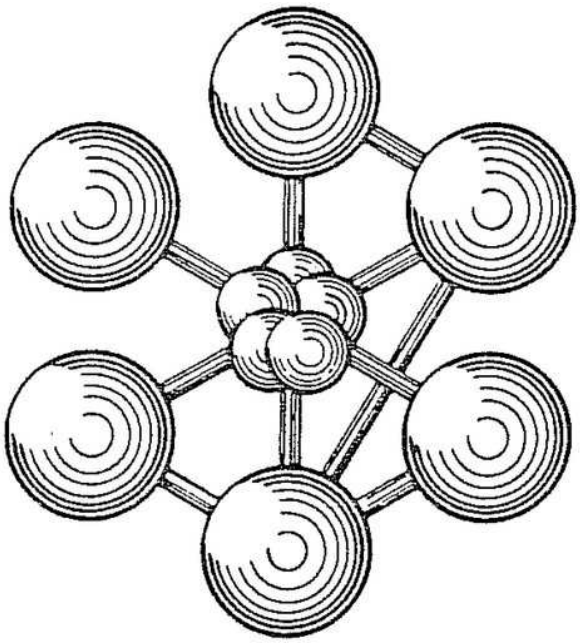
UTILISER CAST3M

E. LE FICHOUX

PRÉSENTATION ET UTILISATION DE CAST3M



ÉDITION 2011

Ce document a été initialement conçu et réalisé par Emmanuelle LE FICHOUX
à l'École Nationale Supérieure des Techniques Avancées (ENSTA).

Documentation Cast3M 2011

http://www-cast3m.cea.fr

Cast3M est un logiciel de calcul par la méthode des éléments finis pour la mécanique des structures et des fluides. Cast3M est développé au Département de Modélisation des Systèmes et Structures (DM2S) de la Direction de l'Énergie Nucléaire du Commissariat à l'Énergie Atomique et aux Énergies Alternatives (CEA).

Le développement de Cast3M entre dans le cadre d'une activité de recherche dans le domaine de la mécanique dont le but est de définir un instrument de haut niveau, pouvant servir de support pour la conception, le dimensionnement et l'analyse de structures et de composants.

Dans cette optique, Cast3M intègre non seulement les processus de résolution (solveur) mais également les fonctions de construction du modèle (pré-processeur) et d'exploitation des résultats (post-traitement). Cast3M est un logiciel « boîte à outils » qui permet à l'utilisateur de développer des fonctions répondant à ses propres besoins.

Cast3M est notamment utilisé dans le secteur de l'énergie nucléaire, comme outil de simulation ou comme plateforme de développement d'applications spécialisées. En particulier, Cast3M est utilisé par l'Institut de Radioprotection et de Sûreté Nucléaire (IRSN) dans le cadre des analyses de sûreté des installations nucléaires françaises.

Table des matières

Chapitre 1

Présentation générale de Cast3M

1.1 Historique

Le développement des codes de calcul a évolué de pair avec le matériel informatique. Au départ, les programmes de calcul étaient écrits pour résoudre des problèmes donnés et fonctionnaient généralement en boîte noire.

Trois programmes constituant le noyau du système Cast3M ont été développés au CEA/DMT : il s'agissait d'un programme de maillage, d'un programme de calcul et d'un programme de visualisation des résultats. La multiplicité des problèmes a ensuite conduit à développer des programmes de calcul distincts capables de communiquer entre eux le cas échéant.

Les progrès du matériel informatique a alors permis de développer de nombreuses fonctions qui réduisaient les temps de préparation des jeux de données. Ainsi, il fallait remettre à niveau tous les codes en répercutant dans chacun les améliorations et corrections effectuées dans l'un d'entre eux. De plus, les jeux de données devenaient de plus en plus compliqués et le post-traitement devait être capable de relire des fichiers différents selon les codes de calcul utilisés. À ces problèmes se sont ajoutés de nouveaux besoins tels que des structures plus complexes mettant en jeu des phénomènes variés et couplés (couplage mécanique-thermique) ou bien intégration des fonctions de maillage dans le processus de calcul (optimisation de forme).

Toutes ces considérations ont conduit à repenser à la base le problème du calcul numérique de manière à développer un code de calcul de nouvelle génération : Cast3M.

Contrairement aux anciens codes de calcul, écrits pour résoudre des problèmes donnés et auxquels l'utilisateur doit se plier, Cast3M peut s'adapter aux besoins de l'utilisateur pour résoudre *ses* problèmes. De plus, Cast3M donne à l'utilisateur la connaissance du problème effectivement posé et des méthodes employées.

1.2 Organisation d'un calcul

Une analyse générale effectuée à l'aide de la méthode des éléments finis peut se décomposer en quatre grandes étapes :

- le choix de la géométrie et du maillage,
- la définition du modèle mathématique,
- la résolution du problème discrétisé,
- l'analyse et le post-traitement des résultats.

Chacune des étapes peut également être décomposée en une série de processus élémentaires.

1. Choix de la géométrie et du maillage :
   - Définition des points, lignes, surfaces et volumes.
   - Discrétisation.
2. Définition du modèle mathématique :
   - Définition des données caractérisant le modèle :

type d'analyse : déformations ou contraintes planes, axisymétrie, séries de Fourier, etc...

formulation : mécanique, thermique, fluide, etc...

comportement du matériau : élastique (isotrope, orthotrope, ...), plastique (isotrope, parfait, ...), etc...

type d'éléments : poutres, barres, coques, etc...

- Définition des propriétés matérielles (constantes d'élasticité, masse volumique, etc...).
- Définition des propriétés géométriques (section des poutres, inerties, épaisseur des coques, etc...).
- Définition des conditions aux limites.
- Définition des sollicitations.
- Conditions initiales.
- ...

3. Résolution du problème discrétisé :
   - Calcul des matrices de rigidité et de masse de chaque élément fini.
   - Assemblage des matrices de rigidité et de masse de la structure complète.
   - Application des conditions aux limites.
   - Application des chargements.
   - Résolution du système d'équations.
4. Analyse et post-traitement des résultats :
   - Quantités locales : déplacements, contraintes, déformations, etc...
   - Quantités globales : déformation maximale, énergie de déformation, etc...

Les programmes de calcul par éléments finis classiques sont structurés selon cette logique, chaque étape étant associée à un module du code :

- le pré-processeur pour la définition du maillage et du modèle mathématique,
- le programme de calcul qui envoie une série de processus selon la procédure de calcul choisi par l'utilisateur, celui-ci ne peut maîtriser l'enchaînement des processus. La procédure agit comme une boîte noire sur laquelle l'utilisateur n'a aucune possibilité d'intervention.
- le post-processeur qui procède aux traitements nécessaires après avoir reçu les résultats des modules précédents.

Il apparaît clairement qu'un code de calcul classique exclut toute intervention de la part de l'utilisateur qui désirerait apporter des modifications répondant à ses propres besoins. Or, il peut s'avérer très utile de pouvoir définir pas-à-pas la séquence la mieux adaptée parmi les processus élémentaires disponibles pour chaque étape. Ceci est d'autant plus valable lorsque l'utilisateur doit résoudre des problèmes variés et localisés en différents points du processus de résolution.

En effet, outre les trois grandes étapes obligatoires, il faut pouvoir disposer de facilités telles que :

- la visualisation des informations à toutes les étapes de manière à contrôler les données introduites et le déroulement du calcul,
- l'archivage et la restauration des informations afin d'être capable d'interrompre un calcul et de le continuer ultérieurement,
- la possibilité d'itérer dans les étapes désirées.

Chaque étape peut être décomposée en une série de processus élémentaires. Chacun de ces processus élémentaires acquiert de l'information existante, la met en forme, la traite et fabrique de nouvelles informations. Faire un calcul revient donc à sélectionner les processus élémentaires adaptés au type du problème et à leur fournir les informations nécessaires existantes ou nouvelles.

C'est dans cette optique que Cast3M a été développé, afin de dépasser les limites d'adaptabilité offertes par les codes de calcul conventionnels.

1.3 Principes de développement de castem

Cast3M est avant tout un outil pour faire des calculs. Il intègre tous les processus utilisés par le passé et doit pouvoir accueillir tous ceux dont on aura besoin dans l'avenir pour traiter de nouveaux problèmes. Il permet à

l'utilisateur d'adapter ses calculs selon les problèmes qu'il doit résoudre. Il donne à l'utilisateur la connaissance du problème posé et des méthodes employées.

Ceci a été rendu possible par l'application :

- d'un principe : la SIMPLICITE,
- l'observation de quatre règles : l'ORTHOGONALITE, la VISIBILITE, la REGULARITE et la DOCUMENTATION,
- la poursuite d'un objectif : la QUALITE.

La simplicité Il s'agit d'un principe de bon sens, mais dont l'application n'est pas si aisée. C'est le fondement de la démarche scientifique : il est facile de faire des choses compliquées à partir de choses simples. Par ailleurs, simple ne signifie pas simpliste.

L'orthogonalité des processus Cette règle est basée sur deux idées :

- Les processus sont indépendants entre eux : on peut changer ou modifier un processus sans toucher aux autres.
- Tous les processus peuvent se combiner, c'est ce qui fait la puissance du code.

Il en résulte que l'on peut tester le fonctionnement des processus de manière indépendante.

La visibilité et la localité des processus La visibilité vise à supprimer l'existence de données cachées, en imposant que toutes les données soient indiquées au processus de manière visible, là où elles sont utilisées. La localité permet principalement de détecter les erreurs de données à l'endroit où elles sont définies et non pas beaucoup plus loin dans le calcul, ce qui rendrait leur diagnostic malaisé.

Cette double règle a un corollaire immédiat : l'information créée par le processus doit aussi être visible.

La régularité des processus Il s'agit d'une part de la régularité dans l'utilisation des processus et d'autre part de la régularité dans les possibilités d'un processus.

- Régularité dans l'utilisation des processus : il n'existe pas d'exception dans la syntaxe des données. Ceci permet d'avoir des syntaxes identiques pour des processus différents ayant besoin des mêmes données.
- Régularité dans les possibilités d'un processus : il n'existe pas d'exception dans les opérations possibles du processus.

La documentation des processus La documentation d'un processus fait partie du processus et elle doit être développée simultanément au processus. C'est elle qui permet à l'utilisateur de connaître la fonction du processus et son utilisation.

La règle est que c'est la documentation qui a raison et non pas la programmation du processus.

1.4 Mise en œuvre

Notion de processus élémentaire Il existe deux types de processus élémentaire : les OPERATEURS et les DIRECTIVES. Les informations sont appelées OBJETS.

- Les OBJETS sont nommés par l'utilisateur. Ils sont typés, ce qui permet aux opérateurs de les exploiter et de vérifier la syntaxe des données.
- Les OPERATEURS ont un nom. Ils créent un ou plusieurs objets dont les noms sont choisis par l'utilisateur. La seule condition de fonctionnement d'un opérateur dépend de l'existence de l'information au moment de son utilisation. L'utilisation d'un opérateur s'écrira :

  {nom du (des) résultat(s)} = {nom de l'opérateur} {nom de(s) l'information(s)} ;

  Le point-virgule indique la fin de l'instruction élémentaire.
- Les DIRECTIVES modifient un ou plusieurs objets existants ou produisent une sortie sur une unité logique (écran, imprimante, ...). Elles ne créent pas de nouveaux objets.

Il n'y a aucune distinction entre les opérateurs ou directives de pré-traitement, de calcul et de post-traitement, ils peuvent donc être utilisés à tout moment du programme selon les besoins du problème.

Langage GIBIANE Afin de convertir les noms des objets en entités informatiques utilisables par le programme, il faut disposer d'une interface. C'est le langage GIBIANE qui va permettre à l'utilisateur de communiquer directement avec le programme.

Les opérations avec GIBIANE consistent en une manipulation des objets existants dans le but de les modifier ou d'en créer de nouveaux.

1.5 Possibilités de castem

Traitement de nouveaux problèmes Pour traiter de nouveaux problèmes, on peut être amené à développer dans Cast3M de nouveaux opérateurs ou de nouveaux types d'objets.

Le développement d'un nouvel opérateur est facilité par l'orthogonalité des opérateurs entre eux. Par contre, l'addition de nouveaux types d'objets est moins fréquente puisqu'elle requiert non seulement l'addition de nouveaux opérateurs capables de traiter ce nouveau type d'objet mais aussi la modification de nombreux opérateurs généraux existants en vertu de la règle de régularité.

Exécutions répétitives et alternatives Certains problèmes nécessitent la répétition de quelques opérations élémentaires ou bien l'exécution conditionnelle de certaines opérations. Ceci est rendu possible grâce à certains opérateurs (REPETER, QUITTER, SI, SINON, FINSI, ITER, etc...) . L'utilisateur peut ainsi élargir les possibilités du programme en fonction de son type de problème.

Utilisation de procédures Une des conséquences de l'architecture adoptée pour Cast3M est la possibilité de créer des procédures, c'est-à-dire des méta-processus constitués d'un ensemble de processus élémentaires.

Ces procédures sont créées pour des besoins de nature diverse :

- certains enchaînement de données peuvent se retrouver de manière répétitive, les regrouper dans une seule instruction permet d'améliorer la lisibilité du programme,
- faciliter l'utilisation du programme pour des personnes peu familiarisées avec la méthode des éléments finis en revenant à un principe de boîte noire,
- pour des problèmes assez difficiles tels que des calculs non linéaires, il n'est pas raisonnable d'obliger l'utilisateur à définir explicitement un algorithme de résolution,
- à l'inverse, l'utilisateur peut vouloir créer de nouveaux opérateurs, modifier ou remplacer certains opérateurs selon ses besoins.

Les procédures sont écrites en langage de données et ont les propriétés suivantes :

- elles sont utilisables comme des opérateurs élémentaires,
- une procédure peut en appeler une autre et peut s'appeler elle-même,
- on peut surcharger un opérateur ou une procédure existante par une procédure,
- la séquence d'opérateurs élémentaires contenus dans une procédure est entièrement visible.

Grâce aux procédures, l'utilisateur peut programmer lui-même les processus nécessaires à la résolution de son problème. Il peut encore écrire et tester très rapidement de nouveaux algorithmes sans avoir besoin de modifier le logiciel.

À l'inverse, les procédures permettent de rendre transparents pour l'utilisateur des algorithmes et des méthodes relativement complexes, tels que les algorithmes de calcul en plasticité, en grands déplacements, contacts unilatéraux, etc...

Chapitre 2

Aspects généraux d'utilisation

2.1 Système d'unités

Cast3M ne dispose d'aucun système particulier d'unités de mesure. C'est à l'utilisateur de fournir les données dans un système cohérent vérifiant la loi fondamentale de la dynamique : $F = M.\gamma$. Une fois que les unités de mesure utilisées dans les données sont définies, tous les résultats seront exprimés dans ces mêmes unités.

Il existe une exception à cette règle concernant la mesure des angles qui doivent toujours être exprimés en **degrés**. En revanche, les températures et le coefficient de dilatation thermique doivent être exprimés dans des unités cohérentes.

Le tableau suivant regroupe quelques exemples de systèmes d'unités de mesure cohérents.

Longueur	Masse	Force	Temps	Masse volumique	Pression
m	kg	N	sec	kg/m^3	$Pa(=N/m^2)$
mm	$10^3 kg$	N	sec	$10^{-12} kg/m^3$	$MPa(=N/mm^2)$
in	$lb.sec^2/in$	lb	sec	$lb.sec^2/in^4$	$psi(=lb/in^2)$

2.2 Règles syntaxiques générales

Voici la liste des principales règles syntaxiques à observer lors de l'utilisation du langage GIBIANE :

- Les caractères **espace**, **virgule**, **égal** et **deux-points** sont des séparateurs.
- Le **point-virgule** termine une instruction.
- Une instruction doit être écrite sur moins de **9 lignes**, mais une même ligne peut contenir plusieurs instructions.
- L'interpréteur GIBIANE ignore toute ligne dont le premier caractère est un astérisque, d'où la possibilité pour l'utilisateur d'insérer des commentaires dans ses jeux de données.
- Les opérateurs et les directives sont définis par leurs **4 premiers caractères**, les caractères suivants n'étant pas pris en compte.
- L'instruction est interprétée de gauche à droite. Lorsque le programme rencontre le nom d'un opérateur, il lui transmet le contrôle de l'exécution.
- Seuls les **72 premiers caractères** d'une ligne sont pris en compte.
- Une instruction peut contenir des **parenthèses**. Conformément aux règles de l'algèbre, les instructions contenues dans les parenthèses les plus internes sont exécutées avant celles contenues dans les parenthèses les plus externes.
- Les **parenthèses** sont remplacées par le résultat de leur contenu avant interprétation des instructions externes. Toutefois, ce résultat n'est plus accessible à l'utilisateur puisqu'aucun nom ne lui a été attribué.
- Le signe = permet à l'utilisateur de donner un nom au résultat de l'instruction.
- La longueur du nom attribué à un objet ne doit pas dépasser **8 caractères**.

– Le programme associe un **type** à chaque chaîne de caractères rencontrée dans une ligne, il lui associe aussi un contenu.
L'analyse d'une suite de caractères se fait de la façon suivante :
- si elle est encadrée par des guillemets, c'est une chaîne de caractères à prendre telle quelle.
Exemple : 'AbC3eD'
- elle peut être interprétée comme une valeur numérique.
Exemple : 3.5
- sinon elle est changée en majuscule puis on cherche si c'est le nom d'un objet existant auquel cas on lui affecte son type et sa valeur.
Exemple : X = 5 ; Y = x + 3 ;
x est traduit en X et vaut 5.
- si ce n'est pas le nom d'un objet, on le crée avec un nom et un contenu identique à la suite de caractères en majuscule.
Exemple : dFIN est interprété comme un mot DFIN qui contient DFIN.
Notons ici que les opérateurs et les mots-clés de Cast3M doivent être en majuscules, ainsi fin ou FIN ou 'FIN' arrêtent le code alors que 'fin' ne l'arrête pas.

Il est fortement déconseillé aux utilisateurs d'attribuer à un objet le nom d'un opérateur existant, car il ne peut plus dans la suite du jeu de données être utilisé pour assurer sa fonction initiale.

Exemple de jeu de données :

```
OPTI DIME 3 ELEM SEG2 MODE TRID ;
OPTI ECHO 0 ;
*-Definition des points-
OEIL1 = 0. 0. 100. ;
P0 = 0. 0. 0. ;
P1 = 5. 0. 0. ;
P2 = 9. 0.5 0. ;
*-Definition des lignes-
C1 = C 15 P0 P1 P2 ;
L1 = D 5 P2 P0 ;
TRAC OEIL1 QUAL (C1 ET L1) ;
MESS 'Fin de l exemple' ;
FIN ;
```

2.3 Exécution du programme

Cast3M peut être utilisé en mode batch ou interactif. La session d'exécution est dans les deux cas mémorisée intégralement dans un fichier (**fort.98**) pouvant être édité puis soumis au programme Cast3M.

Il est possible de procéder en plusieurs étapes pour la mise au point d'un calcul : pour chaque étape, on effectue plusieurs passages successifs puis on transmet les données engendrées à l'étape suivante. Pour cela, il existe des opérateurs qui réalisent la sauvegarde de données sur fichier et leur récupération pour les exécutions suivantes (cf. 2.3).

Dans l'exécution de Cast3M, il faut utiliser au moins deux directives :

– OPTI : cette directive permet de déclarer les principaux paramètres du calcul. Elle se place généralement en début de programme. Elle est suivie d'un mot-clé qui spécifie l'option choisie, en particulier la dimension de l'espace, le type d'éléments utilisés, le type de calcul (contraintes planes, axisymétrique, Fourier, tridimensionnel...), etc...
– FIN : cette directive indique la fin de l'exécution du programme qui s'interrompt dès qu'il la rencontre.

Options générales de calcul Les valeurs des principaux paramètres de contrôle créés par la directive OPTI sont :

DIME : correspond à la dimension de l'espace utilisé par les opérateurs de maillage et de calcul. Valeurs : 1, 2 ou 3.

MODE : correspond au mode de calcul.
PLAN CONT : contraintes planes
PLAN DEFO : déformations planes
PLAN GENE P1 : déformations planes généralisées (P1= point support)
AXIS : axisymétrique
FOUR NN : analyse en série de Fourier (NN=numéro de l'harmonique)
TRID : tridimensionnel

ECHO : correspond à l'écho.
0 : les données n'apparaissent pas en sortie
1 : toutes les données apparaissent en sortie

DONN : correspond à l'unité logique sur laquelle le programme lit les données.
5 : les données sont lues au clavier
3 : numéro affecté par défaut au fichier de données
Pour interrompre l'exécution du programme en mode batch, il faut taper dans le fichier de données la commande OPTI DONN 5 à l'endroit où l'on souhaite l'interruption ; l'utilisateur peut alors taper ses instructions au clavier. S'il veut reprendre l'exécution du programme en cours, il doit taper OPTI DONN 3.

DENS : correspond à la densité courante (la valeur entrée spécifie la longueur de référence d'un élément).

ELEM : correspond au type d'éléments à fabriquer.

Exemple :
```
*Déclaration des paramètres
OPTI 'DIME' 3 'ELEM' QUA4 'MODE' PLAN CONT ;
```
Dans cet exemple, le calcul se fait en 3D et en contraintes planes. Le maillage génère si possible des quadrilatères à 4 nœuds et sinon des éléments de type linéaire.

Opérateurs et directives On peut insérer entre les directives OPTI et FIN n'importe quel opérateur ou directive.

• Un opérateur s'écrit sous la forme : `OBJ1 = NOM_OPERATEUR OBJi ;`

OBJ1 est l'objet 'résultat' créé par l'opérateur.
OBJi est l'objet 'donnée' ou argument nécessaire à l'opérateur. Le nombre d'arguments dépend de l'opérateur choisi.

Exemple :
```
LIG1 = DROITE PA PB ;
```
L'opérateur DROITE crée un segment délimité par les points PA et PB. Dans cet exemple, les arguments de l'opérateur DROITE sont les deux points frontières du segment.

Il existe également des arguments optionnels qui, selon les cas, ne sont pas obligatoires lors de l'appel de l'opérateur. Reprenons l'exemple précédent en y ajoutant un argument optionnel :

Exemple :
```
LIG1 = DROITE 10 PA PB ;
```
L'argument optionnel concerne le nombre d'éléments générés sur le segment. Si on ne le précise pas, le programme prend une valeur par défaut, fonction des densités associées aux points PA et PB.

Certains opérateurs font appel à des arguments qui doivent être précédés par des mots-clés afin d'éviter toute confusion entre les divers arguments de l'opérateur. Toujours sur le même exemple, on peut spécifier les densités associées aux points frontières du segment, pour cela, il faut insérer les mots-clés DINI et DFIN devant les arguments relatif aux densités :

Exemple :

```
LIG1 = DROITE 10 PA PB 'DINI' 1. 'DFIN' 0.5 ;
```

• Une directive s'écrit sous la forme : `NOM_DIRECTIVE OBJi ;`

OBJi est l'objet 'donnée' ou argument nécessaire à la directive. Le nombre d'arguments dépend de la directive choisie.

Exemple :

```
TRAC OEIL1 GEO1 ;
```

La directive TRAC permet de dessiner l'objet GEO1 de type MAILLAGE avec un point de vue OEIL1.

• Si les objets fournis sont incompatibles avec l'opérateur ou la directive, un message est imprimé avertissant l'utilisateur de la non obtention du résultat et donnant une indication sur la nature de l'erreur.

Exemple :

```
$  * OPTI DIME 3 ELEM SEG2;
$  *
$  * PA = 10. 10. 0. ;
$  * PB = 20. 20. ;
*****  ERREUR 37 ***** dans l'operateur =
Troisieme coordonnee ?
La lecture des donnees continue sur le terminal
Premiere ligne = donnees : deuxieme ligne = type des donnees.
20.      20.
FLOTTANT FLOTTANT
$
```

Les messages d'erreur les plus courants sont détaillés dans la partie 9. Nous ne saurions trop conseiller de lire attentivement toutes les lignes du message d'erreur notamment la deuxième et les deux dernières lignes.

