Elimination of the boundary conditions dualized

Summary :
The taking into account of the boundary conditions in *Code_Aster* can be realized either by a direct technique of elimination in simple cases (AFFE_CHAR_CINE), that is to say thanks to a technique of dualisation (AFFE_CHAR_MECA), and with the introduction of multipliers of Lagrange, for the cases more generals.

Nevertheless, this approach presents two disadvantages. On the one hand, the addition of multipliers of Lagrange increases the number of degree of freedom of the total problem, and thus the size of the system to be solved. This point can become penalizing since the number of boundary conditions increases (many interfaces, connections kinematics, rigid impositions of movements, etc). In addition, the particular technique of dualisation leads to the loss of the positive semi-definite character of the matrix of stiffness. This change of nature can lead to defects of robustness, in the case of to see with the failure of the resolution the iterative solveurs.

One presents in this document an alternative method to the introduction of multipliers of Lagrange to take into account the boundary conditions closely connected. This approach does not replace the introduction of the multipliers of Lagrange, in particular for the studies presenting of the contact. On the other hand, in the case of the studies where the boundary conditions are solidified, this approach allows profits of robustness, and performance in the case of a significant number of boundary conditions.

The elimination of the boundary conditions is carried out either by the use of the keyword ELIM_LAGR in SOLVEUR [U4.50.01], that is to say directly with the operator ELIM_LAGR [U4.55.03] which builds new concepts matr_asse.

Contents

1 Notations

$[K]_{Nt\,x\,Nt}$	Matrix of stiffness containing the boundary conditions dualized
$[A]_{NxN}$	Matrix of stiffness of the problem without boundary conditions
$[C]_{Nc\,x\,N}$	Matrix of definition of the boundary conditions or “matrix of the constraints”
$[I_d]_{Nc\,x\,Nc}$	Matrix identity
$[T]_{N\,x\,Nl}$	Matrix associated with the core (Ker) of C
$\{u\}_{Nx1}$	Vector of solutions displacements
$\{u_0\}_{Ncx1}$	Vectors of imposed displacements
$\{u_N\}_{Nlx1}$	Functions of form checking the boundary conditions
$(\omega_0, \{\varphi\}_{Nlx1})$	Clean modes checking the boundary conditions
$\{f\}_{N\,x\,1}$	Vector of the imposed loading
$\{\lambda\}_{Nc\,x\,1}$, $\{\lambda_1\}_{Nc\,x\,1}$, $\{\lambda_2\}_{Nc\,x\,1}$	Vectors of the multipliers of Lagrange
N	Many degrees of freedom of the nonconstrained problem
Nc	Many constraints
Nt	Full number of unknown factors • $N+Nc$ for a simple dualisation, • $N+2\mathrm{Nc}$ for a double dualisation,
Nl	Many independent unknown factors

2 Introduction

The taking into account of the boundary conditions in *Code_Aster* can be realized either by a direct technique of elimination in simple cases (AFFE_CHAR_CINE), that is to say thanks to a technique of dualisation (AFFE_CHAR_MECA), and with the introduction of multipliers of Lagrange, for the cases more generals. This last approach is very general and makes it possible to treat under the same formalism all the types of boundary conditions.

Nevertheless, this approach presents two disadvantages. On the one hand, the addition of multipliers of Lagrange increases the number of degree of freedom of the total problem, and thus the size of the system to be solved. This point can become penalizing since the number of boundary conditions increases (many interfaces, connections kinematics, rigid impositions of movements, etc). In addition, the particular technique of dualisation leads to the loss of the positive semi-definite character of the matrix of stiffness. The search for the solution is not then any more one research of minimum, but a search for point saddles [1]. This change of nature can lead to defects of robustness, to in the case of see with the failure of the resolution the iterative solveurs.

One presents in this document an alternative method to the introduction of multipliers of Lagrange to take into account the boundary conditions closely connected. This approach does not replace the introduction of the multipliers of Lagrange, in particular for the studies presenting of the contact. On the other hand, in the case of the studies where the boundary conditions are solidified, this approach allows profits of robustness, and performance in the case of a significant number of boundary conditions. One initially points out the general technique which governs the elimination of the boundary conditions, then one presents the approach selected to eliminate the boundary conditions in practice, as well as the various associated algorithms. Lastly, one in the case of presents the technique of elimination a problem to the eigenvalues.

