Condensation Based Finite Element Model Reduction Strategy for Structural Optimization Problem

KAKADE Ishaan

SIMONE Coniglio, MORLIER Joseph

# Abstract

The project deals with topology optimization with respect to the structure of an engine pylon on a standard aircraft. Using the finite element method, the engine is representatively modeled to then project a representative load onto the design zone i.e. the pylon, which will then undergo a topological optimization. To account for engine architectures with variable thicknesses, we first integrate geometrical aspects of the engine to the decision variables of the design zone. To further simplify the representative engine, we use static and dynamic condensation techniques to suppress certain degrees of freedom to reduce the time complexity associated with the optimization problem.

**Keywords:** topology optimization, static condensation, dynamic condensation, engine pylon.

# Introduction

Aircraft engine, pylon and engine mounts design is often conducted by different teams and even different companies. Both engine and mounts architecture contribute to the final assembly mass and stiffness of the pylon. In a preliminary phase one would like to know if sizing the engine structure and pylon architecture simultaneously could improve the final assembly. In this context the engine structure can be considered as an input of the problem, since it is often fixed by primary vein fluid-dynamic constraints. On the other hand, the pylon and engine mounts architecture can be considered as more free. The engine structure may be optimized choosing casing thicknesses; on the other hand pylon architecture can be optimized from sketches using topology optimization (see Fig. 1R). In order to efficiently deal with this problem the condensation of engine is often required. This is simple for engine with fixed design but is not as straightforward for variable thickness case and a suitable condensation strategy should be developed here. To develop and test our strategy a very simple structure composed by 2D planar stress elements and a beam truss is studied (see Fig. 1R). This model will be employed for a simultaneous optimization of beam cross sections and 2D topology optimization. The objective will be to minimize the assembly compliance with different mass constraints in a static load cases.

# Problem Definition

The stock code needs to be modified such that the variable thicknesses of the engine can be taken into account in the output of the topology optimization. Further, using static and dynamic condensation, we can reduce the size of the stiffness matrix of the beam truss and thus reduce the time complexity of the problem.

# 1.2 Motivation

During topological optimization processes we use the whole finite element model as our design space (all nodes or degrees of freedom) to carry out optimization process. For large structures with millions of nodes, it can prove challenging in terms of computation time to arrive at a feasible structure within a reasonable span of time. Even with gradient approaches and a reduction in the function calls within the solver, the sheer volume of design variables makes the time complexity of the problem vast. To reduce the scope of the problem means to condense the design space where we know we need an optimization of the structure. For example, for an aircraft engine pylon, we can ignore the design space within the engine and the wing and suppress those degrees of freedom whilst making our optimization solver solve for the design area we are interested in. In some cases, we can reduce the suppress 90% of the design variables and maintain a high degree of accuracy in the dynamic behavior of the structure [1].

*Figure 1: Left - Original FEM engine model; Right - Simplified FEM model*

**Primary Goal:** The overall goal of this project is to establish a strategy to reduce the complexity of a finite element problem used for optimization purposes by the use of some (or a combination) of reduction strategies to suppress some degrees of freedom while maintaining the full response – prediction of those degrees of freedom that have been retained. Given that our problem deals with the engine pylon we can think of narrowing down our goal to the optimization of structures that are loaded as such.

For the primary goal, we can breakdown the tasks as follows:

1. Integration of A and I as MMA decision variables (for the complete beam truss and by set i.e. vertical and horizontal beams separately).

2. Implement the classical condensation and a new condensation strategy to reduce the number of degrees of freedom of the non-design zone.

3. Comparison of 3 approaches with respect to time complexity.

4. Implementation of dynamic condensation.

**Secondary Goal:** Perhaps an additional goal of the project could be to homogenize the material properties of a structure (by using condensation techniques outlined by Flippen [2]) given that more and more aerospace structures now use composite materials with highly discretized material properties across geometries.

