Dynamic Condensation

Author: Oriol Falip
Advisor: Ariel Eijo
Date: 27/07/2020

Index

1.Introduction

Stampack GmbH develops advanced and efficient simulation software for all sheet metal forming processes. It is suitable for tool designers and method planners to simulate body in white parts, trim panels, covers and trim strips in vehicle interiors, connecting components and support plates, metal containers, heat shields and aluminium foils etc.

This wide range of applications in the industry demand highly accurate and also short computation time. When dealing with complex problems, in order to obtain accurate solutions, a fine mesh with really small elements is required. Since Stampack uses an explicit method to solve the equations, time increment (*dt)* is constrained by the element size because explicit methods are unstable for time increments greater than a certain critical time. Then, having small *dt*'s makes the simulation really time consuming.

To overcome this situation, different methods can be used, for instance, selective mass scaling which artificially increases mass for the smallest elements so as to stabilise the solution for higher dt, or Dynamic Condensation (DC) which is a Model Order Reduction method that makes use of a coarser mesh with bigger elements to obtain a solution and then, the solution is projected to the actual fine mesh. This allows to use the critical time from the coarser mesh to obtain the solution in refined mesh.

In this work, Dynamic Condensation method will be implemented and its results analysed to study how this method performs to increase the critical time.

2. Objectives

The main objective of this work is the implementation of the Dynamic Condensation (DC) method in Stampack.

To achieve the goal, two main tasks are required: first, to verify that the auxiliary algorithms needed to implement Dynamic Condensation (DC) method work properly and all information required by DC is properly set. And secondly, to test the DC method itself in order to increase the critical time of a given problem mesh.

3. Dynamic Condensation

Dynamic Condensation is a Model Order Reduction method [2] which allows to solve a problem with a higher critical time than the actual critical time of the mesh. Defining the original mesh as the **Fine mesh** and a coarser mesh as the **Coarse mesh**, all nodes can be classified as:

Reduced nodes (or coarse nodes): Defined as all nodes belonging to the Coarse mesh

Condensed nodes (or fine nodes): additional nodes from the Fine mesh which are not in the Coarse mesh

With the previous definitions, Dynamic Condensation method can be regarded as a two step problem where first a Reduced problem is solved in the coarse nodes and then, the Global solution is obtained by solving in the condensed nodes. Those two problems are stated as follows [3]:

1. Reduced problem:

First, forces from the condensed nodes are transferred to the coarse nodes by using the shape functions of the coarse elements:

$$\sum_n \Phi_m(n) F_n \qquad \textbf{eq.(1)}$$

Then, the solution for the reduced problem comes from solving the following equation:

$$\overline{M}_m \ddot{u}_m = F_m + \sum_n \Phi_m(n) F_n \qquad \textbf{eq. (2)}$$

Where M_m is the lumped mass at the reduced nodes computed by considering the coarse mesh and F_m is the forces at coarse nodes.

2. Global problem

Average acceleration is computed by transferring the accelerations obtained in the Reduced problem from the coarse nodes to condensed nodes as follows:

$$\ddot{\bar{u}}_n = \sum_m \Phi_m(n) \ddot{u}_m \qquad \textbf{eq. (3)}$$

then, the solution comes from solving:

$$\overline{M}_n \left(\ddot{u}_n - \ddot{\bar{u}}_n \right) = F_n \qquad \textbf{eq.(4)}$$

where lumped mass M_n at condensed nodes must be set considering the target critical time.

4. Implementation

The most challenging part on the implementation of Dynamic Condensation method is not the method itself but the indirect tasks required by this method. Those tasks are, for instance, mesh coarsening, transferring the information from node to node, the identification of the set of reduced and condensed nodes, etc.

For general problems, all those tasks must be done by complex algorithms in order to complete them successfully and in an efficient way [1]. Since it is beyond the scope of this study, the implementation was limited to simple problems where **regular-structured rectangular meshes are used**.

In this section, giving that those auxiliary tasks are critical to ensure that DC method is correctly implemented, first of all it will be verified that those indirect tasks work properly and finally, some details on the implementation of the Dynamic Condensation itself will be mentioned.

4.1 Auxiliary tasks

The two main task that must be verified before applying DC method considered to be:

Mesh Coarsening: From a fine mesh, a Coarse mesh is generated by building its connectivity matrix (Tcoarse) and a list of reduced node's ID and coarse element ID where it belongs (reducedNodesList).

Information transfer: List identifying which reduced node sends to each coarse node and its weight.

After the implementation of those tasks, several tests have been done to ensure its correct implementation. Results are shown as follows:

4.1.1 Mesh Coarsening

The following test shows the results obtained after applying the coarsening algorithm for different original meshes and coarsening factor H (relation between coarse and original element size).

The following figures show the obtained meshes with the coarsening algorithm:

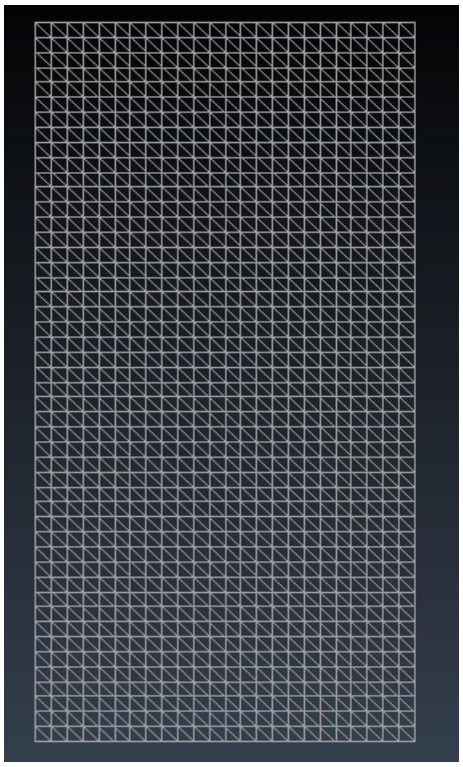

Figura 1: Original 24x48 mesh.

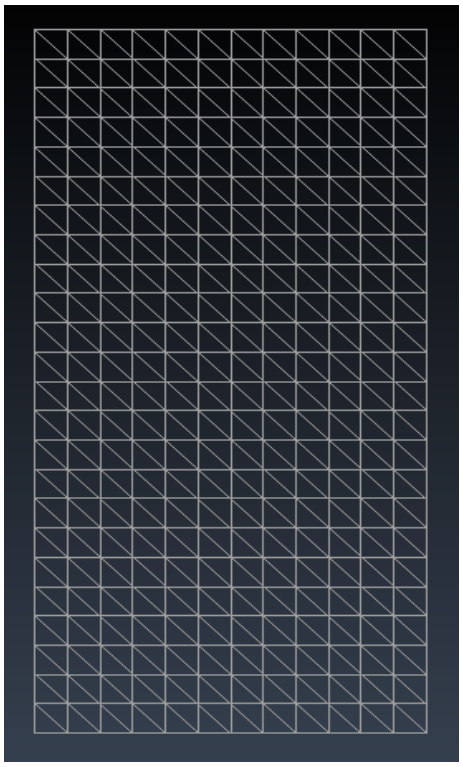

Figura 2: Coarser 12x24 mesh, H=2.

