

---

## Procédure TEST\_RESU

---

### 1 But

---

Comparer une valeur extraite d'une structure de données à une valeur de référence fournie par l'utilisateur.

Cette commande permet de tester une valeur numérique : entier, réel ou complexe extraite d'un concept déjà calculé. Aujourd'hui, on peut tester une composante d'un `cham_no` ou d'un `cham_elem`, une composante d'un champ extrait d'un `resultat`, un paramètre d'un `resultat`, une valeur 'globale' extraite d'un champ ou bien le contenu d'un objet quelconque d'un concept utilisateur.

La procédure écrit alors un message conventionnel :

- "OK" (si c'est bon),
- "NOOK" (sinon).

suivi de la valeur trouvée, la valeur de référence et le pourcentage d'erreur.

Les commandes `TEST_FONCTION` [U4.92.02] et `TEST_TABLE` [U4.92.03] permettent de tester les valeurs extraites des fonctions et des tables.

## 2 Syntaxe

```
TEST_RESU (
  ♦ | CHAM_NO=(_F ( ♦ CHAM_GD = chno, [cham_no]
                    / TYPE_TEST = / 'SOMM_ABS' ,
                              / 'SOMM' ,
                              / 'MAX' ,
                              / 'MIN' ,
                    ◇ NOM_CMP = ncmp, [K8]
                    / ♦ / NOEUD = no , [noeud]
                      / GROUP_NO = grno, [group_no]
                    ♦ NOM_CMP = nomcmp , [K8]
                    ◇ LEGENDE = legende, [K16]
                    # Voir définition de la valeur de référence
                    ),)
  | CHAM_ELEM=(_F ( ♦ CHAM_GD = chel, [cham_elem]
                    / TYPE_TEST = / 'SOMM_ABS' ,
                              / 'SOMM' ,
                              / 'MAX' ,
                              / 'MIN' ,
                    ◇ NOM_CMP = ncmp, [K8]
                    / ♦ MAILLE = ma , [maille]
                      ♦ / POINT = nupoint, [I]
                        / NOEUD = no, [noeud]
                        / GROUP_NO = grno, [group_no]
                        ◇ SOUS-POINT = nuspl,
                      ♦ NOM_CMP = nomcmp, [K8]
                      ◇ LEGENDE = legende, [K16]
                      # Voir définition de la valeur de référence
                    ),),
  | RESU =(_F ( ♦ RESULTAT = res, [resultat_sdaster]
                ◇ SENSIBILITE = (... voir [U4.50.02])
                ♦ / NUME_ORDRE = nuor, [I]
                  / NUME_MODE = numo, [I]
                  / INST = inst, [R]
                  / FREQ = freq, [R]
                  / NOEUD_CMP = (noeud, cmp), [l_Kn]
                  / NOM_CAS = nocas, [Kn]
                  / ANGL =  $\alpha$  , [R]
                ♦ / PARA = para, [K16]
                  / NOM_CHAM = nosymb, [K16]
                  / TYPE_TEST = / 'SOMM_ABS' ,
                              / 'SOMM' ,
                              / 'MAX' ,
                              / 'MIN' ,
                  ◇ NOM_CMP = ncmp, [K8]
                  / ♦ / NOEUD = no, [noeud]
                    / GROUP_NO = grno, [group_no]
                    / ♦ MAILLE = ma, [maille]
                      ♦ / POINT = nupoint, [I]
                        / NOEUD = no, [noeud]
                        / GROUP_NO = grno, [group_no]
```

```

                                ◇ SOUS-POINT = nusp,
                                ◆ NOM_CMP = nomcmp, [K8]
◇ LEGENDE = legende, [K16]
# Voir définition de la valeur de référence
),),
| GENE = (_F ( ◆ RESU_GENE = res, [VECT_ASSE_GENE]
               NUME_CMP_GENE = ncmp, [I]
               ~ RESU_GENE = res, [HARM_GENE]
               [MODE_GENE]
               ◆ / PARA = para, [K16]
               / NOM_CHAM = nosymb, [K16]
               NUME_CMP_GENE = ncmp, [I]
               ◆ / NUME_ORDRE = nuor, [I]
               / NUME_MODE = numo, [I]
               / FREQ = freq, [R]
               RESU_GENE = res, [TRAN_GENE]
               NOM_CHAM = nosymb, [K16]
               NUME_CMP_GENE = ncmp, [I]
               ◆ / NUME_ORDRE = nuor, [I]
               / INST = inst, [R]
               ◇ LEGENDE = legende, [K16]
# Voir définition de la valeur de référence
),),
| OBJET = (_F (
               ◆ NOM = nomobj, [K24]
               / S_I = vali, [I]
               / S_R = valr, [R]
               ◇ PRECISION = / 1.D-3, [DEFAULT]
               / prec, [R]
               ◇ CRITERE = / 'RELATIF', [DEFAULT]
               / 'ABSOLU',
               ◇ LEGENDE = legende, [K16]
# Voir définition de la valeur de référence
),),
)

# Définition de la valeur de référence

◇ REFERENCE = / 'ANALYTIQUE',
               / 'NON_REGRESSION',
               ◇ VERSION = vers, [Kn]
               / 'SOURCE_EXTERNE',
               / 'AUTRE_ASTER',

◆ / VALE = val, [R]
  / VALE_C = val, [C]
  / VALE_I = val, [I]

◇ VALE_ABS = / 'NON', [DEFAULT]
             / 'OUI',

◇ / | PRECISION = / 1.0D-3, [DEFAULT]
             / prec, [R]
             | CRITERE = / 'RELATIF', [DEFAULT]
             / 'ABSOLU',

