Optimization and Engineering (2022) 23:643–687
https://doi.org/10.1007/s11081-021-09599-y

RESEARCH ARTICLE

Discrete variable optimization of structures subjected to dynamic loads using equivalent static loads and metaheuristic algorithms

Mustafa Al-Bazoon[1] · Jasbir S. Arora[2]

Received: 29 May 2020 / Revised: 7 November 2020 / Accepted: 22 January 2021 /
Published online: 16 February 2021

Abstract
This paper presents a new computational procedure for optimization of structures subjected to dynamic loads. The optimization problem is formulated with discrete design variables that represent the members from a table of commercially available members. Also, the requirements in the American Institute of Steel Construction (AISC) manual are formulated as constraints. This results in a nondifferentiable optimization problem. In the new procedure, the dynamic load is transformed into equivalent static loads (ESLs). Then the static response optimization problem having discrete design variables is solved using a metaheuristic optimization algorithm. Three methods to calculate the ESLs are investigated. It is found that the ESL cycles cannot converge to the final design. Therefore after a few ESL cycles, the original dynamic loads need to be used in the optimization process. Four example problems are solved to analyze the procedure. Based on this analysis, it is concluded that the new procedure is more efficient compared to a procedure that does not use the ESL cycles because it reduces the total CPU effort to obtain the final design. Also, better final designs are found. The reason is that many more designs are analyzed very efficiently with the ESL procedure.

Keywords Equivalent static loads method · Metaheuristic algorithms · Discrete variables · Structural optimization · Dynamic loads

List of symbols

α	Loading condition

✉ Mustafa Al-Bazoon
mustafa-jasim@uomisan.edu.iq

Jasbir S. Arora
jasbir-arora@uiowa.edu

1 The Department of Civil Engineering, College of Engineering, The University of Misan, Amarah, Maysan 62001, Iraq

2 Iowa Technology Institute, The University of Iowa, Iowa City, IA 52242, USA

β	Stress correction factor
δ	Lateral displacement
δ_{max}	Allowable lateral displacement
ϵ	Coefficient of restitution parameter
ϕ	Resistance factor or reduction factor
ψ	Exploration penalty coefficient
σ_L	Linear stress response
σ_N	Nonlinear stress response
θ	Rotation
θ_{max}	Allowable rotation
ε	Small number
ζ	Exponent penalty coefficient
CMS	Number of designs in **CM** matrix
D	Set of discrete values
DIF	Dynamic increase factor
F	Merit function
f	Cost function
F_u	Ultimate stress
F_y	Yield stress
$f_{penalty}$	Penalty function
g	Constraint function
$Iter$	Iteration
$Iter_{ESL}$	Iteration in the ESL step
k	Constraint number
L	Length of the member
l	Total number of constraints
m_i	Mass of the *i*th design
m_m	Mass of the *m*th colliding body of the moving group
M_n	Nominal flexural strength
m_s	Mass of the *s*th colliding body of the stationary group
M_u	Requires flexural strength
$MaxIter$	Limit on number of iterations
$MaxIter_{ESL}$	Limit on number of iterations in the ESL step
$MaxIter_{transient}$	Limit on number of iterations for ECBO with transient analysis
MK	Total number of memebers
N	Number of elements in the discrete set
n	Total number of loading conditions
NG	Total number of member groups
$nvar$	Total number of design vraibles
P_n	Nominal axial strength
P_u	Requires axial strength
Pro	Algorithmic parameter
rnp	Random number between 0 and 1
S_i	Section for design variable *i*
$S_{i,max}$	Heaviest section for design variable *i*
$S_{i,min}$	Lightest section for design variable *i*

SIF	Strength increase factor
t	Time
w	Weight per unit length
W_s	Total weight of the strcture
$\ddot{\mathbf{u}}$	Accelerations vector
$\dot{\mathbf{u}}$	Velocities vector
$\mathbf{CB}$	Colliding bodies matrix
$\mathbf{CB}_{ESL}$	Colliding bodies matrix in the ESL step
$\mathbf{CM}$	Colliding memory matrix
$\mathbf{CM}_{ESL}$	Colliding bodies matrix in the ESL step
$\mathbf{C}$	Damping matrix
$\mathbf{K}$	Stifness matrix
$\mathbf{K}_L$	Linear stifness matrix
$\mathbf{K}_N$	Nonlinear stifness matrix
$\mathbf{M}$	Mass matrix
$\mathbf{p}$	Loads vector
$\mathbf{rn}$	Diagonal matrix with diagonal elements as random numbers between − 1 and 1
$\mathbf{u}$	Dynamic displacements vector
$\mathbf{v}_m$	Velocity of the mth colliding body of moving group
$\mathbf{v}_s$	Velocity of the sth colliding body of stationary group
$\mathbf{X}$	Vector of design variables
$\mathbf{X}_m$	Location of the mth colliding body of moving group
$\mathbf{X}_s$	Location of the sth colliding body of stationary group
$\mathbf{z}$	Static displacement vector
CB	Colliding body
ECBO	Enhanced colliding bodies optimization
ESLM	Equivelent static load method
ESLs	Equivelent static loads
KKT	Karush–Kuhn–Tucker
MOESL	Metaheuristic optimization with equivalent static loads

1 Introduction

It is important to consider transient dynamic loads in the design process of many structures in engineering applications since many loads in the real-world act dynamically. At the same time, it is important to consider minimizing the total cost while achieving all the safety and performance requirements for structures (Arora 1999).

Optimization of structures subjected to dynamic loads using gradient-based algorithms includes calculating the gradients of all the problem functions provided the functions are differentiable. Several methods can be used to calculate the gradients such as: direct method, the adjoint method, and the modal approximation method (Kang et al. 2006). Then a gradient-based optimization algorithm is used to determine the design improvement by solving a subproblem. This process involves the integration of the equations of motion and sensitivity equations. Numerical integration of these equations is computationally expensive. Moreover, for some problem with material and/or geometrical nonlinearities, the numerical integration methods can have convergence difficulties. Therefore, it can be difficult to optimize structures subjected to dynamic loads in a mathematical optimization process (Kang et al. 2001). To overcome these difficulties, efforts have been made to transform the dynamic load into static loads.

One of the well-known dynamic to static loads transformation methods is based on the displacement field obtained using dynamic analysis of the structure (Kang et al. 2001). That is, the dynamic load is transformed into multiple equivalent static load sets. Then the equivalent static loads (ESL) are considered as multiple loading conditions in static response optimization process. This is called an ESL cycle of the optimization process. These cycles are repeated until a final design is obtained. More details of this process are provided in Sect. 5.

Mathematical foundation for the ESL method (ESLM) was presented for linear dynamic response optimization by Park and Kang (2003) . Stolpe (2014) suggested some changes to the stopping criterion in Park and Kang (2003) algorithm based on an analysis of the optimality conditions. Park and Lee (2019) made some changes to the ESLM based on the suggestions in Stolpe (2014). The theoretical validation showed that when the ESLM process terminated, the optimum solution satisfied the Karush–Kuhn–Tucker (KKT) necessary condition (first derivative test or first order necessary conditions). A solution to a general optimization problem must satisfy these conditions. For constrained problem, all constrains must satisfy the regularity condition (Arora 2017). Recently, Stolpe et al. (2018) applied the ESLM to topology optimization of a simple two bar truss having just one degree of freedom. The problem was formulated to minimize dynamic compliance subject to constraints on the cross-sectional areas and volume of the truss. For this two-variable problem, the true optimum solution is available by the graphical optimization method (Arora 2017). It was shown that ESLM converged to a design that did not meet the KKT conditions. It was suggested to further evaluate the ESLM on a larger set of topology optimization problems. It is noted that for class of problems considered in the present work, there are no optimality conditions to be satisfied at the final design. Therefore, the analyses presented by Stolpe (2014) and Stolpe et al. (2018) are not applicable for the present class of problems.

Calculus-based local optimization algorithms are applicable to continuous variables and differentiable functions. To solve a differentiable problem with discrete variables, many gradient-based optimization strategies are available (Arora 2017). One strategy is to initially treat the discrete variables as continuous (if possible) and then round-off their values at the optimum point to get their discrete values. With such an approach, the final solution may be infeasible or not optimum. Moreover, for some engineering problems with discrete variables, it is not possible to compute gradient information because the problem functions are not differentiable. Frame design optimization examples presented and discussed later in the paper are a class of problems where gradient-based optimization methods are not applicable.

Stochastic, metaheuristic or nature-inspired algorithms do not require gradient information, such as the well-known Genetic Algorithms (Goldberg and Holland 1988), Particle Swarm Optimization (Eberhart et al. 2001), Ant Colony

Optimization (Dorigo 1992), Harmony Search (Geem et al. 2001), and many others. In these algorithms, the search is not limited to a neighborhood of the current point, and the discrete variables and nondifferentiable functions can be treated routinely. They use random search in the whole design space instead of gradient-based search in a neighborhood of the current point. They are applicable for both continuous and discrete variables and with one or more objective functions. Also, they tend to converge to a global minimum (although there is no guarantee of this) for the problem instead of a local minimum as with the gradient-based methods. Since only the structural response is required in the optimization process, these methods can handle any kind of problems (linear, nonlinear, static, dynamic, differentiable, nondifferentiable). Like gradient-based optimization method, the computation cost of linear or nonlinear dynamic analysis is more than that for linear static analysis. Therefore, using metaheuristic algorithms could be impractical for dynamic response optimization problems since they generally require many structural analyses to reach the final design. Other discrete variable optimum design methods such as the branch and bound method can be used to solve problems with only a few design variables having a few allowable discrete values for the design variables. However, once design variables and discrete elements become large these methods required a tremendous number of simulations. Thus, metaheuristic algorithms are preferred because they can often offer a better trade-off between the number of simulations and the solution quality.

In this study, the ESLM for structures subjected to dynamic loads is investigated numerically with metaheuristic optimization algorithms and discrete design variables. This has not been investigated before in the literature. The problem functions are assumed to be nondifferentiable which is the case with some practical problems as discussed in the sequel. The idea is to investigate if the number of transient structural analyses and computational time required to reach the best design can be reduced compared to a procedure that does not use the ESLs. The method is named metaheuristic optimization with equivalent static loads (MOESL). That is, the dynamic load for linear or nonlinear transient problems will be transformed into multiple equivalent static load sets using the ESLM. Then the linear static response problem will be optimized using a metaheuristic optimization algorithm. These ESL cycles will be repeated as long as the design population keeps improving. Enhanced Colliding Bodies Optimization (ECBO) algorithm will be used as the metaheuristic algorithm (Kaveh and Mahdavi 2014a), although any other such algorithm may also be used.

ESLM with gradient-based optimization (when used for differentiable problems) obtains one solution at the end of an ESL cycle. That solution is used to generate new ESLs for the next cycle (Kang et al. 2001). Metaheuristic algorithms, however, deal with a population of designs. Therefore, at the end of an ESL cycle, there is a population of designs that has been improved based on the linear static analysis process. For the next cycle, only one design can be used to generate new ESLs. The question is which design from the population should be used to calculate ESLs? There are several possibilities for this. Three approaches are examined to select the design that is used to generate the ESLs for the next ESL cycle (see Sect. 7). Example problems are solved to evaluate these approaches and the ESLM with metaheuristic algorithms.

It is important to note that the present work presents a computational and statistical study of the performance of the ESLM for nondifferentiable and discrete variable structural optimization problems. The main purpose of the investigation is to determine if the enormous computational burden of solving this class of problems can be reduced by incorporating the ESLM in metaheuristic algorithms. Also, the quality of the final solution with ESLM will be assessed.

2 General statement for discrete variable structural optimization problem

In many practical design cases, design variables are discrete because members must be selected from the available sizes in a catalog. The formulation of the discrete design variables optimization problem is different from the continuous design variables optimization. In general, the nonlinear dynamic response optimization problem with discrete design variables is stated as:

$$\text{Find } \mathbf{X} = [x_1, x_2, , x_{nvar}];\ x_i \in D_i;\ i = 1, 2, \ldots, nvar \tag{1}$$

$$\text{to minimize } f(\mathbf{X}) \tag{2}$$

$$\begin{aligned} &\text{subject to } \mathbf{M}(\mathbf{X})\ddot{\mathbf{u}}(t) + \mathbf{C}(\mathbf{X})\dot{\mathbf{u}}(t) + \mathbf{K}(\mathbf{X}, \mathbf{u}(t))\mathbf{u}(t) = \mathbf{p}(t) \\ &g_k(\mathbf{X}, \mathbf{u}(t), \dot{\mathbf{u}}(t), \ddot{\mathbf{u}}(t), t) \le 0;\ \text{for all } t \text{ and } k = 1, 2, \ldots, l \end{aligned} \tag{3}$$

where $\mathbf{X}$ is the vector of design variables with $nvar$ unknowns, D_i is a set of discrete values for the ith design variable, $f(\mathbf{X})$ is a cost function (in this study, $f(\mathbf{X})$ is the total mass or weight of the structure), $\mathbf{M}$ is the mass matrix, $\mathbf{C}$ is the damping matrix, $\mathbf{K}$ is the stiffness matrix, $\mathbf{u}$ is the dynamic displacements vector, $\dot{\mathbf{u}}$ is the velocities vector, $\ddot{\mathbf{u}}$ is the accelerations vector, t is time, g_k is the kth constraint function that needs to be imposed at all time points, and l is the total number of constraints. The linear dynamic response problem is the same as the nonlinear dynamic response problem except that $\mathbf{K}$ is not a function of the displacement vector $\mathbf{u}$.

The linear static response optimization problem subjected to the α loading conditions can be stated as:

$$\text{Find } \mathbf{X} = [x_1, x_2, x_{nvar}];\ x_i \in D_i;\ i = 1, 2, \ldots, nvar \tag{4}$$

$$\text{to minimize } f(\mathbf{X}) \tag{5}$$

$$\begin{aligned} &\text{subject to } \mathbf{K}\mathbf{u}_{\boldsymbol{\alpha}} = \mathbf{p}(t) \\ &g_{k\alpha}(\mathbf{X}) \le 0;\ k = 1, 2, \ldots, l;\ \alpha = 1, 2, \ldots, n \end{aligned} \tag{6}$$

where n is the total number of loading conditions.

