In work of a computation of eigen modes of a Résumé structure

This

document bets presents to a total sight of the various approaches available in *Code_Aster* for compute the eigen modes of vibration of a mechanical structure. These approaches are described on the basis of simplest to implement, for standard studies, and while going gradually worms of the implementations more worked out for advanced studies.

One presents the sequences necessary of the operators of *Code_Aster*, without entering in detail of each operator.

Contents

1 Rappels: formulation of the problem

One considers a mechanical structure represented, in the frame of a modelization by finite elements, by its stiffness matrixes K , of mass M and possibly of damping C . The equation governing the evolution of structure is written $M\ddot{x}+C\dot{x}+Kx=0$.

One wants to characterize free vibrations of mechanical structure, defined by eigen frequencies $f_i=\frac{\omega_i}{2\pi}$ (ω_i : own pulsation of the n° mode i) and the modal deformed shapes x_i associated (and modal dampings ζ_i if models it contains damping).

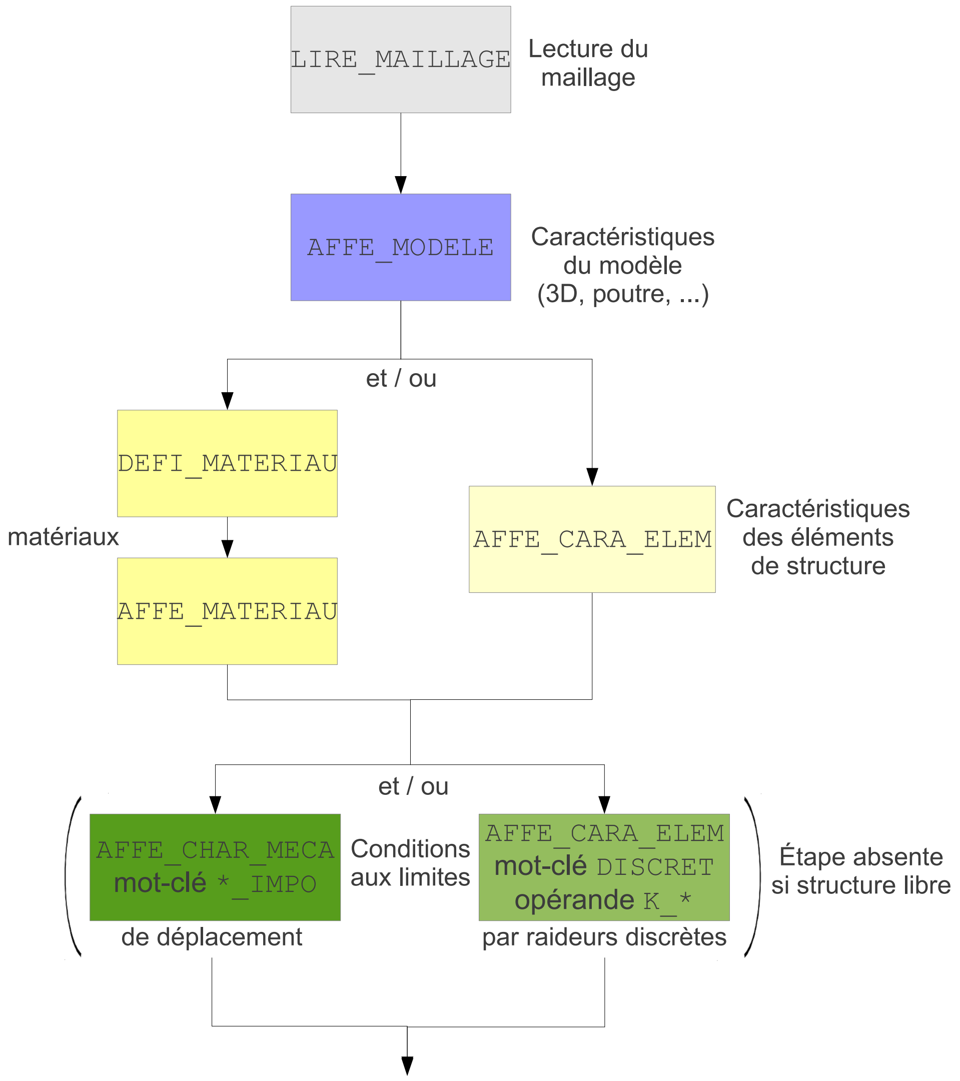
In the absence of damping (the simplest case and most frequent), modal computation consists in finding the couples $\left(\omega_i, x_i\right)$ such as $\left(K-\omega_i^2 M\right)x_i=0$.

2 In fact of the case

the setting in data for a computation of eigen modes of vibrations bets is conventional and common to the majority of computations of mechanics in *Code_Aster* :

- reading of the mesh (operator `LIRE_MAILLAGE`),
- assignment of the characteristics of the model: model of type beam or 3D or…? (`AFFE_MODELE`),
- definition and assignment of materials (`DEFI_MATERIAU` and `AFFE_MATERIAU`) and/or assignment of the characteristics of structural elements (`AFFE_CARA_ELEM`),
- possible imposition of boundary conditions (stage goes away if the structure is completely free). The characteristic of modal computation in *Code_Aster* currently is that it is necessary in general that the boundary conditions of displacement, if there is, are imposed by dualisation (`AFFE_CHAR_MECA`, key word factor `DDL_IMPO` - more running - or `FACE_IMPO` or `ARETE_IMPO`) rather than by kinematical loads (`AFFE_CHAR_CINE`).

Appear 2-a schematizes the setting in data of a problem of modal computation.


LIRE_MAILLAGE
Lecture du maillage
AFFE_MODELE
Caractéristiques du modèle (3D, poutre, ...)
et / ou
DEFI_MATERIAU
matériaux
AFFE_MATERIAU
AFFE_CARA_ELEM
Caractéristiques des éléments de structure
et / ou
AFFE_CHAR_MECA mot-clé *_IMPO
de déplacement
Conditions aux limites
AFFE_CARA_ELEM mot-clé DISCRET opérande K_*
par raideurs discrètes
Étape absente si structure libre

Appear 2-a : Bets in data of a problem of modal computation.

