Note of computation to Summarized

The purpose of this documentation is to present a methodological guide for a nonlinear analysis of buckling of a structure. Mainly two features of Code_Aster there are approached:

• analysis of buckling linear, known as of Eulerian, through MODE_ITER_SIMULT, (option TYPE_RESPU="MODE_FLAMB"),
• the computation of the quasistatic evolution (operator STAT_NON_LINE) of the structure which has geometric nonlinearities and behavioral, of which one seeks instability (option CRIT_STAB), a boundary point, even the post-critical response.

The first stage is, generally, a computation of buckling of Eulerian, which will make it possible to know buckling modes and the corresponding critical loads. From the point of view of the originator, the knowledge of the first mode and its critical load is often sufficient, in order to be defined a margin of operation compared to the imposed loading: the multiplying coefficient between the imposed loading and the weakest critical load gives the safety margin.

Remarks

the knowledge of the first mode of buckling can also be used as indication to optimize the management of nonlinear incremental computation carried out thereafter. Indeed, with the approach of the critical load, one can then decide to modify control or to reduce time step, to even increase the number of iterations of checking the equilibrium in the method of residue, with each step of load.
The pace of the mode of buckling of Eulerian can as serve an initial geometrical default to impose on the structure, in order to make sure, amongst other things, as incremental nonlinear computation will fork this mode of course.

The analysis of Eulerian being per linear definition, it does not make it possible to take into account inelastic behavior models, of the contact or the following aspect of the forces. It is then necessary to do a nonlinear calculation, which into quasi-static will lean on command STAT_NON_LINE of Code_Aster. It is the incremental classical method by residue in equilibrium.

The particular points of its use will be approached thereafter, in particular the use of the nonlinear analysis of stability with the key word CRIT_STAB (which is also available in DYNA_NON_LINE for the dynamic studies).
1 Buckling of Eulerian

buckling modes within the meaning of Eulerian [bib5] analyzes. The computation can be made by the operator be made resolution of problems to eigenvalues MODE_ITER_SIMULT (or MODE_ITER_INV). In the frame of buckling, there is following typical syntax:

\[
\text{MODP1} = \text{MODE\_ITER\_SIMULT} \ (\text{MATR\_RIGI} = \text{RAMEP1}, \\
\quad \text{MATR\_RIGI\_GEOM} = \text{RAGEP1}, \\
\quad \text{TYPE\_RESU} = \text{"MODE\_FLAMB"}, \\
\quad \text{CALC\_FREQ} = _{F} \ (\text{OPTION} = \text{"TAPE"}, \\
\quad \text{CHAR\_CRIT} = (-2.4, - 2.2), \\
\quad \text{DIM\_SOUS\_ESPACE} = 80, \\
\quad \text{NMAX\_ITER\_SOREN} = 80,))
\]

the argument of key word MATR\_RIGI must be the stiffness matrix known as material, whereas key word MATR\_RIGI\_GEOM expects the geometrical stiffness matrix. If operator MODE\_ITER\_INV had been employed, the arguments of key keys MATR\_RIGI and MATR\_RIGI\_GEOM would be the same ones.

For recall, buckling modes are the eigen modes of the problem to the eigenvalues according to:

\[(K + \mu K_g)x = 0 \iff K x = \lambda K_g x\]

with:

- \(K\) : stiffness matrix
- \(K_g\) : geometrical stiffness matrix
- \(\lambda\) : eigenvalue \((\lambda = -\mu\) with \(\mu\) multiplying coefficient of the loading)

the material stiffness (or elastic) is calculated with option “RIGI\_MECA” of CALC\_MATR\_ELEM. The geometrical stiffness is calculated starting from the stress field solution of the linear problem (option “RIGI\_GEOM” of CALC\_MATR\_ELEM). It is thus necessary to have carried out a static linear computation prior to the use of MODE\_ITER\_SIMULT for buckling.