3 Principle of the resolution by elimination

3.1 Definition of the problem to be solved

The system to be solved can be put in the following general form:

$$[A]\{u\}=\{f\} \tag{1}$$

under the constraint

$$[C]\{u\}=\{u_0\} \tag{2}$$

.

With the introduction of multipliers of Lagrange, the problem becomes

$$\begin{cases}[A]\{u\}+[C]^T\{\lambda\}=\{f\}\\ [C]\{u\}=\{u_0\}\end{cases}, \tag{3}$$

and puts itself in the general form

$$\begin{bmatrix}A & C^T\\ C & 0\end{bmatrix}\begin{Bmatrix}u\\ \lambda\end{Bmatrix}=\begin{Bmatrix}f\\ u_0\end{Bmatrix} \tag{4}$$

In the case of the double dualisation (AFFE_CHAR_MECA), this matrix is increased, and the system to be solved becomes

$$\begin{bmatrix}A & C^T & C^T\\ C & -\alpha I_d & \alpha I_d\\ C & \alpha I_d & -\alpha I_d\end{bmatrix}\begin{Bmatrix}u\\ \lambda_1\\ \lambda_2\end{Bmatrix}=\begin{Bmatrix}f\\ u_0\\ u_0\end{Bmatrix}. \tag{5}$$

After having posed naturally $\lambda_1=\lambda_2=\lambda/2$.
To make its character semi-definite positive to the generic problem $Ku=f$, if the matrix K present a topology rising from the setting in the form (4) or (5), one will seek to break up the solution u problem in the form $u=u_p+u_g$, where

- u_p is the particular solution of the problem,
- u_g is the general solution of the problem, associated with the boundary conditions $[C]\{u_g\}=\{0\}$.

3.2 Search for the particular solution

The search for the particular solution simply consists in finding u_p such as $[C]\{u_p\}=\{u_0\}$. To calculate u_p , one considers the decomposition on the image and the core of C , that is to say $u_p=u_{p-i}+u_{p-n}$, with u_{p-i} belonging to the image of C and u_{p-n} belonging to the core of C .

This problem can be solved within the meaning of least squares, and in this case, one considers the solution of (2) of standard L_2 minimal, that is to say $u_p=u_{p-i}$. $\{u_{p-i}\}$ is the solution of the problem of minimization:

$$\{u_{p-i}\}=\underset{\{u_p\}}{\mathrm{ArgMin}}\left(\left\|[C]\{u_p\}-\{u_0\}\right\|^2\right). \tag{6}$$

To calculate effectively u_{p-i} , provided C maybe of full row (row equal to Nc , therefore not of redundant constraints), one exploits the following property:

$$\left([C][C]^T\right)\left([C][C]^T\right)^{-1}=[I_d]_{NcxNc}, \tag{7}$$

and thus

$$\left([C][C]^T\right)\left([C][C]^T\right)^{-1}\{u_0\}=\{u_0\}. \tag{8}$$

The required solution is naturally given by:

$$\{u_p\}=\{u_{p-i}\}=[C]^T\left([C][C]^T\right)^{-1}\{u_0\} \tag{9}$$

In practice, the solution of (9) is not given by the direct calculation of $C^T\left(CC^T\right)^{-1}$, but by the calculation of the matrix R associated with the decomposition QR of C^T . The calculation of Q , full orthogonal matrix, is expensive and does not bring interesting information. On the other hand, the calculation of R is easy and inexpensive, and provides all the necessary information, since:

$$C^T\left(CC^T\right)^{-1}=[C]^T\left([R]^T[Q]^T[Q][R]\right)^{-1}=[C]^T[R]^{-1}[R]^{-T}. \tag{10}$$

By construction, R is a matrix of size $NxNc$, with a higher triangular block on Nc first lines:

$$[R]_{NxNc}=\begin{bmatrix}[R_{ts}]_{Nc\,x\,Nc}\\ [0]_{(N-Nc)xNc}\end{bmatrix} \text{ With } [R_{ts}]=\begin{bmatrix}R_{1,1} & \cdots & R_{1,Nc}\\ 0 & \ddots & \vdots\\ 0 & 0 & R_{Nc,Nc}\end{bmatrix}. \tag{11}$$

In practice, successive calculations of the effect of R^{-1} and R^{-T} on $R^{-T}u_0$ and u_0 respectively are done by simple gone up descent.