3 Formulation for optimization of steel frames

3.1 Design variables

For the truss design examples (Sects. 8.1.1 and 8.1.2), design variables are cross-sectional areas that are selected directly from the discrete set. For framed design examples (Sects. 8.2.1 and 8.2.2), the AISC (2017) W-shapes available in the manufacturers catalog are desired for beams and columns. Figure 1 shows an example of a 2-story 2-bay steel framed and the applied loads. The finite element model of the structure is shown in Fig. 1. This design example is a 2-story 2-bay planar steel frame having 4 beams and 6 columns (more details will be provided later). It represents the class of the problem that cannot be solved using gradient-based optimization algorithms. The design variables are classified as linked discrete variables (Arora 2017). That is, once the section number is known, all the cross-sectional properties are available from the tables for evaluating cost and constraint functions. To further explain the design variables, consider a small part of the AISC (2017) wide-flange sections table shown in Table 1. Once an integer value is assigned to a design variable, a section is specified. For example, if a design variable is assigned a value of 4, then the section from Table 1 is W44X230. For this section, the weight per foot is 230 lbs (104.3 kg), the cross-sectional area is 67.8 inch^2 (437.42 cm^2), the total depth is 42.9 inches (108.97 cm), and so on. In other words, all the cross-sectional properties are available to formulate and check the performance constraints.

This way the design variables in Eqs. (1) and (4) become:

$$\text{Find } \mathbf{X} = [S_1, S_2, , S_{nvar}] \tag{7}$$

$$S_{i,min} \le S_i \le S_{i,max};\ i = 1, 2, \ldots, nvar \tag{8}$$

where S_i is a section selected from the discrete set for design variable i, $S_{i,min}$ and $S_{i,max}$ are the lightest and the heaviest sections, respectively.

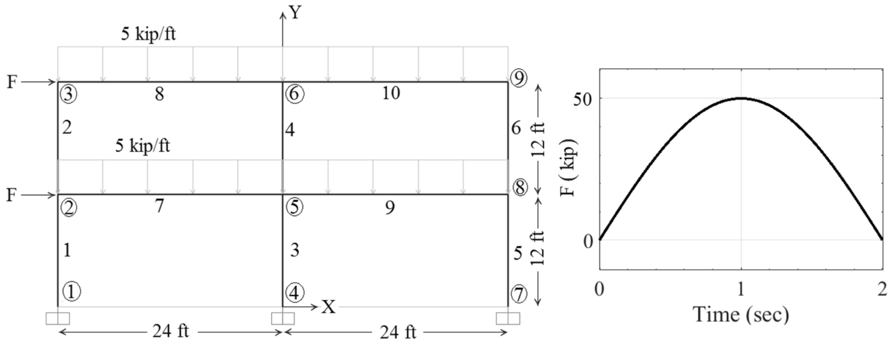


Fig. 1 Schematic of the 2-story 2-bay frame and the applied dynamic load

Table 1 ASIC W-shape database (partial)

Section number	Shape	W (lb/ft)	A (in^2)	d (in)	Web		Flange		Axis X-X				…	h_o (in)	P_A (in)	P_B (in)
					t_w (in)	$t_w/2$ (in)	b_f (in)	t_f (in)	I_x (in^4)	Z_x (in^3)	S_x (in^3)	r_x (in)				
1	W44X335	335	98.5	44.0	1.030	1/2	15.9	1.77	31100	1620	1410	17.8	…	42.2	132	148
2	X290	290	85.4	43.6	0.865	7/16	15.8	1.58	27000	1410	1240	17.8	…	42.0	131	147
3	X262	262	77.2	43.3	0.785	7/16	15.8	1.42	24100	1270	1110	17.7	…	41.9	131	147
4	X230	230	67.8	42.9	0.710	3/8	15.8	1.22	20800	1100	971	17.5	…	41.7	130	146
…	…	…	…	…	…	…	…	…	…	…	…	…	⋰	…	…	…
273	W4X13	13	2.83	4.16	0.280	1/8	4.06	0.345	11.3	6.28	5.46	1.72	…	97	495	599

1 lb = 0.453 kg. 1 in = 2.54 cm. 1 in^2 = 6.45 cm^2

3.2 Cost function

The problem is to minimize the total weight of the structure. Thus, Eqs. (2) and (5) become:

$$W_s(\mathbf{X}) = \sum_{ng=1}^{NG} w_{ng} \sum_{mk=1}^{MK} L_{mk} \tag{9}$$

where W_s is the total weight of the structure, $\mathbf{X}$ is the design vector, NG is the total number of member groups for the structure, w_{ng} is the weight per unit length of the members in the ngth group (available in AISCs tables), MK is the total number of members in the ngth group, and L_{mk} is the length of the mkth member.

3.3 Constraints

Restrictions imposed on the structural members are: displacement and strength constraints. These constraints are implicit functions of the design variables and are explained in the following subsections.

3.3.1 Displacement constraints

Two types of displacement constraints are imposed in this study:

1. *Maximum member end rotation*. The normalized form of this constraint is as follows:

$$\frac{|\theta_j|}{\theta_{max}} - 1 \leq 0 \tag{10}$$

   where θ_j is rotation at joint j, and θ_{max} is the allowed rotation.
2. *Maximum side-sway deflection* (or inter-story drift (*ISD*)). The normalized form of this constraint is as follows:

$$\frac{|\delta_r - \delta_{r-1}|}{\delta_{max}} - 1 \leq 0 \tag{11}$$

   where δ_r and δ_{r-1} are lateral displacements of two adjacent stories, and δ_{max} is the allowable lateral displacement.

3.3.2 Strength constraints

According to the AISC (2017), symmetric members subjected to axial force and bending must satisfy the interaction ratio strength requirement:

$$\begin{aligned}&\frac{P_u}{\phi P_n}+\frac{8}{9}(\frac{M_{uz}}{\phi_b M_{nz}})-1\le 0 \text{ if } \frac{P_u}{\phi P_n}\ge 0.2\\&\frac{P_u}{2\phi P_n}+\frac{M_{uz}}{\phi_b M_{nz}}-1\le 0 \text{ if } \frac{P_u}{\phi P_n}<0.2\end{aligned} \tag{12}$$

here ϕ is the resistance factor (ϕ_c = 0.85 and ϕ_t = 0.90 for compression and tension, respectively). ϕ_b=0.9 is the flexural resistance factor. P_u and P_n are the required and the nominal axial strengths (compression or tension), respectively. M_{uz} is the required flexural strengths. M_{nz} is the nominal flexural strengths. Constraints in Eq. (12) needs to be imposed at each point along the axis of every member in the structure. Thus, the equation represents infinite constraints. In the numerical process, the constraints are evaluated at several points along the axis of the member. These constraint values are then used to evaluate the penalty function.

Also note that the constraints in Eqs. (10) to (12) are functions of time. They are evaluated at all the time grid points and imposed there in numerical calculations.

In Eqs. (12), evaluation of P_n and M_{nz} is an involved process (AISC 2017) that requires checking of several failure modes (i.e., several if then else requirements). For example, to find P_n, first one needs to find whether the member force is tensile or compressive. For tension members, P_n is calculated based on whether the gross section yields or the net section ruptures. For compression members, P_n is calculated based on consideration of several failure modes, such as yielding of the material, local buckling of flanges or the web (elastic or inelastic), and global member buckling (elastic or inelastic). Similarly, calculation of M_{nz} involves checking several flexural failure modes. All the foregoing calculations involve various cross-sectional properties of the sections that are available in Table 1.

Thus, it is concluded that it is not possible to obtain a functional expression for the constraints in Eqs. (12) in terms of the design variables (the integer number of the sections). Even if that were somehow possible, there would be several discontinuities in the functions due to all the ”if then else” requirements mentioned in the foregoing paragraph. Also, notice that constraints in Eq. (12) have a discontinuity at $P_u/(\phi P_n) = 0.2$. Therefore, due to all these reasons, the gradient-based methods are not applicable to this class of applications.

4 Challenges for solving the problem

Section 3 shows that class of problems that require selecting sections from catalogs and imposing codes strength constraints (such as design of framed steel structures to satisfy AISC code requirements) involves noncontinuous design variables and non-differentiable constraints. That is, gradient-based optimization algorithms cannot be used. Also, the number of possible design combinations is extremely large based on the number of design variables and the size of allowable discrete sets. Therefore, integer programming methods such as the branch and bound method are not applicable for this class of problems. Thus, in this study, a metaheuristic algorithm is selected to solve the problem.

The drawback of most metaheuristic algorithms is that they require many simulations to obtain good designs. Therefore, using metaheuristic algorithms could be impractical for dynamic response optimization problems because one dynamic analysis might take long computation time depending on the size of the structure and structural nonlinearity. Therefore, the performance of ESLM with metaheuristic algorithms is evaluated in this study based on the quality of the final solution and computational effort.

It is noted here that computation of metaheuristic algorithms can be speeded up using parallel processing and multicore processors. However, this aspect is not evaluated in the present study.

5 Transformation of dynamic loads into equivalent static loads (ESLs)

In this section, we summarize the basic concepts and steps of the ESLM for continuous design variables using the gradient-based optimization algorithms (Kang et al. 2001). The dynamic response of a structure subjected to dynamic loads is described by the following differential equation obtained after a finite element model for the structure has been developed:

$$\begin{aligned}&\mathbf{M}(\mathbf{X})\ddot{\mathbf{u}}(t)+\mathbf{C}(\mathbf{X})\dot{\mathbf{u}}(t)+\mathbf{K}(\mathbf{K},\mathbf{u}(t))\mathbf{u}(t)=\mathbf{p}(t);\\ &t=t_1,t_2,\dots,t_n\end{aligned}\tag{13}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{K}$ is the stiffness matrix ($\mathbf{K}$ is a function of the design variables and displacement vector of nonlinear dynamic analysis and just the design variables for linear dynamic analysis), $\mathbf{C}$ is the damping matrix, $\mathbf{u}$ is the dynamic displacements vector, $\dot{\mathbf{u}}$ is the velocities vector, and $\ddot{\mathbf{u}}$ is the accelerations vector, $\mathbf{X}$ is the vector of design variables, $\mathbf{p}(t)$ is the applied loads vector, t is time (generally discretized for numerical integration), and n is the total number of the time steps.

Linear static analysis with the finite element method is described by the following equation:

$$\mathbf{K}_L\mathbf{z}=\mathbf{p}_s\tag{14}$$

where $\mathbf{K}_L$ is the linear stiffness matrix, $\mathbf{z}$ is the static displacements vector, and $\mathbf{p}_s$ is the external static loads vector. ESLs are static loads that generate the same displacement fields as from dynamic loads at a given design $\mathbf{X}$. Using Eq. (14), an ESL at an arbitrary time (t_α) is expressed calculated as follows:

$$\mathbf{p}_\alpha=\mathbf{K}(\mathbf{X})\mathbf{u}(t_\alpha);\ \alpha=1,2,\dots,n\tag{15}$$

This way, the dynamic load is transformed into n loading vectors, i.e., n loading conditions for static analysis of the structure.

Figure 2 describes the concept of ESLM. That is, after linear or nonlinear dynamic analysis of the structure, an equivalent load vector ($\mathbf{p}_\alpha$) is generated at each time step using Eq. (15). It is seen that for a given design $\mathbf{X}$, the linear static

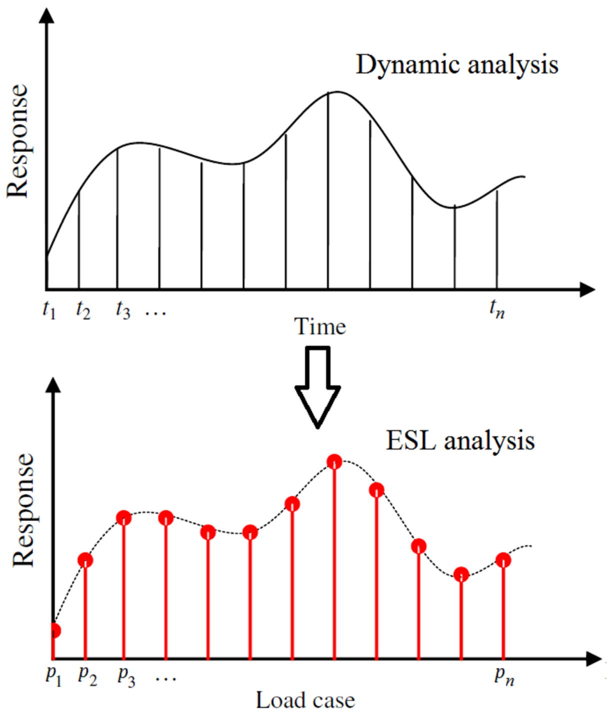


Fig. 2 Dynamic response vs ESL response for a given design (modified from Kim and Park (2010))

response from the αth load vector ($\mathbf{p}_\alpha$) is the same as the dynamic response at the αth time step. Therefore, the displacement profile of the dynamic response is the same as the displacement profile calculated from the linear static analysis (Kim and Park 2010). However, the profile of the ESLs is quite different from those of the dynamic loads because the ESLs are applied at each degree of freedom of the model even if the dynamic load is applied along only one degree of freedom. After the design is changed during the optimization iterations, the static and dynamic displacement profiles would be different because the ESLs are based on the starting design.

Optimum design of structures subjected to dynamic loads using the ESLs proceeds as shown in Fig. 3 and explained as follows (this will be called the ESLM (Park 2011)):

Step 1 Select an initial design for the structure. Perform dynamic analysis of the structure to generate the displacement profile $\mathbf{u}(t)$ using Eq. (13).

