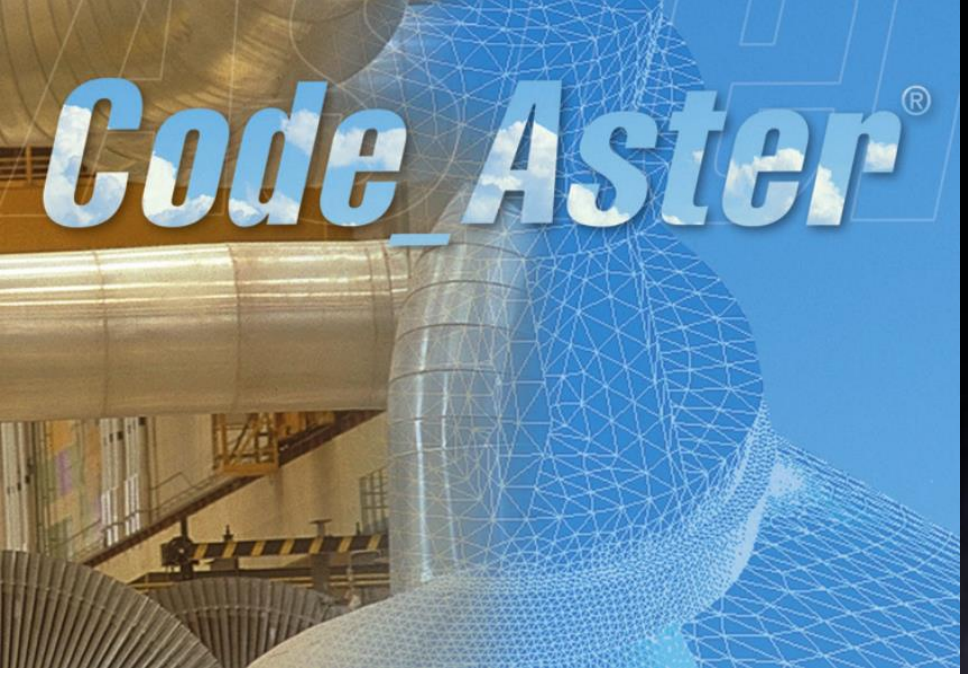


ポスト処理

Session 2012

1. Code_Aster の結果

結果には2つの形式がある:

- **場の数量 : cham_gd**
  - タイプ;
  - いくつかの成分;
  - 一つのアクセス番号 (実行時のステップ時間が無い)

- **結果のコンセプト: resultat**
  - 数量場の集計;
  - いくつかのアクセス番号;
  - パラメーター (モデルに依存している)

1.2 数量場

- 計算結果の値の場所:
  - 節点における場の量 → **NOEU**;
  - 要素において:
    - ✓ ガウス点での要素毎の場 → **ELGA**;
    - ✓ 節点での要素毎の場 → **ELNO**; （節点解）
    - ✓ 要素での一定値の場 → **ELEM**. (要素解)

- 命名規則 **XXXX_YYYY**：
  - 最初の4文字：数量の名前**SIGM**, **EPSI**, **ERRE**;
  - 中央の4文字：場所 **NOEU**, **ELGA**, **ELNO** or **ELEM**.

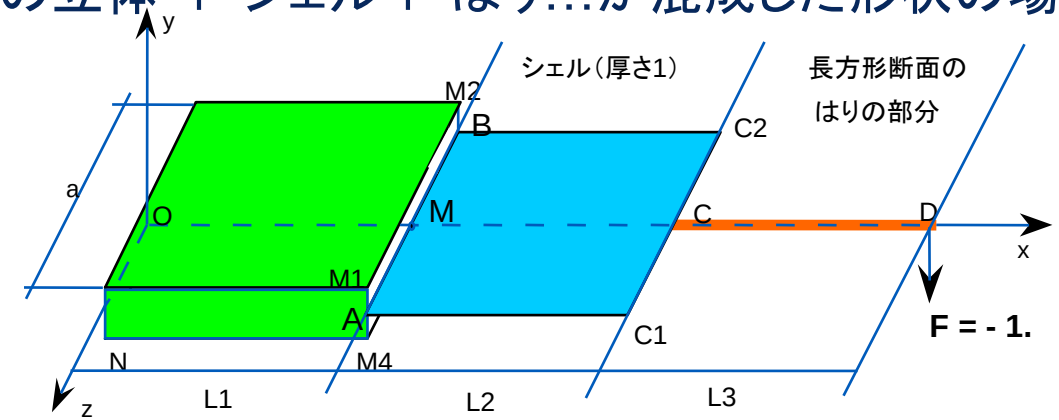
- 例：

  例外！ **DEPL, VITE, ACCE**
  SIEQ_ELNO, SIGM_ELNO

1.3 場の成分

- 成分は以下のものによって決まる：
  - 場のタイプ；
  - 形状

- 例 **SIEF_ELNO**：
  - 2次元と3次元で：**SIXX**, **SIYY**, **SIZZ**, **SIXY**, **SIXZ**, **SIYZ**.
  - 2次元の立体＋シェル＋はり…が混成した形状の場合は：



以下の成分を用いる:
- 3次元では **SIXX**, **SIYY**, **SIZZ**, **SIXY**, **SIXZ**, **SIYZ** ;
- シェル要素では **NXX**, **NYY**, **NXY**, **MXX**, **MYY**, **MXY** ;
- はり要素では **N**, **VX** , **VY** , **MX** , **MY**, **MZ** .