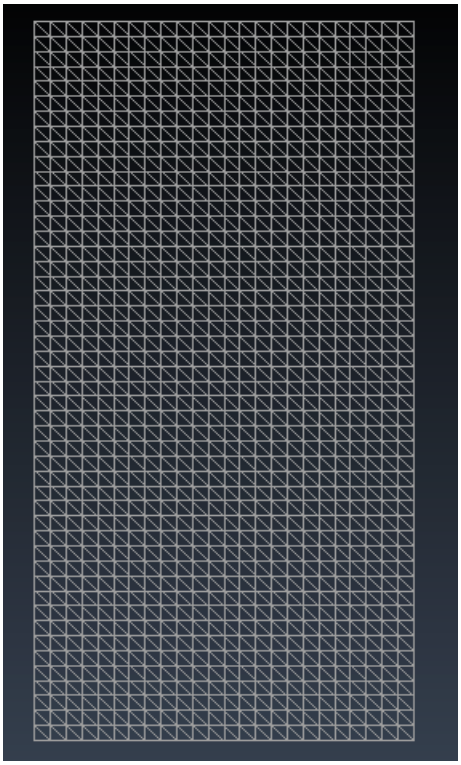

Figura 3: Original 24x48 mesh

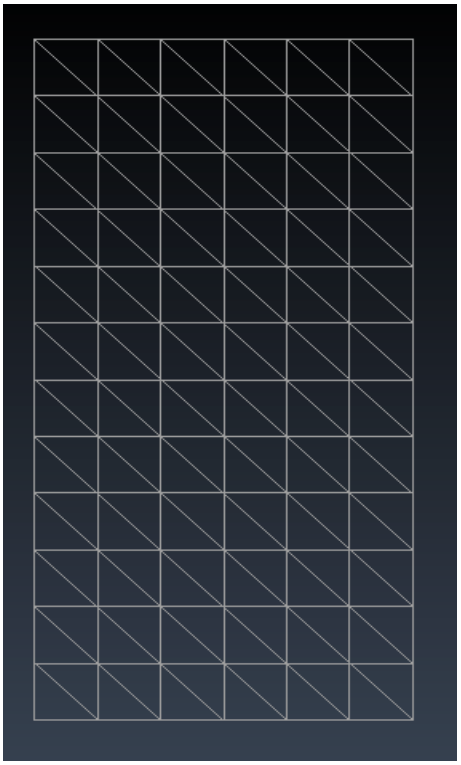

Figura 4: Coarser 6x12 mesh, H=4.

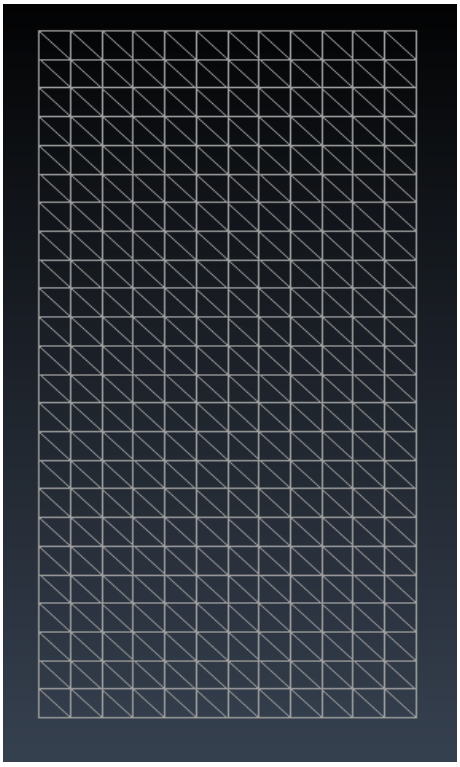

Figura 5: Original 12x24 mesh.

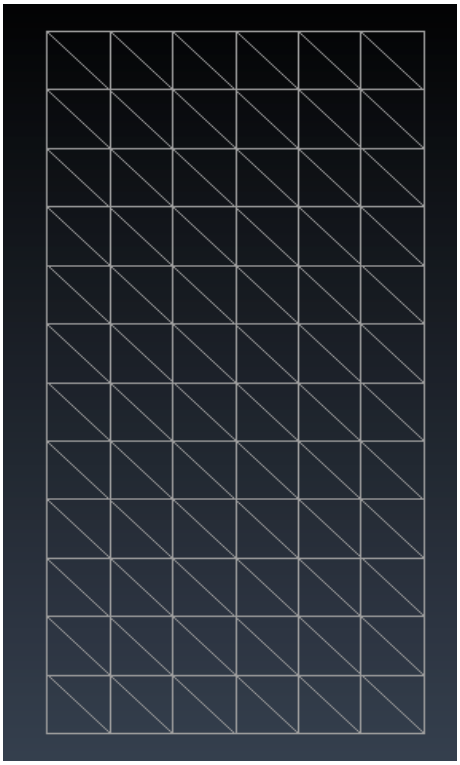

Figura 6: Coarser 6x12 mesh, H=2.

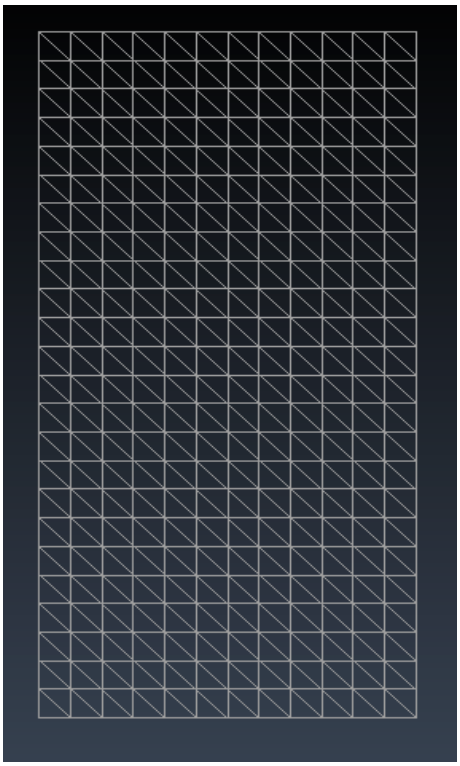

Figura 7: Original 12x24 mesh.

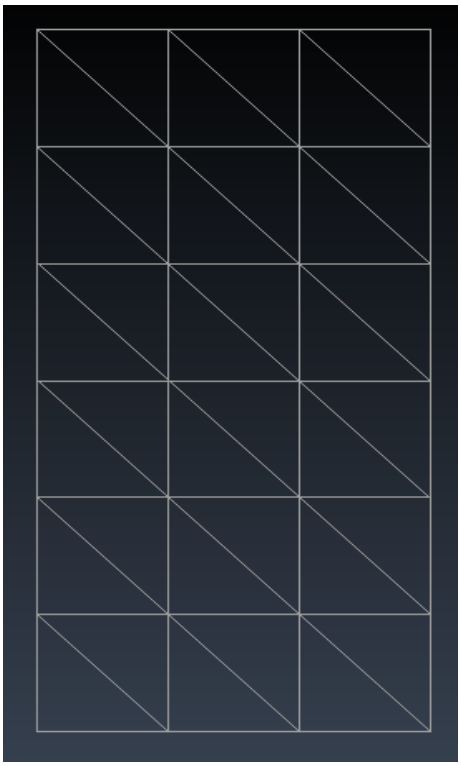

Figura 8: Coarser 3x6 mesh, H=4.