Note:

The calculation of R by decomposition QR allows to manage the case of the redundant constraints. If C is not full row, then of the zeros appear on the diagonal of R. It is then enough to remove the corresponding columns, which corresponds to the suppression of the redundant constraint. One builds a particular solution then checking (2), of minimal standard, even in the case of a matrix C of defective row.

3.3 Search for the general solution

The general solution u_g checking $[C]\{u_g\}=\{0\}$, it belongs to the core T of C. The search for the general solution thus passes by a stage of construction of the core of C. One can then project the problem on the core, and solve this problem, posed better and of reduced size.

3.3.1 Obtaining a base of the core

Formally, the construction of the core of C can be done in several ways:

- Decomposition in singular values of C, that is to say $C=USV^T$. $Ker(C)$ corresponds to the columns of V^T associated with a worthless singular value.
- Decomposition QR of C^T, that is to say $C=R^TQ^T$. $Ker(C)$ corresponds then to the columns of Q associated with a zero value on the diagonal of R^T.
- Decomposition LU of C^T. The calculation of the core is not direct, and is obtained starting from under-blocks of L.

One poses:

$$[L]=\begin{bmatrix}[L_1]_{NcxNc} & 0\\ [L_2]_{(N-Nc)xNc} & [L_3]_{(N-Nc)x(N-Nc)}\end{bmatrix} \quad (12)$$

Where L_1 and L_3 are lower triangular matrices. $Ker(C)$ is obtained by

$$Ker(C)=\begin{bmatrix}-([L_2][L_1]^{-1})^T\\ [I_d]_{(N-Nc)x(N-Nc)}\end{bmatrix} \quad (13)$$

One makes the assumption here that C is of full row. If such is not the case, then it is enough to build L_1 starting from the block of L associated under the nonworthless terms of the diagonal of U.

In practice, these techniques are adapted only for problems of reduced size. The decomposition in singular value is the most expensive approach, followed by the decomposition QR, then decomposition LU. In this case, this decomposition is carried out by a particular algorithm, presented in the section 4, fascinating of account the specificity of the problem to be solved.

3.3.2 Projection and resolution

Once the base of the core is calculated, the resolution is direct. One poses $u_g=T\,\bar{u}$, and the solution of the problem thus is directly given by

$$[T]^T[A][T]\{\bar{u}\}=T^T\{f\}, \quad (14)$$

since, by construction,

$$[T]^T[C]^T\{\lambda\}=\{0\}. \quad (15)$$

In certain cases, on the other hand, it is necessary to reach the multipliers of Lagrange. They thus should be rebuilt, starting from the solutions u_p and u_g. With this intention, one must solve the first line of the system (3), that is to say

$$[C]^T\{\lambda\}=\{f\}-[A]\left([T]\{\bar{u}\}+\{u_p\}\right) \quad (16)$$

If C is of full row, the solution is written formally:

$$\{\lambda\}=\left([C][C]^T\right)^{-1}[C]\{f\}-[A]\left([T]\{\bar{u}\}+\{u_p\}\right), \quad (17)$$

and is calculated in practice by re-using the decomposition QR of C^T calculated for the search for the particular solution. In a similar way, if C is not full row, it is possible to relieve the redundant stresses.

4 Algorithm retained for the construction of the core of the "matrix of the constraints"

4.1 Objectives

To manage the elimination of the constraints kinematics, a calculation algorithm of the core was developed to take into account specificities of the problem to be solved. In the framework of the finite elements, the matrix A corresponds to the matrix of stiffness of the problem, assembled without taking into account of the boundary conditions. This matrix, even in the case of problems of big size, remains very hollow.

In addition, independently of the problems of cost of calculation involved in obtaining the core of C by the techniques of decomposition, the matrices obtained by these processes are in general full. The operation of projection (14) conduit thus, in this case, with a problem of size reduced, but very dense, which is thus unsuited for the solveurs used.

Methodology developed here makes it possible to build, in a sequential way, a base T core of C while ensuring a minimal filling, in order to preserve a hollow character for the problem (14). This approach is all the more effective as the hollow character of the matrix C is important.

