How to use

Topology optimization, BESO method

BESO (Bi-directional Evolutionary Structural Optimization) method is one of methods, which can be used for a topology optimization with connection to FEM (Finite Element Method). BESO method works in iterations where in the each iteration least effective elements are removed (and some elements removed earlier can be added back). This method can be used in an initial phase of product design. It does not mean that product or part is completely designed by the program, but it can give an initial guidance how should a designer place material to be efficiently used. It is important to keep in mind that, firstly, it is a heuristic method so there is no certainty that given solution is the real optimum (i. e. the best solution). Secondly, BESO method needs some input parameters, which influences result quality and speed of convergence.

Presented program uses CalculiX solver and was at the beginning inspired by the algorithm described in the book: Evolutionary topology optimization of continuum structures, Methods and applications 2008, written by X. Huang and Z.M. Xie. The goal of the presented program is to achieve the design with low weight. The most important modification is in formula for computing sensitivity number (which is the measure of effectiveness of the given element in the given iteration). It is computed as a rate of von Mises stress and original material density. This has no sophistical theoretical background. It simply suggests that elements with high stress and low density are used more effectively than elements with low stress and/or high density.

Current implementation idea

Program is intended to work under the FreeCAD, but this version is done to work independently. User prepares input file in the same way as for a common static analysis, than in the beso_conf file sets input parameters, which consist of material inputs of solid sections (shell sections) and parameters for the optimization. The program copies .inp file and redefines material properties of the given sections (according to their element set – elset – name given by the user). Smaller change is that new .inp file is written only with output requests for stresses in integration points, which are used for the optimization. For each new iteration optimization program penalize several elements with the least sensitivity number by multiplying their elastic modulus by a very low „void_coefficient“ constant, which practically removes them from the next iteration. After achieving prescribed volume or allowable stress limit or maximal number of iterations, resulting mesh is saved to the file in the classical CalculiX output format (.frd). Complete description of the implementation would be quite long, thus below is only a simple example presenting possibilities which will hopefully help users to understand how to use the program.

Possibilities / limitations

Optimization domains are chosen by the user, so there can be domains from which material will not be removed.

Volume of the optimization domain(s) is removed until the goal volume (fragment of the initial

volume) is achieved.

Each domain can have prescribed maximal allowable stress. If this stress is achieved, current volume is frozen.

Optimization can take in to account only volume elements C3D4 and C3D10, shell elements S3 and S6. However, .inp file can contain also other types, but they will not be included in optimization (element removing). Centers of gravity and volume of 2nd order elements are, in the current version, computed in the same way as for 1st order elements, which leads to certain error.

It is possible to define different materials for optimization domains, but these materials must be isotropic linear elastic.

Manufacturing constraints are not implemented. It is possible to set only filter range which prevents „checkerboard effect“ caused by FE mesh and non-directly influences minimal size of a material spot.

Multiple load cases can be modeled by adding more steps into .inp file. By default CalculiX adds loads and boundary conditions of current step to those from previous step. Concentrated loads can be reseted by line *CLOAD, OP=NEW, distributed loads by *DLOAD, OP=NEW and boundary conditions by *BOUNDARY, OP=NEW.

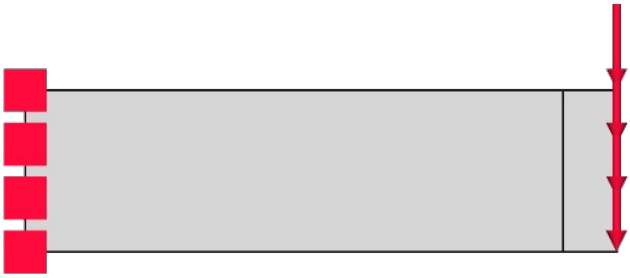
Running of the program is monitored by mean stress and volume of the optimization domain, and maximal stresses in the each domain separately. These values are graphed at the end and the resulting mesh is saved, but mesh can be saved even more often to enable better monitoring of the process.

Example 1: cantilever 2D beam

This example demonstrates the workflow and basic settings of this optimization. For simplicity and low computational costs, only 2D case is modeled. As an input we need 2D rectangular mesh, which will be divided into 2 domains. First domain will be for optimization; second domain with applied load is considered not to be optimized. Whole model is from steel.

