25th ABCM International Congress of Mechanical Engineering
October 20-25, 2019, Uberlândia, MG, Brazil

SAP2000-MATLAB INTEGRATION FOR NONLINEAR DYNAMIC RESPONSE STRUCTURAL OPTIMIZATION

Rodrigo Bianchi Santos
Waldy Jair Torres Zuniga
Anderson Pereira
Pontifical Catholic University of Rio de Janeiro - Rua Marquês de São Vicente 225, Gávea, Rio de Janeiro, RJ - Brazil - 22451-900
rbianchi@aluno.puc-rio.br
wtorres19@aluno.puc-rio.br
anderson@puc-rio.br

Abstract. *Nonlinear phenomena are present in the majority of real-world structural problems. In structural optimization, several difficulties arise when nonlinear relations are considered in the problem formulation. Structural optimization with nonlinearities is challenging in different aspects, e.g. sensitivity analysis and numerous nonlinear analysis that must be evaluated throughout the optimization process. The present work approaches the structural optimization problem using the equivalent static loads (ESL) method. This method solves the nonlinear dynamic response structural optimization problem by performing sequential linear static response optimization sub-problems. SAP2000 software is utilized for nonlinear dynamic finite elements analysis. A MATLAB implementation performs both integration with the OAPI (Open Application Programming Interface) feature of SAP2000 and optimization by using the ESL method. The optimization of a linear truss under dynamic forces is presented as an initial validation of the implementation. The main objective of this work is to develop a robust routine for nonlinear dynamic response structural optimization. The code must be able to efficiently optimize large-scale problems as well.*

Keywords: *Structural Optimization, Nonlinear dynamic analysis, Equivalent Static Loads, SAP2000, MATLAB.*

1. INTRODUCTION

Structural analysis considering nonlinear effects and dynamic loads constitutes a very important class of problems, because almost every engineering structural problem happens to present one or both phenomena. Much research has been made in the past decades regarding to optimization of structures that present such behaviors (Park, 2011). In structural mechanics, nonlinearity is divided in three main types: material nonlinearity, geometric nonlinearity and contact nonlinearity (de Borst et al., 2012). In structural optimization, considering nonlinearity to obtain structural response is fundamental in order to obtain optimum solutions that better suits the real behavior of the structure.

A MATLAB code able to solve linear static response optimization problems with many load cases is presented, where the equivalent static loads are obtained from a nonlinear dynamics finite element analysis performed by SAP2000 software. The use of a commercial software for nonlinear dynamic response analysis is interesting because of its reliability and robustness. Other advantage is the versatility of the provided code regarding to adaptability to solving different nonlinear problems with less effort in changing boundaries and type of nonlinear analysis. Likewise, the ESL method reduces computational cost. The total number of nonlinear analysis throughout the optimization process is considerably lower, and the sensitivity analysis is also less expensive since a linear formulation is employed.

2. THEORETICAL BACKGROUND

2.1 Dynamic-response structural optimization

The general nonlinear dynamic response optimization problem is defined as follows:

$$\text{find} \quad \boldsymbol{b} \tag{1a}$$

$$\text{to minimize} \quad f(\boldsymbol{b}, t) \tag{1b}$$

$$\text{s.t:} \quad g_i(\boldsymbol{b}, t) \leq 0, i = 1, \dots, m \tag{1c}$$

$$\boldsymbol{b}_l \leq \boldsymbol{b} \leq \boldsymbol{b}_u \tag{1d}$$

$$\text{with} \quad \boldsymbol{M}(\boldsymbol{b})\ddot{\boldsymbol{z}}(t) + \boldsymbol{C}(\boldsymbol{b})\dot{\boldsymbol{z}}(t) + \boldsymbol{K}\big(\boldsymbol{b}, \boldsymbol{z}(t)\big)\boldsymbol{z}(t) = \boldsymbol{F}(t), \tag{1e}$$

where $\boldsymbol{b}$ is the design variables vector, $f(\boldsymbol{b},t)$ is the objective function to be minimized, g_i in the equation 1c is the i^{th} inequality constraint imposed on the optimization problem, $\boldsymbol{b}_l$ and $\boldsymbol{b}_u$ are the lower and upper bounds side constraints, respectively. $\boldsymbol{z}, \dot{\boldsymbol{z}}$ and $\ddot{\boldsymbol{z}}$ are the displacement, velocity and acceleration state vectors. Equation 1e is the nonlinear dynamic response equation of the system, this equation is solved for a finite number of time steps, $\boldsymbol{M}$, $\boldsymbol{C}$ and $\boldsymbol{K}$ are the mass, damping and stiffness matrices, obtained from the finite elements method (Weaver and Johnston, 1987).