1.4 コンセプト resultat

- **resultat というデータ構造は，アクセス変数によって識別された1つ以上の数量場を計算ステップ毎に保存している**

- 例:
  - 計算ステップのそれぞれの時刻に対して，温度場 **TEMP**;
  - 初期状態の N 個の節点に対して，変位場 **DEPL**;
  - 計算ステップのそれぞれの瞬間に対して，変位場 **DEPL** と 応力場 **SIEF_ELGA**

- **resultat というコンセプトは，追加の結果と関連付けられ，パラメータから呼び出され，アクセス変数の値と関連付けられている**
- **例:**
  - 計算されたそれぞれのモードに対して，ノルム **NORME**

1.5 データ構造 evol_noli

- evol_noli は, コマンド STAT_NON_LINE と DYNA_NON_LINE によって生成される resultat コンセプトの一つである

- そのアクセス変数:
  NUME_ORDRE または INST

- 関連付けられたパラメーター:
  ITER_GLOB, ITER_LINE, RESI_GLOB_RELA, RESI_GLOB, ETA_PILOTAGE, …

- デフォルトで使用可能な場:
  DEPL, VITE, ACCE, SIEF_ELGA, VARI_ELGA

- 他の場も使用可能である. しかし, 適切なオプションを計算する必要がある

1.6 evol_noli 結果の例

概念(Concept)の構造

```
STRUCTURE DU CONCEPT U
```

31回の繰返計算をしました

```
CALCULE POUR          31 NUMEROS D'ORDRE
```

繰返計算の連番

シンボル名のリスト

```
                    LISTE DES NOMS SYMBOLIQUES:
! NUME_ORDRE !      DEPL       !   SIEF_ELGA    ! SIEF_ELNO_ELGA ! EQUI_ELGA_SIGM !
! ---------- !-----------------!----------------!----------------!----------------!
!          1 !     DEPL_R      !     SIEF_R     !     SIEF_R     !     SIEF_R     !
!        ... !      ...        !      ...       !      ...       !      ...       !
!         31 !     DEPL_R      !     SIEF_R     !     SIEF_R     !     SIEF_R     !
! ---------- !-----------------!----------------!----------------!----------------!
```

アクセス変数のリスト

```
                LISTE DES NOMS DE VARIABLES D'ACCES:
                                    INST             DE TYPE  R
```

ﾊﾟﾗﾒｰﾀ名のリスト

```
                    LISTE DES NOMS DE PARAMETRES:
! NUME_ORDRE !   ITER_GLOB    !   ITER_LINE    ! RESI_GLOB_RELA !   RESI_GLOB    !
! ---------- !----------------!----------------!----------------!----------------!
!          1 !       I        !       I        !       R        !       R        !
!        ... !      ...       !      ...       !      ...       !      ...       !
!         31 !       I        !       I        !       R        !       R        !
! ---------- !----------------!----------------!----------------!----------------!

                                <I> < >
```

1.7 データ構造 mode_meca

- mode_meca はコマンド MODE_ITER_INV, MODE_ITER_SIMULT と MACRO_MODE_MECA によって生成される resultat コンセプトの一つである

- そのアクセス変数は
  NUME_ORDRE, FREQ または NUME_MODE である

- 関連付けられたパラメーターは
  NORME, OMEGA2, AMOR_REDUIT, MASS_GENE, ... である

- 使用可能な場は
  DEGE_ELNO, DEPL, ECIN_ELEM, EFGE_ELNO, EFGE_ELNO
  である

1.8 データ構造 resultat

- **EVOL_ELAS** :時間発展を伴う準静的計算の結果
- **EVOL_NOLI** :非線形準静的計算または非線形動的計算の結果

- **DYNA_TRANS** :過渡的な線形動的計算の結果
- **DYNA_HARMO** :調和動的計算の結果
- **HARM_GENE** :調和モード空間における動的計算の結果(一般化された量)

- **MODE_MECA** :固有値と固有ベクトルの計算結果
- **MODE_GENE** :一般化された固有値と固有ベクトルの計算結果
- **MODE_ACOU** :音響場の固有値と固有ベクトルの計算結果
- **MODE_STAT** :静的モード計算の結果
- **BASE_MODALE** :機械的モードと静的モードの組み合わせによる結果

- **EVOL_THER** :過渡的な熱計算の結果

- **ACOU_HARMO** :調和音響計算の結果

2. 結果の出力 IMPR_RESU

- 何を？
  - メッシュ　**maillage**；
  - 数量場　**champ_gd**；
  - 構造に含まれるデータ　**resultat**.