L'environnement dans castem Il existe plusieurs directives d'environnement qui permettent l'obtention de l'aide en ligne (en mode interactif), la saisie de données au clavier, l'archivage et la restauration d'informations (qui permettent d'interrompre des calculs et de les reprendre ultérieurement), etc... Nous allons présenter ici les directives les plus usuelles.

INFO : fournit la documentation relative à l'opérateur demandé et s'utilise ainsi :

INFO 'nom de l'opérateur' ;

Cette directive permet de produire une notice de l'ensemble des opérateurs et procédures de Cast3M.

AIDE : liste tous les opérateurs utilisant le mot-clé demandé et s'utilise ainsi :

LIS1 = AIDE 'mot-clé' ;

Après cette instruction, la liste de tous les opérateurs trouvés est imprimée.

SORTIR : permet de sortir un objet maillage dans le fichier défini dans les options par 'OPTI SORT Nfichier' (Nfichier peut être un numéro d'unité logique ou un nom de fichier).

LIRE : permet de lire un objet maillage dans le fichier défini dans les options par 'OPTI LECT Nfichier'.

SAUVER : permet d'écrire un ou plusieurs objets dans le fichier défini dans les options par 'OPTI SAUV Nfichier'. Cela permet d'interrompre des calculs et de les reprendre ultérieurement.

RESTITUER : permet de remettre en mémoire les objets contenus dans le fichier défini dans les options par 'OPTI REST Nfichier'.

OBTENIR : permet de saisir un ou plusieurs objets au clavier.

LISTE : permet d'afficher toutes les informations relatives à un objet.

Exemple :

```
OBTE NBRE1*FLOTTANT NBENT2*ENTIER ;
LIST NBRE1 ;
LIST NBENT2 ;
OPTI SORT 7 SAUV 'resu1.sort' ;
SORTIR C1 ;
SAUV NBRE1 NBENT2 ;
```

L'objet maillage C1 est enregistré dans le fichier d'unité logique 7 c'est-à-dire **fort.7**.
Les objets NBRE1 et NBENT2 sont enregistrés dans le fichier de nom 'resu1.sort'.

Il est possible d'obtenir la notice en anglais (ainsi que tous les messages d'exécution) en spécifiant dans les options : OPTI LANG ANGLAIS.

Chapitre 3

Les objets

La structure informatique de Cast3M est basée sur le concept d'objets qui sont des données relatives à chaque processus. Les objets sont classés selon le type d'informations qu'ils renferment et selon la signification que prennent ces informations au cours de l'analyse.

Certains types ne représentent que des données mathématiques ou informatiques (entier, flottant, liste de mots, liste d'entiers, ...). D'autres ont un caractère plus physique et s'adaptent à une modélisation par éléments finis (champs par point, champs par élément, matrices de rigidité, ...).

Au cours d'une exécution de Cast3M, le type des objets peut être obtenu en listant leur contenu avec la directive LIST.

3.1 Classification des objets

La liste suivante répertorie les principaux types d'objet, classés par catégorie :

– Objets d'intérêt général

Type d'objet	**Description**
ENTIER	Objet constitué uniquement d'un nombre entier
FLOTTANT	Objet constitué uniquement d'un nombre réel
LISTENTI	Objet constitué d'une liste d'entiers
LISTREEL	Objet constitué d'une liste de réels
MOT	Objet constitué d'un mot
LISTMOTS	Objet constitué d'une liste de mots
LOGIQUE	Objet contenant une variable logique caractérisée par VRAI ou FAUX
TABLE	Ensemble d'objets dont le type peut être quelconque et caractérisé par un indice de type quelconque
EVOLUTION	Objet définissant un graphe : représentation d'une fonction réelle par une suite de couples (x, f(x))

– Objets de maillage

Type d'objet	**Description**
POINT	Objet définissant les coordonnées d'un point et la densité associée
MAILLAGE	Objet contenant la topologie du domaine discrétisé

– Objets de calcul

Type d'objet	**Description**
CHPOINT	Objet contenant n'importe quel type de données définies aux nœuds du maillage
LISTCHPO	Objet constitué d'une liste de CHPOINT
MMODEL	Objet associant un domaine physique, un maillage, une formulation élément fini et un comportement de matériau.
MCHAML	Objet contenant n'importe quel type de données définies dans les éléments du maillage. Les valeurs du champ peuvent être définies au centre de gravité de l'élément, aux nœuds de l'élément ou aux points d'intégration de l'élément
RIGIDITE	Objet contenant les données relatives à des matrices de rigidité, de masse, de rigidité géométrique, de conductivité. De manière générale, ce sont des matrices couplant des inconnues physiques.
CHARGEMENT	Objet contenant la description spatiale et temporelle d'un chargement
SOLUTION	Objet contenant l'ensemble des valeurs et vecteurs propres associés à une analyse modale
CONFIGURATION	Objet relatif à la description d'un champ de discrétisation
ATTACHE	Objet contenant la description des liaisons entre sous-structures en vue d'une analyse dynamique
BASEMODA	Objet contenant la description des liaisons s'exerçant sur une structure et la spécification de l'ensemble des modes et solutions statiques
BLOQSTRU	Objet contenant la description des liaisons entre sous-structures en vue d'une analyse dynamique
ELEMSTRU	Objet permettant d'écrire des liaisons entre sous-structures et contenant la description d'un élément de structure avec la géométrie associée
STRUCTURE	Objet relatif à la description d'une structure et contenant la rigidité et la masse s'y rapportant

– Objets de post-traitement

Type d'objet	**Description**
VECTEUR	Objet relatif à la visualisation d'un champ par points au moyen de vecteurs
DEFORME	Objet relatif à la caractérisation d'un domaine déformé (obtenu en superposant un objet de type MAILLAGE à un objet de type CHPOINT

3.2 Utilisation des objets élémentaires

Objets de type ENTIER Pour créer un objet de type ENTIER, on utilise le signe =. On peut effectuer des opérations arithmétiques élémentaires sur ces entiers avec les opérateurs +, -, *, /, **.

Exemple :

```
I1 = 24 ;
I2 = 3 ;
I3 = I1 + I2 ;
I4 = I3 / 3 ;
I5 = I4**2 ;
LIST I5 ;
```

I4**2 correspond à $I4^2$. La directive LIST permet de lister à l'écran le contenu et le type de I5.

```
$  * LIST I5 ;
Entier valant: 81
```

Objets de type FLOTTANT Ils sont créés de la même façon que les entiers mais les nombres doivent contenir un point ou être exprimés en notation puissance.

Exemple :

```
F1 = 2.5 ;
F2 = 4E-1 ;
F3 = F1 + F2 ;
MESS 'Valeur de F3 :' F3 ;
```

L'opérateur MESS permet d'éditer sous forme de message les arguments attribués (de type ENTIER, FLOTTANT ou MOTS).

Objets de type LOGIQUE Ces objets contiennent une variable logique de valeur VRAI ou FAUX. Pour les créer, on peut leur affecter directement les mots 'VRAI' ou 'FAUX'. On peut aussi les créer avec les opérateurs de comparaison EGA, >EG, <EG, <, > et NEG. Les variables logiques sont utilisées avec les directives SI, SINON, FINSI.

Exemple :

```
LOG1 = VRAI ;
I1 = 2 ; I2 = 4 ;
I3 = I1 + 2 ;
LOG2 = I2 EGA I3 ;
SI LOG2 ;
MESS 'Les deux entiers I2 et I3 sont égaux ' ;
SINON ; MESS 'Les deux entiers ne sont pas égaux' ;
FINSI ;
```

Objets de type LISTENTI, LISTREEL et LISTMOTS Ces objets sont des listes d'entiers, de réels ou de mots. Pour les créer, on peut utiliser l'opérateur PROG pour les réels, l'opérateur LECT pour les entiers ou l'opérateur MOTS pour les mots.

Exemple :

```
LISR1 = PROG 5*1. ;
LISR2 = PROG 1. 1. 1. 1. 1. ;
LISR3 = PROG 1. 'PAS' 1. 5. ;
LISR4 = PROG 1. 'PAS' 1. 'NPAS' 4 ;
LISI1 = LECT 4*3 ;
```

```
LISI2 = LECT 1 'PAS' 1 4 ;
LISI3 = LISI1 + LISI2 ;
LISI4 = LISI1 ET LISI2 ;
LISM1 = MOTS EXX1 EXX2 EYY1 EYY2 ;
```

Les listes LISR1 et LISR2 sont identiques, elles contiennent les flottants : 1. 1. 1. 1. 1.
Les listes LISR3 et LISR4 sont identiques, elles contiennent les flottants : 1. 2. 3. 4. 5.
La liste LISI1 contient les entiers : 3 3 3 3
La liste LISI2 contient les entiers : 1 2 3 4
La liste LISI3 contient les entiers : 4 5 6 7 . L'opérateur élémentaire + additionne terme à terme les éléments des deux listes.
La liste LISI4 contient les entiers : 3 3 3 3 1 2 3 4 . L'opérateur ET concatène les deux listes.
La liste LISM1 contient les mots : EXX1 EXX2 EYY1 EYY2. Les mots de la liste ne peuvent avoir plus de 4 caractères.

3.3 Utilisation des objets de maillage

Objets de type POINT Ce type d'objet est utilisé pour représenter un point ou un vecteur. Pour créer un objet de type POINT, on peut utiliser le signe =. Il faut déclarer la dimension de l'espace (avec la directive OPTI DIME) avant toute création d'objet de type POINT.

Exemple :

```
OPTI DIME 3 ;
X2 = 10. * (COS 30) ;
Y2 = 10. * (SIN 30) ;
P1 = 0. 0. 0. ;
P2 = X2 Y2 0. ;
LIST P1 ;
LIST P2 ;
```

```
$  * LIST P1 ;
Point dont le numero est actuellement 1
Coordonnees: 0.00000E+00 0.00000E+00 0.00000E+00 Densite: 0.00000E+00
$  * LIST P2 ;
Point dont le numero est actuellement 2
Coordonnees: 8.6603     5.0000     0.00000E+00 Densite: 0.00000E+00
$  *
$  * FIN;
```

Lorsqu'on liste un objet de type POINT, il apparaît une quantité supplémentaire aux coordonnées. Il s'agit de la densité. Cette valeur est nulle par défaut, on peut la modifier avec la directive DENS.

La densité permet de contrôler la taille locale des côtés des éléments se raccordant à un point. Elle s'exprime dans la même unité que les coordonnées des points. Si l'on crée plusieurs éléments entre deux points, la taille des éléments en ces deux points sera à peu près égale à la densité associée et la taille des autres éléments sera une progression géométrique. Nous reviendrons sur la notion de densité dans la partie 5.

Il est possible de créer des objets de type POINT avec l'opérateur PLUS qui permet d'effectuer des translations.

Exemple :

```
P1 = 0. 1. ;
VEC1 = 1. 1. ;
P2 = P1 PLUS VEC1 ;
```

VEC1 représente le vecteur de translation, le point P2 aura pour coordonnées : (1. 2.).

Objets de type MAILLAGE Les objets de type MAILLAGE représentent les éléments qui peuvent être des lignes, des surfaces ou des volumes. Il existe de nombreuses manières de les créer. Nous détaillerons tous ces opérateurs dans la partie 5.

Pour créer un segment de droite, on utilise l'opérateur DROITE et pour créer un arc de cercle, l'opérateur CERCLE. Selon le type d'éléments spécifié dans la directive OPTI ELEM, les lignes créées auront 2 nœuds (SEG2) ou 3 nœuds (SEG3).

Exemple :

```
OPTI ELEM SEG2 ;
P1 = 0. 0. ;
P2 = 1. 1. ;
LIG1 = DROITE 2 P1 P2 ;
OPTI ELEM SEG3 ;
LIG2 = DROITE 2 P1 P2 ;
LIST LIG1 ;
LIST LIG2 ;
```

```
MAILLAGE 4637 : 2 element(S) de type SEG2
0 sous-reference(s)
1ere ligne  numero element : 2eme couleur : 3eme... noeud(s)
         1     2
       DEFA  DEFA

         1     3
         3     2
$  * LIST LIG2 ;
MAILLAGE 4641 : 2 element(S) de type SEG3
0 sous-reference(s)
1ere ligne  numero element : 2eme couleur : 3eme... noeud(s)
         1     2
       DEFA  DEFA

         1     5
         4     6
         5     2
```

Les points intermédiaires ne sont pas nommés, l'utilisateur n'y a pas accès sauf en les nommant.

3.4 Utilisation des objets de calcul

Objet de type CHPOINT Dans un objet de type CHPOINT, appelé aussi champ par points, le champ est défini par ses valeurs à chaque nœud de l'objet maillage référencé. Il peut contenir une ou plusieurs composantes. Les composantes sont repérées par leur nom qui peut être choisi par l'utilisateur ou déterminé arbitrairement par l'opérateur qui a créé l'objet CHPOINT. Un nom de composante comporte au maximum 4 caractères.

Il est possible de créer manuellement un objet CHPOINT, pour cela on utilise l'opérateur MANU suivi du mot-clé CHPO.

Exemple :

```
LIST1 = PROG 0.1 PAS 0.1 NPAS 9 ;
CHP1 = MANU 'CHPO' LIG1 2 'UX' LIST1 'UY' 0.5 ;
LIST CHP1 ;
```

Le champ par points CHP1 s'appuie sur les nœuds du maillage LIG1 (qui contient 10 points). Avant de donner les noms des composantes et leurs valeurs, il faut spécifier le nombre de composantes (ici 2). La liste des valeurs de la composante UX à chaque nœud est donnée sous forme de liste de réels (objets de type LISTREEL). Cette liste doit comporter autant de valeurs qu'il y a de nœuds dans le support géométrique (objet de type MAILLAGE ou POINT). La composante UY est constante sur tous les points du maillage, on peut alors rentrer directement une valeur (type ENTIER ou FLOTTANT) à la place d'une liste de réels.

```
CHPOINT de pointeur 4649 contenant 1 sous-champ(s)
Titre :
Type  :
L'attribut de nature du CHPOINT est:  INDETERMINE
Option de calcul :  PLAN
 Points  Inconnue  .....
                UX             UY                      UX             UY
      1     1.00000E-01    5.00000E-01       2     1.00000E+00    5.00000E-01
      3     2.00000E-01    5.00000E-01       4     3.00000E-01    5.00000E-01
      5     4.00000E-01    5.00000E-01       6     5.00000E-01    5.00000E-01
      7     6.00000E-01    5.00000E-01       8     7.00000E-01    5.00000E-01
      9     8.00000E-01    5.00000E-01      10     9.00000E-01    5.00000E-01
```

Lorsqu'on liste un objet de type CHPOINT, il apparaît diverses informations : le titre, le type, l'attribut de nature et l'option de calcul. La nature du champ peut être indéterminée (comme ici), diffuse ou discrète : quand le champ représente une grandeur continue (par exemple un champ de déplacement), il est de nature diffuse ; quand il représente un champ nodal équivalent (par exemple une force nodale), il est de nature discrète.

Il est possible de créer à partir d'un objet CHPOINT à plusieurs composantes un objet CHPOINT de même type en extrayant une ou plusieurs composantes données. Pour cela, on utilise l'opérateur EXCO. On peut garder les mêmes noms de composante ou bien en donner de nouveaux.

Exemple :

```
CHP2 = EXCO 'UX' CHP1 ;
CHP3 = EXCO 'UX' CHP1 'U1' ;
CHP4 = EXCO (MOTS UX UY) CHP1 (MOTS U1 U2) ;
```

Le champ CHP2 aura une composante de nom UX et dont les valeurs seront égales à celles de la composante UX de CHP1. Le champ CHP3 aura une composante de nom U1 et dont les valeurs seront égales à celles de la composante UX de CHP1. Le champ CHP4 aura deux composantes de nom U1 et U2 et dont les valeurs seront égales à celles des composantes UX et UY de CHP1.

Il est possible de changer le nom des composantes d'un objet CHPO avec l'opérateur NOMC.

Exemple :

```
CHP5 = NOMC 'DEP' CHP2 ;
CHP6 = NOMC (MOTS UX UY) (MOTS U1 U2) CHP1 ;
```

Le champ CHP5 sera identique au champ CHP2 mais le nom de sa composante sera DEP et non plus UX.

Le champ CHP6 sera identique au champ CHP1 mais les noms de ses composantes seront U1 et U2 et non plus UX et UY.

Objet de type MCHAML Un objet de type MCHAML, appelé aussi champ par éléments, contient des données définies pour chaque élément du maillage référencé. Il peut s'agir des caractéristiques des matériaux, des caractéristiques géométriques des éléments (section, épaisseur...), des contraintes, des déformations... Comme les objets CHPOINT, il peut contenir une ou plusieurs composantes.

Dans chaque élément, les valeurs du champ peuvent être définies :

– aux nœuds de l'élément

- au centre de gravité de l'élément
- aux points d'intégration de la raideur
- aux points d'intégration de la masse
- aux points de calcul des contraintes

Un objet MCHAML peut être créé avec l'opérateur MANU suivi du mot-clé CHML ou CHAM, selon que le champ est constant sur tous les nœuds ou bien nul partout sauf en un point.

Exemple :

```
CHAM1 = MANU 'CHML' LIG1 G 9.81 ;
CHAM2 = MANU 'CHAM' MODL1 'POSI' GRAVITE GP 2 1 50. ;
```

Le champ CHAM1 s'appuie sur LIG1 (objet de type MAILLAGE), il a une seule composante de nom G et constante sur tous les éléments du maillage.

Le champ CHAM2 s'appuie sur MODL1 (objet de type MMODEL), sa composante de nom GP est nulle partout sauf en un point (le point 1 de l'élément 2) où elle vaut 50. Le champ est défini au centre de gravité de élément.

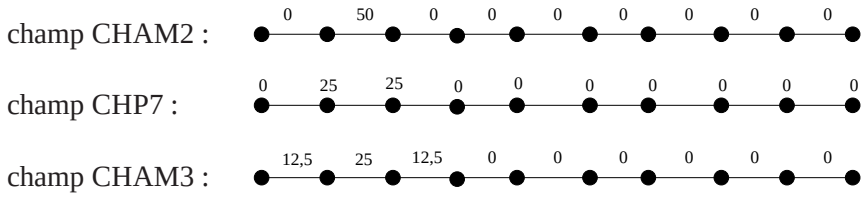
```
$  * LIST CHAM2 ;
 +-------------------------------------------------------------------------+
 |          OBJET  MCHAML  CONTENANT      1 ZONE(S) ELEMENTAIRE(S)         |
 |          TYPE :                                                         |
 |          OPTION DE CALCUL   DEFORMATIONS PLANES                         |
 +-------------------------------------------------------------------------+
           ZONE ELEMENTAIRE NUMERO      1
           ----------------------------
   POINTEUR SUR L'OBJET MAILLAGE   4637
   NUMERO DE L'HARMONIQUE              0
   POINTEUR SUR LES POINTS SUPPORTS      4691
   VALEURS DONNEES AU CENTRE DE GRAVITE
   NOM DU CONSTITUANT                          4683
   NOMBRE DE COMPOSANTES               1

     1-ERE COMPOSANTE    -    NOM :  GP          -    TYPE :  REAL*8
  ELEMENT               1           2           3           4           5
  POINT  1      0.000E+00   5.000E+01   0.000E+00   0.000E+00   0.000E+00
  ELEMENT               6           7           8           9
  POINT  1      0.000E+00   0.000E+00   0.000E+00   0.000E+00
```

Si le support n'est pas précisé lors de la création d'un objet CHAML avec l'opérateur MANU, le champ sera par défaut défini aux nœuds des éléments. On peut utiliser l'opérateur EXCO sur les objets MCHAML de la même manière qu'avec les objets CHPOINT (l'opérateur NOMC ne s'utilise qu'avec les objets CHPOINT).

Manipulation des objets CHPOINT et MCHAML Il est possible de construire un objet CHPOINT à partir d'un objet MCHAML avec l'opérateur CHAN(GER) suivi du mot-clé CHPO. L'opération inverse est possible avec CHAN suivi du mot-clé CHAM.

Exemple :

```
CHP7 = CHAN 'CHPO' MODL1 CHAM2 ;
CHAM3 = CHAN 'CHAM' CHP7 MODL1 'GRAVITE' ;
LIST CHAM2 ;
LIST CHP7 ;
LIST CHAM3 ;
```



On peut remarquer qu'en repassant du champ par points CHP7 au champ par éléments CHAM3, les valeurs obtenues sont différentes du champ par éléments initial CHAM2. Ceci est lié au principe de l'opérateur CHAN qui travaille sur des moyennes.

On peut utiliser sur les objets CHPOINT et MCHAML les opérateurs élémentaires de calcul algébrique : +, -, *, /, **. Il est préférable que les composantes des différents champs aient les mêmes noms et s'appuient sur les mêmes maillages ou modèles. En effet, l'opérateur + par exemple n'effectue pas la somme des deux champs s'ils sont différents mais fait une adjonction.

Objet de type MMODEL Les objets de type MMODEL associent à un maillage un domaine physique, un modèle de comportement du matériau et le type d'éléments finis à utiliser.

Exemple :

```
MODL1 = MODEL LIG1 MECANIQUE ELASTIQUE COQ2 ;
```

Le maillage LIG1 est associé à une formulation mécanique, le matériau est élastique isotrope, les éléments finis sont de type coque mince à 2 nœuds.

3.5 Objets EVOLUTION et TABLE

Objet de type EVOLUTION Ce type d'objet permet de définir l'évolution d'une grandeur c'est-à-dire la représentation de f(x) en fonction de x. Les valeurs de x sont définies dans une liste de réels LIS1, les valeurs correspondantes de f(x) sont définies dans une autre liste de réels LIS2 de même taille que LIS1. L'objet de type EVOLUTION est créé avec l'opérateur EVOL suivi du mot-clé MANU.

Exemple :

```
LIS1 = PROG 0. PAS 10. 360. ;
LIS2 = SIN LIS1 ;
EV1 = EVOL 'MANU' 'Abscisses' LIS1 'Ordonnees' LIS2 ;
DESS EV1 ;
```

L'évolution EV1 représente la courbe de sin(x) de 0 à 360^o.

L'opérateur EVOL supporte d'autres mots-clés, notamment :
-COMP : définit une fonction complexe à partir de trois listes de réels.
-CHPO : définit l'évolution d'une composante d'un champ CHPOINT le long d'une ligne de nœuds (qui doit être continue et non fermée).

Objet de type TABLE Pour créer un objet de type TABLE, on utilise l'opérateur TABLE. Ce type d'objet comporte une structure en arborescence. Chaque objet de la table peut être de type quelconque et est caractérisé par un indice quelconque.

Exemple :

```
TAB1 = TABLE ;
TAB1 .FORCE = TABLE ;
TAB1 .FORCE.GRAVITE = CHAM1 ;
TAB1 .FORCE.LOCAL = CHAM2 ;
```

```
$ list TAB1 .FORCE ;
* list TAB1 .FORCE ;
TABLE de pointeur 4741
        Indice                          Objet
     Type    Valeur                Type   Valeur
MOT       GRAVITE               MCHAML        4719
MOT       LOCAL                 MCHAML        4707
```

Chapitre 4

Procédures

Il est possible de faire appel à des procédures au cours de l'exécution d'un programme. L'utilisateur peut créer ses propres procédures ou utiliser celles mises à sa disposition dans le programme Cast3M. Les procédures existantes dans le programme s'utilisent comme des opérateurs élémentaires. Nous allons détailler dans ce chapitre la manière dont l'utilisateur peut créer ses propres procédures.

Les procédures peuvent être définies dans le programme d'exécution ou bien dans un fichier externe.