```

```
/ | PRECISION = / (prec1, prec2), [1_R]  
| CRITERE = / (crit1, crit2), [1_Kn]
```

**Remarque :**

*Le type des concepts que l'on peut tester n'est pas vérifié par la commande. A priori, le mot clé CHAM\_NO permet de tester tous les types de cham\_no, de même pour les mots clés CHAM\_ELEM et RESU (pour les concepts de type resultat).*

## 3 Généralités

Cette commande permet de tester une valeur numérique scalaire récupérée dans un concept de type cham\_no, cham\_elem ou resultat, par rapport à une valeur de référence fournie par l'utilisateur.

Trois types de valeurs numériques peuvent être testées :

- une composante d'un champ (cham\_no ou cham\_elem ou champ faisant partie d'un resultat),
- un paramètre contenu dans un concept de resultat,
- une valeur globale d'un champ [§4.5].

Pour tester une composante de champ, il faut choisir un champ [§4.2] puis choisir une composante [§4.3].

Pour tester un paramètre, il faut choisir un numéro d'ordre [§4.2.4] et choisir le nom du paramètre.

La valeur numérique attendue (réelle, complexe ou entière) est fournie conformément à [§4.7].

**Remarques concernant les tests dans les structures de données "généralisées" :**

*On peut tester les composantes généralisées (déplacements, vitesses ou accélérations d'un transitoire dans l'espace modal). Il convient néanmoins d'être circonspect avec ce type de test. En effet la valeur d'une composante généralisée dépend entièrement de la norme du mode. Or celle-ci est déterminée de manière arbitraire. Donc sans normalisation préalable des normes, la valeur d'une grandeur généralisée est arbitraire. Enfin, il n'y pas de possibilité dans Code\_Aster de fixer la direction d'un mode. Pour un mode multiple, cela veut dire que, même une fois les modes normés, une grandeur généralisée peut prendre une valeur quelconque. Dans le cas d'un mode simple, il peut être orienté dans une direction ou dans la direction opposée. On obtient alors une valeur généralisée ou son opposée.*

## 4 Opérandes

### 4.1 Sélection d'un champ

Afin de tester un champ qui peut être un champ simple (`cham_no` ou `cham_elem`), ou un champ extrait d'un résultat, ou utilisera les mot-clés facteurs : `CHAM_NO`, `CHAM_ELEM` ou `RESU`.

#### 4.1.1 Champs "simples"

- ◆ `CHAM_GD = champ`  
Nom du `cham_no` ou du `cham_elem` dont on veut extraire une valeur.

#### 4.1.2 Opérande RESULTAT

- ◆ `RESULTAT = res`  
Nom du concept `resultat` traité.

#### 4.1.3 Opérande SENSIBILITE

- ◇ `SENSIBILITE = par_sensi`  
Nom du paramètre sensible. Voir [U4.50.22] pour les détails

#### 4.1.4 Opérande NOM\_CHAM

- ◆ `/ NOM_CHAM = nosymb`  
Nom symbolique du champ à sélectionner.

#### 4.1.5 Sélection d'un numéro d'ordre

- ◆ `/ NUME_ORDRE = nuor,`  
Numéro d'ordre du champ (ou du paramètre) recherché.  
  
`/ NUME_MODE = numo,`  
`/ INST = inst,`  
`/ FREQ = freq,`  
`/ NOEUD_CMP = (noeud, cmp),`  
`/ NOM_CAS = nocas,`  
`/ ANGL =  $\alpha$  ,`

Ces mots clés permettent d'identifier un numéro d'ordre dans un `resultat` [U4.71.00].

On les appelle des "variables d'accès".

Ils ne sont pas tous valables pour tous les types de `resultat`.

Lorsque l'accès se fait pas une valeur réelle (`ANGL`, `FREQ`, `INST`) la valeur donnée ne doit pas être ambiguë (cf [§4.7]).