4.2 Total algorithm

The construction of the base is made in a sequential way. One initializes the process by building a matrix $T_0 = I_d$ size $N \times N$, that is to say the number of degrees of freedom of the nonconstrained problem. The final matrix is also initialized $T = T_0$. One initializes finally the number of eliminated constraints $Nbce$ with 0. One buckles then on the lines of the matrix C, and the following operations are carried out:

As long as $Nbce < N$, one makes:
* Iteration on the lines of the matrix C

- One initializes to zero a vector $CONT$ containing the indices of the DDL already implied in other constraint
- One initializes $NbCont = 0$

- First iteration:

  - One initializes to zero a vector CE of size Nc to store the numbers of the eliminated constraints.
  - One initializes to zero a vector NZ of size N to store the indices of the degrees of freedom already implied in a linear relation.
  - The first line is extracted C_1
  - The extracted line is normalized: $\|C_1\| = 1$
  - One calculates $\|C_1 T_0\|$

  - If $\|C_1 T_0\| < \mathrm{Tol}$, the constraint is already checked
    - It is indicated that the first constraint is eliminated: $CE(1) = 1$
    - One passes to the following iteration

  - If $\|C_1 T_0\| \geqslant \mathrm{Tol}$ (in the code, $\mathrm{Tol} = 10^{-12}$ by default), the constraint is not checked. The indices are sought i_{nz} associated under the nonworthless terms of the first line C_1.
    - If $Card(i_{nz}) = 1$, then the kinematic condition implies one degree of freedom.
      - One thus puts at 0 the term associated on the diagonal with T_0 : $T_0(i_{nz}, i_{nz}) = 0$

- If not, one builds a base of the core of $C_1(i_{nz})$. This base is built in an iterative way, to lead to a triangular matrix. The detail is given in the section 4.3. One note Ker_1 this matrix, of size $Card(i_{nz})\times Card(i_{nz})$
  - One affects Ker_1 in T_0 : $T_0(i_{nz},i_{nz})=Ker_1$. It will be noted that, by construction, one of the elements of the diagonal of Ker_1 is null.
- The positions are stored i_{nz} nonworthless elements of C_1 in NZ : $NZ(i_{nz})=1$
- It is indicated that the first constraint is eliminated: $CE(1)=1$
- It is indicated that the DDL i_{nz} are implied in a constraint: $CONT(NbCont+(1:Card(i_{nz})))=i_{nz}$
- One increments the number of already constrained DDL: $NbCont=NbCont+Card(i_{nz})$
- One passes to the following iteration

* following Iterations:

- The current line is extracted C_k
- The current line is normalized: $\|C_k\|=1$
- One calculates $\|C_k T_0\|$
- If $\|C_k T_0\|<\mathrm{Tol}$:
  - It is indicated that the constraint is eliminated: $CE(k)=1$
  - One passes to the following iteration

- If $\|C_k T_0\|\geqslant \mathrm{Tol}$, the indices are sought i_{nz} associated under the nonworthless terms of the line C_k .
- One determines so certain degrees of freedom are already implied in other linear relations, and one recovers the associated indices: $i_{cs}=\{i \mid NZ(i)=1, i\in i_{nz}\}$
- The complementary unit is built i_{lb} . One thus has $i_{nz}=\{i_{cs};i_{lb}\}$
  - If $Card(i_{lb})=0$: All the degrees of freedom are already implied in linear constraints. Consequently, the block $T_0(i_{nz},i_{nz})$ was already filled, and to modify it would result in not checking one of the preceding constraints more.
    - Nothing is done, and one passes to the following iteration. The constraint C_k is not eliminated for the moment.