4.1 Définition d'une procédure pendant l'exécution

Une procédure contient une suite d'instructions élémentaires dont la première est DEBP et la dernière FINP. L'opérateur DEBP doit être suivi du nom de la procédure et de la liste des arguments requis pour son utilisation.

Exemple :

```
DEBP nom_procédure OBJ1*type1 OBJ2/type2 ;
-
opérations élémentaires
-
FINP ;
```

Les objets dont le type est précédé par * sont des données obligatoires au moment de l'appel de la procédure.

Les objets dont le type est précédé par / sont des données facultatives au moment de l'appel de la procédure.

Les objets créés lors de l'exécution de la procédure ne sont pas accessibles à l'extérieur de la procédure. Si l'on désire récupérer des objets créés au cours de l'exécution de la procédure, il faut spécifier la liste des objets générés après l'opérateur FINP.

Exemple :

```
DEBP NORME1 P1*POINT P2*POINT ;
X1 = COOR 1 P1 ; Y1 = COOR 2 P1 ;
X2 = COOR 1 P2 ; Y2 = COOR 2 P2 ;
L2 = ((X2 - X1)**2) + ((Y2 - Y1)**2) ;
L = L2 ** 0.5 ;
FINP L ;
OPTI DIME 2 ;
P1 = 0. 0. ;
P2 = 1. 1. ;
NORM1 = NORME1 P1 P2 ;
```

NORM1 contient la valeur de l'objet passé en argument après l'opérateur FINP.

4.2 Définition d'une procédure dans un fichier externe

L'utilisateur peut mémoriser ses procédures dans un fichier externe. Il est possible d'écrire plusieurs procédures dans le même fichier. Chaque procédure doit commencer par $$$$ suivis du nom de la procédure (en majuscule). Le contenu de la procédure est identique au contenu décrit précédemment, commençant par l'opérateur DEBP et finissant par FINP. Le fichier doit se terminer par $$$$.

Exemple :

```
$$$$ NORM_2D
DEBP NORM_2D ...;
...
FINP ...;
$$$$ NORM_3D
DEBP NORM_3D ...;
...
FINP ...;
$$$$
```

Ce fichier externe contient deux procédures : NORM_2D et NORM_3D.

Il est nécessaire de mettre au bon format le fichier contenant les procédures en créant un fichier d'accès direct. Ce nouveau fichier sera stocké sous le nom de UTILPROC. Pour créer ce fichier UTILPROC, il faut utiliser la directive UTIL(ISATEUR) suivie du mot-clé PROC(EDURE).

Exemple :

```
UTIL PROC 'nom_fichier_externe.proc';
```

Chapitre 5

Maillage

Dans la plupart des pré-processeurs des codes de calcul par éléments finis, le modèle géométrique est créé en deux étapes : dans un premier temps, la définition de la géométrie par des éléments géométriques de base (points, lignes, surfaces, volumes) puis dans un second temps la génération du maillage à partir des géométries créées. Dans Cast3M, un objet géométrique n'existe que sous forme discrétisée. La discrétisation du domaine en éléments s'effectue au moment de la définition de la géométrie. Nous allons présenter dans ce chapitre les principaux opérateurs géométriques qui créent des objets de type MAILLAGE (points, lignes, surfaces, volumes). La démarche générale est la suivante :

- construction des points
- construction des lignes à partir des points
- construction des surfaces à partir des lignes
- construction des volumes à partir des surfaces.

Les objets de type MAILLAGE constituent le support géométrique des éléments finis qui seront définis ultérieurement. Le type des supports géométriques doit donc être en accord avec les éléments finis qui seront utilisés : par exemple, si les éléments sont de type poutre, barre ou coque axisymétrique, les supports géométriques correspondants devront être des SEG2 (segments à 2 nœuds).

5.1 Création des points et des lignes

Afin de rappeler la création des points et des lignes, nous allons commenter l'exemple suivant. Nous allons notamment mettre en évidence les notions de densité. (Remarque : les opérateurs DROITE et de CERCLE peuvent être abrégés respectivement par D et C).

Exemple :

```
OPTI DIME 2 ELEM SEG2;
P0 = 0. 0.;
P1 = 10. 0.;
P2 = 10. 10.;
P3 = 0. 10.;
CEN1 = 5. 10.;
L1 = D 10 P0 P1;
L2 = D P1 P2 'DINI' 1. 'DFIN' 2.;
N1 = NBNO L2; N2 = NBEL L2;
L3 = D -10 P2 P3 'DINI' 2. 'DFIN' 2.;
L4 = C ((-1)*N2) P3 CEN1 P0 'DINI' 2. 'DFIN' 1.;
CONT1 = L1 ET L2 ET L3 ET L4;
TRAC CONT1;
```

-L1 : le nombre d'éléments est spécifié et positif, L1 est donc divisé en 10 éléments d'égale longueur.

-L2 : le nombre d'éléments n'est pas spécifié, L2 est divisé en éléments dont le nombre et la longueur sont calculés en fonction des densités des extrémités. Ici les densités ne sont pas identiques, les éléments sont donc de longueur différente. La taille est calculée en tenant compte des densités des extrémités. L'élément adjacent au point P1 aura une longueur égale à DINI (= 1) et l'élément adjacent au point P2 une longueur égale à DFIN (= 2).
N1 est le nombre de nœuds de L2, obtenu avec l'opérateur NBNO.
N2 est le nombre d'éléments de L2, obtenu avec l'opérateur NBEL.
-L3 : le nombre d'éléments est spécifié et négatif, L3 est divisé en 10 éléments dont la taille est calculée en fonction des densités des extrémités. Ici les densités sont identiques, les éléments sont donc d'égale longueur.
-L4 : le nombre d'éléments est spécifié et négatif. La ligne est un arc de cercle reliant les points P3 et P0 et de centre CEN1.

On aurait pu obtenir le même résultat en affectant aux points P0 et P1 une densité de 1 et aux points P2 et P3 une densité de 2. La création des lignes n'aurait alors pas nécessité les mots-clés DINI et DFIN.

Exemple :

```
OPTI DIME 2 ELEM SEG2 DENS 1.;
P0 = 0. 0.;
P1 = 10. 0.;
DENS 2.;
P2 = 10. 10.;
P3 = 0. 10.;
CEN1 = 5. 10.;
L1 = D -10 P0 P1;
L2 = D P1 P2; N2 = NBEL L2;
L3 = D -10 P2 P3;
L4 = C ((-1)*N2) P3 CEN1 P0;
CONT1 = L1 ET L2 ET L3 ET L4;
TRAC CONT1;
```

Il est possible de nommer un point qui a été créé automatiquement et n'est pas encore nommé. On utilise pour cela l'opérateur POINT. Pour obtenir les coordonnées d'un point, on utilise l'opérateur COOR.

Exemple :

```
PCHAR1 = L1 POIN 5;
PCHAR2 = L1 POIN 'PROC' (6.1 0.);
X2 = COOR 1 PCHAR2;
Y2 = COOR 2 PCHAR2;
X1 Y1 = COOR PCHAR1;
```

-PCHAR1 est le 5ème point de la ligne L1.
-PCHAR2 est le point de la ligne L1 le plus proche du point (6.1 0.).
-X2 est l'abscisse de PCHAR2, Y2 l'ordonnée.
-X1 est l'abscisse de PCHAR1, Y1 l'ordonnée.

Il existe d'autres opérateurs géométriques :

CER3 : construit un arc de cercle passant par trois points

PARA : construit un arc de parabole

CUBP et CUBT : construisent des arcs de cubique

COURBE : crée une courbe polynomiale d'ordre (n-1) à partir de n points

QUELCONQUE : construit une ligne brisée passant par les points spécifiés

INTERSECTION : construit l'arc de courbe, intersection de deux surfaces

5.2 Création de surfaces

Les surfaces sont généralement créées à partir des lignes. Différents opérateurs sont disponibles :

SURFACE et DALLER : construisent une surface à partir des lignes qui constituent le contour de cette surface

REGLER : construit une surface à partir de 2 lignes sur lesquelles la surface s'appuiera.

TRANSLATION et ROTATION : construit une surface par translation ou rotation d'une ligne de type quelconque

5.2.1 Opérateurs SURFACE et DALLER

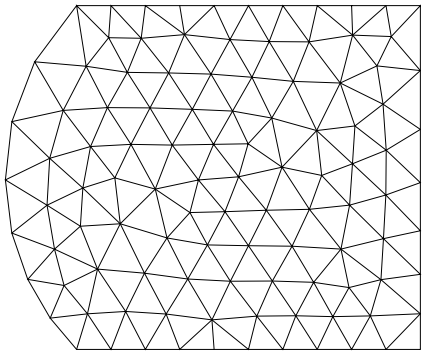
Exemple :

```
OPTI ELEM TRI3 ;
SURF1 = SURF CONT1 'PLAN' ;
TRAC SURF1 ;
OPTI ELEM QUA4 ;
SURF2 = SURF CONT1 'PLAN' ;
SURF3 = DALL L1 L2 L3 L4 ;
TRAC SURF2 ;
TRAC SURF3 ;
```

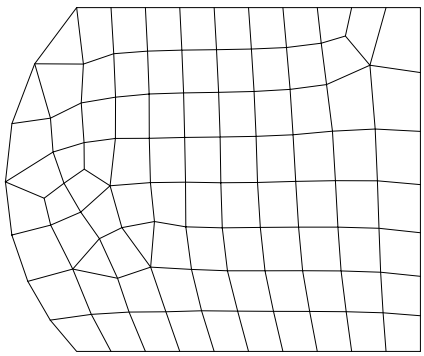
L'opérateur SURFACE construit une surface à partir d'un contour **fermé**. Les éléments utilisés sont ceux spécifiés dans la directive OPTI. On peut remarquer qu'avec les éléments QUA4, le mailleur automatique insère tout de même des éléments TRI3, le maillage obtenu n'est donc pas très satisfaisant.

L'opérateur DALLER permet d'obtenir un maillage régulier même avec des QUA4. Il est nécessaire que les côtés opposés aient le même nombre d'éléments.

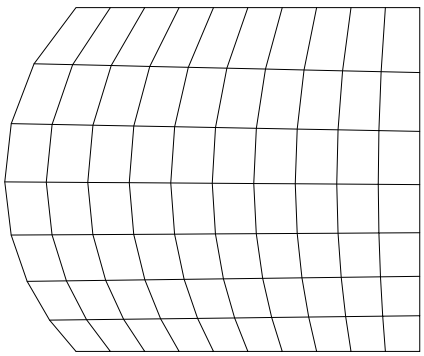
Les côtés définissant le contour doivent être orientés dans le même sens.



SURF1



SURF2

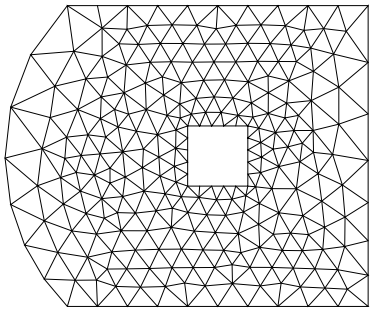


SURF3

On peut utiliser l'opérateur SURFACE pour créer une surface comportant des contours extérieurs et intérieurs (délimitant des trous par exemple). Les contours intérieurs doivent tourner dans le sens opposé des contours extérieurs.

Exemple :

```
OPTI ELEM TRI3 ;
P4 = 4. 4. ; P5 = 6. 4. ;
P6 = 6. 6. ; P7 = 4. 6. ;
L5 = D 5 P4 P7 ;
L6 = D 5 P7 P6 ;
L7 = D 5 P6 P5 ;
L8 = D 5 P5 P4 ;
CONTINT1 = L5 ET L6 ET L7 ET L8 ;
CONTTOT = CONT1 ET CONTINT1 ;
SURF2B = SURF CONTTOT 'PLAN' ;
TRAC SURF2B ;
```



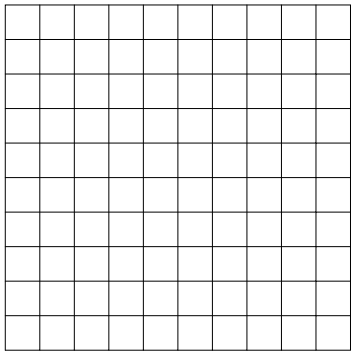
SURF2B

5.2.2 Opérateur REGLER

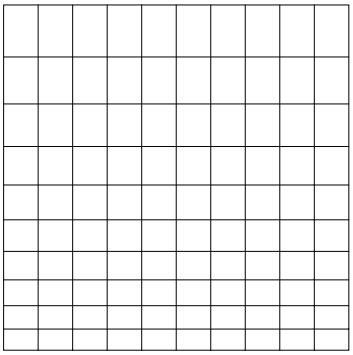
Exemple :

```
L3I = INVE L3 ;
SURF4 = L1 REGLER 10 L3I ;
SURF5 = L1 REGLER -10 L3I 'DINI' 1. 'DFIN' 2. ;
TRAC SURF4 ; TRAC SURF5 ;
```

L'opérateur REGLER construit une surface réglée s'appuyant sur 2 lignes qui doivent être décrites dans le même sens. Afin d'orienter la ligne L3 dans le même sens que la ligne L1, on utilise l'opérateur INVE qui inverse le sens de la ligne. Il est possible de spécifier le nombre de couches d'éléments générés (ici 10). Selon le même principe que la création des lignes, on peut tenir compte des densités en donnant un nombre de couches négatif.



SURF4



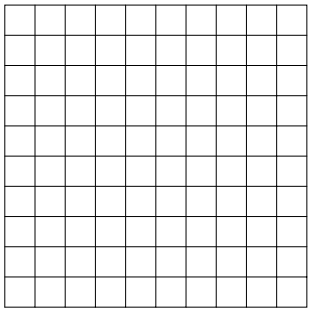
SURF5

5.2.3 Opérateurs TRANSLATION et ROTATION

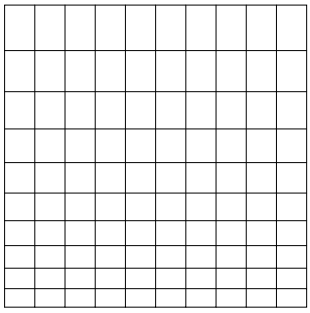
Exemple :

```
VEC2 = 0. 10. ;
PR1 = -5. 0. ;
SURF6 = L1 TRAN 10 VEC2 ;
SURF7 = L1 TRAN -10 'DINI' 1. 'DFIN' 2. VEC2 ;
SURF8 = L1 ROTA 10 30 PR1 ;
TRAC SURF6 ;
TRAC SURF7 ;
TRAC SURF8 ;
```

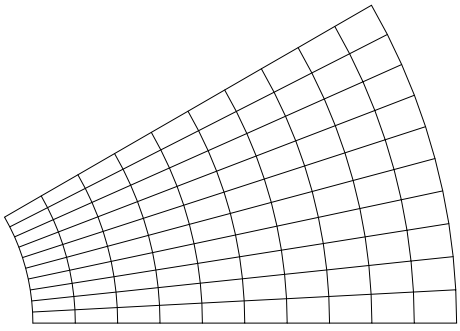
L'opérateur TRANSLATION construit la surface engendrée par la translation d'une ligne suivant un vecteur donné (ici VEC2).
L'opérateur ROTATION construit la surface engendrée par la rotation d'une ligne d'un angle donné (30^o) autour d'un point donné (PR1).
Dans les deux cas, on peut indiquer le nombre de couches engendrées (ici 10) et on peut tenir compte des densités en donnant un nombre de couches négatif.



SURF6



SURF7



SURF8

5.3 Création de volumes

Les volumes sont généralement créés à partir des surfaces. Différents opérateurs sont disponibles :

VOLUME : différentes options existent avec cet opérateur.

PAVER : construit un volume à partir d'une enveloppe parallélépipédique.

5.3.1 Opérateur VOLUME

Exemple :

```
OPTI DIME 3 ELEM CUB8 ;
VEC3 = 0. 0. 10. ;
PR2 = -5. 10. 0. ;
SURF9 = SURF6 PLUS VEC3 ;
VOL1 = SURF6 VOLU 10 'TRANS' VEC3 ;
VOL2 = SURF6 VOLU 10 'ROTA' 30. PR1 PR2 ;
VOL3 = SURF6 VOLU SURF9 ;
TRAC CACH VOL1 TITRE 'VOL1' ;
TRAC CACH VOL2 TITRE 'VOL2' ;
TRAC CACH VOL3 TITRE 'VOL3' ;
```

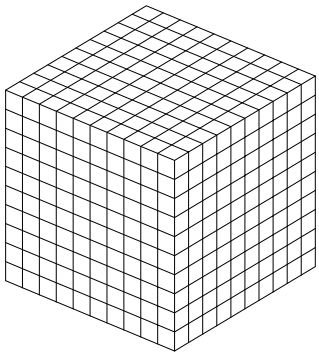
L'option TRANS permet de créer un volume par la translation d'une surface selon un vecteur donné (VEC3).

L'option ROTA permet de créer un volume par la rotation d'une surface selon un angle donné et autour de l'axe défini par 2 points (PR1 et PR2).

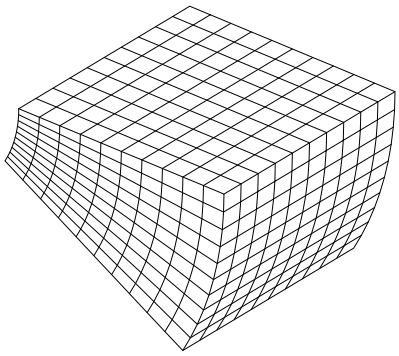
L'opérateur VOLUME utilisé sans mot-clé crée le volume en reliant deux surfaces données (qui doivent être obligatoirement homéomorphes).

Il est possible de spécifier le nombre de couches engendrées (ici 10), on peut aussi tenir compte des densités selon le même principe que les lignes et les surfaces.

L'option CACH de la directive TRAC permet de n'afficher que les parties apparentes de l'objet.



VOL1 et VOL3



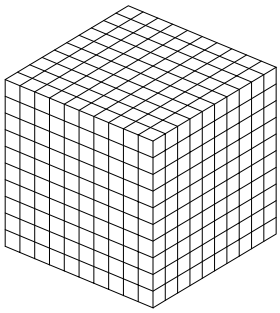
VOL2

5.3.2 Opérateur PAVER

L'exemple suivant illustre l'utilisation de l'opérateur PAVER qui permet de mailler avec des cubes l'intérieur de volumes parallélépipédiques. Il est nécessaire de préciser les 6 faces de l'enveloppe dans un ordre spécifique : les faces opposées doivent se suivre dans la liste et elles doivent être décrites de façon identique.

Exemple :

```
OPTI DIME 3 ELEM CUB8 DENS 1.;
PP1 = 0. 0. 0.; PP2 = 10. 0. 0.;
PP3 = 10. 10. 0.; PP4 = 0. 10. 0.;
PP5 = 0. 0. 10.; PP6 = 10. 0. 10.;
PP7 = 10. 10. 10.; PP8 = 0. 10. 10.;
LL1 = D PP1 PP2; LL2 = D PP2 PP3;
LL3 = D PP3 PP4; LL4 = D PP4 PP1;
LL5 = D PP5 PP6; LL6 = D PP6 PP7;
LL7 = D PP7 PP8; LL8 = D PP8 PP5;
S1 = DALL LL1 LL2 LL3 LL4; S2 = DALL LL5 LL6 LL7 LL8;
LL9 = D PP2 PP6; LL5I = INVE LL5;
LL10 = D PP5 PP1;
S3 = DALL LL1 LL9 LL5I LL10;
LL3I = INVE LL3;
LL11 = D PP3 PP7; LL12 = D PP8 PP4;
S4 = DALL LL3I LL11 LL7 LL12;
LL4I = INVE LL4; LL12I = INVE LL12;
S5 = DALL LL4I LL12I LL8 LL10;
LL6I = INVE LL6; LL9I = INVE LL9;
S6 = DALL LL2 LL11 LL6I LL9I;
V1 = PAVE S1 S2 S3 S4 S5 S6;
TRAC CACH V1;
```



V1

5.4 Récapitulation des opérateurs de maillage

type d'objet	**Opérateurs**
POINTS	=, PLUS, MOINS, POINT, BARYCENTRE
LIGNES	**à partir de points :** DROITE, CERCLE, CER3, CONGE, COURBE, CUBP, CUBT, PARABOLE, INTERSECTION, QUELCONQUE **à partir de lignes :** PLUS, MOINS, AFFINITE, HOMOTHETIE, INVERSE, PROJECTION, SYMETRIE, TOURNER, COMPRIS, ELEMENT **à partir de maillages complexes :** CONTOUR, ARETE, COTE
SURFACES	**à partir de lignes :** TRANS, ROTATION, SURFACE, COUTURE, REGLER, GENERATRICE, DEDUIRE, DALLER **à partir de surfaces :** PLUS, MOINS, PROJECTION, AFFINITE, HOMOTHETIE, SYMETRIE, ORIENTE, TOURNER, REGENERER, INCLUS, ELEMENT **à partir de volumes :** ENVELOPPE, FACE
VOLUMES	**à partir de surfaces :** VOLUME TRANS, VOLUME ROTA, PAVER **à partir de volumes :** REGENERER, AFFINITE, PLUS, MOINS, HOMOTHETIE, TOURNER, SYMETRIE

5.5 Exemple : Maillage d'un cylindre creux

Il existe de nombreuses façons de réaliser un maillage. Voici deux exemples de maillage d'un cylindre creux, le premier compliqué, le second plus simple.

EXEMPLE 1

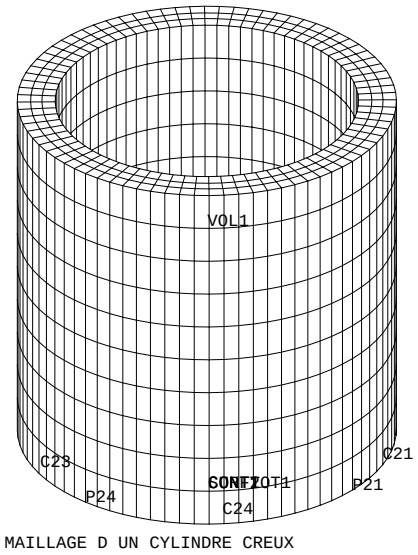
```
OPTI DIME 3 ELEM CUB8 ;
************************************************************************
*POINTS
H1 = 5.  ; COMM Hauteur du cylindre ;
R1 = 2.  ; COMM Rayon interieur ;
R2 = 2.5 ; COMM Rayon exterieur ;
O1 = 0. 0. 0. ; O2 = 0. 0. H1 ; COMM Points de l axe du cylindre ;
P11 =     R1  0. 0. ;  P12 = 0.   R1    0. ;
P13 = ((-1.)*R1) 0. 0. ;  P14 = 0. ((-1.)*R1) 0. ;
P21 =     R2  0. 0. ;  P22 = 0.   R2    0. ;
P23 = ((-1.)*R2) 0. 0. ;  P24 = 0. ((-1.)*R2) 0. ;

************************************************************************
*LIGNES
*Contour interieur de la base
C11 = C 20 P11 O1 P12 ;
C12 = C 20 P12 O1 P13 ;
C13 = C 20 P13 O1 P14 ;
C14 = C 20 P14 O1 P11 ;
CONT1 = C11 ET C12 ET C13 ET C14 ;

*Contour exterieur de la base
C21 = C 20 P22 O1 P21 ;
C22 = C 20 P23 O1 P22 ;
C23 = C 20 P24 O1 P23 ;
C24 = C 20 P21 O1 P24 ;
CONT2 = C21 ET C22 ET C23 ET C24 ;

CONTTOT1 = CONT1 ET CONT2 ;

************************************************************************
*SURFACE
SURF1 = SURF CONTTOT1 'PLAN' ;

************************************************************************
*VOLUME
VOL1 = SURF1 VOLU 10 TRANS (0. 0. H1) ;

************************************************************************
*TRACAGE
TITRE 'MAILLAGE D UN CYLINDRE CREUX' ;
TRAC CACH QUAL VOL1 ;
FIN;
```

Remarques sur le fichier d'exemple 1

– OPTI DIME 3 ELEM CUB8 ;
  On travaille en trois dimensions en utilisant des éléments CUB8. En fait, l'option ELEM permet de définir la classe d'éléments (linéaires ou quadratiques) en donnant l'élément de niveau le plus complexe. Ainsi CUB8 autorise autorise aussi les éléments POI1, SEG2, TRI3, QUA4, TET4, PYR5 et PRI6. La

définition de CUB8 en début de fichier évite de préciser à chaque étape (lignes, surfaces et volume) le nouveau type d'éléments.