If the loading is composed of a fixed part (not controlled) and of a variable part, the multiplying coefficient of the loading should not, of course, relate that to the variable part. The contribution of the other part of the loading is found in the first member. Let us note \(f_c\) the fixed loading and \(f_v\) the controlled loading (proportional to \(\mu\)). The problem with the eigenvalues becomes:

\[(K + K_g(f_c + \mu f_v))x = 0 \iff (K + K_g(f_c))x = \lambda K_g(f_v)x\]

with:

- \(K\) : material stiffness matrix
- \(K_g(f_c)\) : geometrical stiffness matrix for the loading not controlled
- \(K_g(f_v)\) : geometrical stiffness matrix for the variable loading
- \(\lambda\) : eigenvalue \(\lambda = -\mu\)
In this case, it is thus necessary to solve two preliminary linear elastic problems, to be able to calculate the two different geometrical stiffness matrixes. In order to be exhaustive, the presentation will relate to a structure subjected to the displacements imposed as well as forces, which will be the combination of a fixed loading and a variable loading that one will control with a coefficient growing being able to lead to buckling.

1.1 Stage 1: Computation (S) linear (S) preliminary (S)

One will make use of `MECA_STATIQUE`. The structure, with a grid in elements of type shell (elements of voluminal the shells type [bib3]), is subjected to boundary conditions of Dirichlet (`CONDLIM`) and Neumann. These last break up into:

- `WEIGHED`: field of gravity,
- `PRESPH`: field of pressure imposed not controlled,
- `PRESPS1`: field of variable pressure imposed.

For the analysis of buckling, it is necessary to separate the constant forces from those which are variable (controlled by a coefficient). One will thus do two linear static calculations. The first will be the case of structure subjected to imposed displacements and with the constant forces, the second will see structure subjected to imposed displacements and the variable forces.

Controlled loading:

```latex
RESC11P1 = MECA_STATIQUE (MODELS = MODELS,
   CHAM_MATER = CHMAT,
   CARA_ELEM = CARAELEM,
   EXCIT = (_F (CHARGE = CONDLIM,),
            _F (CHARGE = PRESPS1,),),
   OPTION = "SIEF_ELGA",)
```

Loading not controlled:

```latex
RESC12P1 = MECA_STATIQUE (MODELS = MODELS,
   CHAM_MATER = CHMAT,
   CARA_ELEM = CARAELEM,
   EXCIT = (_F (CHARGE = CONDLIM,),
            _F (CHARGE = WEIGHED,),
            _F (CHARGE = PRESPH,),),
   OPTION = "SIEF_ELGA",)
```

One will use the stress field to compute: the associated stiffness matrixes geometrical, for the two loadings:

```latex
SIGC11P1 = CREA_CHAMP (TYPE_CHAM = "ELGA_SIEF_R",
   OPERATION = "EXTR",
   RESULTAT = RESC11P1,
   NOM_CHAM = "SIEF_ELGA",
   TYPE_MAXI = "MINI",
   TYPE_RESU = "VALE",)
#
REGC11P1 = CALC_MATR_ELEM (OPTION = "RIGI_GEOM",
   MODELS = MODELS,
   CARA_ELEM = CARAELEM,
   SIEF_ELGA = SIGC11P1,)
```

`REGC11P1` is thus the geometrical stiffness matrix associated with the variable case of loading (`PRESPS1`).
One calculates, in the same way, the geometrical stiffness matrix for the constant loading (WEIGHED and PRESPH), from RESC12P1:

SIGC12P1 = CREA_CHAMP (TYPE_CHAM = "ELGA_SIEF_R", 
OPERATION = "EXTR", 
RESULTAT = RESC12P1, 
NOM_CHAM = "SIEF_ELGA", 
TYPE_MAXI = "MINI", 
TYPE_RESU = "VALE",)
#
REGC12P1 = CALC_MATR_ELEM (OPTION = "RIGI_GEOM", 
MODELS = MODELS, 
CARA_ELEM = CARAELEM, 
SIEF_ELGA = SIGC12P1,)

It remains to calculate the material stiffness matrix for the total loading:

REMEP1 = CALC_MATR_ELEM (OPTION = "RIGI_MECA", 
MODELS = MODELS, 
CHAM_MATER = CHMAT, 
CARA_ELEM = CARAELEM, 
CHARGE = (CONDLIM, WEIGHED, 
PRESPH, PRESPS1,))

All the elementary matrixes are calculated, the following stage is thus their assembly:

NUP1 = NUME_DDL (MATR_RIGI = REMEP1,)
#
RAMC1P1 = ASSE_MATRICE (MATR_ELEM = REMEP1, 
NUME_DDL = NUP1,)
#
RAGEP1 = ASSE_MATRICE (MATR_ELEM = REGC11P1, 
NUME_DDL = NUP1,)
#
RAGC12P1 = ASSE_MATRICE (MATR_ELEM = REGC12P1, 
NUME_DDL = NUP1,)

One adds then the stiffness matrixes material (RAMC1P1) and geometrical (RAGC12P1) corresponding to the case of constant loading:

RAMEP1 = COMB_MATR_ASSE (COMB_R = (_F (MATR_ASSE = RAMC1P1, 
COEF_R = 1.0,)), 
_F (MATR_ASSE = RAGC12P1, 
COEF_R = 1.0,)),)

the two matrixes necessary to the computation of buckling modes are thus built.

1.2 Stage 2: Computation of the modes of Eulerian

It can be useful to make tests of STURM TYPE (operator INFO_MODE) on the interval of search on which one wants to find the cases of buckling. Thus, that will make it possible to optimize the size of the interval and to control the good progress of later modal computation since one will know the number of existing modes in advance. Syntax is:

INFO_MODE (MATR_RIGI = RAMEP1, 
MATR_RIGI_GEOM = RAGEP1, 
TYPE_MODE = "MODE_FLAMB", 
CHAR_CRIT = (- 2.4, - 2.2) )

Once the interval of search of critical load of buckling chosen, one can then implement MODE_ITER_SIMULT as follows:

MODP1 = MODE_ITER_SIMULT (MATR_RIGI = RAMEP1,
MATR_RIGI_GEOM = RAGEP1,
TYPE_RESU = "MODE_FLAMB",
CALC_FREQ = _F (OPTION = "TAPE",
    CHAR_CRIT = (-2.4, - 2.2,)
    DIM_SOUS_ESPACE = 80,
    NMAX_ITER_SOREN = 80,)
)

Remark

If the algorithm does not converge or if the number of modes is not that predicted by INFO_MODE, it can be useful to increase the values of DIM_SOUS_ESPACE and NMAX_ITER_SOREN.

One normalizes the modes [bib6], only while making use of the degrees of freedom of translation:

MODP1 = NORM_MODE (reuse = MODP1
    MODE = MODP1,
    NORM = "TRAN",)

the modes can then be post-treaties.

Remarks

It is essential to check that the geometrical stiffness of the selected model is well an option available in Code_Aster (for example, it is not the case of the DKT). A finer discretization leads normally to a fall of the critical loads. The discretization must be ready to collect buckling modes, knowing that these modes can generate localised strains (folds). The computation preliminary of the dynamic modes can constitute a first indication on the quality of the mesh, although these modes can be very different from buckling modes. The critical loads of the various modes are proportional to the Young modulus $E$. With this method, one cannot take account of the following character of the forces. Indeed, the geometrical computation of stiffness made with option RIGI_GEOM of CALC_MATR_ELEM is based on the assumption that all the forces imposed on the mechanical problem are of type forces dead. If one wants to take account of the following character of certain requests, it is obligatorily necessary to use nonlinear operators STAT_NON_LINE or DYNA_NON_LINE, with the key word CRIT_STAB, as one will see it in the continuation of this document.
2 Quasi-static nonlinear study of structure

This stage is justified if the structure has strong non-linearities, whose analysis of Eulerian cannot take account of the critical loads and modes associated during incremental computation nonlinear. That results in an analysis of the Eulerian type on the reactualized stiffness matrixes. This kind of analysis is often done on a structure without initial default.

In addition, one can the model take account of defaults introduced on perfect, in order to “force” the bifurcation in solution and to make follow-up of branch to analyze the post-critical response.

Obviously, this follow-up of post-critical solution can be initiated by the analysis with the eigenvalues on reactualized stiffness matrixes, in particular for detecting the bifurcation well and defining the default then introduced while being based on the mode of buckling observed.

2.1 Stability analyzes on reactualized stiffness matrixes

That it is into quasistatic (operator \texttt{STAT\_NON\_LINE}) or in dynamics (operator \texttt{DYNA\_NON\_LINE}), \textit{Code\_Aster} makes it possible to carry out incremental analyses of stability within the meaning of buckling on the current stiffness matrixes. These stages of computation are managed by the key word factor \texttt{CRIT\_STAB} with the option \texttt{TYPE = “FLAMBEMENT”} (cf \cite{U4.51.03} and \cite{U4.53.01}).