Note:

- *For a modal computation, no excitation is necessary, except if one wants to take into account the effect of stiffness geometrical brought by a static loading (advanced study). If necessary, the 3.2.2.3 3.2.2.3 indicates the step to be adopted (advanced study).*
- *For a simple study, the structure is not prestressed: the possible boundary conditions in displacement are generally null. If one wants to take into account the prestressing generated by non-zero displacements, it is necessary to there too adopt the step indicated at the 3.2.2.3 3.2.2.3.*

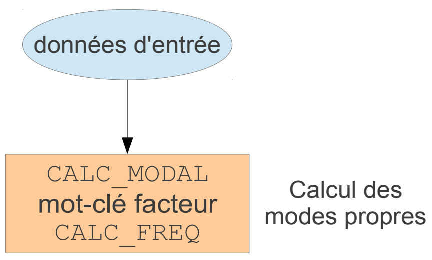
3 Computation of the eigen modes of vibration of a À partir des

structure given of input seen to the preceding paragraph, one presents here to the various possibilities offered by *Code_Aster* for compute the eigen modes of a structure, while going from simplest from implementation, with more intricate sequences.

3.1 Simplest studies

Pour the simple studies (one considers here a structure modelized without damping, prestressing, fluid interaction - structure, without gyroscopy,…, and with a "reasonable" number of degrees of freedom), the most ergonomic solution is to use the operator `CALC_MODAL` whose syntax is very synthetic. This operator carries out the computation of the eigen modes directly starting from the data input of the mechanical problem, while realizing, in a transparent way for the user, the computation of the assembled matrixes representing structure.

For the first computation, one advises to leave the parameters by default of the algorithm of resolution and checking of the results: the user must only with informing his area of search of the eigen frequencies thanks to the key word factor `CALC_FREQ`.



Appear 3.1-a : Computation of the eigen modes by the simplest procedure.

Example:

computation of the eigen mode nearest to $50\,Hz$:

```
modes = CALC_MODAL (MODELE = model,
                    CHAM_MATER = ch_mat,
                    CARA_ELEM = cara_el,
                    LOAD = c_limite,
                    CALC_FREQ = _F (OPTION = "CENTRE",
                                    FREQ = 50. ,
                                    NMAX_FREQ = 1));
```

Note:

Operator `CALC_MODAL` also allows to treat structures with viscous damping. It is necessary for that to inform the key word factor `AMORTISSEMENT='OUI'`.

3.2 More advanced functionalities: *via* a preliminary computation of the assembled matrixes

operator `CALC_MODAL` is actually a macro-command which connects certain elementary commands in a preset way. Its field of application is thus necessarily restricted with relatively simple studies.
For advanced studies, one will need to know the assembled matrixes (stiffness, mass, damping) representing structure. Examples of their utility are given to paragraph 3.2.2.

3.2.1 Sequence of the commands *Code_Aster*

À partir des given of input, one proceeds as follows:

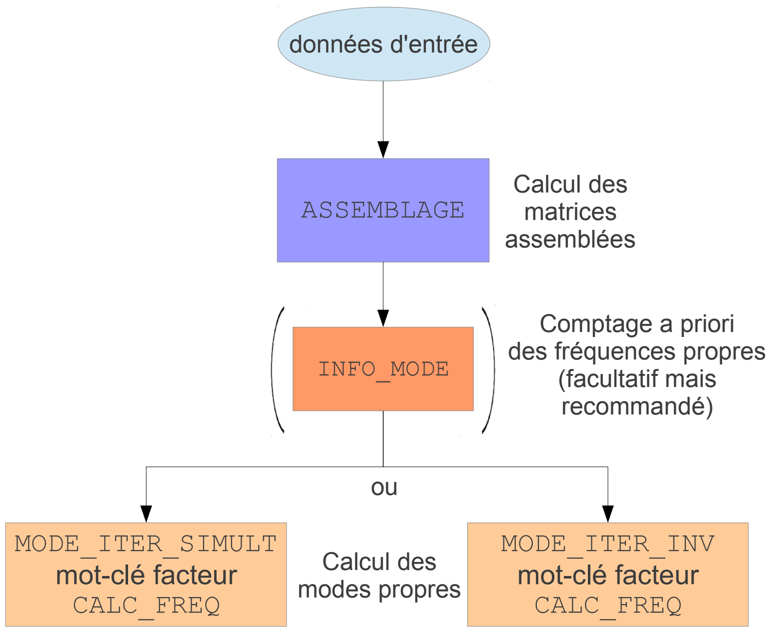
- computation of the assembled matrixes (operator `ASSEMBLY` with options "`RIGI_MECA`" and "`MASS_MECA`" ; other options exist for compute of the more specific matrixes, for example geometrical rigidity, damping, the gyroscopy, etc);

- computation of the eigen modes. For that, one has two operators who differ by their algorithms from resolution : `MODE_ITER_SIMULT` and `MODE_ITER_INV`.

`MODE_ITER_SIMULT` is to be privileged for its performances CPU if one seeks a relatively significant number of modes (up to 50 to 80; beyond, one recommends to cut out search in several sub-bands, cf 3.2.4.1 3.2.4.1), in particular with the option of search on a given tape (`OPTION='BANDE'`) for a better robustness.

On the contrary, `MODE_ITER_INV`, much more expensive, is to be used rather for compute some modes with very a good quality, for example if one wants to refine first estimates of eigen modes (cf paragraph 3.2.3).