– COMM Hauteur du cylindre ;
  La directive COMM(ENTAIRE) permet d'introduire des commentaires dans le jeu de données. Le commentaire prend fin au point-virgule. Cette directive est équivalente à un astérisque placé en début de ligne.
– Le contour CONT2 doit tourner dans le sens opposé de CONT1 (dans l'optique de l'utilisation de SURFACE avec des contours extérieur et intérieur).
– TRAC CACH QUAL VOL1 ;
  L'option QUAL de la directive TRAC permet d'afficher sur le dessin le nom des entités qui y figurent.



EXEMPLE 2

```
OPTI DIME 3 ELEM CUB8 ;
************************************************************************
*POINTS
H1 = 5.  ; COMM Hauteur du cylindre ;
R1 = 2.  ; COMM Rayon interieur ;
R2 = 2.5 ; COMM Rayon exterieur ;
O1 = 0. 0. 0. ; O2 = 0. 0. H1 ; COMM Points de l axe du cylindre ;
P11 =  R1 0. 0. ;
P21 =  R2 0. 0. ;

************************************************************************
L1   = P11 D 3 P21 ;
SEC1 = L1 TRANS 10 (0. 0. H1) ;
VO1  = SEC1 VOLU 40 ROTA 180 O1 O2 ;
VO2  = VO1 VOLU 40 ROTA 180 O1 O2 ;
VOL1 = VO1 ET VO2 ;
ELIM 0.001 VOL1 ;

************************************************************************
*TRACAGE
TITRE 'MAILLAGE D UN CYLINDRE CREUX' ;
TRAC CACH VOL1 ;

FIN;
```

Remarques sur le fichier d'exemple 2

– ELIM 0.01 VOL1 ;
  La directive ELIM(INATION) remplace dans l'objet VOL1 tous les nœuds distants de moins de 0.001 par un seul point. Elle est utilisée pour éliminer les nœuds doubles créés par la construction du volume VO2 avec l'opérateur VOLUME ROTA. En effet, VO1 et VO2 ont des faces communes mais qui sont indépendantes et ont leurs propres nœuds. Ainsi il existe des nœuds doubles qu'il est nécessaire de fusionner.
  Cette directive est à utiliser avec précaution : s'il existe dans un champ par point des valeurs différentes pour les nœuds doubles, il sera difficile de savoir quelle valeur garder.
– Parfois une démarche simple est écartée par le souci de nommer au fur et à mesure des objets dont on aura besoin pour l'application des conditions aux limites. Dans cet exemple, on peut donc regretter de ne pas avoir nommé les points du plan XOY si on veut encastrer le cylindre. Dans ce cas il faut les nommer après la réalisation du maillage.

  Exemple :

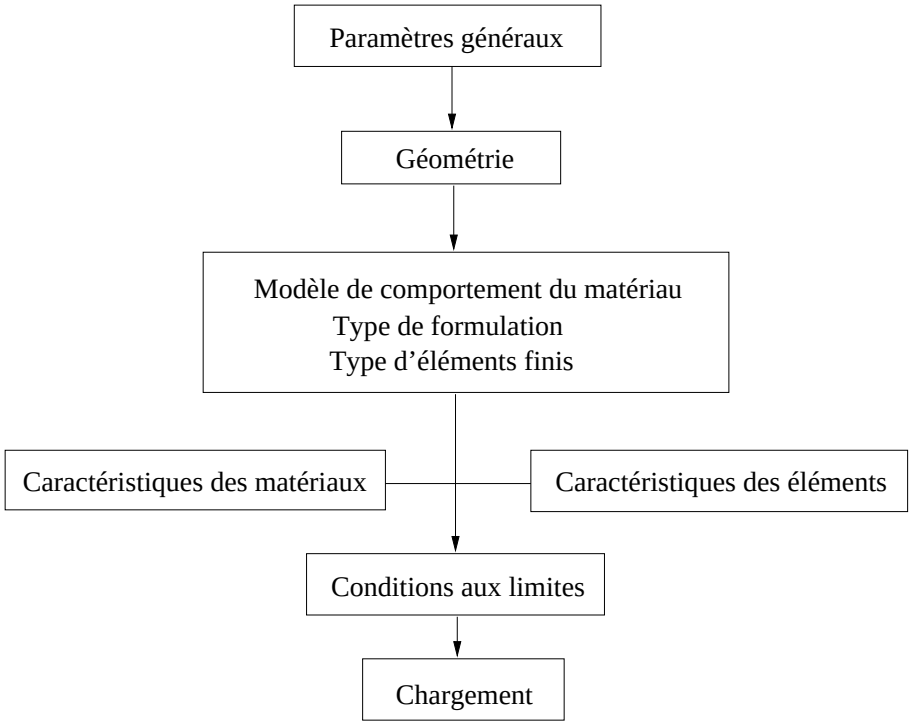
  ```
  P12 = 0. R1 0. ;
  PPE = VOL1 POINT PLAN O1 P11 P12 0.001 ;
  ENC1 = BLOQUER DEPLA ROTA PPE ;
  ```

  L'opérateur POINT suivi du mot-clé PLAN permet d'extraire de l'objet VOL1 l'ensemble des points contenus dans le plan défini par les trois points O1, P11 et P12.

Chapitre 6

Préparation du modèle de calcul

Le modèle de calcul représente l'ensemble des données que doit préparer l'utilisateur pour décrire les caractéristiques complètes du problème à analyser. Il peut être décrit par le schéma suivant :



- Les paramètres généraux sont définis avec la directive OPTI (cf. 2.3).
- La géométrie représente la forme discrétisée du domaine à étudier. Elle est composée d'objets de type MAILLAGE (cf. 5).
- Le modèle associe au maillage une loi de comportement du matériau, la formulation éléments finis ainsi que le type d'éléments utilisés. C'est un objet de type MMODEL.
- Les caractéristiques des matériaux définissent les propriétés physiques des matériaux : module d'Young, coefficient de Poisson, masse volumique, etc...Ces données sont rassemblées dans un objet de type MCHAML à plusieurs composantes (YOUN, NU, RHO...)
- Les caractéristiques des éléments sont relatives à certaines données supplémentaires selon les types d'éléments choisis qui ne peuvent se déduire de la géométrie. Par exemple, pour les coques il est nécessaire d'indiquer leur épaisseur, pour les poutres leur section et leurs différentes inerties. Ces données sont rassemblées dans un objet de type MCHAML à plusieurs composantes (SECT, INRY, EPAI, TORS...). Ce

champ peut être concaténé avec le champ de caractéristiques des matériaux.
- Les conditions limites de type blocage doivent s'ajouter à la matrice de rigidité. Ce sont des objets de type RIGIDITE. S'il s'agit de conditions imposées (température, déplacements...), elles doivent de plus s'ajouter au second membre de l'équation à résoudre. Ce sont des objets de type CHPOINT.
- Le chargement est un objet de type CHPOINT ou CHARGEMENT (dans le cas de description spatiale et temporelle).

6.1 Modèle

Le modèle associe à un objet de type MAILLAGE un domaine physique, une loi de comportement du matériau et la formulation éléments finis. C'est l'opérateur MODE(LISER) qui crée l'objet de type MMODEL.

Exemple :

```
MOD1 = MODE GEO1 MECANIQUE ELASTIQUE ISOTROPE POUT ;
MOD2 = MODE GEO2 MECANIQUE PLASTIQUE ISOTROPE POUT ;
MOD3 = MODE GEO3 MECANIQUE ELASTIQUE ORTHOTROPE COQ4 ;
MOD4 = MODE GEO4 THERMIQUE ISOTROPE QUA8 ;
MOD5 = MODE GEO5 LIQUIDE MECANIQUE RACO ;
```

Chaque modèle est défini en affectant à une géométrie (GEOi) une formulation (MECANIQUE, THERMIQUE, LIQUIDE MECANIQUE), une loi de comportement du matériau et un type d'éléments finis.

Les quatre premiers modèles ont une formulation simple (MECANIQUE ou THERMIQUE), le cinquième a une formulation couplée (LIQUIDE MECANIQUE).

Le comportement du matériau de MOD1 et MOD3 est linéaire (ELASTIQUE), celui du matériau de MOD2 est non linéaire (PLASTIQUE).

La formulation LIQUIDE MECANIQUE ne nécessite pas de matériau, le type d'éléments finis RACO correspond à un raccord liquide-coque.

Le type des éléments finis dépend de la formulation, du type de support géométrique et de l'option de calcul choisie. Par exemple l'élément POUT est supporté par des SEG2 en formulation MECANIQUE avec une option de calcul tridimensionnel.

6.1.1 Liste des différentes formulations

-MECANIQUE
-LIQUIDE
-THERMIQUE
-CONVECTION
-POREUX
-DARCY
-FROTTEMENT
-RAYONNEMENT
-LIQUIDE MECANIQUE

Les noms des formulations et des modèles de comportement doivent être écrits en toute lettre.

6.2 Matériaux et caractéristiques des éléments

Selon le type de calcul et les éléments finis utilisés, il est nécessaire de définir certaines propriétés matérielles et géométriques. On utilise l'opérateur MATE(RIAU) pour définir les propriétés matérielles d'un modèle donné. Pour les propriétés géométriques, on peut utiliser aussi l'opérateur MATE ou bien l'opérateur CARA(CTERISTIQUE). Les objets créés par les deux opérateurs sont de type MCHAML à plusieurs composantes : YOUN, NU, RHO, EPAI... Les caractéristiques peuvent être constantes ou variables selon un paramètre.

Exemple :

```
MAT1 = MATER MOD1 YOUN 2.E11 NU 0.3 RHO 7850. ;
CAR1 = CARAC MOD1 SECT 0.5 INRY 0.4 INRZ 0.4 TORS 1. ;
MATTOT1 = CAR1 ET MAT1 ;
MATTOT2 = MATER MOD1 YOUN 2.E11 NU 0.3 RHO 7850. SECT 0.5 INRY 0.4 INRZ 0.4 TORS
1. ;
EVYOU1 = EVOL MANU 'YOUN' (PROG 2.2E11 2.E11 1.8E11) 'TEMP' (PROG 20. 100. 200.) ;
MATTOT3 = MATER MOD1 YOUN EVYOU1 NU 0.3 ;
```

Les deux objets MATTOT1 et MATTOT2 sont équivalents.
Si l'on définit les propriétés matérielles séparément des propriétés géométriques, il faut concaténer ensuite les deux champs (MAT1 ET CAR1).
Dans MATTOT3, le module d'Young est décrit par une évolution donnant la composante YOUN en fonction du paramètre TEMP.

6.3 Conditions limites et chargement

Conditions limites

Les conditions limites sont traitées dans Cast3M par la méthode des multiplicateurs de Lagrange. Elles s'écrivent sous la forme : $\overline{\overline{C}}.\overline{u} = \overline{q}$
Elles sont prises en compte dans le système d'équations linéaires d'équilibre en résolvant :

$$\begin{cases} \overline{\overline{K}}.\overline{u} + \overline{\overline{C}}^{T}.\overline{\lambda} = \overline{F} \\ \overline{\overline{C}}.\overline{u} = \overline{q} \end{cases}$$

L'utilisateur doit donc construire deux objets :
- La rigidité $\overline{\overline{C}}$ à adjoindre à la rigidité $\overline{\overline{K}}$ du système libre grâce à l'opérateur BLOQUER.
- Le vecteur $\overline{q}$ à adjoindre au vecteur des forces nodales $\overline{F}$ grâce à l'opérateur DEPI(MPOSE). Par défaut ce vecteur est mis à zéro.

Il existe d'autres opérateurs permettant de traiter les conditions limites, notamment :
-SYMT (ou ANTI) : permet de définir des relations de symétrie (ou antisymétrie) par rapport à une droite ou un plan.
- RELA : permet d'imposer une combinaison linéaire entre des déplacements en différents points.

Chargement

La définition du chargement consiste à créer un champ par points correspondant au vecteur du second membre de l'équation : $\overline{\overline{K}}.\overline{u} = \overline{F}$.

Il existe des opérateurs spécifiques permettant de définir un chargement (opérateurs FORCE, MOMENT, PRESSION...). Par contre, il n'y a pas d'opérateurs spécifiques pour certains cas de chargement courants, notamment pour le poids propre et le chargement thermique. Les exemples des parties 10 et 11 présentent des méthodes permettant d'appliquer ces deux types de chargement.

Exemple :

```
CL1 = BLOQUER DEPLA ROTA P1 ;
CL2 = BLOQUER UX UY L1 ;
CL3 = SYMT DEPLA O1 O2 O3 P6 1.E-3 ;
CL4 = RELA 1 UZ P3 - 2 UZ P4 ;
CLTOT = CL1 ET CL2 ET CL3 ET CL4 ;
*
DEP1 = DEPI CL4 0.5 ;
F1 = FORCE FZ 1000. P5 ;
FTOT = DEP1 ET F1 ;
```

- L'opérateur BLOQUER bloque un ou des degrés de liberté d'un objet géométrique. CL1 indique

que le point P1 est encastré, CL2 indique que seuls les déplacements UX et UY de la ligne L1 sont bloqués.
- CL3 permet de définir une relation de symétrie de tous les déplacements du point P6 par rapport au plan défini par les points O1, O2 et O3.
- CL4 permet de définir une combinaison linéaire entre les déplacements UZ des points P3 et P4.
- DEP1 permet d'imposer une valeur à la rigidité CL4 à savoir : UZ(P3) - 2 UZ(P4) = 0.5
- F1 définit une force FZ d'amplitude 1000 appliquée sur le point P5.
- FTOT représente le second membre de l'équation, cet objet regroupe la force appliquée au système ainsi que le vecteur des déplacements imposés.

Chapitre 7

Résolution d'un calcul

Une fois réalisée la préparation du modèle de calcul, on peut constituer le système $\overline{\overline{M}} \cdot \overline{\ddot{u}} + \overline{\overline{C}} \cdot \overline{\dot{u}} + \overline{\overline{K}} \cdot \overline{u} = \overline{F}$ et le résoudre. Il faut donc dans un premier temps construire les matrices de masse, d'amortissement et de rigidité. L'opérateur RIGI(DITE) permet de construire la matrice de rigidité, AMOR(TISSEMENT) la matrice d'amortissement et MASSE la matrice de masse.

On utilise l'opérateur RESO(UDRE) pour résoudre un système linéaire, l'opérateur VIBR(ATION) pour effectuer une analyse modale. Nous allons illustrer l'utilisation de ces deux opérateurs par deux exemples simples :
- calcul de la flèche d'une poutre encastrée-libre (cf. 7.2)
- calcul des modes propres d'une plaque carrée (cf. 7.3).

7.1 Construction des matrices

Exemple :

```
RIG1 = RIGI MOD1 MATTOT1 ;
MAS1 = MASS MOD1 MATTOT1 ;
AMO1 = AMOR TAB1 LREEL1 ;
```

- L'opérateur RIGI construit la matrice de rigidité à partir du modèle et des caractéristiques matérielles et géométriques. Il peut aussi créer des raideurs additionnelles.

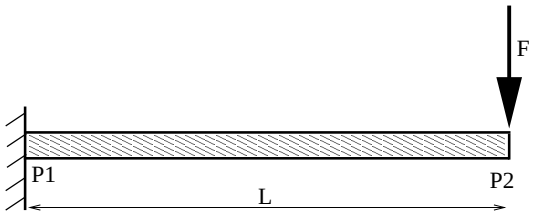
- L'opérateur MASSE construit la matrice de masse à partir du modèle et des caractéristiques matérielles et géométriques. Il peut aussi créer des masses additionnelles.

- L'opérateur AMOR construit une matrice diagonale d'amortissements modaux en affectant à chaque mode de la base (données contenues dans TAB1 de type TABLE et sous-type BASE_MODALE) un amortissement réduit (données contenues dans LREEL1 de type LISTREEL).

7.2 Analyse statique linéaire

Afin d'illustrer l'enchaînement d'un calcul, nous allons calculer la flèche d'une poutre encastrée-libre avec une force fléchissante appliquée sur l'extrémité libre.

- $F = 1000N$
- $L = 1m$
- $\phi = 0,1m$
- $\rho = 7800kg/m^3$
- $E = 2,1.10^{11}Pa$



Données du problème

Fichier de données

```
OPTI DIME 3 ELEM SEG2 OPTI TRID ;

*DONNEES
AMPF = -1000. ;
LON1 = 1. ;
DIA1 = 0.1 ;
RHOP = 7800. ;
NUP = 0.3 ;
YOP = 2.1E11 ;


*CALCUL DES DONNEES GEOMETRIQUES
S1 = PI*(DIA1**2)/4. ;
IY = PI*(DIA1**4)/64. ;
IZ = IY ;
IG = PI*(DIA1**4)/32. ;

*GEOMETRIE
P1 = 0. 0. 0.;
P2 = LON1 0. 0.;
L1 = D 10 P1 P2 ;

*DEFINITION DU MODELE ET DU MATERIAU
MOD1 = MODEL L1 MECANIQUE ELASTIQUE ISOTROPE POUT ;
MAT1 = MATER MOD1 YOUN YOP NU NUP RHO RHOP ;
CAR1 = CARAC MOD1 SECT S1 INRY IY INRZ IZ TORS IG ;
MATTOT = MAT1 ET CAR1 ;

*CONDITIONS LIMITES
CL1 = BLOQ DEPLA ROTA P1 ;

*CHARGEMENT
FOR1 = FORCE FY AMPF P2 ;

*MATRICE DE RIGIDITE
RIG1 = RIGI MOD1 MATTOT ;
RIG1 = RIG1 ET CL1 ;

*RESOLUTION
```

```
RES1 = RESO RIG1 FOR1 ;

*RESULTATS
DY = EXCO RES1 UY ;
DYABS = ABS DY ;
DYMAX = MAXI DYABS ;
MESS 'FLECHE MAXIMALE EN METRES (F=1000N) :'DYMAX ;

*DEFORMEE
DEF0 = DEFO L1 RES1 0. VERT ;
DEF1 = DEFO L1 RES1  ROUGE ;
OEIL1 = 0. 0. 1000. ;
TRAC OEIL1 (DEF0 ET DEF1) ;

FIN;
```

Commentaires

`OPTI DIME 3 ELEM SEG2 OPTI TRID ;`
La dimension requise pour travailler avec des éléments de type poutre est 3 (option de calcul tridimensionnelle).

```
P1 = 0. 0. 0. ;
P2 = LON1 0. 0. ;
L1 = D 10 P1 P2 ;
```
La géométrie est décrite par une ligne L1 divisée en 10 éléments d'égale longueur.

```
MOD1 = MODEL L1 MECANIQUE ELASTIQUE ISOTROPE POUT ;
MAT1 = MATER MOD1 YOUN YOP NU NUP RHO RHOP ;
CAR1 = CARAC MOD1 SECT S1 INRY IY INRZ IZ TORS IG ;
MATTOT = MAT1 ET CAR1 ;
```
L'objet MOD1 de type MMODEL définit une loi de comportement mécanique élastique et affecte au maillage L1 des éléments de type poutre.
Le matériau MAT1 définit les propriétés du matériau.
Les caractéristiques de CAR1 complètent les données géométriques ne pouvant se déduire du maillage.
MAT1 et CAR1 sont rassemblés dans le même objet. Il est possible de les créer simultanément mais la lecture des données s'en trouve alourdie.

```
CL1 = BLOQ DEPLA ROTA P1 ;
```
Le point P1 est encastré. Tous ses déplacements et rotations sont bloqués ; il n'est pas nécessaire d'adjoindre le vecteur $\overline{q}$ au vecteur second membre de l'équation car $\overline{q}$ est mis à zéro par défaut.

```
FOR1 = FORCE FY AMPF P2 ;
```
Le second membre de l'équation est défini par le champ par point créé par FORCE. On applique ici une force de composante FY et d'amplitude AMPF. On aurait pu aussi écrire :
`FOR1 = FORCE (0. AMPF 0.) P2 ;`

```
RIG1 = RIGI MOD1 MATTOT ;
RIG1 = RIG1 ET CL1 ;
```
On construit la matrice de rigidité à partir du modèle et de ses caractéristiques. On adjoint ensuite la matrice de blocage $\overline{C}$ (CL1) à la matrice de rigidité.

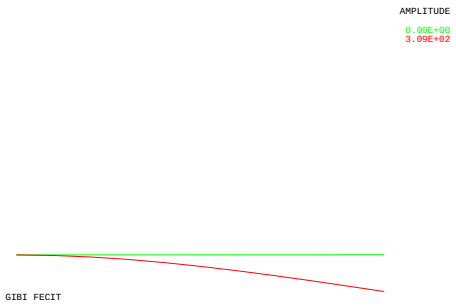
```
RES1 = RESO RIG1 FOR1 ;
```
On résout le système $\overline{\overline{K}}.\overline{u} = \overline{F}$. L'objet RES1 est de type CHPOINT, il contient les déplacements $\overline{u}$.

```
DY = EXCO RES1 UY ;
DYABS = ABS DY ;
DYMAX = MAXI DYABS ;
MESS 'FLECHE MAXIMALE EN METRES (F=1000N) :'DYMAX ;
```

On crée le champ DY de même type que RES1 en extrayant la composante UY avec l'opérateur EXCO. On crée ensuite le champ DYABS à partir des valeurs absolues de DY avec l'opérateur ABS. On extrait ensuite la valeur maximale du champ DYABS avec l'opérateur MAXI.
On aurait pu écrire directement `DYMAX = MAXI (ABS DY)` pour obtenir la valeur absolue maximale du champ DY.

```
DEF0 = DEFO L1 RES1 0. VERT ;
DEF1 = DEFO L1 RES1 ROUGE ;
OEIL1 = 0. 0. 1000. ;
TRAC OEIL1 (DEF0 ET DEF1) ;
```

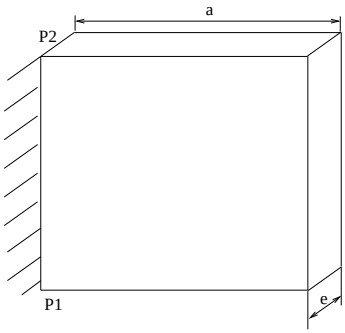
L'opérateur DEFO(RME) permet de construire la déformée d'une structure à partir de la géométrie initiale et du champ de déplacements. Il est possible de préciser la couleur ou le facteur d'amplification. Ici DEF0 est affecté d'un facteur 0 afin de visualiser la structure non déformée.
Pour visualiser les deux objets de type DEFORME, on utilise la directive TRAC en précisant le point de vue OEIL1 selon lequel on doit tracer les déformées.



7.3 Analyse modale

Pour effectuer une analyse modale, on utilise l'opérateur VIBR(ATION). Nous allons calculer les deux premiers modes propres d'une plaque carrée encastrée d'un côté afin d'illustrer l'utilisation de cet opérateur.

