.926	-0.064	0.064
25	4	0	323.9	7.1	13654	438	1.043	0.053	0.053
26	12	0	508.0	10.0	13654	33	1.054	0.064	0.064
27	5	0	355.6	6.3	14000	334	1.017	0.027	0.027
28	5	0	355.6	6.3	14000	404	1.017	0.027	0.027
29	8	-1	406.4	8.0	14000	978	1.121	0.131	0.131
30	4	0	323.9	7.1	14000	246	1.015	0.025	0.025
31	8	0	406.4	8.0	14000	819	0.949	-0.041	0.041
32	5	+1	355.6	6.3	14000	242	1.006	0.016	0.016
33	4	0	323.9	7.1	14000	181	0.810	-0.180	0.180
34	8	0	406.4	8.0	14000	177	0.858	-0.132	0.132
35	5	0	355.6	6.3	14000	588	0.971	-0.019	0.019
36	8	0	406.4	8.0	14000	297	0.886	-0.104	0.104
37	8	0	406.4	8.0	17205	302	1.114	0.124	0.124
38	8	0	406.4	8.0	17205	872	0.890	-0.100	0.100
39	5	0	355.6	6.3	17205	597	0.957	-0.033	0.033

40	9	0	457.0	8.0	17205	836	1.004	0.014	0.014
41	8	0	406.4	8.0	17205	810	0.948	-0.042	0.042
42	6	0	355.6	7.1	17143	624	1.132	0.142	0.142
43	8	0	406.4	8.0	17143	339	1.168	0.178	0.178
44	6	0	355.6	7.1	17143	518	1.198	0.208	0.208
45	10	0	457.0	10.0	17143	33	1.045	0.055	0.055
46	8	0	406.4	8.0	17143	404	0.932	-0.058	0.058
47	8	0	406.4	8.0	17143	756	1.003	0.013	0.013
48	8	0	406.4	8.0	17143	250	1.011	0.021	0.021
49	8	0	406.4	8.0	17143	714	1.144	0.154	0.154
50	8	0	406.4	8.0	17143	444	1.052	0.062	0.062
51	10	0	457.0	10.0	17143	712	0.946	-0.044	0.044
52	6	0	355.6	7.1	17143	685	1.203	0.213	0.213
53	9	0	457.0	8.0	17143	380	1.057	0.067	0.067
54	8	0	406.4	8.0	17205	246	1.156	0.166	0.166
55	8	0	406.4	8.0	17205	865	1.023	0.033	0.033
56	8	0	406.4	8.0	14000	741	1.009	0.019	0.019
57	9	0	457.0	8.0	14000	978	0.887	-0.103	0.103
58	5	0	355.6	6.3	14000	771	0.989	-0.001	0.001
59	8	0	406.4	8.0	14000	819	0.974	-0.016	0.016
60	9	0	457.0	8.0	14000	291	1.172	0.182	0.182
61	6	0	355.6	7.1	14000	622	1.057	0.067	0.067
62	8	0	406.4	8.0	14000	1002	0.996	0.006	0.006
63	8	0	406.4	8.0	14000	297	0.893	-0.097	0.097
64	6	0	355.6	7.1	17205	715	1.282	0.292	0.292
65	8	0	406.4	8.0	17205	636	0.883	-0.107	0.107
66	8	0	406.4	8.0	17205	868	1.102	0.112	0.112
67	9	0	457.0	8.0	17205	450	0.888	-0.102	0.102
68	8	0	406.4	8.0	14000	608	0.934	-0.056	0.056
69	8	0	406.4	8.0	14000	123	0.929	-0.061	0.061
70	9	0	457.0	8.0	17143	813	1.025	0.035	0.035
71	9	0	457.0	8.0	17143	1002	1.015	0.025	0.025
72	10	0	457.0	10.0	17143	385	0.921	-0.069	0.069
73	10	0	457.0	10.0	17143	638	0.924	-0.066	0.066
74	10	0	457.0	10.0	17143	741	0.926	-0.064	0.064
75	10	0	457.0	10.0	17143	782	0.963	-0.027	0.027
76	5	0	355.6	6.3	10000	114	1.104	0.114	0.114
77	2	0	273.0	6.3	10000	788	1.082	0.092	0.092
78	3	0	323.9	5.6	10000	108	0.980	-0.010	0.010
79	2	0	273.0	6.3	10000	347	0.946	-0.044	0.044

80	3	0	323.9	5.6	10000	638	0.894	-0.096	0.096
81	2	-1	273.0	6.3	10000	168	1.063	0.073	0.073
82	6	-2	355.6	7.1	10000	124	1.043	0.053	0.053
83	13	0	508.0	12.0	13654	704	0.972	-0.018	0.018
84	6	0	355.6	7.1	13654	88	1.040	0.050	0.050
85	6	0	355.6	7.1	13654	523	1.050	0.060	0.060
86	9	0	457.0	8.0	13654	608	0.867	-0.123	0.123
87	9	-1	457.0	8.0	13654	608	1.065	0.075	0.075
88	13	0	508.0	12.0	13654	987	0.926	-0.064	0.064
89	2	0	273.0	6.3	10000	87	1.043	0.053	0.053
90	1	0	273.0	5.0	10000	875	0.982	-0.008	0.008
91	2	0	273.0	6.3	10000	28	1.038	0.048	0.048
92	3	0	323.9	5.6	10000	832	1.126	0.136	0.136
93	2	0	273.0	6.3	10000	147	0.955	-0.035	0.035
94	3	0	323.9	5.6	10000	223	0.986	-0.004	0.004
95	2	-1	273.0	6.3	10000	199	1.078	0.088	0.088
96	9	0	457.0	8.0	13654	689	0.901	-0.089	0.089
97	8	0	406.4	8.0	13654	298	0.952	-0.038	0.038
98	10	0	457.0	10.0	13654	338	0.984	-0.006	0.006
99	9	0	457.0	8.0	13654	29	0.963	-0.027	0.027
100	8	0	406.4	8.0	13654	888	0.933	-0.057	0.057
101	10	0	457.0	10.0	13654	29	0.865	-0.125	0.125
102	10	0	457.0	10.0	17205	987	1.028	0.038	0.038
103	9	0	457.0	8.0	14000	688	1.011	0.021	0.021
104	6	0	355.6	7.1	14000	523	1.062	0.072	0.072
105	8	0	406.4	8.0	14000	70	0.808	-0.182	0.182
106	8	0	406.4	8.0	14000	29	0.937	-0.053	0.053
107	8	0	406.4	8.0	14000	597	1.194	0.204	0.204
108	8	0	406.4	8.0	14000	123	1.378	0.388	0.388
109	10	0	457.0	10.0	14000	689	0.885	-0.105	0.105
110	5	0	355.6	6.3	14000	500	0.990	0.000	0.000
111	5	0	355.6	6.3	14000	101	0.979	-0.011	0.011
112	5	0	355.6	6.3	14000	358	1.026	0.036	0.036
113	4	0	323.9	7.1	14000	387	0.998	0.008	0.008
114	5	0	355.6	6.3	14000	26	1.003	0.013	0.013
115	5	0	355.6	6.3	14000	185	0.953	-0.037	0.037
116	5	0	355.6	6.3	14000	179	0.946	-0.044	0.044
117	10	0	457.0	10.0	17205	267	0.997	0.007	0.007
118	8	0	406.4	8.0	17205	385	1.115	0.125	0.125
119	16	0	610.0	11.0	19556	246	1.375	0.385	0.385