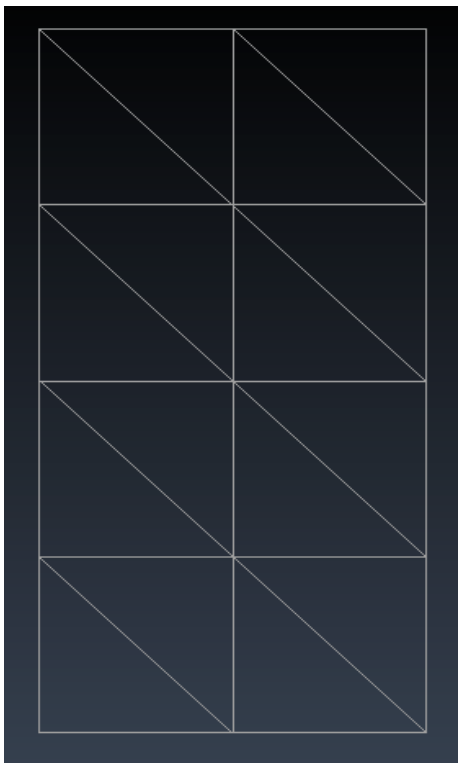

Figura 9: Original 2x4 mesh.

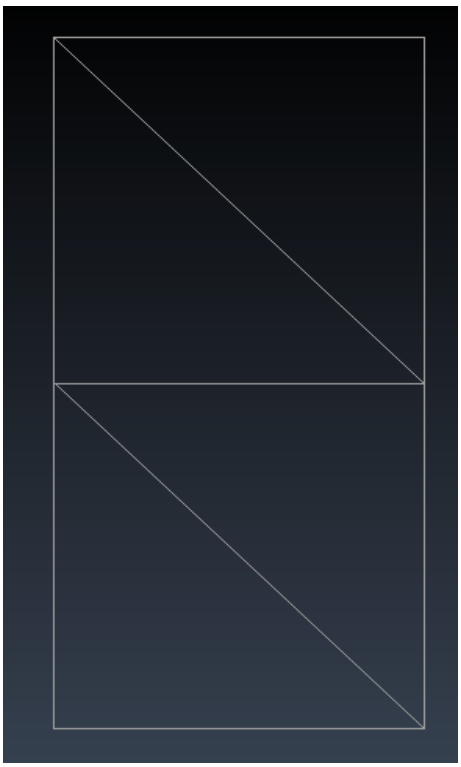

Figura 10: Coarser 1x2 mesh, H=2.

In figures 1-10, it can be seen that coarsening algorithm works properly for any initial mesh and any coarsening factor (here H = 2 and H = 4 although other factors have been successfully tested).

Finally, to check the correct identification of reduced nodes, each node in *reducedNodesList* has been plotted overlapped with the coarse mesh. Each white dot corresponds to a reduced node.

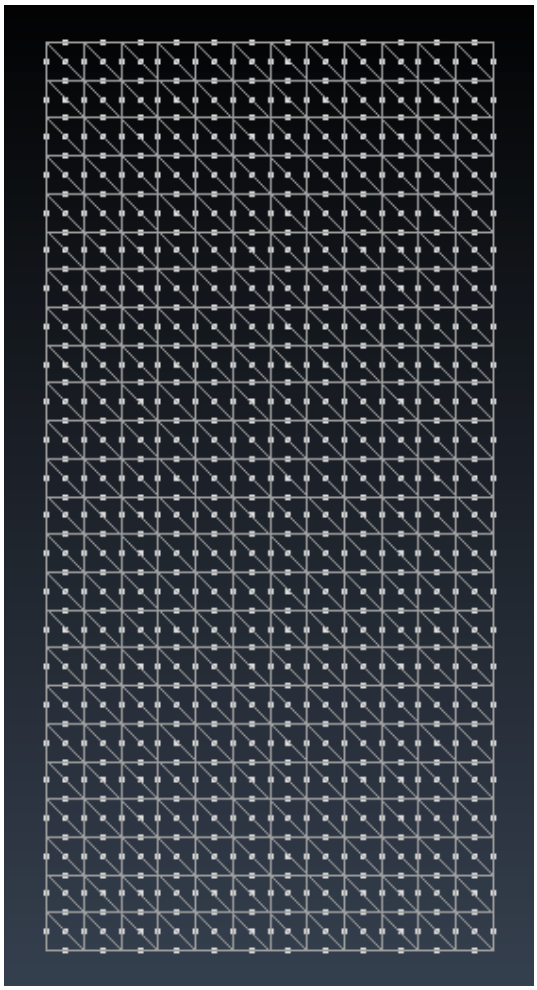

Figura 11: Original mesh of 24x48 - Coarser 12x24 mesh, H=2, with reduced nodes overlapped (white dots).

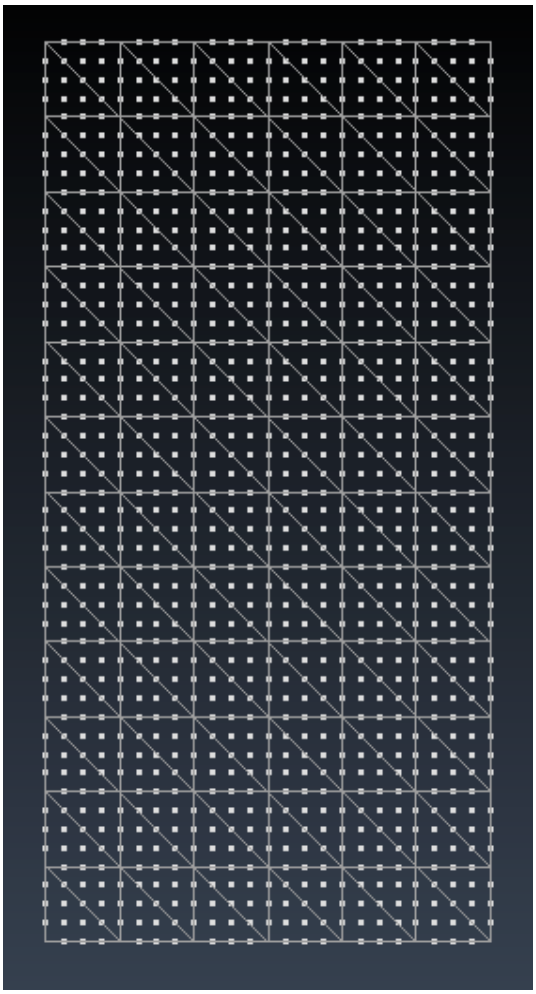

Figura 12: Original mesh of 24x48 - Coarser 6x12 mesh, H=4, with reduced nodes overlapped (white dots).

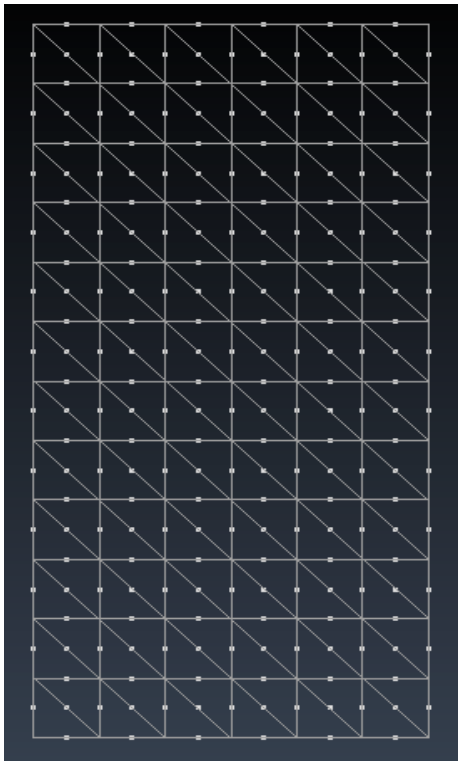

Figura 13: Original mesh of 12x24 - Coarser 6x12 mesh, H=2, with overlapped condensed nodes (white dots).