# Background

**1.3.1 Modified SIMP Approach**

The design domain is discretized by square finite elements and a “density-based approach to topology optimization” or a “modified solid isotropic material with penalization” is followed; i.e. each element *e* is assigned a density *xe* that determines its Young’s modulus *E*e:

where *E0* is the stiffness of the material, *Emin* is a very small stiffness assigned to void regions in order to prevent the stiffness matrix from becoming singular, and *p* is penalization factor (typically p = 3) introduced to ensure black-and-white solutions. Equation (1) corresponds to the modified SIMP approach, which differs from the classical SIMP approach used in the original paper in the occurrence of the term *Emin*. In the classical SIMP approach, elements with zero stiffness are avoided by imposing a lower limit slightly larger than zero on the densities *xe*. The mathematical formulation of the optimization problem reads as follows:

Minimise:

Subject to:

Where *c* is the compliance, *U* and *F* are the global displacement and force vectors, respectively, *K* is the global stiffness matrix, *ue* is the element displacement vector, *k0* is the element stiffness matrix for an element with unit Young’s modulus, *x* is the vector of design variables (i.e. the element densities), *N* is the number of elements used to discretize the design domain, *V(x)* and *V0* are the material volume and design domain volume, respectively, and *f* is the prescribed volume fraction.

* + 1. **Method of Moving Asymptotes**

Ideally, a method for structural optimization should be flexible and general. It should be able to handle not only element sizes as design variables, but also, for instance, shape variables and material orientation angles. It should also be able to handle ‘all kinds’ of constraints provided only that the derivatives of the constraint functions with respect to the design variables could be calculated (analytically or numerically). Thus, the method should be able to handle general nonlinear programming problems. In addition, it should take into consideration the characteristics of structural optimization problems, e.g. usually very expensive function evaluations but still the possibility to calculate gradients. Further, the method should be ‘stable’ and generate a sequence of improved feasible (or almost feasible) solutions of the considered problem.

The Method of Moving Asymptotes is a mathematical programming method which has been implemented in several large systems for structural optimization (for example in OPTSYS at the Aircraft division of Saab-Scania and in OASIS at ALFGAM Opt. AB). MMA is an iterative method. In each iteration, a convex sub problem which approximates the original problem is generated and solved. An important role in the generation of these sub problems is played by a set of parameters which influence the "curvature" of the approximations and also act as "asymptotes" for the sub problem. By moving these asymptotes, between each iteration, the convergence of the overall process can be stabilized [2].

The MATLAB version of the author's MMA code is based on the assumption that the users optimization problem is written on the following form, where the optimization variables are *x = (x1,…, xn)T*, *y = (y1,…, ym)T* and *z*:

Minimise:

Subject to:

Here, *x1… xn* are the “true” optimization variables, while *y1, …, ym*and z are “artificial” optimization variables; *f0 f1… fm* are given, continuously differentiable, real-valued functions and constitute the constraint functions evaluated at the current values of the variable *x*; *xmin* and *xmax* are given real numbers which satisfy *xminj < xmaxj*; *a0* and *ai* are given real numbers which satisfy *a0 > 0* and *ai ≥ 0*; *ci* and *di* are given real numbers which satisfy *ci ≥ 0*, *di ≥ 0* and *ci + di > 0*.

The flexible nature of the MMA problem description and its MATLAB implementation by Svanberg [2] makes it suitable for structural engineering problems which are riddled with non-linearities in constraint functions. To converge, a simple maximum iteration value or KKT optimality criteria are used.

* + 1. **Static Condensation (Sub-Structuring)**

The process of reducing the degrees of freedom is known as static condensation. The same process is also applied to dynamic problems although, in that case, it is only approximate and in general may result in large errors. The static condensation method has recently been modified for applications to dynamic problems. This method is known as the dynamic condensation method (Paz, M. 1997); its application to dynamic problems gives solution that is virtually exact. [4]

**Physics:** In a beam element in the finite element method, the displacements of neighboring nodes are related by the coefficients of the stiffness matrix. It is thus possible, by some mathematical conditioning, that the displacements or rotations of some of these nodes may be suppressed and represented through other coefficients in the stiffness matrix. Sub structuring requires the reduction of nodal coordinates to allow the independent analysis of portions of the structure (sub structuring). The process of reducing the number of free displacements or degrees of freedom is known as static condensation. [4][5]

A practical method of accomplishing the reduction of the number of degrees of freedom and hence the reduction of the stiffness matrix, is to identify those degrees of freedom to be condensed as secondary degrees of freedom and ‘suppress’ them, and to express them in terms of the remaining primary degrees of freedom. The relationship between secondary and primary degrees of freedom is found by establishing the static relation between them, hence the name Static Condensation Method. This relationship provides the means to reduce the number of unknowns in the system stiffness matrix equation with limited error in the resulting model. [4][5]

