LANGAGE ET PROCEDURES

Philippe PASQUET
30/10/1997

TABLE DES MATIERES

AVERTISSEMENT

Le volume Langage fait partie d'un ensemble comprenant les titres suivants

Langage et Description des procédures
Maillage
Vérification des données
Thermique des Structures
Mécanique des Structures - I
Mécanique des Structures - II
Mécanique des Fluides
Electromagnétisme
Post-Traitements

Nous avons repris dans ce volume, l'ensemble des opérateurs, procédures, directives permettant de programmer avec CASTEM2000®. Ils ne sont pas décrits dans leur intégralité mais dans leur acception la plus couramment utilisée. Le lecteur intéressé peut, pour obtenir l'intégralité des possibilités d'un opérateur, faire **INFO** nom ; dans CASTEM2000®.

Nous avons aussi essayé de faire un peu plus qu'un guide d'utilisation. Le lecteur s'en rendra, nous l'espérons, compte tout au long de ce volume et en particulier dans les premiers et derniers chapitres.

Dans la pratique de la description et de l'utilisation, il ne sera jamais possible de dissocier l'aspect langage de l'aspect plus classique, regroupant les fonctionnalités de maillage, de calcul, de post-traitements et de tracé (pour ne parler que d'éléments finis), celles-ci étant rééllement partie intégrante de celui-là.

Ce volume, comme l'ensemble de ce manuel, est nécessairement incomplet et malheureusement, il n'est pas exempt d'erreurs. Nous serions particulièrement reconnaissants aux lecteurs qui nous signaleront toute imperfection.

Nous n'avons pas repris de manière systématique la description des erreurs possibles dans CASTEM2000®. Les erreurs de syntaxe sont bien contrôlées et le diagnostic est relativement clair sauf dans le cas où le point virgule (**;**) a été omis. Les erreurs les plus sournoises sont la conséquence de l'ouverture et de la permissivité de CASTEM2000® qui permet d'enchaîner toutes les opérations.

Il y a très peu de valeurs par défaut dans CASTEM2000® : dans la suite, on trouvera un essai de recensement de ces valeurs (voir page 45). Pour attirer l'attention du lecteur-utilisateur signalons les numéros d'unités logiques dans **OPTIon**.

Rappelons enfin que tout nom d'objets (choisi par l'utilisateur) doit être différent d'un nom d'opérateur (imposé par CASTEM2000® -sauf directive **MOT**-). Pour ne pas être handicaper par cette restriction, on peut mettre les noms d'opérateurs entre **''**.

Le contenu de ce volume est cohérent avec la version 1998.

1. FABRICATION DE NOMBRES

Il est possible de construire des variables qui pourront servir de paramètres. Le type des variables est imposée par leur définition.

R = 5 ; R est un nombre ENTIER
J = 4. ; J est un nombre FLOTTANT (réel)

1.1 Les opérations élémentaires

+, -, *, /, **

Elles sont utilisables avec les mêmes restrictions qu'en arithmétique mais ce sont des opérateurs qui doivent donc être séparés des opérandes (par un blanc). Les opérations sont effectuées par ordre d'apparition en tenant compte des parenthèses les plus internes (il n'y a pas d'opérations prioritaires). Chacun des opérateurs nécessite deux opérandes. Certains (+, *) sont commutatifs.

x = 2 **+** 3 ***** 2 ; (x = 10)
x = 2 **+** (3 ***** 2) ; (x = 8)

Remarques importantes:

*La variable **-X** provoque une erreur puisque - est un opérateur qui attend deux opérandes. Pour obtenir l'opposé d'une variable, il faut faire x = 0. - x ; (ou x = -1. * x).*

Ne pas confondre + et PLUS d'une part et - et MOINs d'autre part (voir le volume MAILLAGE).

*On peut utiliser ** pour calculer une racine (par exemple ** 0.5 pour la racine carrée).*

1.2 Les fonctions élémentaires

Dans ce chapitre, on applique les opérateurs à des nombres. On peut aussi les utiliser avec d'autres types d'objets (CHPOINT, MCHAML en particulier). On prendra garde au fait que certaines sont reconnues par trois caractères et d'autres par quatre.

a) ABSolu

Calcul de la valeur absolue d'un nombre. Le résultat a le même type que l'argument.

x = **ABS** y ;
y ENTIER ou FLOTTANT ou LISTREEL ou CHPOINT ou MCHAML

b) ArcTanGente

Calcul de l'arc tangente d'un nombre (ou du rapport des deux nombres qui suivent). Le résultat a le même type que l'argument.

a = **ATG** x (y) ;

c) COSinus

Calcul du cosinus d'un angle en degrés. Le résultat a le même type que l'argument.

c = **COS** a ;

a ENTIER ou FLOTTANT ou LISTREEL ou CHPOINT ou MCHAML

d) COSinusHyperbolique

Calcul du cosinus hyperbolique d'un angle en degrés. Le résultat a le même type que l'argument.

c = **COSH** a ;

a ENTIER ou FLOTTANT ou LISTREEL ou CHPOINT ou MCHAML

e) ENTIer

Conversion d'un nombre FLOTTANT en nombre ENTIER.

x = **ENTI** y ;

y FLOTTANT

f) ERF

Calcul de l'intégrale $\frac{2}{\sqrt{\pi}}\int_0^z e^{-x^2}\,dx$. Le résultat a le même type que l'argument.

y = **ERF** z ;

z FLOTTANT (limite de l'intégration) ou ENTIER ou LISTREEL ou CHPOINT ou MCHAML

y FLOTTANT

g) EXPonentielle

Calcul de l'exponentielle d'un nombre. Le résultat a le même type que l'argument.

x = **EXP** y ;

y FLOTTANT ou LISTREEL ou CHPOINT ou MCHAML

h) FLOTtant

Conversion d'un nombre ENTIER en nombre FLOTTANT.

x = **FLOT** y ;

y ENTIER

i) LOGarithme

Calcul du logarithme naturel d'un nombre positif. Le résultat a le même type que l'argument.

x = **LOG** y ;

y FLOTTANT ou LISTREEL ou CHPOINT ou MCHAML

j) SIGNe

Le résultat est de type ENTIER ou FLOTTANT.

x= **SIGN** (**FLOT**) y ;

y ENTIER ou FLOTTANT

x ENTIER par défaut ou FLOTTANT

k) SINus

Calcul du sinus d'un angle en degrés. Le résultat a le même type que l'argument.

c = **SIN** a ;

a ENTIER ou FLOTTANT ou LISTREEL ou CHPOINT ou MCHAML

l) SINusHyperbolique

Calcul du sinus hyperbolique d'un angle en degrés. Le résultat a le même type que l'argument.

c = **SINH** a ;

a ENTIER ou FLOTTANT ou LISTREEL ou CHPOINT ou MCHAML

m) TANgenteHyperbolique

Calcul de la tangente hyperbolique d'un angle en degrés. Le résultat a le même type que l'argument.

c = **TANH** a ;

a ENTIER ou FLOTTANT ou LISTREEL ou CHPOINT ou MCHAML

D'une manière générale, on retrouve les mêmes restrictions qu'en arithmétique ou en FORTRAN 77.

2 LES TYPES D'OBJET

L'opérateur TYPE permet de retrouver le type de n'importe quel objet.

```
mm = TYPE ob1 ;
        ob1
        mm     MOT
```

Les types d'objets sont des mots de huit caractères au maximum. On peut classer de la manière suivante les types d'objets actuellement disponibles sachant qu'il y a aussi une notion de sous-type que nous allons décrire également.
On peut aussi retrouver tous les objets d'un même type (voir page 14).

2.1 Intérêt général

ENTIER (voir le chapitre 1. FABRICATION DE NOMBRES page 6)

LISTENTI créé par les opérateurs LECTure, INSErer, REMPlacer, ENLEver.

ll = **LECT** (i) (i1 **PAS** ip i2) (n ***** i) ;
- i suite de nombres ENTIER
- PAS engendre les ENTIER de i1 à i2 par pas ENTIER de ip (qui peut être négatif)
- * engendre n (ENTIER) fois le nombre ENTIER i

ll1 = **ENLE** ll ipo ;
- ll LISTENTI
- ipo ENTIER

suppression de l'ENTIER en ipo^{e} position dans ll

ll1 = **INSE** ll ipo j1 ;
- ll LISTENTI
- ipo ENTIER
- j1 ENTIER

insertion de l'ENTIER j1 en ipo^{e} position dans ll

REMP ll ipo j1 ;
- ll LISTENTI
- ipo ENTIER
- j1 ENTIER

remplacement par l'ENTIER j1 de la ipo^{e} position dans ll

FLOTTANT (voir le chapitre 1. FABRICATION DE NOMBRES page 6)

LISTREEL créé par les opérateurs PROGression, INSErer, REMPlacer, ENLEver.