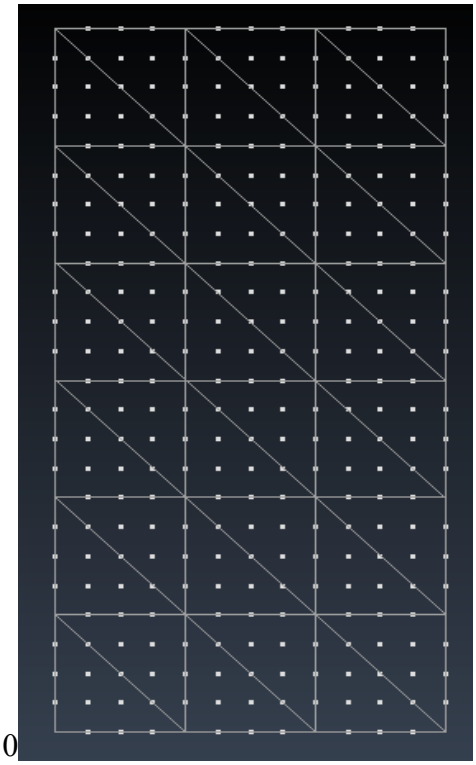

Figura 14: Original mesh of 12x24 - Coarser 3x6 mesh, H=4, with overlapped condensed nodes (white dots).

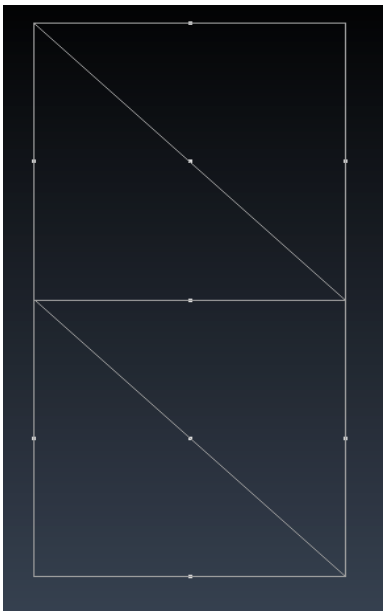

Figura 15: Original mesh of 2x4 - Coarser 1x2 mesh, H=2, with overlapped condensed nodes (white dots).

In figures 11-15, it can be checked that all reduced nodes are correctly identified and added to the *reducedNodesList*.

4.1.2. Information transfer

Information transfer algorithm takes as starting point the information obtained at the coarsening mesh algorithm. Having checked that the first task is properly implemented, results for the information transfer algorithm were verified by checking: Weight values and Node to node transfer.

4.1.2.a. Weight values

For H=2, due to the position of reduced nodes, all weights must be w = 0,5 or w = 0.
Writing all weights, it is possible to visually check that this condition is fulfilled.

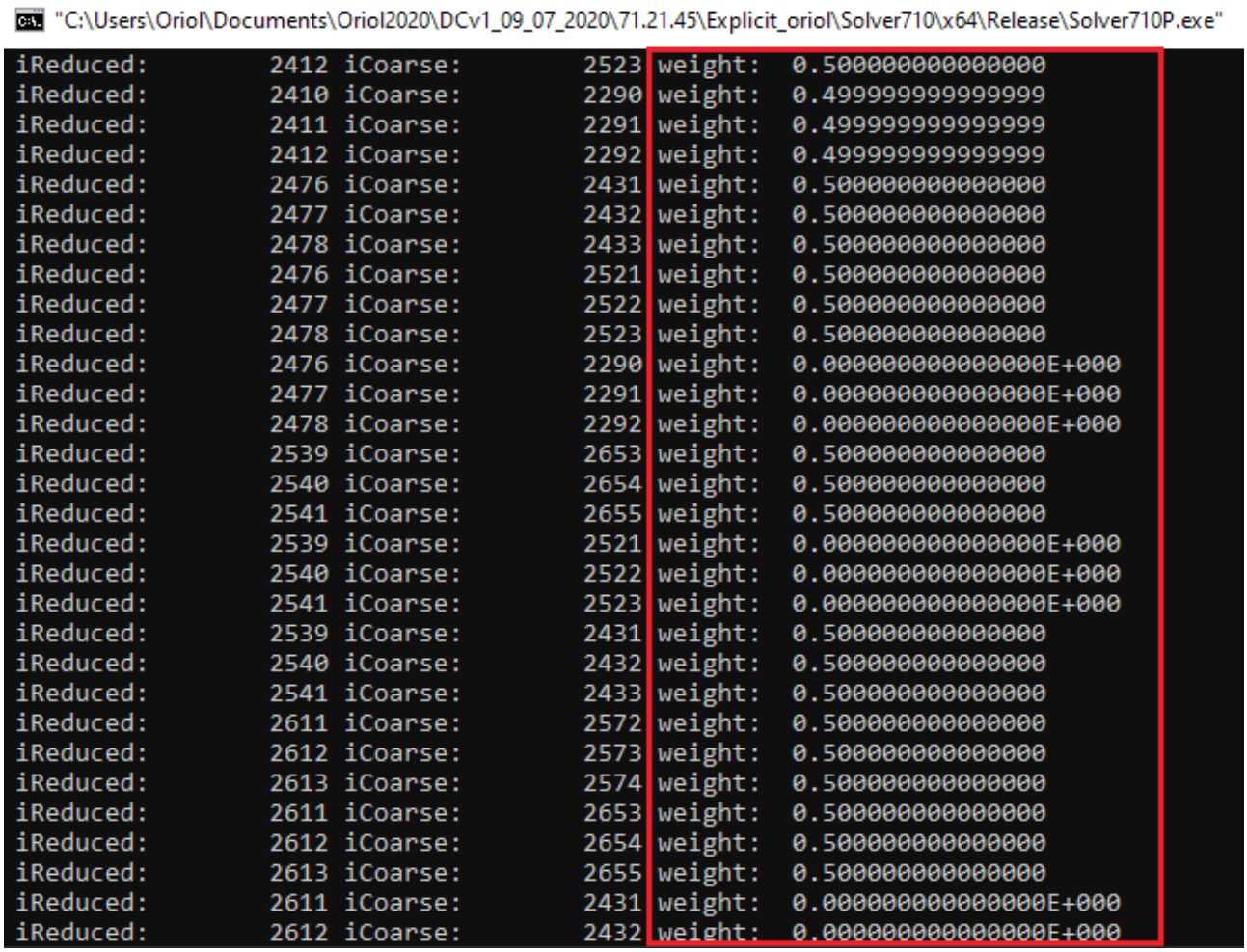

Figura 16: Console screenshot showing equation IDs and weights.

For H = 4, all weights must be w = 0,25, w = 0,50, w = 0,75 or w = 0.