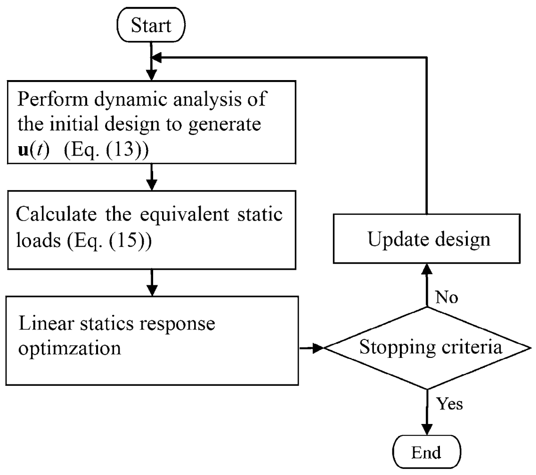
Step 2 Calculate the ESLs using Eq. (15).

Step 3 Perform static response optimization of the structure using ESLs calculated in Step 2. These loads are kept fixed during this optimization process. This is called an ESL cycle of the ESLM.

Step 4 Check the stopping criteria; if satisfied stop; otherwise continue.

Step 5 Since the final design from *Step 3* is different from the starting design, the static displacements will be different from dynamic displacements for the

Fig. 3 Optimization process with the ESLM (modified from Kim and Park (2010))


Start
Perform dynamic analysis of the initial design to generate **u**(*t*) (Eq. (13))
Calculate the equivalent static loads (Eq. (15))
Linear statics response optimzation
Stopping criteria
No
Update design
Yes
End

final design. Therefore, perform the dynamic analysis of the structure for the design obtained at the end of the ESL cycle and go to *Step 2*.

After a few cycles of the above process, the design changes are quite small such that the ESLs do not change much and a solution to the original dynamic response optimization problem is achieved.

6 Enhanced colliding bodies optimization (ECBO)

Metaheuristic optimization algorithms deal with unconstraint objective functions to improve designs. One way of treating constraints in metaheuristic algorithms is to combine constraints with the cost function to define a merit function $F(\mathbf{X})$ for linear or nonlinear dynamic response formulation (Eqs. (1) to (3)) that is then minimized:

$$F(\mathbf{X}) = f(\mathbf{X}) \times f_{penalty}(\mathbf{X}) \tag{16}$$

$$f_{penalty}(\mathbf{X}) = [1 + \psi \sum_{i=1}^{n} \sum_{k=1}^{l} max(0, g_k(t_i))]^{\zeta} \tag{17}$$

where $f_{penalty}(\mathbf{X})$ is the penalty function which is based on violations of the constraints for the problem, $\psi \geq 1$ is exploration penalty coefficient (in this study, $\psi = 1$), $\zeta > 1$ is penalty function exponent (in this study, $\zeta = 2$), and $max(0, g_k(t_i)) \geq 0$ is the violation value of the kth inequality constraint at the time point t_i.

For linear static formulation (Eqs. (4) to (6)), the merit function to be minimized, is:

$$F(\mathbf{X}) = f(\mathbf{X}) \times f_{penalty}(\mathbf{X}) \tag{18}$$

$$f_{penalty}(\mathbf{X}) = [1 + \psi \sum_{\alpha=1}^{n} \sum_{k=1}^{l} max(0, g_{k\alpha})]^{\zeta} \tag{19}$$

where $g_{k\alpha}$ is the kth inequality constraint of the αth loading condition. This penalty function magnifies the weight of the design if the constraints are violated; otherwise, its value is one. This type of constraints handling is well-known in metaheuristic optimization (Kaveh 2014).

Kaveh and Mahdavi (2014a) developed CBO that is inspired by the laws of the one-dimensional collision. The algorithm works with a population of designs at each iteration. The initial population is generated randomly, and the designs are stored in a matrix called the colliding bodies matrix (**CB**). Each design in the population is considered as an object or body having pseudo-mass that is calculated using the merit function value for each design as follows:

$$m_i = \frac{1/F_i(\mathbf{X})}{\sum_{k=1}^{2o} 1/F_k(\mathbf{X})}; i = 1, 2, ..., 2o \tag{20}$$

where m_i is the mass of the ith body (design), $F_i(\mathbf{X})$ and $F_k(\mathbf{X})$ are the merit function values of the ith and kth bodies (designs), respectively, and $2o$ is the total number of CBs or the population size. This way the designs are sorted from the best to the worst. Note that larger mass in Eq. (20) corresponds to a smaller value for the merit function.

The entire population is ranked and divided into moving objects and stationary objects as follows:

$$\mathbf{v}_s = 0; s = 1, 2, ..., o \tag{21}$$

$$\mathbf{v}_m = \mathbf{X}_m - \mathbf{X}_s; m = l + 1, ..., 2o \text{ and } s = m - o \tag{22}$$

where $\mathbf{v}_s$ and $\mathbf{X}_s$ are the velocity and position of the sth CB in the stationary group, respectively, and $\mathbf{v}_m$ and $\mathbf{X}_m$ are the velocity and position of the mth CB in the moving group, respectively.

Using the conservation law of linear momentum and the coefficient of restitution, one dimensional collision between the bodies is simulated. Based on that, new velocities of the stationary and moving objects are calculated as follows:

$$\acute{\mathbf{v}}_s = \frac{(1 + \epsilon)m_m \mathbf{v}_m}{m_m + m_s}; s = 1, 2, ..., o \text{ and } m = s + o \tag{23}$$

$$\acute{\mathbf{v}}_m = \frac{(m_m - \epsilon m_s)\mathbf{v}_m}{m_m + m_s}; m = o + 1, ..., 2o \text{ and } s = m - n \tag{24}$$

$$\epsilon = 1 - \frac{Iter}{MaxIter} \tag{25}$$

where $\acute{\mathbf{v}}_s$ is the velocity of the sth CB of stationary group after collision; $\acute{\mathbf{v}}_m$ is the velocity of the mth CB of the moving group after collision, respectively, m_s is the mass of the sth CB of the stationary group, m_m is the mass of the mth CB of the moving group, ϵ is the coefficient of restitution parameter, *Iter* is the current iteration of ECBO, and *MaxIter* is the limit on number of iterations for ECBO. Using these velocities and random numbers, each design in the population is updated as follows:

$$\mathbf{X}_s^{new} = \mathbf{X}_s + [\mathbf{rn}_s]\acute{\mathbf{v}}_s;\ s = 1, 2, ..., o \tag{26}$$

$$\mathbf{X}_m^{new} = \mathbf{X}_m + [\mathbf{rn}_m]\acute{\mathbf{v}}_m;\ m = o + 1, ..., 2o \tag{27}$$

where $\mathbf{X}_s^{new}$ and $\mathbf{X}_m^{new}$ are the new positions of the stationary and moving bodies, respectively, $\mathbf{X}_s$ and $\mathbf{X}_m$ are the old positions of the stationary and moving bodies, respectively, and $[\mathbf{rn}_s]$ and $[\mathbf{rn}_m]$ are diagonal matrices with diagonal elements as random numbers between -1 and 1. To obtain discrete values of designs, $\mathbf{X}_s^{new}$ and $\mathbf{X}_m^{new}$ are rounded to the nearest permissible discrete values. This process is repeated until a limit on the iterations is reached or there is very little change in the best design for several iterations.

In the enhanced version of the colliding bodies optimization (ECBO), a colliding memory matrix called **CM** is used to store some good designs. These designs replace the worst designs in the **CB** matrix at every iteration. This way the good designs are always preserved. In addition, a parameter $Pro \in [0, 1]$ is introduced that is used along with random numbers to regenerate a component of selected designs in the **CB** matrix. This mechanism is shown to give diversity to the design population leading to a better final design (Kaveh and Ghazaan 2014). The mathematical model of this step is as follows:

$$x_i^j = x_{j,min} + rnp \times (x_{j,max} - x_{j,min})\ if\ Pro > rnp_i;\ i = 1, ..., 2o \tag{28}$$

where x_i^j is the jth variable of the ith design, rnp is a random number between 0 and 1, and $x_{j,min}$ and $x_{j,max}$ are the lower and upper bounds of the jth variable, respectively.

Many metaheuristic algorithms need a selection of several algorithmic parameters in their calculations which is a major drawback because their specification determines the performance of these algorithms. ECBO, however, requires just one algorithmic parameter specification and performs well in term of the quality of solutions and convergence time. In addition, ECBO has been used to solve truss, frame, and other engineering optimization problems (Kaveh and Mahdavi (2014b, 2014c)). It has shown very good convergence behavior compared to other metaheuristic algorithms such as genetic algorithm, particle swarm, and harmony search (Kaveh and Mahdavi 2015). Therefore, this metaheuristic algorithm is selected for use in this study.

7 Discrete variable optimization using ESL for transient problems

As mentioned earlier, metaheuristic optimization algorithms search not only near the current design point but also in the entire design space. That is, small changes in design variables are not guaranteed which is an important assumption in the ESLM (at least near the local optimum point) with gradient-based optimization (Kang et al. 2001). Also, in ESLM with gradient-based optimization, there is one solution at the end of an ESL cycle. That solution is used to generate new ESLs for the next cycle. In metaheuristic algorithms, however, there is a population of designs at the end of an ESL cycle. Since most metaheuristic optimization algorithms deal with a population of designs, it is not obvious which design should be used to calculate the ESLs for the static response optimization cycle. Three approaches are proposed and investigated for calculating ESLs in this study:

1. The best design from static analysis ESL1.
   Design that has the lowest merit function value based on linear static analyses (the best design at the end of an ESL cycle) is used to generate ESLs for the next ESL cycle. In each cycle, only one dynamic analysis is needed in this approach.
2. The best design from dynamic analysis ESL2.
   It was observed that the best design at the end of an ESL cycle (which is based on linear static analyses) may not be the best design when a transient analysis is performed for the final population. Therefore, dynamic analyses are performed for designs in **CM** (4 designs in this study) and the first 25% of **CB** (the first 10 designs in this study). Just the first 25% of **CB** is used instead of the entire population because it is expected that the best design will be in this range. Then design that has the lowest merit function is used to generate ESLs for the next cycle. In each cycle, the number of dynamic analyses is 14 in this approach.
3. The heaviest feasible design from dynamic analysis ESL3.
   This approach is like ESL2 except that heaviest feasible design is used to generate ESLs for the next cycle. This design usually generates smaller ESLs values because the heavier structure is usually stiffer giving smaller displacements. If there is no feasible design (which usually happens in the first few cycles), ESL2 is used to generate ESLs for the next cycle. In each cycle, the number of dynamic analyses is 14 in this approach.

In the proposed algorithm, ESL method is used to transform the problem to linear static response optimization problem subjected to load cases that give the same displacement field as for the transient problem for the selected design (see

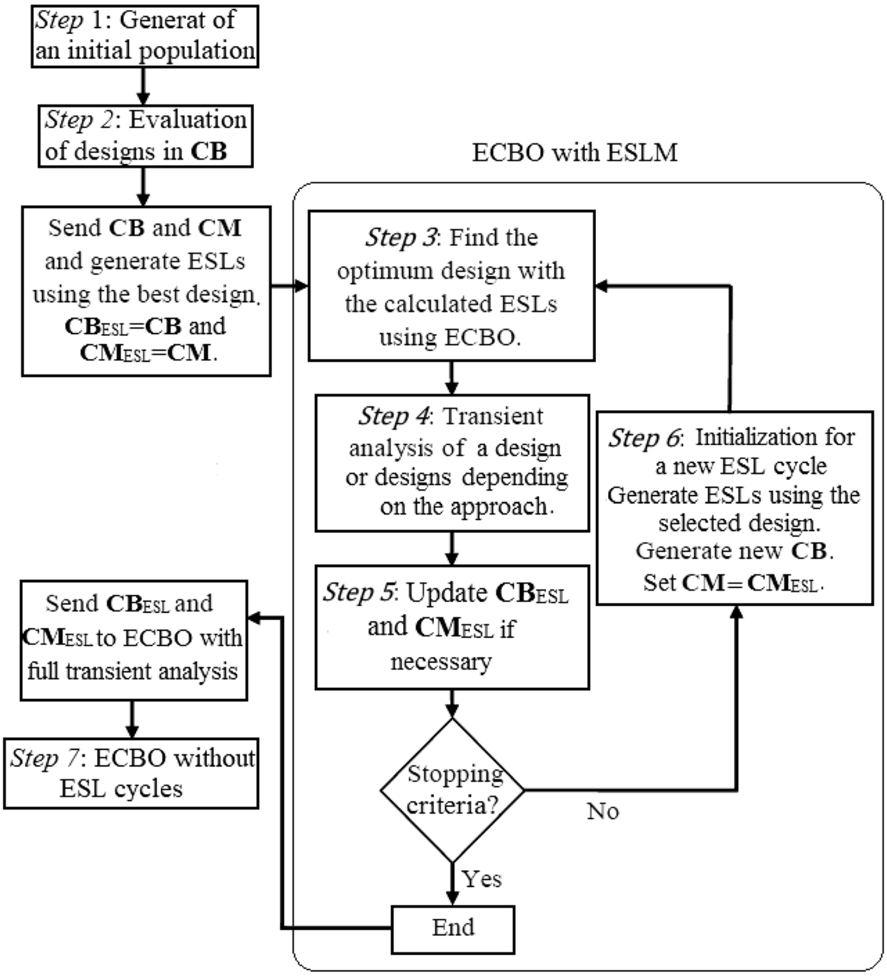

Step 1: Generat of an initial population
Step 2: Evaluation of designs in CB
Send CB and CM and generate ESLs using the best design. CBESL=CB and CMESL=CM.
ECBO with ESLM
Step 3: Find the optimum design with the calculated ESLs using ECBO.
Step 4: Transient analysis of a design or designs depending on the approach.
Step 5: Update CBESL and CMESL if necessary
Step 6: Initialization for a new ESL cycle Generate ESLs using the selected design. Generate new CB. Set CM=CMESL.
Stopping criteria?
No
Yes
End
Send CBESL and CMESL to ECBO with full transient analysis
Step 7: ECBO without ESL cycles

Fig. 4 Metaheuristic optimization with equivalent static loads process

Sect. 5). Then the linear static problem is optimized using ECBO. MOESL is explained in flowchart (Fig. 4), pseudocode (Algorithm 1), and as follows:

Algorithm 1. Pseudocode of MOESL.

```
Inputs: The population size, $MaxIter_{transient}$, $Pro$, minimum and maximum number of cycles, chose an approach
        (ESL1, ESL2, or ESL3), and Initialize the population $X_i (i = 1,2,...,2l)$.
Outputs: Best design ($\mathbf{X}_{Best}$) and its fitness value.
Evaluate design in CB, use the best design to generate ESLs, and save the best 4 designs in CM
Set $\mathbf{CB}_{ESL}$=CB and $\mathbf{CM}_{ESL}$=CM
Start ECBO with ESLM
while1 cycle< maximum number of cycles
    while2 $Iter_{ESL} < MaxIter_{ESL}$
        linear static optimization using the formulation given in Eqs. (4) to (6)
        if $(Iter_{ESL}) \geq 0.25 \times MaxIter_{ESL}$ and $\frac{Merit(Iter_{ESL}) - Merit(Iter_{ESL} - 0.1 \times MaxIter_{ESL})}{Merit(Iter_{ESL})} \leq 0.001$
            terminate the current cycle
        end if
    end while2
    ESL1:
    $\mathbf{X}_B$ =the best design based on linear static analyses
    if $F(\mathbf{X}_B) < F(\mathbf{X}_{\mathbf{BCM}_{ESL}})$     transient analysis
        $\mathbf{CB}_{ESL}$=$\mathbf{CB}_C$ and $\mathbf{CM}_{ESL}$=$\mathbf{CM}_C$
    end if
    generate ESLs for the next cycle using $\mathbf{X}_B$

    ESL2:
    $\mathbf{X}_B$ =the best of CM and 25% of CB based on transient analysis
    if $F(\mathbf{X}_B) < F(\mathbf{X}_{\mathbf{BCM}_{ESL}})$     transient analysis
        $\mathbf{CB}_{ESL}$= $\mathbf{CB}_C$
        $\mathbf{CM}_{ESL}$=Best four design from (previous $\mathbf{CM}_{ESL}$, $\mathbf{CM}_C$, Best 10% of $\mathbf{CB}_C$)
    end if
    generate ESLs for the next cycle using $\mathbf{X}_B$

    ESL3:
    $\mathbf{X}_B$ =the best of CM and 25% of CB based on transient analysis
    if $F(\mathbf{X}_B) < F(\mathbf{X}_{\mathbf{BCM}_{ESL}})$     transient analysis
        $\mathbf{CB}_{ESL}$= $\mathbf{CB}_C$
        $\mathbf{CM}_{ESL}$=Best four design from (previous $\mathbf{CM}_{ESL}$, $\mathbf{CM}_C$, Best 10% of $\mathbf{CB}_C$)
    end if
    if the heaviest design of CM and 25% of CB based on transient analysis is feasible
        generate ESLs for the next cycle using this design
    else if
        generate ESLs for the next cycle using the best of CM and 25% of CB based on transient analysi
    end if
    if cycle> 5 and $F(\mathbf{X}_{Best}(\text{cycle})) \geq F(\mathbf{X}_{Best}(\text{cycle}-1))$ and $F(\mathbf{X}_{Best}(\text{cycle})) \geq F(\mathbf{X}_{Best}(\text{cycle}-2))$
        terminate the ESL method
    else if
        Generate new CB
    end
end while1
End of ECBO with ESLM
set CB= $\mathbf{CB}_{ESL}$ and CM= $\mathbf{CM}_{ESL}$
ECBO with full transient analysis
while $Iter < MaxIter_{tansient}$
    use the formulation given in Eqs. (1) to (3)
end while
```

$\mathbf{X}_{\mathbf{BCM}_{ESL}}$is the best design in $\mathbf{CM}_{ESL}$

$\mathbf{CB}_C$ and $\mathbf{CM}_C$ are the colliding bodies matrix and colliding memory matrix of the current cycle, respectively.

Step 1 Generation of an initial population.

A population of designs is randomly generated from the design domain and saved in the **CB** matrix.

Step 2 Evaluation of designs in **CB**.

In this step, all designs in **CB** are analyzed using a transient solver. Using the simulation results and Eqs. (16) and (17), the merit function $F(\mathbf{X})$ is calculated for each design (Eq. (18)). Then the designs are arranged in an ascending order based on their merit function values. The colliding memory matrix **CM** is generated. The best design of the population is used to generate the ESLs; **CB** and **CM** are passed to ECBO with ESLM block in Fig. 4. Also, two matrices $\mathbf{CB}_{ESL}$ and $\mathbf{CM}_{ESL}$ are set to **CB** and **CM**, respectively. These two matrices save the population that has the best design (so far) at the end of ECBO with ESLM, $\mathbf{CB}_{ESL}$ and $\mathbf{CM}_{ESL}$ will be passed to ECBO without the ESLM block instead of last cycle **CB** and **CM**.

Step 3 Optimum design with the calculated ESLs.

Using linear static analyses of the structure, optimum design is found with the formulation given in Eqs. (4) to (6). This completes a cycle of the ESLM. The termination criteria for one ESL cycle are as follows:

```
If_1 $Iter_{ESL} \geq 0.25 \times MaxIter_{ESL}$
If_2 $\dfrac{Merit(Iter_{ESL}) - Merit(Iter_{ESL} - 0.1 \times MaxIter_{ESL})}{Merit(Iter_{ESL})} \leq \epsilon$
  Terminate the current cycle
End_2
End_1
```

$$MaxIter_{ESL} = 0.5 \times MaxIter_{transient} \tag{29}$$

$$MaxIter_{transient} = \sum_{i=1}^{nvar} N_i \tag{30}$$

where $Iter_{ESL}$ is the current iteration, $MaxIter_{ESL}$ is the limit on number of iterations for the ESL cycle, ε is a small number (in this study $\varepsilon = 10^{-3}$), N_i is the number of elements in the discrete set D_i, and $nvar$ is number of design variables. That is, when there is no or small improvement in the current merit function value after many iterations, the current ESL cycle is terminated.

Step 4 Transient analysis of final design(s).

Perform transient analysis of a design or multiple designs depending on ESL1, ESL2, or ESL3 approach used.

Step 5 Updating $\mathbf{CB}_{ESL}$ and $\mathbf{CM}_{ESL}$.

In this step, $\mathbf{CB}_{ESL}$ and $\mathbf{CM}_{ESL}$ matrices are updated depending on the approach as follows:

1- ESL1:

If *the transient analysis for the best design from static analysis at the end of an ESL cycle shows this design to be better than the best design in* $\mathbf{CM}_{ESL}$, *update* $\mathbf{CB}_{ESL}$ *and* $\mathbf{CM}_{ESL}$ *as follows*:

$$\begin{aligned} \mathbf{CB}_{ESL} &= \mathbf{CB}(\text{colliding bodies matrix of the current ESL cycle}) \\ \mathbf{CM}_{ESL} &= \mathbf{CM}(\text{colliding bodies matrix of the current ESL cycle}) \end{aligned} \tag{31}$$

Else *do not update* $\mathbf{CB}_{ESL}$ and $\mathbf{CM}_{ESL}$.

2- ESL2: in this approach, the design that has the lowest merit function is used to generate ESLs for the next cycle (as described above).

If *the design that has the lowest merit function is better that the best design in* $\mathbf{CM}_{ESL}$, *update* $\mathbf{CB}_{ESL}$ *and* $\mathbf{CM}_{ESL}$ *as follows:*

$$\begin{aligned} \mathbf{CB}_{ESL} &= \mathbf{CB}(\text{colliding bodies matrix of the current ESL cycle}) \\ \mathbf{CM}_{ESL} &= \text{best 4 designs from}\{\text{ previous } \mathbf{CM}_{ESL} \text{ (4 designs)}, \\ &\quad \mathbf{CM} \text{ (4 designs) of the current ESL cycle, or 25\% of} \\ &\quad \mathbf{CB} \text{ of the current ESL cycle (10 designs)}\} \end{aligned} \tag{32}$$

Else *do not update* $\mathbf{CB}_{ESL}$ and $\mathbf{CM}_{ESL}$.

3- ESL3: in this approach, the heaviest feasible design is used to generate ESLs for the next cycle (as described above).

If *the design that has the lowest merit function is better that the best design in* $\mathbf{CM}_{ESL}$, *update* $\mathbf{CB}_{ESL}$ *and* $\mathbf{CM}_{ESL}$ *as follows:*

$$\begin{aligned} \mathbf{CB}_{ESL} &= \mathbf{CB}(\text{colliding bodies matrix of the current ESL cycle}) \\ \mathbf{CM}_{ESL} &= \text{best 4 designs from}\{\text{ previous } \mathbf{CM}_{ESL} \text{ (4 designs)}, \\ &\quad \mathbf{CM} \text{ (4 designs) of the current ESL cycle, or 25\% of} \\ &\quad \mathbf{CB} \text{ of the current ESL cycle (10 designs)}\} \end{aligned} \tag{33}$$

Else *do not update* $\mathbf{CB}_{ESL}$ and $\mathbf{CM}_{ESL}$.

This way, the population that generates the best design ($\mathbf{CB}_{ESL}$) and the best designs saved from cycle to cycle ($\mathbf{CM}_{ESL}$) are passed to ECBO at the end of ESLM.

To terminate the ESL method, the following criterion is used (note that the minimum number of ESL cycles is set to 5): no better design is found for two ESL cycles. The stopping criteria are checked at this stage; if satisfied, the ESL method is terminated and we go to the ECBO with full transient analyses (*Step 7*); otherwise, we continue to *Step 6*.

Step 6 Initialization for a new ESL cycle.

In this step, new ESLs are re-calculated based on ESL1, ESL2, or ESL3 approach, and a new population of designs is generated from the design domains in the $\mathbf{CB}_{ESL}$ matrix. Analysis of results from some preliminary runs shows that the convergence behavior of MOESL is better when new $\mathbf{CB}_{ESL}$ matrix is generated at the beginning of each ESL cycle compared to passing the last $\mathbf{CB}_{ESL}$ to the next cycle. The updated $\mathbf{CM}_{ESL}$ is passed to the next cycle as $\mathbf{CM}_{ESL}$ to keep improving the best designs obtained from previous cycles. This way, new designs are explored by generating new $\mathbf{CB}_{ESL}$ when the best designs (so far) are saved by setting $\mathbf{CM} = \mathbf{CM}_{ESL}$.

Step 7 ECBO without ESL cycles.

If the stopping criteria for the ESL step are satisfied, $\mathbf{CB}_{ESL}$ matrix and the $\mathbf{CM}_{ESL}$ are passed to ECBO with full transient analyses. These two matrices have improved designs using ECBO with ESLM. Then, the formulation given in Eqs. (1) to (3) is used to find the final best design. It was found that with just the ESL cycles, the algorithm could not reach the best design. Therefore, *Step 7* was necessary to further improve the design.

In nonlinear dynamic problems, ESLs generate the same displacements as those from the nonlinear dynamic analysis; however, they do not generate the same stress responses because of the nonlinear relationship between stress and strain and strain and displacement (Kim and Park 2010). Therefore, when there are stress constraints, the difference in stresses can be adjusted to $\bar{\sigma}_{L_\alpha}$ as follows:

$$\begin{aligned} \beta_{\alpha,i} &= \frac{\sigma_{N_{\alpha,i}}}{\sigma_{L_{\alpha,i}}} \\ \bar{\sigma}^{j}_{L_\alpha} &= \sigma^{j}_{L_{\alpha,i}} \times \beta_{\alpha,i}; \alpha = 1, 2, \ldots, n \end{aligned} \tag{34}$$

where β is the stress correction factor, $\sigma_{L_{\alpha,i}}$ and $\sigma_{N_{\alpha,i}}$ are the linear and nonlinear stress responses from Eqs. (15) and (13), respectively, i is the element number, and j is the iteration number. This procedure is used in nonlinear truss design example (Sect. 8.1.2).

8 Numerical examples

In the following sections, four discrete structural optimization examples are solved for minimum structural mass or weight to test the performance of the proposed algorithm. ECBO and the first two design examples (truss structures) are coded using MATLAB and the models and simulation are verified using the commercial finite element analysis program ANSYS (Bhatti 2006). These two examples are solved in Choi and Park (2002) and Kim and Park (2010) using gradient-based algorithms and continuous design variables. In this study, they are used to test the proposed algorithm. The frame design examples are coded in MATLAB and interfaced with the structural analysis program SAP2000 using the Open Application Programming Interface (OAPI). SAP2000 provides analysis and design tools that are easy to use, however, any other software having similar capabilities can be used. The frame design examples are solved for the first time in this study. They represent the class of problem that cannot be solved using gradient-based algorithms (as described in Sects. 3 and 4).

The first numerical example is solved using the two simultaneous single-step Runge-Kutta method (ODE23 MATLAB function). For the rest of the examples, Newmarks method ($\beta = 1/4$ and $\alpha = 1/2$, the implicit and unconditionally stable method (Paz and Kim 2019)) is used for linear and nonlinear dynamic analysis while the direct stiffness method is used for linear static analysis.

ECBO parameters are set as follows: population size is 40, *Pro* is 0.4, and the number of designs to be saved in **CM** (*CMS*) is 4 (10% of the population). For all design examples, the time duration for dynamic analysis is set so that the maximum response is covered.

Since the optimization algorithms are stochastic in nature, 10 independent optimization runs were performed for each case to test the performance of ECBO with ESL. In each individual run, the initial population was the same for ECBO without the ESL cycles and for MOESL to make a fair comparison.

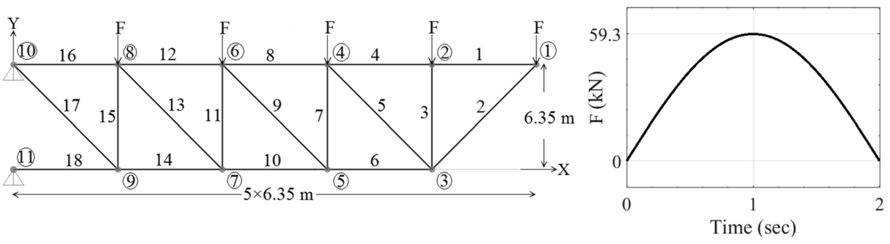


Fig. 5 Schematic of the 18-bar truss and the applied dynamic load

Performance of the proposed method is evaluated based on the final cost function value, the cost function value after the ESL step, the total number of transient structural analyses needed to reach the best design, and the total CPU time needed to reach the best design. A Core i7 with CPU of 3.4 GHz and Ram of 8 GB desktop computer is used.