ll = **PROG** (i) (i1 **PAS** ip i2) (n ***** i) ;
- i suite de nombres FLOTTANT
- PAS engendre les FLOTTANT de i1 à i2 par pas FLOTTANT de ip (qui peut être négatif)
- * engendre n (ENTIER) fois le nombre FLOTTANT i

ll1 = **ENLE** ll ipo ;
- ll LISTREEL
- ipo ENTIER

suppression du FLOTTANT en ipo^{e} position dans ll

ll1 = **INSE** ll ipo j1 ;

ll LISTREEL
ipo ENTIER
j1 FLOTTANT
insertion du FLOTTANTj1 en ipo^{e} position dans ll

REMP ll ipo j1 ;
ll LISTREEL
ipo ENTIER
j1 ENTIER
remplacement par le FLOTTANT j1 de la ipo^{e} position dans ll

MOT créé par l'opérateur MOT
ll = **MOT** ;

LISTMOTS créé par l'opérateur MOTS, INSErer, REMPlacer, ENLEver.
ll = **MOTS** (mot) ;
i suite de MOT
ll1 = **ENLE** ll ipo ;
ll LISTMOTS
ipo ENTIER
suppression du MOT en ipo^{e} position dans ll
ll1 = **INSE** ll ipo m1 ;
ll LISTMOTS
ipo ENTIER
m1 MOT
insertion du MOT m1 en ipo^{e} position dans ll
REMP ll ipo m1 ;
ll LISTMOTS
ipo ENTIER
m1 MOT
remplacement par le MOT m1 de la ipo^{e} position dans ll

LOGIQUE (voir chapitre 3.1 Fabrication d'objets de type LOGIQUE page 12)

TABLE (voir chapitre 3.5 Les tables page 15 et utilisation de la procédure PASAPAS dans le volume MECANIQUE DES STRUCTURES - II ou le volume THERMIQUE DES STRUCTURES)

PROCEDUR (voir chapitre 4. DESCRIPTION DES PROCEDURES page 16)

EVOLUTIO créé par l'opérateur EVOLution (voir page 42 ou dans le volume POST-TRAITEMENTS pour des exemples)

TEXTE créé par l'opérateur TEXTe

2.2 Objets de maillage

MAILLAGE (voir le volume MAILLAGE pour la liste des opérateurs).

POINT (voir le volume MAILLAGE pour la liste des opérateurs).

2.3 Objets de calcul

CHPOINT (voir Note sur les CHPOINT dans le volume MECANIQUE DES STRUCTURES - I ou dans le volume THERMIQUE DES STRUCTURES).

LISTCHPO créé par l'opérateur

MMODEL créé par l'opérateur MODEle à partir d'un MAILLAGE (voir le volume MECANIQUE DES STRUCTURES - I ou le volume THERMIQUE DES STRUCTURES).

MCHAML créé par les opérateurs MATEriau, CARActéristique à partir d'un MMODEL (voir le volume MECANIQUE DES STRUCTURES - I ou le volume THERMIQUE DES STRUCTURES), ou par les opérateurs SIGMa (à partir d'un CHPOINT, d'un MCHAML et d'un MMODEL) ou VMISes (à partir d'un MCHAML et d'un MMODEL) (voir le volume POST-TRAITEMENTS).

RIGIDITE créé par les opérateurs RIGIdité, MASSe, CONDuctivité, CAPAcité à partir d'un MMODEL et d'un MCHAML (voir le volume MECANIQUE DES STRUCTURES - I ou le volume THERMIQUE DES STRUCTURES).

SOLUTION créé par l'opérateur VIBRation à partir de deux RIGIDITE (raideur et masse) (voir le volume MECANIQUE DES STRUCTURES - I).

CHARGEME créé par l'opérateur CHARgement à partir d'un CHPOINT et d'un EVOLUTIO (voir le volume MECANIQUE DES STRUCTURES - II ou le volume THERMIQUE DES STRUCTURES).

2.4 Post-traitements

DEFORME créé par l'opérateur DEFOrmé à partir d'un CHPOINT et d'un MAILLAGE (voir le volume POST-TRAITEMENTS).

VECTEUR créé par l'opérateur VECTeur à partir d'un CHPOINT ou d'un MCHAML (voir le volume POST-TRAITEMENTS).

2.5 Remarques

Ils sont reconnus par l'orthographe indiquée ci-dessus. De manière interne à CASTEM2000®, ces différents types permettent à un opérateur ou à une procédure:

De vérifier si les opérandes représentent bien l'information attendue,

D'avoir (en général), une syntaxe non positionnelle quand les opérandes sont tous de type différent.

3 LES OPERATEURS GENERAUX

Ce sont, en général, des opérateurs qui sont des équivalents des fonctions disponibles dans un langage informatique. Par analogie, on prendra l'exemple du Fortran77 et du C.

3.1 Fabrication d'objets de type LOGIQUE

Les objets de type LOGIQUE peuvent prendre deux valeurs: VRAI ou FAUX. Ils sont fabriqués par les opérateurs suivants:

GIBIANE	FORTRAN77	C	Objets possibles
<	LT	<	ENTIER, FLOTTANT
<EGal	LE	<=	ENTIER, FLOTTANT
EGAl	EQ	==	ENTIER, FLOTTANT, MOT, TEXTE, POINT, LOGIQUE, LISTENTI
>EGal	GE	>=	ENTIER, FLOTTANT
>	GT	>	ENTIER, FLOTTANT
NEGal	NE	|=	ENTIER, FLOTTANT, MOT, CHPOINT, POINT
EXISte			
ET	AND	&&	LOGIQUE
OU	OR	||	LOGIQUE
NON	NOT	|	LOGIQUE

D'une manière générale, il vaut mieux mettre le nom de l'opérateur en première position.

- Pour l'opérateur EGAl, il peut y avoir une tolérance

  y = **EGA** x z (crit) ;

  y est VRAI si |x-z| < crit

- Pour l'opérateur EXISte, il y a plusieurs possibilités:

  tester l'existence d'un objet donné (toto)

  lg = **EXIS** toto ;

  tester l'existence d'un indice (ind1) dans une table (ta1)

  lg = **EXIS** ta1 ind1 ;

  tester l'existence d'une composante donnée (nc1) dans un CHPOINT, un NUAGE ou un MCHAML (ch1)

  lg = **EXIS** ch1 nc1 ;

  tester l'existence d'un mot (mo1) dans un LISTMOTS (lm1)

  lg = **EXIS** lm1 mo1 ;

  tester la présence de CONTraintes dans une SOLUTION (sol1)

  lg = **EXIS CONT** sol1 ;

  tester s'il existe une FORMulation donnée dans un MMODEL (md1)

  lg = **EXIS FORM** md1 ;

  tester la présence d'un CONStituant donné dans un MMODEL (md1)

  lg = **EXIS CONS** md1 ;

  tester la présence d'un ELEMent fini donné dans un MMODEL (md1)

  lg = **EXIS ELEM** md1 ;

  tester la présence d'un chargement élémentaire (mc1) donné dans un CHARGEME

(ch1)

lg = **EXIS** ch1 mc1 ;

3.2 Les tests

Les opérateurs suivants utilisent des objets de type LOGIQUE.

GIBIANE	FORTRAN77	C
SI	IF	if
SINOn	ELSE ELSEIF	else
FINSi	ENDIF	end

Dans tous les cas, ils opèrent sur des logiques écrits par leurs noms ou leurs définitions entre parenthèses. La séquence s'écrit ainsi:

SI log ;
...
SINO ;
...
FINS ;

log LOGIQUE créé par un opérateur du chapitre précédent.
On trouvera une première utilisation page 18.

3.3 Les boucles

Les opérateurs suivants permettent de construire des objets de type BOUCLE.

GIBIANE	FORTRAN77	C
REPEter	DO	do
ITERer	GOTO	goto
QUITter	GOTO	break
FIN	ENDDO	enddo

La directive REPEter permet de construire l'objet de type BOUCLE.

REPE toto (n) ;
...
QUIT toto ;
...
ITER toto ;
...
FIN toto ;

toto MOT nom de la boucle. L'indice variable est dans l'objet de type ENTIER &toto.
n ENTIER - nombre de parcours de la boucle (facultatif donc boucle infinie, il faut en sortir par QUITter)

L'ordre derrière REPEter est impératif.

QUITter permet de sortir de la boucle
ITERer permet de passer directement à l'indice suivant.
Pour un exemple voir par exemple page 18.

3.4 Les entrées-sorties

GIBIANE	FORTRAN77	C
MESSage	PRINT	printf
ACQUérir	READ	readf
OBTEnir	READ	readf

Ces opérateurs permettent d'écrire des procédures "interactives", par exemple, il est possible de récupérer des arguments sans les citer dans la liste d'appel.
Par un ordre OBTEnir, il n'est possible de récupérer qu'un ENTIER ou un FLOTTANT ou un MOT ou un LOGIQUE ou un LISTENTI ou un LISTREEL à l'exclusion de tout autre type.

MESS ' valeur de nomb ' ;
OBTE nom1(*type) [nomi(*type)];
nom1 le programme attend la réponse avant d'enchaîner. On peut imposer le type attendu. Dans le cas de LISTENTI ou de LISTREEL, il ne peut y avoir qu'un seul objet.