120	14	0	508.0	14.2	19556	720	0.984	-0.006	0.006
121	14	0	508.0	14.2	19556	617	0.994	0.004	0.004
122	12	0	508.0	10.0	19556	810	0.909	-0.081	0.081
123	10	0	457.0	10.0	19556	158	1.020	0.030	0.030
124	17	0	610.0	12.5	19556	728	1.052	0.062	0.062
125	6	0	355.6	7.1	17143	818	1.142	0.152	0.152
126	8	0	406.4	8.0	17143	689	0.941	-0.049	0.049
127	8	0	406.4	8.0	17143	385	0.961	-0.029	0.029
128	9	0	457.0	8.0	17143	798	1.021	0.031	0.031
129	9	0	457.0	8.0	17143	987	1.006	0.016	0.016
130	8	0	406.4	8.0	17143	616	0.997	0.007	0.007
131	8	0	406.4	8.0	17205	120	1.174	0.184	0.184
132	8	0	406.4	8.0	17205	826	1.023	0.033	0.033
133	8	+2	406.4	8.0	17205	586	0.834	-0.156	0.156
134	10	0	457.0	10.0	17205	8	0.994	0.004	0.004
135	10	0	457.0	10.0	14000	771	0.953	-0.037	0.037
136	14	0	508.0	14.2	19556	241	0.975	-0.015	0.015
137	12	0	508.0	10.0	19556	688	0.893	-0.097	0.097
138	10	+1	457.0	10.0	14200	33	0.937	-0.053	0.053
139	8	0	406.4	8.0	14200	123	0.841	-0.149	0.149
140	8	0	406.4	8.0	14200	259	0.924	-0.066	0.066
141	9	0	457.0	8.0	14200	987	0.998	0.008	0.008
142	10	0	457.0	10.0	14200	821	0.891	-0.099	0.099
143	8	0	406.4	8.0	14200	126	1.013	0.023	0.023
144	8	0	406.4	8.0	14200	259	0.937	-0.053	0.053
145	9	0	457.0	8.0	14200	987	0.987	-0.003	0.003
146	13	0	508.0	12.0	14200	33	0.967	-0.023	0.023
147	9	0	457.0	8.0	14200	449	0.920	-0.070	0.070
148	8	0	406.4	8.0	14200	33	1.032	0.042	0.042
149	12	0	508.0	10.0	14200	962	1.049	0.059	0.059
150	13	0	508.0	12.0	14200	33	0.969	-0.021	0.021
151	9	0	457.0	8.0	14200	97	0.910	-0.080	0.080
152	8	0	406.4	8.0	14200	33	1.004	0.014	0.014
153	12	0	508.0	10.0	14200	741	0.932	-0.058	0.058
154	10	0	457.0	10.0	14200	529	0.914	-0.076	0.076
155	10	0	457.0	10.0	14200	529	0.943	-0.047	0.047
156	8	0	406.4	8.0	14200	15	0.824	-0.166	0.166
157	5	0	355.6	6.3	14200	737	0.991	0.001	0.001
158	5	0	355.6	6.3	14200	821	1.042	0.052	0.052
159	5	0	355.6	6.3	14200	723	0.952	-0.038	0.038

160	9	0	457.0	8.0	14200	878	0.938	-0.052	0.052
161	9	0	457.0	8.0	14200	741	0.962	-0.028	0.028
162	6	+1	355.6	7.1	14000	617	0.889	-0.101	0.101
163	9	0	457.0	8.0	14000	21	1.100	0.110	0.110
164	9	0	457.0	8.0	14000	631	1.054	0.064	0.064
165	8	0	406.4	8.0	14000	181	0.810	-0.180	0.180
166	3	0	323.9	5.6	14000	39	0.991	0.001	0.001
167	8	+2	406.4	8.0	17205	858	0.847	-0.143	0.143
SUM								**1.856**	**13.691**
MIN								**-0.287**	**0.000**
MAX								**1.621**	**1.621**
MEAN								**0.011**	**0.082**

Table 47. Results of the optimization of the beams of the three-dimensional frame with additional training (Approach A)

	u_{max}-$\bar{u}$	$\|u_{max}$-$\bar{u}\|$
SUM	0.235	12.070
MIN	-0.287	0.000
MAX	0.388	0.388
MEAN	0.001	0.073

Table 48. Overall deviation values between actual and target utilization rates of the beams of the optimized three-dimensional frame with additional training (Approach A), excluding the beam with utilization rate greater than 1.4 (i.e., beam number 18)