Initially, it is advised to leave the parameters by default of these operators, and to specify only the area of search of the eigen frequencies.


données d'entrée
ASSEMBLAGE
Calcul des matrices assemblées
INFO_MODE
Comptage a priori des fréquences propres (facultatif mais recommandé)
ou
MODE_ITER_SIMULT mot-clé facteur CALC_FREQ
Calcul des modes propres
MODE_ITER_INV mot-clé facteur CALC_FREQ

Appear 3.2.1-a : Computation of the modes via the assembled matrixes.

Example:

computation of the eigen modes on the tape $[20\ ;300]\,Hz$:

```
ASSEMBLY (MODELE = model,
            CHAM_MATER = ch_mat,
            CARA_ELEM = cara_el,
            LOAD = c_limite,
              NUME_DDL  = CO  (“numbered”), # creation of a classification
of
                                                    # D.O.F.
            MATR_ASSE = (
                                    _F  (MATRICE=  CO  (“matr_k”),  OPTION=
“RIGI_MECA”),
                                    _F  (MATRICE=  CO  (“matr_m”),  OPTION=
“MASS_MECA”),
                        )
          )

modes = MODE_ITER_SIMULT (MATR_RIGI = matr_k,
                        MATR_MASS = matr_m,
                        CALC_FREQ =_F (OPTION = "BANDE",
                                       FREQ = (20. , 300.))
                      )
```

3.2.2 Utility of an intermediate computation of the assembled matrixes: some preliminary

3.2.2.1 Comptage examples of the Avant

eigen frequencies the computation itself of the eigen modes, it is strongly recommended to carry out a counting of the eigen modes contained in one or the wavebands given (in the standard case of real-modes; if the modes with compute are complex, it will be about a counting around a point of the complex plane). This counting is much faster than the computation strictly speaking of the eigen modes.
The counting of the eigen modes is carried out by operator `INFO_MODE` .
Knowledge *a priori* amongst eigen frequencies contained in the tape seeks has a double utility of checking and performance optimization CPU of modal computation:

- checking: one can check that the number of eigen modes computed by the modal solver is indeed equal to the number of eigen modes counted *a priori* ;
- tweaking CPU: if the number of eigen frequencies counted on the frequential tape of search is too high (a threshold ranging between 50 and 80 is usually noted), the user will be able to cut out his tape of search in several sub-bands, thanks to operator `MACRO_MODE_MECA` (cf 3.2.4.1 3.2.4.1).

In a first approach, the user can be satisfied to inform

- in the standard case of real-modes: matrixes of structure with key words `MATR_*` like its (its) tape (S) of search with key word `FREQ` ;
- in the case of complex modes (for example: structures with damping,…): TYPE_MODE= should `be specified'COMPLEXE'` and give the disc of search in the complex plane by `RAYON_CONTOUR` (and possibly `CENTRE_CONTOUR`) instead of `FREQ`.

3.2.2.2 Structures with hysteretic damping

operator simple `CALC_MODAL` does not allow to compute the eigen modes of a structure with hysteretic damping. One thus needs compute explicitly the assembled matrix of overall rigidity including the hysteretic contribution (complex matrix).
The sequence of the operators is the following:

- computation of the assembled matrixes of overall rigidity (conventional rigidity + hysteretic rigidity) and of mass (`ASSEMBLY` with options `"RIGI_MECA_HYST"` and `"MASS_MECA"` respectively);
- modal computation with like input the total stiffness matrix (complex) and mass matrix (`MODE_ITER_SIMULT` or `MODE_ITER_INV`).

One will refer to documentation [U2.06.03] for more information on the taking into account of hysteretic damping in *Code_Aster*.

3.2.2.3 Taking into account of prestressed

the taking into account of prestressed (non-zero boundary conditions, static external loadings,…) require of compute the assembled matrixes of mechanical and geometrical rigidity. One can then combine them to form the assembled matrix of overall rigidity which is that used for modal computation.
The sequence of the operators is the following, in a relatively simple case of static external loading:

- definition of the external loading (operator `AFFE_CHAR_MECA`, with for example key word `FORCE_NODALE`),

- computation of the stress field associated with this loading (operator `MECA_STATIQUE` or `STAT_NON_LINE` or `MACRO_ELAS_MULT` to calculate the static response, then `CREA_CHAMP` with key word `OPERATION='EXTR'` to recover the stress field),
- computation of the assembled matrixes of mechanical rigidity, geometrical rigidity associated with the stress field, and with mass (`ASSEMBLY`),
- mechanical combination of the stiffness matrixes and geometrical rigidity to form the total stiffness matrix (`COMB_MATR_ASSE`),
- modal computation with like input the total stiffness matrix and mass matrix (`MODE_ITER_SIMULT` or `MODE_ITER_INV`).

Example:

Benchmark SDLL101 presents an example of computation of the modes of a beam subjected to static forces.

3.2.2.4 Taken into account of the gyroscopy (revolving machines)

En plus des other matrixes assembled (of stiffness, mass and possibly damping other than gyroscopic), it is necessary to calculate the gyroscopic damping matrix with option "`MECA_GYRO`". Operator `CALC_MODE_ROTATION` then allows to calculate the eigen modes of structure for various rotational speeds defined by the user under key word `VITE_ROTA`.
One can then plot the diagram of Campbell (evolution of the eigen frequencies according to rotational speed) of revolving structure thanks to operator `IMPR_DIAG_CAMPBELL`.

Note:

Operator `CALC_MODE_ROTATION` is actually a macro-command calling `MODE_ITER_SIMULT` : if need be, the user can thus carry out the various elementary stages of `CALC_MODE_ROTATION` "to the hand" but in a way much less ergonomic. Benchmark SDLL129 illustrates the step in the case of a rotor with bearings whose characteristics depend on rotational speed.

