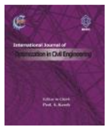
INTERNATIONAL JOURNAL OF OPTIMIZATION IN CIVIL ENGINEERING
Int. J. Optim. Civil Eng., 2021; 11(3): 357-382



COST OPTIMIZATION OF STEEL-CONCRETE COMPOSITE I-GIRDER BRIDGES WITH SKEW ANGLE AND LONGITUDINAL SLOPE, USING THE SM TOOLBOX AND THE PARALLEL PATTERN SEARCH ALGORITHM

R. Javanmardi and B. Ahmadi-Nedushan[*, †]
Department of Civil Engineering, Yazd University, Yazd, Iran

ABSTRACT

In this research, the optimization problem of the steel-concrete composite I-girder bridges is investigated. The optimization process is performed using the pattern search algorithm, and a parallel processing-based approach is introduced to improve the performance of this algorithm. In addition, using the open application programming interface (OAPI), the SM toolbox is developed. In this toolbox, the OAPI commands are implemented as MATLAB functions. The design variables represent the number and dimension of the longitudinal beam and the thickness of the concrete slab. The constraints of this problem are presented in three steps. The first step includes the constraints on the web-plate and flange-plate proportion limits and those on the operating conditions. The second step consists of considering strength constraints, while the concrete slab is not yet hardened. In the third step, strength and deflection constraints are considered when the concrete slab is hardened. The AASHTO LRFD code (2007) for steel beam design and AASHTO LRFD (2014) for concrete slab design are used. The numerical examples of a sloping bridge with a skew angle are presented. Results show that active constraints are those on the operating conditions and component strength and that in terms of CPU time, a 19.6% improvement is achieved using parallel processing.

Keywords: optimization, CSI OAPI, SM toolbox, steel-concrete composite I-girder bridges, parallel processing, pattern search algorithm.

Received: 5 May 2021; Accepted: 10 July 2021

---

[*]Corresponding author: Department of Civil Engineering, Yazd University, Yazd, Iran
[†]E-mail address: behrooz.ahmadi@yazd.ac.ir (B. Ahmadi-Nedushan)

1. INTRODUCTION

Undoubtedly, one of the most fundamental infrastructures of the economic development of any country is its transport network. Thus, the development of this network is a necessity for economic growth in each country. Bridges are considered as one of the vital components of transportation networks. Hence, it is essential to provide useful tools and methodologies for the optimal design of bridges [1, 2].

Optimization can be defined as the act of obtaining the best result under given constraints. Optimal designs result in saving material and energy resources. Developments in computer hardware and software and advances in numerical optimization methods make it possible to formulate the design of complicated discrete engineering problems as an optimization problem and solve them by one of the optimization methods [3]. In the recent decades, many researchers have investigated the optimal design of bridges.

Simões and Negrão applied a multi-objective optimization algorithm for cable bridges using maximum and minimum stress and deflection under dead load as constraints [4]. Guan et al. attempted to optimize the topology of bridges by considering stress, deflection, and frequency constraints [5]. Srinivas and Ramanjaneyulu studied two-lane bridges and three longitudinal girders and used a combination of genetic algorithms and artificial neural networks to achieve optimum cross-section [6]. Cheng attempted to optimize an arch bridge with a steel truss using a hybrid genetic algorithm, considering the weight as an objective function [7]. The optimization of cable bridges was proposed by Baldomir et al. [8] and the objective function was the volume of consumed steel. Wei et al. optimized an arch bridge with a 420-meter span [9]. The cross-section of the bridge was considered to be a box with a concrete flange and steel web. Lute et al. presented a genetic algorithm for cable bridge optimization [10]. They considered the cost of materials as a cost function. Also, single-cell box cross-sections were used. Makiabadi et al. presented the optimal design of a single-span steel bridge using the teaching-learning-based optimization algorithm [11].

In the past decade, many researchers have applied various multi-criteria optimization in the field of bridge design [12], considering other factors, besides the cost, like the security of the infrastructure and the CO2 emissions [13], the embodied energy [14], or the lifetime reliability [15]. In the field of optimization of concrete-steel composite structures, various studies have been conducted by researchers [16–20].

Kaveh and his students firstly used open application programming interface (OAPI) in combination with parallel processing in 2012 and 2014. Kaveh et al. optimized the self-weight of steel structures using the SAP2000 and MATLAB software links, as well as the parallel processing toolbox in MATLAB. For this purpose, they used the Cuckoo Search (CS) algorithm. CS is a population-based algorithm based on the behavior of Cuckoo species in combination with Lévy flight. In this study, the variables were considered as the number of wide-flange-shape (W-shape) sections. Constraints such as member strength, geometric limitation, and frame displacement, were considered. The results showed the efficiency of this method in designing practical structures [21].

Kaveh et al. optimized the self-weight of a multi-span composite box girder bridge using the Cuckoo Search (CS) algorithm. The considered variables WERE the dimensions of steel beams and concrete slabs in different parts of the bridge. The constraints were strength, service, and geometric limits. To increase the efficiency of the proposed method, they used the parallel processing toolbox in MATLAB software. Furthermore, the performance of PSO [22] and HS [23] algorithms were compared, and the efficiency of the proposed method was demonstrated [24]. Briefly, the results showed the effectiveness of this method in saving material consumption in practical bridges.

Kaveh et al. [21, 24] optimized double-layer barrel vaults using the improved magnetic charged system search (IMCSS) algorithm and the open application programming interface (OAPI) [25]. In the IMCSS algorithm, the magnetic charged system search (MCSS) and an improved scheme of harmony search (IHS) algorithm were upgraded for better results and convergence. The OAPI was utilized for the structural analysis process to link the analysis software with the IMCSS algorithm through the programming language. The results demonstrated the efficiency of OAPI as a powerful interface tool for analysis of large-scale structures, such as double-layer barrel vaults, and the robustness of IMCSS as an optimization algorithm in achieving the optimal results [26].

Kaveh et al. optimized the problem of simultaneous shape and size optimization of single-layer barrel vault frames, which contains both discrete and continuous variables problem, using IMCSS and OAPI. They proved the efficiency of the proposed method by comparing it with some of the existing structures [27].

Cai et al. investigated the use of polymeric reinforced carbon fiber materials in a genetic algorithm-based optimization process to improve the aerodynamic performance of cable systems [28]. They considered static and dynamic behavior and vibration performance of the bridge. Cable force on a cable bridge was optimized by Martins et al. using of a gradient-based optimization approach [29].

Their study included the time-dependent features of concrete, construction sequence, and nonlinear geometry. They used Euler-Bernoulli beams in finite element modeling. Gocál and Ďuršová conducted a parametric study to optimize the beams' placement on a steel-concrete composite bridge [30]. They modeled 32 potential structures with the SCIA software and examined the amount of consumed steel. Pedro et al. presented a two-step optimization method for optimizing I-girder steel-concrete composite bridges [31]. In the first step, using a model prepared by the bridge designer, a starting design is obtained to start the second step. In the second step, the optimization process is completed using this point and the three-dimensional finite element model. To reduce the CPU time, in the first step, they used a simplified two-dimensional model. While this idea may be useful in the reduction the CPU time, due to its simplification, the proposed methodology does not consider many of the model's specifications, which may not ultimately guide the design to the optimal point. Also, in the second phase where a 3D finite element model is used, they do not take into account some of the effective parameters, such as skew angle, longitudinal slope, elastomers effect, and change in steel cross-section dimensions.

Kaveh and Zarandi optimized steel-concrete composite bridges using CBO, ECBO, and VPS algorithms. In this study, they used a two-dimensional model for bridge analysis. They also used a simplified by-law method to determine the live load distribution coefficient due to using the simplified 2D beam model instead of the 3D model [2].

In this research, the optimization problem of the steel-concrete composite bridge is investigated. The pattern search algorithm is used for this purpose. In order to improve the performance of this algorithm, a method with parallel processing principles is proposed to speed up the convergence of the algorithm and to increase the probability of reaching the optimal point. Using parallel processing makes it possible to use supercomputers to solve these problems. As the computing power of the supercomputers increases, the accuracy and speed of problem-solving also increase since this method's performance depends on the computational power of the computing machine. In this method, the limitation of the iterative approaches is eliminated. This limitation includes the dependency of the current result on that of the previous step. In Section 0, using the open application programming interface (OAPI), the SM toolbox is developed. In this toolbox, the OAPI commands are implemented as MATLAB functions. The toolbox is updated for SAP2000 and CSI BRIDGE software from version 17 to the latest version. Using this toolbox, a three-dimensional finite element model with all the details of the problem is formed. MATLAB software optimization toolbox is used to perform the optimization process, which is integrated with its parallel processing capabilities. The objective function is the final cost of the deck construction. Problem constraints include web-plate and flange-plate proportion limits, operating conditions, steel beam strength constraints, and bridge deflection.