2.2 The ESL method

The present work uses the equivalent static loads method for nonlinear dynamic response optimization (Kim and Park, 2010). By definition, equivalent static loads are loads that generate the same response fields in linear static analysis as in the original nonlinear dynamic response analysis. The dynamic load at each time step is considered as a single static load that is applied in static analysis. At each iteration of the ESL method, a new set equivalent loads are generated as the design variables update. When the method converges, the ESL obtains the same response field that the original dynamic loading, for all time steps. First, the generation of ESL for displacements is shown. In order to the response derived from these load cases to be the same as the dynamic load at all time intervals, it must verify

$$\boldsymbol{f}_{eq}^{z}(t) = \boldsymbol{K}_L \boldsymbol{z}_N(t), \tag{2}$$

where the ESL load vector $\boldsymbol{f}^z$ is obtained from multiplication of the linear analysis stiffness matrix $\boldsymbol{K}_L$ with the displacement vector $\boldsymbol{z}_N$ from nonlinear analysis. If $\boldsymbol{f}^z$ is applied as external loads in linear static analysis as:

$$\boldsymbol{K}_L \boldsymbol{z}_L = \boldsymbol{f}_{eq}^{z}(s), \tag{3}$$

then the displacement vector $\boldsymbol{z}_L$ is exactly the same as $\boldsymbol{z}_N$, i.e., the displacement fields in both linear static analysis and nonlinear dynamic analysis are identical. Note that there is a distinction in Equations 2 and 3 with respect to variables t and s because the former equation refers to dynamic analysis domain, so the independent variable is time t. On the other hand, the latter equation is written in static domain, and therefore variable s is interpreted as a load case variable.

When the linear dynamic response optimization problem is considered, one must simply replace the displacements from nonlinear analysis $\boldsymbol{z}_N$ with the displacement vector from linear analysis $\boldsymbol{z}_L$.

Even though the ESL calculated in Equation 3 generate the same displacement field, the stresses produced are not the same as from nonlinear analysis (Kim and Park (2010)), and therefore modifications are needed in the generation of such loads. For linear dynamic response optimization, the equivalent loads from Equation 3 generates displacement and stress responses with no further adjustments required.

When ESL are generated for nonlinear response stress fields, as it requires for a dynamic analysis with material nonlinearity, the displacements are calculated using a concept from initial stress analysis (Bathe, 2006) in order to the displacements from ESL generate the same stress field as in nonlinear analysis. The nonlinear stress is used as initial stress in linear analysis as:

$$\boldsymbol{K}_L \boldsymbol{z}_N = -\bar{\boldsymbol{f}}_I\big(\boldsymbol{\sigma}_N(t)\big) = \boldsymbol{f}_{eq}^{\sigma}. \tag{4}$$

where $\boldsymbol{\sigma}_N$ is the nonlinear stress response from the original dynamic analysis. Figure 1 illustrates flowchart of the ESL generation for same stress field in nonlinear analysis. After the process initializes, for a given k iteration, first the dynamic analysis is performed in SAP2000, then the outputs are processed in MATLAB and equivalent static loads for displacement or stresses fields are generated. Once the equivalent static loads are calculated, the solution for the original nonlinear dynamic response problem is achieved by solving sub-problems of linear static response optimization. The equivalent loads are applied as multiple load cases in a linear static analysis and an optimization routine is carried out. If the difference between two consecutive objective functions is less than the established tolerance, the process terminates. Otherwise, a new iteration (or cycle) of the ESL method is made with the updated design variables.

The ESL method states that the solution found when the process converges satisfies the KKT conditions, and it is the solution of the original problem (Park and Kang, 2003).

3. NUMERICAL IMPLEMENTATION

The structural optimization process is divided in two main parts: first, the finite element model is generated and the nonlinear dynamic analysis is carried out in SAP2000 using Newmark‘s method for the direct integration of the finite element model. The communication between both software is made via the OAPI feature (CSI, 2019) of SAP2000. OAPI is a feature that allows the connection of SAP2000 with other commercial software. Calling internal functions is made by means of "SAP2000.exe" and "ActiveX" command. The latter command plays the role of establishing the communication between SAP2000 and the external software to work with.

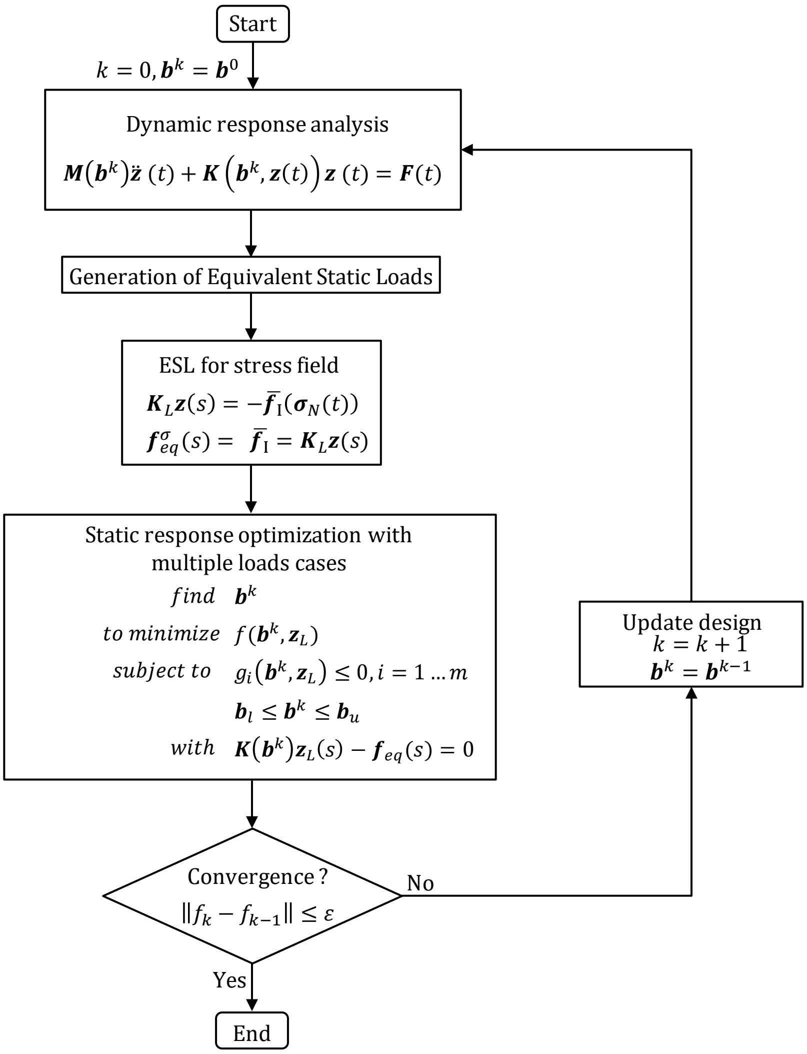

Figure 1. Flowchart for the generation of ESL for stress fields for nonlinear analysis.