3.2.2.5 Use of the modes for a dynamic computation on modal base

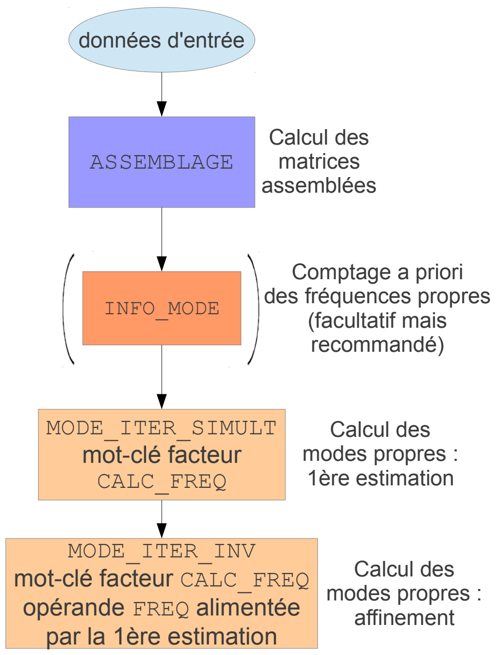
It is there too necessary to have access to the assembled matrixes: their projection on a modal base provides the generalized matrixes usable for a dynamic computation, with performances CPU much better than the direct use of the assembled matrixes. This method of reduction of model is described in documentations of reference [R5.06.01] and of use [U2.06.04].

3.2.3 To improve quality of the eigen modes

One draws the attention to the fact that **the quality of a modal computation depends above all on data quality on input and the physical modelization**. One can in particular quote:

- the choice of the boundary conditions: is they representative of reality? Their influence is strong on the result of computation;
- the smoothness of the mesh: a study of convergence of the mesh is necessary, as for any numerical study;
- the choice of the modelization: by structural elements (beam, shell,…) or in 3D? For example, for a hurled structure, a modelization out of beam will be generally better than a modelization 3D even with a good smoothness of mesh.

If the data input and the modelization are fixed, it is possible to improve "data-processing" quality of the result.
For that, the first computation by the method of subspace (operator `MODE_ITER_SIMULT`) gives a first estimate of the eigen modes (modal eigen frequencies, deformed shapes,…) of a structure. This first estimate is generally already good and satisfactory. For more complicated models, it however is advised to refine this estimate by the second computation by the method of the powers opposite (operator `MODE_ITER_INV`).
The sequence of the commands is then the following:


données d'entrée
ASSEMBLAGE
Calcul des matrices assemblées
INFO_MODE
Comptage a priori des fréquences propres (facultatif mais recommandé)
MODE_ITER_SIMULT mot-clé facteur CALC_FREQ
Calcul des modes propres : 1ère estimation
MODE_ITER_INV mot-clé facteur CALC_FREQ opérande FREQ alimentée par la 1ère estimation
Calcul des modes propres : affinement

Appear 3.2.3-a : Improvement of the quality of the eigen modes.

Example:

The first computation of modes on the tape [0; 2000] Hz with the command below:

```
mode1 = MODE_ITER_SIMULT (MATR_RIGI = k_asse,
                          MATR_MASS = m_asse,
                          CALC_FREQ =_F (
                                        OPTION = "BANDE",
                                        FREQ= (0. , 2000.),
                                        ),
                         )
```

give the following, viewable results in file MESSAGE :

```
------------------------------------------------------------------------

   FREQUENCIES CALCULEES INF. AND SUP. SONT:
      FREQ_INF:   4.65661E+01
      FREQ_SUP:   1.60171E+03


------------------------------------------------------------------------
       COMPUTATION MODAL:  METHODE Of ITERATION SIMULTANEE
                      METHODE OF SORENSEN

    NUMERO     FREQUENCY (HZ)     Error norm
         1       4.65661E+01        1.82405E-07
         2       2.91827E+02        3.47786E-09
         3       8.17182E+02        9.83625E-11
         4       1.60171E+03        4.31692E-11
 Error norm MOYENNE:   0.46506E-07


------------------------------------------------------------------------

          VERIFICATION A POSTERIORI DES MODES
   DANS the INTERVALLE (4.64496E+01, 1.60571E+03)
   IT Y A BIEN    4 FREQUENCY (S)
------------------------------------------------------------------------
```

One can then refine for example the first two eigen modes, while launching the second computation starting from the eigen frequencies previously computed:

```
mode2 = MODE_ITER_INV (MATR_RIGI = k_asse,
                       MATR_MASS = m_asse,
                       CALC_FREQ =_F (OPTION = "PROCHE",
                                      FREQ = (46.6,291.8),
                                     ),
                      )
```

what gives

```
------------------------------------------------------------------------
    COMPUTATION MODAL:  OPPOSITE ITERATION METHOD
                                                         INVERSE
NUMERO  FREQUENCY (HZ) DAMPING  NB_ITER  PRECISION   Error norm
    1    4.65661E+01    0.00000E+00      3    3.33067E-16     3.99228E-08
    2    2.91827E+02    0.00000E+00      3    2.22045E-16     1.23003E-09
```

One observes that the error norm is slightly improved (certainly slightly but it is here about a very simple case).

Note:
One can also automate the recovery of the eigen frequencies resulting from the first estimate to feed the second computation, thanks to the Python language:

```
# recovery of the list of the eigen frequencies estimated in the variable
Python f_estimation:
f_estimation = MODE1.LISTE_VARI_ACCES () ["FREQ"]

mode2 = MODE_ITER_INV (MATR_RIGI = k_asse,
                       MATR_MASS = m_asse,
                       CALC_FREQ =_F (OPTION = "PROCHE",
                                      FREQ = f_estimation,
                                     ),
                      )
```

3.2.4 Optimisation of performances CPU

3.2.4.1 Découpage of the frequential tape of search

If one seeks many eigen modes (either because the tape of search is very broad, or because the modal density is strong), the performances of modal computation will be better by cutting out the tape of total search $[f_{min}; f_{max}]$ in several (n) sub-bands: $[f_{min}; f_2]$, $[f_2; f_3]$,…, $[f_n; f_{max}]$. That is done thanks to operator `MACRO_MODE_MECA` by specifying the frequential spanning with key word `FREQ= (fmin, f2,…, fn, fmax)`. To define the sub-bands, the user can lean on counting *a priori* eigen frequencies provided by the operator `INFO_MODE` who can also function by sub-bands.