  - If $Card(i_{nz})=1$, and $i_{lb}=i_{nz}$, then the kinematic condition implies one degree of freedom, not yet implied in a preceding relation.
    - One thus puts at 0 the term associated on the diagonal with T_0 : $T_0(i_{nz},i_{nz})=0$
    - It is indicated that the DDL i_{nz} is implied in a constraint: $CONT(NbCont+1)=i_{nz}$
    - One increments the number of already constrained DDL: $NbCont=NbCont+1$

  - If not, one builds a base of the core of $C_k(i_{nz})$ under the constraint not to modify the blocks of T_0 already initialized, that is to say $T_0(i_{cs},CONT(1:NbCont))$. The problem to be solved is written formally

To find $T_0(i_{cs},i_{lb})$, $T_0(i_{lb},CONT(1:NbCont))$ and $T_0(i_{lb},i_{lb})$ such as

$$[C_k(i_{cs})\ \ C_k(i_{lb})]\begin{bmatrix} T_0(i_{cs},CONT(1:NbCont)) & T_0(i_{cs},i_{lb}) \\ T_0(i_{lb},CONT(1:NbCont)) & T_0(i_{lb},i_{lb}) \end{bmatrix}=[0] \quad (18)$$

- If $Card(i_{cs})=0$, one only builds $T_0(i_{lb},i_{lb})$, the other terms are degenerated, with the algorithm presented in the section 4.3, which calculates the solution of the problem $C_k(i_{lb})T_0(i_{lb},i_{lb})=0$

- If $Card(i_{cs})>0$:
  - One builds $T_0(i_{lb},i_{lb})$ with the algorithm presented in the section 4.3, which calculates the solution of the problem $C_k(i_{lb})T_0(i_{lb},i_{lb})=0$
  - One calculates $T_0(i_{lb},CONT(1:NbCont))$ as solution of the problem to least squares

$$T_0(i_{lb},CONT(1:NbCont))=\underset{T}{\mathrm{Argmin}}\left(\left\|C_k(i_{cs})T_0(i_{cs},CONT(1:NbCont))+C_k(i_{lb})T\right\|^2\right) \quad (19)$$

  - One calculates $\left\|C_k(i_{cs})T_0(i_{cs},i_{cs})+C_k(i_{lb})T\right\|$
  - If $\left\|C_k(i_{cs})T_0(i_{cs},i_{cs})+C_k(i_{lb})T\right\|\geqslant Tol$, the constraint cannot be eliminated with this stage, one passes to the following iteration.
  - One imposes $T_0(i_{cs},i_{lb})=0$

- The positions are stored i_{nz} nonworthless elements of C_k in NZ : $NZ(i_{nz})=1$
- It is indicated that the DDL i_{lb} are implied in a constraint: $CONT(NbCont+(1:Card(i_{lb})))=i_{lb}$
- One increments the number of already constrained DDL: $NbCont=NbCont+Card(i_{lb})$
- It is indicated that the current constraint is eliminated: $CE(k)=1$
- One passes to the following iteration

* After the last iteration, one calculates the sum of the terms of CE : $Nbne=Nbne+\sum_{k=1}^{Nc}CE(k)$.

- One updates the base of the core starting from the already eliminated constraints $T=T_0\ T$
- One updates the matrix of the constraints for iterative construction: $C=C\ T_0$
- If $Nbne<N$
  - One resets $T_0=I_d$
  - One eliminates the lines from C correspondent with already eliminated constraints
  - The vector is reset NZ .

End

The process as described Ci above converges, since one ensures to eliminate at least the first constraint with each passage in the total loop. On the other hand, the quality of the matrix T (many nonworthless terms) degrades itself quickly when the total iteration count increases.

4.3 Iterative construction of the core of a vector

The total algorithm presented in the section 4.2 rest on successive constructions of cores of vectors, corresponding to the lines extracted the matrix C . The process of extraction of the lines of C ensuring itself to recover only the nonworthless values of the current line, an algorithm was built to assemble an orthogonal base in the shape of a higher triangular matrix, in order to limit the filling of the matrix T finale. The idea consists in building an orthogonal base repeatedly, by ensuring that each new vector is orthogonal:

- with the vector V ,
- with the columns of Ker already filled.

It is supposed, for the continuation, that the vector V is of unit standard. This treatment is carried out before the call to this routine.

Higher triangular topology makes it possible to easily ensure the construction of such a matrix. The algorithm is presented as follows:
That is to say V the vector given as starter, length Nnz .