In Section 0 of this article, a description of serial processing and parallel processing is provided. Section 0 contains descriptions of the optimization problem and the parallelization process of the pattern search algorithm. The problem is described in Section 5. Finally, in Section 6, the conclusions of this study are presented.

2. SM TOOLBOX

Today, in various fields of engineering, including structural and earthquake engineering, much research is done concerning artificial intelligence and optimization. Structural analysis is required in the programming process in most of such studies. Researchers use different methods to solve this problem, one of the most effective of which is linking MATLAB software with powerful structural analysis software. SAP2000 software, founded in 1975 by CSI affiliated with Berkeley University, is one of the most powerful software. The software developed by CSI has been used by thousands of engineering firms in over 160 countries for the design of major projects, including the Taipei 101 Tower in Taiwan, One World Trade Center in New York, the 2008 Olympics Birds Nest Stadium in Beijing, and the cable-stayed Centenario Bridge over the Panama Canal. CSI's software is backed by more than four decades of research and development, making it the trusted choice of sophisticated design professionals everywhere [32]. The company offers CSI OAPI to link its products to programming languages based on the Visual Basic programming language and provides only one example for other programming languages. In relation to MATLAB software, indirect commands are used for modeling, and string variables and vectors follow the Visual Basic commands, which is not easy to use for MATLAB programmers. In the SM toolbox, CSI OAPI commands are provided as explicit MATLAB functions, making it much easier for MATLAB programmers. The toolbox is also designed to support SAP2000 and CSI BRIDGE software from version 17 to the latest version [33].

In general, the problems with CSI OAPI for MATLAB programming can be summarized as follows:

2.1 Difficulty in recognizing the main structure of commands

The main structure of the commands provided by CSI OAPI is as follows:

$$\begin{gathered}\mathrm{R} = \mathrm{NET.explicitCast}\left(\mathrm{C}_1, 'C_2'\right) \\ \left[ret\right] = R.FuncName\left(I\right)\end{gathered} \tag{1}$$

where C_2 classes are not specified for all commands, and also inputs are not defined for use in MATLAB.

In the SM toolbox, all commands are used as follows, and all inputs and outputs (I, O) are clearly defined using variables known in MATLAB [33].

$$\begin{gathered}[O] = SM.Class.Func(I) \\ [O] = SM.Func(I)\end{gathered} \tag{2}$$

2.2 Difficulty in defining numerical and string vector variables

The method provided by CSI OAPI forces the programmers to initialize a variable before using it. They also use commands (3) to (5) to introduce numerical, string, and boolean vectors.

$$D = NET.createArray('System.Double', N_1) \tag{3}$$

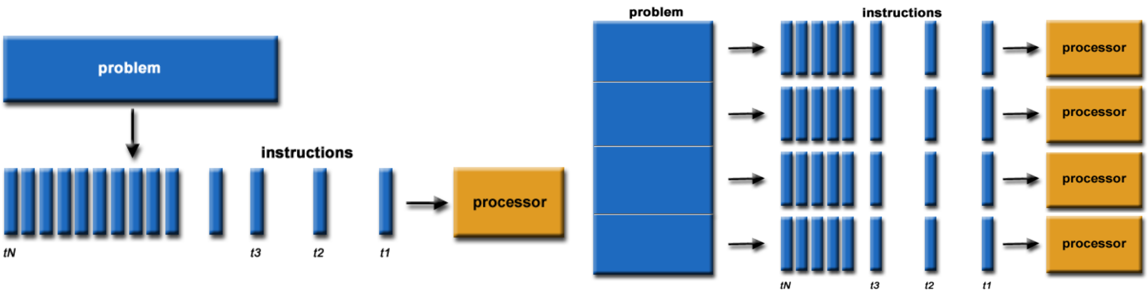
$$S = NET.createArray('System.String', N_2) \tag{4}$$

$$B = NET.createArray('System.Boolean', N_3) \tag{5}$$

A loop must be used to initialize these vectors. Also, the size of vectors (N_1, N_2, N_3), in some cases, is not known in advance. In the SM toolbox, there is no need to introduce variables before using them. Cell vectors and arrays are also applied for this purpose. Therefore there is no need for using additional loops in the code.

3. PARALLEL PROCESSING AND SERIAL PROCESSING

The concept of parallel processing is briefly explainedsa in this section. In serial processing, a problem is divided into smaller parts, then each of these parts runs on the processor in sequence (Fig. 1.a). In this type of processing, there is only one operation per processor at a time.



(a) (b)

Figure 1. (a) How to solve a problem in serial processing [34]. (b) How to solve a problem in parallel processing[34]

Parallel processing means running one or more programs simultaneously on multiple processors. In this way, a problem is divided into several parts so that each part can be solved simultaneously. Each segment is then transformed into a series of parallel commands, running in parallel on the processors (Fig. 1.b). In general, parallel processing means using at least two microprocessors in the same task. For this purpose, scientists divide a particular problem into several components by special software, then send each component to a dedicated processor. Next, each processor performs its task of solving the problem. Later, the software assembles the results to solve the initial complex problem [34]. The advantages of parallel processing include saving CPU time for problem-solving, solving large and complex problems in a fixed time interval, and performing multiple operations simultaneously. Another significant benefit is the use of non-local resources, such as computers on a network, which can dramatically increase processing performance.

4. PARALLEL PATTERN SEARCH ALGORITHM AND OPTIMIZATION

Optimization means achieving the best results in operation while satisfying certain constraints [35]. Mathematically, an optimization problem can be categorized as constrained and unconstrained problems. In terms of the types of variables, different categories are problems with continuous variables, problems with discrete variables, and problems with combined variables. A constrained optimization problem can be defined as follows:

$$
\begin{aligned}
& Find\ \tilde{x}^T = \{x_1, x_2, \ldots, x_n\} \\
& Minimize\ \{f(\tilde{x})\} \\
& Subjected\ \ to: \\
& g_i(\tilde{x}) \leq 0\ , i = 1, 2, 3, \ldots, n_g \\
& h_k(\tilde{x}) = 0\ , k = 1, 2, 3, \ldots, n_k \\
& \tilde{x}_{low} \leq \tilde{x} \leq \tilde{x}_{up}
\end{aligned}
\tag{6}
$$

where f is the objective function, $\tilde{x}$ is the vector of design variables, g_i is inequalities constraints, h_k is equality constraints, $\tilde{x}_{low}$ is the lower bound for the design variables, $\tilde{x}_{up}$ is the upper bound for the design variables, n_g is the number of inequalities, and n_k is the number of equality constraints.

To solve the optimization problem (1), a variety of methods have been proposed thus far. Pattern Search Algorithm is an effective search method in solving several engineering problems with a large number of objective function evaluations [36]. The basic idea of this method is to create a mesh around the last obtained optimal point and advance around that point to achieve a better result. For achieving this, the algorithm moves to the optimal point by changing the mesh size. The general steps of this algorithm for an ICU[1] can be considered as follows (Fig. 3.a):

1. The algorithm starts with X_0 point.
2. n Unit vectors are created to form the mesh.

$$\begin{aligned} \vec{M}_1 &= [1,0,0,\ldots,0]^t \\ \vec{M}_2 &= [0,1,0,\ldots,0]^t \\ &\ldots \\ \vec{M}_n &= [0,0,0,\ldots,1]^t \end{aligned} \tag{7}$$

3. Point X_0 is added to the grid vectors, and the value of the cost function is calculated for each point. In this way, meshing is formed as follows (Fig. 2).

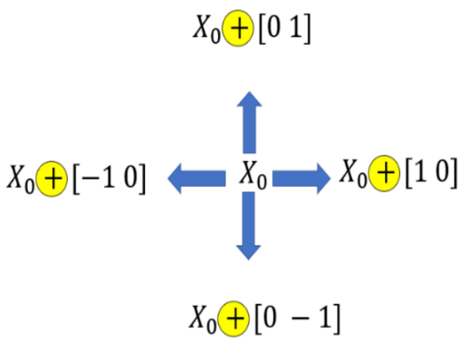


Figure 2. How to mesh in a pattern search algorithm with two variables

4. If we have a better value than the previous one, we have a successful poll. Then, the algorithm considers this point to be X_1. At this point, the mesh vectors are multiplied by an expansion coefficient (usually 2).

[1]Independent search unit

5. If there is no improvement over the current point in any meshing points, we have an unsuccessful poll. In this case, the mesh size is multiplied by a decreasing factor (usually 0.5).