La directive ACQUérir permet d'acquérir des données sur un fichier quelconque que l'on peut spécifier dans OPTIon ACQUérir (par défaut l'unité logique 9) mais il ne faut pas oublier de faire le lien. On doit fournir autant d'arguments qu'il y a de données à lire. On peut lire des objets de type ENTIER, FLOTTANT, MOT, LISTENTI, LISTREEL. Derrière chaque argument, on peut imposer le type attendu.

MESS ' on attend quelques valeurs ' ;
OPTI ACQU 'nom du fichier ';
ACQU n*entier i*flottant m*mot ;
Sur le fichier "nom du fichier", il y aura les valeurs 1 2. toto.
Dans le cas ou on lit une liste, il faut indiquer le nombre de valeurs à lire
ACQU ll*listenti 4 ; permet de lire 4 valeurs de ll.

La directive LISTe permet d'afficher le contenu d'un objet. Le mode d'affichage du contenu dépend de l'objet. Elle peut être placée à n'importe quel endroit dans les données.

LIST ;
permet de lister les objets créés par ordre chronologique inverse et par type.
LIST a ;
permet de lister la valeur de a.
Si a n'existe pas il est de type MOT
Si a résulte d'une erreur, il est de type ANNULE
Si a est une variable, il y a impression de son type et de sa valeur.
LIST *type;
permet d'obtenir le nom de tous les objets de type "type".

La directive RESTituer est la directive duale de SAUVer. Elle permet à CASTEM2000® de

relire son propre résultat et d'enchaîner directement d'autres opérations. La lecture se fait par défaut sur l'unité logique 2 mais que l'on peut spécifier dans OPTIon RESTituer. La relecture doit se faire sous le format de la sauvegarde (binaire ou ASCII - FORMat).

REST (**FORM**) ;

La directive SAUVer permet de sauver plusieurs objets. La sauvegarde se fait par défaut sur l'unité logique 8 mais que l'on peut spécifier dans OPTIon SAUVer. La sauvegarde peut se faire sous forme de fichier ASCII (FORMat) en général très volumineux, ou binaire.

SAUV toto (titi ....) (**FORM**) ;

toto MAILLAGE ou autre type

Description du fichier :

Le fichier de sauvegarde dépend en grande partie de(s) (l') objet(s) sauvé(s).Néanmoins, il est constitué de

3.5 Les tables

La directive TABLe permet d'utiliser la notion de tableau indicé, pour stocker divers types d'objets. L'objet ainsi créé est de type TABLE. Les indices peuvent être n'importe quel type d'objets (en général MOT ou ENTIER mais aussi FLOTTANT). L'utilisation est très simple si l'on n'oublie pas de déclarer qu'un objet est de type TABLE. Pour voir une utilisation de cette notion, on se reportera avec profit à la procédure PASAPAS dans le volume MECANIQUE DES STRUCTURES - II ou dans le volume THERMIQUE DES STRUCTURES.

ta = **tabl** (tyta);

ta . ind = toto ;

ind indice de la table. Il faut séparer le nom de la table du nom de l'indice par ' . ' (un blanc, un point, un blanc).

toto objet contenu à l'indice ind de la table. Peut être lui même le mot TABL pour former des tables de tables ... sans aucune limitation.

4. DESCRIPTION DES PROCEDURES

Dans le cas général, le jeu de données peut, soit devenir complexe, soit, en partie ou en totalité, être réutilisable.
Il est possible de regrouper une série d'instructions dans un objet de type particulier, la PROCEDUR. C'est à peu près l'équivalent d'un sous-programme Fortran qui, à partir d'une liste d'arguments d'entrée fournit un (ou plusieurs) résultat(s).
L'objet PROCEDUR est créé par la directive DEBProcédure auquel on donne un nom et on ajoute la liste des arguments d'entrée avec éventuellement leur type.
Une directive ARGUment permet de lire les arguments (typés) à l'intérieur de la procédure.
Le(s) argument(s) de sortie est (sont) fourni(s) (s'il(s) existe(nt)) dans la dernière instruction de la procédure: FINProcédure. Les résultats seront de type quelconque, attribué par construction. Les arguments (qu'ils soient d'entrée ou de sortie) seront reconnus par leur ordre (et pas seulement par leur type). A part cette restriction, la procédure s'utilise, au moment ou elle est appelée et non au moment ou elle est construite, de la même manière qu'un opérateur. Elle doit être appelée par son nom entier (huit caractères au maximum) et non par sa réduction aux quatre premiers caractères.

4.1 Construction d'une procédure

L'opérateur DEBProcédure est suivi des arguments (objets) typés.

```
DEBP toto ob1*type ;
....
SI lo1 ;
        ARGU ob2*type ;
FINS ;
...
RESP (ob3) ;
...
FINP (ob4) ;
```

toto	MOT - nom de la procédure
ob1, ob2	MOT - argument(s) d'entrée
type	MOT - type de l'argument (qui sera obligatoire si précédé de *, facultatif si précédé de /)
ob3, ob4	MOT - argument(s) de sortie. Le type est défini par construction.

La directive ARGUment permet de fournir un (des) argument(s) d'entrée sans qu'il(s) soi(en)t défini(s) dans la liste d'entrée (derrière DEBProcédure). Elle est très intéressante dans les cas conditionnels.
La directive RESProcédure permet d'obtenir un (des) résultat(s) sans qu'il(s) soi(en)t défini(s) dans la liste de sortie (derrière FINProcédure).

4.2 Utilisation d'une procédure

Lors de la construction, les instructions ne sont pas exécutées mais stockées dans un objet de type PROCEDUR. Il faut l'appeler pour déclencher son exécution. Elle s'appelle comme un opérateur ou une directive dans lequel les opérandes seraient positionnels. Une procédure peut en appeler une autre ou s'appeler elle-même. Pour l'utiliser efficacement, il y a trois solutions:

• L'insérer dans un jeu de données, avant son utilisation effective
• La cataloguer dans son environnement à l'aide de la directive UTILisateur qui la mettra dans le fichier UTILPROC. Dans ce cas, DEBProcédure doit être précédée de la ligne $$$$ nomp

UTIL nomp nomfic ;

nomp PROCEDUR - nom de la procédure
nomfic TEXTE - nom du fichier '/.../..'

• La proposer à l'équipe de développement, pour insertion dans le fichier des procédures de CASTEM2000®, qui ne l'acceptera que si elle est d'intérêt général et suffisamment commentée.

L'une de ces trois opérations effectuée, il suffit de faire:

nsort = **NOMP** nent ;

NOMP	PROCEDUR - nom de la procédure
nent	liste ordonnée des arguments d'entrée (y compris ceux qui sont dans ARGUment)
nsort	liste ordonnée des arguments de sortie (si nécessaire)

4.3 Remarques

Une procédure peut ne pas avoir d'argument d'entrée ni de sortie.
Elle peut aussi avoir plusieurs arguments de sortie.
Une procédure peut en appeler une autre.
Une procédure peut s'appeler elle-même.
La valeur des arguments d'entrée n'est pas modifiée en sortie.

5. QUELQUES EXEMPLES UTILES DE PROCEDURE

On donne ici quelques exemples, certains académiques (inutiles !), d'autres plus utiles, de procédures avec des exemples d'utilisation et la liste des opérateurs, directives, procédures utilisés.
On peut construire des procédures pour de nombreuses applications (voir par exemple CALCULER ou PASAPAS).

5.1 Calcul de factorielle n

Calcul de factorielle n (voir aussi la procédure FACTORIE

a) Construction

```
debp facto n*entier ;
resu = 1 ;
repe ind1 n ;
        resu = resu * &ind1;
fin ind1 ;
finp resu ;
```

b) Utilisation

```
x = facto n ;
        n       ENTIER
        x       ENTIER
```

c) Opérateurs utilisés

*, DEBProcédure, FIN, FINProcédure, REPEter

5.2 Calcul de la racine carrée

a) Construction

```
debp sqrt x*flottant ;
si (x < 0.) ;
        mess 'racine carre d un nombre negatif' ;
        quit sqrt ;
sino ;
        y = x ** 0.5 ;
fins ;
finp y ;
```

b) Utilisation

```
x = sqrt y ;
        x       FLOTTANT ou MOT (si la racine n'existe pas)
        y       FLOTTANT
```

c) Opérateurs utilisés

**, <, DEBProcédure, FINProcédure, FINSi, QUITter, SI, SINOn

5.3 Calcul de la tangente

a) Construction

```
debp tan x*flottant ;
c = cos x ;
si (c ega 0.) ;
        mess 'tangente 90 degres ' ;
        quit tan ;
sino ;
        s = sin x ;
        y = s / c ;
oubl s ;
fins ;
oubl c ;
finp y ;
```

b) Utilisation

```
x = tan y ;
```

x	FLOTTANT ou MOT (si la tangente est infinie)
y	FLOTTANT - angle en degrés

c) Opérateurs utilisés

/, COSinus, DEBProcédure, EGAl, FINProcédure, FINSi, MESSage, OUBLier, QUITter, SI, SINus, SINOn

5.4 Calcul du barycentre

On peut aussi utiliser l'opérateur BARYcentre (voir le volume MAILLAGE).

a) Construction

```
debp barycent toto*maillage ;
n = nbno tot ;
p1 = toto poin init ;
pb = p1 moin p1 ;
repe ind1 n ;
        i = &ind1 ;
        p1 = toto poin i ;
        pb = pb plus p1 ;
fin ind1 ;
pb = pb / (flot n) ;
oubl n ; oubl i; oubl p1
finp pb ;
```

b) Utilisation

```
pp = BARYCENT toto ;
```

toto	MAILLAGE
pp	MAILLAGE (POI1)

c) Opérateurs utilisés

/, DEBProcédure, FIN, FINProcédure, MOINs, NBNOeud, OUBLier, PLUS, POINt, REPEter.