-$a = 0,4m$
-$e = 0,006m$
-$\rho = 4200kg/m^3$
-$E = 7.10^{10}Pa$
-$\nu = 0,3$



Données du problème

Fichier de données

```
OPTI DIME 3 ELEM QUA4;

A1 = 0.4 ;
PI= 3.1415927 ;

*MAILLAGE
P1 = 0.  0.  0. ; P2 = 0. A1  0. ;
VEC1 = A1 0. 0. ;
L1 = D 12 P1 P2 ;
S1 = L1 TRAN 12 VEC1 ;

*MODELE ET PROPRIETES DE MATERIAUX
MOD1= MODEL S1 MECANIQUE ELASTIQUE COQ4;
MAT1= MATER MOD1 YOUN 7.E10 NU 0.3 RHO 4200. ;
CAR1= CARAC MOD1 EPAI 0.006 ;
MATTOT = MAT1 ET CAR1 ;

*CONDITIONS AUX LIMITES
CL1= BLOQ DEPLA ROTA L1 ;

*MATRICES DE RIGIDITE ET DE MASSE
RIG1= RIGI MOD1 MATTOT ; RIG1 = RIG1 ET CL1 ;
MAS1= MASS MOD1 MATTOT ;

*CALCUL DES MODES
TAB1= VIBR INTER 0. 100. MAS1  RIG1 IMPR TBAS ;
TAB2= TAB1.modes ;

*RESULTATS
F1= (TAB2.1).frequence ;
F2= (TAB2.2).frequence ;
MESS 'FREQUENCE 1 (HZ) :' F1;
MESS 'FREQUENCE 2 (HZ) :' F2;

*TRACE DES DEFORMEES MODALES
```

```
DEP1= (TAB2.1).deformee_modale ; DEF1= DEFO DEP1 S1 1. ROUGE;
DEP2= (TAB2.2).deformee_modale ; DEF2= DEFO DEP2 S1 1. VERT ;
DEF0= defo DEP1 S1 0.;
TITRE 'Deformees modales (MODE 1)' ;
TRAC  (DEF0 ET DEF1);
TITRE 'Deformees modales (MODE 2)' ;
TRAC  (DEF0 ET DEF2);

FIN;
```

Commentaires

`TAB1= VIBR INTER 0. 100. MAS1 RIG1 IMPR TBAS ;`

`TAB2= TAB1.modes ;`

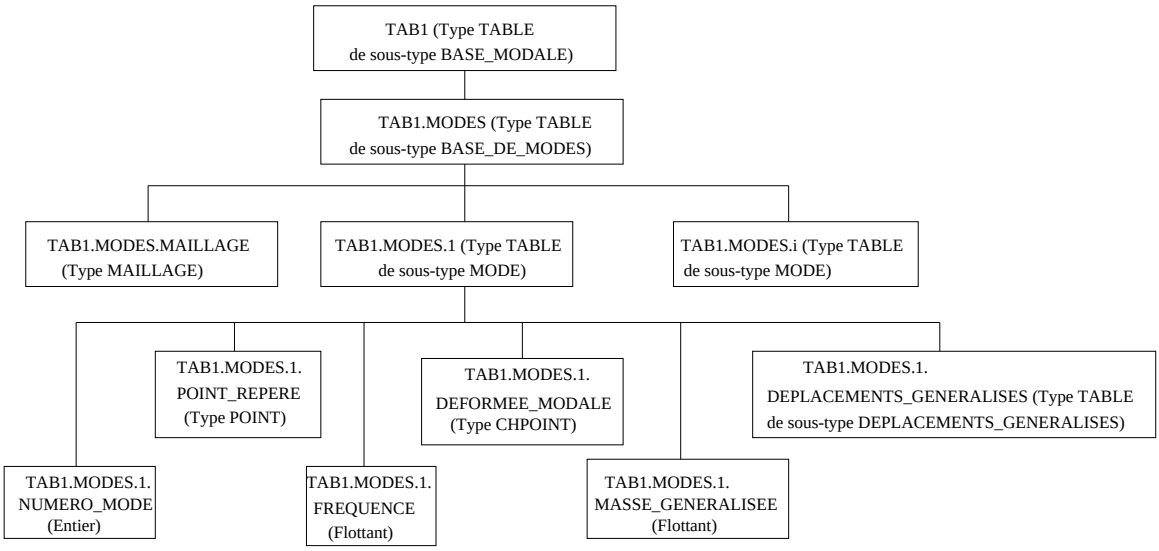
L'opérateur VIBR recherche les valeurs et modes propres du système : $\overline{\overline{K}} - \overline{\omega^2}.\overline{\overline{M}} = 0$ (ici $\overline{\overline{K}}$ = RIG1 et $\overline{\overline{M}}$ = MAS1).

L'option INTER(VALLE) calcule les modes propres dont les fréquences sont contenues dans un intervalle donné (ici [0.,100.]).

Le mot-clé IMPR indique que des messages de calcul seront affichés pendant le déroulement de la procédure.

Le mot-clé TBAS indique que l'objet TAB1 sera de type TABLE et de sous-type BASE_MODALE (sinon il serait de type SOLUTION).

Organisation de la table de sous-type BASE_MODALE :



Pour simplifier le post-traitement des résultats, on crée TAB2 qui contient la table TAB1.MODES. Elle aura pour sous-type BASE_DE_MODES et pour indices : MAILLAGE et IMOD (entier variant de 1 au nombre de modes calculés).

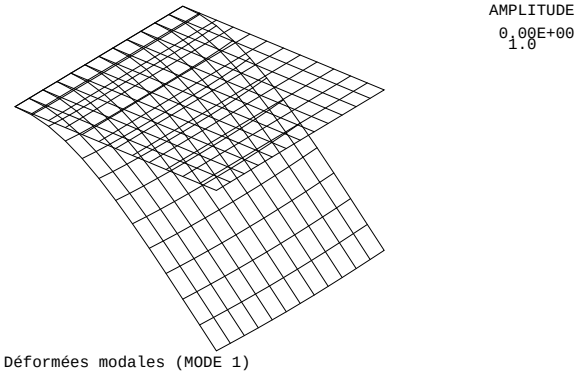
```
F1= (TAB2.1).frequence ;
F2= (TAB2.2).frequence ;
MESS 'FREQUENCE 1 (HZ) :' F1 ;
MESS 'FREQUENCE 2 (HZ) :' F2 ;
```

TAB2.1 est une table de sous-type MODE. On extrait ici la fréquence du mode 1 : (TAB2.1).frequence.

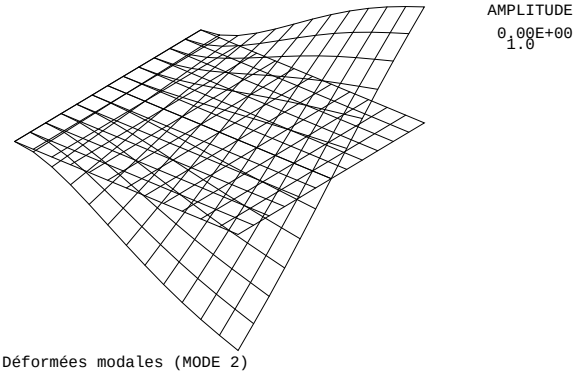
TAB2.2 est de même type que TAB2.1

```
DEP1= (TAB2.1).deformee_modale ; DEF1= DEFO DEP1 S1 1. ROUGE ;
DEP2= (TAB2.2).deformee_modale ; DEF2= DEFO DEP2 S1 1. VERT ;
DEF0= defo DEP1 S1 0. ;
```

TAB2.1.deformee_modale est un CHPOINT représentant le vecteur propre du mode 1. On l'utilise ensuite avec l'opérateur DEFO comme dans l'exemple précédent (calcul linéaire).



Déformées modales (MODE 1)



Déformées modales (MODE 2)

Chapitre 8

Traitement des résultats

Il s'avère indispensable de traiter les résultats d'une analyse effectuée avec Cast3M, ces résultats étant contenus dans des objets de type CHPOINT (analyse statique linéaire) ou de type SOLUTION (analyse modale). Ce traitement permet le calcul de grandeurs dérivées plus faciles à appréhender ou une visualisation en vue d'une meilleure interprétation des résultats. Deux groupes d'opérateurs sont disponibles à cet effet, l'un étant réservé au post-traitement des données et l'autre à la représentation graphique. Il est ensuite possible de sauvegarder les résultats en vue de les restituer dans des calculs ultérieurs.

8.1 Opérateurs de post traitement

Après un calcul statique linéaire, le résultat est un champ de déplacement contenu dans un objet de type CHPOINT. Selon les problèmes, il est nécessaire de calculer les champs de contraintes ou de déformations. On utilise à cet effet les opérateurs SIGMA et EPSI qui calculent respectivement les tenseurs de contraintes et de déformations à partir du champ de déplacements. On peut aussi calculer les contraintes équivalentes de Von Mises (opérateur VMIS), les contraintes équivalentes de Tresca (opérateur TRES) ou les contraintes principales (opérateur PRIN). On peut calculer les réactions au niveau des blocages (opérateur REAC).

Exemple :

```
SIG1 = SIGM MOD1 (MAT1 ET CAR1) RES1 ;
CALSUP = CALP SIG1 CAR1 MOD1 SUPE ; EPS1 = EPSI MOD1 RES1 CAR1 ;
VMIS1 = VMIS MOD1 SIG1 CAR1 ;
TRES1 = TRES MOD1 SIG1 CAR1 ;
PRIN1 = PRIN SIG1 MOD1 CAR1 ;
REAC1 = REAC RIG1 RES1 ;
```

L'opérateur SIGM(A) calcule un champ de contraintes SIG1 (type MCHAML) à partir du champ de déplacements RES1. Il nécessite la donnée du modèle (MOD1) et des caractéristiques matérielles et géométriques du modèle (MAT1 et CAR1). Les contraintes calculées peuvent être des efforts ou des contraintes généralisées selon les types d'éléments pris en considération. Le repère de calcul (local ou général) varie aussi selon le type des éléments.

L'opérateur CALP (CALcul en Peau) calcule un champ de contraintes (ou de déformations) au sens des milieux continus c'est-à-dire des contraintes locales (ici en peau supérieure). L'opérateur EPSI calcule un champ de déformations EPS1 (type MCHAML) à partir du champ de déplacement RES1. Il nécessite la donnée du modèle (MOD1) et des caractéristiques géométriques du modèle (CAR1). De même que pour les contraintes, selon le type des éléments, le repère varie et les déformations peuvent être des déformations généralisées ou relatives.

Il existe aussi des opérateurs permettant de manipuler les objets de type CHPOINT ou MCHAML.

EXCO : extrait une ou plusieurs composantes d'un champ.

MAXI, MINI : détermine la valeur maximale ou minimale d'un champ.

EXTR : extrait la valeur numérique d'une composante d'un champ en un point précis.

REDU : réduit le support d'un champ, plusieurs critères sont possibles.

RTEN : calcule le tenseur des contraintes dans un nouveau repère (pour certains types d'éléments).

RESU : calcule la résultante d'un objet de type CHPOINT.

XTX : calcule la norme d'un champ.

8.2 Graphiques

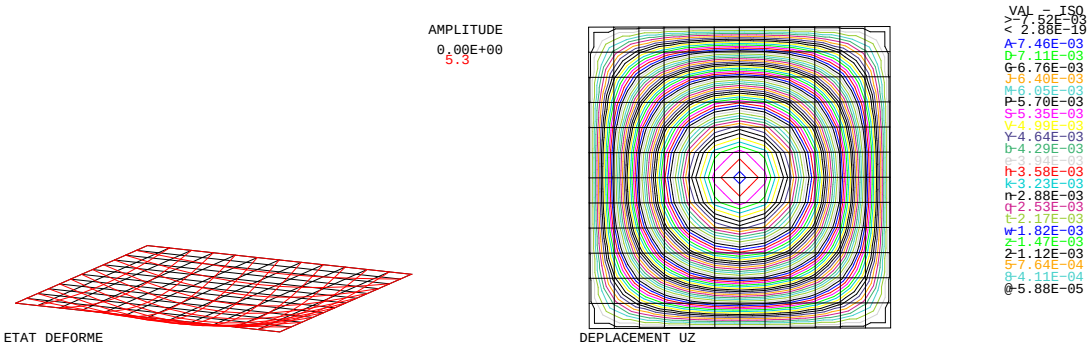
Il existe deux directives pour tracer les résultats :
- DESS(IN) : permet de dessiner des courbes contenues dans des objets de type EVOLUTION
- TRAC(ER) : permet de visualiser plusieurs types d'objets (maillages, isovaleurs, déformées, vecteurs, dessins animés)

L'exemple suivant illustre les utilisations de la directive TRAC. Il s'agit d'une plaque carrée appuyée sur ses bords (mêmes données géométriques et matérielles que le problème du paragraphe 7.3). Une pression de 1 bar est appliquée sur la plaque.

Fichiers de données

```
OPTI DIME 3 ELEM QUA4;

*DONNEES
A1 = 0.4 ; PI= 3.1415927 ;
PRES1 = -1.E5 ;

*MAILLAGE
P1 = 0.  0.  0. ; P2 = 0. A1  0. ;
VEC1 = A1 0. 0. ; LL1 = D 12 P1 P2 ;
S1 = LL1 TRAN 12 VEC1 ; L1 L2 L3 L4 = COTE S1;

*MODELE ET PROPRIETES DE MATERIAUX
MOD1= MODEL S1 MECANIQUE ELASTIQUE COQ4;
MAT1= MATER MOD1 YOUN 7.E10 NU 0.3 RHO 4200. ;
CAR1= CARAC MOD1 EPAI 0.006 ;
MATTOT = MAT1 ET CAR1 ;

*CONDITIONS AUX LIMITES
CL1 = BLOQ DEPLA L1 ; CL2 = BLOQ DEPLA L2 ;
CL3 = BLOQ DEPLA L3 ; CL4 = BLOQ DEPLA L4 ;
CLTOT = CL1 ET CL2 ET CL3 ET CL4 ;

*APPLICATION DU CHARGEMENT
CHAR1 = PRES COQUE MOD1 PRES1 DIRE (0. 0. 1.) ;

*MATRICES DE RIGIDITE ET DE MASSE
RIG1= RIGI MOD1 MATTOT ; RIG1 = RIG1 ET CLTOT ;
MAS1= MASS MOD1 MATTOT ;

*RESOLUTION
RES1 = RESO RIG1 CHAR1 ;

*POST-TRAITEMENT

*-----------CONTRAINTES-----------*
**********************************
```

```
SIG1 = SIGM MOD1 MATTOT RES1 ;
VEC1 = 1. 0. 0. ;
VEC2 = 0. 1. 0. ;
SIG2 = RTEN SIG1 MOD1 VEC1 VEC2 ;
CALSUP = CALP SIG2 CAR1 MOD1 SUPE ;

*-----TRACE DE L ETAT DEFORME-----*
************************************
DEF0= DEFO RES1 S1 0.;
DEF1= DEFO RES1 S1 ROUGE;
OEIL1 = 100. -200. 50. ;
TRAC OEIL1 (DEF0 ET DEF1) TITRE 'ETAT DEFORME' ;
ZZ1 = EXCO UZ RES1 ;
OPTI ISOV LIGNE ;
OEIL2 = 0. 0. 100. ;
TRAC OEIL2 S1 ZZ1 TITRE 'DEPLACEMENT UZ' ;

*--------TRACE D ISOVALEURS-------*
************************************
OPTI ISOV SURFACE ;
CALSUPX = EXCO SMXX CALSUP; TRAC OEIL1 MOD1 CALSUPX S1 TITRE 'SMXX' ;
CAL1 = CHAN CHPO MOD1 CALSUP ; CALX = EXCO SMXX CAL1 ;
TITRE 'CONTRAINTE DE PEAU SMXX, FACE SUPERIEURE' ;
TRAC OEIL1 CALX S1 ;
CALY = EXCO SMYY CAL1 ;
TITRE 'CONTRAINTE DE PEAU SMYY SUR DEFORME, FACE SUPERIEURE' ;
TRAC OEIL1 CALY DEF1 S1 ;

*-----TRACE DU VECTEUR REACTION---*
************************************
REA1 = REAC RES1 RIG1 ;
VECT1 = VECT REA1 2.E-5 FX FY FZ ROUGE ;
TRAC OEIL1 VECT1 S1 ;

FIN ;
```



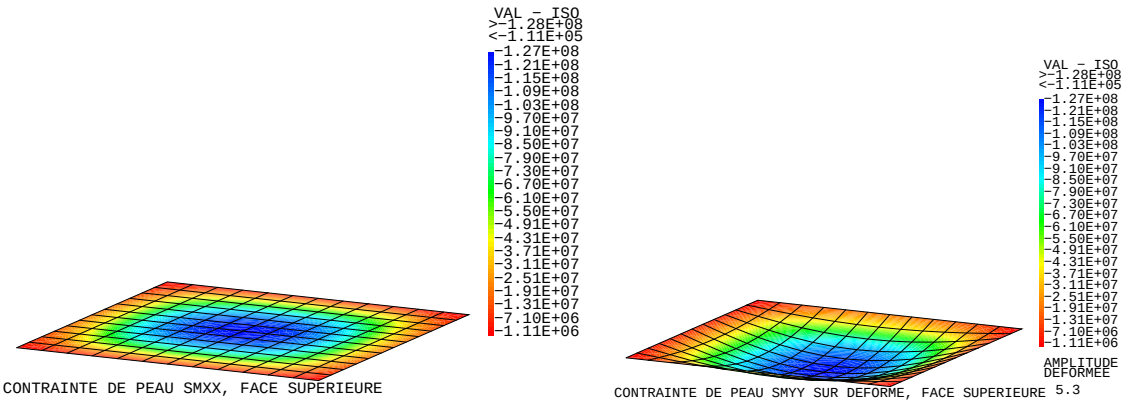
Commentaires

```
*---TRACE DE L ETAT DEFORME---*
*********************************
DEF0= DEFO RES1 S1 0.;
DEF1= DEFO RES1 S1 ROUGE;
OEIL1 = 100. -200. 50.;
TRAC OEIL1 (DEF0 ET DEF1) TITRE 'ETAT DEFORME';
ZZ1 = EXCO UZ RES1;
OPTI ISOV LIGNE;
OEIL2 = 0. 0. 100.;
TRAC OEIL2 S1 ZZ1 TITRE 'DEPLACEMENT UZ';
```

On peut donner à la directive TRAC des arguments de type DEFORME.
On peut aussi représenter les isovaleurs d'un champ par points (ici ZZ1), il faut alors spécifier l'objet de type MAILLAGE sur lequel le champ s'appuie (ici S1).
On spécifie la forme des isovaleurs dans la directive OPTI (ici LIGNE).

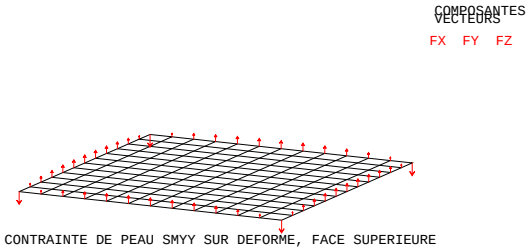
```
*-----TRACE D ISOVALEURS-----*
*********************************
OPTI ISOV SURFACE;
CALSUPX = EXCO SMXX CALSUP; TRAC OEIL1 MOD1 CALSUPX S1 TITRE 'SMXX';
CAL1 = CHAN CHPO MOD1 CALSUP; CALX = EXCO SMXX CAL1;
TITRE 'CONTRAINTE DE PEAU SMXX, FACE SUPERIEURE';
TRAC OEIL1 CALX S1;
CALY = EXCO SMYY CAL1;
TITRE 'CONTRAINTE DE PEAU SMYY SUR DEFORME, FACE SUPERIEURE';
TRAC OEIL1 CALY DEF1 S1;
```

Les isovaleurs sont définies par des surfaces.
L'objet CALSUPX est de type MCHAML. On extrait une de ses composantes pour représenter ses isovaleurs.
Il faut préciser l'objet MMODEL sur lequel s'appuie le champ par éléments.
L'objet CALX est un CHPOINT, on le trace de la même manière que ZZ1.
On peut aussi tracer un champ sur l'état déformé, il faut alors spécifier l'objet de type DEFORME (ici DEF1).



```
*---TRACE DU VECTEUR REACTION--*
*********************************
REA1 = REAC RES1 RIG1 ;
VECT1 = VECT REA1 2.E-5 FX FY FZ ROUGE ;
TRAC OEIL1 VECT1 S1 ;
```

On calcule les réactions aux appuis avec REAC. On construit ensuite l'objet de type VECTEUR à partir des composantes du champ REA1.
L'opérateur TRAC permet aussi de représenter un objet de type VECTEUR, qui se superpose à l'objet de type MAILLAGE (S1).



Chapitre 9

Erreurs classiques

Lorsque Cast3M rencontre une erreur dans la suite des instructions, il interrompt son exécution et affiche le message d'erreur relatif à l'erreur reconnue. Ce message peut se décomposer en trois parties :
- le nom de l'opérateur où l'erreur est reconnue (première ligne du message),
- les objets manquants ou mal utilisés suivis de leur type et de leur nom (lignes intermédiaires),
- les données de la commande suivie de leur type (deux dernières lignes).

Ce sont généralement les deux dernières lignes du message d'erreur qui permettent à l'utilisateur de reconnaître la nature de son erreur.
Ce chapitre n'a pas pour objectif de dresser une liste exhaustive des messages d'erreurs, il présente simplement les erreurs les plus courantes. Nous ne saurons trop conseiller à l'utilisateur de toujours lire attentivement les dernières lignes du message d'erreur.

– Omission d'un point-virgule

Exemple :

```
$ y1 = 4.
$ P1 = x1 y1 ;
**** ERREUR 11 ***** dans l'opérateur =
Il y a un résultat de type ANNULE et de nom P1
en trop par rapport aux noms à affecter
Première ligne = données : deuxième ligne = type des données.
4.       P1      =   X1       Y1
FLOTTANT ANNULE MOT FLOTTANT FLOTTANT
```

Dans cet exemple, l'utilisateur a omis le point-virgule de la première instruction. La ligne suivante est donc considérée comme la suite des données d'où l'annulation de P1 car cet objet n'est pas connu. L'utilisateur peut localiser son erreur en lisant la dernière ligne du message d'erreur qui met à bout les deux instructions.

– Omission de cote

Lorsqu'un message de ce type apparaît, il est conseillé de vérifier si chaque cote ouverte a bien été fermée.

Exemple :

```
$ mess 'Entree des points ;
$ obte x1 ; obte y1 ;
$ obte x2 ; obte y2 ;
$ P1 = x1 y1 ; P2 = x2 y2 ;
**** ERREUR 3 ***** dans l'opérateur =
Une directive ne peut pas faire plus 9 cartes
```

Dans cet exemple, l'utilisateur a omis la dernière cote de son message. Ainsi les instructions

suivantes sont considérées comme des chaînes de caractères faisant partie du message et l'erreur remarquée par le programme est la longueur trop excessive de l'instruction qu'il considère comme élémentaire.

– Dimensions

Exemple :

```
$ opti dime 3 ;
$ P2 = 3. 6. ;
**** ERREUR 37 ***** dans l'opérateur =
Troisième coordonnée ?
Première ligne = données : deuxième ligne = type des données.
3.       6.
FLOTTANT FLOTTANT
```

La dimension déclarée est 3 alors que le point P2 n'est décrit que par 2 coordonnées d'où le message d'erreur qui est très explicite.

Exemple :

```
$ opti dime 2 ;
$ P2 = 1. 1. 1. ;
**** ERREUR 11 ***** dans l'opérateur =
Il y a un résultat de type FLOTTANT et de nom
en trop par rapport aux noms à affecter
Première ligne = données : deuxième ligne = type des données.
1.       1.       1.
FLOTTANT FLOTTANT FLOTTANT
```

Ici la dimension déclarée est 2 alors que le point P2 est décrit par 3 coordonnées. Le message d'erreur est différent du précédent, il est seulement indiqué qu'il y a des données en trop et c'est à l'utilisateur de vérifier si la dimension correspond bien aux données qu'il rentre.