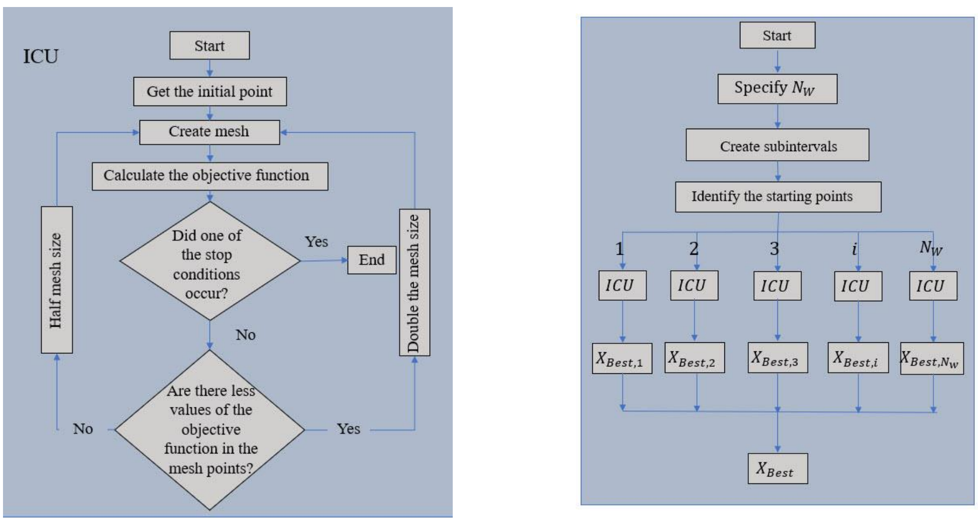
The stopping criteria in this algorithm are as follows (the numbers in parentheses correspond to the numerical example):

- If the mesh size is less than the allowed limit (10^{-6}).
- If the number of iterations exceeds the allowed limit (2000).
- If the distance between the point found in a successful poll to that found in the next successful poll is less than the specified limit (10^{-6}).

In the present study, the concept of parallelization is used to improve the performance of the pattern search algorithm. Accordingly, the search areas are divided into several subareas, and the starting point of each ICU unit is selected from that area. The parallel pattern search algorithm is described below (Fig. 3.b):

1. The number of parallel workers is determined (N_W).
2. The search intervals are divided into N_W subcategories.
3. From each subcategory, a random initial point is extracted.
4. Each starting point is sent to an ICU using a parallel processor.
5. The best output from ICUs is considered as the best point (X_{Best}).

The pattern search algorithm is designed for continuous variables. In this research, variables are considered as discrete. For this purpose, search intervals are regarded as vectors of discrete values, where the location of each variable in the interval is considered as the intermediate design variable. During the optimization process, this intermediate variable is rounded to the nearest possible location in the interval and determines the optimal value. For example, suppose $S=[s_1,s_2,...,s_y,...,s_n]$ is a discrete interval for the variable S. Then, the location of any possible value is considered as the intermediate variable (y).



(a) (b)

Figure 3. (a) Flowchart of an ICU unit. (b) Flowchart of a parallel pattern search algorithm

5. PROBLEM DESCRIPTION

This section describes the formulation of optimization problem for the I-girder composite steel-concrete bridges with skew angle and longitudinal slope. In the following subsections, definitions of parameters and variables, finite element modeling, problem-solving and constraint expression, objective function, and a numerical example are presented.

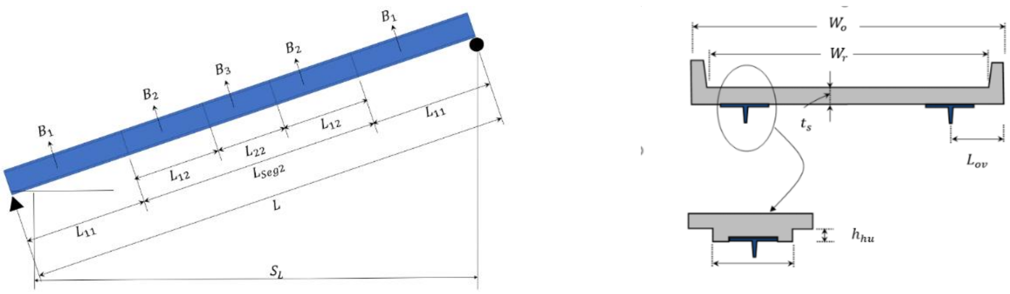
5.1 Parameters and variables

In this section, the parameters affecting the mathematical expression of a steel-concrete composite bridge problem are described. These parameters are associated with the problem dimensions, the material specifications and loads, and the control parameters' specifications. The parameters for the problem dimensions are provided in Table 1. The design variables associated with each parameter are shown alongside that parameter. It should be noted that steel beams are made of two segments and three sections. Segment 1 has a fixed cross-section, and segment two is composed of two different sections.

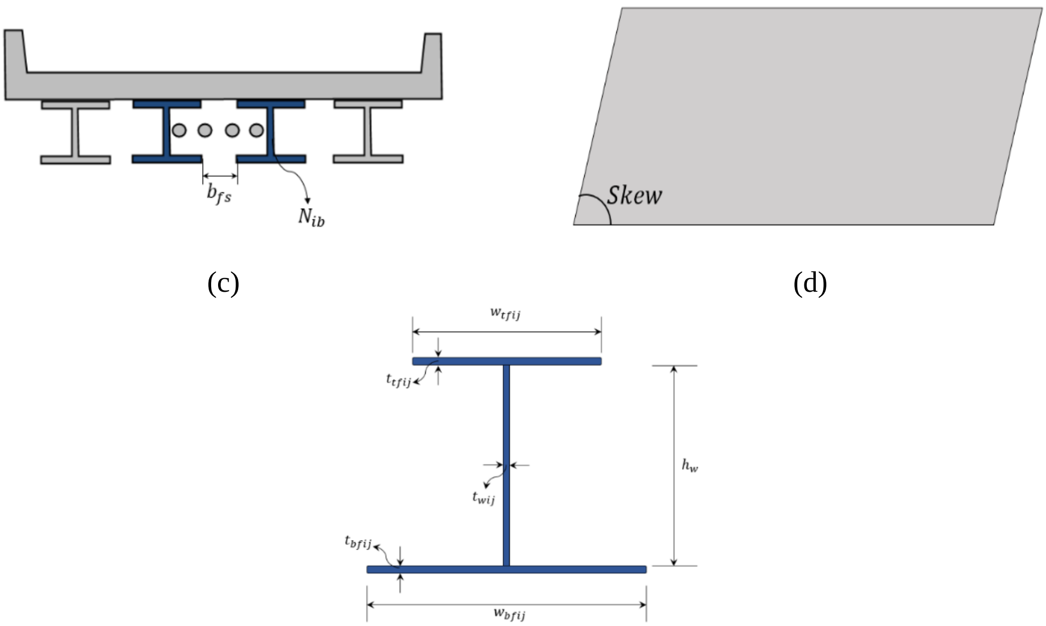
Table 1: Parameters related to the dimensions of the steel-concrete composite bridge problem

Parameter	Description	The value considered in the numerical example
S_L	Bridge span,Fig. 4.a	25.91 m (85 ft)
$Slope$	Longitudinal slope, Fig. 4.a	0.1
$Skew$	Skew angle, Fig. 4.d	75 deg
L_{ij}	Length of the cross-section ith in the segment jth, Fig. 4.a	-
W_0	Width of the deck, Fig. 4.b	12.8 m (42 ft)
W_r	Road width, Fig. 4.b	11.88 m (39 ft)
$h_{t,\max}$	Final permissible depth of bridge cross-section, Fig. 4.c	203.2 cm (80 in)
L_{ov}	Length of the balcony area, Fig. 4.b	.9 m (3 ft)
h_{hu}	Depth of hunch, Fig. 4.b	7 cm ($2\frac{3}{4}$ in)
$N_{ib}(x_1)$	Number of internal longitudinal beams, Fig. 4.c	-
$h_w(x_2)$	Depth of longitudinal beams web, Fig. 4.e	-
$t_{w11}(x_3)$	Flange thickness of beams, section 1, segment 1, Fig. 4.e	-
$t_{w12}(x_4)$	Flange thickness of beams, section 1, segment 2, Fig. 4.e	-
$t_{w22}(x_5)$	Flange thickness of beams, section 2, segment 2, Fig. 4.e	-
$\alpha_1(x_6)$	The parameter specifying the length of segment 2, Equation (8)	-
$\alpha_2(x_7)$	The parameter specifying the length of	-

	section 1, segment 2, Equation (8)	
$t_s(x_8)$	Thickness of concrete slab, Fig. 4.b	-
$w_{tf11}(x_9)$	Top flange width of section 1, segment 1, Fig. 4.e	-
$w_{tf12}(x_{10})$	Top flange width of section 1, segment2, Fig. 4.e	-
$w_{tf22}(x_{11})$	Top flange width of section2, segment 2, Fig. 4.e	-
$w_{bf11}(x_{12})$	Bottom flange width of section 1, segment 1, Fig. 4.e	-
$w_{bf12}(x_{13})$	Bottom flange width of section 1, segment 2, Fig. 4.e	-
$w_{bf22}(x_{14})$	Bottom flange width of section 2, segment 2, Fig. 4.e	-
$t_{tf11}(x_{15})$	Top flange thickness of section 1, segment 1, Fig. 4.e	-
$t_{tf12}(x_{16})$	Top flange thickness of section 1, segment 2, Fig. 4.e	-
$t_{tf22}(x_{17})$	Top flange thickness of section 2, segment 2, Figure 4.e	-
$t_{bf11}(x_{18})$	Bottom flange thickness of section 1, segment 1, Fig. 4.e	-
$t_{bf12}(x_{19})$	Bottom flange thickness of section 1, segment 2, Fig. 4.e	-
$t_{bf22}(x_{20})$	Bottom flange thickness of section 2, segment 2, Fig. 4.e	-
K_S	stiffness of elastomers	$193.1\,\frac{N}{m}\ (.5\,\frac{kip}{in})$
b_{fs}	Minimum free spacing between longitudinal beams, Fig. 4.c	25.4 cm (10 in)
$w_{f,sub}$	Flange width of the support beams	20.32 cm (8 in)
$t_{f,sub}$	Flange thickness of the support beams	2.54 cm (1 in)
$t_{w,sub}$	Web thickness of the support beams	1.43 cm ($\frac{9}{16}$ in)
$h_{w,sub}$	Web depth of the support beams	38.1 cm (15 in)