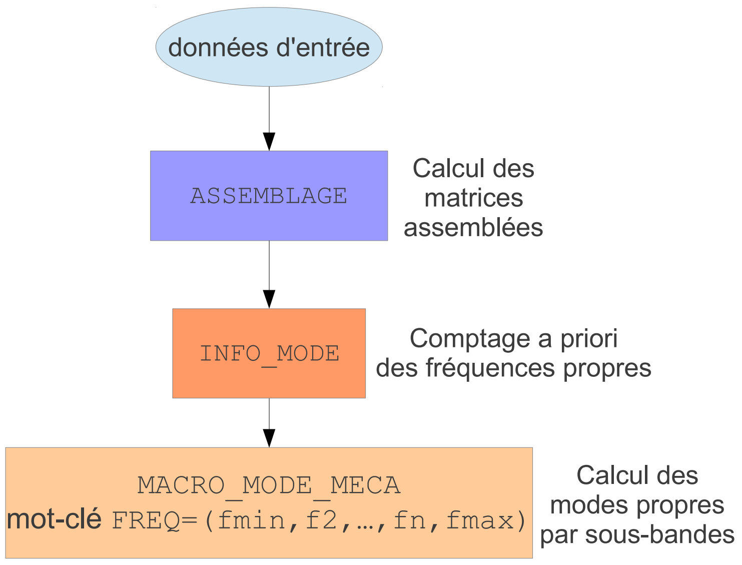

données d'entrée
ASSEMBLAGE
Calcul des matrices assemblées
INFO_MODE
Comptage a priori des fréquences propres
MACRO_MODE_MECA
mot-clé FREQ=(fmin,f2,…,fn,fmax)
Calcul des modes propres par sous-bandes

Figure 3.2.4.1-a : Computation of the eigen modes by spanning in sub-bands.

Example:

identical to paragraph 3.2.1 by cutting out the tape $[20\ ;300]\,Hz$ in three sub-bands:

```
ASSEMBLY (MODELE = model,
            CHAM_MATER = ch_mat,
            CARA_ELEM = cara_el,
            LOAD = c_limite,
            NUME_DDL = CO (“numbered”), # creation of a classification of
                                       # D.O.F.
            MATR_ASSE = (
                        _F (MATRICE= CO (“matr_k”), OPTION= “RIGI_MECA”),
                        _F (MATRICE= CO (“matr_m”), OPTION= “MASS_MECA”),
                       )
          );

nb_modes = INFO_MODE (MATR_RIGI = matr_k,
                       MATR_MASS = matr_m,
                       FREQ = (20. , 300.),
                     );

modes = MACRO_MODE_MECA (MATR_RIGI = matr_k,
                          MATR_MASS = matr_m,
                               CALC_FREQ =_F (FREQ = (20. , 100. , 200. ,
300.))
                        );
```

Note:

- *There is a gain in performance CPU even when the sub-bands are treated sequentially (what is the case by default). The implementation of parallelism (cf next paragraph) makes it possible to improve even more the performances.*
- *For optimal performances, it is advised to have the most balanced possible sub-bands (either with a number of modes sought by relatively uniform sub-band).*

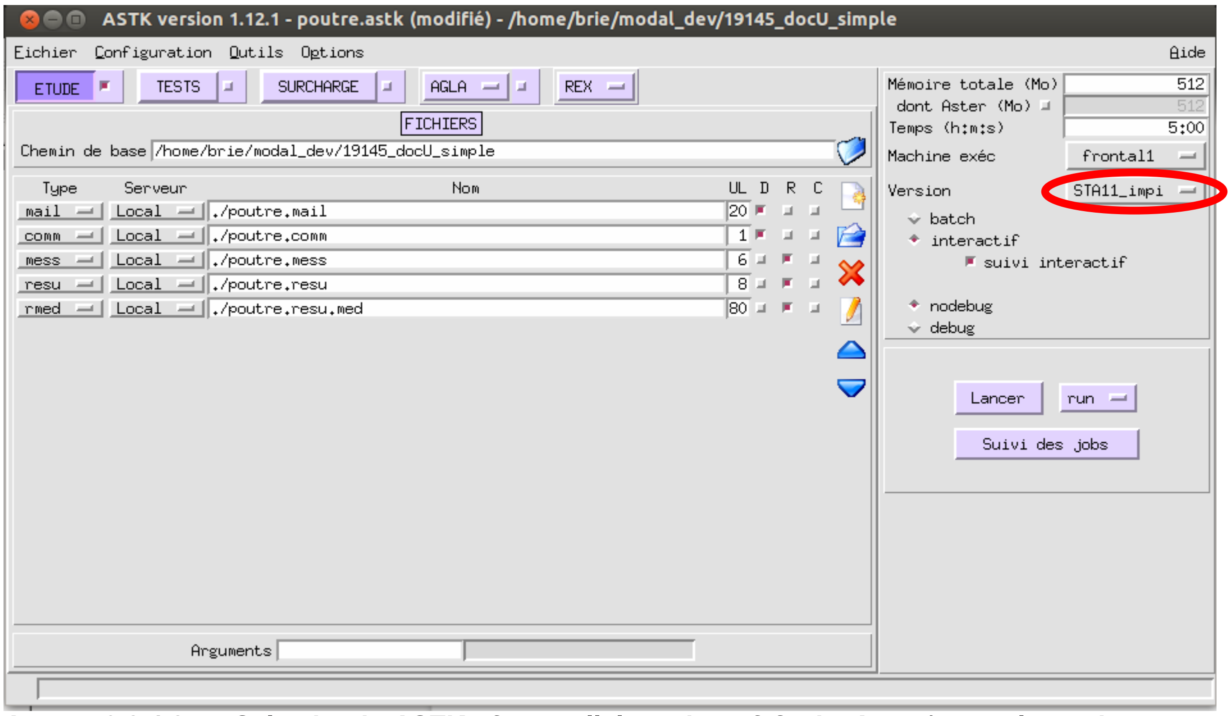
3.2.4.2 Parallelism

Pour modal computation, parallelism can be levelled in work two:

- parallelization of the modal computations carried out on each sub-band, in operators `INFO_MODE` and `MACRO_MODE_MECA` ;
- parallelism on the level of linear solver MUMPS, in operators `INFO_MODE`, `MODE_ITER_SIMULT`, `MODE_ITER_INV` and `MACRO_MODE_MECA`.