8.1 Testing MOESL with known problems

8.1.1 Eighteen-bar truss (linear dynamic analysis)

Figure 5 shows the configuration of the 18-bar truss subjected to a half sine wave load at nodes 1, 2, 4, 6, 8. This example was solved in Choi and Park (2002) for continuous design variables with gradient-based optimization algorithms. The modulus of elasticity and the density are 69 GPa and 2765 kg/m3, respectively. All members are subjected to stress limitations of 138 MPa in both tension and compression. The allowable displacement for all nodes in both vertical and horizontal directions is ±203 mm. The optimization problem is to minimize the total mass of the structure. Four sizing variables and eight shape variables are selected as the design variables.

To test the performance of the proposed algorithm, this example is re-formulated as a discrete variable optimization problem. The sizing variables are selected from the discrete set of 100 elements where the range of the cross-sectional area is from 1 to 150 cm^2 with 1.505 cm^2 increment. The shape variables are the x and y coordinates of nodes 3, 5, 7, 9. The shape variables are selected from the discrete set of 100 elements where the range is from -317.5 (half the span of 635 cm) to 317.5 cm with 6.141 cm increment. The members of the truss are divided into 4 groups giving 4 sizing design variables (Choi and Park 2002): all top chord members, all bottom chord members, all vertical members, and all diagonal members. Considering the peaks of the displacements and the stresses, the time duration for dynamic analysis is set from 0 to 8 second. The time interval is divided into 100 increments giving 100 loading conditions for static response optimization with the ESL approach. Each loading condition vector has 18 elements since there are 18 degrees of freedom for the truss. Table 2 summarizes results for 10 different runs for ECBO without the ESL cycles and for the three ESL methods. The data in the table for the 10 runs for each ESL method includes: Final mass (kg), mass at the end of ESL cycles (kg), number of ECBO iterations

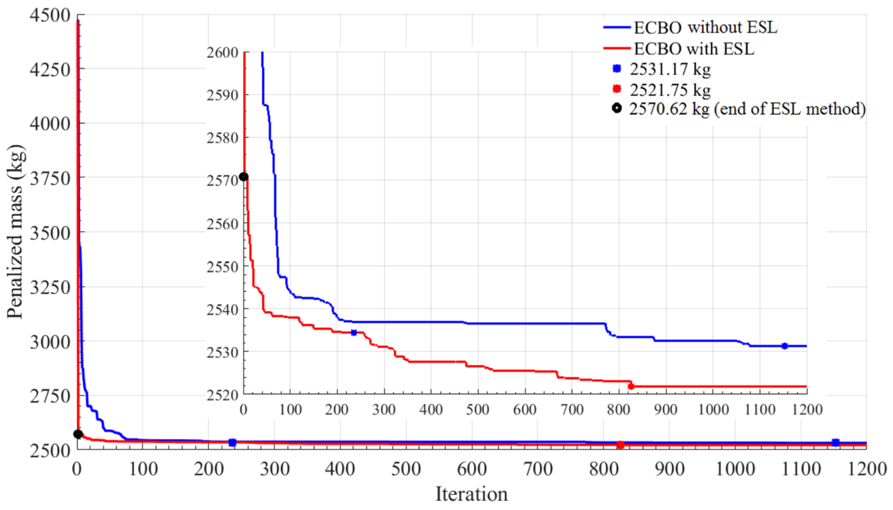


Fig. 6 Convergence history of 18-bar truss of the first run

without the ESL cycles, number of dynamic analyses, number of static analyses, CPU time at the end of ESL step, and the total CPU time needed to obtain the final design. It is interesting to note that of the 30 runs (the total runs of ESL1, ESL2, and ESL3), 6 runs converged to a mass value of about 2520.5, 17 runs converged to the mass that was within 0.1% of the best value and the remaining 7 runs converged to within 0.2% of the best value. This shows the robustness of the proposed algorithm for this example because all the designs would be acceptable from a practical applications point of view.

It is noted from the data in Table 2 that at the end of ESL cycles, the best design has not been reached for all the runs. Therefore, the algorithm must switch to ECBO with dynamic analysis of the entire population to obtain the final design. It is also noted that many more ESL cycles beyond the ones shown in Table 2, did not result in improved designs.

To compare the ECBO with and without ESL cycles, averages and standard deviations of some key parameters for 10 runs for each method are examined. These data are summarized in Table 3. The averages of the final masses and the total number of dynamic analyses show that the proposed method (MOESL with ESL1, ESL2 or ESL3) obtains not only better final designs but also needs a significantly smaller number of dynamic analyses compared to ECBO without the ESL cycles. That is, the average of dynamic analyses of ECBO without ESL cycles is 42524 analyses whereas ESL1, ESL2, and ESL3 have averages of 20,964, 22,616, and 23,000 analyses, respectively. Tables 2 and 3 show that not only the quality of final designs is improved but also the total CPU time is reduced for MOSEL compared to ECBO without ESL. That is, the average of the total CPU time of 10 runs for ECBO without ESL is 51.35 min whereas for ESL1, ESL2, and ESL3 it is 27.13 min, 29.04 min, and 28.98 min, respectively.

To study the performance of three proposed ESL approaches, averages and standard deviations for the 10 runs of each ESL method given in Table 3 are

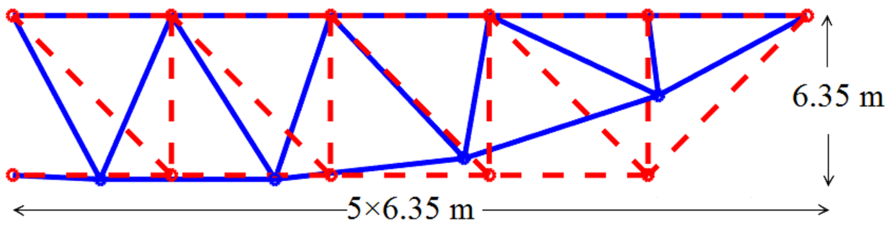


Fig. 7 Optimum configuration for the 18-bar truss

examined. It is seen that ESL2 has the smallest averages and standard deviations for the final mass as well as the mass at the end of ESL cycles. This shows that ESL2 approach is more reliable in obtaining the final solution. Although ESL2 approach has a slightly higher average for the number of dynamic analyses and CPU time, it is preferred because of its reliability in obtaining the final design. Performance of ESL1 is a close second to ESL2 for this example.

The best initial and final designs of the first run of ECBO without the ESL cycles and MOESL using ESL2 approach are shown in Table 4. For the same initial population, MOESL found a lighter design of 2521.75 kg. After 6 ESL cycles (614 + 40 = 124 dynamic analyses), the total structure mass became 2570.62 kg (this is just 1.94 % heavier than the best design). As shown in Fig. 6, MOESL converges faster than ECBO without ESLs. That is, when ECBO obtains the total mass of 2531.17 kg at iteration 1153, and ECBO with ESL needs just 236 iterations to reach the same mass. That is, with a population of 40 designs, ECBO without ESL cycles needs 917 iterations (91,740 - 614 = 36,596 dynamic analyses) more than MOESL to reach the same mass of 2531.17 kg. MOESL final design configuration is depicted in Fig. 7. The problem was also run 100 times. The data from these runs did not result in much difference in average and standard deviation. That is, the average and standard deviation of final mass for 100 runs are 2525.93 kg and 4.02 kg using ECBO, respectively, while when ESL2 is used the average and standard deviation of final mass are 2522.78 kg and 1.22 kg, respectively (the results of 10 runs are shown in Table 3).

The optimum structural mass found by Choi and Park (2002) is 3260.9 kg using a gradient-based algorithm. In this study, the best structural mass is found to be 2520.5 kg. This is a somewhat surprising result since the optimum mass is expected to be higher with discrete design variables. Since MOESL is a metaheuristic algorithm, it is likely to converge to a global minimum point whereas a gradient-based method converges to a local minimum point.

8.1.2 Ten-bar truss (nonlinear dynamic analysis)

Figure 9 shows the configuration of the 10-bar cantilever truss subjected to a half sine wave load at nodes 2 and 4. This example was solved in Kim and Park (2010) for continuous design variables using a gradient-based optimization algorithm. The material nonlinearity is considered in this problem. The Youngs modulus is 200

Table 2 Data for 10 different runs of the18-bar truss

	Run									
	1	2	3	4	5	6	7	8	9	10
ECBO without ESL cycles										
Mass (kg)	2531.2	2535.7	2525.4	2526.3	2521.8	2524.0	2525.6	2522.7	2523.0	2527.5
No. of iterations[a]	1153	1198	1163	1182	840	1097	1133	1186	482	1197
No. of dynamic analyses	46120	47920	46520	47280	33600	43880	45320	47440	19280	47880
Total CPU time (min)	55.69	57.86	56.17	57.09	40.57	52.99	54.72	57.28	23.28	57.82
MOESL										
ESL1										
Final mass (kg)	2523.5	2521.2	2521.1	2521.1	2523.5	2523.9	2520.5	2520.5	2522.9	2520.8
Mass at end of ESL cycles (kg)	2594.2	2607.5	2591.0	2594.8	2588.4	2600.5	2585.3	2590.1	2589.4	2585.7
No. of iterations[a]	452	591	568	424	568	583	573	545	433	475
No. of cycles	7	7	12	7	13	5	9	6	6	11
No. of dynamic analyses	18178	23738	22888	17058	22902	23390	23046	21884	17404	19154
No. of static analyses	61360	51240	84960	71840	87000	32960	71880	45960	44400	78280
ESL step CPU time (min)	1.90	1.59	2.64	2.23	2.70	1.02	2.23	1.43	1.38	2.43
Total CPU time (min)	23.74	30.14	30.07	22.71	30.13	29.18	29.91	27.75	22.29	25.37
ESL2										
Final mass (kg)	2521.8	2523.0	2520.5	2521.5	2522.9	2521.5	2521.4	2522.2	2523.9	2520.6
Mass at end of ESL cycles (kg)	2570.6	2589.3	2589.1	2580.7	2585.5	2580.0	2590.4	2579.3	2577.7	2578.7
No. of iterations[a]	826	539	379	600	563	598	480	597	600	442
No. of cycles	6	6	11	11	8	7	7	9	7	14
No. of dynamic analyses	33124	21644	15314	24154	22632	24018	19298	24006	24098	17876
No. of static analyses	45920	43240	82480	67840	50320	56960	46320	56240	50440	104320
ESL step CPU time (min)	1.43	1.34	2.56	2.11	1.56	1.77	1.44	1.75	1.57	3.24
Total CPU time (min)	41.32	27.38	20.87	31.09	28.75	30.65	24.62	30.58	30.55	24.59

Table 2 (continued)

	Run									
	1	2	3	4	5	6	7	8	9	10
ESL3										
Final mass (kg)	2522.3	2522.5	2525.5	2524.1	2520.8	2523.8	2525.5	2523.3	2524.8	2522.6
Mass at end of ESL cycles (kg)	2594.9	2618.8	2607.4	2613.4	2597.8	2603.8	2619.4	2610.2	2598.8	2615.3
No. of iterations[a]	559	535	588	561	594	598	541	565	589	600
No. of cycles	7	5	6	5	7	5	5	5	5	7
No. of dynamic analyses	22458	21470	23604	22510	23858	23990	21710	22670	23630	24098
No. of static analyses	45720	30920	52160	45160	43240	35880	38640	38720	38880	52080
ESL step CPU time (min)	1.42	0.96	1.62	1.40	1.34	1.11	1.20	1.20	1.21	1.62
Total CPU time (min)	28.42	26.80	30.02	28.50	30.03	30.00	27.33	28.49	29.66	30.60

[a]ECBO iterations without ESL cycles

Table 3 Comparison of averages and standard deviations for 10 runs of the18-bar truss

Metric	Averages				Standard deviation			
	ECBO Alone	ESL1	ESL2	ESL3	ECBO Alone	ESL1	ESL2	ESL3
Final mass (kg)	2526.32	2521.92	2521.85	2523.5	4.28	1.37	1.10	1.50
Mass at end of ESL cycles (kg)	–	2592.68	2583.41	2607.97	–	6.90	5.13	8.90
No. of dynamic analyses	42,524	20,964	22,616	23,000	9208	2691	4791	962
No. of static analyses	–	62,988	60,408	42,140	–	18,644	19,488	6848
ESL step CPU time (min)	–	1.96	1.88	1.31	–	0.58	0.66	0.21
Total CPU time (min)	51.35	27.13	29.04	28.98	11.12	3.28	5.49	1.27

GPa, the tangent modulus is 50 GPa, the yield stress is 200 MPa, the Poisson ratio is 0.3, and the mass density is 7860 kg/m^3.

To evaluate the proposed algorithm, this problem is also re-formulated as a discrete variable problem. The optimization problem is to minimize the total mass of the structure. The design variables are the cross-sectional areas of the members (Table 7). The size variables are selected from the discrete set of 100 elements where the range of the cross-sectional areas is from 78.5 to 2826 cm^2 with 27.752 mm^2 increment. All members are subjected to stress limitations of 250 MPa in both tension and compression. Considering the peaks of the displacements and the stresses, the time duration for the analysis is set from 0 to 0.03 second with a time step of 0.0002 second. This gives 150 loading conditions for static response optimization with the ESL approach. Each loading condition vector has 8 elements since there are 8 degrees of freedom for the truss.