(a) (b)



(e)

Figure 4. Parameters of the steel-concrete composite bridge problem

The parameters for the properties of the materials and the loads are presented in Table 2:

Table 2: Parameters of material specifications and reinforced steel - concrete bridge problem

Parameter	Description	The value considered in the numerical example
g_c	The specific weight of concrete	$2.4\ \frac{t}{m^3}$ (150 pcf)
g_s	The specific weight of steel	$7.85\ \frac{t}{m^3}$ (490 pcf)
g_b	Specific weight of reinforcement bars	$7.85\ \frac{t}{m^3}$ (490 pcf)
w_{ws}	The specific surface coating weight	2.87 kpa (0.06 $\frac{kip}{ft^2}$)
q_{bw}	Weight of unit length of side barriers	30.65 kpa (0.64 $\frac{kip}{ft^2}$)
F_y	The yield stress of steel plates	344.74 mpa (50 ksi)
F_{yb}	The yield stress of reinforcement bars	413.69 mpa (60 ksi)
E_s	The elastic modulus of steel	199.95 Gpa (29000 ksi)
E_b	The elastic modulus of reinforcement bars	199.95 Gpa (29000 ksi)
E_c	The elastic modulus of concrete	24.82 Gpa (3600 ksi)
u_s	The poisson ratio of steel	0.3
u_b	The poisson ratio of reinforcement bars	0.3

u_c	The poisson ratio of concrete	0.2
w_{dll}	The amount of lane load	20.6 $\frac{N}{m}$ (0.64 $\frac{kip}{ft}$)
P_1	Front axle load of HL 93 truck, Figure 5.b	78.45 N (8 kip)
P_2	Middle axle load of HL 93 truck, Figure 5.b	313.81 N (32 kip)
P_3	Rear axle load of HL 93 truck, Figure 5.b	313.81 N (32 kip)
d_1	Front axle and middle axle distance of HL 93 truck, Figure 5.a	4.27 m (14 ft)
d_2	Middle axle and rear axle distance of HL 93 truck, Figure 5.a	4.27–9.14 m (14–30 ft)
d_3	Center to center distance of HL 93 truck wheels, Figure 5.c	1.83 m (6 ft)
d_4	HL 93 truck wheel center distance with lane edge, Figure 5.c	.61 m (2 ft)

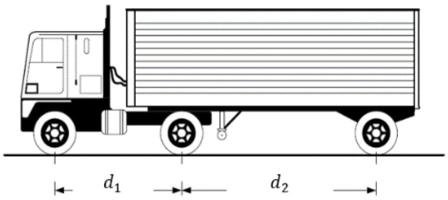
$$\alpha_1 = \frac{L_{Seg\,2}}{L} \tag{8}$$

$$\alpha_2 = \frac{L_{12}}{L_{Seg\,2}} \tag{9}$$

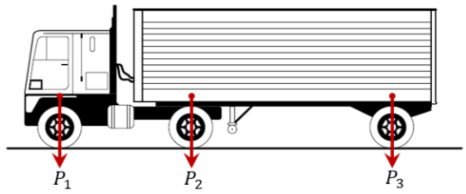
The specifications for the control parameters are presented in Table 3.

Table 3: Specifications of control parameters of steel-concrete composite bridge

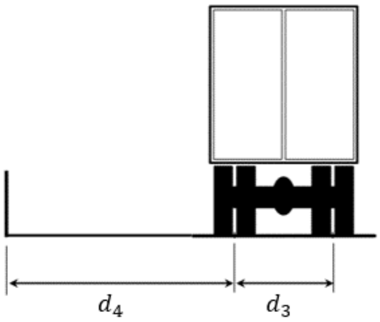
Parameter	**Description**	**The value considered in the numerical example**
IM_{LL}	Impact factor for live loads	.33
N_{sfw}	Number of stop stations for truck front wheel	5
N_{sbw}	Number of stop stations for truck rear wheel	3
$S_{u,dis}$	The distance between the side supports	2.59 m (8.5 ft)
HLS	This parameter is equal to 1 if using longitudinal stiffeners and otherwise equal to 0	0
MIM	The correction factor of concrete slab inertia moment	.25



(a)



(b)

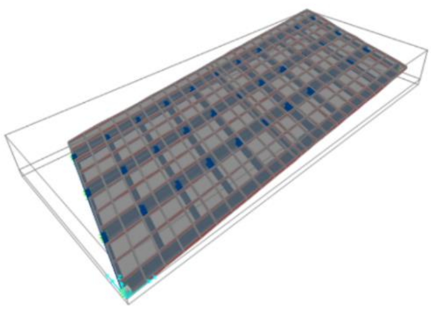


(c)

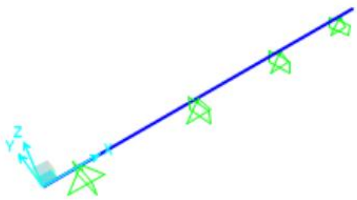
Figure 5. Specifications, dimensions, and installation of HL 93 truck

5.2 Finite elements model

Subprograms are used to model finite element components. They are used to calculate problem constraints. These subprograms deal with modeling the main slabs, the side slabs, the main beams, support beams, the rigid-links, supports, and main slab bars. By running these subprograms, the finite element model is automatically generated according to the problem parameters. Fig. 6.a and Fig. 6.b, illustrate the output of numerical example generated by these subprograms.



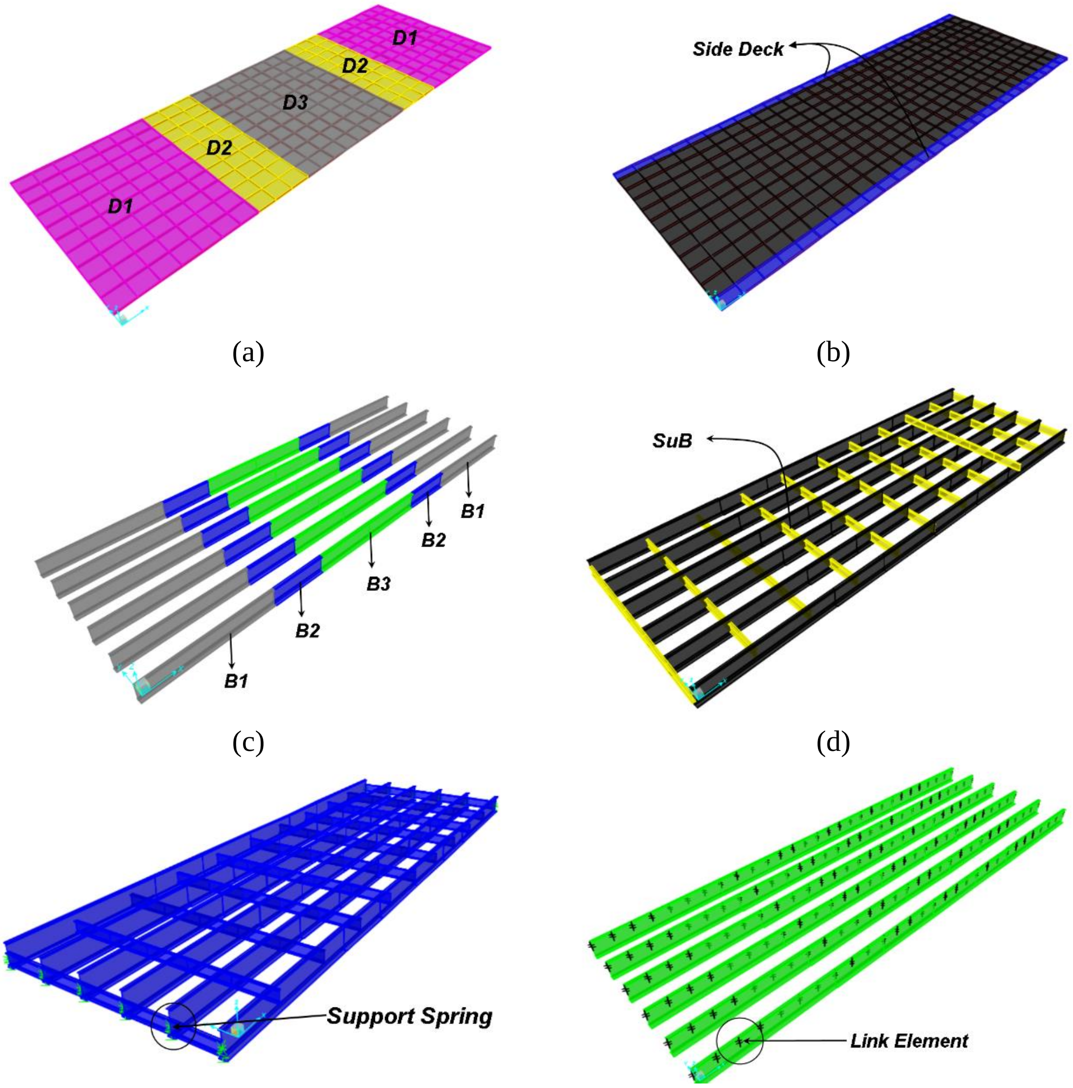
(a)