FEM model

Mesh was generated by a GMSH macro in the FreeCAD on the fusion of two plane objects 100 x 30 mm and 10 x 30 mm (according to 2 domains; this approach is not the best, but it is viable for this example). The maximum element size was set to 1 mm and only 1st order was used. The left edge is fixed, the right edge is loaded by a force 100 N. Since it is needed to define 2 domains, we define first material for the first domain and second material for the rest (with identical properties). This enforces FreeCAD to create different element sets in the .inp file, which we need. Shell thickness is set to 1 mm for the whole model (shell thicknesses can be used in the same way as materials to enforce creating two element sets).



Now it is possible to check the model by running the analysis.

Optimization settings

Parameters setting is done in the file beso_conf.py. It contains python code with input parameters and comments to follow. Set path to CalculiX, path to working directory and file name of the input (.inp) for the solver generated by FreeCAD or other preprocessor. Next block is for the domain definition, where you define material properties – this will during iteration rewrite shell section (solid section), no matter values which were set in preprocessor. First variable, domain_elset, contain the name of an element set from which elements will be taken into that domain. You should find the name in the .inp file for example by opening in a text editor and ctrl+f for the „elset“. FreeCAD generates something like „MechanicalMaterialShellThickness“ and „MechanicalMaterial001ShellThickness“. Than set True for optimized domain resp. False for non-optimized domain. Material properties should be in units consistent with the model. You can set these values

E = 210000

Poisson's number = 0.3

density = 7.9e-9

thickness = 1.0

offset = 0.0

stress allowable = 0.0 (stress allowable will not be enforced)

The block of code for the domain variables is copied to define second domain. It differs only in the domain elset and domain_optimized. Next, for our example volume_goal was set to 0,4 (we would like to obtain 40% of the initial volume of the optimization domain) and radius of the filter r_min to 2.0 which is double size of the mesh element size. In advanced inputs you can let default values. These parameters can influence our solution

use_filter = 1

evolutionary_volume_ratio = 0.015

volume_additional_ratio_max = 0.05

iterations_limit = 0 (this means automatic)