Remarks for the transient analysis

\texttt{option CRIT\_STAB acts as for the quasistatic case: one always carries out an analysis of buckling (thus basing itself only on the study of the stiffness matrixes), not a dynamic analysis of instability (damping becoming negative). For the modelization coupled fluid-structure \(u, p, \phi\) \cite{R4.02.02}, it is necessary to modify the assembled stiffness matrix (as well as the geometrical stiffness when it is used). For that, it is necessary to inform the key word following, under \texttt{CRIT\_STAB}:

\begin{itemize}
  \item \texttt{MODI\_RIGI = “OUI’},
  \item \texttt{DDL\_EXCLUS= (”PHI”, “NEAR”, “DH”).}
\end{itemize}

The list of the excluded degrees of freedom must comprise all the types of degrees of freedom related to the fluid model: in the example of benchmark FDNV100 there is thus the potential \texttt{PHI}, the pressure \texttt{NEAR} and vertical displacement on the level of free face \texttt{DH}. If this processing is not made, then the call to \texttt{CRIT\_STAB} will plant due to singular matrix and no strategy of shift could surmount that.

Into quasi-static, this problem does not arise because the modelization coupled fluid-structure then does not have a meaning.

This key word makes it possible to start computation, at the end of each increment of time, of a criterion of stability. This criterion is useful to detect, during the loading, the point from which one loses stability (by buckling for example).

This criterion is calculated in the following way : at the end of one time step, in small disturbances, one solves

\[
det(K_T^T - \lambda.K^g) = 0.\]

\(K_T^T\) is the coherent tangent matrix at this time. \(K^g\) is the geometrical stiffness matrix, calculated starting from the stress field at this time.
In practice, the loading is unstable if $|\lambda| < 1$ (in fact $-1 < \lambda < 0$). One calculates the eigenvalues by the method of Sorensen (cf MODE_ITER_SIMULT [U4.52.03]). This can be expensive enough for the problems of big size.

Key word CHAR_CRIT makes it possible to save time by making only one test of Sturm type in the provided bande de fréquence. If at least a frequency is found, then one calculates really the values of the critical loads in this interval.

For large displacements and the large deformations, one solves $det\left| K^T - \lambda I \right| = 0$ because $K^T$ contains then $K^g$. The criterion is then a criterion of instability: when $\lambda$ changes sign (thus passes by 0) the loading is unstable.

**Remarks concerning the geometrical stiffness matrix**

Certain finite elements as the DKT cannot calculate this matrix, even in small disturbances. The analysis of stability will be more precise if one has the geometrical stiffness matrix: indeed, in its absence, certain instabilities cannot be detected like, for example, in the purely elastic cases. Between sufficiently realistic models, it is thus necessary to privilege the use, for precise analyses of stability, models allowing the computation of this geometrical stiffness. For example for a model of type shell, by means of elements COQUE_3D rather than of the DKT.

Key word NB_FREQ indicates the number of critical loads to calculating. Often the first is enough but there can be multiple modes.

One stores the eigen mode corresponding to the smallest critical load (in absolute value) in the object result, under name MODE_FLAMB. This eigen mode can be extracted and visualized (like a field of displacements or a classical eigen mode). It is standardized to 1 on the largest component of displacement.

In practice, in order to limit the overcost of computation, one advises to optimize the calls to CRIT_STAB. One can for that use keywords LIST_INST/INST/PAS_CALCUL in the keyword factor CRIT_STAB. One can thus specify with which time step one will calculate buckling modes.

In complement, it is judicious to use CRIT_STAB only on the time intervals where one suspects the possibility of instabilities.

Lastly, if a very good evaluating of the critical loads is wanted, it is advisable to refine well time step with the approach of this zone. This advice is all the more relevant into quasi-static because the user then resorts often to time step the larger than in dynamics.

It is possible to properly stop (bases reusable in poursuite) a computation with STAT_NON_LINE or DYNA_NON_LINE during the detection of an instability. It is not operation by default where the code will try to continue to solve the problem, and if there is convergence, then that means that one succeeded in following one of the branches of solution.