Exemple :

```
$ opti dime 2 elem seg2 ;
$ P1= 0. 0. ; P2 = 1. 0. ;
$ L1 = D 4 P1 P2 ;
$ MODL1 = MODE L1 MECANIQUE ELASTIQUE POUT ;
**** ERREUR 11 ***** dans l'opérateur =
Il y a un résultat de type MOT et de nom POUT
en trop par rapport aux noms à affecter
Première ligne = données : deuxième ligne = type des données.
MODE L1       MECANIQUE ELASTIQUE POUT
MOT  MAILLAGE MOT       MOT       MOT
```

Le programme ne reconnaît pas le mot POUT alors que c'est un mot-clé désignant le type d'éléments finis utilisé. L'erreur ici commise est la dimension de calcul : en effet les éléments de type POUT sont utilisés dans les calculs TRID(imensionnels), ils requièrent donc une dimension d'ordre 3. L'utilisateur doit donc toujours vérifier si les différents types d'éléments qu'il utilise sont compatibles avec le mode de calcul et la dimension de l'espace indiqués dans les options de calcul.

– Type d'éléments finis

Exemple :

```
$ opti dime 2 elem seg2 ;
$ P1 = 0. 0. ; P2 = 1. 0. ;
$ L1 = D 4 P1 P2 ;
```

```
$ SUR1 = L1 TRANS 6 (0. 1.);
**** ERREUR 16 ***** dans l'opérateur TRAN
Type d'élément incorrect
Première ligne = données : deuxième ligne = type des données.
L1       TRANS 6      #1
MAILLAGE MOT   ENTIER POINT
```

L'opérateur TRAN doit créer une surface alors que le type d'éléments finis requis est SEG2. Le message d'erreur est explicite et indique à l'utilisateur qu'il doit modifier le type de ses éléments.

– Contour mal fermé

Exemple :

```
$ P0 = 0. 0.;
$ P1 = 0. 0.; P2 = 1. 0.;
$ P3 = 1. 1.; P4 = 0. 1.;
$ L1 = D 1 P1 P2; L2 = D 1 P2 P3;
$ L3 = D 1 P3 P4; L4 = D 1 P4 P0;
$ SUR1 = DALL L1 L2 L3 L4 PLAN;
**** ERREUR 28 ***** dans l'opérateur DALL
Le contour n'est pas reconnu fermé
Première ligne = données : deuxième ligne = type des données.
DALL L1       L2       L3       L4       PLAN
MOT  MAILLAGE MAILLAGE MAILLAGE MAILLAGE MOT
```

Certains opérateurs de maillage requièrent des contours fermés. Ici les 2 objets P0 et P1 sont distincts même s'ils désignent le même point géométrique. Comme le contour commence par P1 et se termine par P0, il n'est pas considéré comme fermé.

– Composantes connexes

Parfois un message du type : "Cet objet contient 2 composantes connexes" s'affiche en cours d'exécution. Cela peut signifier qu'il existe des nœuds doubles dans le maillage ou bien qu'il existe des objets inutilisés dans la modélisation. Ce message n'interrompt pas l'exécution, mais les résultats obtenus ne seront pas fiables.

Chapitre 10

Calculs mécaniques

L'objectif de ce chapitre est de présenter différents cas de calculs mécaniques afin de mettre en évidence l'enchaînement des différentes étapes.

10.1 Chargement type poids propre

Nous allons reprendre l'exemple de la poutre encastrée-libre du paragraphe 7.2, cette fois-ci la poutre sera soumise à son poids propre. En effet, il n'existe pas d'opérateur spécifique à ce type de chargement, il est donc nécessaire de comprendre comment le représenter de manière équivalente avec Cast3M.

10.1.1 Données du problème

- $L = 1m$
- $\phi = 0,1m$
- $\rho = 7800kg/m^3$
- $E = 2,1.10^{11}Pa$

10.1.2 Fichier de données

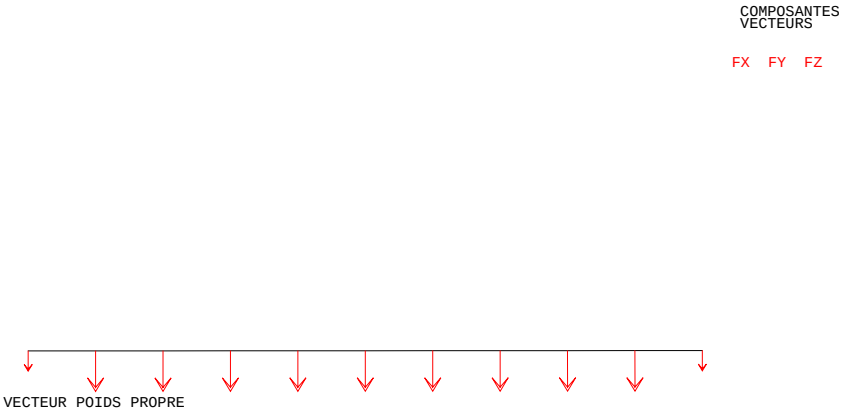
```
OPTI DIME 3 ELEM SEG2 MODE TRID ;
*------------------DONNEES--------------------*
LON1 = 1. ;
DIA1 = 0.1 ;
RHOP = 7800. ;
NUP = 0.3 ;
YOP = 2.1E11 ;
*------CALCUL DES DONNEES GEOMETRIQUES-------*
S1 = PI*(DIA1**2)/4. ;
IY = PI*(DIA1**4)/64. ;
IZ = IY ;
IG = PI*(DIA1**4)/32. ;
*-----------------GEOMETRIE------------------*
P1 = 0. 0. 0.;
P2 = LON1 0. 0.;
L1 = D 10 P1 P2 ;
*----DEFINITION DU MODELE ET DU MATERIAU-----*
MOD1 = MODEL L1 MECANIQUE ELASTIQUE ISOTROPE POUT ;
MAT1 = MATER MOD1 YOUN YOP NU NUP RHO RHOP ;
CAR1 = CARAC MOD1 SECT S1 INRY IY INRZ IZ TORS IG ;
MATTOT = MAT1 ET CAR1 ;
*-------------CONDITIONS LIMITES-------------*
```

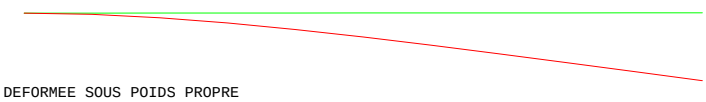
```
CL1 = BLOQ DEPLA ROTA P1 ;
*-----------------CHARGEMENT-----------------*
MAS1 = MASSE MOD1 MATTOT ;
CHP1 = MANU CHPO L1 1 UY -9.81 ;
CHA1 = MAS1*CHP1 ;
*--------------TRACE DU VECTEUR--------------*
VEC1 = VECT CHA1 1.E-3 FX FY FZ ROUG ;
OEIL1 = 0. 0. 1000. ;
TRAC OEIL1 VEC1 L1 TITRE 'VECTEUR POIDS PROPRE' ;
*-------------MATRICE DE RIGIDITE------------*
RIG1 = RIGI MOD1 MATTOT ;
RIG1 = RIG1 ET CL1 ;
*------------------RESOLUTION----------------*
RES1 = RESO RIG1 CHA1 ;
*------------------RESULTATS-----------------*
DY = EXCO RES1 UY ;
DYABS = ABS DY ;
DYMAX = MAXI DYABS ;
MESS 'FLECHE MAXIMALE EN METRES :'DYMAX ;
*--------------------DEFORMEE----------------*
DEF0 = DEFO L1 RES1 0. VERT ;
DEF1 = DEFO L1 RES1  ROUGE ;
TITRE 'DEFORMEE SOUS POIDS PROPRE' ;
TRAC OEIL1 (DEF0 ET DEF1) ;
FIN;
```

10.1.3 Commentaires

```
MAS1 = MASSE MOD1 MATTOT ;
CHP1 = MANU CHPO L1 1 UY -9.81 ;
CHA1 = MAS1*CHP1 ;
```

La force de poids propre (CHA1) est calculée en multipliant la masse par l'accélération ($F = M.\gamma$). On construit donc la matrice de masse MAS1 à l'aide l'opérateur MASSE en précisant le modèle (MOD1) et les caractéristiques (MATTOT). On crée ensuite manuellement le champ d'accélération CHP1 à l'aide de l'opérateur MANU CHPO en précisant le maillage support (L1), le nombre de composantes (1), le nom des composantes (UY) et la valeur (-9.81). On multiplie alors la matrice de masse par le champ d'accélération ce qui permet de pondérer le chargement au sens des éléments finis.


COMPOSANTES
VECTEURS
FX FY FZ
VECTEUR POIDS PROPRE
AMPLITUDE
0.00E+00
1.37E+03


DEFORMEE SOUS POIDS PROPRE

10.2 Calcul axisymétrique

Nous allons calculer une coque sphérique chargée par une pression interne uniforme. Pour simplifier le problème qui est symétrique autour des trois axes, on effectue le calcul en 2D en ne maillant que le quart d'un cercle, en mode axisymétrique.

10.2.1 Données du problème

- $P = 1bar$
- $R = 2m$
- $Ep = 0.01m$
- $\nu = 0.3$
- $E = 2.10^{11}Pa$

10.2.2 Fichier de données

```
OPTI DIME 2 ELEM SEG2 MODE AXIS ;
*--------------------DONNEES--------------------*
PRES1 = 1.E5 ;
RAY1 = 2. ;
EPA1 = 0.01 ;
NUP = 0.3 ;
YOP = 2.1E11 ;
*-------------------GEOMETRIE--------------------*
P0 = 0. 0. ;
P1 = 0. RAY1 ;
P2 = RAY1 0. ;
L1 = C 24 P1 P0 P2 ; TRAC L1 ;
*------DEFINITION DU MODELE ET DU MATERIAU--------*
MOD1 = MODEL L1 MECANIQUE ELASTIQUE ISOTROPE COQ2 ;
MAT1 = MATER MOD1 YOUN YOP NU NUP ;
CAR1 = CARAC MOD1 EPAI EPA1 ;
MATTOT = MAT1 ET CAR1 ;
*--------------CONDITIONS LIMITES-----------------*
CL1 = SYMT DEPLA P1 P0 L1 1.E-4 ;
CLR1 = SYMT ROTA P1 P0 L1 1.E-4 ;
CL2 = SYMT DEPLA P2 P0 L1 1.E-4 ;
CLR2 = SYMT ROTA P2 P0 L1 1.E-4 ;
CLTOT = CL1 ET CLR1 ET CL2 ET CLR2 ;
*-----------------CHARGEMENT--------------------*
CHA1 = PRESS COQUE MOD1 PRES1 NORM ;
VEC1 = VECT CHA1 5.E-6 FR FZ ROUGE ;
TRAC VEC1 L1 ;
*-------------MATRICE DE RIGIDITE----------------*
RIG1 = RIGI MOD1 MATTOT ;
RIG1 = RIG1 ET CLTOT ;
*-----------------RESOLUTION--------------------*
RES1 = RESO RIG1 CHA1 ;
*-----------------RESULTATS---------------------*
DUR = EXCO RES1 UR ;
DURMAX = MAXI (ABS DUR) ;
MESS 'DILATATION EN METRES :'DURMAX ;
*-----------------DEFORMEE----------------------*
DEF0 = DEFO L1 RES1 0. VERT ;
DEF1 = DEFO L1 RES1  ROUGE ;
```

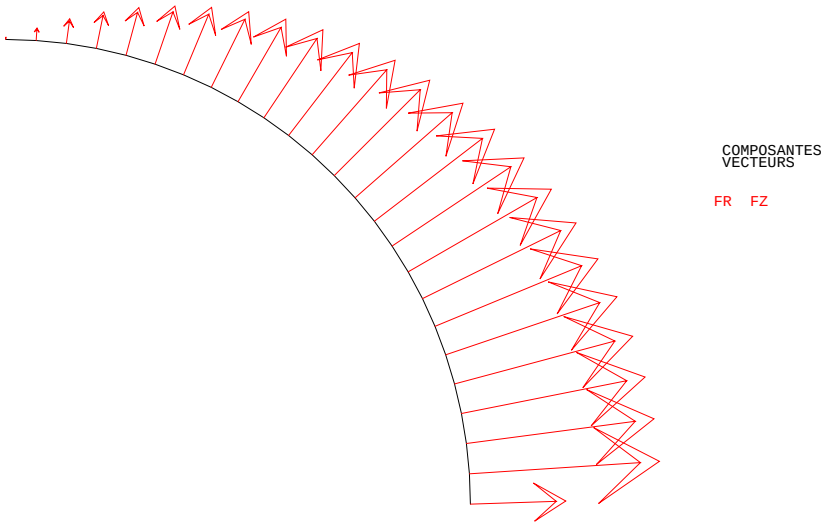
```
TITRE 'DEFORMEE SOUS PRESSION INTERNE de 'PRES1'Pa' ;
TRAC (DEF0 ET DEF1) ;
FIN ;
```

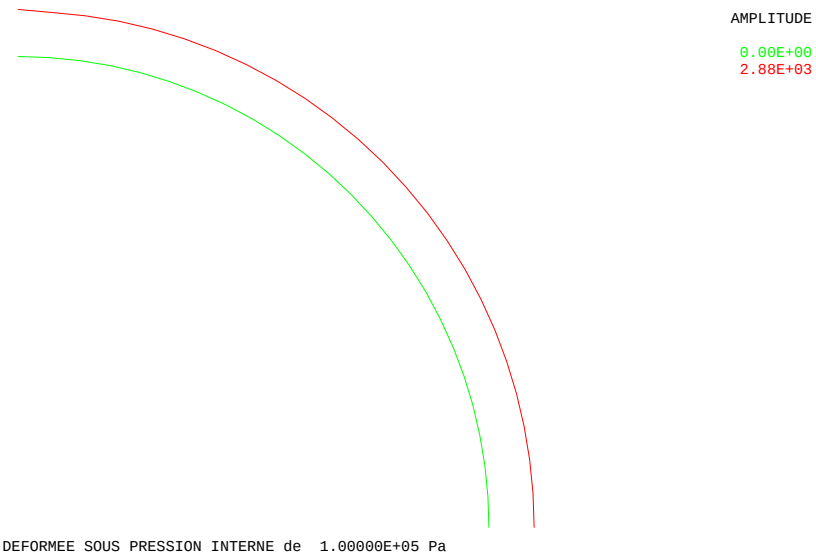
10.2.3 Commentaires

`OPTI DIME 2 ELEM SEG2 MODE AXIS ;`
L'option MODE AXIS précise que le calcul s'effectue en axisymétrie. Les abscisses correspondent à l'axe radial U_r et les ordonnées à l'axe d'axisymétrie U_z. Les degrés de libertés sont désignés par UR, UZ et RT (respectivement déplacements radiaux, axiaux et rotations).

```
CL1 = SYMT DEPLA P1 P0 L1 1.E-4 ;
CLR1 = SYMT ROTA P1 P0 L1 1.E-4 ;
CL2 = SYMT DEPLA P2 P0 L1 1.E-4 ;
CLR2 = SYMT ROTA P2 P0 L1 1.E-4 ;
CLTOT = CL1 ET CLR1 ET CL2 ET CLR2 ;
```
Pour traiter ce problème, on n'étudie qu'un quart de cercle : on doit par conséquent écrire des conditions limites relative à la symétrie dans le plan. La symétrie axiale est automatiquement prise en compte par le mode de calcul axisymétrique. L'opérateur SYMT permet d'imposer des conditions de symétrie sur les degrés de liberté concernés de la géométrie L1 par rapport au plan défini par les points (P1, P0) puis (P2, P0). On aurait également pu écrire les conditions limites suivantes :
CL1 = BLOQ UR RT P1 ;
CL2 = BLOQ UZ RT P2 ;
CLTOT= CL1 ET CL2 ;

`CHA1 = PRESS COQUE MOD1 PRES1 NORM ;`
L'opérateur PRESS permet de calculer le chargement nodal dû à une force de pression. Il est ici suivi du mot-clé COQUE pour indiquer que la pression s'appuie sur des éléments de coques. On doit aussi préciser le modèle (MOD1) sur lequel est appliquée la pression, la valeur de cette pression (PRES1) et la direction de la pression (NORM) qui est ici normale à la surface et orientée selon la normale positive à l'élément.


COMPOSANTES
VECTEURS
FR FZ
GIBI FECIT

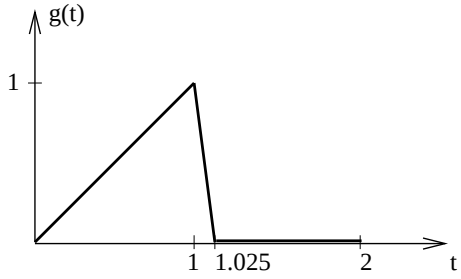

AMPLITUDE
0.00E+00
2.88E+03
DEFORMEE SOUS PRESSION INTERNE de 1.00000E+05 Pa

10.3 Calcul dynamique pas à pas

Nous allons calculer la réponse d'une poutre soumise à un chargement variable au cours du temps. Pour résoudre l'équation dynamique, on utilise une procédure basée sur l'algorithme de Newmark centré.

10.3.1 Données du problème

- $F = 100N$
- $F(t) = F * g(t)$
- $L = 50m$
- $A = 1m^2$
- $I_y = 10m^4$
- $I_z = 10m^4$
- $\nu = 0.3$
- $E = 2.10^{11}Pa$
- $\rho = 1000.kg/m^3$



10.3.2 Fichier de données

```
OPTI DIME 3 ELEM SEG2 MODE TRID;

*---------------------DONNEES---------------------*
AMP1 = -100000. ;
LON1 = 50. ;
SEC1 = 1. ;
IY1 = 10. ;
IZ1 = 10. ;
NUP = 0.3 ;
YOP = 2.1E11 ;
RHOP = 1000. ;
RAID1 = 5.E8 ;
*--------------------GEOMETRIE--------------------*
P1 = 0.   0. 0. ;
P2 = LON1 0. 0. ;
L1 = D 4 P1 P2 ; TRAC L1 ;
*------DEFINITION DU MODELE ET DU MATERIAU--------*
MOD1 = MODEL L1 MECANIQUE ELASTIQUE POUT ;
MAT1 = MATER MOD1 YOUN YOP NU NUP RHO RHOP ;
CAR1 = CARAC MOD1 SECT SEC1 INRY IY1 INRZ IZ1
       TORS 1. VECT (0. 1. 0.) ;
MATTOT = MAT1 ET CAR1 ;
*--------------CONDITIONS LIMITES-----------------*
CL1 = APPUI UY RAID1 P1 ;
CL2 = BLOQ L1 UX UZ RX RY ;
CL3 = APPUI UY RAID1 P2 ;
CLTOT= CL1 ET CL2 ET CL3 ;
*-------MATRICES DE RIGIDITE ET DE MASSE----------*
RIG1 = RIGI MOD1 MATTOT ;
RIG1 = RIG1 ET CLTOT ;
MAS1 = MASS MOD1 MATTOT ;
```

```
*------------EVOLUTION TEMPORELLE-----------------*
LIS1 = PROG 0. 1. 1.025 3.025 ;
LIS2 = PROG 0. 1. 0. 0. ;
EVT1 = EVOL MANU T LIS1 G(T) LIS2 ;
*-----------------CHARGEMENT---------------------*
FOR1 = FORCE FY AMP1 L1 ;
CHA1 = CHAR FORC FOR1 EVT1 ;
VEC1 = VECT FOR1 1.e-4 FX FY FZ ROUGE ;
TRAC VEC1 L1 ;
*---------------TABLE DE DONNEES-----------------*
TAB2 = TABLE;
TAB2.CHAR = CHA1 ;
TAB2.RIGI = RIG1 ;
TAB2.MASS = MAS1 ;
TAB2.FREQ = 20. ;
TAB2.INST = PROG 0. PAS 1.25E-2 3.;
TAB2.DEPL = MANU CHPO L1 3 UX 0. UY 0. UZ 0. ;
TAB2.VITE = MANU CHPO L1 3 UX 0. UY 0. UZ 0. ;
*-------------RESOLUTION PAS A PAS---------------*
TAB1 = DYNAMIC TAB2 ;
*-------------------RESULTATS--------------------*
LT   = PROG ; LUY1 = PROG ;
LUY2 = PROG ; LUY3 = PROG ;
P3 = L1 POIN 3 ;
I = 0 ;
NBB = (DIME TAB2.INST) - 1 ;
REPETER BOUC1 NBB ;
  I = I + 1 ;
  LT = LT ET (PROG (TAB1.I.TEMP)) ;
  DEP1 = TAB1.I.DEPL ;
  LUY1 = LUY1 ET (PROG (EXTR DEP1 UY P1));
  LUY2 = LUY2 ET (PROG (EXTR DEP1 UY P2));
  LUY3 = LUY3 ET (PROG (EXTR DEP1 UY P3));
 FIN BOUC1 ;
EVY1 = EVOL MANU 'TEMPS' LT 'UY(P1)' LUY1 ;
EVY2 = EVOL MANU 'TEMPS' LT 'UY(P2)' LUY2 ;
EVY3 = EVOL MANU 'TEMPS' LT 'UY(P3)' LUY3 ;
DESS EVY1 TITR 'UY POINT P1' MIMA ;
DESS EVY2 TITR 'UY POINT P2' MIMA ;
DESS EVY3 TITR 'UY POINT P3' MIMA ;
*------------------DEFORMEE--------------------*
DEP1 = TAB1. 80 .DEPL ;
DEF0 = DEFO L1 DEP1 0. VERT ;
DEF1 = DEFO L1 DEP1  ROUGE ;
TITRE 'DEFORMEE TEMPS T=1 I=80 ' ;
TRAC (DEF0 ET DEF1) ;
FIN;
```

10.3.3 Commentaires

```
LIS1 = PROG 0. 1. 1.025 3.025 ;
LIS2 = PROG 0. 1. 0. 0. ;
EVT1 = EVOL MANU T LIS1 G(T) LIS2 ;
FOR1 = FORCE FY AMP1 L1 ;
CHA1 = CHAR FORC FOR1 EVT1 ;
```

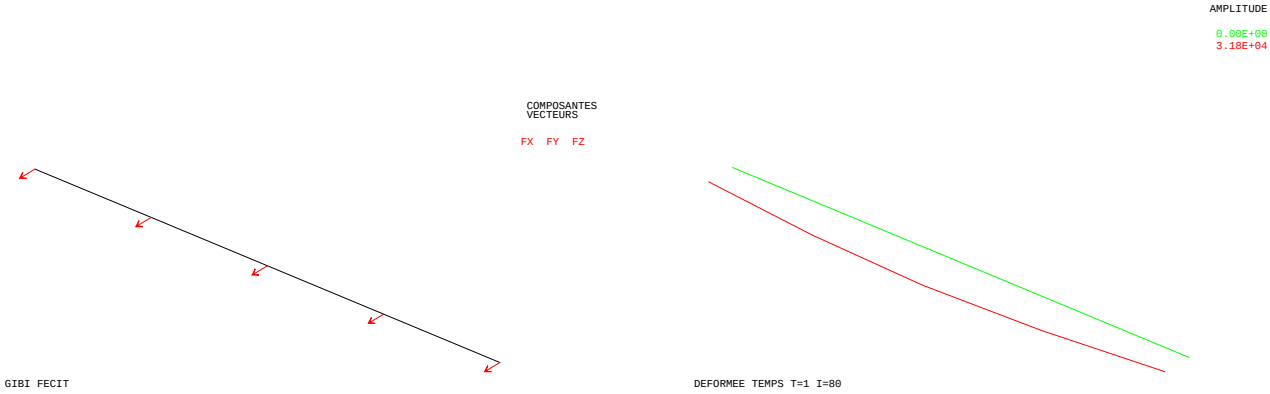
L'opérateur CHAR(GEMENT) crée un objet de type CHARGEMENT qui contient la description spatiale f(x,y,z) (contenue dans le champ par points FOR1) et la description temporelle g(t) (contenue dans l'évolution EVT1) du chargement F(x,y,z,t) tel que F = f(x,y,z).g(t).