- The matrix is initialized Ker to zero
- One buckles on the values of V to determine the position i_{nz} first element whose standard is higher than a tolerance Tol fixed at the precision machine (R8PREM [D6.01.01])
- If $i_{nz}=1$, then $Ker(1,1)=0$. If not, $Ker(1,1)=1$
- One buckles on the values of V , while starting with the second value: $i=2:Nnz$
  - One tests the value of $V(i)$
    - If $V(i)<Tol$, one makes $Ker(i,i)=1$
    - If not, if $i=i_{nz}$, one makes $Ker(i,i)=0$
    - If not,
      - one buckles on $j=1:i-1$
        - The scalar product is calculated $ps=(Ker(1:j-1,j),Ker(1:j-1,i))$
        - One fills $Ker(j,i)=\frac{-ps}{Ker(j,j)}$
      - The scalar product is calculated $ps=(Ker(1:i-1,i),V(1:i-1))$
      - One fills $Ker(i,i)=\frac{-ps}{V(i)}$
      - One normalizes $Ker(1:i,i)=\frac{Ker(1:i,i)}{\|Ker(1:i,i)\|}$

4.4 Calculation of the decomposition “QR-Qless”

To calculate the particular solution, and to rebuild the multipliers of Lagrange, one calls on a decomposition QR matrix of the constraints. With this intention, one used the standard algorithm, published in [2]. This algorithm, whose form was adapted for its use through the PETSc bookstore, is simply reproduced here. The matrix C is given as starter. The diagonal of the matrix is stored R in a vector d . The decomposition crushes C , since R is stored in the higher triangular part. The lower part corresponds under the nonworthless terms of the vectors w being used with the transformation as Householder. The matrix C is of size $m\times n$.

for J = 1 to N C
$s=\sqrt{\|C(j:m,j)\|^2}$;
calculation of the diagonal of R
yew $C(j,j)>0$ then
$d(j)=-s$;
else
$d(j)=s$;
endif
# calculation of the higher triangular part of R
$\alpha=\sqrt{(s(s+|C(j,j)|))}$;
$C(j,j)=C(j,j)-d(j)$;
$C(j:m,j)=C(j:m,j)/\alpha$;
calculation of the vectors w_i for the transformation of Householder: $P_i=I_d-w_i w_i^T$
for I: = J + 1 to N C
$s=C(j:m,j)^T C(j:m,i)$;
$C(j:m,i)=C(j:m,i)-s\,C(j:m,j)$;

```
        end for I;
    end for J;
```

5 Clean modes and elimination

This section echoes section 7 of R3.03.01 documentation, and justifies the method of elimination for the calculation of the clean modes. One is interested here in the problem following general:

To find the couples $\left(\omega_0^2,\{\varphi\}\right)$ checking $\left([K_c]-\omega_0^2[M_c]\right)\{\varphi\}=0$ (20)

Matrices M_c and K_c must take into account the boundary conditions, and are supposed to be obtained here starting from functions of forms u_N checking $Cu_N=0$. However, in this case, the assembled problem is not the problem (20), but a problem of more important size, for which the functions of forms do not check the constraints kinematics. Matrices M_c and K_c are of dimension $(N-Nc)\times(N-Nc)$.

On the basis of the assembled matrices M and K, of size $N\times N$, built on the basis of function of forms u not checking $Cu=0$, there exist two alternatives to solve the problem (20). The first consists in increasing the matrix K to reveal the constraints kinematics, by adding unknown factors in the form of multipliers of Lagrange. It is the approach adopted until now in the code. The problem to be solved is written, with the technique of double dualisation:

$$\left(\begin{bmatrix} K & C^T & C^T \\ C & -Id & Id \\ C & Id & -Id \end{bmatrix}-\omega_0^2\begin{bmatrix} M & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\right)\begin{Bmatrix} \varphi \\ \lambda_1 \\ \lambda_2 \end{Bmatrix}=\begin{Bmatrix} 0 \\ 0 \\ 0 \end{Bmatrix} \quad (21)$$

The approach by additions of multipliers of Lagrange, in the case of the research of the clean modes, is not satisfactory, since besides the loss of the properties of positivity of the operator, one adds a significant number of degrees of freedom, and one thus widens the spectrum of the problem. This widening of spectrum poses digital problems, which oblige with an important work in this case.

The second approach, that proposed here, are more natural, and approach the initial problem. The matrices of mass and stiffness are certainly assembled on the basis of function of forms not checking the limiting conditions, but the research of the clean modes and the eigenvalues can be done under adapted space. It is enough to build a base of under space of the vectors v checking $Cv=0$. It under space is naturally the core of C. It is then enough to search the modes of the problem nonconstrained project in the core of C. That is to say T a base of the core, one seeks the couples then $\left(\omega_0^2,\{\psi\}\right)$ who check

$$\left[T^T\left(K-\omega_0^2M\right)T\right]\{\psi\}=\{0\} \quad (22)$$

One identifies then $M_c=T^T M T$ and $K_c=T^T K T$.