5.5 Transformation coordonnées cylindriques/cartésiennes

On peut aussi utiliser la procédure POINTCYL (et POINTSPH dans le cas des coordonnées sphériques)

POINTCYL permettant de définir un point par ses coordonnées cylindriques (rayon, angle, altitude si OPTIon DIMEnsion 3)

POINTSPH permettant de définir un point par ses coordonnées sphériques (rayon, angle, angle)

a) Construction

b) Utilisation

c) Opérateurs utilisés

5.6 Construction de l'objet repère

Construction d'un maillage représentant le repère.

a) Construction

```
debp repere ;
dim = vale dime ;
den = vale dens ;
si ( dim ega 2 ) ;
        x = den 0. ;
        y = 0. den ;
        glob = ((0. 0.) droi 1 x) et ((0. 0.) droi 1 y) ;
sino ;
        x = den 0. 0. ;
        y = 0. den 0. ;
        z = 0. 0. den ;
        glob = ((0. 0. 0.) droi 1 x) et ((0. 0. 0.) droi 1 y) et ((0. 0. 0.) droi 1 z ;
fins ;
finp glob ;
```

b) Utilisation

```
gl = repere ;
```

gl MAILLAGE (que l'on concatène avec le MAILLAGE dans la directive TRACer et l'option QUALifier dans l'instruction ou à la souris)

c) Opérateurs utilisés

DEBProcédure, DROIt, EGAl, FINProcédure, FINSi, SI, SINOn, VALEur

5.7 Maillage d'un cercle de centre et de rayon donné

Dicrétisation d'un cercle de rayon r, passant par un point de rayon donné et découpé automatiquement ou en un nombre d'éléments donné.

a) Construction

```
debp cerayon r*flottant centre*point deco*entier ;
x = coor 1 centre ;
y = coor 2 centre ;
xmr = x - r ;
xpr = x + r ;
ymr = y - r ;
ypr = y + r ;
p1 = xpr y ;
p2 = x ypr ;
p3 = xmr y ;
p4 = x ymr ;
lod = deco ega 0 ;
si lod ;
        ll = p1 cer3 p2 p3 cer3 p4 ;
sino ;
        dec = (deco + 1) / 2 ;
        ll = p1 cer3 dec p2 p3 cerc3 dec p4 ;
fins ;
oubl x; oubl y; oubl xmr; oubl xpr; oubl ymr; oubl ypr;
oubl dec ;
fin ll;
```

b) Utilisation

ll = **CERAYON** r q1 n ;

r	FLOTTANT
p1	POINT
n	ENTIER (nombre d'éléments, 0 si découpage automatique)
ll	MAILLAGE (les quatre points cardinaux sont nommés p1, p2, p3, p4)

c) Opérateurs utilisés

-, +, /, CER3, COORdonnées, DEBProcédure, EGAl, FINProcédure, FINSi, OUBLier, QUITter, SI, SINOn

5.8 Intersection de deux droites

Les droites sont définies chacune par deux points.

a) Construction

```
DEBP DROIDROI P1*POINT P2*POINT
        Q1*POINT Q2*POINT;
* EQUATION DE LA DROITE P1 P2
X1 = COOR 1 P1;
```

```
Y1 = COOR 2 P1;
X2 = COOR 1 P2;
Y2 = COOR 2 P2;
SI (X1 EGA X2);
  A1 = 1E6;
SINO;
  A1 = (Y2 - Y1 ) / (X2 - X1);
FINS;
B1 = Y1 - (A1 * X1);
* EQUATION DE LA DROITE Q1 Q2
X1 = COOR 1 Q1;
Y1 = COOR 2 Q1;
X2 = COOR 1 Q2;
Y2 = COOR 2 Q2;
SI (X1 EGA X2);
  A2 = 1E6;
SINO;
  A2 = (Y2 - Y1 ) / (X2 - X1);
FINS;
B2 = Y1 - (A2 * X1);
BB = B2 - B1;
AA = A1 - A2;
si (aa ega 0) ;
  quit droidroi ;
fins;
XX = BB / AA;
YY = A1 * XX + B1;
PC = XX YY;
oubl bb ;oubl aa ;oubl b2 ;oubl a2 ;oubl x1 ;oubl x2 ;oubl y1 ;oubl y2 ;
FINP PC;
```

b) Utilisation

p1 = **DROIDROI** q1 q2 r1 r2 ;

q1, q2, r1, r2 sont de type POINT

p1 est de type POINT ou MOT (si l'intersection n'existe pas)

c) Opérateurs utilisés

-, +, *, /, COORdonnées, DEBProcédure, EGAl, FINProcédure, FINSi, OUBLier, QUITter, SI, SINOn

5.9 Intersection d'une droite et d'un cercle

C'est un cas particulier de l'intersection d'une droite et d'une ellipse (voir page 35) qui est ici résolu analytiquement. La droite est définie par deux points et le cercle par trois points ou par son centre et un point.

a) Construction

DEBP DROICER3 P1*POINT P2*POINT

```
          Q1*POINT Q2*POINT Q3*POINT;
O1 = CENTCER3 Q1 Q2 Q3;
PC1 PC2 = DROICERC P1 P2 O1 Q1;
FINP PC1 PC2;
*****************************************
DEBP DROICERC P1*POINT P2*POINT
          O1*POINT Q1*POINT ;
* EQUATION DE LA DROITE P1 P2
X1 = COOR 1 P1;
Y1 = COOR 2 P1;
X2 = COOR 1 P2;
Y2 = COOR 2 P2;
SI (X1 EGA X2);
  A1 = 1E6;
SINO;
  A1 = (Y2 - Y1 ) / (X2 - X1);
FINS;
B1 = Y1 - (A1 * X1);
*  CALCUL DU RAYON
V1 = Q1 MOIN O1;
R1 = NORM V1;
R1 = R1 * R1;
XC = COOR 1 O1;
YC = COOR 2 O1;
* EQUATION DU SECON DEGRE
AA = A1 * A1 + 1;
BB = 2 * ( (A1 * B1) - (A1 * YC) - XC);
CC = (XC * XC) + (B1 * B1) + (YC * YC) - (2 * B1 * YC) - R1 ;
XX1 XX2 = SECOND AA BB CC;
si (ega (type xx1) mot) ;
  quit droicerc;
fins ;
  YY1 = A1 * XX1 + B1 ;
  YY2 = A1 * XX2 + B1 ;
  PC1 = XX1 YY1;
  PC2 = XX2 YY2;
oubl aa ;oubl bb ;oubl cc ;
oubl v1 ;oubl r1 ;oubl xc ;
oubl yc ;oubl a1 ;oubl b1 ;
oubl x1 ;oubl y1 ;oubl x2 ;oubl y2 ;
FINP PC1 PC2;
DEBP SECOND AA*FLOTTANT BB*FLOTTANT CC*FLOTTANT;
DELTA = (BB * BB) - (4 * AA * CC);
SI (DELTA < 0.);
  MESS ' PAS DE SOLUTION ';
  quit second ;
FINS;
SI (DELTA EGA 0.);
```

```
  XX1 = (0. - BB) / (2 * AA);
  XX2 = (0. - BB) / (2 * AA);
FINS;
SI (DELTA > 0.);
  XX1 = ((0 - BB) + (DELTA ** .5)) / (2 * AA);
  XX2 = ((0 - BB) - (DELTA ** .5)) / (2 * AA);
FINS;
oubl delta ;
FINP XX1 XX2;
```

b) Utilisation

p1 p2 = **DROICERC** q1 q2 o1 r1 ;
p1 p2 = **DROICER3** q1 q2 r1 r2 r3 ;

p1, p2 sont des POINT ou des MOT (si l'intersection n'existe pas). Les deux points peuvent être à la même position si droite et cercle sont tangents.

q1, q2 sont des POINT définissant la droite,
r1, r2, r3 sont des POINT qui appartiennent au cercle
o1 est un POINT centre du cercle.

c) Opérateurs utilisés

-, +, *, /, **, >, <, COORdonnées, DEBProcédure, EGAl, FINProcédure, FINSi, OUBLier, QUITter, SI, TYPE

On utilise la procédure SECOND pour résoudre une équation du second degré (on aurait pu aussi utiliser l'opérateur DEG3) et la procédure CENTCER3 (voir page 24)pour déterminer le centre d'un cercle défini par trois points.