Next, the displacements and axial forces are exported to MATLAB and the optimization with ESL's is performed. When the design vector is updated, the process starts again and a new analysis is carried out with the new design values. These two parts are repeated in a cyclic fashion until the change in the objective in two consecutive iterations is less than a given tolerance, as defined by Equation 5,

$$\|f_k - f_{k-1}\| \leq \varepsilon, \tag{5}$$

where in the present work the tolerance $\varepsilon = 10^{-5}$ is adopted. SAP2000 is used to perform linear and nonlinear dynamic analysis. For a truss element, it is necessary to release the rotational degrees of freedom in order to consider such element. Geometry and boundary conditions are set via OAPI. The loads are defined by time history function, as a discrete load for defined time steps or even as a predefined function in SAP2000.

When considering an analysis with material nonlinearity, such relation is defined in SAP2000. Also, a property called "link" needs to be defined in order to represent the energy loss by hardening. The link property has six degrees of freedom associated with each node that can be individually set to represent either linear or nonlinear behavior (Nguyen and Kim (2014)). In the present work, link property is set to "Multilinear Plastic" (CSI (2019)), whereas the material nonlinearity is generated through the definition of a load-displacement curve. No damping is considered in the development. The Newmark parameters for time integration of the dynamic response are $\beta = 0.25$ and $\gamma = 0.5$.

3.1 MATLAB implementation – ESL: code description

In the following, a brief description of the main parts of the code developed in order to perform the complete optimization process using the ESL method is presented. The code is developed in MATLAB and the OAPI feature is used to establish external communication with SAP2000. With such feature, it is possible to run the finite element model in SAP2000 by passing commands from MATLAB and perform the dynamic response analysis. After that, the output fields are retrieved from SAP2000, the equivalent static loads are defined and then the static response linear optimization in MATLAB is performed by using the native function `fmincon` from the MATLAB optimization toolbox. This routine is repeated with updated design variables until convergence is reached.

The routine begins selecting the `.sdb` executable SAP2000 file, then the file name is read and current directory path is saved via `uigetfile` MATLAB command as follows:

```
[filenam,pathnam] = uigetfile({'*.sdb'},'Selecionar o Arquivo');
```

In order to run SAP2000 as external application, it requires to reference SAP2000 from the application and create an object within MATLAB, as below:

```
ProgramPath = [pwd '\SAP2000 21\sap2000.exe'];
APIDLLPath = [pwd '\SAP2000 21\sap2000v20.dll'];
a = NET.addAssembly(APIDLLPath;
helper = SAP2000v20.Helper;
helper = NET.explicitCast(helper,'SAP2000v20.cHelper');
SapObject = helper.CreateObject(ProgramPath);
SapObject = NET.explicitCast(SapObject,'SAP2000v20.cOAPI');
feature('COM_SafeArraySingleDim', 1);
feature('COM_PassSafeArrayByRef', 1);
```

where the first two command lines are the SAP2000 installation directory path and `pwd` command identifies the current folder, which is set to SAP2000 installation path.

Since that an instance of the SAP2000 object is defined, it can be started using command `SapObject.ApplicationStart;` once the application is running, an existing model can be opened, or a new one be created. OAPI commands usually uses an object `SapModel` to perform any desirable actions to it. It can be defined as follows:

```
SapModel = NET.explicitCast(SapObject.SapModel,'SAP2000v20.cSapModel');
File = NET.explicitCast(SapModel.File,'SAP2000v20.cFile');
```

The two lines above initialize the SAP2000 model object. Then an existing model file is opened with:

```
File.OpenFile(filenam);
```

When ESL cycles begin, SAP2000 performs the dynamic analysis and updates the design variables. In order to update the variables, the section type must be defined and assigned to the correspondent truss element. It is accomplished by first defining the `PropFrame` object:

```
PropFrame = NET.explicitCast(SapModel.PropFrame,'SAP2000v20.cPropFrame');
FrameObj = NET.explicitCast(SapModel.FrameObj,'SAP2000v20.cFrameObj');
```

For the geometry selection of the frame section, the function `SetCircle` is used. This function enables a solid circular frame section properties to be defined . The design variables are updated inside the following loop:

```
for i = 1:Ne
    Diameter = sqrt(4*Area(i)/pi);
    ret = PropFrame.SetCircle(section,material,Diameter);
    FrameID = num2str(i);
    ret = FrameObj.SetSection(FrameID,section);
end
```

where `Ne`, `FrameID` and `section` are the total number of elements, the element number identifier to each cross-section and the name of an existing property to modify. When a property section is added or modified it is necessary to be assigned to the corresponding element in the structure with `SetSection` command.