- どんな形式か？
  - Listing　**FORMAT='RESULTAT'** と **'ASTER'**
  - MED　**FORMAT='MED'**
  - GMSH　**FORMAT='GMSH'**
  - CASTEM　**FORMAT='CASTEM'**
  - IDEAS　**FORMAT='IDEAS'**

2.1 IMPR_RESU 形式 'RESULTAT' と'ASTER'

- 目的:
  - リスト形式'RESULTAT'またはメッシュ形式 'ASTER'でメッシュまたは計算結果を書き込む

- この手順によって、以下のいずれかが出力される:
  - メッシュ;
  - 節点の上の場(変位、温度、固有モード、静的モードなど);
  - 節点またはガウス点の上の要素毎の場(応力, 一般化された内力, 内部変数,...)

2.2 IMPR_RESU 形式 'RESULTAT' : 構文 (1)

データ構造の結果から:

- 内容は?

```
INFO_RESU          =     / 'OUI'
                         / 'NON'   [DEFAUT]
```

- 抽出 : いつ ?

```
/ TOUT_ORDRE       =     'OUI'     [DEFAUT]
/ NUME_ORDRE       =     l_nuor
/ NUME_MODE        =     l_numo
/ INST             =     l_inst
/ FREQ             =     l_freq
/ NOM_MODE         =     l_nom
/ LIST_INST        =     l_reel
/ LIST_FREQ        =     l_reel
/ NOEUD_CMP        =     l_noecmp
```

2.2 IMPR_RESU 形式 'RESULTAT' : 構文 (2)

データ構造の結果から:

- 抽出 : 何を ?

```
/ TOUT_CHAM      =    / 'OUI' [DEFAUT]
                      / 'NON'
/ NOM_CHAM       =    l_nomsymb
```

- パラメータ : 何を ?

```
/ TOUT_PARA      =    / 'OUI'
                      / 'NON' [DEFAUT]
/ NOM_PARA       =    l_nompara
```

- パラメータ : テーブルは ?

```
/ FORM_TABL      =    / 'OUI'
                      / 'EXCEL'
                      / 'NON' [DEFAUT]
```

2.2 IMPR_RESU 形式 'RESULTAT' : 構文 (3)

データ構造の結果 と 数量場から:

- **抽出 : 何を？**

```
    / TOUT_CMP          =     / 'OUI' [DEFAUT]
                              / 'NON'
    / NOM_CMP           =     l_cmp
```

- **抽出 : どこを？**

```
    / TOUT              =     'OUI'   [DEFAUT]
    / | NOEUD           =     l_noeud
      | GROUP_NO        =     l_grno
      | MAILLE          =     l_maille
      | GROUP_MA        =     l_grma
```

2.2 IMPR_RESU 形式 'RESULTAT' : 構文 (4)

データ構造の結果と数量場から:

- **抽出 : 極値は？**

```
    / VALE_MAX          =     / 'OUI'
                              / 'NON' [DEFAUT]
    / VALE_MIN          =     / 'OUI'
                              / 'NON' [DEFAUT]
```

- **抽出 : 制限は?**

```
    / BORNE_SUP         =     vsup
    / BORNE_INF         =     vinf
```

2.3 IMPR_RESU 形式 'MED'(1)

MED (Modélisation et Échange de Données)

英語ではModeling and Data Exchange（モデルとデーターの交換）

- ソフトウェア間のデータ交換用にEDFによって開発された中間形式;
- バイナリー形式の移動可能なファイル ;
- MED のインターフェースをもった他のソフトウェアは，**IMPR_RESU**.を使ってCode_Aster が生成した結果を読む込むことができる