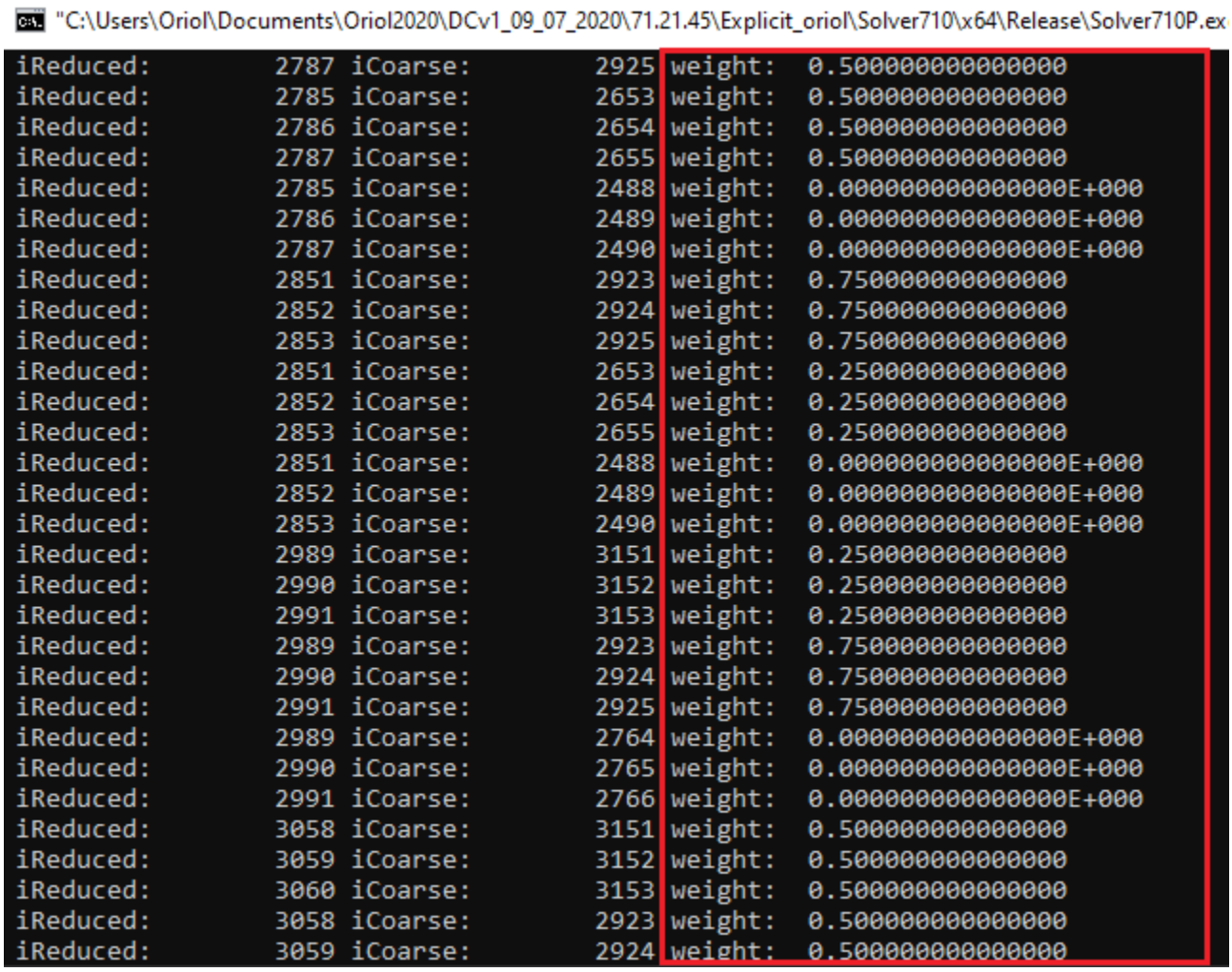

Figura 17: Console screenshot showing equation IDs and weights.

Regarding weights too, adding them all, it has been seen that the results is equal to 3* # reduced nodes, being 3 equal to number of degrees of freedom. This condition must be verified because if it wasn't, that would mean that information is overweight if weight per node is > 1 or under considered if weight per node is < 1.

4.1.2.b. Node to node transfer

For small meshes, list of which reduced nodes sends to each coarse node has been verified manually by writhing the whole *Vinfo* list. For bigger meshes, this task is impossible, so two auxiliary subroutines were used:

checkInfo: taking as input a coarse node ID, writes to a ".msh" file the connectivity matrix for all coarse elements this node belongs to and ID+coordinates of all reduced nodes sending information to that coarse node.

ReducedSendsTo: taking as an input a reduced node ID, writes all coarse node it sends information and its weight.

CoarseSendsTo: taking as an input a coarse node ID, writes all reduced nodes its sends information and its weight or, what is the same, all reduced nodes it receives information from.

With the previous subroutines, it is easier to check for a given node if the information is being send properly.

Below are showing results for a random coarse node ID = 525 for a coarsening facto H of 2 and 4:

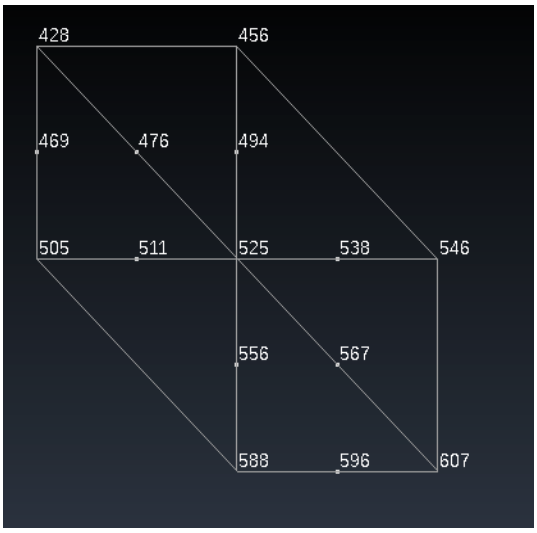

Figura 19: H=2 - Results obtained with checkInfo for node ID=525.

```
    Results have been reported for postprocess
 Reduced Node:         556 sends to Coarse Node:   588.000000000000
  with weight  0.500000000000000
 Reduced Node:         556 sends to Coarse Node:   607.000000000000
  with weight  0.000000000000000E+000
 Reduced Node:         556 sends to Coarse Node:   525.000000000000
  with weight  0.500000000000000

 Coarse:         525 recives from reduced Node:   469.000000000000
  with weight  0.000000000000000E+000
 Coarse:         525 recives from reduced Node:   476.000000000000
  with weight  0.500000000000001
 Coarse:         525 recives from reduced Node:   511.000000000000
  with weight  0.500000000000001
 Coarse:         525 recives from reduced Node:   494.000000000000
  with weight  0.500000000000000
 Coarse:         525 recives from reduced Node:   556.000000000000
  with weight  0.500000000000000
 Coarse:         525 recives from reduced Node:   538.000000000000
  with weight  0.500000000000000
 Coarse:         525 recives from reduced Node:   567.000000000000
  with weight  0.500000000000000
 Coarse:         525 recives from reduced Node:   596.000000000000
  with weight  0.000000000000000E+000
  ieq to node         472
Fortran Pause - Enter command<CR> or <CR> to continue.
```

Figura 18: H=2 - Results obtained with ReducedSendsTo and CoarseSendsTo subroutines for node ID=525.