Table 5 shows results for 10 different runs of ECBO without the ESL cycles and the results of MOESL with the three approaches, ESL1, ESL2, and ESL3. The data in the table for all the 10 runs includes: final mass (kg), mass at end of ESL cycles (kg), number of ECBO iterations without the ESL cycles, number of dynamic analyses, number of static analyses, CPU time at the end of ESL step, and the total CPU time needed to obtain the final design. It is interesting to note that of the 30 runs (the total runs of ESL1, ESL2, and ESL3), 4 runs converged to a mass value of 25.2 kg, 10 runs converged to the mass that was within 2.5% of the best value and 8 runs converged to within 5.0% of the best value. An examination of the averages and standard deviations in Table 6 for this example leads to the same conclusion as for Example 1: MOESL obtains better designs with less number of dynamic analyses compared to ECBO without the ESL cycles, GOESL needs less CPU time to reach the final design, and ESL2 approach performs more reliably in obtaining the final design than ESL1 and ESL2. The best initial and final designs of the first run of ECBO without ESL cycles and MOESL using ESL2 are shown in Table 7. For the same initial population ECBO with ESL2 found a lighter design of 26.05 kg. After 7 ESL cycles, the total structure mass became 41.58 kg (the

Table 4 Initial and final designs of 18-bar truss for the first run

Design variables		Best initial design	ECBO	MOESL (ESL2)						
			Final design	Cycle 1	Cycle 2	Cycle 3	Cycle 4	Cycle 5	Cycle 6	Final design
1	Area_{top} (mm^2)	9431.31	10635.35	14548.48	10484.85	10484.85	10785.86	11237.37	11237.37	10334.34
2	$\text{Area}_{\text{bottom}}$ (mm^2)	13946.46	10635.35	10033.33	10785.86	10785.86	11688.89	11538.38	11538.38	11387.88
3	$\text{Area}_{\text{vertical}}$ (mm^2)	14698.99	4013.13	2207.07	3561.62	3561.62	3561.62	3260.61	3260.61	3561.62
4	$\text{Area}_{\text{diagonal}}$ (mm^2)	7324.24	4314.14	4163.64	5518.18	5518.18	4314.14	4314.14	4314.14	4163.64
5	X_3 (mm)	2533.59	737.63	−3175.00	1250.76	1250.76	−3110.86	−2854.29	−2854.29	416.92
6	Y_3 (mm)	−2277.02	3175.00	2726.01	2148.74	2148.74	2148.74	2277.02	2277.02	3175.00
7	X_5 (mm)	−96.21	−865.91	−3175.00	−3046.72	−3046.72	−3110.86	−3175.00	−3175.00	−994.19
8	Y_5 (mm)	−1186.62	481.06	1250.76	865.91	865.91	930.05	1058.33	1058.33	673.48
9	X_7 (mm)	2341.16	−2341.16	−1956.31	−2405.30	−2405.30	−2597.73	−2533.59	−2533.59	−2212.88
10	Y_7 (mm)	−1571.46	−224.49	224.49	−32.07	−32.07	−32.07	224.49	224.49	−160.35
11	X_9 (mm)	−2084.60	−3175.00	−1699.75	−2341.16	−2341.16	−2148.74	−2277.02	−2277.02	−2854.29
12	Y_9 (mm)	−1250.76	−96.21	−288.64	−288.64	−288.64	−224.49	−160.35	−160.35	−160.35
Max. Displacement (mm)		14.76 (1[a])	203.00 (1)	205.12 (1)	202.50 (1)	202.50 (1)	202.46 (1)	202.76 (1)	202.76 (1)	203.00 (1)
Max. stress (MPa)		86.98 (17[b])	111.18 (18)	130.13 (15)	110.11 (18)	110.11 (18)	116.93 (17)	109.86 (17)	109.86 (17)	114.01 (17)
Mass (kg)		4470.86	2531.17	2621.58	2621.58	2621.58	2577.38	2570.62	2570.62	2521.75
Merit		4470.86	2531.17	2621.58	2621.58	2621.58	2577.38	2570.62	2570.62	2521.75
Iteration		-	1153[c]	233[d]	233[d]	232[d]	150[d]	150[d]	150[d]	826[c]
Dynamic, static analyses		-	46120, 0	14, 9320	14, 9320	14, 9280	14, 6000	14, 6000	14, 6000	33040, 0

Top members: 1, 4, 5, 12, and 16. Bottom members: 2, 6, 10, 14, and 18. Vertical members: 3, 7, 11, and 15. Diagonal members: 5, 9, 13, and 17

[a]Node number where the maximum displacement occurs

[b]Member number where the maximum stress occurs

[c]Transient analysis.

[d]Linear static analysis

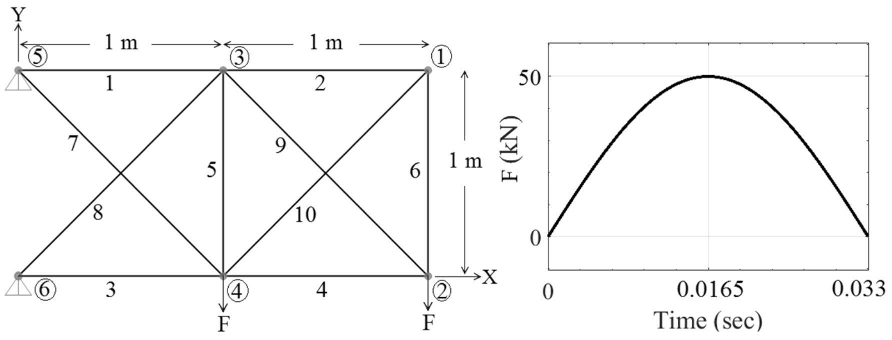


Fig. 8 Schematic of the 10-bar truss and the applied dynmaic load

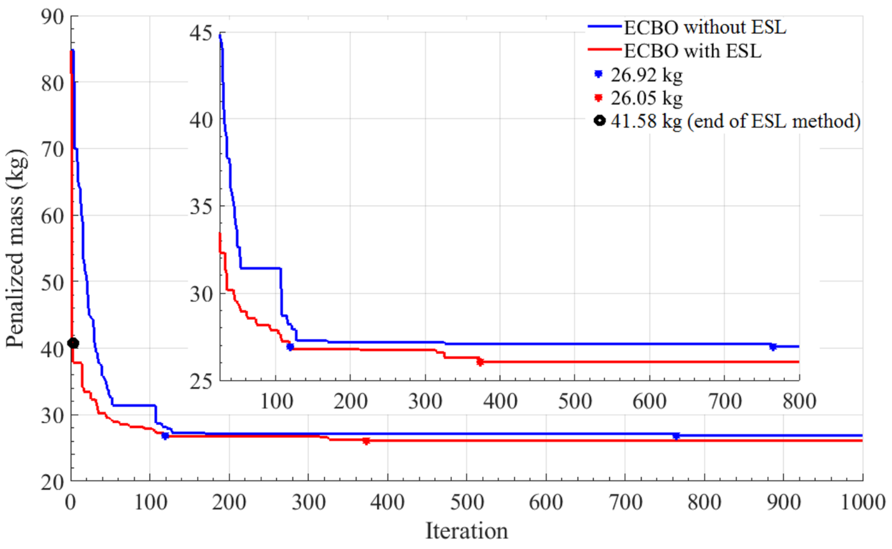


Fig. 9 Convergence history of 10-bar truss of the first run

best design is 59.62% lighter). As shown in Fig. 8, MOESL converges faster than ECBO without ESL cycles. ECBO reaches the best design of 26.92 kg at iteration 765 and MOESL needs just 119 iterations to obtain a similar mass. That is, with a population of 40 designs, ECBO needs 25742 more dynamic analyses (646 iterations) than MOESL.

The average of final masses and the average of the total number of dynamic analyses of the proceeding two examples show that the proposed method (MOESL) gives better results. The best design using the ESL2 approach shows better convergence behavior and final designs of the three approaches. Therefore, this approach is used in the next two examples. The problem was also run 100 times. The data from these runs did not result in much difference in average and standard deviation. That is, when ECBO alone was used the average and standard deviation of final mass for 100 runs are 26.72 kg and 2.24 kg, respectively,

Table 5 Data for 10 different runs of the10-bar truss

	Run									
	1	2	3	4	5	6	7	8	9	10
ECBO without ESL cycles										
Mass (kg)	26.9	30.0	26.2	25.6	25.4	31.2	25.9	26.0	25.5	25.4
No. of iterations[a]	765	395	624	936	406	772	833	380	422	664
No. of dynamic analyses	30,600	15,800	24,960	37,440	16,240	30,880	33,320	15,200	16,880	26,560
Total CPU time (min)	18.36	9.48	14.98	22.46	9.74	18.53	19.99	9.12	10.13	15.94
MOESL										
ESL1										
Final mass (kg)	29.6	25.8	25.2	25.4	25.4	25.6	27.7	25.8	25.9	25.6
Mass at end of ESL cycles (kg)	61.7	58.3	47.0	63.6	43.5	87.4	77.0	45.9	140.4	54.9
No. of iterations[a]	465	252	313	187	219	202	180	351	363	194
No. of cycles	6	7	14	8	7	7	5	7	7	5
No. of dynamic analyses	18,606	10,087	12,534	7488	8767	8087	7205	14,047	14,527	7765
No. of static analyses	37,640	44,800	81,360	42,960	50,160	45,240	30,720	48,480	59,120	33,720
ESL step CPU time (min)	0.96	1.14	2.08	1.10	1.28	1.16	0.78	1.24	1.51	0.86
Total CPU time (min)	12.12	7.20	9.60	5.59	6.54	6.01	5.11	9.67	10.23	5.52
ESL2										
Final mass (kg)	26.1	25.2	26.1	25.7	26.1	26.1	25.2	25.3	25.7	25.9
Mass at end of ESL cycles (kg)	41.6	29.6	39.0	36.1	45.2	61.1	68.9	35.0	51.6	39.6
No. of iterations[a]	374	294	227	251	267	258	417	162	339	345
No. of cycles	7	21	9	10	6	6	5	11	5	14
No. of dynamic analyses	15058	12054	9206	10180	10764	10404	16750	6634	13630	13996
No. of static analyses	46200	119080	59720	63200	47640	33120	29000	66000	39720	80120
ESL step CPU time (min)	1.18	3.04	1.53	1.61	1.22	0.85	0.74	1.69	1.01	2.05
Total CPU time (min)	10.21	10.27	7.05	7.72	7.68	7.09	10.79	5.67	9.19	10.44

Table 5 (continued)

	Run									
	1	2	3	4	5	6	7	8	9	10
ESL3										
Final mass (kg)	27.7	25.2	27.0	25.9	27.6	30.2	28.7	27.1	25.3	26.2
Mass at end of ESL cycles (kg)	54.3	79.4	63.7	74.6	83.8	86.1	77.7	63.2	65.2	75.3
No. of iterations[a]	255	242	341	204	283	354	138	113	324	409
No. of cycles	5	5	5	7	6	5	5	5	5	5
No. of dynamic analyses	10270	9750	13710	8258	11404	14230	5590	4590	13030	16430
No. of static analyses	29280	25480	27160	35280	30560	25520	27240	26120	35200	33560
ESL step CPU time (min)	0.75	0.65	0.69	0.90	0.78	0.65	0.70	0.67	0.90	0.86
Total CPU time (min)	6.91	6.50	8.92	5.86	7.62	9.19	4.05	3.42	8.72	10.72

[a]ECBO iterations without ESL cycles

Table 6 Comparison of averages and standard deviations for 10 runs of the10-bar truss

Metric	Averages				Standard deviation			
	ECBO Alone	ESL1	ESL2	ESL3	ECBO Alone	ESL1	ESL2	ESL3
Final mass (kg)	26.8	26.2	25.73	27.08	2.1	1.36	0.39	1.56
Mass at end of ESL cycles (kg)	–	67.96	44.75	72.32	–	28.97	12.33	10.26
No. of dynamic analyses	24788	10911	11869	10726	8268	3849	3025	3812
No. of static analyses	–	47420	58380	29540	–	14496	26569	3909
ESL step CPU time (min)	–	1.21	1.49	0.75	–	0.37	0.68	0.10
Total CPU time (min)	14.87	7.76	8.61	7.19	4.96	2.44	1.79	2.32

while when ESL2 is used the average and standard deviation of final mass are 25.81 kg and 0.41 kg, respectively (the results of 10 runs are shown in Table 6).

The optimum structural mas found by Kim and Park (2010) is 21.77 kg. In this study, the best structural mass is found to be 25.2 kg. This is an expected result as the minimum mass is likely to increase with the discrete design variables. The discretized domain imposes additional constraints on the problem.

This nonlinear optimization problem is also solved using two additional nonlinear procedures.

1- The loads that are used in ESL cycles are calculated using the nonlinear stiffness matrix and displacements from the equation of motion (Eq. (13)) as follows:

$$\mathbf{p}_\alpha = \mathbf{K}_N(\mathbf{X})\mathbf{u}(t_\alpha);\alpha = 1, 2, \dots, n \tag{35}$$

where $\mathbf{p}_\alpha$ are the equivalent load vectors, $\mathbf{K}_N$ is the nonlinear stiffness matrix, and $\mathbf{u}$ is the dynamic displacements vectors. These load vectors are calculated at the beginning on each cycle. Then, in each cycle, the linear stiffness matrix is used in the optimization process of an ESL cycle.

2- The equivalent load vectors are calculated using procedure 1. However, instead of using the linear stiffness matrix in the optimization process, the nonlinear stiffness matrix is used. That is, in each cycle, nonlinear static analyses are used in the optimization process. In both procedures ESL2 is used.

The results of the two nonlinear procedures showed there are no advantages over the method described in Sect. 7. That is, the averages and standard deviations of 10 individual runs of the structural mass in the end of ESL step and final designs are similar to those in Table 6. However, the CPU time of the second procedure is 5 time more the ESL2 CPU time in Table 6 because of using nonlinear static analyses in the optimization process. These two procedures are not discussed for continuous variable problems in Kim and Park (2010). The reason perhaps is that these procedures would require gradient evaluation for the nonlibear problems which is a tedious process.