The following step is necessary release the rotational degrees of freedom for each element using the follows commands:

```
SelectObj = NET.explicitCast(SapModel.SelectObj,'SAP2000v20.cSelect');
FrameObj = NET.explicitCast(SapModel.FrameObj,'SAP2000v20.cFrameObj');
ret = SelectObj.All;
```

```
for i = 1:Ne
    FrameID       = num2str(i);
    pointi        = NET.createArray('System.Boolean',6);
    pointj        = NET.createArray('System.Boolean',6);
    pointj(4)     = true;
    StartValue(4) = 0;
    pointi(5)     = true;
    pointj(5)     = true;
    StartValue(5) = 0;
    EndValue(5)   = 0;
    pointi(6)     = true;
    pointj(6)     = true;
    StartValue(6) = 0;
    EndValue(6)   = 0;
    ret = FrameObj.SetReleases(FrameID, pointi, pointj, StartValue, EndValue);
end
```

where the positions 4, 5 and 6 are the rotational degrees of freedom indicating the initial and final joint in each element.

Now, can be run the analysis and generate the results, by using:

```
Analyze = NET.explicitCast(SapModel.Analyze,'SAP2000v20.cAnalyze');
Analyze.RunAnalysis();
AnalysisResults = NET.explicitCast(SapModel.Results,'SAP2000v20.cAnalysisResults');
AnalysisResultsSetup = NET.explicitCast(AnalysisResults.Setup,...
'SAP2000v20.cAnalysisResultsSetup');
```

In order to retrieve the outputs to MATLAB and follow to the optimization phase, the command in the following is needed:

```
ret = AnalysisResultsSetup.SetOptionDirectHist(2);
```

where option 2 saves the output step-by-step-wise. After importing the SAP2000 displacement and axial forces outputs, the ESL are generated, and the axial stress are calculated. The commands `AnalysisResults.JointDispl` and `AnalysisResults.FrameForce` reports the joint displacements for the specified point elements and the axial forces in each element from SAP2000.

After the model is finished, it needs to be saved and closed, respectively with the following commands:

```
ModelPath = strcat(cd, filesep,filenam);
ret = File.Save(ModelPath);
ret = SapObject.ApplicationExit(false);
```

The `ret` in the functions used in the OAPI SAP2000 should returns zero if the command is successfully defined, otherwise it returns a nonzero value.

Finally, the optimization is carried out using MATLAB native function `fmincon`. The principal inputs for this function are the objective function, the constraints vector and initial point vector. Moreover, additional configurations can be passed as inputs in order to select, for example, the desirable algorithm for finding the optimum solution.

After the optimization process is completed, the convergence criterion is checked in order to verify whether a new cycle of ESL method is required or not.

3.2 Six-bar truss under stress, displacement and natural frequency constraints

A linear truss subject to dynamic loading is shown as an initial validation for the developed routine. The example follows from Park (2011), as shown in Figure 2. The problem is formulated to minimize the total mass and the truss is subjected to displacement, stress and natural frequency constraints. Under linear response analysis, the equivalent loads generated for displacement response and the addition of natural frequency constraint does not change how ESL are obtained. Therefore, the frequency constraint equation is inserted with no further modifications required either in the optimization formulation or in the generation of ESL. Material parameters are $E = 70\ GPa$ and $\rho = 2700\ \ kg/m^3$. The maximum stress in each bar should not be greater than $\bar{\sigma} = 3.5\ MPa$, the maximum allowed vertical displacement in node 5 is $\bar{u} = 3.5 \times 10^{-3} m$ and the first natural frequency is set to be greater than $\bar{f} = 10\ Hz$. The lower and upper bound for each element area are $\underline{b} = 1.0 \times 10^{-4}\ m^2$ and $\overline{b} = 4.0 \times 10^{-4}\ m^2$, respectively. More details about the dynamic analysis can be found also in Figure 2. The analysis time is discretized in 75 time steps, each one with $0.15\ s$ of duration. Therefore, following the ESL method, 75 load cases are generated for the sequential static linear optimization sub problems. In the following is shown the equivalent static response linear optimization problem present in each cycle:

$$\text{minimize} \quad f(\boldsymbol{b}) = \sum_{e=1}^{6} \rho l_e A_e(\boldsymbol{b}) \tag{6a}$$

$$\text{subjected to} \quad g_{6(s-1)+e}(\boldsymbol{b}, s) = \|\sigma_e(\boldsymbol{b}, s)\| \leq \bar{\sigma}, \qquad e = 1, \dots, 6, s = 1, \dots, 75 \tag{6b}$$

$$g_{450+s}(\boldsymbol{z}_s(\boldsymbol{b}, s)) = \|z_{s,5}^{y}(\boldsymbol{b}, s)\| \leq \bar{u}, \qquad s = 1, \dots, 75 \tag{6c}$$

$$g_{526}(\boldsymbol{b}) = \frac{\sqrt{\lambda_1}}{2\pi} \geq \bar{f} \tag{6d}$$

$$\underline{b} \leq b_j \leq \bar{b}, \qquad j = 1, \dots, 6 \tag{6e}$$

$$\text{With} \quad (\boldsymbol{K} - \lambda_1 \boldsymbol{M})\boldsymbol{v}_1 = 0 \tag{6f}$$

$$\boldsymbol{K}\boldsymbol{z}_s = \boldsymbol{F}_s, \qquad s = 1, \dots, 75. \tag{6g}$$

where $z_{s,5}^{y}$ is the vertical displacement in node 5, λ_1 and $\boldsymbol{v}_1$ are, respectively, the eigenvalue associated with the first natural frequency and the correspondent eigen vector. $\boldsymbol{z}_s$ is the displacement field in linear static analysis with load cases $\boldsymbol{F}_s$, calculated from Equation 2. Equation (6g) represents the linear static analysis subjected to load cases assembled in $\boldsymbol{F}_s$, and $\boldsymbol{z}_s$ is the static response vector. When the ESL method converges, $\boldsymbol{z}_s$ is equal to $\boldsymbol{z}(t)$ obtained in dynamic response optimization. The constraints vector $\boldsymbol{g}$ is assembled by first stacking stress constraints for all elements, followed by vertical displacement constraints of node 5 where load $\boldsymbol{F}$ is applied (both constraints are applied in all load cases). The last element of the constraint vector is the structure's natural frequency constraint equation.