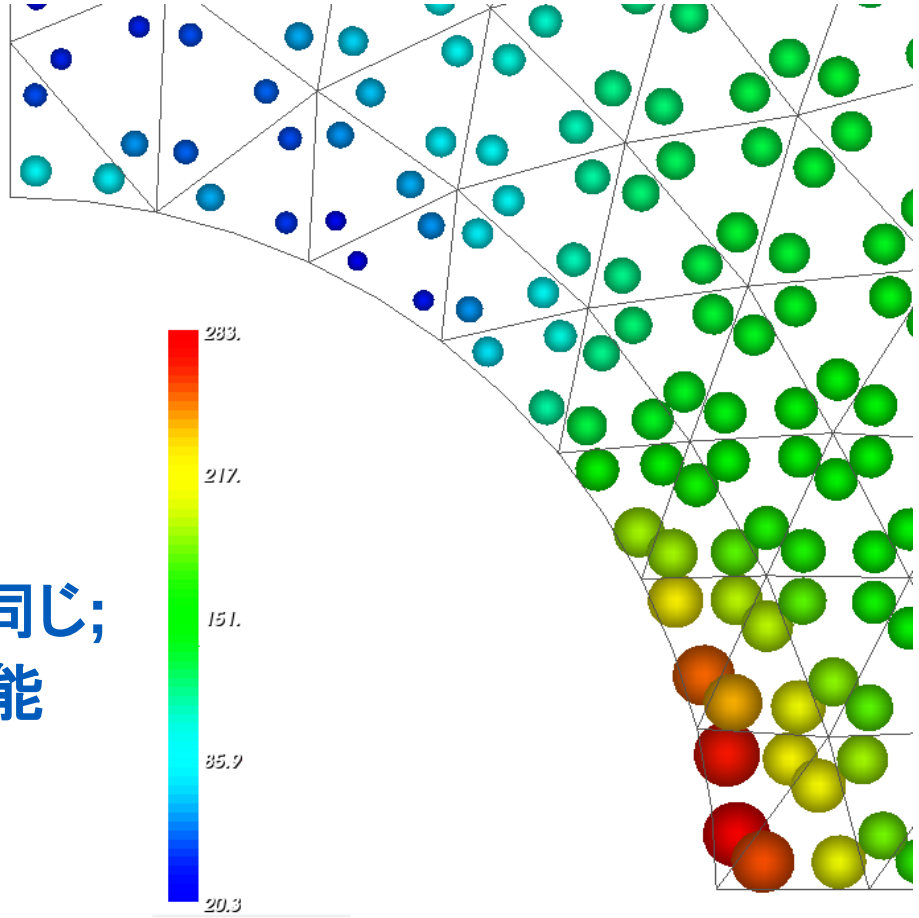
- これは Salome-Meca.でのポスト処理に使う形式である
- *.med files; と*.mmed files はメッシュ用で，*.rmed はAstkでの結果用

次のいずれかが出力される:

- メッシュ;
- 節点での場 ;
- 節点での要素毎の場 ;
- ガウス点での要素毎の場

2.3 IMPR_RESU 形式 'MED'(2)

ガウス点での可視化の例



注釈：
- 構文は 'RESULAT' 形式と同じ;
- モデルの一部を制限する機能

2.4 IMPR_RESU 形式 'GMSH'

以下のいずれかが *.msh ファイルに出力される:

- メッシュ;
- 節点での場;
- 節点での要素毎の場;

しかし、以下のものは出力されない:

- ガウス点での要素毎の場.
- **MED 形式を使おう！**

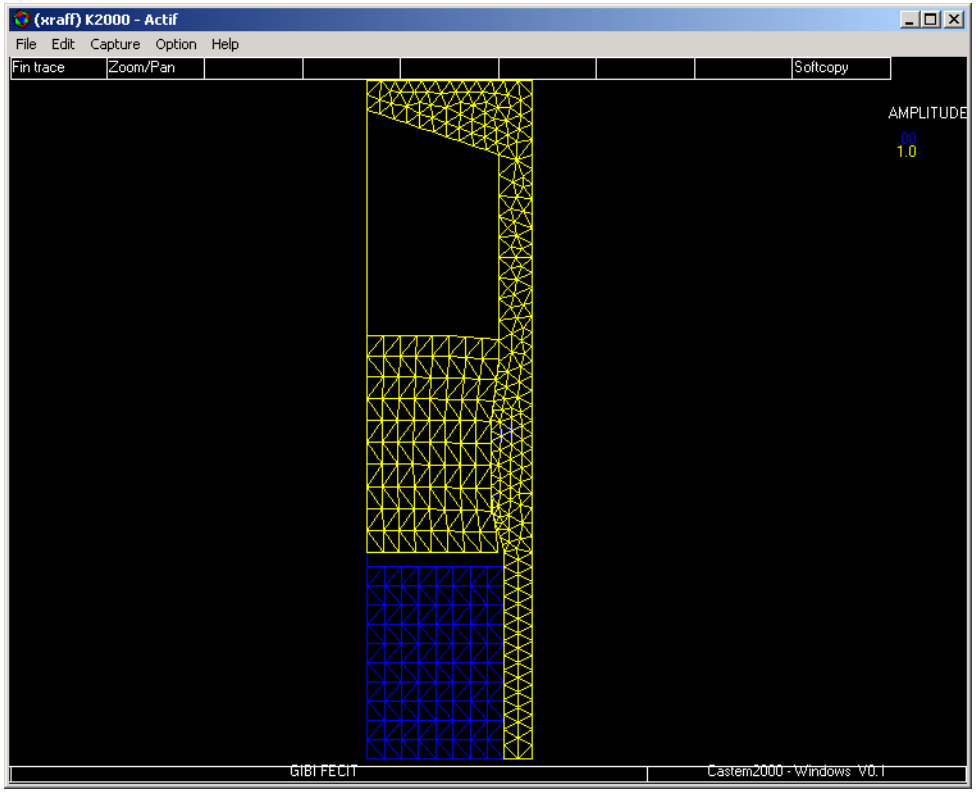
- **数量の種類を指定することができる**

```
TYPE_CHAM    =      / 'SCALAIRE', [DEFAUT]
                    / 'VECT_3D',
                    / 'TENS_2D',
                    / 'VECT_2D',
                    / 'TENS_3D',
NOM_CMP      =      lnomcmp,
```