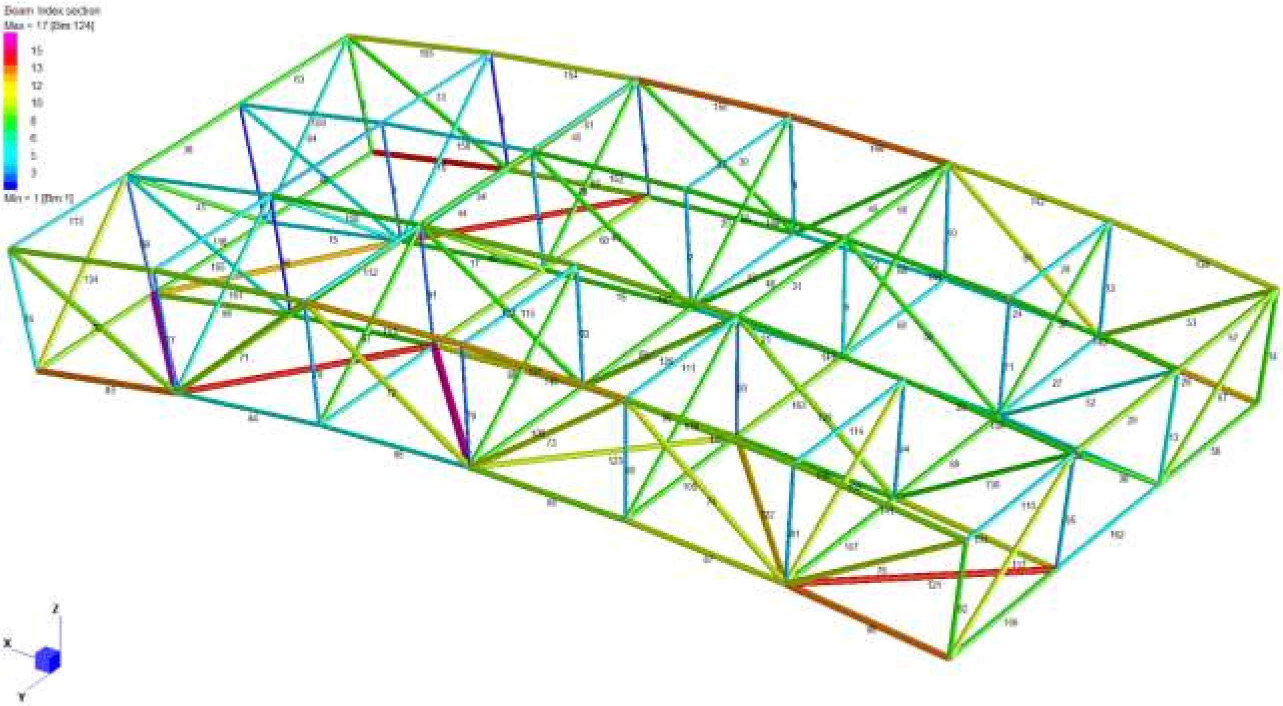


Figure 109. Indices of the sections of the optimized three-dimensional model beams obtained using the additional training (Approach B)

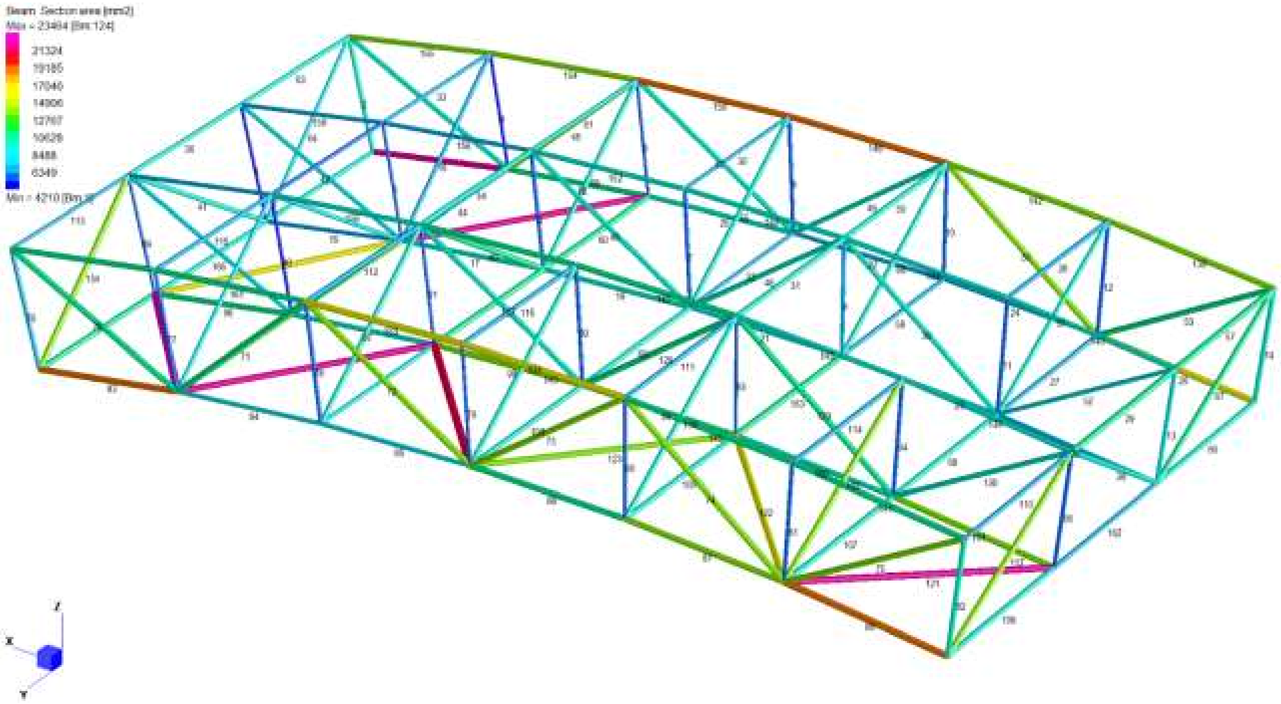


Figure 110. Area of the sections of the optimized three-dimensional model beams obtained using the additional training (Approach B)

Beam	ID Sect.	ID Sect. – ID Sect.$_{\text{NO TRAIN}}$	Diam [mm]	Thk [mm]	Length [mm]	Result case max	u_{max}	u_{max}-ū	|u_{max}-ū|
1	1	0	273.0	5.0	10000	723	1.013	0.023	0.023
2	8	0	406.4	8.0	10000	529	0.963	-0.027	0.027
3	2	0	273.0	6.3	10000	36	2.068	1.078	1.078
4	1	0	273.0	5.0	10000	229	0.701	-0.289	0.289
5	3	0	323.9	5.6	10000	254	0.936	-0.054	0.054
6	2	0	273.0	6.3	10000	989	16.279	15.289	15.289
7	3	0	323.9	5.6	10000	436	0.868	-0.122	0.122
8	3	0	323.9	5.6	10000	868	0.949	-0.041	0.041
9	4	0	323.9	7.1	10000	530	0.880	-0.110	0.110
10	3	+1	323.9	5.6	10000	471	0.810	-0.180	0.180
11	3	0	323.9	5.6	10000	819	0.823	-0.167	0.167
12	3	0	323.9	5.6	10000	787	0.999	0.009	0.009
13	6	0	355.6	7.1	10000	819	0.862	-0.128	0.128
14	8	0	406.4	8.0	10000	652	1.037	0.047	0.047
15	5	0	355.6	6.3	13654	772	0.976	-0.014	0.014
16	14	0	508.0	14.2	13654	33	0.928	-0.062	0.062
17	8	0	406.4	8.0	13654	70	0.910	-0.080	0.080
18	9	0	457.0	8.0	13654	728	2.577	1.587	1.587
19	8	0	406.4	8.0	13654	798	0.802	-0.188	0.188
20	8	0	406.4	8.0	13654	903	0.904	-0.086	0.086
21	6	0	355.6	7.1	13654	212	1.121	0.131	0.131