(b)

Figure 6. (a) 3D optimized bridge view in numerical example. (b) Simplified beam model for concrete rebar design

The main slab modeling subprogram creates the upper slab of the bridge deck using the plate elements in three groups D_1, D_2, D_3 (Fig. 7.a). The side slab modeling subprogram models the upper slab balcony area; it also groups the main side slabs into a separate group while applying the main slab grouping (Fig. 7.b). The subprogram used in modeling the main beams creates the main bridge deck beams; it also divides the beams into three groups B_1, B_2, B_3 (Fig. 7.c). According to the model parameters, in the beam modeling, the beam eccentricity is applied to the concrete slab. The subprogram in support-beam modeling creates a bridge deck support beam; it also places the beams in a SU_B group (Fig. 7.d).



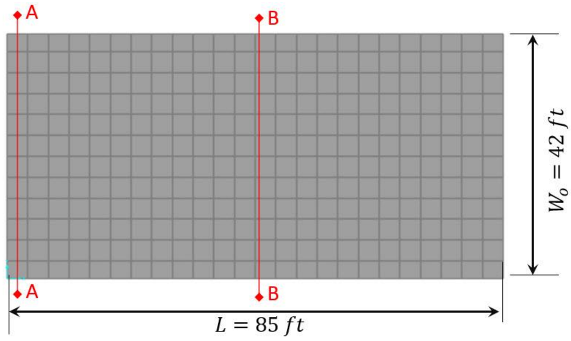
(a) (b) (c) (d) (e) (f)

Figure 7. The finite element model formed by the sub-programs for modeling: (a) the main slab. (b) the side slabs. (c) the main beams. (d) the support beams. (e) the bridge supports. (f) the rigid-link

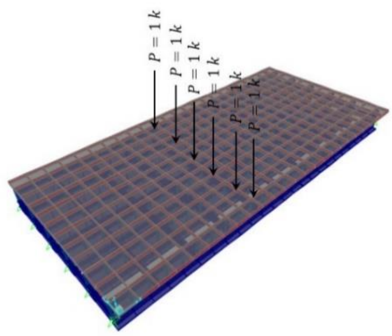
The support modeling subprogram models bridge supports automatically; it uses spring elements to model the effect of elastic supports on the problem. It can also model the supports as hinges (Fig. 7.e). The rigid -link modeling subprogram creates rigid-link elements to model the eccentricity between the deck and steel beams. These elements are made automatically between the concrete slab points and the longitudinal beams (Fig. 7.f). Concrete slab reinforcement bar subprogram calculating uses a hypothetical concrete beam transversely placed on simple supports with several longitudinal beams, this beam also has an effective width and depth equal to the slab thickness (Fig. 6.b).

In order to validate the finite element model created by these subprograms, a three-dimensional model of a bridge with the dimensions is created (Fig. 8.a.) The A-A and B-B cross sections are defined at the beginning and middle of the span, respectively. Two load cases are also considered.

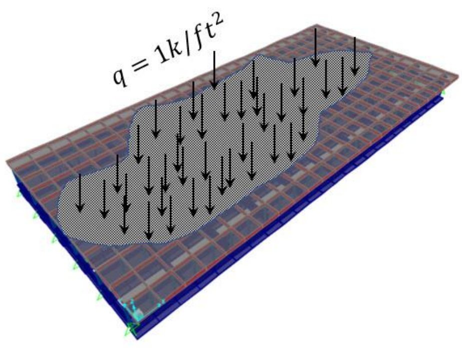
In the first case, six centralized loads (equal to the longitudinal beams in the model) are applied to the center of the span (Fig. 8.b). The value of these loads is $P = 1k$. In the second case, a distributed surface load is applied to all parts of the concrete slab (Fig. 8.c). The value of these loads is $q = 1kip / ft^2$.



(a)



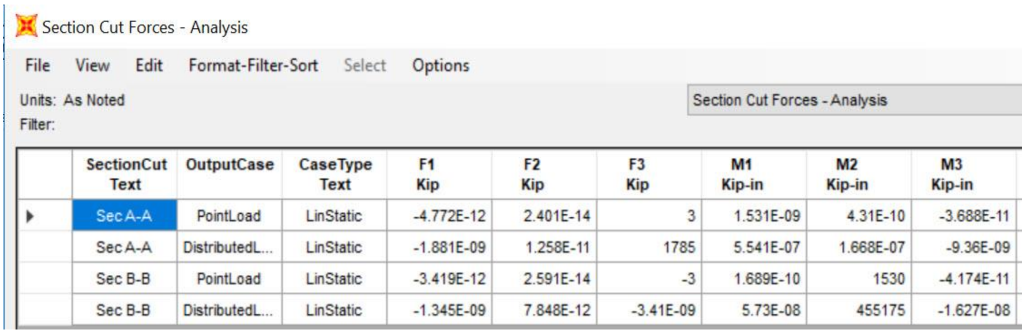
(b)



(b)

Figure 8. (a) 3D optimized bridge view in a numerical example, (b) Simplified beam model for concrete rebar design

Fig. 9 shows the output of SAP2000 software for A-A and B-B sections. Table 4 presents the outputs of the SAP2000 program and the results of the static equilibrium. It can be noted that the program correctly analyzes the bridge structure.



Section Cut Forces - Analysis

File View Edit Format-Filter-Sort Select Options

Units: As Noted

Filter:

Section Cut Forces - Analysis

SectionCut Text	OutputCase	CaseType Text	F1 Kip	F2 Kip	F3 Kip	M1 Kip-in	M2 Kip-in	M3 Kip-in
Sec A-A	PointLoad	LinStatic	-4.772E-12	2.401E-14	3	1.531E-09	4.31E-10	-3.688E-11
Sec A-A	DistributedL...	LinStatic	-1.881E-09	1.258E-11	1785	5.541E-07	1.668E-07	-9.36E-09
Sec B-B	PointLoad	LinStatic	-3.419E-12	2.591E-14	-3	1.689E-10	1530	-4.174E-11
Sec B-B	DistributedL...	LinStatic	-1.345E-09	7.848E-12	-3.41E-09	5.73E-08	455175	-1.627E-08

Figure 9. SAP2000 software outputs for A-A and B-B sections

Table 4: Comparison of SAP2000 software output and static equilibrium results

section	Load cases	Shear force in static equilibrium (k)	SAP2000 output shear force (k)	Shear force difference in percentage	Bending moment in static equilibrium $(k.in)$	SAP2000 Output Moment $(k.in)$	Moment difference in percentage
A-A	1	3	3	0	0	≈ 0	≈ 0
A-A	2	1785	1785	0	0	≈ 0	≈ 0
B-B	1	–3	–3	0	1530	1530	0
B-B	2	0	≈ 0	≈ 0	4.55×10^5	4.55×10^5	≈ 0

5.3 Problem solving and constraints

Three steps are considered to calculate the constraints of this problem. Before starting the calculations, the essential parameters are set. After adjusting the input parameters, the constraints related to this problem are calculated in three steps, using the AASHTO code [37]. In the first step, constraints related to flange and web plates proportion limits are considered. Calculations of these constraints do not require the finite element model of the structure. These constraints are divided into two groups: flange and web plates proportion limits. The constraints for the web plates are expressed as follows:

$$\frac{h_w}{rt_w} - 1 \le 0 \tag{1}$$

where $r = 300$ if longitudinal stiffeners are used, and otherwise $r = 150$.