* + 1. **Dynamic Condensation**

**Physics:** to understand the fundamentals of dynamic condensation, it is easy to imagine the redistribution and concentration of the mass of any object (say a cantilever beam) to fewer nodes on the body in such a way that the dynamic behavior of the body is largely unchanged and the mass matrix of the object is condensed in size, this reducing computation time in finite element problems**;** because the inertia effects are ignored in static condensation, the accuracy of the resulting reduced model is generally very low for dynamic problems.

To achieve reasonably accurate results, the ‘master nodes’ must be chosen with great care and the number of master nodes should be greater than the number of modes interested. To alleviate the limitations, the inertia effects could be partially or fully included in the condensation. The corresponding condensation approaches are generally called dynamic condensation. We can use exact condensation, classical dynamic condensation, high-order Guyan condensation, dynamic sub structuring scheme, and modal-type condensation to include these effects. [4][5]

* + 1. **Condensation Techniques by Zu – Qing Qu and Zhi - Fangfu**

For the purposes of computation, we can focus our efforts in the project to iterative condensation process for which there exist papers like [6]. According to [Zu-Qing Qu and](https://www.sciencedirect.com/science/article/pii/S0888327098913024" \l "!) Z[hi-Fangfu](https://www.sciencedirect.com/science/article/pii/S0888327098913024#!)’s condensation method Dynamic condensation techniques have been broadly applied to the domains of test-analysis-model correlation, vibration control, damage detection and so on to reduce the structural matrices (stiffness, mass and/or damping matrices) of finite element models. It is iterative. Comparing almost all the iterative schemes for dynamic condensation proposed in the past, the present approach has three advantages: (1) the convergence is much faster than all these methods, especially when the Eigen pairs of the reduced model are close to those of the full model. (2) The convergence of the iterative scheme can be proved simply. (3) It is computationally efficient since it is unnecessary to calculate the stiffness and mass matrices as well as the Eigen solutions of the condensed model in each iteration. Two iterative schemes, which are based on the convergence of the eigenvalues of the reduced model and the column vectors of the dynamic condensation matrix, respectively, are given in this paper. Not only the accuracy of eigenvalues, but also that of eigenvectors is considered in each iteration. Numerical examples are also presented to show the efficiency of the proposed method. [6]

# Numerical Aspects

For the structural engineering optimization problem in consideration, the problem looks as follows:

Min:

Subject to:

Where, from the general MMA optimization problem description in section 1.2.2: *a0 = 1*, *ai = 0*, *di = 0* and *ci = “large number”*. The reason *ci* is large is because we force the variable *yi* to be “expensive” such that generally, *y = 0* for any optimal solution for *fo*.

In many applications, the constraints are on the form *σi(x) ≤ σimax*, where *σi(x)* stands for example, a certain stress, while *σimax* is the largest permitted value on this stress. This means that *fi(x) = σi(x) - σimax*. The user should then preferably scale the constraints in such a way that *1 ≤ σimax ≤ 100* for each *i* (and not *σimax = 1010*). The objective function *f0(x)* should preferably be scaled such that *1 ≤ f0(x) ≤ 100* for reasonable values on the variables. The variables *xj* should preferably be scaled such that *0.1 ≤ xmaxj - xminj ≤ 100*, for all *j*. Concerning the “large numbers” on the coefficients *ci* (mentioned above), the user should for numerical reasons try to avoid “extremely large” values on these coefficients (like 1010). It is better to start with “reasonably large” values and then, if it turns out that not all *yi = 0* in the optimal solution, increase the corresponding values of *ci* by a factor 100 (for example) and solve the problem again, and so on. If the functions and the variables have been scaled according to above, then “reasonably large” values on the parameters *ci* could be, say, *ci* = 1000 or 10000.