2.5 IMPR_RESU 形式 'CASTEM'

以下のいずれかが *.cast ファイルに出力される:

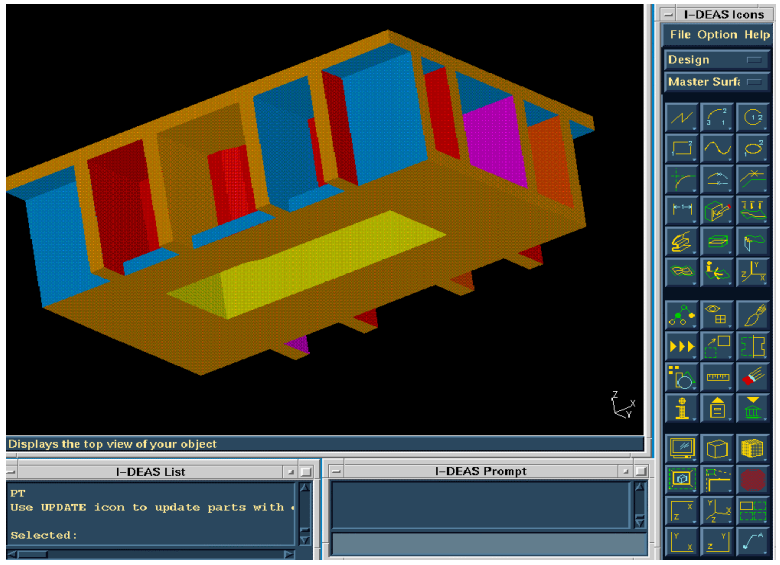
- メッシュ;
- 節点での場;
- 節点での要素毎の場



2.6 IMPR_RESU 形式 'IDEAS'

以下のいずれかが *.unv ファイルに出力される:

- メッシュ;
- 節点での場;
- 節点での要素毎の場;



- ユニバーサルファイルに使用されるデータセット:
  - dataset 55　節点での場(変位, 温度,...)
  - dataset 57　節点での要素毎の場(ひずみ, 応力,...)
  - dataset 56　ガウス点での要素毎の場(実際には, 要素上の場の一定値はガウス点での平均値になっている)

2.7 ガウス点での要素毎の場 (1)