Table 7 Initial and final design of 10-bar truss of the first run

Design variables (mm2)		Best initial design	ECBO	MOESL (ESL2)							
			Final design	Cycle 1	Cycle 2	Cycle 3	Cycle 4	Cycle 5	Cycle 6	Cycle 7	Final design
1	A1	439.28	439.28	855.57	855.57	661.30	661.30	661.30	661.30	661.30	605.80
2	A2	494.79	300.52	411.53	272.77	134.01	134.01	161.76	189.51	189.51	134.01
3	A3	1632.64	661.30	550.29	550.29	550.29	550.29	550.29	550.29	550.29	522.54
4	A4	1771.40	78.50	439.28	411.53	217.26	217.26	217.26	189.51	189.51	161.76
5	A5	1632.64	106.25	411.53	217.26	134.01	134.01	134.01	134.01	134.01	161.76
6	A6	1382.87	217.26	1438.37	356.03	161.76	161.76	161.76	217.26	217.26	106.25
7	A7	383.78	550.29	1188.60	800.07	689.06	689.06	689.06	661.30	661.30	328.27
8	A8	134.01	189.51	1327.36	1327.36	1299.61	1299.61	1299.61	1299.61	1299.61	439.28
9	A9	605.80	78.50	189.51	633.55	245.02	245.02	245.02	217.26	217.26	217.26
10	A10	1299.61	328.27	494.79	439.28	217.26	217.26	189.51	189.51	189.51	161.76
Max. stress (MPa)		242.60 (7[a])	249.88 (9)	243.92 (3)	248.57 (3)	252.37 (2)	252.37 (2)	250.50 (3)	246.16 (3)	246.16 (3)	249.15 (7)
Mass (kg)		84.74	26.92	67.85	56.51	41.83	41.83	41.76	41.58	41.58	26.05
Merit		84.74	26.92	67.85	56.51	43.10	43.10	41.93	41.58	41.58	26.05
Iteration		–	765	285	163	125	207	125	125	125	374
Dynamic, static analyses		–	30600, 0	14, 11400	14, 6520	14, 5000	14, 8280	14, 5000	14, 5000	14, 5000	14960, 0

[a]Member number where the maximum stress occurs

8.2 Frame problems

8.2.1 Two-story two-bay frame (linear dynamic analysis)

This design example is a 2-story 2-bay planar steel frame having 4 beams and 6 columns and has not been solved in the literature before. It is modeled using SAP2000 and MATLAB with 19 nodes and 20 elements. Note that in order to get more accurate analysis results intermediate nodes are introduced for each member of the frame. The frame has 48 degrees of freedom that is subjected to a half sine wave load at nodes 2 and 3 and uniformly distributed static load of 5 kip/ft (72.97 kN/m) on members 13 to 20 as shown in Fig. 1. All ground supports are fixed. Material properties are: Youngs modulus, E=29000 ksi (200 GPa), yield stress, F_y=50 ksi (344.7 MPa), and poisons ratio, $\nu = 0.3$.

Based on primarily analysis of the problem, columns 1 and 2 are selected from the first lightest 50 standard W-shapes provided in AISC tables (AISC 2017) while the rest of the members are selected from the lightest 50-99 standard W-shapes provided in AISC tables (AISC 2017). The sections are rearranged in an ascending order based on their weight. The problem is formulated as an integer variable optimization problem where the section number is treated as a design variable (see Sect. 3.1). All members are subjected to interaction ratio strength requirement (Eq. 12).

Considering the peaks of the displacements and the stresses, the time range for the analysis is set from 0 to 4 second with a time step of 0.04 second. This gives 100 loading conditions for static response optimization with the ESLM. Each loading condition vector has 48 elements since there are 48 degrees of freedom for the frame.

This example was also solved without the intermediate nodes for the members. That model was more efficient to solve. However, the final designs were not as good as with the increased degrees of freedom. The reason is that with more degrees of freedom a more accurate dynamic response is obtained resulting in better ESLs as well.

Table 8 gives the best initial and final designs of the first run. For the same initial population, MOESL found a lighter design of 9369 lb (4249.7 kg). After 5 ESL cycles, the total structure weight becomes 10,392 lb (4713.7 kg). The best design is 10.9% lighter. As shown in Fig. 10, MOESL converges faster than ECBO without ESL cycles. ECBO reaches the best design of 9624 lb (4365.4 kg) at iteration 149 whereas MOESL needs 64 iterations to obtain a design of 9636 lb (4370.8 kg). From Table 9, the average of 10 individual runs for final weight is better with MOESL (9361.9 lb (4246.5 kg)) than with ECBO without the ESLM (9466.8 lb (4294.1 kg)). The average of the total number of dynamic analyses shows that MOESL needs a smaller number of dynamic analyses than ECBO without ESL cycles to reach the final design. However, the average of the total CPU time is almost similar.

Table 8 Initial and final design of 2-story 2-bar frame of the first run

Design variable no.	Member no.	Best initial design	ECBO	MOESL (ESL2)					
			Final design	Cycle 1	Cycle 2	Cycle 3	Cycle 4	Cycle 5	Final design
1	1–2	W16X31	W12X35	W14X43	W14X43	W14X43	W14X43	W14X43	W16X36
2	3–4	W10X39	W5X19	W8X31	W8X31	W8X31	W8X31	W8X31	W5X16
3	5–6	W27X84	W18X86	W24X84	W24X76	W24X76	W24X76	W24X76	W24X76
4	7–8	W21X62	W18X46	W18X50	W21X48	W21X48	W21X48	W21X48	W12X45
5	9–10	W8X48	W14X48	W21X68	W24X76	W24X76	W24X76	W24X76	W24X76
6	11–12	W8X48	W21X44	W21X44	W21X44	W21X44	W21X44	W21X44	W21X44
7	13–14	W14X53	W24X68	W16X67	W16X67	W16X67	W16X67	W16X67	W16X67
8	15–16	W18X76	W18X86	W21X68	W24X68	W24X68	W24X68	W24X68	W24X68
9	17–18	W12X50	W21X48	W24X84	W18X71	W18X71	W18X71	W18X71	W21X55
10	19–20	W24X76	W18X60	W24X68	W24X68	W24X68	W24X68	W24X68	W21X55
Max. interaction ratio		1.450 (7)	0.991 (4)	0.909 (6)	0.927 (3)	0.927 (3)	0.927 (3)	0.927 (3)	0.978 (6)
Weight (lb)		9864	9624	10728	10392	10392	10392	10392	9396
Merit function		37848.5	9624	10728	10392	10392	10392	10392	9396
Iteration		1	149	140	118	63	63	63	88
Dynamic; static analyses		40, 0	5960; 0	14; 5600	14; 4720	14; 2520	14; 2520	14; 2520	3520; 0

[a] Member number where the maximum interaction ratio occurs. 1 lb = 0.453 kg

Table 9 Data for 10 different runs of the 2-story 2-bay frame

	Run										Average	SD
	1	2	3	4	5	6	7	8	9	10		
ECBO without ESL cycles												
Weight (lb)	9624	9468	9480	9240	9540	9600	9564	9276	9468	9408	9466.8	128.6
No. of iteration	149	210	184	241	199	115	209	210	210	168	189.5	36.6
No. of dynamic analyses	5960	8400	7360	9640	7960	4600	8360	8400	8400	6720	7580	1464.6
Total CPU time (hr)	9.78	13.79	12.08	15.83	13.07	7.55	13.73	13.79	13.79	11.03	12.44	2.40
MOESL												
ESL2												
Final weight (lb)	9396	9216	9076	9480	9430	9252	9414	9564	9432	9360	9361.9	142.8
Weight at end of ESL cycles (lb)	10,392	9948	10,369	10,500	10,128	10,140	10,440	10,356	10,284	10,092	10,264.9	178.3
No. of iterations	88	220	219	106	182	148	155	92	149	76	143.5	52.6
No. of cycles	5	5	5	7	5	5	5	6	5	5	5.3	0.7
No. of dynamic analyses	3520	8800	8760	4240	7280	5920	6200	3680	5960	3040	5740	2104.5
No. of static analyses	17880	16,800	16,040	34,960	19,280	14,760	19,400	23,720	19,280	17,360	19,948	5810.5
ESL step CPU time (h)	2.48	2.33	2.23	4.86	2.68	2.05	2.69	3.29	2.68	2.41	2.77	0.81
Total CPU time (h)	8.38	16.90	16.72	11.98	14.74	11.88	12.99	9.47	12.58	7.52	12.32	3.23

1 lb = 0.453 kg

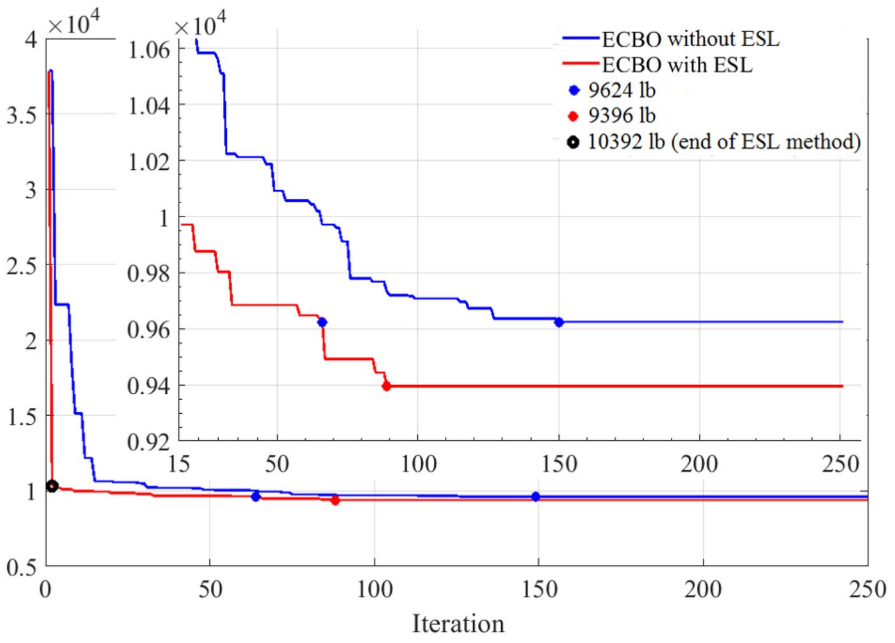


Fig. 10 Convergence history of 2-story 2-bay frame of the first run (just 250 iterations are shown)

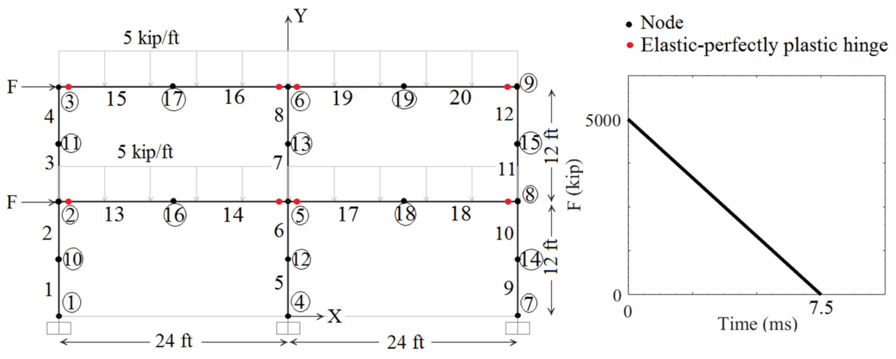


Fig. 11 Schematic of the 2-story 2-bay frame and the applied blast load

8.2.2 Two-story two-bay frame subjected to blast loads (nonlinear dynamic analysis)

The configuration of this numerical example is the same as the previous example except that it is subjected to blast load. This example has also not been solved in the literature. For simplicity, the blast load is modeled as a triangle and the negative pressure phase is neglected. In addition, a uniformly distributed static load of 5 kip/ft (72.97 kN/m) on members 13 to 20 is added as shown in Fig. 11. All ground

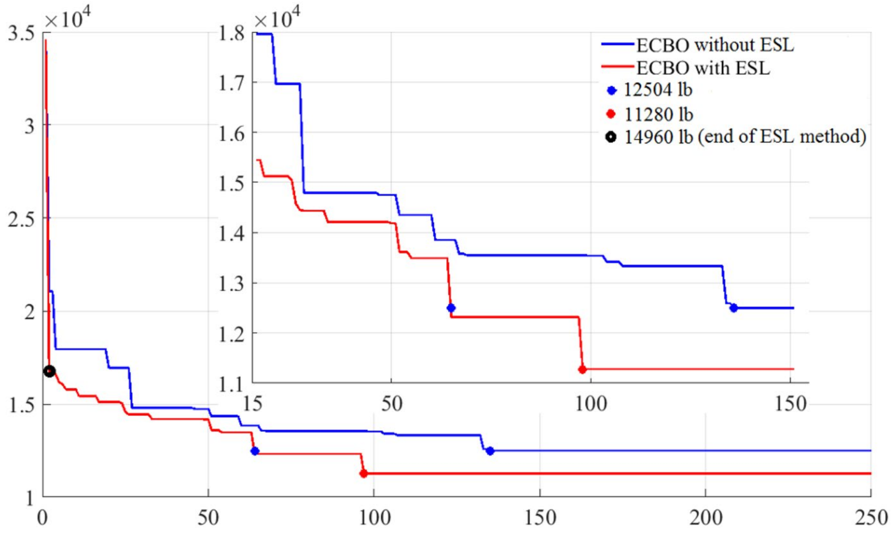


Fig. 12 Convergence history of 2-story 2-bay frame subjected to blast load of the first run (just 250 iterations are shown)

supports are fixed. Youngs modulus, E=29000 ksi (200 GPa), yield stress, F_y=50 ksi (344.7 MPa), ultimate stresses, F_u=65 ksi (448.16 MPa) and poisons ratio, $\nu = 0.3$. Due to the dynamic effects resulting from the rapid strain rates, the dynamic increase factors (*DIF*) for yield and ultimate stresses of 1.19 and 1.05, respectively, are used (Gilsanz et al. 2013). Since the average yield stress for structural steels having a specified minimum yield stress of 50 ksi or less is generally higher than the specified minimum, it is recommended that the minimum design yield stress, as specified by the AISC (AISC 2017) specification, be increased by 10 percent. This factor is called the strength increase factor (*SIF*). Also, for all modes of failure, it is permissible to use a strength reduction factor (ϕ) of 1.0 (ASCE 2011). The reader is referred to ASCE (2010), Dusenberry (2010), Gilsanz et al. Gilsanz et al. (2013), and DoD (2008) for more details. Therefore, the new yield stress F_{dy} and new ultimate stress F_{du} values are as follows:

$$F_{dy} = (SIF)(DIF)F_y = (1.1)(1.19)(50) = 65.45 \text{ ksi} \tag{36}$$

$$F_{du} = (DIF)F_u = (1.05)(65) = 68.25 \text{ ksi} \tag{37}$$

Table 10 Initial and final design of 2-story 2-bar frame subjected to blast load of the first run