22	5	0	355.6	6.3	13654	771	1.148	0.158	0.158
23	6	0	355.6	7.1	13654	798	0.990	0.000	0.000
24	4	0	323.9	7.1	13654	397	0.929	-0.061	0.061
25	4	0	323.9	7.1	13654	438	1.043	0.053	0.053
26	12	0	508.0	10.0	13654	33	1.058	0.068	0.068
27	5	0	355.6	6.3	14000	334	1.018	0.028	0.028
28	5	0	355.6	6.3	14000	404	1.006	0.016	0.016
29	9	0	457.0	8.0	14000	978	0.879	-0.111	0.111
30	4	0	323.9	7.1	14000	246	1.009	0.019	0.019
31	8	0	406.4	8.0	14000	819	0.954	-0.036	0.036
32	4	0	323.9	7.1	14000	242	1.065	0.075	0.075
33	4	0	323.9	7.1	14000	181	0.828	-0.162	0.162
34	8	0	406.4	8.0	14000	177	0.858	-0.132	0.132
35	5	0	355.6	6.3	14000	503	0.976	-0.014	0.014
36	8	0	406.4	8.0	14000	297	0.883	-0.107	0.107
37	8	0	406.4	8.0	17205	302	1.111	0.121	0.121
38	8	0	406.4	8.0	17205	872	0.898	-0.092	0.092
39	5	0	355.6	6.3	17205	597	0.990	0.000	0.000
40	9	0	457.0	8.0	17205	836	1.005	0.015	0.015
41	8	0	406.4	8.0	17205	810	0.951	-0.039	0.039
42	6	0	355.6	7.1	17143	624	1.137	0.147	0.147
43	8	0	406.4	8.0	17143	339	1.170	0.180	0.180
44	6	0	355.6	7.1	17143	518	1.220	0.230	0.230
45	10	0	457.0	10.0	17143	33	1.048	0.058	0.058
46	8	0	406.4	8.0	17143	404	0.934	-0.056	0.056
47	8	0	406.4	8.0	17143	756	1.007	0.017	0.017
48	8	0	406.4	8.0	17143	250	1.015	0.025	0.025
49	8	0	406.4	8.0	17143	714	1.144	0.154	0.154
50	8	0	406.4	8.0	17143	444	1.046	0.056	0.056
51	10	0	457.0	10.0	17143	712	0.946	-0.044	0.044
52	6	0	355.6	7.1	17143	685	1.200	0.210	0.210
53	9	0	457.0	8.0	17143	380	1.057	0.067	0.067
54	8	0	406.4	8.0	17205	246	1.147	0.157	0.157
55	8	0	406.4	8.0	17205	865	1.023	0.033	0.033
56	8	0	406.4	8.0	14000	741	1.016	0.026	0.026
57	9	0	457.0	8.0	14000	978	0.897	-0.093	0.093
58	5	0	355.6	6.3	14000	518	1.010	0.020	0.020
59	8	0	406.4	8.0	14000	819	0.978	-0.012	0.012
60	9	0	457.0	8.0	14000	291	1.143	0.153	0.153
61	6	0	355.6	7.1	14000	622	1.055	0.065	0.065

62	8	0	406.4	8.0	14000	1002	0.996	0.006	0.006
63	8	0	406.4	8.0	14000	529	0.892	-0.098	0.098
64	6	0	355.6	7.1	17205	715	1.289	0.299	0.299
65	8	0	406.4	8.0	17205	636	0.889	-0.101	0.101
66	8	0	406.4	8.0	17205	868	1.100	0.110	0.110
67	9	0	457.0	8.0	17205	450	0.896	-0.094	0.094
68	8	0	406.4	8.0	14000	608	0.842	-0.148	0.148
69	8	0	406.4	8.0	14000	123	0.934	-0.056	0.056
70	9	0	457.0	8.0	17143	813	1.023	0.033	0.033
71	9	0	457.0	8.0	17143	1002	1.020	0.030	0.030
72	10	0	457.0	10.0	17143	385	0.925	-0.065	0.065
73	10	0	457.0	10.0	17143	638	0.927	-0.063	0.063
74	10	0	457.0	10.0	17143	741	0.921	-0.069	0.069
75	10	0	457.0	10.0	17143	782	0.970	-0.020	0.020
76	5	0	355.6	6.3	10000	114	1.104	0.114	0.114
77	2	0	273.0	6.3	10000	788	1.083	0.093	0.093
78	3	0	323.9	5.6	10000	108	0.977	-0.013	0.013
79	2	0	273.0	6.3	10000	347	0.947	-0.043	0.043
80	3	0	323.9	5.6	10000	638	0.898	-0.092	0.092
81	3	0	323.9	5.6	10000	168	0.827	-0.163	0.163
82	8	0	406.4	8.0	10000	124	0.744	-0.246	0.246
83	13	0	508.0	12.0	13654	704	0.959	-0.031	0.031
84	6	0	355.6	7.1	13654	88	0.997	0.007	0.007
85	6	0	355.6	7.1	13654	523	1.014	0.024	0.024
86	9	0	457.0	8.0	13654	608	0.908	-0.082	0.082
87	10	0	457.0	10.0	13654	608	0.870	-0.120	0.120
88	13	0	508.0	12.0	13654	987	0.943	-0.047	0.047
89	2	0	273.0	6.3	10000	87	1.043	0.053	0.053
90	1	0	273.0	5.0	10000	875	1.005	0.015	0.015
91	2	0	273.0	6.3	10000	28	1.039	0.049	0.049
92	3	0	323.9	5.6	10000	832	1.126	0.136	0.136
93	2	0	273.0	6.3	10000	147	0.955	-0.035	0.035
94	3	0	323.9	5.6	10000	223	0.984	-0.006	0.006
95	3	0	323.9	5.6	10000	199	0.839	-0.151	0.151
96	9	0	457.0	8.0	13654	689	0.902	-0.088	0.088
97	8	0	406.4	8.0	13654	298	0.954	-0.036	0.036
98	10	0	457.0	10.0	13654	338	0.978	-0.012	0.012
99	9	0	457.0	8.0	13654	29	0.963	-0.027	0.027
100	8	0	406.4	8.0	13654	888	0.932	-0.058	0.058
101	10	0	457.0	10.0	13654	29	0.860	-0.130	0.130