To manage this stop (cf benchmark ERREU10), it is necessary as a preliminary to have used DEFI_LIST_INST with the following arguments:

```
ECHEC=_F (EVENEMENT=' INSTABILITE', ACTION=' ARRET'),
```

Which means that in the event of event of type instability, the started action will be the stop.

Under CRIT_STAB, the optional options following allow to control these stopping criteria:

- **PREC_INSTAB** to define the accuracy (adimensional) stopping criteria,
- **SIGNE** to specify the breaking values to consider.
The second key word is useful only when the geometrical stiffness matrix is used. By default one considers the solution as being unstable if the critical load becomes ranging between 1 and -1, but one can, if need be, to take into account only the positive or negative part of this interval.

Without geometrical stiffness matrix, instability will be detected when an eigenvalue of the assembled total stiffness matrix, is:

• will tend towards 0 (with a relative accuracy given by \texttt{PREC\_INSTAB}),
• will change sign.

For example, in the case of a tank filled with water under seisme, one can begin incremental or transitory computation with a critical load being worth 0.8 (analyzes with geometrical stiffness): what means that the tank would flame if one imposed a depression being worth 0.8 times the imposed hydrostatic pressure (the positive value of the critical load corresponds to an inversion of meaning of the loading considered). Thus if nothing is specified, computation would be considered unstable and would stop. Like, in this case, one makes the assumption then that there will not be depressurization (for example by draining), one clears the interval $[0, 1]$ in the analysis of stability. Thus the problem will become unstable if the critical load reached the interval $[−1, 0]$.

For the analyses in monotonous evolution this kind of reasoning is easily conceived, which is well the case for the static part of the loading, but for the dynamic part, the loading being cyclic, and except having specific information, it is surer and more conservative to keep the option by default and thus to consider unstable structure if the critical load becomes lower than 1 in absolute value.

During a stop on instability, computation will stop by closing the base properly: the user will be able to exploit it in poursuite.

### 2.2 Followed by unstable solution

For the study of a structure potentially unstable or likely to know a boundary point, which is thus likely to meet a bifurcation in solution during the evolution of the loading, it is often useful to be able to choose a branch of particular solution (often the physical solution when it is a priori defined without ambiguities). For that, the user can have to introduce an initial default which "will force" structure to fork on the branch of particular solution.

Several methods exist to define this default.

One the most adapted of is pre-to slightly deform structure according to the pace of the mode of buckling corresponding to the branch than one wants to follow. The amplitude of this predeformation must be low, for example less 1/10ème of the thickness for a thin structure. The ideal being to find the default minimal which is compatible with a satisfactory performance of the algorithm of residue in equilibrium. Indeed, a too weak default can involve a difficulty of convergence of the residue, mainly in the case of a control in force. This initial default can extremely judiciously be built starting from the mode of instability calculated with option \texttt{CRIT\_STAB} of \texttt{STAT\_NON\_LINE} / \texttt{DYNA\_NON\_LINE} (cf preceding paragraph). This mode then takes account of all the non-linearities introduced into the model complete. The more economic alternative is to use the mode of buckling of Eulerian, but which corresponds to the linear case.

The geometrical default can also be defined by experimental measurements of the real part whose geometry could not be perfect.

The default can also take the shape of a disturbance of the loading (misalignment, addition of a loading located,…) or of the mechanical characteristics of the material (local weakening of the Young modulus, for example). He can nevertheless be then more difficult to adapt the default to the mode of wished buckling, especially if the structure presents relatively nearby modes.

Notice

\textit{In certain cases, even on the nondisturbed problem, the loading is such as it causes the desired bifurcation.}

One of the other particular points, related to instability, is the choice of the technique of control of algorithm \texttt{STAT\_NON\_LINE}. Indeed, classical control in force is not adapted any more because it
cannot collect an unstable branch of solution. In the same way, with the approach of a boundary point, convergence with control in force will become increasingly difficult, the tangent stiffness matrix becoming singular. It is then necessary to reduce the increment of load and to increase the maximum number of iteration to continue computation.

One can also serve as the possibility of stopping properly in the event of instability (cf preceding paragraph) to manage in pursue the bifurcation on the branch of solution chosen by initiating the continuation of computation by a disturbance according to this mode of instability.