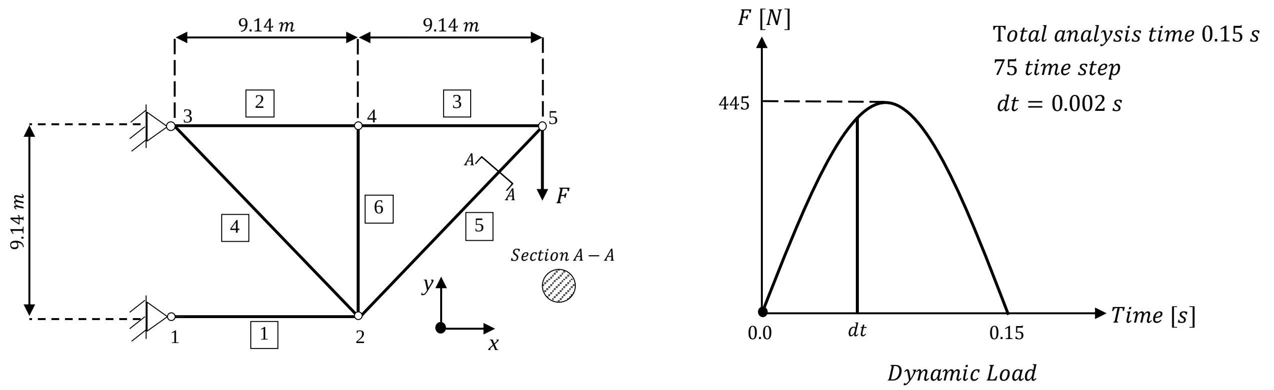

Figure 2. Six-bar truss scheme and applied load in node 5.

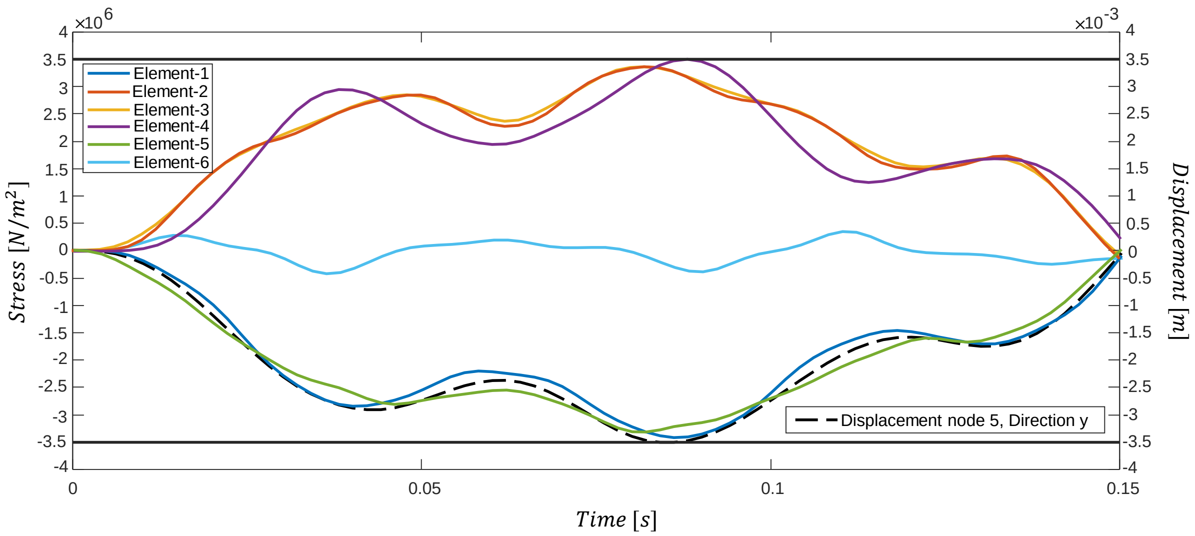

Figure 3. Stresses (bold lines) and vertical displacement of node 5 (dashed line) constraints attended in optimum design using the equivalent static load method.

It was necessary 6 cycles of ESL method in order to the convergence criterion to be verified. Table 1 compares the results obtained for the parametric optimization under the considered constraints with the work done by Park (2011). It is convenient to remark that the initial mass is slightly different from the reference. The mass obtained for the optimized structure in the present work is consistent with the value from literature, and the same can be said about the attended constraints and individual design variables.

Figure 3 shows the vertical displacement at node 5 and stress for all bars, along time, for the optimized structure. By the evaluation of the obtained results, the achieved results of the present framework are consistent with the literature studied. Therefore, the six-bar truss example is used as an initial validation for the linear dynamic response optimization using the ESL method.