5.10 Intersection de deux cercles

C'est un cas particulier de l'intersection de deux ellipses (voir page 27) qui est ici résolu analytiquement en fournissant 0, 1 (double) ou 2 points. Les deux cercles sont définis par

Chacun trois points,
Chacun un centre et un point,
Le premier par trois points et le second par un centre et un point.

a) Construction

```
DEBP CER3CER3 P1*POINT P2*POINT P3*POINT
        Q1*POINT Q2*POINT Q3*POINT;
O1 = CENTCER3 P1 P2 P3;
O2 = CENTCER3 Q1 Q2 Q3;
PC1 PC2 = CERCCERC O1 P1 O2 Q1;
FINP PC1 PC2;
************************************************
DEBP CER3CERC P1*POINT P2*POINT P3*POINT
        O2*POINT Q1*POINT ;
O1 = CENTCER3 P1 P2 P3;
PC1 PC2 = CERCCERC O1 P1 O2 Q1;
FINP PC1 PC2;
************************************************
```

```
DEBP CERCCERC O1*POINT P1*POINT
          O2*POINT Q1*POINT ;
*  CALCUL DU RAYON
V1 = P1 MOIN O1;
R1 = NORM V1;
RPR1 = R1;
RPR = R1 * R1;
XC1 = COOR 1 O1;
YC1 = COOR 2 O1;
*  CALCUL DU RAYON
V1 = Q1 MOIN O2;
R1 = NORM V1;
RDE1 = R1;
RDE = R1 * R1;
XC2 = COOR 1 O2;
YC2 = COOR 2 O2;
* CHANGEMENT DE VARIABLE
A = XC1 - XC2;
B = YC1 - YC2;
DIST = (A*A) + (B*B);
DIST = DIST ** .5;
RR = RPR1 + RDE1;
AA = RR - DIST;
 SI (B EGA 0.);
   ITEST = 0;
  SINO;
   ITEST = 1;
  FINS;
 SI (ITEST EGA 1);
*  EQUATION DE LA CORDE SI ELLE EXISTE
A1 = (0 - A) / B;
B1 = (A * A) + ( B * B) + RPR - RDE;
B1 = -.5 * B1 / B;
AA = (A1 * A1 ) + 1.;
BB = 2 * (A1 * B1) ;
CC = (B1 * B1) - RPR ;
* INTERSECTION DE LA CORDE AVEC LE PREMIER CERCLE
XX1 XX2 = SECOND AA BB CC;
  YY1 = A1 * XX1 + B1 ;
  YY2 = A1 * XX2 + B1 ;
  XX1 = XX1 + XC1;
  YY1 = YY1 + YC1;
  XX2 = XX2 + XC1;
  YY2 = YY2 + YC1;
  PC1 = XX1 YY1;
  PC2 = XX2 YY2;
FINS;
SI (ITEST EGA 0);
```

```
   RR = RPR1 + RDE1;
  RA = RR - A;
  SI ((RA < 1E-2) ET (RA > 0));
    XX1 = RPR1;
   FINS;
  SI ((RA > -1E-2) ET (RA < 0));
    XX1 = 0. - RPR1;
   FINS;
  SI ((RA < -1E-2) ET (RA > 1E-2));
  XX1 = RDE - RPR - (A * A) ;
  XX1 = .5 * XX1 / A;
  FINS;
  XX2 = XX1 * XX1;
 XX1 = XX1 + XC1;
  YY1 = RPR - XX2;
  YY2 = YY1 ** .5;
  YY1 = YY2 + YC1;
   XX2 = XX1;
  YY2 = YC1 - YY2 ;
 PC1 = XX1 YY1;
 PC2 = XX2 YY2;
FINS;
FINP PC1 PC2;
DEBP CENTCER3 Q1*POINT Q2*POINT Q3*POINT;
* EQUATION DE LA DROITE PERPENDICULAIRE A Q1 Q2
X1 = COOR 1 Q1;
Y1 = COOR 2 Q1;
X2 = COOR 1 Q2;
Y2 = COOR 2 Q2;
SI (Y1 EGA Y2);
 A1 = 1E6;
SINO;
 A1 = (X2 - X1 ) / (Y2 - Y1);
 A1 = 0. - A1;
FINS;
* CALCUL DU POINT MILIEU
XM1 = .5 * ( X1 + X2);
YM1 = .5 * ( Y1 + Y2);
B1 = YM1 - (A1 * XM1);
* EQUATION DE LA DROITE PERPENDICULAIRE A Q2 Q3
X3 = COOR 1 Q3;
Y3 = COOR 2 Q3;
SI (Y3 EGA Y2);
 A2 = 1E6;
SINO;
 A2 = (X2 - X3 ) / (Y2 - Y3);
 A2 = 0. - A2;
FINS;
```

```
* CALCUL DU POINT MILIEU
XM2 = .5 * ( X3 + X2);
YM2 = .5 * ( Y3 + Y2);
B2 = YM2 - (A2 * XM2);
XO1 = (B2 - B1) / (A1 - A2);
YO1 = A1 * XO1 + B1;
O1 = XO1 YO1;
FINP O1;
```

b) Utilisation

```
p1 p2 = CER3CER3 q1 q2 q3 r1 r2 r3 ;
p1 p2 = CER3CERC q1 q2 q3 o1 r1 ;
p1 p2 = CERCCERC o2 r2 o1 r1 ;
```

c) Opérateurs utilisés

-, +, *, /, **, >, <, COORdonnées, DEBProcédure, EGAl, ET, FINProcédure, FINSi, MOINs, NORMe, SI, SINOn

On utilise les procédures SECOND pour résoudre une équation du second degré (voir page 22 mais on aurait pu utiliser l'opérateur DEG3), CENTCER3 pour calculer le centre d'un cercle défini par trois points.

5.11 Intersection de deux ellipses

L'intersection de deux ellipses quelconques est recherchée par une méthode de Newton à partir d'un point initial fourni par l'utilisateur.