# Current Results and Comments

**4.1 Integration of Area and Inertia of the System**

The new formulation of the topology optimization problem is as follows:

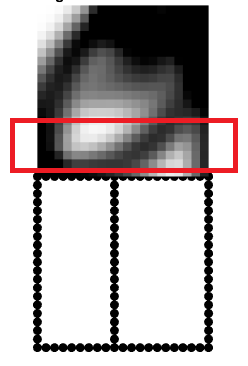
Minimise:

Subject to:

Where *x* is the vector containing the values of the pseudo-densities of the elements in the *x* and *y* directions with values ranging from 0 to 1; is the total of the pseudo densities and is the minimum desired volume reduction, used as a constraint; *U* and *F* are the global displacement and force vectors, respectively, *K* is the global stiffness matrix, *ue* is the element displacement vector, *k0* is the element stiffness matrix for an element with unit Young’s modulus, *x* is the vector of design variables (i.e. the element densities); *N* is the number of elements used to discretize the design domain; *f* is the prescribed volume fraction. We take a few elements of the *x* vector near the beam truss to vary the A and I of the beam truss elements.

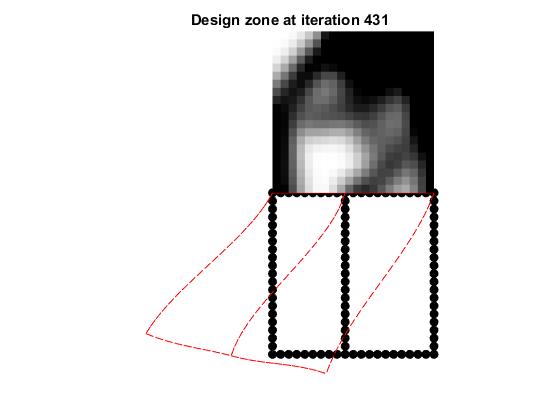
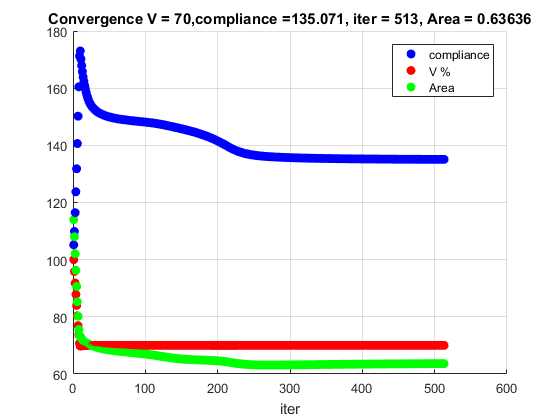
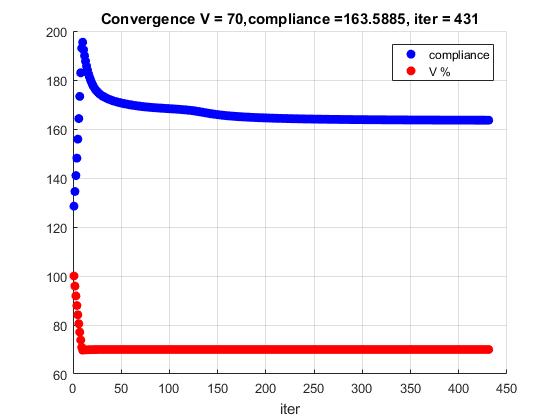
**4.1.1 Implementation and Formulation**

For the results of the optimization without re-sizing the non-design zone’s area of cross section and inertial area moment, the results acquired as seen in Fig. 2(a) and Fig. 2(b) in the final topology and the compliance and volume fraction. By adding additional decision variables in the form of modifying the area of cross section and inertia of the non-design zone and the results changed for the better by resulting in a faster convergence (for larger number of elements) of the optimal topology as can be seen in the Fig. 3 and Fig. 4. The formulation for the dependency of the area on the design zone is:

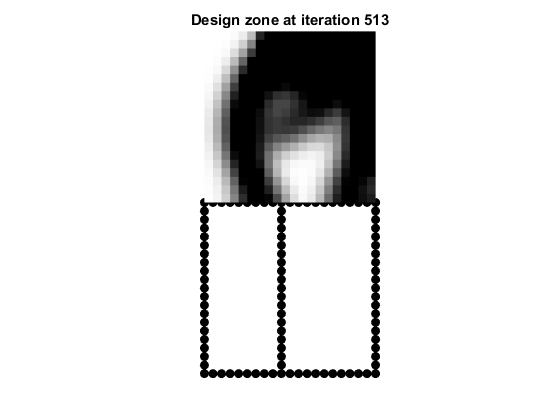


The above equations are used to resize the beam truss in order to simply vary the dimensions of the beam and observe the variation in the topology optimization output. Here, *i* and *j* are indices that correspond to the *x* and *y* coordinate of the elements of the topology; *A* and *I* are the areas and inertias respectively and *x* is the matrix of the current values of the *x* vector in current MMA optimization iteration; *n* is the number of rows taken into account.