The constraints for the flange plates are expressed as follows:

$$\frac{w_f}{24t_f} - 1 \le 0 \tag{2}$$

$$\frac{h_w}{6w_f} - 1 \le 0 \tag{3}$$

$$\frac{1.1t_w}{t_f}-1\le 0 \tag{4}$$

$$\frac{I_{yc}}{10I_{yt}}-1\le 0 \tag{5}$$

$$\frac{I_{yt}}{10I_{yc}}-1\le 0 \tag{6}$$

where w_f is the width of the flange, t_f is the thickness of the flange, t_w represents the thickness of the web, h_w the depth of the web, I_{yc} represents the moment of inertia of the tensile flange about the vertical axis of the cross-section, and I_{yt} is the moment of inertia of compression flange about the vertical axis of the cross-section.

$$\frac{h_t}{h_{t,\max}}-1\le 0 \tag{7}$$

$$\frac{.04L}{h_t}-1\le 0 \tag{8}$$

$$\frac{.033L}{t_3}-1\le 0 \tag{9}$$

$$\frac{b_{fs}}{S_b-w_{f,\max}}-1\le 0 \tag{10}$$

$$\frac{w_{tf}}{w_{bf}}-1\le 0 \tag{11}$$

where h_t represents the depth of the cross-section of the bridge, $h_{t,\max}$ is the upper limit for the depth of the cross-section of the bridge, L represents the bridge span length, t_3 is the depth of steel beam cross-section, b_{fs} is the free spacing between longitudinal beams, S_b is the distance from center to center of longitudinal beams, $w_{f,\max}$ is the upper limit for steel cross-section flanges, w_{tf} represents the width of top flange, and w_{bf} is width of bottom flange.

In the second step, the structural model is constructed without considering the concrete slab effect. This step indicates the state of the structure before concrete hardening. The concrete slab load is uniformly distributed over the longitudinal beams at this stage. The finite element model is used to determine the internal forces and the deformation. For this purpose, a three-dimensional model is developed using the SAP2000 software. To build this model, the parameters are first set in a MATLAB code; afterward, through the link to the SAP2000 software, a structural model is created using the defined parameters.

The load case used in this step is 1.5DC, where DC is the dead load on the structure (weight of steel beams and concrete slab). The constraints of this step are illustrated in the following equations:

$$R_{B1,2} - 1 \leq 0 \tag{12}$$

$$R_{B2,2} - 1 \leq 0 \tag{13}$$

$$R_{B3,2} - 1 \leq 0 \tag{14}$$

where $R_{B1,2}$ is the strength ratio of the B_1 group, $R_{B2,2}$ is the strength ratio of the B_2 group, and $R_{B3,2}$ represents the strength ratio of the B_3 group, all reported by the SAP2000 software in step 2.

In the third step, the structural model is constructed by considering the concrete slab effect. This stage indicates the status of the structure after the concrete hardening. In this step, all loads are applied to slab shell elements. Live loads are positioned at different places along the bridge to obtain the most critical state for calculating constraints. The synchronous effect of live loads is determined by the coefficient of the number of lanes [37]. This is intended for bridges with a crossing line of 1.2, two crossings equal to 1, three crossing lines equal to 0.85, and for those with more than three crossing lines equal to 0.65.

At this step, the load cases of 1.25DC+1.5DW+1.75(LL+IM) and 1.5DC+1.5DW are used, where DC is the dead load on the structure, DW is the dead loads representing surface weight, and LL is the live load (HL is the live load representing truck design, DLL is the line load), and IM is the live load impact factor of the HL-93 truck. The constraints of this step are illustrated in the following equations:

$$R_{B1,3} - 1 \leq 0 \tag{15}$$

$$R_{B2,3} - 1 \leq 0 \tag{16}$$

$$R_{B3,3} - 1 \leq 0 \tag{17}$$

$$\frac{800\Delta}{L} - 1 \leq 0 \tag{18}$$

where $R_{B1,3}$ is the strength ratio of the B_1 group, $R_{B2,3}$ represents the strength ratio of the B_2 group, and $R_{B3,3}$ is the strength ratio of the B_3 group, all reported by SAP2000 software in step 3, further, Δ is the magnitude of the displacement midpoints of the bridge span in step 3.

The following equation is also used to determine the number of lanes:

$$\#\,Design\ Lane\ Load = INT\left(\frac{W_r}{120\ ft}\right) \tag{19}$$

Bridges with widths from 20 to 24 ft should be designed for two lanes where the design load for each of which is $.5W_r$.

5.4 The objective function

To perform the optimization process, the most important steps are determining the design variables, the objective function, and the constraints. Design variables and problem constraints are presented in previous Sections. The objective function is defined as the cost of the bridge construction and is calculated by the following equation:

$$f(X) = \mathrm{Cos}\,t(X) + P(X) \tag{24}$$

where $\mathrm{Cos}\,t(X)$ is the final cost of constructing the bridge deck, and $P(X)$ is the penalty for the constraint.

The following equation is used to calculate $\mathrm{Cos}\,t(X)$:

$$\mathrm{Cos}\,t(X) = C_s W_s + C_c V_c + C_b W_b (p_f\ \mathrm{Cos}\,t(X))^{(1-acc)} \tag{25}$$

where C_s represents the price per unit weight of steel, C_c is the price per unit weight of concrete; C_b is the price per unit weight of reinforcement bars; W_b represents the weight of reinforcement bars used in the slab; W_s is the weight of steel used in beam construction; V_c denotes the volume of concrete used in bridge construction; acc is the parameter indicating an acceptable design of reinforcement bars; and p_f represents penalty coefficient.

To determine the number of required reinforcement bars at the top and bottom of the concrete slab, a continuous beam model positioned on simple supports with several longitudinal beams is modeled in SAP2000 software by a link. Since this software calculates the number of the rebars required at the top and bottom of the section, the cost of rebars is taken into account in the objective function. In this study, if the design of the bars is acceptable, $acc = 1$, otherwise $acc = 0$. The variables used in the above equations can be calculated as the following:

$$W_s = N_{ib} W_{beam} \tag{20}$$

$$\begin{aligned} W_{beam} = g_s \{ & L_{11}(w_{tf11}t_{tf11} + h_w t_{w11} + w_{bf11}t_{bf11}) + \ldots \\ & L_{12}(w_{tf12}t_{tf12} + h_w t_{w12} + w_{bf12}t_{bf12}) + \ldots \\ & L_{22}(w_{tf22}t_{tf22} + h_w t_{w22} + w_{bf22}t_{bf22}) \} \end{aligned} \tag{21}$$

$$V_c = W_0 L t_s \tag{22}$$

$$W_b = \left(\frac{L^2}{b_{s,eq}}\right)(A_t + A_b) g_b \tag{23}$$

where L_{11} is the length of cross-section 1 in segment 1; L_{12} is the length of cross-section 1 in segment 2; L_{22} denotes the length of cross-section 2 in segment 2; L is the bridge span; A_t is the maximum number of reinforcement bars required for the concrete slab at the top of cross-section; A_b is the maximum number of reinforcement bars required for the concrete slab at the bottom of the cross-section; and $b_{s,eq}$ is the effective width of the assumed concrete beam.

For the design of the reinforcement bars of the slab, the following equation is used [37]:

$$b_{s,eq} = \min\{26 + 6.6S_b, 48 + 3S_b\} \tag{30}$$

where S_b is the free distance between longitudinal beams in feet. The following equation is used to calculate $P(X)$:

$$P(X) = p_f \, \mathrm{Cos}\,t(X)\sum_{i=1}^{n_g} \max(0, g_i(X)) \tag{31}$$

Where $g_i(X)$ are constraints of the problem, and n_g is the number of constraints.

5.5 Numerical example

In this section, a numerical example of the bridge is presented. In solving this problem, the following parameters: $C_s = 1.56\$/kg$ (.7 \$/lb), $C_b = 1.56\ \$/kg$(.7 \$/lb), $C_c = 107.14\ \$/m^3(3\ \$/ft^3)$, $p_f = 10^6$ are considered. The discrete search ranges are provided in Table 5. The fixed parameters for this example are listed in Section 0.

Table 5: Search bands for discrete variables

Parameters	l_b	Δ_S	u_b
S_1	2	1	6
S_2	30.48 *cm* (12 *in*)	.635 *cm* (.25 *in*)	254 *cm* (100 *in*)
S_3	.1587 *cm* ($\frac{1}{16}$ *in*)	.1587 *cm* ($\frac{1}{16}$ *in*)	6.35 *cm* (2.5 *in*)
S_4	.2	.05	.8
S_5	20.23 *cm* (8 *in*)	1.27 *cm* (.5 *in*)	30.48 *cm* (12 *in*)
S_6	25.4 *cm* (10 *in*)	.635 *cm* (.25 *in*)	60.96 *cm* (24 *in*)
S_7	.1587 *cm* ($\frac{1}{16}$ *in*)	.1587 *cm* ($\frac{1}{16}$ *in*)	6.35 *cm* (2.5 *in*)

In Table 5, S_1 is the search range for the number of internal steel beams related to the variable x_1; S_2 is the search range for the depth of steel cross-sections related to variable x_2; S_3 represents the search range for the web thicknesses of steel cross-sections related to the variables x_3, x_4, x_5; S_4 is the search range for parameters α_1 and α_2 related to variables x_6, x_7; S_5 is the search range for concrete slab thickness related to variable x_8, S_6 is the search range for flange width related to the variables $x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}$, S_7 is the search range for flange thickness related to the variables $x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}$; l_b represents the lower bound of the search range; Δ_s is the amount of each variable change in the search range, and u_b denotes the upper bound of the search range.