102	10	0	457.0	10.0	17205	987	1.019	0.029	0.029
103	9	0	457.0	8.0	14000	688	1.008	0.018	0.018
104	6	0	355.6	7.1	14000	523	1.071	0.081	0.081
105	8	0	406.4	8.0	14000	70	0.809	-0.181	0.181
106	8	0	406.4	8.0	14000	29	0.941	-0.049	0.049
107	8	0	406.4	8.0	14000	597	1.219	0.229	0.229
108	8	0	406.4	8.0	14000	123	1.342	0.352	0.352
109	10	0	457.0	10.0	14000	689	0.887	-0.103	0.103
110	5	0	355.6	6.3	14000	500	0.992	0.002	0.002
111	5	0	355.6	6.3	14000	101	0.977	-0.013	0.013
112	5	0	355.6	6.3	14000	358	1.026	0.036	0.036
113	4	0	323.9	7.1	14000	387	1.021	0.031	0.031
114	5	0	355.6	6.3	14000	26	0.997	0.007	0.007
115	5	0	355.6	6.3	14000	185	0.952	-0.038	0.038
116	5	0	355.6	6.3	14000	179	0.948	-0.042	0.042
117	10	0	457.0	10.0	17205	267	0.993	0.003	0.003
118	8	0	406.4	8.0	17205	385	1.103	0.113	0.113
119	16	0	610.0	11.0	19556	246	1.353	0.363	0.363
120	14	0	508.0	14.2	19556	720	0.984	-0.006	0.006
121	14	0	508.0	14.2	19556	617	0.989	-0.001	0.001
122	12	0	508.0	10.0	19556	810	0.906	-0.084	0.084
123	10	0	457.0	10.0	19556	158	1.021	0.031	0.031
124	17	0	610.0	12.5	19556	728	1.030	0.040	0.040
125	6	0	355.6	7.1	17143	818	1.142	0.152	0.152
126	8	0	406.4	8.0	17143	689	0.941	-0.049	0.049
127	8	0	406.4	8.0	17143	385	0.950	-0.040	0.040
128	9	0	457.0	8.0	17143	798	1.014	0.024	0.024
129	9	0	457.0	8.0	17143	987	1.009	0.019	0.019
130	8	0	406.4	8.0	17143	616	0.997	0.007	0.007
131	8	0	406.4	8.0	17205	120	1.176	0.186	0.186
132	8	0	406.4	8.0	17205	826	1.021	0.031	0.031
133	6	0	355.6	7.1	17205	586	1.260	0.270	0.270
134	10	0	457.0	10.0	17205	8	0.992	0.002	0.002
135	10	0	457.0	10.0	14000	771	0.958	-0.032	0.032
136	14	0	508.0	14.2	19556	241	0.973	-0.017	0.017
137	12	0	508.0	10.0	19556	688	0.893	-0.097	0.097
138	10	+1	457.0	10.0	14200	33	0.941	-0.049	0.049
139	8	0	406.4	8.0	14200	1002	0.849	-0.141	0.141
140	8	0	406.4	8.0	14200	259	0.908	-0.082	0.082
141	9	0	457.0	8.0	14200	987	0.991	0.001	0.001

142	10	0	457.0	10.0	14200	821	0.891	-0.099	0.099
143	8	0	406.4	8.0	14200	126	1.017	0.027	0.027
144	8	0	406.4	8.0	14200	259	0.935	-0.055	0.055
145	9	0	457.0	8.0	14200	987	0.981	-0.009	0.009
146	13	0	508.0	12.0	14200	33	0.969	-0.021	0.021
147	9	0	457.0	8.0	14200	449	0.920	-0.070	0.070
148	8	0	406.4	8.0	14200	33	1.018	0.028	0.028
149	12	0	508.0	10.0	14200	962	1.051	0.061	0.061
150	13	0	508.0	12.0	14200	33	0.971	-0.019	0.019
151	9	0	457.0	8.0	14200	97	0.912	-0.078	0.078
152	8	0	406.4	8.0	14200	33	0.992	0.002	0.002
153	12	0	508.0	10.0	14200	741	0.927	-0.063	0.063
154	10	0	457.0	10.0	14200	529	0.911	-0.079	0.079
155	10	0	457.0	10.0	14200	529	0.941	-0.049	0.049
156	8	0	406.4	8.0	14200	15	0.832	-0.158	0.158
157	5	0	355.6	6.3	14200	737	0.999	0.009	0.009
158	5	0	355.6	6.3	14200	821	1.037	0.047	0.047
159	5	0	355.6	6.3	14200	723	0.970	-0.020	0.020
160	9	0	457.0	8.0	14200	878	0.934	-0.056	0.056
161	9	0	457.0	8.0	14200	741	0.958	-0.032	0.032
162	5	0	355.6	6.3	14000	21	1.637	0.647	0.647
163	9	0	457.0	8.0	14000	21	1.096	0.106	0.106
164	9	0	457.0	8.0	14000	631	1.058	0.068	0.068
165	8	0	406.4	8.0	14000	181	0.812	-0.178	0.178
166	3	0	323.9	5.6	14000	783	0.989	-0.001	0.001
167	6	0	355.6	7.1	17205	858	1.318	0.328	0.328
SUM								**18.024**	**31.245**
MIN								**-0.289**	**0.000**
MAX								**15.289**	**15.289**
MEAN								**0.108**	**0.187**

Table 49. Results of the optimization of the beams of the three-dimensional frame with additional training (Approach B).

	u_{max}-$\bar{u}$	$\lvert u_{max}$-$\bar{u}\rvert$
SUM	-0.577	12.643
MIN	-0.289	0.000
MAX	0.363	0.363
MEAN	-0.004	0.078

Table 50. Overall deviation values between actual and target utilization rates of the beams of the optimized three-dimensional frame with additional training (Approach B), excluding the beams with utilization rates greater than 1.4 (i.e., beams number 3, 6, 18 and 162).

5.4 Conclusion

The optimization tool proposed in this chapter resulted in a distribution of beam cross-sections in the structure such that these elements have a utilization rate close to that desired. No cost reasoning has been done in this tool, and optimization is driven by utilization rate. Also, joints were not considered. However, the result obtained is interesting and is a good basis for future developments in which to include these other aspects for a more complete optimization.

The MLP network for defining beam sections has the advantage of being a model that is not specific to one structure to be optimized but can be used in multiple structures. Furthermore, additional training of the network performed during optimization can improve the results and also better direct them toward the desired goal.

The use of low-rank stiffness matrices made it possible to create a surrogate model with low computational cost and quickly and can also be a useful tool in contexts other than optimization, such as having a quick estimate of a new load condition applied to the structure.

Conclusions

Can Artificial Intelligence offer techniques that benefit the design process of steel megastructures?

The purpose of this research is to try to provide an answer to this question. Through an interdisciplinary approach, the contributions that computer science knowledge can make to the world of structural design were explored. Contamination between even very different sciences, allowing cultural enrichment that leads to seeing the problem from a different point of view, can lead to the development of innovative tools for solving a problem.