*Figure 2: Area in red from where dependency of the area and inertia is acquired where n = nely-5*



*(b)*



*Figure 2: (a) – Topology output for the design zone with nelx = 20; nely = 20 and no area and inertial dependency and showing displacements in red (b) - Compliance and volume fraction reduction versus iteration number (c) - Topology output for the design zone with nelx = 20; nely = 20 and with area and intertial dependency showing changes in topology output (d) – Compliace, volume fraction and area reduction versus iteration number.*

*(d)*

*(c)*

*(a)*

We take a region close to the truss such that the ‘density’ of the elements in that region determines the average area of cross-section and inertia values for the following iteration of the generation of the stiffness matrix. It can be seen between Fig. 2(a) and Fig. 2(c), the topology is quite different when the area of cross-section or thickness of the engine is varied.

*(c)*

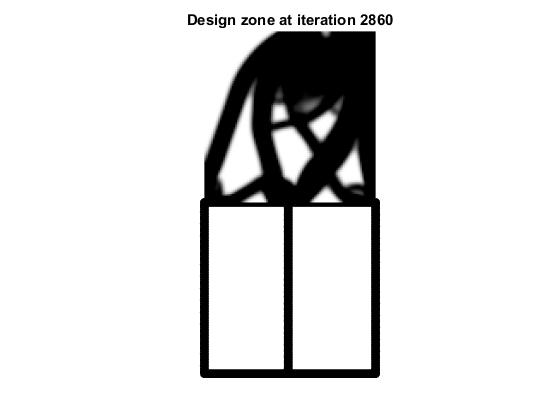
*(d)*

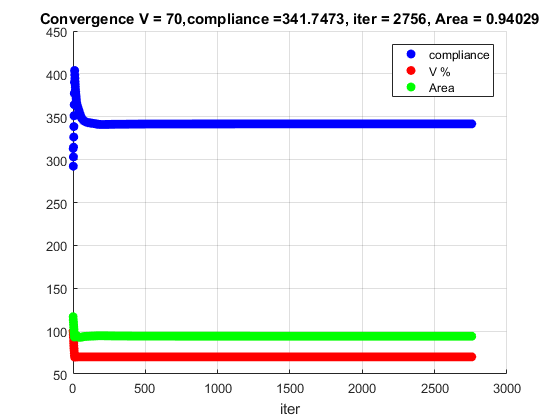
Closer observation of Fig. 2(d) shows a reduction in the area (in green) of the beam truss with the corresponding variation in the compliance. The idea behind this variation in area is to observe a change in the topological output of the design zone when the thickness of the engine architecture changes. Between Fig. 2(a) and Fig. 2(c) we can see a large change in the resultant topology, in that, a beam that has a lower area and inertial moment gives pylon that is pinned onto the engine closer to the middle.

**4.1.2 Parametric Analysis**

In order to further understand the effect of the stiffness of the beam truss on the topology optimization, a classical parametric analysis was carried between a representative ‘stiffness’ of the truss and a representative ‘output’ of the topology optimization process as seen in Fig. 5. In Fig. 3, Fig. 4 and Fig. 5, the outputs are shown for a topology of 10000 elements.