## 4.2 Sélection d'un paramètre dans un résultat

Pour sélectionner un paramètre dans un résultat, il faut préciser le numéro d'ordre voulu [§4.2.4] et donner le nom du paramètre.

♦ `PARA = para`

Nom du paramètre cherché. Ce nom est attaché au type du concept `resultat` traité.

## 4.3 Sélection d'une composante d'un champ

L'accès à une grandeur se fait pour un `cham_no` par :

- le nom du nœud qui porte cette grandeur.

L'accès à une grandeur se fait pour un `cham_elem` par :

- le nom de la maille qui supporte l'élément,
- quelque chose qui précise :
  - soit le nom d'un nœud de cette maille pour les `cham_elem` "aux nœuds" (`_ELNO_`).
  - soit le numéro du point de GAUSS pour les `cham_elem` "aux points de GAUSS" (`_ELGA_`).

♦ `MAILLE = ma`

Nom de la maille où l'on veut tester le `cham_elem`.

♦ `/ NOEUD = no`

Nom du nœud dont on veut vérifier une composante.

`/ GROUP_NO = grno`

Pour faciliter l'utilisation de cette commande, on peut remplacer le mot clé `NOEUD` par `GROUP_NO`.

Dans ce cas, il faut que le groupe se réduise à un seul nœud.

`/ POINT = nupoint`

L'entier `nupoint` précise le numéro du point de GAUSS dont on veut tester la valeur (cas des `cham_elem` "aux points de GAUSS").

♦ `SOUS-POINT = nusp`

L'entier `nusp` précise le numéro du sous-point duquel on souhaite obtenir la valeur (cas des `cham_elem` à sous-points, utilisés par les éléments de structure : poutre, tuyaux, coques).

Dans le cas des plaques et des coques multicouches, le numéro du sous-point correspond au niveau dans l'ensemble des couches. Chaque couche est décrite par une peau inférieure, moyenne et supérieure. Par convention, pour  $N$  couches, ce numéro varie entre 1 et  $3N$  où le premier point se situe au niveau de la peau inférieure de la première couche et le  $3N$ ème point au niveau de la peau supérieure de la dernière couche (cf. [R3.07.03] et [R3.07.04] pour la numérotation des couches).

Dans le cas des poutres multifibres, cet entier est le numéro de la fibre dont la numérotation est décrite dans la documentation [U4.26.01] et [R3.08.08].

Dans le cas des tuyaux, il faut se référer à la description faite dans le document [R3.08.06].

♦ `NOM_CMP = ncmp`

Nom de la composante que l'on veut tester [U2.01.04].

## 4.4 Tester un champ "globalement"

Une fois un champ sélectionné [§4.2], on peut tester une quantité calculée globalement sur tout le champ. Pour cela, `NOM_CMP` ne doit pas être renseigné pour permettre de prendre en compte toutes les composantes du champ.

`/ TYPE_TEST = 'SOMM_ABS'`

La somme des valeurs absolues des composantes du champ.

```
/ TYPE_TEST = 'SOMM'
```

La somme des valeurs des composantes du champ.

```
/ TYPE_TEST = 'MAX'
```

Le maximum des valeurs des composantes du champ.

```
/ TYPE_TEST = 'MIN'
```

Le minimum des valeurs des composantes du champ.

## 4.5 Tester le contenu d'un objet JEVEUX

Cette fonctionnalité est réservée aux développeurs du Code. Pour l'utiliser, il faut connaître les noms des objets JEVEUX composant les concepts de l'utilisateur. Elle est destinée à vérifier la non régression des structures de données autres que les RESULTAT, CHAMPS, TABLE et FONCTION.

### 4.5.1 Opérande NOM

NOM = nomobj

Nom de l'objet jeux que l'on veut tester.

### 4.5.2 Opérandes S\_I / S\_R

S\_I = vali

Pour les objets de type entier, somme des valeurs contenues dans l'objet.

S\_R = valr

Pour les objets de type réel (ou complexe), somme des valeurs des nombres contenus dans l'objet. Pour les nombres complexes, on fait la somme des parties réelle et imaginaire.

#### Remarque :

*L'objectif de ce type de test ( NOM + S\_I ou S\_R ) est de tester globalement tout un vecteur. La "somme" qui est testée est malheureusement un mauvais "check sum" de l'objet : une permutation au sein du vecteur ne change pas cette somme. Un test plus prudent consiste à imprimer l'objet dans un fichier ( IMPR\_CO ) puis à tester le contenu de ce fichier avec un vrai "check sum" ( TEST\_FICHIER ).*

### 4.5.3 Opérandes CRITERE et PRECISION

Voir ci-dessous [§4.7].