Scale of the problem

The search for a solution to a problem begins with knowledge of the problem itself. Based on this philosophy, the present research began with a description of the structural problem, with a focus on the design of steel megastructures. This specific category of structures plays a crucial role in society. Bridges, stadiums, buildings, port access gates, etc. represent a small example of this type of works. They have different functions, but all are important for the economic development of the geographical area. The large number of parameters needed to describe a design solution, which are interconnected and influence each other, makes the size of the problem particularly high. In addition, the complex physical phenomena require in-depth knowledge of the subject matter and onerous analyses to assess structural performance. The complexity of the problem is also amplified by the design process itself, which involves numerous actors with different roles and skills, and among whom it is particularly important that there be proper information transmissions. Their large number prevents with traditional design techniques to systematize all design variables simultaneously, thus generating an iterative problem-solving approach. Commercially available software used by designers also fails to meet the need for a complete view of the design problem, being limited by their specificity and/or by the computational burden that excessive model size and degree of detail achieved. The solution resulting from this design approach is therefore generally inefficient. The economic and social impact of such structures drives to search for new approaches to solving the design problem that lead to more optimized, safe, efficient, and cost-effective results. As one of the biggest obstacles to achieving this goal is the large number of variables involved in the problem, the idea is to use the tools offered by Artificial Intelligence to find the best solution within the range of possible solutions.

State of the art

In recent decades, machine learning techniques have been increasingly used in various scientific fields, including structural engineering. A review of the literature, however, shows that the research conducted for the use of these techniques in civil engineering is mainly aimed at material behavior assessment, damage detection, and structural monitoring. In contrast, applications in design are very limited and consider simple problems of very limited size. Thus, there emerges a gap in machine learning tools applicable to real structures, particularly megastructures, for which the need to apply such tools would instead be higher because the complexity of the problem limits the designer's ability to identify the best solution within a reasonable time and, in addition, the considerable size of the structure means that the reduction in environmental, economic and social impact resulting from their optimization is particularly relevant. The difficulties in devising an optimization tool for large steel structures are related both to the size of the problem, which implies the need for adequate computational resources, and to the required level of knowledge of the structural problem. The aspects to be considered for steel structures are numerous, very different in nature, and concerning different skills. They range from local phenomena such as plasticity and fracture toughness, which require in-depth theoretical knowledge in the structural field and on the behavior of steel, to the more practical problems of transportation, execution, and erection of the structure, which designers often consider little or have little information about during the design process. All these aspects should concur in formulating the optimization problem, that is, in identifying the variables and correctly writing the cost function, which the chosen optimization technique will have to minimize.

Preliminary work

Optimized structural solutions must meet the verifications and performance required by the standards. The types of checks and performance depend on the reference standards and the types of elements available for solutions. Therefore, preliminary work needs to be done to define the reference standards and the types of allowable elements and geometric configurations, based on which to create a structural behavior verification form that provides the utilization rate of the elements. In this thesis, reference was made to the checks by the

Eurocodes, only the hollow circular sections were evaluated for the beams and only the "single plate connection" type joints were considered, being the main purpose of the research to investigate the optimization methods. However, the verifier can be expanded in the future to include other cases. Limiting the allowed structural types makes it possible to simplify the verifier's processing and narrow the algorithm's search range for the optimized solution, which on the one hand facilitates the algorithm's convergence, but on the other hand limits its ability to explore different solutions. For the calculation of stresses on structural elements to be used for verifications, it is necessary to rely on FEM software with which to create a calculation model. Correctly estimating the size of the elements needed for both verification and estimating the costs of materials and workmanship requires the support of CAD software with which to create a three-dimensional geometric model and measure the dimensions of individual elements. Therefore, it was necessary to create a framework that would link the optimization algorithm with FEM and CAD software with which to create candidate solution models and obtain the parameters to be used in calculating the cost function.

Combined beam and joint optimization

In this research, the problem of optimizing beams and joints of a multi-way node was addressed. The same approach applies to a larger portion of the structure. Two philosophically different approaches to solve the problem were studied: the Genetic Algorithm and the Gradient-based algorithm. A third algorithm has only been sketched and not yet applied, in which elements are grouped on the basis of their stresses, and, taking a cue from the Particle Swarm Algorithm, for each element the update of variables also takes into account the value taken by them in the best element of the group. The Genetic Algorithm, due to its versatility and intuitiveness, was not difficult to apply to the structural problem. The fact that structural variables generally lend themselves to being described by discrete variables, either because they are related to the commercial availability of the elements or because they represent a quantification of the elements, facilitated the algorithm's search for the solution. As it is designed, the trend of the curve representing the value of the fitness function of the best individual in the population as a function of the number of iterations is monotonically decreasing. This implies that the best solution will always be found at the last iteration performed, without the need to search for it in previous iterations. The fact that the Genetic Algorithm evaluates only candidate solutions as a whole, without capturing whether variation in a single variable has positive or negative effects, suggests that this type of algorithm is effective but not efficient. This consideration prompted the evaluation of another algorithm, based on gradients. With this approach, we want to estimate the influence of variations of single variables on the objective function to define the new candidate solution. Given the amount of variables to describe a candidate solution, the variables were divided into blocks with the aim of trying to facilitate the convergence of the algorithm to the final solution. Unlike the previous one, many difficulties were encountered in applying this approach to the structural case. First, the non-derivable nature of the cost function necessitated the use of a gradient estimate. The presence of very different variables, some discrete and others continuous, and which take values with very different scales has required numerous tests to evaluate the best way to treat them, especially in the choice of the deviation of their value necessary for the calculation of the gradient estimation and to evaluate how to update the values of the variables based on the gradient obtained. In the evaluation of the gradient, the variation in the value of the variables was performed using a different delta depending on the nature of the variables. The same delta is then applied when updating the solution, through a sign function that considers the estimated gradient. Of the two algorithms, the best results were obtained with the genetic one. The computational cost of the analysis is related not only to the calculations performed by the algorithm for creating and analyzing the candidate solutions and for calculating the update of the variables, but also to the creation and analysis of the calculation and geometric models with which the stresses and effective dimensions of the elements were derived, respectively. The number of iterations required to achieve convergence, which is related to the number of models created during the analysis, was lower for the Genetic Algorithm. As a result, it can be said that the Genetic Algorithm approach was less computationally burdensome. However, it should be noted that the worse performance demonstrated by the Gradient-based algorithm compared to the Genetic Algorithm may also be related to the difficulty of computing a gradient-dependent variable update function. The function used in this thesis, which merely evaluates the direction of gradient-dependent variable updating, appears very crude, and future studies may lead to its improvement, thus affecting the performance of the algorithm.

Surrogate models

The need to obtain design solutions in a reasonable amount of time clashes with the need to conduct finite element simulations with which to derive parameters for evaluating structural behavior. In fact, the time and computational resources required for such analyses are high for large models or in the case where nonlinear analyses are required. This consideration has prompted an attempt to develop a surrogate model that provides stress estimation without the need to perform FEM analysis. A review of the existing literature revealed a lack of studies on this topic. Several approaches to surrogate model development have been studied. Among these, the one that has brought the best results in relation also to the burden for its construction is the one involving the creation of a low-rank stiffness matrix. It should be noted that this matrix is related to the stiffness distribution of the structure with which it was created. Therefore, changing the cross sections of the beams in the structure will affect the quality of the results. However, it should be kept in mind that the purpose is only to estimate stresses, to speed up iterations in an optimization workflow, and is not intended to replace FEM analyses, which still need to be performed but less frequently.