GIBI またはIDEAS は *Code_Aster* のガウス点を認識しない
平滑化しないで計算した結果を確認するのに便利である

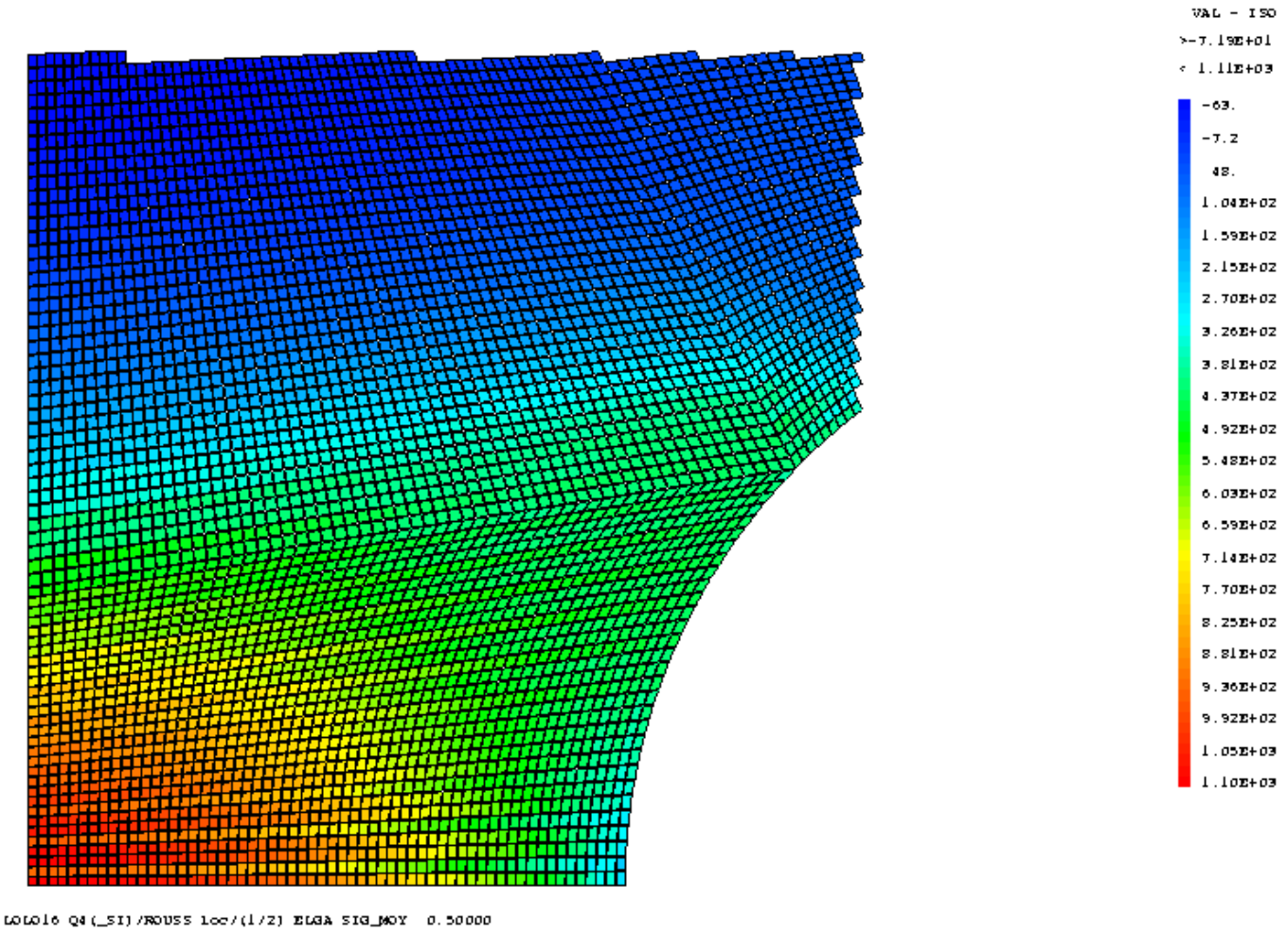
- The method (**MACR_ECLA_PG**):
  - 各要素は、(各ガウス点ごとに1つの)サブ要素に分割される
  - 場の量はサブ要素毎に一定である => 視認可能

- 例 :

```
ma2    = CREA_MAILLAGE( MAILLAGE = ma,
           ECLA_PG = _F( MODELE = MO, SHRINK = 0.90));
mo2    = AFFE_MODELE (MAILLAGE = ma2, AFFE (…));
resu2 = CREA_RESU (TYPE_RESU= 'EVOL_NOLI,'
           ECLA_PG = _F( MODELE_INIT= mo,
           RESU_INIT = resu,  MAILLAGE = ma2,
           NOM_CHAM =('SIEF_ELGA','VARI_ELGA'),),)
IMPR_RESU ( MODELE = mo2,
                RESU=_F( MAILLAGE = ma2, RESULTAT = resu2,
                                FORMAT = 'IDEAS'));
```

2.7 ガウス点での要素毎の場 (2)

- 例



2.8 関連文書の中の番号

- IMPR_RESU 作成:
  - [U4.91.01] **'RESULTAT' と 'ASTER'**
  - [U7.05.21] **'MED'**
  - [U7.05.32] **'GMSH'**
  - [U7.05.11] **'CASTEM'**
  - [U7.05.01] **'IDEAS'**

- ガウス点での要素毎の場:
  - [U4.23.02] **CREA_MAILLAGE**
  - [U4.44.12] **CREA_RESU**

- 以下のものを含む :
  - シンタックス,
  - 各々のオペランドの定義,
  - 例

3. CALC_ELEM / POST_ELEM

- 場とテーブルを作る

- **CALC_ELEM** オペレータ [u4.81.01]:
  - データ構造 の結果を作成または完成
  - 応力，ひずみ，エネルギー，クライテリア，誤差指標,...を計算

- **POST_ELEM** オペレータ [u4.81.22]:
  - テーブルを作成.
  - エネルギーの計算（ 位置，弾性，運動，散逸...）;
  - 積分値または平均値の計算

4. CREA_CHAMP / CREA_RESU / CREA_TABLE / CALC_TABLE

- 場，結果，テーブルを作る

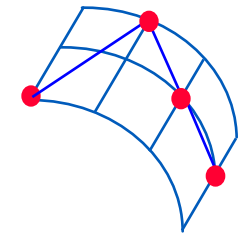
- **CREA_CHAMP** オペレータ [u4.72.04]:
  - 数量場 **champ_gd** を作成