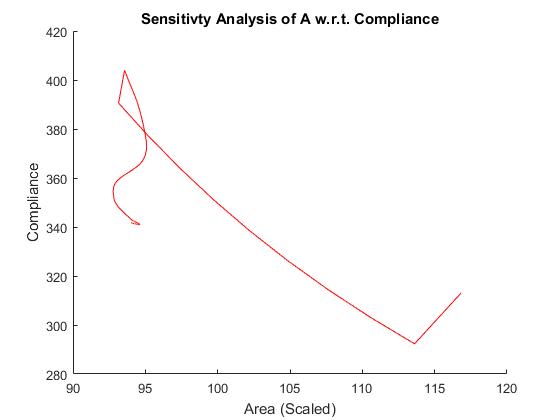
As it can be seen in Fig. 5, the compliance of the system reduces as the area increases which is expect as compliance is defined as a ‘virtual work’ of the system which translates as a force times a displacement and with lower displacements due to higher stiffnesses, the compliance is lower. The initial variations in the graph may be accounted for by the changes in the topology in near the start of the iterative process.





*Figure 4: Compliance variation with 10000 elements and inertia and area dependency*

*Figure 3: Output with 10000 elements*



*Figure 5: Dependency of compliance on area of the beam truss*

**4.2 Static Condensation**

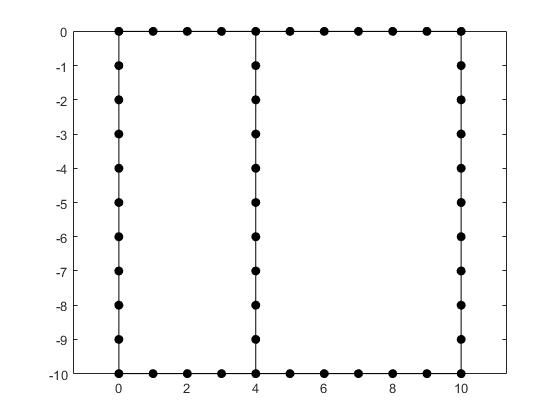
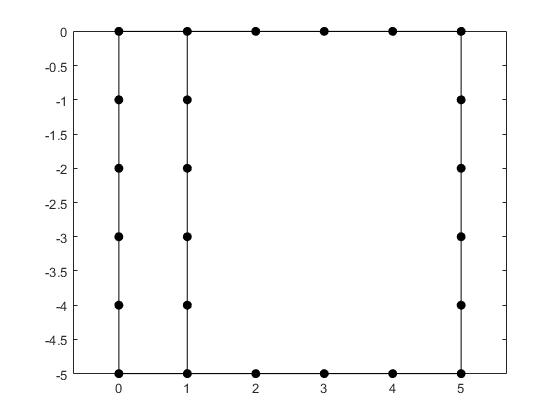
A practical method of accomplishing the reduction of the number of degrees of freedom and hence the reduction of the stiffness matrix, is to identify those degrees of freedom to be condensed as secondary degrees of freedom, and to express them in terms of the remaining primary degrees of freedom. The relationship between secondary and primary degrees of freedom is found by establishing the static relation between them, hence the name Static Condensation Method. This relationship provides the means to reduce the number of unknowns in the system stiffness matrix equation. In order to describe the Static Condensation Method, assume that those secondary degrees of freedom to be reduced or condensed are arranged in the first *‘s’* nodal coordinates and the remaining primary degrees of freedom are the last *‘p’* nodal coordinates. With such an arrangement the stiffness equation for a structure may be written using partitioned matrices as:

where *{us}* is the displacement vector corresponding to the *‘s’* secondary degrees of freedom to be reduced and *{u­p}* is the vector containing the remaining *‘p’* primary degrees of freedom. A simple multiplication of the partitioned system in yields the following two matrix equations:

(2)

By solving for vector {us} and substituting its solution in equation (2) we get a relation that looks something like:

Where *K’* and *F’* are the condensed stiffness and force vectors and *{u}p* is the vector of primary displacement which are in turn related to one another, thus attaining the displacement on all nodes of the system. In the scope of the problem with respect to the pylon, our main objective here would be to reduce the size of the matrix that we would be calling each time we run an optimization loop. This will result in a speed up of the optimization process. Shown below (Fig. 6) is the truss where the larger black dots represent the nodes of the elements within the truss. Here, each element represents a generalized beam with 3 degrees of a freedom at each node (both locally and globally). The current code uses the number of elements in *x* and *y* to determine the number of nodes per member of the truss. The goal is to effectively reduce the number of nodes in the truss members. By relating the displacements of one node to the other, the effective number of nodes in the truss members may be thus reduced and could like Fig. 7 with the possibility of a full recovery of displacements on the ‘suppressed’ DOFs.