Efforts of reactions f_0, calculated by $f_0=\lambda_1+\lambda_2$ in the method with double dualisation (see relation 21), are simply provided by

$$\left[C\left(K-\omega_0^2M\right)T\right]\{\psi\}=\{f_0\} \quad (23)$$

6 Bibliography

[1] Benzi, Mr. and Golub, G.H., and Liesen, J.: Numerical solution of saddle not problems, Recorded Numerica, vol. 14, pp1 - 137 (2005).

[2] Gander, W.: Algorithms for the QR-Decomposition, Research carryforward n° 80-02 (1980).

7 Appendix: code matlab of the construction of the core (section 4.2)

```
function T=algorithme_elim (C);

C_ini=C;
T=speye (size (C, 2));
CONT=zeros (size (C, 2), 1);
NbCont=1;
nv_const= []; %-- handling of not verified constrainsts
boucle=1;
Tf=T;
while boucle==1
    boucle=0;
    for i1=1: size (C, 1)
        C (i1:) =C (i1:). /norm (C (i1:));
        yew norm (C (i1:) *T) >0  %-- constraint not verifiee
            ind=find (C (i1:));
            yew length (ind) ==1
                CONT (NbCont) =ind;
                NbCont=NbCont+1;
                T (ind, ind) =0;
            else
                lib= [];
                cont= [];
                for j1=1: length (ind)
                    yew ~isempty (find (CONT==ind (j1)))
                        cont= [cont ind (j1)];
                    else
                        lib= [lib ind (j1)];
                    end ;
                end ;
                yew isempty (cont)
                    [Q, R] =qr (C (i1, ind) ');
                    Q (: , 1) =0;
                    T (ind:) =q*T (ind:);
                    CONT (NbCont+ (0: length (ind) - 1))=ind;
                    NbCont=NbCont+length (ind);
                else
                    yew ~isempty (lib)

                        out= (CONT (1: NbCont-1));
                            T (lib, out) = (C (i1, lib))\ (C (i1, cont) *T (cont,
out));

                        [Q, R] =qr (C (i1, lib) '); Q (: , 1) =0;
                        T (lib, lib) =q;

                        lev=max (max (ABS (C (i1:) *T)));
                        yew ABS (lev) >1e-10
                            disp ([ 'forced' num2str (i1) 'badly eliminee…' ]);
                            disp ([ 'max (|Ci.T|) =' num2str (full (lev))]);
                            T (lib, lib) =speye (length (lib));
                            T (lib, out) =0;
                        else
                            CONT (NbCont+ (0: length (lib) - 1))=lib;
                            NbCont=NbCont+length (lib);
                        end ;
```

```
                    else
                        lev=max (max (ABS (C (i1:) *T)));
                         yew ABS (lev) < 1e-15
                            disp ([ 'forced' num2str (i1) 'eliminee…' ]);
                        else
                            disp ([ 'forced' num2str (i1) …
                                'not verifiee! ' , num2str (full (lev))]);
                            nv_const= [nv_const i1];
                            boucle=1;
                        end ;
                    end
                end ;
            end ;
        end ;

    end ;

    yew buckle ==1
        disp ( 'Restarting…' );
        %-- only the active columns are kept
        ii=find (sum (ABS (T), 1));
        T=T (: , II);

        %-- If that did not succeed the first time, the constraint is projected
        %-- in new space, and one starts again…
        C=C*T;
        C=C (nv_const:);
        Tf=Tf*T;
        T=speye (size (C, 2));
        nv_const= [];
        CONT=zeros (size (C, 2), 1);
        NbCont=1;
    end ;
end ;

%-- last projection
T=Tf*T;

%-- one keeps only the columns of the active ddl
ii=find (sum (ABS (T), 1));
T=T (: , II);

res=max (max (ABS (C_ini*T)));
s2=sprintf ( '%s %0.5g' , 'Max. residue: ' , full (LMBO));
disp (s2);
```