Design Cycle	Initial Design	Cycle 1	Cycle 2	Cycle 3	Cycle 4	Cycle 5	Cycle 6	Park (2011
Mass (kg)	32.84	32.18	33.46	33.46	33.46	33.46	33.46	33.26
Area 1 (cm^2)	1.94	3.03	3.15	3.16	3.15	3.15	3.155	3.17
Area 2 (cm^2)	1.94	1.45	1.53	1.52	1.52	1.52	1.52	1.52
Area 3 (cm^2)	1.94	1.49	1.51	1.51	1.51	1.51	1.51	1.50
Area 4 (cm^2)	1.94	2.25	2.36	2.35	2.35	2.35	2.35	2.22
Area 5 (cm^2)	1.94	2.03	2.10	2.10	2.10	2.10	2.10	2.18
Area 6 (cm^2)	1.94	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Displacement node 5 (cm)	0.398	0.349	0.349	0.349	0.349	0.349	0.349	0.350
Maximum stress (MPa) $(Element)$	5.37 (1)	3.49 (4)	3.49 (4)	3.49 (4)	3.50 (4)	3.50 (4)	3.50 (4)	3.50(4)
Frequency (Hz)	17.22	21.34	21.41	21.41	21.41	21.41	19.38	20.9

Table 1. Results comparison for the six-bar truss.

3.3 Ten-bar truss with material nonlinearity

The second example comprises a ten-bar truss, Kim and Park (2010), subjected to stress constraint that considers that there is a nonlinear relation between stress and strain. The example consists in minimizing the structure's mass that is subject to stress constraints. The stress-strain curve, as well as information about linear and tangent modulus are shown also in Figure 4. The nonlinear analysis is carried out in SAP2000 using link elements at each elements connection in order to reproduce the bi-linear behavior in Figure 4.

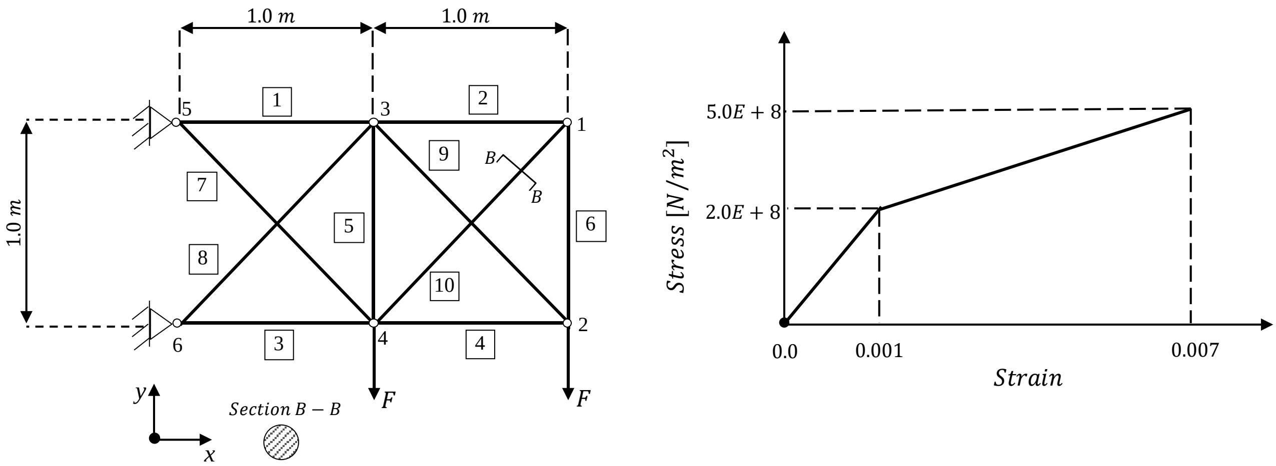

Figure 4. Ten-bar truss and bilinear elastoplastic strain-stress nonlinear relation.

The allowed axial stress in either traction or compression is set to $250\ MPa$. Material density is $\rho = 7860 kg/m^3$. The lower and upper bounds for all bars are $\underline{b} = 7.85 \times 10^{-5}\ m^2$ and $\bar{b} = 2.83 \times 10^{-3}\ m^2$, and the initial point for every bar is set to be $b_0 = 3.14 \times 10^{-4}\ m^2$. The total time of analysis is $0.03\ s$ divided in 150 time steps and the duration of each time step is $0.0002\ s$. The vertical load applied in nodes 2 and 4 is a pulse with amplitude equals to $50\ kN$ and duration of $0.0033\ s$. From this moment on, no external loads are applied and the structure undergoes a free vibration regime. Equation 7 shows the static linear response optimization sub problem that is solved in each ESL cycle:

minimize $$f(\boldsymbol{b}) = \sum_{e=1}^{10} \rho l_e A_e(\boldsymbol{b}) \quad (7a)$$

subjected to $$g_{10(s-1)+e}(\boldsymbol{b}, t) = \|\sigma_e(\boldsymbol{b}, s)\| \leq \bar{\sigma}, \qquad e = 1, \ldots, 10, s = 1, \ldots, 150 \quad (7b)$$

$$\underline{b} \leq b_j \leq \bar{b}, \qquad j = 1, \ldots, 10 \quad (7c)$$

with $$\boldsymbol{K}\boldsymbol{z}_s = \boldsymbol{F}_s, \qquad s = 1, \ldots, 150. \quad (7d)$$

The link property of SAP2000 is responsible for the decay of the stress amplitude, as expected in a material nonlinear analysis. After nonlinear analysis, the displacement and stress fields are retrieved from SAP2000 in order to generate ESL's for stress fields. Table 2 shows the results obtained in the present work and a comparison with Kim and Park (2010). The results are very consistent with the reference.