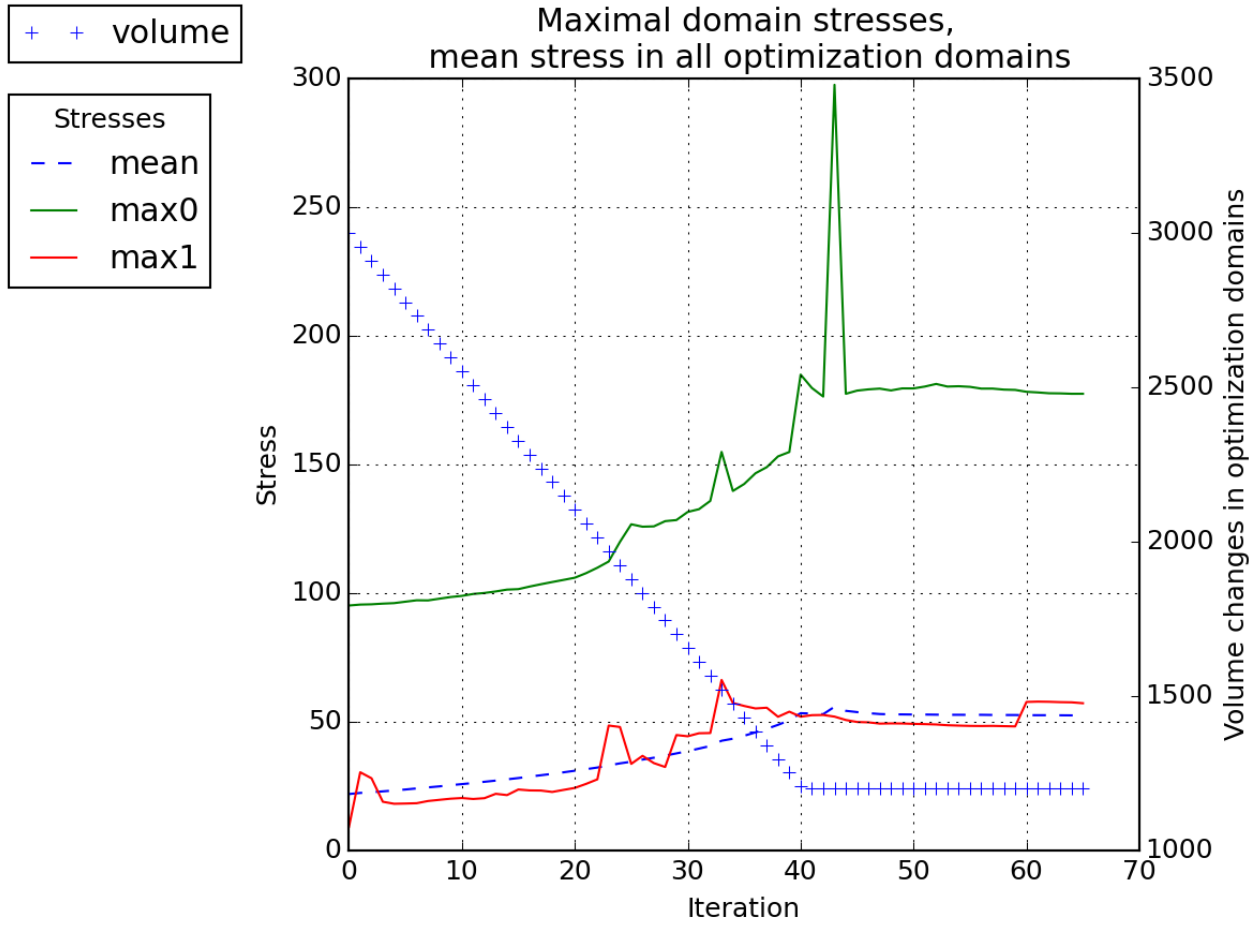
tolerance = 1e-3

void_coefficient = 1e-6

Ongoing result saving is driven by save_iteration_meshes = 10 which will enable to visualize every 10th iteration mesh.

Running and postprocessing

Run the file beso_main in your working directory (same as set in configuration file). Files beso_lib and beso_conf should be also in this directory. At the end you should get graphs (stresses_and_volume.png) which show the history of volume in optimization domain, mean stress in optimization domain and maximal stresses in all defined domains.



Smooth lines of the maximal stresses and the mean stress during first 40 iterations show a fluent convergation. Increase in them is typical, because material is removed. If the optimization finished successfully, you can drug and drop generated .frd file with resulting mesh (only a mesh without stresses) to import it back into the FreeCAD.

To get a better image of the solution progress, you can import also files file011.frd, file021.frd, … A stress peak in the iteration 43 is given by disappearing relatively big connection, which led to the increase of the stress.

Result notes

1. Although effectiveness of the material increases, overall stiffness of the part decreases due to material removing.
2. The resulting mesh has not smooth boundary, so you would probably use it for inspiration how to create real part instead of sending it to manufacture as it is.
3. Output of the stress distribution is not supported by now.
4. Stresses used in the algorithm are von Mises stresses of the elements averaged over integration points of each element. They are different from stresses in nodes.
5. Quality of the result depends on the input parameters and the mesh. For comparison between different settings, mean and maximal stresses at the end can be used (if maximum is not given by concentrated load). In case of the same final volume lower stress may point to better result.

Conclusion – back to the beginning

After getting rough image how it works, it is desirable to mention what to consider before the start of your own tries:

1. Decide what is non-optimized domain(s) (which should remain untouched) and where is optimization domain(s) (where elements will be removed). Also decide how much material should be removed from them and if there is some stress limit.
2. Size of a requested material spot. This size can be used as a range of the filter and than a mesh element size might be half of them (this ratio comes from tries of older code version).
3. Computational costs and memory limitations: Python script consumes a significant ammount of memory beside CalculiX. Using a large filter range (in compare to the mesh element size) can significantly increase memory demands. Computing tens of iterations (means tens of full analysis) also takes corresponding time. For this example it was more

than 100 MB (Python) + 100 MB (CalcuilX) for only 4463 nodes and 8644 shell elements.

4. Keep in mind that this code is under development, not fully tested, and the method does not have to give a real mathematical optimum.

Known issues

Full (not removed) elements may be slightly stuck to the domain face because elements on a straight mesh boundary could be more effective than elements along an algorithm created bumpy boundary.

A card *INCLUDE in .inp file is not supported yet.

Elements in ELSET are loaded to algorithm domains only if their numbers are listed. Link to other ELSET is ignored except EALL, which loads all supported types of elements.