Figura 20: H=4 - Results obtained with checkInfo for node ID=525.

```
Coarse:         525 recives from:         415 w  0.000000000000000E+000
Coarse:         525 recives from:         453 w  0.000000000000000E+000
Coarse:         525 recives from:         389 w  0.250000000000000
Coarse:         525 recives from:         423 w  0.250000000000000
Coarse:         525 recives from:         461 w  0.250000000000000
Coarse:         525 recives from:         497 w  0.250000000000000
Coarse:         525 recives from:         529 w  0.000000000000000E+000
Coarse:         525 recives from:         393 w  0.250000000000000
Coarse:         525 recives from:         428 w  0.500000000000000
Coarse:         525 recives from:         469 w  0.500000000000000
Coarse:         525 recives from:         505 w  0.500000000000000
Coarse:         525 recives from:         540 w  0.250000000000000
Coarse:         525 recives from:         570 w  0.000000000000000E+000
Coarse:         525 recives from:         409 w  0.250000000000000
Coarse:         525 recives from:         443 w  0.500000000000000
Coarse:         525 recives from:         476 w  0.750000000000000
Coarse:         525 recives from:         511 w  0.750000000000000
Coarse:         525 recives from:         545 w  0.500000000000000
Coarse:         525 recives from:         578 w  0.250000000000000
Coarse:         525 recives from:         608 w  0.000000000000000E+000
Coarse:         525 recives from:         419 w  0.250000000000000
Coarse:         525 recives from:         456 w  0.500000000000000
Coarse:         525 recives from:         494 w  0.750000000000000
Coarse:         525 recives from:         556 w  0.750000000000000
Coarse:         525 recives from:         588 w  0.500000000000000
Coarse:         525 recives from:         614 w  0.250000000000000
Coarse:         525 recives from:         466 w  0.250000000000000
Coarse:         525 recives from:         502 w  0.500000000000000
Coarse:         525 recives from:         538 w  0.750000000000000
Coarse:         525 recives from:         567 w  0.750000000000000
```

Figura 21: H=4 - Results obtained with CoarseSendsTo subroutine for node ID=525.

It must be remarked that *checkInfo* subroutine uses information obtained from task 1: **mesh coarsening** since it uses *listReducedNodes* while *reducedSendsTo* and *coarseSendsTo* uses information from *Vinfo*, obtained in task 2: **information transfer**. Then, using this tools we can

easily check if the information is properly sent for a given coarse node and, at the same time, prove that both tasks are properly implemented.

4.1.3. Other considerations

The correct implementation would have been to create a new set of elements for the coarse mesh, that would allow to compute coarse mass for the coarse nodes and also to determine the critical time of the coarse mesh. However, since the problem to be solved is really simple, it was easier to introduce those values manually on the code, having to change coarse mass and critical time accordingly to which original mesh and coarsening parameter is being used.

4.2. Dynamic Condensation

Two different versions on Dynamic Condensation are implemented in this study: the first one presents a slightly modification on the procedure proposed by the method and the second one following exactly all steps accordingly to the method.

4.2.1. Dynamic Condensation v1

In the Dynamic Condensation method, the acceleration for the coarse nodes is directly the solution obtained in the Reduced problem. Let's call this acceleration "**aclrcoarse"**. Regarding the acceleration on condensed nodes, its accelerations can be understood as "**aclrcoarse**" transferred from coarse nodes to fine nodes (average acceleration) plus the acceleration one would compute normally if no DC method was applied but with a modified lumped mass.

In this version, the solution for the final acceleration **"aclr"** was obtained as follows:

Step 1: Compute acceleration normally (**aclr**) as if no DC was considered.
Step 2: Compute coarse acceleration (**aclrcoarse**) in all coarse nodes accordingly to the method
Step 3: Transfer **aclrcoarse** from coarse nodes to fine nodes
Step 4: Compute final acceleration according to the following rules:

For coarse nodes: **aclr = aclrcoarse**
For condensed nodes: **aclr = factor * aclr + aclrcoarse**

It must be noticed that in this version, the lumped mass matrix for reduced nodes M_n is used to compute the normal acceleration, while in the method this mass must be set big enough according to the target critical time. This role is played by **factor** which multiplies **aclr**, since $\mathbf{aclr = F_n / M_n}$ **factor** can be understood as a factor which modifies the mass matrix accordingly to the target critical time.

4.2.2. Dynamic Condensation v2

In this version, a modification was introduced in the method. Here, the steps followed are exactly the same fro step 1,2 and 3, but step 4 now reads:

Step 4: Compute final acceleration according to the following rules:

For all nodes: **aclr = factor * aclr + (1-factor) * aclrcoarse**

5. Results

The example chosen to check results obtained with Dynamic Condensation is chosen taking into consideration several things.

First of all, the implementation limitations imposed by the mesh coarsening algorithm forces this example to have a specific geometry, in this case, a rectangular Blank.

Secondly, the physical problem must be simple enough so as its solution is intuitive and easy to analyse. By doing this, if the solution provided by Dynamic Condensation method is wrong, it is much easier to intuitively identify the anomalous behaviour and fix the problem which causes it.

Finally, the example must not include special features such as autoRefinement, remeshing, blank cuts, etc. since it could interfere with DC implementation and in case there are some error, it would be harder to debug.

With the previous considerations, the example chosen to check results obtained with Dynamic Condensation is a simple example (without pad) of a rectangular blank under bending produced by a punch acting in the z-direction. *Figure 22.* shows its geometry and the final configuration for the tools.

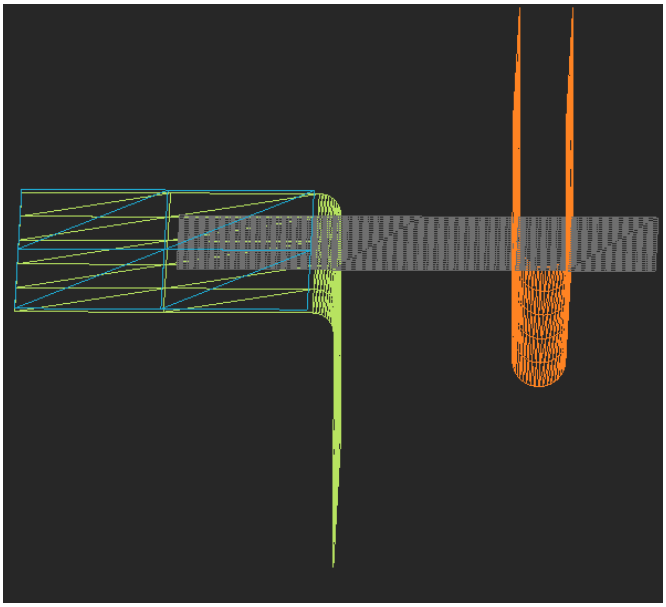

Figura 22: Blank, punch and pad's geometry for the pure bending problem and final configuration for the tools.

The verification of the results is done by means of a reference solution obtained without Dynamic Condensation (*Figure 23-24*). A visual inspection is done to the dynamics and final stage deformation. Additionally, a node from the right corner is chosen to obtain the numerical values for the displacements at this node.