- **CREA_RESU** オペレータ [u4.44.12] :
  - 場から データ構造 **resultat** を作成

- **CREA_TABLE** オペレータ [u4.33.02]:
  - 関数または実数の表からテーブル **table** を作成

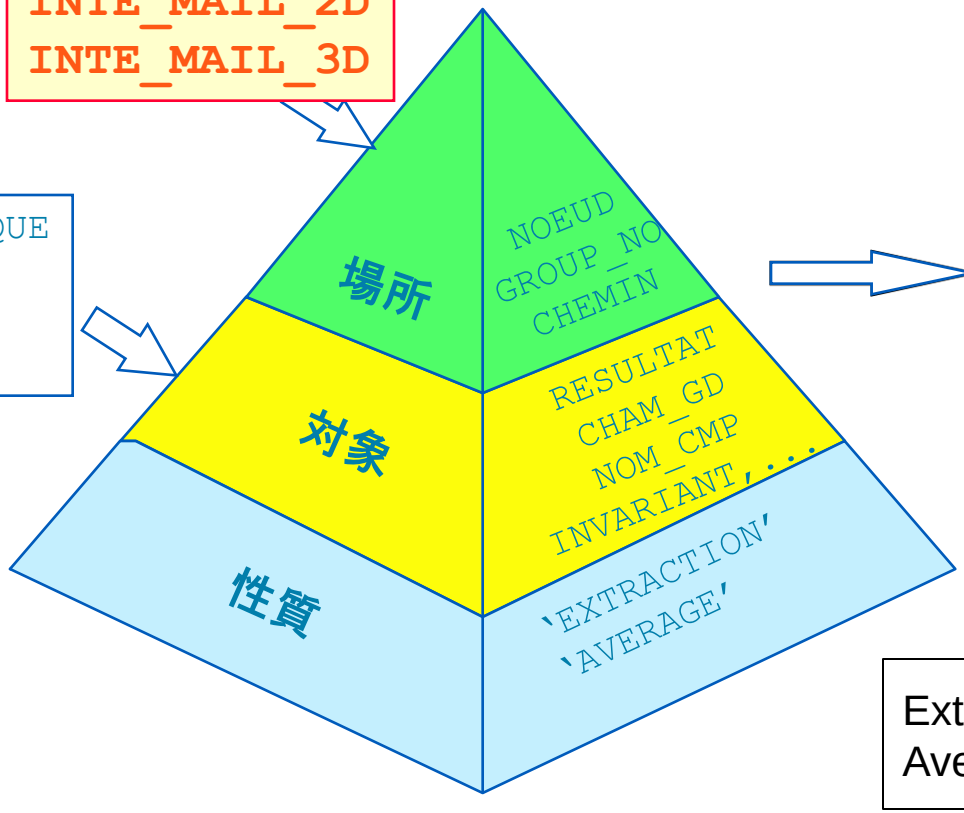
- **CALC_TABLE** オペレータ [u4.33.03]:
  - 表計算シートのようなテーブル **table** からデーターを操作

5. POST_RELEVE_T (1)



INTE_MAIL_2D
INTE_MAIL_3D

MECA_STATIQUE
CALC_NO
CALC_ELEM
...



IMPR_TABLE

Extraction→抽出
Average→平均

POST_RELEVE_T

5. POST_RELEVE_T (2)

- 数量場から成分値を抽出する
  - 例: 変位ベクトルDEPLから，成分 DX, DY, DZ.

- 実行した結果:
  - 平均値;
  - ベクトル場から各方向成分の量とモーメント量;
  - テンソル場の不変量:
  - 符号付きの行列の不変量;
  - 参照座標系の変更: 全体, 局所, 極, 円筒 および ユーザー設定

- テーブルを作成

5. POST_RELEVE_T (3)

- 原則

場所 → / CHEMIN (INTE_MAIL_2D) or (INTE_MAIL_3D)
/ NOEUD (N01 N045)
/ GROUP_NO (APPUI)

DSはデータ構造のことです。

対象 → / RESULTAT (DS resultat : EVOL_ELAS,...)
/ CHAM_GD (CHAM_NO : DEPL, ...
CHAM_ELEM : SIGM_ELNO,...)

性質 → / 'EXTRACTION' (値,...)
/ 'MOYENNE' (平均値, 最大値, 最小値,..)