To implement parallelism, it is necessary:

- have a version of *Code_Aster* built with a parallel compiler (for example: OpenMPI,…). On the server centralized Aster4, parallel versions already exist: STAxx_impi;
- select in ASTK a parallel version of *Code_Aster* ;


ASTK version 1.12.1 - poutre.astk (modifié) - /home/brie/modal_dev/19145_docU_simple

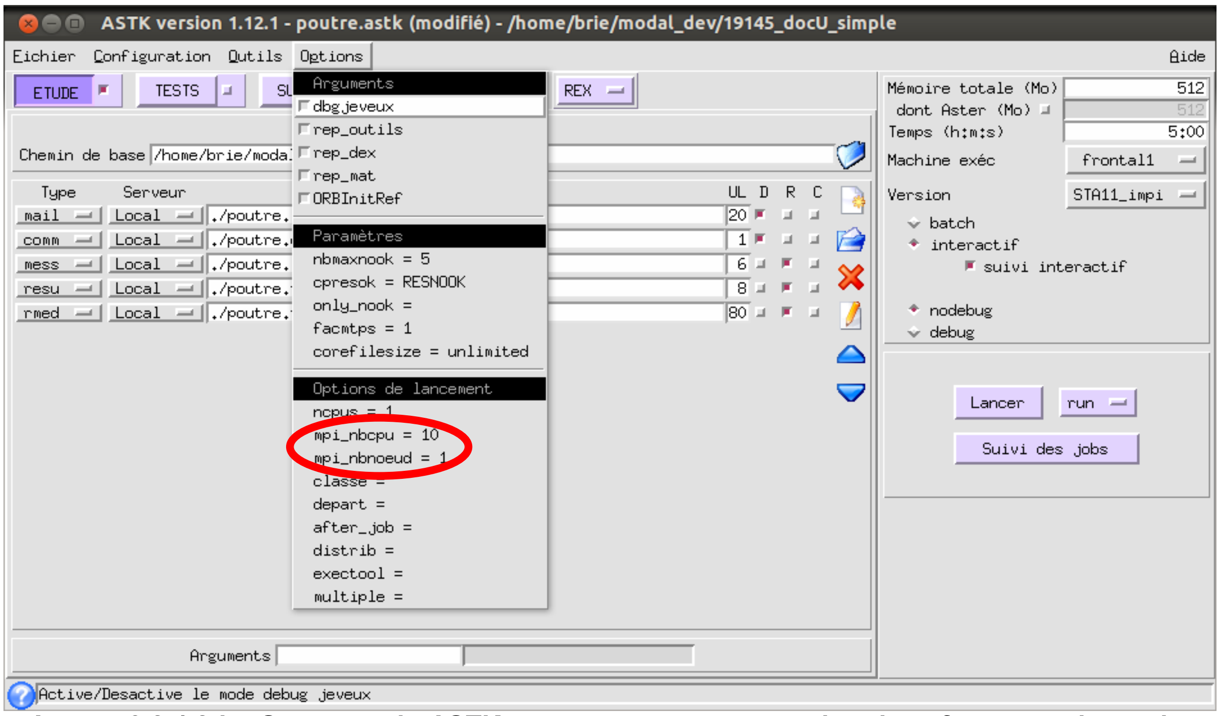
Chemin de base /home/brie/modal_dev/19145_docU_simple

Type	Serveur	Nom	UL
mail	Local	./poutre.mail	20
comm	Local	./poutre.comm	1
mess	Local	./poutre.mess	6
resu	Local	./poutre.resu	8
rmed	Local	./poutre.resu.med	80

Mémoire totale (Mo) 512
dont Aster (Mo) 512
Temps (h:m:s) 5:00
Machine exéc frontal1
Version STA11_impi
interactif
suivi interactif
nodebug

Appear 3.2.4.2-a : Selection in ASTK of a parallel version of *Code_Aster* (example on the server centralized Aster4).

- specify in ASTK the number of processors and nodes of computation to be exploited; it is necessary to use at least as many processors as of frequential sub-bands nonempty, and one advises to use a multiple number of processors amongst nonempty sub-bands (for example: if there are 5 nonempty sub-bands, use 10 or 15 or… processors);


ASTK version 1.12.1 - poutre.astk (modifié) - /home/brie/modal_dev/19145_docU_simple
Eichier Configuration Outils Options Aide
ETUDE TESTS SU REX
Arguments
Paramètres
Options de lancement
Lancer run
Suivi des jobs
Arguments

Chemin de base /home/brie/moda

Type	Serveur		UL	D	R	C
mail	Local	./poutre.	20			
comm	Local	./poutre.	1			
mess	Local	./poutre.	6			
resu	Local	./poutre.	8			
rmed	Local	./poutre.	80			

dbg jeveux
rep_outils
rep_dex
rep_mat
ORBInitRef

nbmaxnook = 5
cpresok = RESNOOK
only_nook =
facmtps = 1
corefilesize = unlimited

ncpus = 1
mpi_nbcpu = 10
mpi_nbnoeud = 1
classe =
depart =
after_job =
distrib =
exectool =
multiple =

Mémoire totale (Mo)	512
dont Aster (Mo)	512
Temps (h:m:s)	5:00
Machine exéc	frontal1
Version	STA11_impi

batch
interactif
suivi interactif
nodebug
debug

Active/Desactive le mode debug jeveux

Appear 3.2.4.2-b : Statement in ASTK amongst processors and nodes of computation to be exploited.