Design Cycle	Initial Design	Final Design	Kim and Park (2010)
Mass (kg)	28.77	21.85	21.77
Area 1 (cm^2)	3.140	5.088	4.976
Area 2 (cm^2)	3.140	0.785	0.955
Area 3 (cm^2)	3.140	4.596	4.806
Area 4 (cm^2)	3.140	1.697	1.569
Area 5 (cm^2)	3.140	0.785	0.786
Area 6 (cm^2)	3.140	0.785	0.786
Area 7 (cm^2)	3.140	3.172	3.163
Area 8 (cm^2)	3.140	3.627	3.368
Area 9 (cm^2)	3.140	2.358	2.099
Area 10 (cm^2)	3.104	0.785	1.138
Maximum stress (MPa)	365.3	250.0	249.0

Table 2. Results comparison for the ten-bar truss.

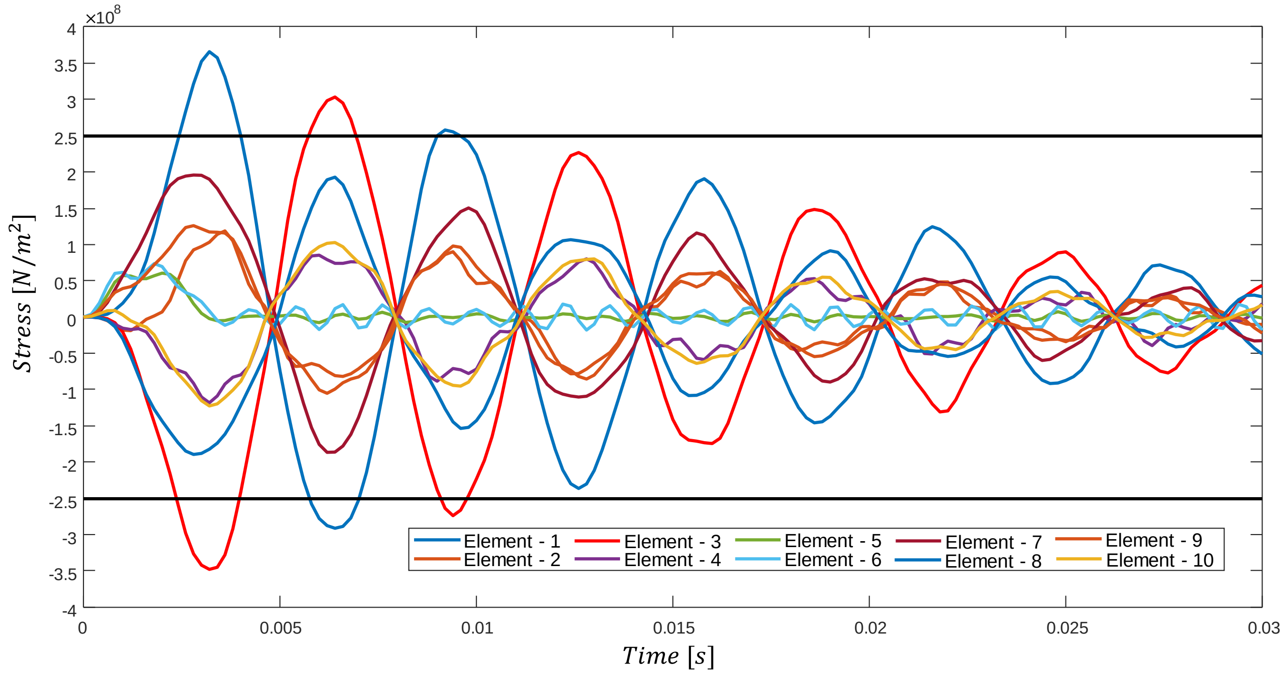

Figure 5. Axial stress in each element versus time at initial design for the ten-bar truss with constraints violated.

Figure 5 shows the axial tress for all elements at the initial design. The axial stress constraint is violated at initial design, since the amplitude peak for two bars are greater than $\bar{\sigma}$. Figure 6 shows the axial stress for the optimized structure along time. The constraints have been attended for all analysis, and the attenuation in stress amplitude due to material nonlinear behavior is observed.