Other approaches studied for surrogate modeling involve the creation of Neural Networks. In particular, the frame structure lends itself particularly well to being described by means of graphs, on the basis of which Graph Neural Networks can be constructed. The specificity of the structural problem prompted the creation of ad hoc convolutional layers that would take into account the underlying physicality of the problem. Many difficulties were encountered in the training phase of the networks. Even with only frames of a hundred beams, learning was onerous and acceptable error levels could not be achieved. Therefore, these approaches are interesting but need further study to improve model learning and thus be applicable to real cases.

Optimization with Neural Network

Neural Networks are increasingly being used to model large complex problems. In this research, this technique was used to create a model that optimizes the beam sections of a frame structure, looking for the configuration of cross profiles for which all elements have the desired utilization rate. The use of Neural Networks has the advantage that pre-existing designs like the one under consideration can be used for training the network. Alternatively, the database to be used for training can be created ad hoc, thus having the possibility of creating dummy cases with only the sections belonging to the database available for optimization and with lengths and stresses similar to those of the starting model to be optimized. Another advantage of Neural Networks is related to the fact that the trained models are not specific to one structure but can be used to design different structures.

Changing the cross sections of the beams with those in output by the network results in a change in the weight acting on the structure and thus a change in the forces on the beams, on which the output of the Neural Network itself depends. The optimization process is therefore iterative, and to avoid having to update the FEM model and perform finite element analysis at each iteration, a surrogate model was created using the low-rank stiffness matrix approach. This optimization flow was applied to two case studies, a two-dimensional frame of 11 elements and a three-dimensional frame of 167 elements. In both cases the optimization approach yielded good results, providing a set of cross sections for the frame beams with which they have utilization rates closer to the target than the starting models. The results were further improved by performing additional training during the optimization process itself, using a loss function that takes into account the deviation of the actual utilization rate from the target one, in addition to the loss function already used during the previous network training. Two approaches have been proposed to carry out this additional training, which differ in the way the gradient is calculated. Careful selection of the configuration of the two optimizers that update the network parameters based on the two loss functions proved to be critical to properly perform the additional training. The use of additional training also has the advantage of reducing the risk of under-sizing, which is obviously a much less desirable condition than over-sizing, through the introduction of a penalty coefficient.

This optimization approach does not consider joints and nodes, but only regards beams. A more correct optimization would require the introduction of joints and nodes, since, as explained earlier, there is a mutual influence between beams and the elements that connect them together. To introduce these elements into the optimization process as well, one idea might be to take the same approach used for beams, that is, to create a Neural Network that, taking the forces as input, provides the joint configuration with the desired utilization rate. Different Neural Networks could be trained depending on the type of joint to be designed. The workflow that includes both beam and joint Neural Networks would be iterative in nature because changing the geometry of the elements affects the loads and forces and because of the mutual influence of beams and joints.

It should be noted that the proposed Neural Network optimization workflow only considers the utilization rate of the elements and not the costs of materials and processing. For a more accurate and correct optimized solution, these concepts should also be introduced into the optimization process.

Research findings

Through this thesis, different approaches were investigated for the development of AI-based tools to provide support for structural design activities, with a focus on megastructures, which are characterized by complexity and onerousness in analysis. The lack of previous research in this area that emerged after an in-depth study of the state of the art made it necessary to investigate different approaches for the development of these tools. In particular, the search for the optimum was pursued both through the development of two algorithms, one meta-heuristic and the other gradient-based, and through the implementation of a neural network. The meta-heuristic algorithm used is the genetic algorithm, which has already been used in a number of optimization problems in other areas, and has performed well, but does not appear to be the most efficient in the context of structural design because it does not allow for efficient exploitation of information from the analysis of candidate solutions. The gradient-based algorithm would have good intentions of making up for this shortcoming of the genetic method, but to be competitive it would need further study to devise a more efficient function for updating variables based on the gradient. Both algorithms were applied to a small portion of the structure in order to evaluate the pros and cons of the two methods. These approaches can also be applied to larger portions of the structure. However, the large increase in variables to the problem may require contrivances to reduce the size of the search space for the optimum and make it easier to reach the solution. A third algorithm, the Grouping Algorithm, based on groupings of similar elements, has been sketched in this direction.

In addition to the algorithms, the use of Neural Networks was also investigated, which has the advantage of being able to exploit databases of existing structures for model training. The neural network has been used within an iterative type workflow in which at each iteration the loads are updated on the basis of the profiles of the members in output to the neural network and an additional training specially created of the Neural Network is carried out. The latter represents an approach that allows to improve the result coming from the optimization process through an adaptation of the network to the structural problem in question. The onerousness of finite element analyzes for complex structures has led to the search for surrogate models that would speed up the updating of the loads resulting from the updates of the profiles of the members of the structure with those provided in output by the Neural Network. The approach based on surrogate models is particularly interesting as it could also be useful in other situations, for example for a quick estimation of the stresses deriving from a new loading condition.

Closing remarks

Answering the question posed at the beginning of this concluding chapter, Artificial Intelligence methods could have interesting applications in supporting the design of steel megastructures. On the one hand, they provide algorithms for the evaluation of different design solutions for optimization purposes; on the other hand, they offer methods for the creation of surrogate models that allow reducing the number of finite element analyses performed with complex computational models as they allow rapid estimation of results, which are useful for quick evaluation of different design solutions. The creation of these tools requires a well-rounded knowledge of steel structures, since proper design requires, on the one hand, in-depth knowledge of civil engineering and structural steel in order to be able to assess structural performance and, on the other hand, knowledge of machining, transportation, and erection processes, which are critical for feasibility assessment and cost estimation. In addition, preliminary work on the study of regulations is required, which, together with theoretical knowledge, allows for the calculation of element utilization rates. Added to this is the accurate creation of a database of available materials and elements on which to limit the choice, thus enabling a reduction in the search space for the solution. The size of the problem, which needs a large number of variables, moreover very different ones, to be described, and the impossibility of expressing the problem in terms of simple equations requires a deep knowledge of machine learning for appropriate data management and efficient application of computer science notions in order to develop a tool that provides valid results quickly. Given the extent and varied amount of knowledge and expertise required, the development of efficient and effective tools to support the design of steel megastructures requires significant investment. Justifying this investment is the economic and social impact of these structures, the improvement of which can easily translate into benefits for the whole society.

.

References

1. *Performance trends in the construction industry worldwide: an overview of the turn of the century.* **I. M. Horta, A. S. Camanho, Jill Johnes & Geraint Johnes.** 2013, Journal of Productivity Analysis, pp. 89-99.