- In the command file *Code_Aster* :
  - to use the basic mode with `INFO_MODE` or `MACRO_MODE_MECA` (parallelization of the sub-bands only), there is nothing to make: the key words by default activate the parallelization of the sub-bands;
  - to use the advanced mode (parallelization of the sub-bands for `INFO_MODE` and `MACRO_MODE_MECA`, and of the linear solver for all the modal operators): use linear solver MUMPS (key word factor `SOLVER`, operand `METHODE='MUMPS'` ; one also recommends to parameterize operands `RENUM='QAMD'` and `GESTION_MEMOIRE='IN_CORE'`).

Example:

identical to the 3.2.4.1 3.2.4.1 by paralleling at the same time computations on the sub-bands and the linear solver:

```
modes = MACRO_MODE_MECA (MATR_RIGI = matr_k,
                         MATR_MASS = matr_m,
                            CALC_FREQ = F (FREQ = (20. , 100. , 200. ,
300.)),
                         NIVEAU_PARALLELISME = "COMPLET",
                         SOLVER = F (METHODE = "MUMPS",
                                 RENUM = "QAMD",
                                 GESTION_MEMOIRE = "IN_CORE"),
                      );
```

Note:

For optimal performances, it is advised to have the most balanced possible sub-bands (i.e.: with a number of modes sought by relatively uniform sub-band).

The implementation of parallelism is presented in a more detailed way in generic documentation [U2.08.06] and documentations of use of `INFO_MODE` [U4.52.01] and `MACRO_MODE_MECA` [U4.52.02].

3.2.4.3 Reduction of model: computation by Lorsque

substructuring the digital model comprises a high number of degrees of freedom or that the studied structure is an assembly of components with a grid separately, one can use methods of reduction of model per substructuring, which rest on a geometrical partitioning of total structure. On great models, these methods present better performances CPU that a direct computation.

- Dynamic substructuring

  Cette method has a very general field of application. Documentation [U2.07.05] details its implementation.

- Cyclic substructuring

  Cette method has a field of application much more restrictive than the preceding one: it makes it possible to treat only structures with cyclic repetitivity (for example: coil aubagée,...). Benchmark SDLV301 gives an example of implementation.

4 Parameters contained in a modal computation result

the execution of the one of the operators of modal computation is accompanied by the automatic printing of certain parameters in file `RESULTAT` :

```
------------------------------------------------------------------------
LE NOMBRE OF D.O.F.

   TOTAL IS:                        234

   OF LAGRANGE IS:                  120

LE NOMBRE OF D.O.F. CREDITS IS:         54
------------------------------------------------------------------------
THE OPTION SELECTED IS: CENTRE

THE VALEUR OF DECALAGE IN FREQUENCY IS:  5.00000E+01
------------------------------------------------------------------------

 INFORMATION ON LE COMPUTATION REQUIRES:
 NOMBRE OF MODES SEARCH     :                     1

 THE DIMENSION OF SPACE REDUIT IS:                     0
 IT IS LOWER THAN THE NOMBRE OF MODES, ONE TAKES IT EQUALIZES A4




     =============================================
     =       METHODE OF SORENSEN (CODE ARPACK)   =
     =       VERSION:  2.4                       =
     =          DATE:  07/31/96                  =
     =============================================
     NOMBRE OF RESTARTINGS                      =     2
     NOMBRE OF PRODUCTS OP*X                     =     7
     NOMBRE OF PRODUCTS B*X                      =    20
     NOMBRE OF REORTHOGONALISATIONS  (STAGE 1)   =     6
     NOMBRE OF REORTHOGONALISATIONS  (STAGE 2)   =     0
     NOMBRE OF RESTARTINGS OF THE A NULL V0       =     0

------------------------------------------------------------------------
```

```
   FREQUENCIES CALCULEES INF. AND SUP. SONT:
      FREQ_INF:   5.22037E+01
      FREQ_SUP:   5.22037E+01


------------------------------------------------------------------------
       COMPUTATION MODAL:  METHODE OF ITERATION SIMULTANEE
                      METHODE OF SORENSEN
                                               eigen frequencies
    NUMERO   FREQUENCY (HZ)    Error norm
         2     5.22037E+01       7.02498E-10
         3     6.74211E+01       9.12843E-10
 Error norm MOYENNE:   0.80767E-09
  position of the mode in the total
  spectrum
------------------------------------------------------------------------

          VERIFICATION A POSTERIORI DES MODES

   DANS the INTERVALLE (5.20730E+01, 5.23340E+01)
   IT Y A BIEN     1 FREQUENCY (S)
------------------------------------------------------------------------
```

If the computed modes are complex, there is in more one column giving modal dampings:

```
------------------------------------------------------------------------
LE NOMBRE OF D.O.F.

   TOTAL IS:                        74

   OF LAGRANGE IS:                  44

LE NOMBRE OF D.O.F. CREDITS IS:          8
------------------------------------------------------------------------
 INFORMATION ON LE COMPUTATION REQUIRES:
 NOMBRE OF MODES SEARCH      :                       5


  the dealt with problem being quadratic, one doubles the space of search

    Méthode QZ in MODE_ITER_SIMULT: One finds a number of eigenvalues
    17 different amongst ddls physical credits 8!

 your problem is strongly damped.
 value (S) clean (S) real (S)                      :  14
 value (S) clean (S) complex (S) with combined:  10
 value (S) clean (S) complex (S) without combined:  0
       COMPUTATION MODAL:  METHODE GLOBALE OF TYPE QR
eigen frequencies (damped)    ALGORITHME QZ_SIMPLE

    NUMERO   FREQUENCY (HZ)    DAMPING     Error norm
         1     5.52718E+00       8.68241E-03        4.00918E-13
         2     1.08852E+01       1.71010E-02        7.31808E-14
         3     1.59105E+01       2.50000E-02        5.40182E-14
         4     2.04500E+01       3.21394E-02        4.03817E-14
         5     2.43661E+01       3.83022E-02        3.48265E-14
 Error norm MOYENNE:   0.12067E-12
                                           modal dampings
 Attention: for time, there is no checking of the type STURM (counting of
the good number of the calculated eigenvalues)
 when one is in the complex plane:
       modal problem generalized with MATR_RIGI complex,
    or modal problem generalized with matrix (S) asymmetric (S),
    or quadratic modal problem (préence of key word MATR_AMOR).

------------------------------------------------------------------------

          VERIFICATION A POSTERIORI OF THE MODES

------------------------------------------------------------------------
```