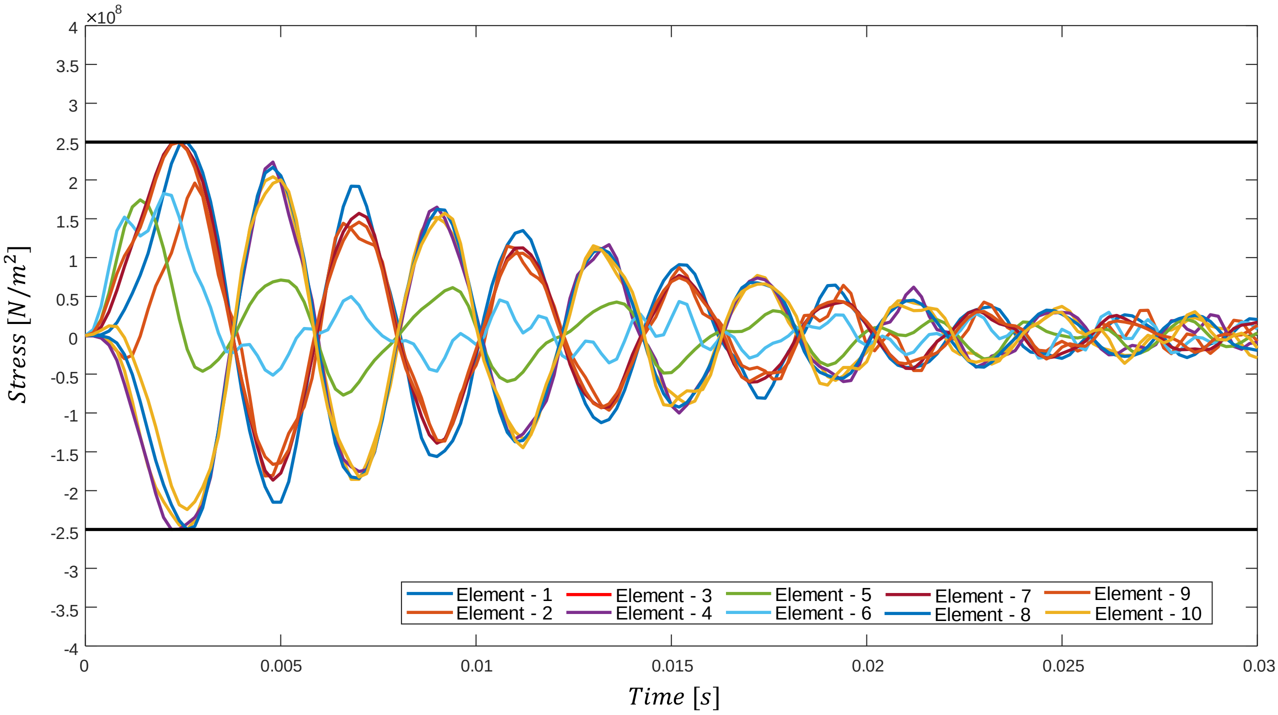

Figure 6. Ten-bar truss stress nonlinear response at optimum design. Constraints attended using the ESL method.

4. CONCLUSIONS

The Equivalent Static Loads (ESL) method is used in order to solve linear and nonlinear dynamic response parametric optimization of truss structures. The equations that generate equivalent loads for displacement and stress constraints are presented. A routine in MATLAB was developed in order to perform an integration with SAP2000 software to perform the dynamic response analysis via finite elements and direct integration. The optimization module is carried out within MATLAB with a native optimization toolbox, using retrieved responses from SAP2000. The OAPI feature, present in SAP2000, was utilized in order to enable the communication between both software. The principal commands that OAPI feature uses in order to perform the integration with an external application, are presented, in a manner that an initial guideline facility is presented for a first-contact user. Advantages of a commercial FEM software are robustness and reliability of a finite element analysis source, as well as the adaptability to different structures, since changes required imply small effort when compared to the implementation of codes for a new geometry or boundary conditions. The routine developed was used to solve classical linear and nonlinear benchmark examples found in literature. The parametric optimization of truss structures with dynamic loads and under displacement, stress and natural frequency using the ESL method are studied and results are consistent with presented literature. The current state of the present work follows with studying and solving large-scale problems that constitute a more realistic picture of real-world structural engineering problems.

5. ACKNOWLEDGEMENTS

The authors would like to express their thanks for the financial support provided by CNPq (National Council for Scientific and Technological Development) and CAPES (Coordination for the Improvement of Higher Education). Anderson Pereira is thankful for the support from FAPERJ (Research Support Foundation of Rio de Janeiro State) under grant E-26/203.189/2016.

6. REFERENCES

Bathe, K., 2006. *Finite Element Procedures*. Prentice Hall. ISBN 9780979004902.

Choi, W.S. and Park, G.J., 2002. "Structural optimization using equivalent static loads at all time intervals". *Computer Methods in Applied Mechanics and Engineering*, Vol. 191, No. 19-20, pp. 2105–2122.

CSI, 2019. "CSI, A. Open application programming interface (OAPI)". <https://wiki.csiamerica.com/display/kb/OAPI>.

De Borst, R., Crisfield, M.A., Remmers, J.J. and Verhoosel, C.V., 2012. Nonlinear finite element analysis of solids and structures. John Wiley & Sons.

Kim, Y.I. and Park, G.J., 2010. "Nonlinear dynamic response structural optimization using equivalent static loads". *Computer Methods in Applied Mechanics and Engineering*, Vol. 199, No. 9-12, pp. 660–676.

Lagaros, N.D. et Al, 2019. "A C# code for solving 3D topology optimization problems using SAP2000". *Optimization and Engineering*, Vol 20, Issue 1, pp. 1-35.

Nguyen, P.C. and Kim, S.E., 2014. "Nonlinear inelastic time-history analysis of three-dimensional semi-rigid steel frames". *Journal of Constructional Steel Research*, Vol. 101, p. 192–206. doi:10.1016/j.jcsr.2014.05.009.

Park, G. and Kang, B., 2003. "Validation of a structural optimization algorithm transforming dynamic loads into equivalent static loads". *Journal of optimization theory and applications*, Vol. 118, No. 1, pp. 191–200.

Park, G.J., 2011. "Technical overview of the equivalent static loads method for non-linear static response structural optimization". *Structural and Multidisciplinary Optimization*, Vol. 43, No. 3.

Weaver, W. and Johnston, P., 1987. Structural Dynamics by Finite Elements. Prentice-Hall civil engineering and engineering mechanics series. Prentice-Hall. ISBN 9780138535087.

7. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.