2. *China's energy consumption in construction and building sectors: An outlook to 2100.* **Guangyue Xu, Weimin Wang.** March 15, 2020, Energy, Vol. 195.

3. **Mueller, Caitlin T.** *Computational Exploration of the Structural Design Space.* Massachusetts Institute of Technology : s.n., 2014.

4. *Design iteration in construction projects – Review and directions.* **Maheswari, P. Mujumdar and J. U.** 1, March 2018, Alexandria Engineering Journal, Vol. 57, pp. 321-329.

5. *Explanation of the collapse of Terminal 2E at Roissy–CDG Airport by nonlinear deterministic and reliability analyses.* **H. Daou, W. A. Salha, W. Raphael and A. Chateauneuf.** June 2019, Case Studies in Construction Materials, Vol. 10.

6. **Ajouz, R.** *Optimising production costs of steel trusses - A computational approach of designing cost-effective steel trusses with welded connections.* s.l. : Delft University of Technology, 2018.

7. *Optimal design of beam-column connections of plane steel frames using the component method.* **Falcòn, R. d. S. Hortencioa and G. A. S.** 2018, Latin American Journal of Solids and Structures, Vol. 15.

8. *Optimum design of semi-rigid connections using metamodels.* **C. Díaz, M. Victoria, O. M. Querin, P. Martí.** s.l. : Journal of Constructional Steel Research, November 2012, Vol. 78, pp. 97-106.

9. **1993-1-1:2005+A1:2014, BS EN.** *Eurocode 3: Desiign of steel structures - Part 1-1: General ruels and rules for buildings.*

10. **[Online] https://www.tekla.com/resources/blogs/an-advanced-brim-workflow.**

11. *Artificial Intelligence in Civil Engineering.* P. Lu, S. Chen and Y. Zheng. s.l. : Hindawi Publishing Corporation, 2012, Mathematical Problems in Engineering, Vol. 2012, p. 22.

12. *Emerging artificial intelligence methods in structural engineering.* Burgueño, H. Salehia and R. September 15, 2018, Engineering Structures, Vol. 171, pp. 170-189.

13. *Machine Learning in Structural Design: An Opinionated Review.* Málaga-Chuquitaype, C. 2022, Frontiers in Built Environment, Vol. 8. 2297-3362.

14. *Neuro-fuzzy based prediction of the durability of self-consolidating.* Nehdi, M. T. Bassuoni and M. L. 6, December 2008, Computers and Concrete, Vol. 5, pp. 573-597.

15. *Prediction of compressive strength of SCC and HPC with high volume fly ash using ANN.* B. K. R. Prasad, H. Eskandari and B. V. V. Reddy. 1, January 2009, Construction and Building Materials, Vol. 23, pp. 117–128.

16. J. Farkas, K. Jàrmai. *Optimization Design of Steel Structures.* Berlin : Springer, 2013.

17. *Optimization in a realistic structural engineering context: Redesign of the Market Hall in Ghent.* W. Dillen, G. Lombaert, R. Mertens, H. Van Beurden, D. Jaspaert, M. Schevenels. February 1, 2020, Engineering Structures.

18. *Learning to Simulate and Design for Sructural Engineering.* Cheng, Kai-Hung Chang and Chin-Yi. Online : s.n., 2020. Proceeding of the 37th International Conference on Machine Learning. Vol. 119.

19. *Application of an Efficient Gradient-Based Optimization Strategy for AircraftWing Structures.* O. Dababneh, T. Kipouros and J. F. Whidborne. 2018, Aerospace, Vol. 5.

20. *A Continuous Space Neural Language Model for Bengali Language.* Chowdhury, Hemayet & Imon, Md & Rahman, Anisur & Khatun, Aisha & Islam, Md Saiful. 2020.

21. *A hybrid gradient-based/metaheuristic method for Eurocode-compliant size, shape and topology optimization of steel structures.* W. Dillen, G. Lombaert and M. Schevenels. 239, 2021, Engineering Structures.

22. *Modified particle swarm optimization algorithm for engineering structural optimization problem.* Sanyang, R. Yanzhi and L. 2017, 13th International Conference on Computational Intelligence and Security.

23. *Data-driven approximation algorithms for rapid performance evaluation and optimization of civil structures.* S. Tseranidis, N. C. Brown, C. T. Mueller. 2016, Automation in Construction, Vol. 72, pp. 279–293.

24. Bishop, Christopher M. *Pattern Recognition and Machine Learning.* [ed.] Springer Science+Business Media. 2006.

25. *Densely Connected Convolutional Networks.* G. Haung, Z. Liu, L. van der Maaten. 2017, 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

26. *Graph neural networks: A review of methods and applications.* J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun. 2020, AI Open.

27. *A State-of-the-Art Survey on Deep Learning Theory and Architectures.* Md. Z. Alom, T. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. Nasrin, M. Hasan, B. Essen, A. Awwal, V. Asari. 292, 2019, Electronics, Vol. 8.

28. *Electric Load Forecasting in Smart Grid Using Long-Short-Term-Memory based Recurrent Neural Network.* J. Zheng, C. Xu, Z. Zhang, X. Li. 2017, The 51st Annual Conference on Information Systems and Sciences Proceedings.

29. *Best Graph Neural Networks architectures: GCN, GAT, MPNN and more.* Karagiannakos, S. 2021, https://theaisummer.com/.

30. *A Zeroth-Order Block Coordinate Descent Algorithm for Huge-Scale Black-Box Optimization.* H. Cai, Y. Lou, D. McKenzie, W. Yin. 2021. Proceedings of the 38th International Conference on Machine Learning.

31. *CoSaMP: Iterative signal recovery from incomplete and inaccurate samples.* D. Needell, J. A. Tropp. 3, May 2009, Applied and Computational Harmonic Analysis, Vol. 26, pp. 301-321.

32. Savage, T. repository GitHub. [Online] September 3, 2021. [Cited: April 11, 2022.] https://github.com/trsav/particle-swarm.

33. *Utilization of structural steel in buildings.* Allwood, M. C. Moynihan and J. M. 2168, August 8, 2014, Proceedings of the royal society A, Vol. 470.

34. I. Goodfellow, Y. Bengio and A. Courville. *Deep Learning.* s.l. : MIT Press, 2016.

35. *Learning to be Global Optimizer.* H. Zhang, J. Sun and Z. Xu. 2003.

36. *Position-aware Graph Neural Networks.* J. You, R. Ying and J. Leskovec. 2019.

37. ***Applications of Artificial Intelligence Techniques for Optimization of Structural Steel Connections.*** **E. Duong, A.J.Darras,R. Driver, M. Essa, A. Imanpour. 2021, EasyChair Preprint, Vol. 6837.**