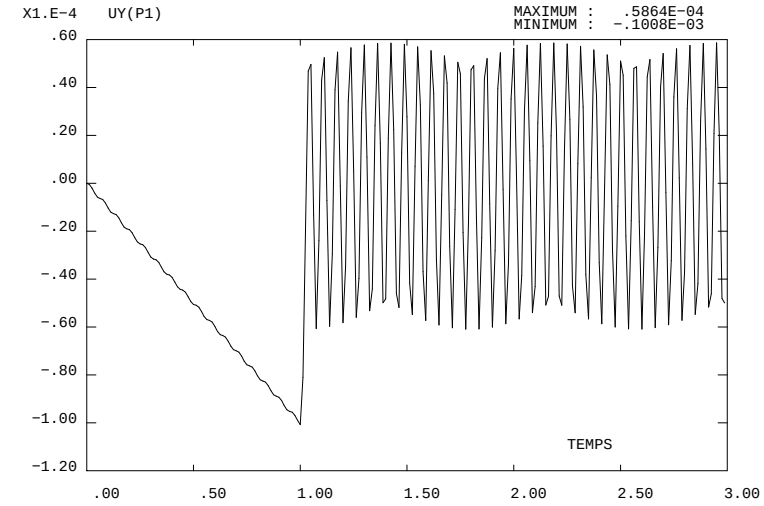
```
TAB2 = TABLE ;
TAB2.CHAR = CHA1 ;
TAB2.RIGI = RIG1 ;
TAB2.MASS = MAS1 ;
TAB2.FREQ = 20. ;
TAB2.INST = PROG 0. PAS 1.25E-2 3. ;
TAB2.DEPL = MANU CHPO L1 3 UX 0. UY 0. UZ 0. ;
TAB2.VITE = MANU CHPO L1 3 UX 0. UY 0. UZ 0. ;
TAB1 = DYNAMIC TAB2 ;
```

Les arguments d'entrée et de sortie de la procédure DYNAMIC sont regroupés dans des tables. Les différents indices de la table d'entrée (TAB2) sont :
- CHAR : contient le chargement (objet de type CHARGEMENT)
- RIGI : contient la matrice de rigidité (objet de type RIGIDITE)
- MASS : contient la matrice de masse (objet de type RIGIDITE)
- FREQ : contient la fréquence de coupure (objet de type FLOTTANT). Cette valeur permet de déterminer le pas de temps du calcul : $\Delta t = 0.25/F_{coup}$.
- INST : contient la liste des instants de sortie (objet de type LISTREEL). Le dernier instant de sortie doit être inférieur au dernier temps de la description temporelle du chargement.
- DEPL : contient les déplacements initiaux (objet de type CHPOINT)
- VITE : contient les vitesses initiales (objet de type CHPOINT). Les noms de composantes des vitesses doivent être identiques aux noms des composantes des déplacements.

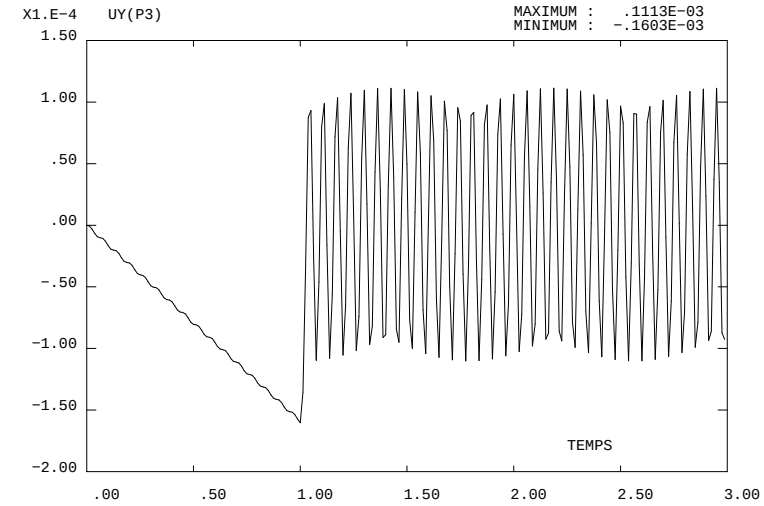
La table de sortie (TAB1) est construite de la manière suivante :
TAB1.i (objet de type TABLE) avec en indice :
- TEMP : la valeur du ième instant (type FLOTTANT)
- DEPL : les déplacements du ième instant (type CHPOINT)
- VITE : les vitesses du ième instant (type CHPOINT)

Ainsi il y a autant de tables TAB1.i qu'il y a d'instants de sortie.





UY POINT P1



UY POINT P3

Chapitre 11

Calculs thermiques

Il est possible de faire l'analogie suivante entre le calcul thermique et le calcul mécanique :

CALCULS MECANIQUES : $\overline{\overline{K}}.\{U\} = \{F\}$	CALCULS THERMIQUES : $\overline{\overline{\Lambda}}.\{T\} = \{\Phi\}$
Matrice de rigidité $\overline{\overline{K}}$	Matrice de conductivité $\overline{\overline{\Lambda}}$
Matrice de masse $\overline{\overline{M}}$	Matrice de capacité $\overline{\overline{C}}$
Vecteur des déplacements généralisés $\{U\}$	Vecteur des températures $\{T\}$
Vecteur des forces généralisées $\{F\}$	Vecteur des flux nodaux $\{\Phi\}$

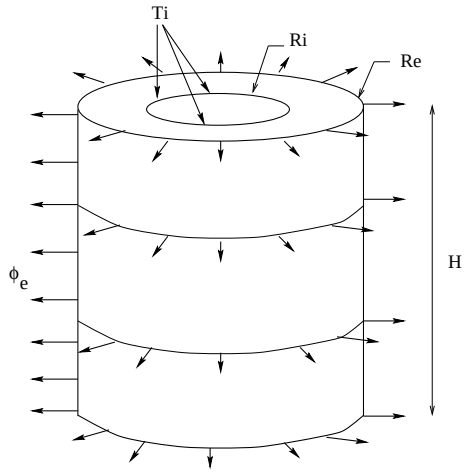
À partir d'exemples simples, nous allons présenter les différents opérateurs permettant d'effectuer des calculs thermiques.

11.1 Calcul du champ de températures d'un tube épais

Le problème suivant consiste à calculer le champ de températures dans un tube épais soumis à une température imposée sur sa paroi intérieure et à un flux sur sa paroi extérieure. Le problème étant à symétrie de révolution, le calcul sera effectué en mode axisymétrique.

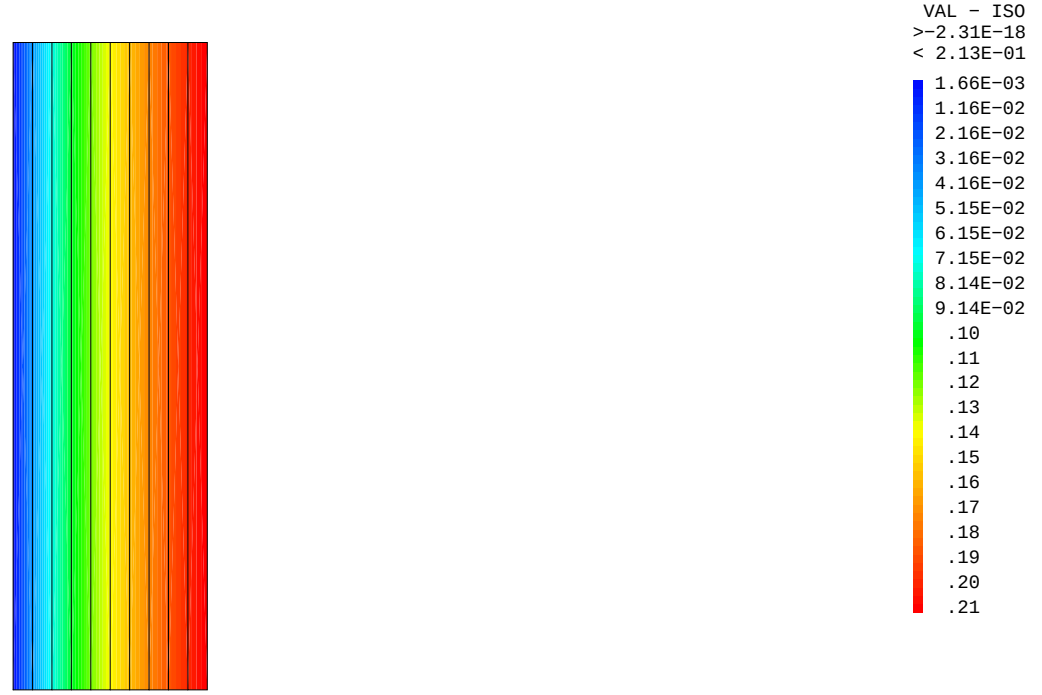
11.1.1 Données du problème

- $R_i = 0.05m$
- $R_e = 0.2m$
- $H = 0.5m$
- $T_i = 0^oC$
- $\Phi_e = 100.W/m^2$
- $K = 130W/m^o$



11.1.2 Fichier de données

```
TITRE 'CALCUL THERMIQUE D UN TUBE ' ;
OPTION DIME 2 ELEM QUA4 MODE AXIS ;
OPTION ECHO 0 ; SAUT LIGNE ;
*
*---  CREATION DE LA GEOMETRIE:
*
P1 = 0.05 0. ; P2 = 0.2  0.  ;
VEC1 = 0. 0.5 ;
L1 = D 10 P1 P2 ;
SURF1  = L1 TRANS 1 VEC1 ;
TRAC SURF1 ;
*
D1 = COTE 4 SURF1 ;
D3 = COTE 2 SURF1 ;
*
*---  DONNEES DU PROBLEME DE THERMIQUE:
*
*---  MODELISATION:
MOD1 = MODE SURF1 THERMIQUE ISOTROPE ;
*
*---  CARACTERISTIQUES DU MATERIAU:
MAT1 = MATE MOD1 K 130. ;
*
*---  MATRICE DE CONDUCTIVITE:
COND1 = CONDUCTIVITE MOD1 MAT1 ;
*
*---  CONDITIONS AUX LIMITES : TEMPERATURES IMPOSEES
CL1 = BLOQUE T D1 ;
DCL1 = DEPI CL1 0. ;
*
*---  CHARGEMENT : FLUX IMPOSE
FLU1 = FLUX MOD1 100. D3 ;
*
*---  ASSEMBLAGE DES PREMIER ET SECOND MEMBRES:
CONTOT = COND1 ET CL1 ;
FLUTOT = DCL1 ET FLU1 ;
*
*---  RESOLUTION:
CHTER1 = RESOUDRE CONTOT FLUTOT ;
*
*--- POST TRAITEMENT
*
TITRE 'CHAMP DE TEMPERATURES' ;
TRAC CHTER1 SURF1 ;
*
TETA2 = EXTR CHTER1 T P2 ;
TTHP2 = 0.213 ;
MESS 'TEMPERATURE THEORIQUE EN P2 :'TTHP2 ;
MESS 'TEMPERATURE CALCULEE EN P2 :'TETA2 ;
*
*--- FIN DU FICHIER
FIN ;
```



11.1.3 Commentaires

```
MOD1 = MODE SURF1 THERMIQUE ISOTROPE ;
```
On définit un objet de type MMODEL qui s'appuie sur le maillage SURF1 et suit un comportement thermique isotrope.

```
MAT1 = MATE MOD1 K 130. ;
```
On définit le champ de caractéristiques matérielles du modèle MOD1 en précisant la conductivité K.

```
COND1 = CONDUCTIVITE MOD1 MAT1 ;
```
L'opérateur COND(UCTIVITE) construit la matrice de conductivité de l'objet MOD1. L'objet créé COND1 est de type RIGIDITE.

```
CL1 = BLOQUE T D1 ;
DCL1 = DEPI CL1 0. ;
```
Pour fixer les conditions limites, on utilise les mêmes opérateurs que pour les calculs mécaniques : l'opérateur BLOQUE permet de fixer la température T sur une partie du maillage (droite D1) en créant un objet de type RIGIDITE ; l'opérateur DEPI spécifie la valeur du blocage CL1 en créant un objet de type CHPOINT.
L'objet CL1 doit être ajouté à la matrice de conductivité et l'objet DCL1 doit être ajouté au chargement.

```
FLU1 = FLUX MOD1 100. D3 ;
```
L'opérateur FLUX permet d'imposer un flux de valeur 100 sur une partie du contour de la structure définie dans l'objet MOD1 (droite D3). L'objet créé FLU1 est de type CHPOINT.

```
CONTOT = COND1 ET CL1 ;
FLUTOT = DCL1 ET FLU1 ;
```

```
CHTER1 = RESOUDRE CONTOT FLUTOT ;
```

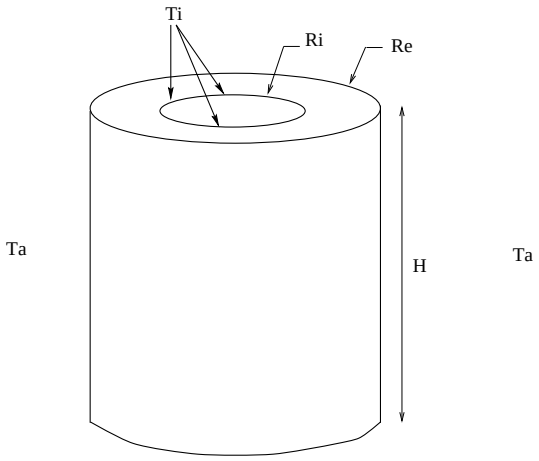
La matrice de conductivité totale est obtenue en ajoutant les blocages CL1 à la matrice de conductivité COND1. Le second membre du système à résoudre est obtenu en rassemblant le flux FLU1 et la température imposée DCL1.
L'opérateur RESO(UDRE) permet de résoudre le système $\overline{\overline{\Lambda}}.\{T\} = \{\Phi\}$. L'objet créé CHTER1 est de type CHPOINT et contient la valeur des températures en chaque point du modèle.
Le post traitement s'effectue de la même manière que pour les calculs mécaniques.

11.2 Tube soumis à une convection forcée

L'objectif de cet exemple est de calculer l'état stabilisé d'un transfert de chaleur avec convection forcée. Il s'agit d'un tube dont la paroi intérieure est soumise à une température imposée et dont la paroi extérieure est soumise à une convection forcée.

11.2.1 Données du problème

- $R_i = 6m$
- $R_e = 12m$
- $H = 4m$
- $T_i = 100^oC$
- $T_a = 125^oC$
- $H = 5W/m^{2^o}$
- $K = 8W/m^o$



11.2.2 Fichier de données

```
TITRE 'TUBE SOUMIS A UNE CONVECTION FORCEE' ;
OPTION DIME 2 ELEM QUA4 MODE AXIS ;
OPTION ECHO 0 ; SAUT LIGNE ;
*
*---  CREATION DE LA GEOMETRIE:
*
P1 = 6. 0. ; P2 = 16.  0.  ;
VEC1 = 0. 4. ;
*
L1 = D 8 P1 P2 ;
SURF1  = L1 TRANS 5 VEC1 ;
TRAC SURF1 ;
*
D1 = COTE 4 SURF1 ;
D3 = COTE 2 SURF1 ;
P3 = SURF1 POIN PROC (16. 4.) ;
*
*---  DONNEES DU PROBLEME DE THERMIQUE:
*
*---  MODELISATION:
*
MOD1 = MODEL SURF1 THERMIQUE ISOTROPE ;
MOD2 = MODEL D3 CONVECTION ;
*
*---  CARACTERISTIQUES DU MATERIAU:
*
MAT1 = MATER MOD1 K 8. ;
MAT2 = MATER MOD2 H 5. ;
*
```

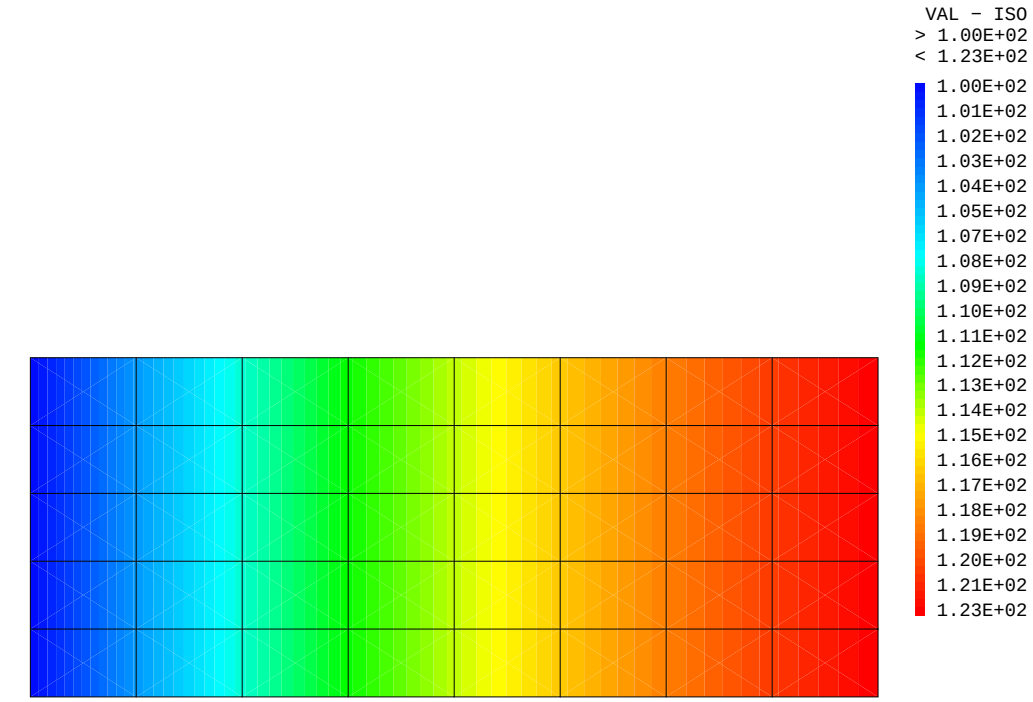
```
*---  MATRICE DE CONDUCTIVITE:
*
COND1 = CONDUCTIVITE MOD1 MAT1 ;
COND2 = CONDUCTIVITE MOD2 MAT2 ;
*
*---  CONDITIONS AUX LIMITES : TEMPERATURES IMPOSEES
*
CL1 = BLOQUE T D1 ;
DCL1 = DEPI CL1 100. ;
*
*---  FLUX EQUIVALENTS A LA CONVECTION:
*
FF1 = CONVECTION MOD2 MAT2 T 125. ;
*
*---  ASSEMBLAGE DES PREMIER ET SECOND MEMBRES:
*
CONTOT = COND1 ET COND2 ET CL1 ;
FLUTOT = DCL1 ET FF1 ;
*
*---  RESOLUTION:
*
CHTER1 = RESOUDRE CONTOT FLUTOT ;
*
*--- POST TRAITEMENT
*
TITRE 'CHAMP DE TEMPERATURES' ;
TRAC CHTER1 SURF1 ;
*
TETA3 = EXTR CHTER1 T P3 ;
TTHP3 = 122.68 ;
MESS 'TEMPERATURE THEORIQUE EN P2 :'TTHP3 ;
MESS 'TEMPERATURE CALCULEE EN P2 :'TETA3 ;
*
*--- FIN DU FICHIER
FIN ;
```

11.2.3 Commentaires

`MOD1 = MODEL SURF1 THERMIQUE ISOTROPE ;`
`MOD2 = MODEL D3 CONVECTION ;`
On définit deux objets de type MMODEL : le premier s'appuie sur le maillage SURF1 et suit un comportement thermique isotrope ; le second s'appuie sur la ligne D3 et permet de préciser le comportement de convection de ce côté.

`MAT2 = MATER MOD2 H 5. ;`
On définit le champ de caractéristiques matérielles du modèle de convection MOD2 en précisant le coefficient d'échange H.

`FF1 = CONVECTION MOD2 MAT2 T 125. ;`
Pour définir le flux de convection, on utilise l'opérateur CONV(ECTION) suivi du modèle et des caractéristiques de convection, on indique également la valeur de la température ambiante.
Les opérations suivantes sont analogues à celles du problème précédent.



CHAMP DE TEMPERATURES

11.3 Calcul thermo-mécanique

Il peut être intéressant de coupler les effets thermiques aux effets mécaniques. En effet, dans certains problèmes les champs de températures induisent des contraintes thermiques non négligeables, ces contraintes doivent donc être prises en compte en s'ajoutant aux contraintes mécaniques.

La démarche de ce type de calcul est la suivante :

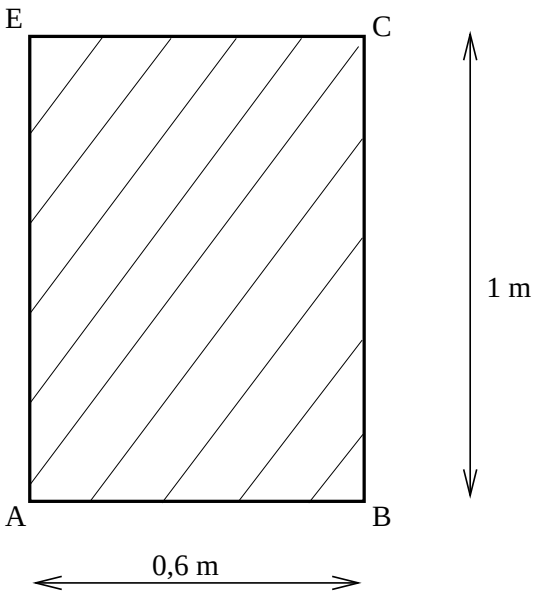
– Calcul thermique afin d'obtenir le champ de températures de la structure.
– Calcul mécanique en appliquant le chargement mécanique et le chargement thermique, ce dernier étant obtenu à partir du champ de températures.

L'exemple suivant va illustrer l'enchaînement d'un calcul thermo-mécanique : il s'agit d'une structure soumise selon ses côtés à une température imposée et à une convection forcée en ce qui concerne le chargement thermique ; une pression et un encastrement constituent le chargement mécanique.