a) Construction

```
debp pereli l1*flottant;
ta = tabl;
ta . 0 = 1.;
ta . 1 = 1.0025;
ta . 2 = 1.01;
ta . 3 = 1.0226;
ta . 4 = 1.0404;
ta . 5 = 1.0635;
ta . 6 = 1.0922;
ta . 7 = 1.1269;
ta . 8 = 1.1679;
ta . 9 = 1.2162;
ta . 10 = 4. / pi;
n = 10;
j = 1 ;
repe b1 n;
  xj = (flot j ) / 10.;
  si (l1 < xj);
    xs = ta . j;
    xi = ta . (j - 1 );
    quit b1;
```

```
  fins ;
  j = j + 1;
fin b1;
fl1 = 10. * (xs - xi);
fl1 = fl1 * (l1 - xj);
fl1 = fl1 + xs;
finp fl1;
debp posi p1*point ce*point ax*flottant by*flottant;
x1 = coor 1 p1;
y1 = coor 2 p1;
xe = coor 1 ce;
ye = coor 2 ce;
xx = x1 - xe;
yy = y1 - ye;
* coef c2
bx2 = by * by * xx * xx;
by2 = by * by * yy * yy;
ay2 = ax * ax * yy * yy;
ax2 = ax * ax * xx * xx;
aa = bx2 - by2 + ay2 - ax2;
* coef const
cc = by2 + ax2 - (ax*ax*by*by);
* coef sc
bb = 2. * xx * yy * ((by*by) - (ax*ax));
* passage en tang arc moitie
*a4 = aa + cc;
a4 = bx2 + ay2  - (ax*ax*by*by);
a3 = -2. * bb;
a2 = 2. * ( cc - aa);
a1 = 0. - a3;
a0 = a4;
*debut newton
si (a1 ega 0.) ;
  t0 = .1;
sino ;
  t0 = 0.;
fins ;
thet1 = atg t0 1.;
thet1 = 2. * thet1;
repe bo;
t02 = t0 * t0;
t03 = t02 * t0;
t04 = t03 * t0;
  ff = (a4 * t04)
     + (a3 * t03)
     + (a2 * t02 )
     + (a1 * t0  )
     + a0 ;
```

```
  fp = (a4 * t03 * 4.)
     + (a3 * t02 * 3.)
     + (a2 * t0 * 2. )
     + a1 ;
  dt = ff / fp ;
  t0 = t0 - dt ;
  thet2 = atg t0 1.;
  thet2 = 2. * thet2;
  dt1 = thet2 - thet1;
  si ( ((abs dt) < 1e-9) et ((abs dt1) < 1e-9));
    quit bo;
  fins;
  thet1 = thet2;
  oubl dt;oubl ff; oubl fp;oubl dt1;
fin bo;
  thet = thet2 ;
oubl x1; oubl xe; oubl y1; oubl ye;
oubl xx; oubl yy; oubl by2; oubl bx2;
oubl ay2; oubl ax2; oubl aa; oubl bb;
oubl cc; oubl a0; oubl a1; oubl a2;
oubl a3; oubl a4; oubl t0; oubl dt;
oubl thet1; oubl a4; oubl t0; oubl dt;
finp thet;
debp ellielli
            ce1*point ax1*flottant by1*flottant p1*point
            ce2*point ax2*flottant by2*flottant p2*point;
si (ax1 ega by1);
  thet1 = 0.;
sino ;
  tht1 = posi p1 ce1 ax1 by1;
  thet1 = tht1;
si (tht1 > 90.);
  thet1 = tht1 - 90.;
  tr = ax1 ;
  ax1 = by1 ;
  by1 = tr ;
fins;
fins;
oubl tr;oubl tht1;
si (ax2 ega by2);
  thet2 = 0.;
sino ;
  tht2 = posi p2 ce2 ax2 by2;
  thet2 = tht2;
si (tht2 > 90.);
  thet2 = tht2 - 90.;
  tr = ax2 ;
  ax2 = by2 ;
```

```
  by2 = tr ;
fins;
fins;
oubl tr;oubl tht2;
*mess ' orientation 1 ' thet1 ' orientation 2 ' thet2;
a1 = pi * ax1 * by1;
a2 = pi * ax2 * by2;
ax12 = ax1 * ax1;
by12 = by1 * by1;
ax22 = ax2 * ax2;
by22 = by2 * by2;
* centre de l ellipse 1
xe1 = coor 1 ce1;
ye1 = coor 2 ce1;
* centre de l ellipse 2
xe2 = coor 1 ce2;
ye2 = coor 2 ce2;
l1 = (abs (ax1 - by1)) / (ax1 + by1);
fl1 = pereli l1;
*fl1 = 1.;
l1 = pi * (ax1 + by1) * fl1;
e1 = ((ax1 * ax1 ) - (by1 * by1));
e1 = (abs e1) ** .5;
de1 = 2. * e1;
si (ax1 > by1 );
  e1 = e1 / ax1;
sino ;
  e1 = e1 / by1;
fins;
l2 = (abs (ax2 - by2)) / (ax2 + by2);
*fl2 = 1.;
fl2 = pereli l2;
l2 = pi * (ax2 + by2) * fl2;
e2 = ((ax2 * ax2 ) - (by2 * by2));
e2 = (abs e2) ** .5;
de2 = 2. * e2;
si (ax2 > by2 );
  e2 = e2 / ax2;
sino ;
  e2 = e2 / by2;
fins;
xe2me1 = xe2 - xe1;
ye2me1 = ye2 - ye1;
c1 = cos thet1;
s1 = sin thet1;
c2 = cos thet2;
s2 = sin thet2;
*****************************************
```

```
* visualisation de l ellipse 1 en 20 elements
lc1 lm1 sm1 = vielli ce1 ax1 by1 thet1;
************************************
* visualisation de l ellipse 2
lc2 lm2 sm2 = vielli ce2 ax2 by2 thet2;
mess '      caracteristiques de l ellipse 1 ' ;
mess 'centre ' xe1 ye1 ;
mess 'demi axe x ' ax1 ;
mess 'demi axe y ' by1 ;
mess 'orientation par rapport au repere ' thet1 'deg';
mess 'excentricite ' e1 ;
mess 'distance entre les foyers ' de1 ;
mess 'aire reelle' a1 'aire discretisee' sm1;
mess 'perimetre reel' l1 'perimetre discretise' lm1;
mess '      caracteristiques de l ellipse 2 ' ;
mess 'centre ' xe2 ye2 ;
mess 'demi axe x' ax2 ;
mess 'demi axe y' by2 ;
mess 'orientation par rapport au repere ' thet2 'deg';
mess 'excentricite ' e2 ;
mess 'distance entre les foyers ' de2 ;
mess 'aire reelle' a2  'aire discretisee' sm2;
mess 'perimetre reel' l2 'perimetre discretise' lm2;
* on met l ellipse 1 dans le repere de l ellipse 2
oubl a1;oubli l1;oubl de1;oubl e1;oubl lm1; oubl sm1;
oubl a2;oubli l2;oubl de2;oubl e2; oubl lm2; oubl sm2;
mess ' entrez une abscisse et une ordonnee';
mess ' initiale dans votre repere';
mess ' en vous aidant du trace suivant';
q1 = p1 plus (0 0);
q2 = p2 plus (0 0);
cn1 = ce1 plus (0 0);
cn2 = ce2 plus (0 0);
lct = lc1 et lc2 et q1 et q2 et cn1 et cn2;
modi lct ;
oubl lct;
te1mte2 = thet2 - thet1;
c1m2 = cos te1mte2;
s1m2 = sin te1mte2;
oubl te1mte2;
* on cherche l intersection dans le repere de l ellipse 2
*equation de l ellipse 1 dans le repere de l ellipse 2
*x12/ax12 + y12/by12 = 1
*x1 = x2*c1m2 - y2*s1m2 + xe2me1*c1 + ye2me1*s1
*y1 = x2*s1m2 + y2*c1m2 - xe2me1*s1 + ye2me1*c1
* equation de l'ellipse 2
*x22/ax22 + y22/by22 = 1
obte x0*flottant y0*flottant;
```

```
* changement de repere pour aller dans le repere de l ellipse
*x0 = coor1 p0;
*y0 = coor2 p0;
x0p = x0 - xe2;
y0p = y0 - ye2;
x0 = x0p * c2;
x0 = x0 + (y0p * s2);
y0 = y0p * c2;
y0 = y0 - (x0p * s2);
oubl x0p;
oubl y0p;
sig = sign flot y0 ;
*boucle newton
alp = (xe2me1*c1) + (ye2me1*s1);
bet = (ye2me1*c1) - (xe2me1*s1) ;
repe bo;
*ruc = xr1 - ((x0 + xemxc) * (x0 + xemxc));
*si (ruc < 0.) ;
* mess ' le point n appartient pas au cercle';
* quit cercelli;
*fins;
  rac = y0;
*  rac = rac ** .5;
  si (rac ega 0.); rac = 1e-10; fins;
  racp = x0 / ax22 / rac ;
  racp = 0. - racp;
*  rac = by2 * rac ;
  racp = by22 * racp ;
  x1 = (x0*c1m2) - (rac*s1m2) + alp;
  y1 = (x0*s1m2) + (rac*c1m2) + bet;
  x12 = x1 * x1;
  y12 = y1 * y1;
  ff = (x12 / ax12 ) + (y12 / by12) - 1.;
  fp = (x1 * (c1m2 - (s1m2 * racp)) / ax12) +
       (y1 * (s1m2 + (c1m2 * racp)) / by12) ;
  dx = ff / fp / 2.;
  si ( (abs dx) < 1e-12);
    quit bo;
  fins;
  x0 = x0 - dx ;
  y0 = 1. - (x0 * x0 / ax22 );
  si ((abs y0) < 1e-5); y0 = 0.;fins;
  si (y0 < 0.) ;
   mess ' le point n appartient pas a l ellipse 2';
   mess ' donc il n y a pas d intersection possible';
   quit ellielli;
  fins;
  y0 = y0 ** .5;
```

```
  y0 = by2 * y0 * sig ;
oubl ff; oubl fp;oubl racp;
fin bo;
* fin de boucle newton
y01 = rac;
* changement de repere pour ellipse 2 vers cartesien
xx1 = (x0 * c2) - (y01 * s2);
yy1 = (y01 * c2) + (x0 * s2);
* om = oe + em
x1 = xx1 + xe2;
y1 = yy1 + ye2;
pp1 = x1 y1;
oubl x1; oubl y1;
oubl xe2me1; oubl ye2me1;
oubl xx1; oubl yy1;
oubl xe1; oubl ye1;
oubl xe2; oubl ye2;
oubl x0; oubl y01;
oubl s2; oubl c2;
oubl s1; oubl c1;
oubl alp; oubl bet;
oubl rac; oubl racp;
oubl c1m2; oubl s1m2;
oubl dx; oubl ff; oubl fp;
oubl x12; oubl y12; oubl ax12; oubl by12;
oubl x22; oubl y22; oubl ax22; oubl by22;
finp pp1 ;
**********
debp vielli ce*point ax*flottant by*flottant thet*flottant;
by2 = by * by;
ax2 = ax * ax;
xmi1 = 0. - ax;
xma1 = ax ;
dx1 = xma1 - xmi1;
n = 80 ;
dx1 = dx1 / (flot n);
* calcul de y
x = xmi1;
y = by2 * ( 1. - (x * x / ax2));
y = y ** .5;
pe0 = x y;
i = 0;
repe b1 n;
  i = i + 1;
  x = x + dx1;
  y = by2 * ( 1. - (x * x / ax2));
  si ((abs y) < 1e-5 );
    y = 0.;
```

```
  sino ;
    y = y ** .5;
  fins;
  si (i ega 1);
    lc1 = pe0 d 1 (x y);
  sino ;
    lc1 = lc1 d 1 (x y);
  fins;
fin b1;
lc1 = lc1 plus ce;
lc2 = lc1 tour 180. ce;
lc1 = lc1 et lc2;
lc1 = lc1 tour (0. + thet) ce;
elim lc1 .00001;
l = mesu lc1;
s = mesu lc1 surf;
oubl by12; oubl ax12; oubl xmi1; oubl xma1; oubl dx1;
oubl x; oubl y; oubl pe0;
finp lc1 l s;
```