Fig. 10 shows the CPU time of parallel and serial processing in 10 simulation times. On average, parallel processing reduces the CPU time by 19.6% compared to serial processing.

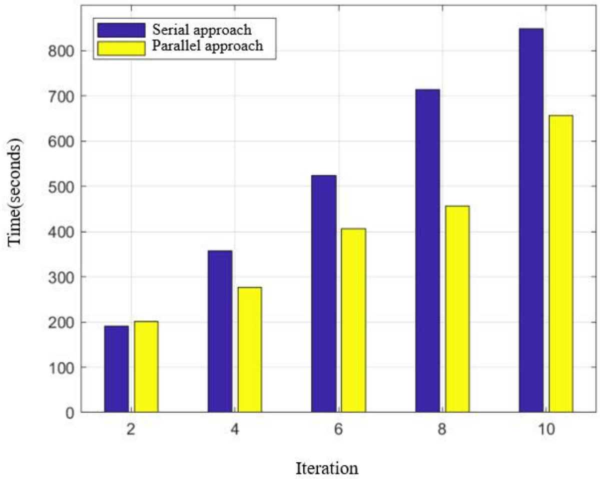


Figure 10. Comparison of time consumed in the serial approach versus parallel approach

This comparison is made by Intel® Core (TM) i5-6200U CPU@2.3GHz, a processor with two physical and four logic cores. Four parallel workers are used. According to Table 6, the best results are obtained on the parallel worker No. 1.

Table 6: Runs results on a parallel processor

CPU No	Cost function value $\times 10^7$\$	Iteration	Number of cost function evaluations
1	1.3221	171	2524
2	1.4793	105	1812
3	1.5273	145	2132
4	1.3743	196	2612

The optimal variables are presented in Table 7. According to the results, the second segment holds about 30% of the steel volume consumed, distributed almost equally between the two sections. The convergence curve of the objective function is shown in Fig. 11. The main constraints on the optimal design are displayed in Fig. 12 to Fig. 14.

Table 7: Optimal variables for bridge problem

Design variable	Value	Design variable	Value	Design variable	Value	Design variable	Value
$N_{ib}(x_1)$	2	$\alpha_1(x_6)$	.2	$w_{tf\,22}(x_{11})$	46.99 *cm* (18.5 *in*)	$t_{tf\,12}(x_{16})$	5.24 *cm* (2.0625 *in*)
$h_w(x_2)$	75.57 *cm* (29.75 *in*)	$\alpha_2(x_7)$	.8	$w_{bf\,11}(x_{12})$	59.06 *cm* (23.25 *in*)	$t_{tf\,22}(x_{17})$	4.76 *cm* (1.875 *in*)
$t_{w11}(x_3)$	.79 *cm* (.3125 *in*)	$t_s(x_8)$	21.59 *cm* (8.5 *in*)	$w_{bf\,12}(x_{13})$	45.72 *cm* (18 *in*)	$t_{bf\,11}(x_{18})$	4.13 *cm* (1.625 *in*)
$t_{w12}(x_4)$	.64 *cm* (0.25 *in*)	$w_{tf\,11}(x_9)$	37.47 *cm* (14.75 *in*)	$w_{bf\,22}(x_{14})$	46.69 *cm* (18.5 *in*)	$t_{bf\,12}(x_{19})$	5.24 *cm* (2.0625 *in*)
$t_{w22}(x_5)$	.64 *cm* (.25 *in*)	$w_{tf\,12}(x_{10})$	45.72 *cm* (18 *in*)	$t_{tf\,11}(x_{15})$	5.56 *cm* (2.1875 *in*)	$t_{bf\,22}(x_{20})$	5.56 *cm* (2.1875 *in*)

Required reinforcement bars are calculated by the SAP2000 at the top of the concrete slab $A_t = 18.61\ cm^2 (2.885\ in^2)$ and the bottom of the concrete slab $A_b = 16.557\ cm^2 (2.566\ in^2)$, for effective width $b_{s,eq} = 213.36\ cm (84\ in)$. The starting point of the calculation corresponding to the best results are presented in Table 8.

Table 8: The starting point of the calculation

Design variable	Value	Design variable	Value	Design variable	Value	Design variable	Value
$N_{ib}(x_1)$	3	$\alpha_1(x_6)$	.25	$w_{tf\,22}(x_{11})$	41.91 *cm* (16.5 *in*)	$t_{tf\,12}(x_{16})$	.16 *cm* (.0625 *in*)
$h_w(x_2)$	53.34 *cm* (21 *in*)	$\alpha_2(x_7)$	.4	$w_{bf\,11}(x_{12})$	28.58 *cm* (11.25 *in*)	$t_{tf\,22}(x_{17})$	2.22 *cm* (.875 *in*)
$t_{w11}(x_3)$	.19 *cm* (.75 *in*)	$t_s(x_8)$	24.13 *cm* (9.5 *in*)	$w_{bf\,12}(x_{13})$	34.93 *cm* (13.75 *in*)	$t_{bf\,11}(x_{18})$	1.59 *cm* (.625 *in*)
$t_{w12}(x_4)$	.16 *cm* (.0625 *in*)	$w_{tf\,11}(x_9)$	38.1 *cm* (15 *in*)	$w_{bf\,22}(x_{14})$	42.55 *cm* (16.75 *in*)	$t_{bf\,12}(x_{19})$	.16 *cm* (.0625 *in*)
$t_{w22}(x_5)$	2.86 *cm* (1.125 *in*)	$w_{tf\,12}(x_{10})$	36.83 *cm* (14.5 *in*)	$t_{tf\,11}(x_{15})$	1.11 *cm* (.4375 *in*)	$t_{bf\,22}(x_{20})$	3.02 *cm* (1.1875 *in*)

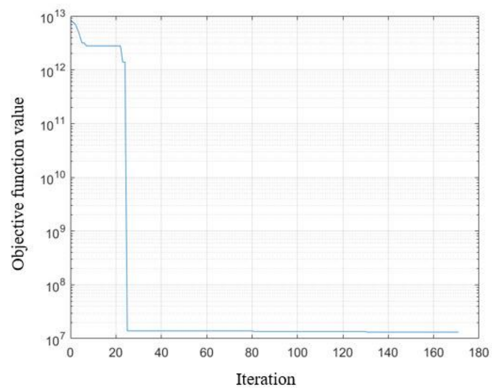


Figure 11. Convergence curve

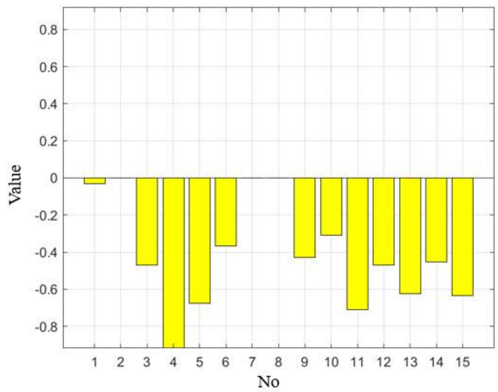


Figure 12. Operational constraints

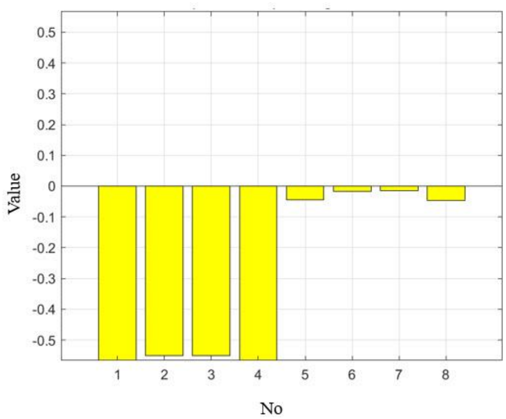


Figure 13. Strength constraints, second step, group B3

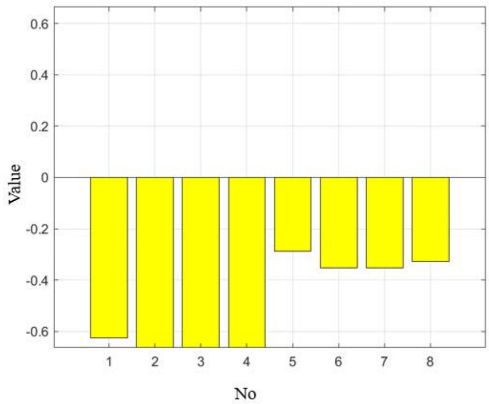


Figure 14. Strength constraints, third step, group B3

6. CONCLUSION

In this research, the optimization of the steel-concrete composite bridge is investigated. The parallel pattern search algorithm is used to perform the optimization process. Since the proposed algorithm starts to search at different places, it achieves better results in less time than serial processing. The results demonstrate that there is an average time improvement of 19.6% for the processor Intel® Core (TM) i5-6200U CPU@2.3GHz, with 4 parallel workers compared to serial processing. To perform the bridge deck analysis and design process, the link between MATLAB and SAP2000 is utilized using the SM toolbox.