11.3.1 Données du problème

Chargement thermique :
- sur AB : température imposée $T_i = 100^oC$
- sur BC et CE : convection forcée avec température ambiante $T_a = 0^oC$
- sur EA : droite isolée

Chargement mécanique :
- sur AB et CE : encastrement
- sur EA : pression $P = 100MPa$



Caractéristiques matérielles :
- $H = 750 W/m^{2^o}$
- $K = 52 W/m^o$
- $E = 2.10^{11} Pa$
- $\nu = 0.3$
- $\rho = 7800 kg/m^3$
- $\alpha = 10^{-5}$

11.3.2 Fichier de données

```
TITRE 'CALCUL THERMO-MECANIQUE' ;
OPTION DIME 2 ELEM QUA8 MODE PLAN CONT ;
OPTION ECHO 0 ; SAUT LIGNE ;
*
*---  CREATION DE LA GEOMETRIE:
A = 0.  0. ; B = 0.6 0. ;
C = 0.6 1. ; E = 0.  1. ;
AB = D 6 A B ;
BC = D 10 B C ;
CE = D 6 C E ;
EA = D 10 E A ;
SURF1  = DALL AB BC CE EA PLAN ;
TRAC SURF1 ;
*
*------------------  CALCUL THERMIQUE  ---------------------*
*
*---  MODELISATION:
MOD1 = MODE SURF1 THERMIQUE ISOTROPE ;
MOD1 = MODE SURF1 THERMIQUE ISOTROPE QUA8 ;
MOD2 = MODE (BC ET CE) CONVECTION ;
*
*---  CARACTERISTIQUES DU MATERIAU:
MAT1 = MATE MOD1 K 52. ;
MAT2 = MATE MOD2 H 750. ;
*
```

```
*---  CONDITIONS AUX LIMITES : TEMPERATURES IMPOSEES
CL1 = BLOQUE T AB ;
FCL1 = DEPI CL1 100. ;
*
*---  CHARGEMENT : FLUX DE CONVECTION
FLU1 = CONV MOD2 MAT2 T 0. ;
*
*---  MATRICES DE CONDUCTIVITE:

COND1 = COND MOD1 MAT1 ;
COND2 = COND MOD2 MAT2 ;
CONDTOT = COND1 ET COND2 ET CL1 ;
*
*---  ASSEMBLAGE DU SECOND MEMBRE:
FLUTOT = FCL1 ET FLU1 ;
*
*---  RESOLUTION:
CHTER1 = RESOU CONDTOT FLUTOT ;
*
*--- POST TRAITEMENT
TITRE 'CHAMP DE TEMPERATURES (MAXI='(MAXI CHTER1)')' ;
TRAC CHTER1 SURF1 ;
*
*------------------  CALCUL MECANIQUE  ---------------------*
*
*---  MODELISATION:
MOD1 = MODE SURF1 THERMIQUE ISOTROPE ;
MOD3 = MODE SURF1 MECANIQUE ELASTIQUE ISOTROPE QUA8 ;
*
*---  CARACTERISTIQUES DU MATERIAU:
MAT3 = MATE MOD3 YOUN 2.E11 NU 0.3 RHO 7800. ALPH 1.e-5 ;
*
*---  CONDITIONS AUX LIMITES :
CL3 = BLOQ DEPL (AB ET CE) ;
*
*---  CHARGEMENT : THERMIQUE ET PRESSION
SIGT3 = THET MOD3 CHTER1 MAT3 ;
F3 = BSIG MOD3 SIGT3 ;
PRES3 = PRESSION MASS MOD3 100.E6 EA ;
CHAR3 = F3 ET PRES3 ;
*
*---  RESOLUTION:
RIG3 = RIGI MOD3 MAT3 ;
RIGCL3 = RIG3 ET CL3 ;
DEP3 = RESOU RIGCL3 CHAR3 ;
*
*--- POST TRAITEMENT
*DEFORMEE
DEF0 = DEFO SURF1 DEP3 0. BLAN ;
DEF1 = DEFO SURF1 DEP3    ROUG ;
TITR 'DEFORMEE APRES CHARGEMENT THERMIQUE ET PRESSION' ;
TRAC (DEF0 ET DEF1) ;
*
*CALCUL DES CONTRAINTES TOTALES ET MECANIQUES
SIGTOT3 = SIGMA MOD3 MAT3 DEP3 ;
SIGM3 = SIGTOT3 - SIGT3 ;
*
```

```
S_XX = (EXCO SMXX SIGM3)/1.e6 ;
TITR ' CONTRAINTES S_XX (MAXI=' (MAXI S_XX) 'MPa)' ;
TRAC MOD3 S_XX ;
S_YY = (EXCO SMYY SIGM3)/1.e6 ;
TITR ' CONTRAINTES S_YY (MAXI=' (MAXI S_YY) 'MPa)' ;
TRAC MOD3 S_YY ;
*
*--- FIN DU FICHIER
FIN ;
```

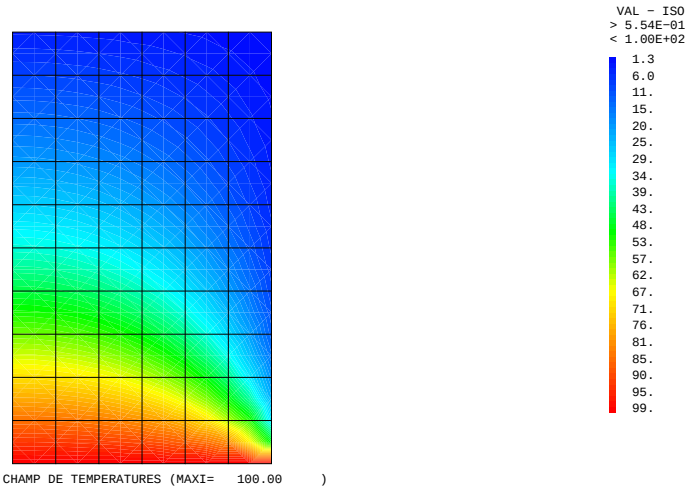
11.3.3 Commentaires

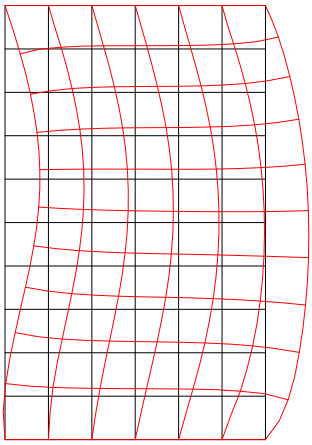
```
SIGT3 = THET MOD3 CHTER1 MAT3 ;
F3 = BSIG MOD3 SIGT3 ;
PRES3 = PRESSION MASS MOD3 100.E6 EA ;
CHAR3 = F3 ET PRES3 ;
```

L'opérateur THET(A) calcule les contraintes associées à un champ de températures. On associe donc au modèle mécanique (défini par MOD3 et MAT3) le champ de températures CHTER1 calculé au cours de la première étape (avec les modèles thermiques MOD1 et MOD2).
L'opérateur BSIG calcule le champ de forces nodales résultant de l'intégration d'un champ de contraintes. L'objet F3 représente donc le champ de forces thermiques.
L'opérateur PRES(SION) calcule les forces nodales équivalentes à une pression. Le mot-clé MASS précise que la pression est appliquée sur des éléments massifs.

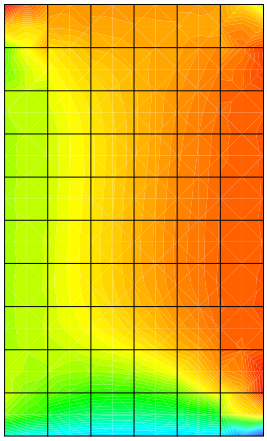
```
SIGTOT3 = SIGMA MOD3 MAT3 DEP3 ;
SIGM3 = SIGTOT3 - SIGT3 ;
```

L'opérateur SIGM(A) calcule le champ de contraintes à partir du champ de déplacements DEP3 résultant des chargements thermiques et mécaniques. L'objet SIGTOT3 représente donc le champ de contraintes totales. Afin d'obtenir uniquement les contraintes mécaniques (SIGM3), on soustrait les contraintes thermiques (SIGT3) aux contraintes totales.
Nous ne reviendrons pas sur le post-traitement où sont tracées les isovaleurs des contraintes mécaniques.

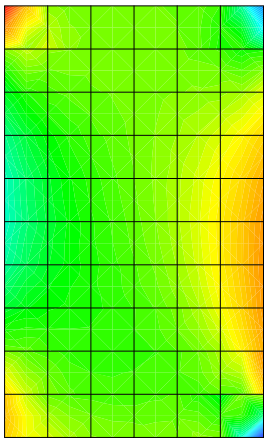




DEFORMEE APRES CHARGEMENT THERMIQUE ET PRESSION



CONTRAINTES S_XX (MAXI= 18.635 MPa)



CONTRAINTES S_YY (MAXI= 118.02 MPa)

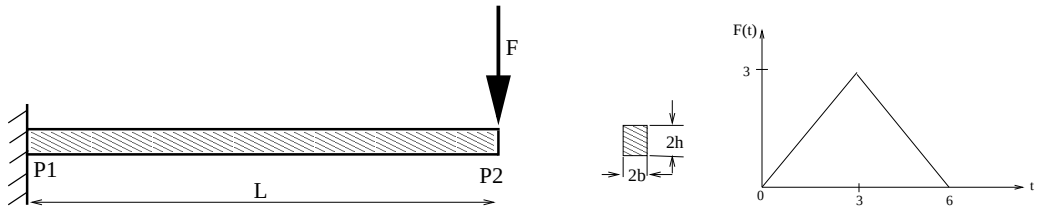
Chapitre 12

Calculs non linéaires

Nous allons présenter l'utilisation de la procédure PASAPAS qui est l'opérateur principal de calculs non linéaires de Cast3M. Cette procédure peut s'utiliser pour des calculs non linéaires de mécanique ou de thermique ou encore des calculs couplés thermo-mécaniques :
- en mécanique elle effectue un calcul non linéaire incrémental. La non linéarité peut provenir soit du matériau (plasticité), soit des grands déplacements, soit des deux à la fois.
- en thermique elle effectue un calcul linéaire et non linéaire en tenant compte de la conduction, de la convection et du rayonnement.

L'exemple choisi pour illustrer l'utilisation de PASAPAS est le calcul d'une poutre en flexion en comportement élasto-plastique. La poutre est encastrée d'un côté et une force fléchissante s'exerce sur l'extrémité libre.

12.0.4 Données du problème



- $L = 10$
- $b = 0.05$
- $h = 0.5$
- $E = 10^7$
- $E_t = 0.1 * E$
- $\sigma_e = 10^4$
- $F_{tot} = -10 * F(t)$

12.0.5 Fichier de données

```
OPTION ECHO 0;
TITRE 'POUTRE EN FLEXION  ';
OPTION DIME 3 ELEM SEG2 ;
*
*--------------------DEFINITION DE LA GEOMETRIE-------------------*
*
P1 = 0.  0.  0. ;
```

```
P2 = 10. 0.  0. ;
L1 = P1 D 1 P2  ;
TRACE QUAL L1 ;
*
*-----------------CHOIX DU COMPORTEMENT ET DU MODELE----------------*
*
MOD1  = MODEL L1 MECANIQUE ELASTIQUE PLASTIQUE CINEMATIQUE POUT ;
MAT1 = MATE MOD1 YOUN 1.E7 NU 0.3 SIGY 1.E4 H (0.1*1.E7) ;
CAR1 = CARA MOD1 SECT 0.1 INRY 8.333E-5 INRZ 8.333E-3 TORS 0.0001
       VECT (0. 1. 0.) DX 0. DY 0. DZ 0.3333 ;
*
*-----------------FORCES ET CONDITIONS AUX LIMITES----------------*
*
AMPFY = -10. ;
F1 = FORCE (0. AMPFY 0.) P2 ;
CL1 = BLOQ  DEPLA ROTA P1 ;
*
*----------------------------CHARGEMENT---------------------------*
*
LI1 = PROG 0. 3. 6. ;
LI2 = PROG 0. 3. 0. ;
EV = EVOL MANU 'Temps' LI1 'F(t)' LI2 ;
CHA1 = CHAR MECA F1 EV ;
DESS EV TITRE 'Ftot= -10.F(t)' ;
*
*-------------------------CALCUL PAS A PAS------------------------*
*
LIS_TPS = PROG 0. PAS 0.05 6. ;
TAB1 = TABLE ;
TAB1.'MODELE'               = MOD1 ;
TAB1.'CARACTERISTIQUES'     = MAT1 ET CAR1 ;
TAB1.'CHARGEMENT'           = CHA1 ;
TAB1.'BLOCAGES_MECANIQUES' = CL1  ;
TAB1.'TEMPS_CALCULES'       = LIS_TPS ;
TAB1.'TEMPS_SAUVES'         = LIS_TPS ;
PASAPAS TAB1 ;
*
*-------------------------POST TRAITEMENT-------------------------*
*
SI (TAB1.ERREUR) ;
  MESS 'Erreur reperee lors de l execution du calcul' ;
SINON ;
  MESS 'Aucune erreur lors de l execution du calcul' ;
  SI TAB1.CONV ;
    MESS 'Convergence du calcul' ;
  SINON ;
    MESS
    'Pas de convergence du calcul pour le nombre d'iterations demande' ;
  FINSI ;
FINSI ;
TAB2 = TAB1.DEPLACEMENTS ;
NDIM = DIME TAB2 ;
IB = 0 ;
LIS1 = PROG ;
LIS2 = PROG ;
REPETER BOUC1 NDIM ;
  DDD = EXTR (TAB2.IB) P2 UY ;
```

```
  LIS1 = PROG 0. PAS 0.5 30. PAS -0.5 0. ;
  LIS2 = INSER LIS2 &BOUC1 (-1.*DDD) ;
  IB = IB+1 ;
FIN BOUC1 ;
TITRE 'COURBE DEPLACEMENT-FORCE' ;
EV1 = EVOL MANU LIS2 'DEPLACEMENT' LIS1 'FORCE' ;
DESS EV1 ;
*
FMAX = (NDIM/2) ;
FFIN = NDIM - 1 ;
CHDEPM = TAB2.FMAX ;
CHDEPF = TAB2.FFIN ;
DEF0 = DEFO L1 CHDEPF 0. BLAN ;
DEF1 = DEFO L1 CHDEPM 1. ROUGE ;
DEF2 = DEFO L1 CHDEPF 1. VERT ;
OEIL = 0. 0. 1000. ;
TITRE 'DEFORMEE DE LA POUTRE APPLICATION FORCE MAXI' ;
TRAC OEIL (DEF0 ET DEF1)  ;
TRAC OEIL (DEF0 ET DEF2) TITRE 'DEFORMEE DE LA POUTRE APRES RELACHE' ;
*
FIN;
```

12.0.6 Commentaires

```
MOD1 = MODEL L1 MECANIQUE ELASTIQUE PLASTIQUE CINEMATIQUE POUT ;
MAT1 = MATE MOD1 YOUN 1.E7 NU 0.3 SIGY 1.E4 H (0.1*1.E7) ;
```

Le modèle MOD1 permet de définir le comportement plastique cinématique du matériau. Les caractéristiques du matériau sont précisées dans le champ MAT1 : on définit le module de Young (YOUN 1.E7), le coefficient de Poisson (NU 0.3), la limite élastique (SIGY 1.E4), le module d'écrouissage (H (0.1*1.E7)).
Il est bien évident que les caractéristiques matérielles dépendent du modèle de plasticité utilisé, il n'est que trop conseillé de se reporter à la notice de MATE pour bien vérifier les données.

```
LI1 = PROG 0. 3. 6. ;
LI2 = PROG 0. 3. 0. ;
EV = EVOL MANU 'Temps' LI1 'F(t)' LI2 ;
CHA1 = CHAR MECA F1 EV ;
DESS EV TITRE 'Ftot= -10.F(t)' ;
```

La liste de réels LI1 correspond à une liste de temps, la liste LI2 correspond au facteur multiplicatif de la force en fonction du temps. Ces deux listes sont rassemblées dans l'évolution EV.
L'objet de chargement mécanique CHA1 permet de définir le chargement comme étant le produit de la force F1 par un facteur multiplicatif qui varie en fonction du temps selon l'évolution EV.

```
LIS_TPS = PROG 0. PAS 0.05 6. ;
TAB1 = TABLE ;
TAB1.'MODELE' = MOD1 ;
TAB1.'CARACTERISTIQUES' = MAT1 ET CAR1 ;
TAB1.'CHARGEMENT' = CHA1 ;
TAB1.'BLOCAGES_MECANIQUES' = CL1 ;_ TAB1.'TEMPS_CALCULES' = LIS_TPS ;
TAB1.'TEMPS_SAUVES' = LIS_TPS ;
PASAPAS TAB1 ;
```

La résolution du calcul non linéaire se fait avec la procédure PASAPAS qui utilise un objet de type TABLE dans lequel sont stockés les données et les résultats. Dans un premier temps, l'utilisateur doit donc définir cette

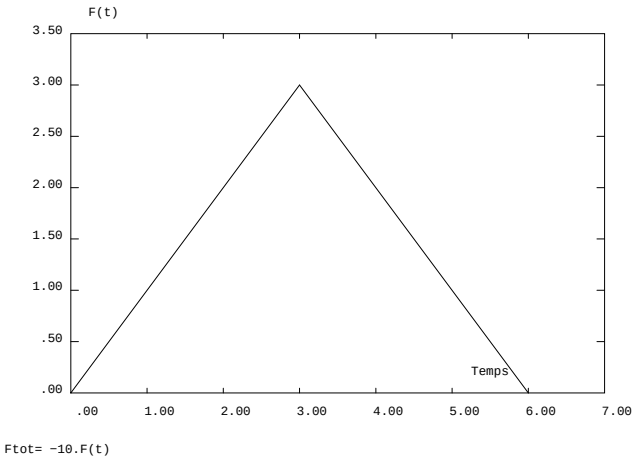


table qui est indicée par des mots-clés. Dans notre cas, on précise :
- le modèle : TAB1.'MODELE'
- les caractéristiques matérielles : TAB1.'CARACTERISTIQUES'
- le chargement : TAB1.'CHARGEMENT'
- les conditions aux limites : TAB1.'BLOCAGES_MECANIQUES'
- les incréments temporels du calcul : TAB1.'TEMPS_CALCULES'. Le calcul va donc réaliser 121 itérations temporelles pour chaque pas de pseudo-temps entre 0 et 6.
- la liste des temps à sauver : TAB1.'TEMPS_SAUVES'. Ici la liste est identique à la liste des temps de calcul, on aurait pu l'omettre.

Le lancement du calcul se fait par l'instruction : PASAPAS TAB1 .
Notons que pour un incrément de temps on peut avoir plusieurs itérations qui sont dues à des corrections plastiques. On aurait pu spécifier le nombre de ces itérations dans TAB1.MAXITERATION.

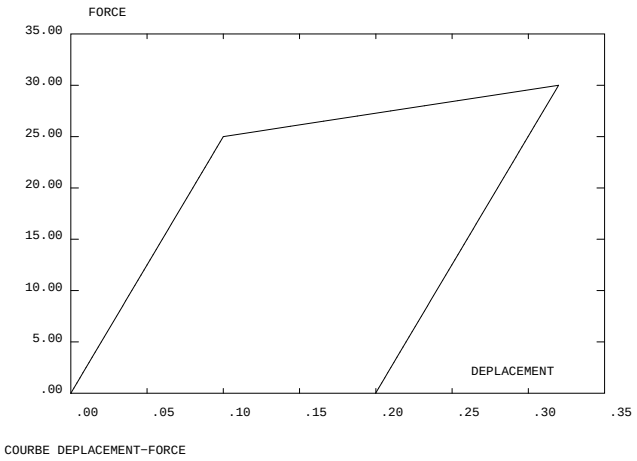
```
SI (TAB1.ERREUR) ;
MESS 'Erreur reperee lors de l execution du calcul' ;
SINON ;
MESS 'Aucune erreur lors de l execution du calcul' ;
SI TAB1.CONV ;
MESS 'Convergence du calcul' ;
SINON ;
MESS 'Pas de convergence du calcul pour le nombre d'iterations demande' ;
FINSI ;
FINSI ;
```

On utilise ici des conditions logiques pour vérifier si le calcul s'est déroulé correctement :
- si le contenu de la variable logique TAB1.ERREUR est 'VRAI' cela indique qu'il y a eu une erreur au cours de l'exécution de la procédure ;
- si le contenu de la variable logique TAB1.CONV est 'VRAI' cela indique que le calcul a convergé avant ou pour le nombre maximal d'itérations autorisées.

```
TAB2 = TAB1.DEPLACEMENTS ;
NDIM = DIME TAB2 ;
IB = 0 ;
LIS1 = PROG ;
LIS2 = PROG ;
```

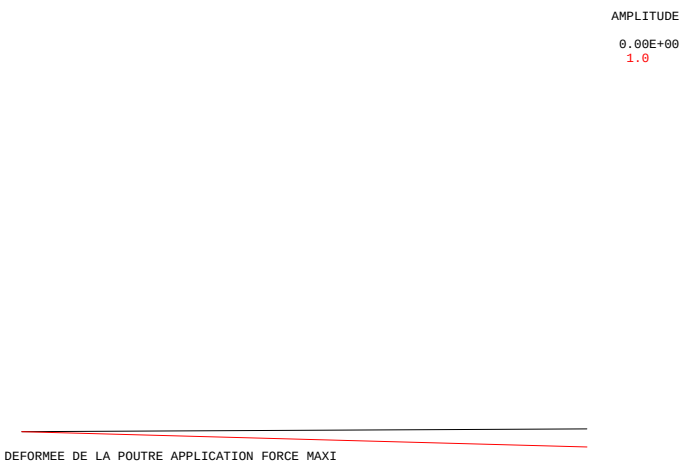
```
REPETER BOUC1 NDIM ;
DDD = EXTR (TAB2.IB) P2 UY ;
LIS1 = PROG 0. PAS 0.5 30. PAS -0.5 0. ;
LIS2 = INSER LIS2 &BOUC1 (-1.*DDD) ;
IB = IB+1 ;
FIN BOUC1 ;
TITRE 'COURBE DEPLACEMENT-FORCE' ;
EV1 = EVOL MANU LIS2 'DEPLACEMENT' LIS1 'FORCE' ;
DESS EV1 ;
```

TAB1.DEPLACEMENTS est une table contenant les résultats des déplacements. Les indices de cette table sont des entiers correspondant aux numéros des temps de sauvegarde (0, 1 ,... ,NDIM-1) ainsi TAB1.DEPLACEMENTS.0 est le champ de déplacements initial.
La liste de réels LIS2 contient le déplacement (en valeur absolue) UY du point P2 pour chaque pas de pseudo-temps. La liste de réels LIS1 contient la force appliquée (en valeur absolue) pour chaque pas de pseudo-temps. Les deux listes permettent de créer l'objet EV1 représentant l'évolution de la force en fonction du déplacement.

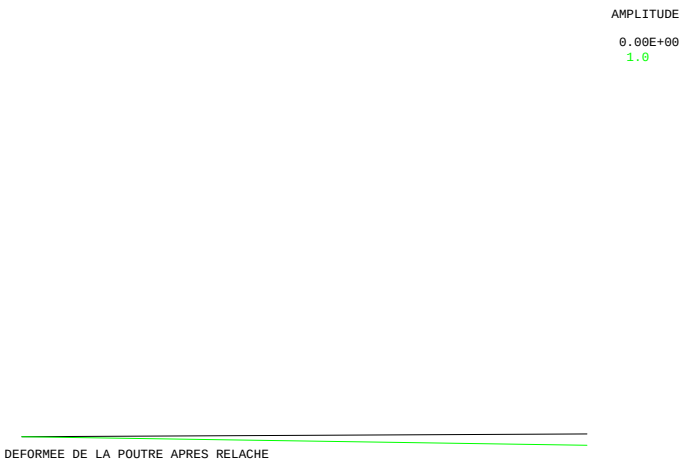



```
FMAX = (NDIM/2) ;
FFIN = NDIM - 1 ;
CHDEPM = TAB2.FMAX ;
CHDEPF = TAB2.FFIN ;
DEF0 = DEFO L1 CHDEPF 0. BLAN ;
DEF1 = DEFO L1 CHDEPM 1. ROUGE ;
DEF2 = DEFO L1 CHDEPF 1. VERT ;
OEIL = 0. 0. 1000. ;
TITRE 'DEFORMEE DE LA POUTRE APPLICATION FORCE MAXI' ;
TRAC OEIL (DEF0 ET DEF1) ;
TRAC OEIL (DEF0 ET DEF2) TITRE 'DEFORMEE DE LA POUTRE APRES RELACHE' ;
```

L'objet CHDEPM contient le champ de déplacements résultant de l'effort maximal (pour le pseudo-temps t=3) ; l'objet CHDEPF contient le champ de déplacement final (effort nul et pseudo-temps t=6). À partir de ces champs de déplacements et de la géométrie initiale (L1), on crée des objets de type DEFORME. Afin d'obtenir la structure non déformée, on utilise un coefficient d'amplification nul (cet artifice est nécessaire pour la directive TRAC qui ne s'applique qu'à des objets de même type). La déformée de la poutre après relâche montre qu'il y a bien eu plastification.



DEFORMEE DE LA POUTRE APPLICATION FORCE MAXI



DEFORMEE DE LA POUTRE APRES RELACHE

Index