b) Utilisation

```
opti dime 2 elem seg2;
ce1 = 218.72 -132.68 ;
p1 =  111.02 -72.067;
e = 0.825;
ax1 = 120.3959;
by1 = 135.6059;
p1 = (coor 1 ce1) ((coor 2 ce1) + by1);
ce2 =  0 -10;
by2 = 138.41 + e;
p3 = 0 (by2 + (coor 2 ce2) );
p13 = 123.2 -10;
p2 = p13 plus (e 0);
ax2 = 123.2 + e;
p2 = p2 / 100.;
p3 = p3 / 100.;
p1 = p1 / 100.;
ce1 = ce1 / 100.;
ce2 = ce2 / 100.;
ax1 = ax1 / 100.;
ax2 = ax2 / 100.;
by2 = by2 / 100.;
by1 = by1 / 100.;
q1 = ELLIELLI ce1 ax1 by1 p1 ce2 ax2 by2 p2;
l1 = ELLI 10 p1 q1 ce1 ax1 by1;
opti donn 5;
q1 = ELLIELLI ce2 ax2 by2 p3 ce1 ax1 by1 p1;
l2 = ELLI 10 p2 q1 ce2 ax2 by2;
```

l3 = **ELLI** 10 p3 q1 ce2 ax2 by2;

c) Opérateurs utilisés

+, -, *, /, **, <, >, ABSolu, ATGente, COORdonnée, COSinus, DEBProcédure, DROIt, EGAl, ELIMiner, ET, FIN, FINProcédure, FINSi, FLOTtant, MESSage, MESUre, MODIfier, OBTEnir, OUBLier, PLUS, QUITter, REPEter, SI, SIGNe, SINus, SINOn, TABLe, TOURner, TYPE.

On utilise les procédures PERELI pour calculer le périmètre vrai de l'ellipse, POSI pour calculer l'angle que fait le grand axe de l'ellipse avec l'axe des X et VIELLI pour construire les deux ellipses.

5.12 Intersection d'une droite et d'une ellipse

a) Construction

```
debp droielli p1*point p2*point
pc*point ax*flottant by*flottant q1*point ;
si (ax ega by1);
  thet = 0.;
sino ;
  tht1 = posi q1 ce1 ax by;
  thet = tht1;
si (tht1 > 90.);
  thet = tht1 - 90.;
  tr = ax ;
  ax = by ;
  by = tr ;
 dd
fins ;
fins ;
ax2 = ax * ax;
by2 = by * by;
x1 = coor 1 p1;
y1 = coor 2 p1;
x2 = coor 1 p2;
y2 = coor 2 p2;
ad = x2 - x1;
co = cos thet;
s1 = sin thet;
xc = coor 1 pc;
yc = coor 2 pc;
* on cherche l intersection dans le repere de l ellipse
si ( ad ega 0);
*equation de la droite dans le repere de l ellipse
*ysi = xco - x1+xc
   si (s1 ega 0.) ;
     s1 = 1e-20;
```

```
   fins ;
   av = co / s1;
   bv = (xc - x1) / s1;
sino;
   a = (y2 - y1 ) / ad;
   b = y2 - (a * x2);
*equation de la droite dans le repere de l ellipse
*y(co+asi) = x(aco-si) + b+axc - yc
   asi = a * s1;
   aco = a * co;
   axc = a * xc;
   ad = co + asi;
   si (ad ega 0.) ;
     ad = 1e-20;
   fins ;
   av = (aco - s1) / ad;
   bv = (b + (a * xc) - yc) / (co + asi);
fins ;
* equation de l'ellipse
*x2/ax2 + y2/by2 = 1
* abscisse de l intersection
* x2/ax2 + (avx+bv)2/by2 = 1
* trinome
aa = by2 + (ax2 * av * av);
bb = 2. * ax2 * av * bv;
cc = ax2 * bv * bv - (ax2 * by2);
xx1 xx2 = second aa bb cc;
yy1 = (av * xx1 ) + bv;
yy2 = (av * xx2 ) + bv;
* changement de repere
xx1 = xx1 - xc;
yy1 = yy1 - yc;
x1 = (xx1 * co) - (yy1 * s1);
y1 = (yy1 * co) + (xx1 * s1);
pp1 = x1 y1;
xx2 = xx2 - xc;
yy2 = yy2 - yc;
x2 = (xx2 * co) - (yy2 * s1);
y2 = (yy2 * co) + (xx2 * s1);
pp2 = x2 y2;
finp pp1 pp2 ;
```

b) Utilisation

```
opti dime 2;
* ellipse
pc =0 0 ;
thet = 45;
```

```
ax = 1;
by = .5;
p1 = 0 0;
p2 = -1 1 ;
pp1 pp2 = DROIELLI p1 p2 pc ax by ((cos thet) (sin thet)) ;
```

c) Opérateurs utilisés

On utilise les procédures POSI (voir page 27) et SECOND (voir page 22 mais on aurait pu utiliser l'opérateur DEG3)

5.13 Visualisation de la normale à une ligne

a) Construction

```
DEBP PATIN L12*MAILLAGE ;
*****************************************
*
*       DESSIN DE LA NORMALE A UNE LIGNE 2D
*         S UTILISE COMME UNE DIRECTIVE
*             PATIN (NOM DE LA LIGNE) ;
*
*
*********************************************
*   LA LIGNE EST ROUGE, L ORIENTATION EST BLANCHE
*
**MESS ' QUEL EST LE NOM DE LA LIGNE  ? ';
**OBTE L12*MAILLAGE ;
L12= L12 COUL ROUG ;
TOTO = L12;
NBL=NBEL L12;
IEL = 0;
*   BOUCLE SUR LES ELEMENTS DE LA LIGNE
*
REPE BLOEL NBL;
   IEL = IEL + 1;
   ELIEL = L12 ELEM IEL;
    PI1 = ELIEL POIN  1;
    PI2= ELIEL POIN 2;
    VEC= PI2 MOIN PI1;
    ALPHA = 0. - ( COOR 2 VEC);
    BETA = COOR 1 VEC;
    ALPHA = ALPHA / 4.;
    BETA = BETA / 4.;
    VEC1= ALPHA BETA;
    Q1=PI1 PLUS VEC1;
    Q2 = PI2 PLUS VEC1;
    LOR1= PI1 D 1 Q1;
    LOR2= PI2 D 1 Q2;
    TOTO = TOTO ET LOR1 ET LOR2;
```

```
    OUBL LOR1;OUBL LOR2;
    OUBL PI1;OUBL PI2;
    OUBL ELIEL;OUBL Q1;OUBL Q2;
FIN BLOEL;
TEX = ' ORIENTATION DE LA LIGNE  ' ;
TITR TEX;
TRAC TOTO ;
FINP;
```

b) Utilisation

```
PATIN l1 ;
        l1      MAILLAGE (de ligne)
```

c) Opérateurs utilisés

-, +, /, COORdonnées, COULeur, DEBProcédure, DROIt, ELEMent, ET, FIN, FINProcédure, MOINs, NBELément, OUBLier, PLUS, POINt, REPEter, TITRe, TRAcer

5.14 Visualisation de la normale à une surface

a) Construction

b) Utilisation

c) Opérateurs utilisés

5.15 Maillage d'un arc d'ellipse

Discrétisation d'un arc d'ellipse dont les axes sont parallèles aux axes cartésiens. Pour une ellipse quelconque, on peut s'inspirer de l'intersection de deux ellipses (voir pages 27 et suivantes).