6. MACR_LIGNE_COUPE

- 2つの端点と間隔で定義されたカットライン上の数量場から成分値を抽出する

- 原則:
  - カットライン上で線形メッシュを作成する;
  - 節点の上の場がそのメッシュ上に投影される;
  - 節点の値はテーブルに集計される

```
resu = STAT_NON_LINE   (        . . . )
MACR_LIGN_COUPE(RESULTAT=resu,
                MODELE=modele,
                LIGN_COUPE= _F(NB_POINTS=17,
                               COOR_ORIG=(0.,0.,0.),
                               COOR_EXTR=(10.,0.,0.),
                               TABLE=CO('tab1'),),
```

7. テーブルの出力：IMPR_TABLE

- テーブルの内容をリスト または Excel のファイルに出力する
  (ドキュメント [u4.91.03])
- 曲線のプロットもできる ！

- 選択：
  - **NOM_PARA ：**
    - 出力する列の選択；
  - **FILTRE ：**
    - 出力する線の選択（いくつかの線はクライティアをチェックする）；
  - **TRI ：**
    - 出力する行の順序の選択（昇順または降順）；
  - **FORMAT ：**
    - 出力形式の選択；

8. 関数を構築する: RECU_FONCTION

- 関数として抽出, 別の関数の関数として数量の進行 (document [u4.32.04]);
  - データ構造 result から: 生成された関数はメッシュの節点またはガウス点の成分の時間的進行に対応する.

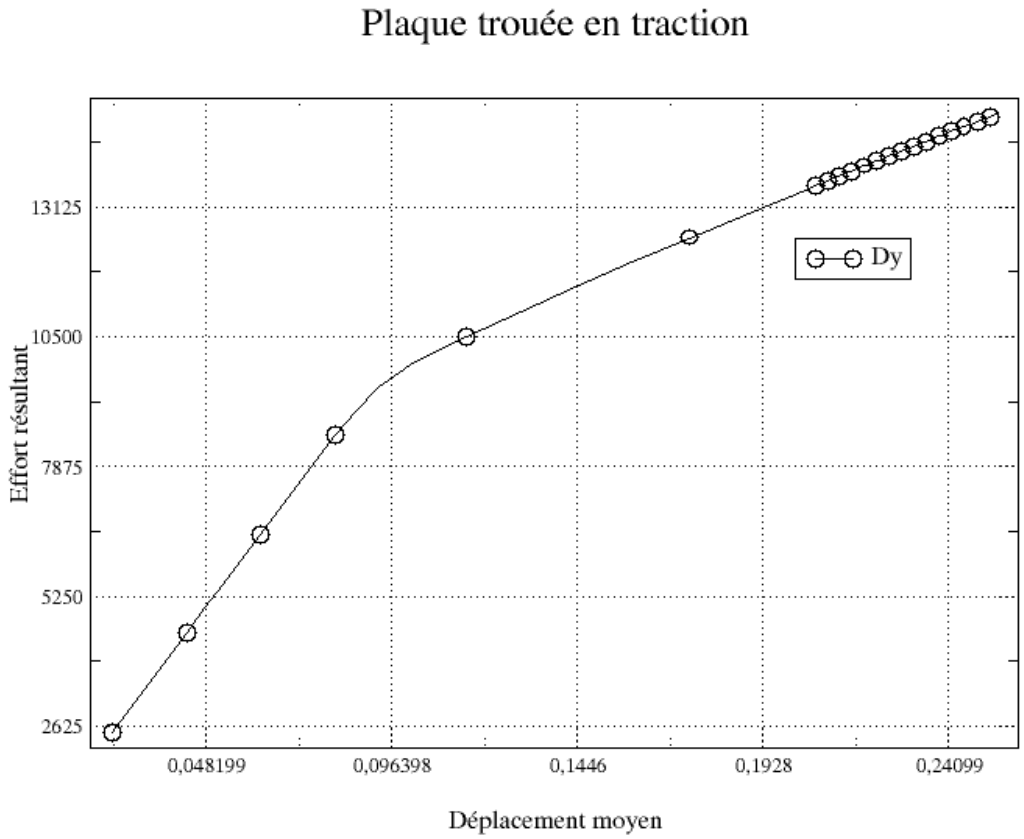
```
% COURBE SIGMA YY AU POINT G EN FONCTION DU TEMPS
   SYYG = RECU_FONCTION ( RESULTAT = RESUNL,
                          NOM_CMP = 'SIYY',
                          NOM_CHAM = 'SIEF_NOEU',
                          GROUP_NO = G,
                          TOUT_ORDRE = 'OUI') ;
```

  - データ構造 table から: この演算子は2つのパラメータの進行を抽出できる;

```
% COURBE SIGMA XX A T DONNE EN FONCTION DE L'ABSCISSE
   fct = RECU_FONCTION  ( TABLE = tab,
                          PARA_X = 'ABSC_CURV',
                          PARA_Y = 'SIXX') ;
```

9. 関数の出力：IMPR_FONCTION

- *Code_Asterの* **fonction** コンセプトから関数を出力する:
  - 形式 **'TABLEAU'** または **'XMGRACE'**,
  - document [u4.33.01];
  - 例:



10. STANLEY：対話式ポスト処理ツール

- 対話式のポスト処理の操作；
- Salome-Meca または Gmsh の中で可視化；
- 場　または　曲線；
- **STANLEY()** というコマンド入力だけ；
- Astkの中で，結果のベースから