Design variable no.	Member no.	Best initial design	ECBO	MOESL (ESL2)						
			Final design	Cycle 1	Cycle 2	Cycle 3	Cycle 4	Cycle 5	Cycle 6	Final design
1	1-2	W16X89	W10X100	W12X106	W24X76	W24X76	W24X76	W30X116	W30X116	W8X67
2	3-4	W24X76	W27X84	W24X104	W18X76	W18X76	W18X76	W27X102	W27X102	W21X73
3	5-6	W30X132	W10X112	W40X149	W14X109	W14X109	W14X109	W30X148	W30X148	W10X100
4	7-8	W30X99	W30X90	W33X118	W24X84	W24X84	W24X84	W21X111	W21X111	W30X116
5	9-10	W18X130	W10X88	W27X146	W24X103	W24X103	W24X103	W27X146	W27X146	W14X99
6	11-12	W30X99	W18X106	W10X88	W10X88	W10X88	W10X88	W24X146	W24X146	W18X97
7	13-14	W18X55	W14X74	W27X84	W21X44	W21X44	W21X44	W18X60	W18X60	W16X50
8	15-16	W8X67	W12X65	W27X84	W10X26	W10X26	W10X26	W8X58	W8X58	W12X50
9	17-18	W21X73	W21X44	W27X84	W10X54	W10X54	W10X54	W14X61	W14X61	W21X44
10	19-20	W18X46	W21X48	W24X55	W21X50	W21X50	W21X50	W10X60	W10X60	W16X50
Max. interaction ratio		1.150 (3[a])	0.997 (1)	1.2964 (6)	1.139 (3)	1.139 (3)	1.139 (3)	1.000 (1)	1.000 (1)	0.961 (5)
Max. rotation (degree)		1.221 (2[b])	0.958 (2)	0.897 (2)	0.833 (2)	0.833 (2)	0.833 (2)	0.949 (2)	0.949 (2)	0.973 (2)
Max. *ISD*		0.905	0.810	0.6609	0.718	0.718	0.718	0.685	0.685	0.931
Weight (lb)		13260	12504	15900	16320	16320	16320	14964	14964	11280
Merit		34586	12504	31025	21184	21184	21184	14964	14964	11280
Iteration		1	135	131	63	63	63	63	63	97
Dynamic; static analyses		40; 0	5400; 0	14; 5240	14; 2520	14; 2520	14; 2520	14; 2520	14; 2520	3380; 0

[a]Member number where the maximum interaction ratio occurs. [b]Node number where the maximum rotation occurs. 1 lb = 0.453 kg

Table 11 Ten individual runs data of the 2-story 2-bar frame subjected to blast load

	Run										Average	SD
	1	2	3	4	5	6	7	8	9	10		
ECBO without ESL cycles												
Weight (lb)	12,504	12,132	11,400	12,192	12,192	12,240	12,204	13,020	13,032	12,636	12,355.2	477.6
No. of iteration	135	216	180	132	132	201	201	232	209	248	189	42
No. of dynamic analyses	5400	8640	7200	5280	5280	8040	8040	9280	8360	9920	7544	1699
Total CPU time (hr)	20.71	33.13	27.61	20.25	20.25	30.83	30.83	35.59	32.06	38.04	28.93	6.52
MOESL												
ESL2												
Final weight (lb)	11,280	10,092	12,816	13,272	13,272	12,204	12,396	11,080	12,000	11,644	12,005.6	1012.8
ESL final weight (lb)	14,960	10,435	18,376	14,793	14,960	14,960	19,405	20,162	20,947	12,651	16,164.9	3426.3
No. of iterations	97	2	148	97	97	97	177	103	192	100	111	52.8
No. of cycles	6	5	5	5	5	5	8	5	8	5	5.7	1.3
No. of dynamic analyses	3964	150	5990	3950	3950	3950	7192	4190	7792	4070	4520	2126.5
No. of static analyses	17,840	18,160	2,2000	16,480	16,480	16,480	18,160	20,560	30,440	15,480	19,208	4423.1
ESL step CPU time (h)	2.29	2.52	3.06	2.29	2.29	2.29	2.52	2.86	4.23	2.15	2.65	0.62
Total CPU time (h)	17.49	3.10	26.03	17.44	17.44	17.44	30.10	18.92	34.11	17.76	19.98	8.53

1 lb = 0.453 kg

Considering the blast load duration and the peaks of the response, the time range from the analysis is set from 0 to 1.25 s with a time step of 0.0025. This gives 500 loading conditions for static response optimization with the ESLM. Each loading condition vector has 48 elements since there are 48 degrees of freedom for the frame.

This example was also solved without the intermediate nodes in the members. Similar to the previous example, it was shown that increasing the degrees of freedom makes the algorithm converge to better designs for the same reasons as noted earlier. Table 10 gives the best initial and final designs for the first run of the problem. For the same initial population, MOESL found a lighter design of 11280 lb (5116.5 kg). After 6 ESL cycles, the total structure mass becomes 14964 lb (6787.6 kg). The best design is 32.6% lighter. As shown in Fig. 12, MOESL converges faster than ECBO without ESL cycles. That is, when ECBO without ESL cycles obtains the total weight of 12504 lb (5671.7 kg) at iteration 135, MOESL needs 64 iterations to reach a similar structural weight. That is, with a population of 40 designs, ECBO without ESL cycles needs 71 more iterations (2756 dynamic analyses) than ECBO with ESLs. Table 11 summarizes results for 10 different runs for ECBO without the ESL cycles and for MOESL. The average of the final weights and the total number of dynamic analyses show that MOESL obtains not only better average but also needs a smaller number of dynamic analyses compared to ECBO without the ESL cycles. Unlike the previous study case, the average of CPU time of MOESL is less than ECBO with ESL step (Table 11). This is because the numerical solver takes more time to solve the nonlinear problems. It was noticed that for some designs the numerical solver needs several minutes to converge and sometimes the solver stops converging. That is, for bigger problems with material and geomatical nonlinearity, the numerical solver might need very long time to solve one design which makes metaheuristic algorithms inefficient to use. ESL step, however, takes shorter CPU time (in compression with total CPU time as shown in Table 11) and it gives a good start to ECBO with full dynamic analysis.

9 Concluding remarks

Optimizing transient problems using metaheuristic algorithms is computationally expensive because every simulation requires solving a system of differential equations. In the search for a more efficient method for dynamic response structural optimization, the Equivalent Static Load Method (ESLM) with gradient-free algorithms was examined in this study. In the proposed method, the transient problem was transformed to ESL sets that generated the same displacement field as with the transient analysis for a given design. Then, the sets of generated ESLs were used as multiple loading conditions in the static response structural optimization process. Since it was not clear which design should be used at the end of each ESL cycle to generate ESLs in metaheuristic algorithms, three approaches were studied: the best

Table 12 Comparative data for design examples

Study case	ECBO without ESL			MOESL (ESL2)		
	Average final mass (or weight)	Average no. of dynamic analyses	Average no. of static analyses	Average final mass (or weight)	Average no. of dynamic analyses	Average no. of static analyses
18-bar truss (linear dynamic)	2526.32 kg	42524	–	2521.85	22616	60408
10-bar truss (nonlinear dynamic)	26.8 kg	24788	–	25.73	11869	58380
2-story 2 bay frame (linear dynamic)	9466.8 lb	7580	–	9361.9	5740	19948
2-story 2 bay frame (nonlinear dynamic)	12355.2 lb	7544	–	12005.6	4520	15480

1 lb = 0.453 kg

Table 13 Comparative of average CPU time for designs examples

Study case	Mass or weight	CPU time	
		ECBO without ESL	MOESL (ESL2)
18-bar truss (linear dynamic)	2526.32 kg	51.35 min	29.04 min (1.88 min)
10-bar truss (nonlinear dynamic)	26.8 kg	14.87 min	8.61 min (1.49 min)
2-story 2 bay frame (linear dynamic)	9466.8 lb	12.44 h	12.32 h (2.77 h)
2-story 2 bay frame (nonlinear dynamic)	12,355.2 lb	28.93 h	19.98 h (2.65 h)

1 lb = 0.453 kg

design from static analysis (ESL1), the best design from dynamic analysis (ESL2), and the heaviest feasible design from dynamic analysis (ESL3).

Based on the analysis of the results of four numerical examples (2 linear and 2 nonlinear), the following conclusions are drawn:

1. ESLM with metaheuristic algorithms is not able to obtain the best design because the ESLs calculated for the chosen member of the population are not suitable for the remaining members of the population. Also, a small change in design variables is not guaranteed in metaheuristic algorithms from one ESL cycle to the next. This violates the assumption of small changes in design from one ESL cycles to the next with the gradient-based methods (at least near the local minimum point). At the end of ESL cycles, improved designs are obtained although not the best design.
2. At the end of ESL cycles, the better designs and the improved population should be passed on to the metaheuristic method without the ESL cycles to improve these designs further.
3. In most cases, it is shown that the proposed method can reach the best design with a smaller number of dynamic analyses than with the metaheuristic algorithm without the ESL cycles as shown in Table 12. However, many static analyses of the structure must be performed. Table 13 shows a comparison of CPU time needed by ECBO without ESL and MOESL (ESL2). It is seen that in most cases, the CPU effort is smaller with MOSEL compared to the approach without ESL cycles.
4. Better final designs are obtained with MOSEL since the method explores lot more designs without increasing the total computational effort.
5. Among the three ESLMs investigated, ESL2 ranked first, ESL1 was close second and ESL3 was third based on the reliability of obtaining the best design.
6. Two additional procedures based on the nonlinear stiffness matrix are investigated in the ESLM to solve the nonlinear truss problem. In the first one, ESLs are calculated using the nonlinear stiffness matrix, but the linear static structural analysis is used during the ESL cycles. In the second one, nonlinear static structural analysis is used during the ESL cycles. The results showed no advantages over the method that used linear stiffness matrix for calculating ESLs and

during the ESL cycles (Sect. 7) in term of the quality of the final solution or the CPU time.

References

AISC (2017) Steel construction manual. American Institute of Steel Construction, Chicago

Arora J (1999) Optimization of structures subjected to dynamic loads. Structural dynamic systems computational techniques and optimization 7:1–73

Arora J (2017) Introduction to optimum design Introduction to optimum design, 4th edn. Elsevier Inc, Amsterdam

ASCE (2010) Design of blast-resistant buildings in petrochemical facilities. American Society of Civil Engineers

ASCE (2011) Blast protection of buildings Blast protection of buildings. ASCE/Structural Engineering Institute

Bhatti MA (2006) Advanced topics in finite element analysis of structures: with mathematica and matlab computations. Wiley, New York

Choi W-S, Park G-J (2002) Structural optimization using equivalent static loads at all time intervals. Comput Methods Appl Mech Eng 191(19–20):2105–2122

DoD (2008) Structures to resist the effects of accidental explosions. UFC 3-340-02

Dorigo M (1992) Optimization, learning and natural algorithms. PhD Thesis, Politecnico di Milano

Dusenberry DO (2010) Handbook for blast resistant design of buildings. Wiley, Hoboken

Eberhart RC, Shi Y, Kennedy J (2001) Swarm intelligence. Elsevier, Amsterdam

Geem ZW, Kim J. H, Loganathan G. V (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68

Gilsanz R, Hamburger R, Barker J, Smith D, Rahimian A (2013) Design of blast resistant structures. steel design guide no. 26l. American Institute of Steel Construction

Goldberg DE, Holland JH (1988) Genetic algorithms and machine learning

Kang B, Choi W, Park G-J (2001) Structural optimization under equivalent static loads transformed from dynamic loads based on displacement. Comput Struct 79(2):145–154

Kang B, Park G-J, Arora JS (2006) A review of optimization of structures subjected to transient loads. Struct Multidiscip Optim 31(2):81–95

Kaveh A (2014) Advances in metaheuristic algorithms for optimal design of structures. Springer, Berlin

Kaveh A, Ghazaan M (2014) Computer codes for colliding bodies optimization and its enhanced version. Int J Optim Civil Eng 4(3):321–332

Kaveh A, Mahdavi V (2014a) Colliding bodies optimization: a novel meta-heuristic method. Comput Struct 139:18–27

Kaveh A, Mahdavi V (2014b) Colliding bodies optimization method for optimum design of truss structures with continuous variables. Adv Eng Softw 70:1–12

Kaveh A, Mahdavi V (2014c) Colliding bodies optimization method for optimum discrete design of truss structures. Comput Struct 139:43–53

Kaveh A, Mahdavi V (2015) Colliding bodies optimization: extensions and applications. Springer, Berlin

Kim Y-I, Park G-J (2010) Nonlinear dynamic response structural optimization using equivalent static loads. Comput Methods Appl Mech Eng 199(9–12):660–676

Park G (2011) Technical overview of the equivalent static loads method for non-linear static response structural optimization. Struct Multidiscip Optim 43(3):319–337

Park G, Kang B (2003) Validation of a structural optimization algorithm transforming dynamic loads into equivalent static loads. J Optim Theory Appl 118(1):191–200

Park G, Lee Y (2019) Discussion on the optimality condition of the equivalent static loads method for linear dynamic response structural optimization. Struct Multidiscip Optim 59(1):311–316

Paz M, Kim YH (2019) Structural dynamics. Springer, Berlin

Stolpe M (2014) On the equivalent static loads approach for dynamic response structural optimization. Struct Multidiscip Optim 50(6):921–926

Stolpe M, Verbart A, Rojas-Labanda S (2018) The equivalent static loads method for structural optimization does not in general generate optimal designs. Struct Multidiscip Optim 58(1):139–154