*Figure 6: Non-condensed beam truss*

*Figure 7: Condensed beam truss*

# Key Challenges and Questions

Choosing appropriate assumptions for mass distributions across the structure in the condensation model to accurately model physics can be a big challenge. If the mass distribution is not appropriately chosen we can get a large variation in the value of natural frequencies and hence dynamic behavior of structures (20% in some cases [6]).

So, how should we distribute the mass associated with a node to avoid this error?

Given that our problem is one on optimization and not pure condensation, we have to choose our decision variables quite carefully so as to not lose out important regions in the structure to optimize. There mostly exists a trade-off between how well we can condense and how well we can optimize the structure. Which is the minimal number of retained DOFs of beam truss you should keep? Can we improve the classic strategy?

The gain in terms of computational cost is quite unclear and may not be as high as one might think if the code has to condense the model every time we run an optimization. Reduction in time complexity of the code remains a mystery. Is our code reasonably fast?

# Future Work

The steps that follow for the completion of this project are:

1. Literature review on static and dynamic condensation processes (contd.).
2. Implementation of local variables (such as beam section shape) in a sensitivity analysis.
3. Choice of condensation modeling techniques.
4. Comparative study of techniques.
5. Implementation of working code on baseline test case.
6. Testing of code on sample cases

# APPENDIX: Current MATLAB Implementation

The MATLAB code for optimization used to simulate our problem is based on the top88.m [7] format for topology optimization with all the variables named accordingly. The change is in the optimizer which is now the MMA optimizer and the new KKT optimality criteria. We initialize the problem by calling the following function:

>>top88\_plustruss\_mma\_nocond\_internship(nelx,nely,volfrac,penal,rmin,ft)

Where nelx is the number of elements in the x-direction; nely is the number of elements in the y-direction; volfrac is the desired reduction in the volume or volume fraction; penal is the penalty in the modified SIMP algorithm; rmin is the filter radius and ft, the additional argument, specifies whether sensitivity filtering (ft = 1) or density filtering (ft = 2) should be used.

For the non-design zone, the stiffness matrix and the corresponding force and displacement vectors is obtained by calling the following function:

>>truss\_stiffness\_no\_condensation(nelx,nely,E,A,I)

Where nelx is the number of elements in the x-direction; nely is the number of elements in the y-direction; E is the elemental stiffness; A is the area of cross-section; I is the area moment of inertia. This function uses the genrealised beam stiffness matrix with 3 degrees of degree per node locally and globally such that each element has 2 axial displacements, 2 vertical displacements and 2 in-plane rotations. This function returns the following vector:

[Kcc,Kce,Kee,Fe,Fc,Recovery\_matrixc,Recovery\_matrixe]

Where Kcc, Kee and Kce are sub-matrices used to create the ‘projection matrix’ that is used to affect a new stiffness on the design zone at the interface elements.

For the optimizer i.e. MMA, the following function is called in which all variables follow the exact nomenculture mentioned in section 1.2.2 and it prepares the the bounds and other necessary conditions to solve the convex sub-problem:

>>mmasub(m,n,outeriter,xval,xmin,xmax,… xold1,xold2,f0val,df0dx,fval,dfdx,low,upp,a0,a,C,d);

Where m is the number of constraints; n in the number of decision variables which in our case is the number of elements; outriter is the iteration number; xval is the current value of the *x* vector; xold1 and xold2 are storage variables used to pass the *x* vector values from 1 and 2 iterations before respectively; f0val is the current objective function value, df0dx is the current value of the derivative of the objective function with respect to the decision variables; fval is the value of the constraint functions at the present *x* vector value; dfdx is the the value of the derivative of the constraint functions with respect to the decision variables; low is the lower asymptote value from the previous iteration; up is the upper asymptote value from the previous iteration, a is *ai*; c is *ci* and d is *di* as in section 1.2.2 which then returns the following vector:

[y,ymma,zmma,lam,xsi,eta,mu,zet,S,low,upp]

Where the new varibles: ymma and zmma are the *y* and *z* values respectively; lam, xsi, eta, mu and zet are Lagrange multipliers; S is constraint slack variables. These will be used for KKT stopping criteria. For the actual solution of the MMA convex subproblem the following function is called within the mmasub function:

>>subsolv(m,n,epsimin,low,upp,alfa,beta,p0,q0,P,Q,a0,a,b,c,d);

Where the new variables: epsimin is a very small number to avoid matrix singularities; alfa and beta are ‘move limits’ to avoid the possibility of a division by zero while solving the subproblem; p0, q0, P and Q are the values calculated in mmasub as specified by Svanberg in [2] and calculated using the differential of the constraint functions . This function returns the vector:

[xmma,ymma,zmma,lam,xsi,eta,mu,zet,s]

This is in turn returned by the mmasub function to the main script.