There exist techniques of control [bib9] making it possible to circumvent these numerical difficulties. Among the methods suggested by Code_Aster, that called by length of arc [bib12] (option TYPE='LONG ARC' of the key word CONTROL in STAT_NON_LINE), which is adapted for instabilities of type buckling, in the possible case of "soft" snap-backs [bib13]. In the case of more brutal snap-backs, Crisfield proposes an alternative [bib13], nonavailable in Code_Aster. Other methods exist, like that of Riks [bib14] (nonavailable either), which treats also the dynamic case.

If one wants only to obtain the point limits, including with a good accuracy, a control in loading can be enough, on condition that managing well the parameters of step of increment of load (think of using command DEFI_LIST_INST) and of authorized maximum number of iterations (ITER_GLOB_MAXI of CONVERGENCE). It can also be useful, with the approach of the boundary point, not to more use the tangent matrix reactualized for the solver, since it is quasi-singular. One can then be satisfied not to reactualize this matrix with each computation (parameters REAC_INCR and REAC_ITER) or, in worst of the cases, to adopt the basic elastic matrix (PREDICTION='ELASTIQUE' and MATRICE='ELASTIQUE' of key word NEWTON).

Here an example of use of STAT_NON_LINE for an elastoplastic computation in large displacements ([bib4] for the elements employed, which are of voluminal shells type), with control in forces:

```
RESU = STAT_NON_LINE (…
    EXCIT = ( _F (CHARGE = CONDLIM,
               TYPE_CHARGE = "FIXE_CSTE"),
              _F (CHARGE = WEIGHED,
               TYPE_CHARGE = "FIXE_CSTE"),
              _F (CHARGE = PRESH,
               FONC_MULT = FONCMUL2,
               TYPE_CHARGE = "SUIV"),
              _F (CHARGE = PRESPS1,
               FONC_MULT = FONCMUL,
               TYPE_CHARGE = "SUIV"),),
    …)
```

Remarks

- One uses the tangent matrix reactualized with each computation, by authorizing the undercutting of the step of load.
- The imposed pressures are following forces (TYPE_CHARGE='SUIV').
- In the case of a modelization in solid elements, the strain tensor recommended in large displacements is "SIMO_MIEHE".

If one wants to replace control in force by a method by length of arc, it is enough to add:

```
RESU = STAT_NON_LINE (…
    CONTROL = _F (GROUP_NO = "G",
                 TYPE = "LONG_ARC",
                 NOM_CMP = ("DY"),
                 COEF_MULT = 7. ),
    …)
```

Remarks

- In Code_Aster, one cannot control follower forces.
For control by length of arc, it, in general, is recommended that GROUP_NO contains all structure.

To finish, let us quote two articles of Crisfield which give a good general vision of the problems and methods involved in nonlinear computations being able to present various types of instabilities ([bib15] and [bib11]).

Documentation [U2.06.11] shows an example of use of CRIT_STAB for the study of the behavior of a metal tank.

Some benchmarks of Code_Aster dealing with buckling:

- Modes of Eulerian:
  - sdls504
  - sdls505
  - ssl1103
  - ssl1105
  - ssl403
  - ssl404
  - ssla110

- Modes of Eulerian and nonlinear computation:
  - ssn123

- nonlinear Modes (CRIT_STAB):
  - sdnv106 (presents in more MODE_VIBR for computation of the oscillatory modes on reactualized stiffness)
  - ssl1105
  - ssn126
  - ssnp306

- nonlinear Computation:
  - ssnl502
  - ssnp305: computation until a snap-through
3 Bibliography

1) External forces of pressure in large displacements [R3.03.04].
2) Following pressure for the voluminal shell elements [R3.03.07].
3) Voluminal finite elements of shells [R3.07.04].
4) Voluminal shell elements in nonlinear geometrical [R3.07.05].
5) Algorithm of resolution for the generalized problem [R5.01.01].
6) Modal parameters and norm of the eigenvectors [R5.01.03].
7) Quasi static nonlinear algorithm [R5.03.01].
8) Elastoplastic integration of the behavior models of Von Mises [R5.03.02].
9) Continuation methods of the loading [R5.03.80].
10) Multifrontal method [R6.02.02].