Moreover, the data-processing data structure produced during a modal computation can contain the following parameters:

Heading of the parameter in *Code_Aster*	Définition
FREQ	Eigenfrequency (damped, if necessary)
AMOR_GENE	Modal damping generalized
AMOR_REDUIT	Modal damping reduces
FACT_PARTICI_D* (* = X or Y or Z)	Participation factor of the mode in the direction D*
MASS_EFFE_D* (* = X or Y or Z)	Masse modal effective in the direction D*
MASS_EFFE_UN_D* (* = X or Y or Z)	Masse modal effective unit in the direction D*
MASS_GENE	Generalized mass of clean
mode	OMEGA2 Pulsation (deadened, if necessary) to square
RIGI_GENE	Stiffness generalized of the mode

Tableau 4.1 : list modal parameters.

These parameters are mathematically defined in documentation of reference [R5.01.03]. The user has there access by printing the contents of structure of data with operator IMPR_RESU to the FORMAT='RESULTAT' with option TOUT_PARA='OUI'.

5 Postprocessings of the modal

5.1 eigen modes

Visualisation Les deformed computed by one of the methods described previously can be exported in various formats in order to be visualized in platforms of mechanical computation: format MED for the Salome platform, format UNV,…

the user can thus graphically characterize the calculated modes: mode of bending? mode in a given plane? local mode? etc

Exemple:

printing with format MED.

```
modes =…          # computation by one of methods previously described
```

```
IMPR_RESU (FORMAT = "MED",
           RESU=_F (RESULTAT = modes));
```

One can then open the file created in the Salomé platform to visualize the modal deformed shape, to animate it,…

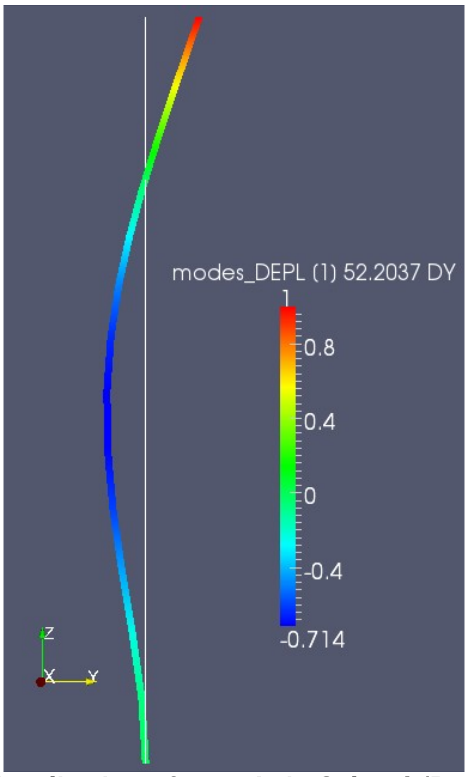


Figure 5.1-a : Visualization of a mode in Salomé (ParaVis modulus): here mode of bending of order 2 of a beam.

5.2 Normalization of the modal

deformed Les modes are defined except for a multiplicative factor (cf formulation of the modal problem in paragraph 1).

By defaults, the modes computed by operators `CALC_MODAL`, `MODE_ITER_SIMULT` and `MODE_ITER_INV` are normalized so that the largest physical component is equal to 1. The user can modify this normalization thanks to the operator `NORM_MODE` [U4.52.11], who also calculates or updates the following modal parameters, which depend on the selected normalization: `FACT_PARTICI_D*`, `MASS_GENE` and `RIGI_GENE`. He also enriches structure of data with the parameters `MASS_EFFE_UN_D*` (which are them independent of the normalization). These parameters (definite in paragraph 4) can be useful in particular to eliminate from a modal base certain nondesired modes (cf paragraph 5.3).

Example:

normalizes compared to the mass.

```
modes =…        # computation by one of the methods previously described

modes = NORM_MODE (reuse = modes,
                   MODE = modes,
                   NORM = "MASS_GENE");
```

Note:

if the user calculates the modes with operator `MACRO_MODE_MECA` , it can choose the norm directly inside this operator, with the key word factor `NORM_MODE` .

5.3 Pattern matching of the modes according to a criterion

Dans the prospect for a computation of transient response for example, the user can choose to preserve in its modal base of projection, only certain modes considered to be important in the dynamic response or filling a given criterion. That is done thanks to the operator `EXTR_MODE` [U4.52.12] who allows to filter the modes according to various options: starting from their number in the total spectrum, of their generalized mass, etc

Exemple:

elimination of the modes of which the unit effective mass in the direction DX is lower than 5 %, and display in file `RESULTAT` of the office plurality of the unit effective masses of the preserved modes.

```
modes =…        # computation by one of the methods previously described

modes_f = EXTR_MODE (FILTRE_MODE =_F (MODE = modes,
                                      CRIT_EXTR = "MASS_EFFE_UN",
                                      SEUIL_X = 0.05),
                     =_F PRINTING (CRIT_EXTR = "MASS_EFFE_UN",
                                     CUMUL = "OUI"),
                    );
```

Note:

if the user calculates the modes with operator `MACRO_MODE_MECA` , it can produce this filter directly inside this operator, with the key word factor `FILTRE_MODE` .