For the stopping optimiality criteria we use the KKT conditions [2] by calling the following the following function:

>>kktcheck(m,n,x,y,z,lam,xsi,eta,mu,zet,s,xmin,xmax,df0dx,fval,dfdx,a0,a,c,d);

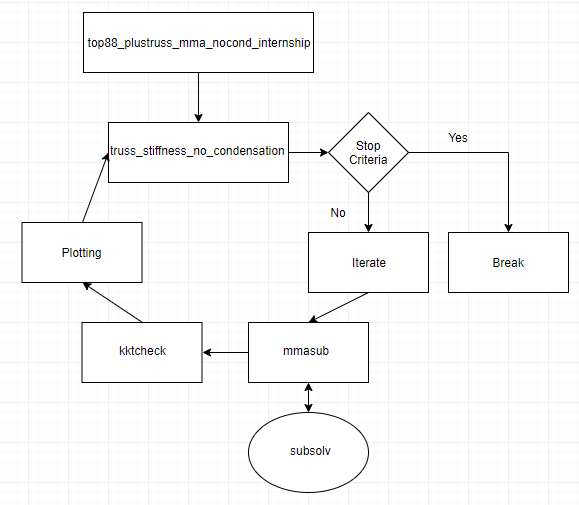
With the input variables as we have already seen with the other functions and it returns the following vector:

[residu,residunorm,residumax]

Where residunorm = kktnorm is used as a stopping criterion along with others as seen below:

while kktnorm > kkttol && outit < maxoutit && change>0.001

Where outit is the current iteration value whereas maxoutit is the maximum persmissible number of iterations and where change is the difference in the *x* vector values between two iterations. The way in which the code works is demonstrated in a clearer way by the chart in Fig. A1.



*Figure A1: Function call flowchart for MATLAB Implementation*

# References

[1]William Anderson. (2007). *Dynamic Reduction Methods. Lecture 12.* [Online Video]. 13 September 2013. Available from: <https://www.youtube.com/watch?v=wrV6GyXVu2M>. [Accessed: 25 March 2018].

[2] Svanberg K. (1987), The Method of Moving Asymptotes – A New Method for Structural Optimization, *International Journal for Numerical Methods in Engineering,* Vol. 24 (No. 2), pp. 359 -373, <https://doi.org/10.1002/nme.1620240207>

[3] L.D. Flippen Jr, (1994), A Theory of Condensation Model Reduction, *Computers Math. Application, Vol. 27* (No. 2), pp. 9 - 40.

[4] Paz M., Leigh W. (2001*); Static Condensation and Substructuring*. In: Integrated Matrix Analysis of Structures. Springer, Boston, MA

[5] Zu-Qing Qu, (2004), *Model Reduction Techniques*, London: Springer-Verlag.

[6] [Zu-Qing Qu, Zhi-Fangfu](https://www.sciencedirect.com/science/article/pii/S0888327098913024#!), (2000), An Iterative Method For Dynamic Condensation Of Structural Matrices, [*Mechanical Systems and Signal Processing*](https://www.sciencedirect.com/science/journal/08883270), Vol. 14 (No. 4), pp. 667 - 678.

[7] Andreassen, E., Clausen, A., Schevenels, M. et al. *Struct Multidisc Optim* (2011) 43: 1. <https://doi.org/10.1007/s00158-010-0594-7>

[8] Zhu, J.H., Zhang, WH. & Xia, L. *Arch Computat Methods Eng* (2016) 23: 595. <https://doi.org/10.1007/s11831-015-9151-2>

[9] Deaton, J.D. & Grandhi, R.V. *Struct Multidisc Optim* (2014) 49: 1. <https://doi.org/10.1007/s00158-013-0956-z>