a) Construction

```
DEBP ELLI NDEC*ENTIER POINT1*POINT POINT2*POINT CENTRE*POINT
      AX*FLOTTANT BY*FLOTTANT ;
*CALCUL DE L ANGLE DE ROTATION
*LE VECTEUR DE TRANSLATION EST CENTRE
* REPERE LOCAL DE L ELLIPSE ON CONNAIT L EQUATION
NDE = 0 - NDEC;
XC = COOR 1 CENTRE;
YC = COOR 2 CENTRE;
X1 = COOR 1 POINT1;
Y1 = COOR 2 POINT1;
XX = X1 - XC;
YY = Y1 - YC;
D1 = COOR 3 POINT1;
D2 = COOR 3 POINT2;
XX2 = XX * XX;
```

```
YY2 = YY * YY;
AX2 =AX * AX;
BY2 = BY * BY;
P = (XX2 * BY2) + ( AX2 * YY2) - (YY2 * BY2) - (XX2 * AX2);
P = P / 2. ;
Q = XX * YY * (BY2 - AX2);
R = ( XX2 * BY2) + (YY2 * AX2) + (YY2 * BY2) + (XX2 * AX2);
R = R / 2.;
R = (AX2 * BY2) - R;
IND = 0;
SI ( (ABS Q) < 1E-5) ;
  IND = 1;
FINS;
DELTA = (Q * Q) + (P * P) - (R * R);
SI ( IND EGA 1) ;
  T11 = 0.;
  T12 = 0.;
SINO;
  DELTA = DELTA ** .5;
  SI ( (ABS (R + P ) ) < 1E-5 );
    T11 = (R - P ) / (2 * Q);
    T12 = (R - P ) / (2 * Q);
  SINO;
    T11= (Q + DELTA) / (R + P);
    T12= (Q - DELTA) / (R + P);
  FINS;
FINS;
X2 = COOR 1 POINT2;
Y2 = COOR 2 POINT2;
XX = X2 - XC;
YY = Y2 - YC;
XX2 = XX * XX;
YY2 = YY * YY;
AX2 =AX * AX;
BY2 = BY * BY;
P = (XX2 * BY2) + ( AX2 * YY2) - (YY2 * BY2) - (XX2 * AX2);
P = P / 2. ;
Q = XX * YY * (BY2 - AX2);
R = ( XX2 * BY2) + (YY2 * AX2) + (YY2 * BY2) + (XX2 * AX2);
R = R / 2.;
R = (AX2 * BY2) - R;
IND = 0;
SI ( (ABS Q) < 1E-5) ;
  IND = 1;
FINS;
DELTA = (Q * Q) + (P * P) - (R * R);
SI ( IND EGA 1) ;
  T21 = 0.;
```

```
  T22 = 0.;
  SINO;
  DELTA = DELTA ** .5;
  SI ( ( ABS (R + P ) ) < 1E-5 );
    T21 = (R - P ) / (2 * Q);
    T22 = (R - P ) / (2 * Q);
  SINO;
    T21= (Q + DELTA) / (R + P);
    T22= (Q - DELTA) / (R + P);
  FINS;
FINS;
oubl delta; oubl q; oubl r; oubl p;
SI ( ( ABS (T11 - T21 ) ) < 1E-2 );
  T1 = T11;
FINS;
SI ( ( ABS (T11 - T22 ) ) < 1E-2 );
  T1 = T11;
FINS;
SI ( ( ABS (T12 - T21 ) ) < 1E-2 );
  T1 = T12;
FINS;
SI ( ( ABS (T12 - T22 ) ) < 1E-2 );
  T1 = T12;
FINS;
T1 = ATG T1 1.;
O1 = 0 BY;
RAP = BY / AX;
O2 = RAP BY;
* IL FAUT PASSER EN MAILLAGE POUR AFFINITE
* PUIS REPASSER EN POINT POUR CERCLE
PO1 = LOC POINT1 CENTRE T1;
PO2 = LOC POINT2 CENTRE T1;
POO1 = MANU POI1 PO1;
PC1= POO1 AFFI RAP O1 O2;
PCC1 = PC1 POIN 1;
POO2 = MANU POI1 PO2;
PC2= POO2 AFFI RAP O1 O2;
PCC2 = PC2 POIN 1;
DI = D1 / RAP;
DF = D2 * RAP;
SI ( NDEC EGA 0);
  LC = PCC1 CERC (0. 0.) PCC2 DINI DI DFIN DF;
SINO;
  LC = PCC1 CERC NDEC (0. 0.) PCC2 DINI DI DFIN DF;
FINS;
RAP = 1. / RAP;
LE = LC AFFI RAP O1 O2;
LE = LE PLUS CENTRE;
```

```
LE = LE TOUR T1 CENTRE;
FINP LE ;
DEBP LOC POINT1*POINT CENTRE*POINT ANGLE*FLOTTANT;
X1 = COOR 1 POINT1;
Y1 = COOR 2 POINT1;
XC = COOR 1 CENTRE;
YC = COOR 2 CENTRE;
XX = X1 - XC;
YY = Y1 - YC;
XP = (XX * (COS ANGLE)) + (YY * (SIN ANGLE));
YP = (YY * (COS ANGLE)) - (XX * (SIN ANGLE));
POINP = XP YP;
oubl xx; oubl yy; oubl xc; oubl yc; oubl x1; oubl y1;
FINP POINP ;
```

b) Utilisation

l1 = **ELLI** n p1 p2 o1 ax by ;

n	ENTIER - nombre d'éléments (positif, négatif en tenant compte des densités, 0 en respectant les densités)
p1, p2	POINT - origine et extrémité
o1	POINT - centre de l'ellipse
ax, by	FLOTTANT - axe x, axe y

c) Opérateurs utilisés

+, -, *, /, <, ABSolu, AFFInité, ATGente, CERCle, COORdonnée, COSinus, DEBProcédure, EGAl, FINProcédure, FINSi, MANUel, OUBLier, POINt, SI, SINus, SINOn

On utilise la procédure LOC pour déterminer les angles correspondant aux extrémités de l'arc.

5.16 Zoom imposé

Tracé d'un MAILLAGE dans une fenêtre de zoom. L'avantage par rapport à l'option zoom de TRACer (voir volume MAILLAGE) est que la fenêtre est toujours précisément la même.

a) Construction

```
debp zoom m1*maillage ;
mess 'entrez le premier point ';
obte x1*flottant y1*flottant;
mess 'entrez le deuxieme point ';
obte x2*flottant y2*flottant;
xma = x1;
xmi = x2;
si (x2 > x1);
        xma = x2;
        xmi = x1;
fins;
yma = y1;
ymi = y2;
```

```
si (y2 > y1);
        yma = y2;
        ymi = y1;
fins;
p1 = xmi ymi;
p2 =  xma ymi;
p3 = xma yma;
p4 = xmi yma;
ll = p1 droi 1 p2 droi 1 p3 droi 1 p4 droi 1;
ma1 = m1 incl ll;
oubl ll ;oubl p1 ;oubl p2 ; oubl p3 ;oubl p4 ;
oubl y1 ;oubl y2 ;oubl yma ; oubl ymi ;
oubl x1 ;oubl x2 ;oubl xma ; oubl xmi ;
finp ma1;
```

b) Utilisation

```
ZOOM mail1 ;
        mail1   MAILLAGE (de surface)
        puis on répond aux questions
```

c) Opérateurs utilisés

>, DEBProcédure, DROIte, FINProcédure, FINSi, INCLus, MESSage, OBTEnir, OUBLier, SI, SINOn

3.17 Bornes d'isovaleurs

Découpage d'un intervalle d'isovaleurs en 22 zones de couleur.

a) Construction

```
debp borne min*flottant max*flottant;
si (max < min) ;
        tr = min ;
        min = max ;
        max = tr ;
fins ;
mm = max - min;
m1 = min;
icoul = 22;
evd = prog m1;
mm = mm / icoul;
i = 2;
repe bo icoul ;
  m1 = m1 + mm;
  evd = inse i evd m1;
i = i + 1;
fin bo;
finp evd;
```

b) Utilisation

evd = **BORNE** f1 f2 ;
f1, f2 FLOTTANT
evd EVOLUTIO
On utilisera evd dans la directive TRACer

c) Opérateurs utilisés

>, /, +, -, DEBProcédure, FIN, FINProcédure, FINSi, INSErer, PROGression, REPEter, SI

3.18 Projection d'un point sur une droite

Projection orthogonale d'un point donné sur une droite définie par deux points.

a) Construction

debp projd p1*point p2*point p3*point ;
v1 = p2 moin p3 ;
aa = **coor** 1 v1 ;
bb = **coor** 2 v1 ;
v2 = bb (0**.-** aa) ;
q1 = p1 **proj** v2 **droi** p2 p3 ;
finp q1 ;

b) Utilisation

q1 = **PROJD** p1 p2 p3 ;
p1 POINT (à projeter)
q1 POINT (projeté)
p2, p3 POINT (définissant la droite)

c) Opérateurs utilisés

-, COORdonnées, DEBProcédure, DROIt, FINProcédure, MOINs, PROJection

6. TYPE D'OBJETS CREES

Ils sont définis par des mots de huit lettres au maximum. Le type d'un objet peut être retrouver par l'opérateur **TYPE**.

motype = **TYPE** objet ;

ENTIER
Créé par **:**
Utilisé par **:**

FLOTTANT
Créé par **:**
Utilisé par **:**

LISTENTI
Créé par **:** **LECT**
Utilisé par **:**

LISTMOTS
Créé par **:** **MOTS**
Utilisé par **:**

LISTREEL
Créé par **:** **PROG**
Utilisé par **:**

LOGIQUE
Créé par **:**
Utilisé par **:**

MAILLAGE
Créé par **:** **MANU**
Utilisé par **:**

MOT
Créé par **:** **MOT**, **TYPE, VALE**
Utilisé par **:**

POINT
Créé par **:**
Utilisé par **:**

TABLE
Créé par **:** **TABL**
Utilisé par **:**

TEXTE
Créé par **:**
Utilisé par **:**

7. ESSAI DE RECENSEMENT DES VALEURS PAR DEFAUT

Pour chacun des opérateurs, on fournit, quand elles existent , les valeurs par défaut prises par CASTEM2000®.

OPTIon

8. REFERENCES GENERALES

9. ANNEXE THEORIQUE

9.1 METHODE DE

10. REPERES BIOGRAPHIQUES

NEWTON Isaac

11. INDEX

N

O

P

Q

R

S

T

U

V