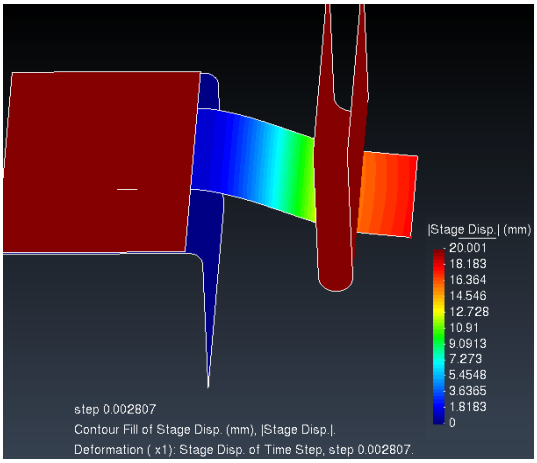

Figura 23: Reference solution computed by the solver in Visual Studio.

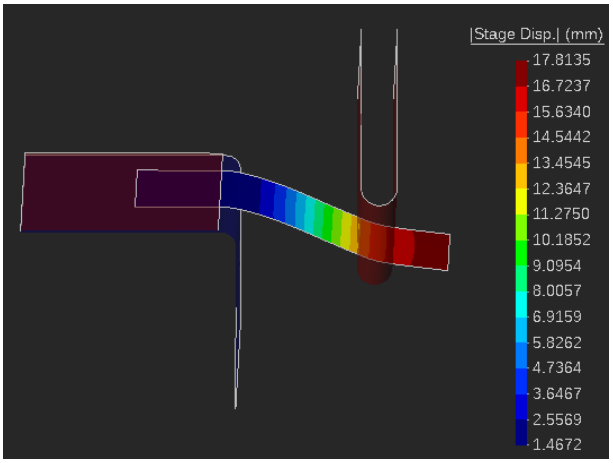

Figura 24: Reference solution computed by Stampack.

The following tables show the numerical value for displacements on the node in the right corner obtained without dynamic condensation (Reference solution in *Table 1*.) and with DC for different coarsening parameter H and corresponding critical time.

	x	y	z	|r|
Stampack	-0,00309	-1,41962	-17,75684	17,81350
Visual Studio	-0,00243	-1,40568	-17,74860	17,80417

Tabla 1: Reference solution obtained with Stampack and Visual Studio Solver.

factor	x	y	z	|r|
0,6	-0,00363	-1,06787	-15,70636	15,74262
0,3	-0,00495	-1,06530	-15,60920	15,64551
0,25	-0,00716	-1,06342	-15,62206	15,65822
0,2	-0,00602	-1,06982	-15,66716	15,70365
0,1	-0,00233	-1,10913	-15,70800	15,74711
0	-0,10006	-1,17605	-17,43010	17,47002

Tabla 2: Displacements and total displacement for different factors when H = 2, dt factor = 1,8 and method version v = 1.

factor	x	y	z	|r|
0,7	divergence			
0,6	divergence			
0,55	divergence			
0,54	divergence			
0,53	0,03615	-1,36832	-17,34583	17,39976
0,52	0,00833	-1,38976	-17,35148	17,40705
0,5	-0,00311	-1,37634	-17,08668	17,14202
0,3	-0,00248	-1,32353	-16,85565	16,90754
0,2	-0,00499	-1,22257	-16,63520	16,68007
0,1	-0,00320	-1,19000	-16,26270	16,30618
0,05	0,00058	-1,18223	-16,18548	16,22860
0	-0,09930	-1,17677	-17,42838	17,46834

Tabla 3: Displacements and total displacement for different factors when H = 2, dt factor = 1,8 and method version v = 2.

factor	x	y	z	|r|
0,2	div			
0,15	div			
0,1	-0,01226	-0,98003	-13,67838	13,71345
0,08	-0,01286	-0,89930	-13,67915	13,70868
0,0625	-0,01134	-0,91040	-13,69292	13,72316
0,01	-0,02634	-0,94746	-13,62023	13,65317
0	-0,03974	-0,70406	-16,72078	16,73564

Tabla 4: Displacements and total displacement for different factors when H = 4, dt factor = 3,8 and method version v = 1.

factor	x	y	z	|r|
0,4	problems detected			
0,2	divergence			
0,15	divergence			
0,12	0,00536	-1,00933	-14,65558	14,69030
0,1	-0,00012	-0,99610	-15,10326	15,13607
0,05	-0,01653	-0,91862	-14,40756	14,43682

Tabla 5: Displacements and total displacement for different factors when H = 4, dt factor = 3,8 and method version v = 2.

When the coarsening factor H = 2, i.e. coarse elements are twice bigger than the original ones, and the critical time has been almost doubled, it can be seen that DC v2 performed well once a certain value for the factor which weights coarse and fine accelerations is achieved. However, the v1 failed to obtain a good numerical solution for the displacements on the corner node with the exception of a factor = 0, which corresponds to only the solution on the coarse mesh linearly extrapolated to the fine nodes.

Regarding a coarsening parameter H = 4 to target a four times bigger critical time, no good solution was found for any of the versions implemented and for any value of the factor. The reason for this

to happen is because when mass is added, over damping effects change the dynamics of the problem, therefore, the solution obtained is visually good, but with different dynamics.

6. Damping

Results from the previous section showed that the more mass is added, the greater is damping. For this reason, in this section damping factors are modified so as to counter the effects produced by the fact that mass is being added.

The following results shows the final stage displacement for coarsening parameters H=2 and H=4, which means that mass was augmented by a factor H^2, therefore, damping factor was reduced by a factor H^2 too. Solution is compared with reference solution and also with not modified damping parameter.

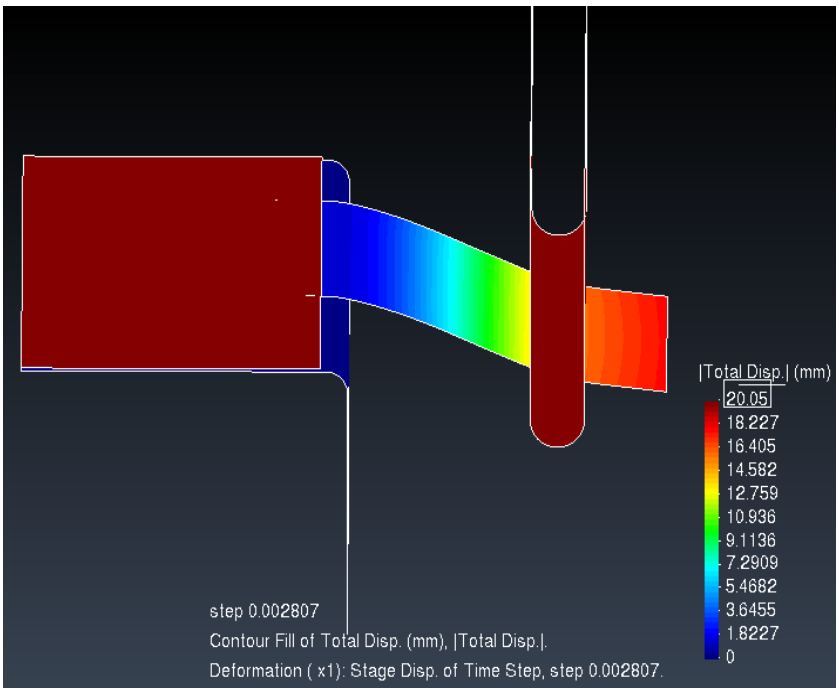

Figura 25: Reference solution without DC and using a 24x48 mesh. Deformed shape and colour representation for the total displacement.