## 4.6 Définition de la valeur de référence

◇ REFERENCE =

/ 'ANALYTIQUE' : la valeur de référence fournie est "analytique"

/ 'NON\_REGRESSION' : la valeur de référence fournie a été obtenue lors d'un précédent calcul par le Code\_Aster

VERSION = vers  
vers est le numéro de la version d' Aster qui a permis d'obtenir la valeur de référence (ex : '5.3.4')

/ 'SOURCE\_EXTERNE' : la valeur de référence fournie provient d'un programme autre qu' Aster (ou d'une référence bibliographique)

/ 'AUTRE\_ASTER' : la valeur de référence fournie est celle obtenue par un autre chemin Aster (autre commande, autre modélisation, option de

calcul, ...)

- ◆ / VALE = val  
Valeur réelle attendue pour la composante `nomcmp`, le paramètre `para` ou l'objet `nomobj`.
- / VALE\_C = val  
Valeur complexe attendue pour la composante `nomcmp` ou le paramètre `para`.
- / VALE\_I = val  
Valeur entière attendue pour le paramètre `para` ou l'objet `nomobj`.
- ◇ VALE\_ABS =  
= 'NON' la valeur de référence et la valeur calculée par Aster sont comparées telles quelles.  
= 'OUI' la valeur de référence et la valeur calculée par Aster sont comparées en valeurs absolues.
- ◇ PRECISION =  
Précision demandée (par défaut 1.D-3) pour accepter la valeur calculée.
- ◇ CRITERE =  
Type de test à effectuer.  
Si `v` est la valeur extraite, le test portera pour :
  - 'RELATIF' sur :  $|val - v| \leq prec \cdot |val|$
  - 'ABSOLU' sur :  $|val - v| \leq prec$ .

**Remarque :**

Lorsque la définition du numéro d'ordre d'un `RESULTAT` se fait par une variable d'accès réelle ( `FREQ` , `INST` , `ANGL` ), il ne faut pas qu'il y ait ambiguïté sur ce numéro d'ordre. Pour cela l'utilisateur définit un petit intervalle autour de la valeur demandée grâce aux mots clés `CRITERE` et `PRECISION`.

Dans ce cas (accès "réel") les mots clés `CRITERE` et `PRECISION` attendront donc 2 valeurs chacun : ( `crit1` , `crit2` ) et ( `prec1` , `prec2` ).

`crit1` et `prec1` concernent la valeur de référence.

`crit2` et `prec2` permettent de choisir l'intervalle de recherche du numéro d'ordre.

Les valeurs par défaut de `crit1` et `prec1` sont 'RELATIF' et 1.D-3.

Les valeurs par défaut de `crit2` et `prec2` sont 'RELATIF' et 1.D-3.

On ne peut définir explicitement `crit2` et `prec2` sans définir `crit1` et `prec1`.

## 4.7 Ajout d'une spécificité à la valeur testée

- ◇ LEGENDE =  
Chaîne de caractères d'au plus 16 caractères décrivant le test effectué.  
L'utilisateur a donc la possibilité de customiser son test.

## 5 Exemples

```
TEST_RESU
(  CHAM_NO = _F (  CHAM_GD   = f,
                   NOEUD     = 'N2',
                   NOM_CMP    = 'DX' ,
                   REFERENCE  = 'ANALYTIQUE' ,
                   VALE       = 1.2E-5 ,
                   PRECISION  = 1D-4 ,),

    CHAM_ELEM=(_F (  CHAM_GD   = SIGGA,
                   MAILLE     = 'M3',
                   POINT      = 3,
                   NOM_CMP     = 'SIXX' ,
                   REFERENCE   = 'NON_REGRESSION',
                   VALE        = 3.4E6 ,      ),

    _F (  CHAM_GD   = SIGNO,
         MAILLE     = 'M5',
         NOEUD      = 'N1',
         NOM_CMP     = 'SIXX' ,
         VALE        = 3.4E6 ,      ),),)

TEST_RESU
(  RESU =(_F      (  RESULTAT = mo_meca,
                   NUME_ORDRE= 2,
                   PARA       = 'FREQ' ,
                   VALE        = 1.5782, ),

    _F (  RESULTAT = evolth,
         SENSIBILITE = ps,
         NOM_CHAM   = 'TEMP' ,
         INST       = 12.0,
         NOEUD      = 'N37',
         NOM_CMP     = 'TEMP_R' ,
         VALE        = 0.0,
         CRITERE     = 'ABSOLU' ,),

    _F (  RESULTAT = evolth,
         NOM_CHAM   = 'FLUX_ELGA_TEMP' ,
         TYPE_TEST  = 'MAX' ,
         VALE        = 154.35, ),

    OBJET = _F (  NOM = 'CH1      .CHME.LIGREL.LIEL' ,
                 S-I = 1278484 , ),

)
```