The numerical example is a bridge with a longitudinal slope and skew angle. In this example, the active constraints are those of operating and component strength. According to the results, the second segment holds about 30% of the steel volume consumed, distributed almost equally between the two cross-sections. Due to the generality of the method presented in this study, the methodology presented in the article can be effectively used to optimize other types of structures.

REFERENCES

1. Javanmardi R, A Nedushan B. Optimization of I-shaped composite bridge decks with two and four crossing lines by genetic algorithm, In: *6th National and 2nd International Conference on New Materials and Structures in Civil Engineering*, Yazd, Iran, 2017.
2. Kaveh A, Zarandi MMM. Optimal design of steel-concrete composite I-girder bridges using three meta-heuristic algorithms, *Period Polytech Civ Eng* 2019; **63**(2): 317-37, doi:10.3311/PPci.12769.
3. Rana S, Islam N, Ahsan R, Ghani SN. Application of evolutionary operation to the minimum cost design of continuous prestressed concrete bridge structure, *Eng Struct* 2013; **46**: 38-48. doi:10.1016/j.engstruct.2012.07.017
4. Simões LMC, Negrão JHJO. Optimization of cable-stayed bridges with box-girder decks, *Adv Eng Softw* 2000; **31**(6): 417-23, doi:10.1016/S0965-9978(00)00003-X.
5. Guan H, Chen YJ, Loo YC, Xie YM, Steven GP. Bridge topology optimisation with stress, displacement and frequency constraints, *Comput Struct* 2003; **81**(3): 131-45. doi:10.1016/S0045-7949(02)00440-6.
6. Srinivas V, Ramanjaneyulu K. An integrated approach for optimum design of bridge decks using genetic algorithms and artificial neural networks, *Adv Eng Softw* 2007, **38**(7): 475-87. doi:10.1016/j.advengsoft.2006.09.016.
7. Cheng J, Jin C. Optimum design of steel truss arch bridges using a hybrid genetic algorithm, *J Constr Steel Res* 2010; **66**(8-9): 1011-17. doi:10.1016/j.jcsr.2010.03.007.
8. Baldomir A, Hernandez S, Nieto F, Jurado JA. Cable optimization of a long span cable stayed bridge in la Coruña (Spain), *Adv Eng Softw* 2010; **41**(7-8): 931-8. doi:10.1016/j.advengsoft.2010.05.001
9. Wei J, Huang Q, Chen B. Trial design of arch bridge of composite box section with steel web-concrete flange, *Front Archit Civil Eng China* 2010; **4**(3): 370-5, doi:10.1007/s11709-010-0073-7.
10. Lute V, Upadhyay A, Singh KK. Genetic algorithms-based optimization of cable stayed bridges, *J Softw Eng Appl* 2011; **04**(10): 571-8, doi:10.4236/jsea.2011.410066.
11. Makiabadi MH, Baghlani A, Rahnema H, Hadianfard MA. Optimal design of truss bridges using teaching-learning-based optimization algorithm, *Int J Optim Civil Eng* 2013; **3**(3): 499-510.
12. Yepes V, García-Segura T, Moreno-Jiménez JM. A cognitive approach for the multi-objective optimization of RC structural problems, *Arch Civil Mech Eng* 2015; **15**(4): 1024-36. doi:10.1016/j.acme.2015.05.001.
13. García-Segura T, Yepes V. Multiobjective optimization of post-tensioned concrete box-girder road bridges considering cost, CO2 emissions, and safety, *Eng Struct* 2016; **125**: 325-36. doi:10.1016/j.engstruct.2016.07.012
14. Martí JV, García-Segura T, Yepes V. Structural design of precast-prestressed concrete U-beam road bridges based on embodied energy. *J Clean Prod* 2016; **120**: 231-40, 231-40. doi:10.1016/j.jclepro.2016.02.024.
15. García-Segura T, Yepes V, Frangopol D, Yang D. Lifetime reliability-based optimization of post-tensioned box-girder bridges, *Eng Struct* 2017; **145**: 381-91.
16. Adeli H, Kim H. Cost optimization of composite floors using neural dynamics model, *Commun Numer Meth Eng* 2001; **17**(11): 771-87, doi:10.1002/cnm.448.

17. Kravanja S, Šilih S. Optimization based comparison between composite I beams and composite trusses, *J Constr Steel Res* 2003; **59**(5): 609-25, doi:10.1016/S0143-974X(02)00045-7.
18. Senouci AB, Al-Ansari MS. Cost optimization of composite beams using genetic algorithms, *Adv Eng Softw* 2009; **40**(11): 1112-18, doi:10.1016/j.advengsoft. 2009.06.001.
19. Kaveh A, Shakouri Mahmud Abadi A. Cost optimization of a composite floor system using an improved harmony search algorithm, *J Constr Steel Res* 2010; **66**(5): 664-9. doi:10.1016/j.jcsr.2010.01.009.
20. Kaveh A, Behnam AF. Cost optimization of a composite floor system, one-way waffle slab, and concrete slab formwork using a charged system search algorithm, *Sci Iran* 2012; **19**(3): 410-6. doi:10.1016/j.scient.2012.04.001.
21. Kaveh A, Bakhshpoori T, Ashoory M. An efficient optimization procedure based on cuckoo search algorithm for practical design of steel structures, *Int J Optim Civil Eng* 2012; **2**(1): 1-14.
22. Kennedy J, Eberhart RC. Swarm intelligence. In: *IEEE Technol Society Mag* 2002; **21**: 9-10. doi:10.1109/MTAS.2002.993595.
23. Gendreau M, Potvin JY. *Handbook of Metaheuristics* 2010; http://dl.acm.org/citation. cfm?id=1941310.
24. Kaveh A, Bakhshpoori T, Barkhori M. Optimum design of multi-span composite box girder bridges using Cuckoo Search algorithm, *Steel Compos Struct* 2014; **17**(5): 703-17. doi:10.12989/scs.2014.17.5.703.
25. Computers and Structures Inc (CSI). open application programming interface (OAPI). Published online 2011.
26. Kaveh A, Mirzaei B, Jafarvand A. Optimal design of double layer barrel vaults using improved magnetic charged system search, *Asian J Civil Eng* 2014; **15**(1): 135-55.
27. Kaveh A, Mirzaei B, Jafarvand A. Shape-size optimization of single-layer barrel vaults using improved magnetic charged system search, *Int J Civil Eng* 2014; **12**(4): 447-65.
28. Cai H, Aref AJ. On the design and optimization of hybrid carbon fiber reinforced polymer-steel cable system for cable-stayed bridges, *Compos Part B Eng* 2015; **68**: 146-52. doi:10.1016/j.compositesb.2014.08.031.
29. Martins AMB, Simões LMC, Negrão JHJO. Optimization of cable forces on concrete cable-stayed bridges including geometrical nonlinearities, *Comput Struct* 2015; **155**: 18-27. doi:10.1016/j.compstruc.2015.02.032.
30. Gocál J, Duršová A. Optimisation of transversal disposition of steel and concrete composite road bridges, *Procedia Eng* 2012; **40**: 125-30. doi:10.1016/j. proeng.2012.07.067.
31. Pedro RL, Demarche J, Miguel LFF, Lopez RH. An efficient approach for the optimization of simply supported steel-concrete composite I-girder bridges, *Adv Eng Softw*. 2017;112:31-45. doi:10.1016/j.advengsoft.2017.06.009.
32. CSi COMPUTERS & STRUCTURES.INC. https://www.csiamerica.com/
33. Javanmardi R, Ahmadi-Nedushan B. SM Toolbox, https://www.mathworks.com/ matlabcentral/fileexchange/79271-sap-matlab
34. ShahBahrami JS. *Parallel Processing and Programming by GPU*, Nas, 2016.

35. Haftka RT, GurdalL Z. *Elements of Structural Optimization: Third Edition (Google EBook)*, Springer Science & Business Media, 1992.
36. Booker AJ, Dennis JE, Frank PD, Serafini DB, Torczon V, Trosset MW. A rigorous framework for optimization of expensive functions by surrogates, *Struct Optim* 1999; **17**(1): 1-13. doi:10.1007/BF01197708.
37. *Movable Highway Bridge Design Specifications*, AASHTO LRFD, 2007.