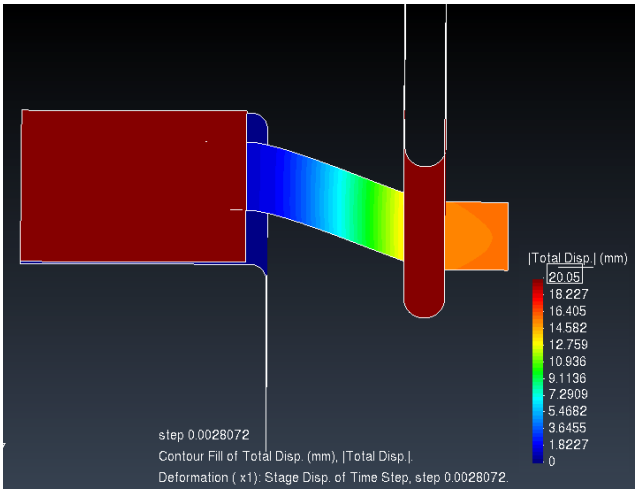

Figura 26: Solution using DC method with H=2. dt = 1,8·dt with default damping parameter.

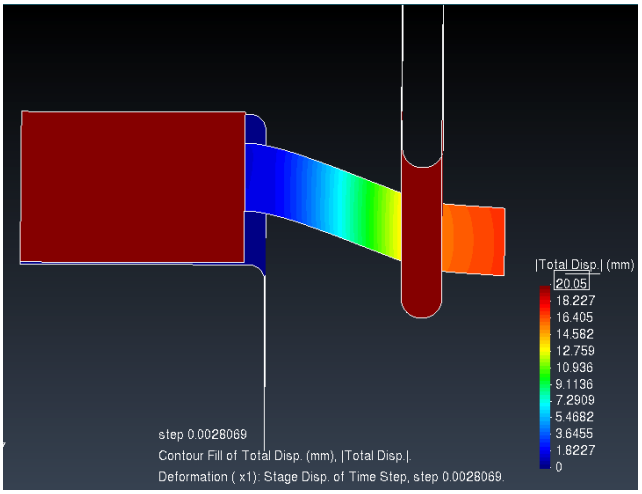

Figura 27: Solution using DC method with H=2. dt = 1,8·dt with changed damping parameter factor H^2.

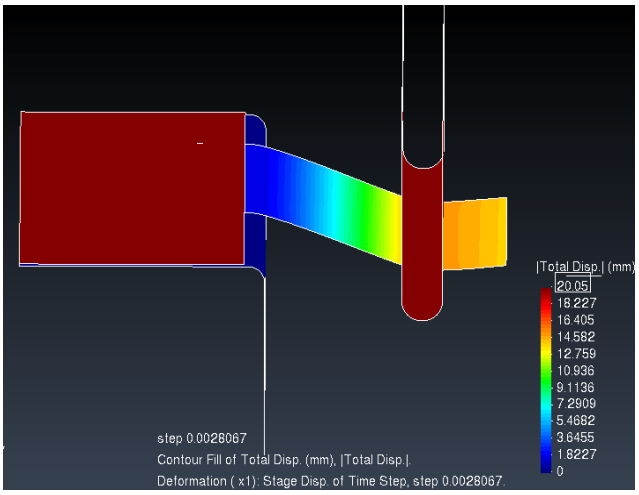

Figura 28: Solution using DC method with H=4, dt = 3,8·dt with default damping parameter.

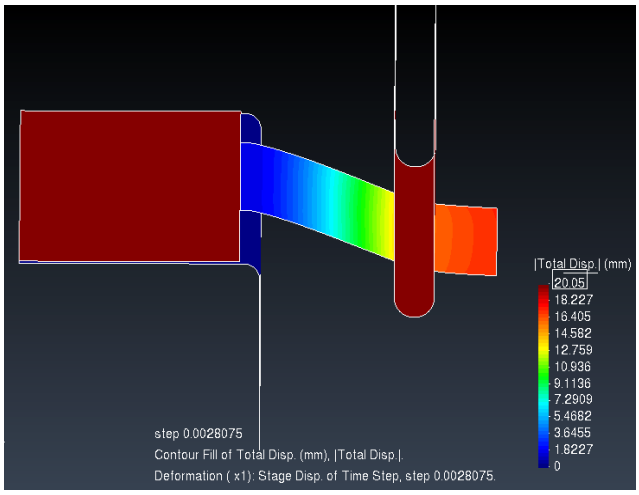

Figura 29: Solution using DC method with H=4. dt = 3,8·dt with changed damping parameter factor H^2.

Comparing the reference solution without DC and the solution with DC and default damping parameter, it is clear how mass affects the dynamics of the system. When mass is added, damping plays a greater role, and the deformation is smaller. This effect is greater, the more mass is added, as it can bee seen comparing *Figure.26* and *Figure.28* where final shape is clearly less deformed when H=4.

Finally, comparing the solution obtained by modifying the damping parameter, shows how this added mass effect is countered by changing the damping parameter a factor equal to the augmented mass factor.

	x	y	z	|r|
Stampack	-0,00309	-1,41962	-17,75684	17,81350
Visual Studio	-0,00243	-1,40568	-17,74860	17,80417

Tabla 6: Reference solution obtained with Stampack and Visual Studio solver.

factor	1/factor	damping	x	y	z	|r|
0,25	4	5	-0,00716	-1,06342	-15,62206	15,65822
0,25	4	20	-0,00917	-1,15680	-16,98077	17,02013

Tabla 7: Displacements and total displacement for factor equal to H^2 when H = 2, dt factor = 1,8, method version v = 1 and different damping factors (default and H^2).

factor	1/factor	damping	x	y	z	|r|
0,0625	16	5	-0,01134	-0,91040	-13,69292	13,72316
0,0625	16	80	0,00068	-0,95043	-17,13386	17,16020

Tabla 8: Displacements and total displacement for for factor equal to H^2 when H = 4, dt factor = 3,8, method version v = 1 and different damping factors (default and H^2).

As it can be seen, when damping is changed, the dynamics of the problem changes and a better solution is found. However, it must be said that the final deformed shape is slightly different from the reference solution, since the curvature of the blank is a little bit different.

7. Conclusions and future work

This work has proved, by means of several tests, that all algorithms needed to complete the auxiliary tasks required by the Dynamic Condensation method have been properly implemented and worked perfectly for the 'ad hoc' example used in this study. In this sense, all the information required to implement the DC method has been successfully set up.

Regarding the implementation of the DC method itself, it has been seen that the instability and oscillations produced when dt is higher than the critical time had been overcome with the DC implementation. However, the dynamics of the system changed considerably the higher the coarsening parameter H was.

It has also been seen that changing the damping factor to reduce its effects makes the system's dynamics behave similar to the reference solution although the final deformed shape presented slightly different curvatures.

As a future work:

- A general coarsening algorithm must be implemented for any problem
- More research must be done to determine the optimum augmented mass and damping coefficient, since no rule has been found so as to choose those values.

8. References

[1] Randolph E. Bank and Jinchao Xu, An algorithm for coarsening unstructured meshes

[2] Peter A. Model Reduction Techniques Powerpoint, Mechanical Engineering Department, University of Massachusetts Lowell.

[3] L.Morancay, G.Winkelmuller, Dynamic condensation and selective mass scaling in RADIOSS Explicit, 19th Congrès Français de Mécanique, 24-28 August 2009.