

# A la découverte du logiciel Maple

Ces quelques pages ont pour but de vous présenter le logiciel MAPLE, ses possibilités et son fonctionnement ; elles n'ont cependant pas la prétention d'être exhaustives. Cette première approche sera complétée au cours de l'année.

## Table des matières

<b>1</b>	<b>L'outil informatique</b>	<b>1</b>
1.1	L'ordinateur . . . . .	1
1.2	Le calcul formel . . . . .	2
1.3	Algorithmes et programmation . . . . .	2
<b>2</b>	<b>Le logiciel Maple</b>	<b>3</b>
2.1	Présentation . . . . .	3
2.1.1	Ligne de commande . . . . .	3
2.1.2	L'aide . . . . .	3
2.1.3	Les packages . . . . .	3
2.1.4	Opérations usuelles . . . . .	3
2.1.5	Variables, affectation et égalité . . . . .	4
2.1.6	Résoudre une équation . . . . .	4
2.1.7	Constantes et fonctions usuelles . . . . .	5
2.1.8	Le calcul différentiel . . . . .	5
2.2	Les objets Maple . . . . .	5
2.2.1	Les types . . . . .	5
2.2.2	Représentation arborescente . . . . .	6

## 1 L'outil informatique

### 1.1 L'ordinateur

L'objet des quelques lignes qui suivent est de décrire très sommairement le principe de fonctionnement d'un *ordinateur*.

Un ordinateur est un ensemble de circuits électroniques permettant de manipuler des données sous forme d'impulsions électriques. Cette machine permet de traiter automatiquement les données, ou informations, selon des séquences d'instructions prédéfinies appelées aussi programmes.

Des dispositifs d'entrée (exemple : le clavier) et de sortie (exemple : l'écran) permettent à l'ordinateur de communiquer avec un opérateur extérieur. L'information récupérée est convertie en données compréhensibles par l'ordinateur et à l'inverse, les périphériques de sortie décodent l'information fournie par l'ordinateur afin de la rendre compréhensible par l'utilisateur.

Les instructions que l'ordinateur peut comprendre ne sont pas celles du langage humain. Il sait juste exécuter un nombre limité d'instructions bien définies dans un langage (code) qui lui est propre

(exemple : le langage ASCII). Ces instructions et données qui sont comprises par l'ordinateur sont constituées uniquement de 0 et de 1 (0 = Le courant électrique ne passe pas. 1 = Le courant électrique passe). Le langage ASCII possède 256 caractères comprenant les minuscules, les majuscules, les chiffres et divers symboles, et est écrit sur 8 bits autrement dit un octet.

En général, on n'utilise pas ce type de langage ; on passe par un langage de haut niveau qui est ensuite transformé en langage binaire par un programme adapté (interpréteur ou compilateur). Les programmes ainsi obtenus sont des programmes compilés compréhensibles par l'ordinateur.

Exemples de langages de programmation : **C**, **C++**, **JAVA**...

Maple peut être considéré comme un langage de "très haut niveau" : une instruction Maple correspondra, après interprétation, à une longue séquence d'instructions machines.

## 1.2 Le calcul formel

L'accroissement des besoins en calculs longs et complexes (en aéronautique, aérospatiale, météorologie...) a conduit au développement du *calcul scientifique* ainsi qu'au développement de logiciels permettant de substituer pour ces calculs l'ordinateur à l'homme. Même si l'histoire des machines à calculer est ancienne, les possibilités de calculs offertes par les machines se sont essentiellement développées ces 50 dernières années.

Un système de calcul formel est un logiciel qui permet de faire du calcul en utilisant des représentations symboliques d'objets mathématiques tels que les nombres entiers, nombres rationnels, polynômes ou divers types de fonctions. "Calculer" signifie ici transformer et simplifier des expressions mathématiques en suivant les règles mathématiques satisfaites par ces objets (il peut s'agir d'additionner, de multiplier mais aussi d'intégrer, de dériver ou bien encore d'évaluer numériquement des expressions avec une la précision donnée.)

Il s'agit de reproduire des calculs que l'on pourrait effectuer à la main tout en bénéficiant de la puissance et la rapidité de l'ordinateur : on peut espérer un gain de temps, de fiabilité, et ainsi aborder des problèmes intraitables à la main.

## 1.3 Algorithmes et programmation

Un *algorithme* est un processus décrivant une suite d'opérations à effectuer pour obtenir la réponse à un problème donné.

Il suffit donc de donner les *variables* d'entrée, d'appliquer les opérations décrites dans l'algorithme et on récupère en sortie la solution du problème.

L'exemple le plus célèbre est l'algorithme d'Euclide qui permet de calculer le PGCD de deux entiers non nuls.

**Exemple :** l'algorithme d'Euclide.

$a$  et  $b$  sont deux entiers naturels tels que  $a > b > 0$ .

Etape 1. Calculer le reste  $r$  de la division euclidienne de  $a$  par  $b$ .

Etape 2. Si  $r = 0$  alors l'algorithme se termine : le PGCD de  $a$  et  $b$  est  $b$ .

Etape 3. Sinon,  $a$  prend la valeur de  $b$  et  $b$  prend la valeur de  $r$  et on recommence à l'étape 1.

La principe même d'algorithme (succession finie d'opérations) est bien adapté au fonctionnement d'un ordinateur ; le problème est donc de traduire un algorithme dans un langage compréhensible par la machine : on dit alors qu'on fait de la *programmation*.

On peut finalement dire qu'un logiciel de calcul formel permet d'effectuer toutes les opérations pour lesquelles on dispose d'un algorithme.

## 2 Le logiciel Maple

### 2.1 Présentation

Le logiciel Maple est un logiciel de calcul formel. Il permet également de réaliser des graphiques en 2D ou 3D.

#### 2.1.1 Ligne de commande

L'utilisateur rédige une série de commandes et l'exécute directement. Toutes les commandes exécutables se situent dans une ligne de texte qui est précédée du symbole :

>

On saisie dans cette ligne l'opération que Maple doit effectuer et on rajoute le symbole ";" si on souhaite que le résultat s'affiche ou le symbole ":" sinon. On appuie enfin sur la touche ENTER et la commande s'exécute.

Le raccourci "Ctrl + k" permet d'ajouter une ligne au dessus et "Ctrl + j" une ligne au-dessous.

En résumé :

[> <i>instruction</i> ;	ligne de commande avec affichage du résultat
[> <i>instruction</i> :	ligne de commande sans affichage du résultat
%	dernier résultat calculé; %% l'avant dernier et %%% celui encore avant
print( <i>expression</i> )	affichage forcé de <i>expression</i>

#### 2.1.2 L'aide

Nous découvrirons au cours de l'année toutes les possibilités offertes par ce logiciel au cours des séances de TD et TP. Il est cependant conseillé de se référer à l'aide disponible dans le logiciel pour connaître la syntaxe exacte des fonctions utilisées. Pour cela, on peut effectuer les opérations suivantes :

**?nom\_de\_commande** : la page d'aide associée s'ouvre.

**F1** : placer le curseur sur le mot dont vous voulez plus d'informations et appuyer sur la touche F1.

**Table de matière dynamique** : les fonctions y sont classées par thèmes et sous-thèmes.

**Help/Topic Search** : recherche l'ensemble des commandes qui commencent par les lettres que l'on tape au fur et à mesure.

#### 2.1.3 Les packages

Certaines fonctions de MAPLE ne sont pas accessibles immédiatement : on y fait appel en chargeant le package qui les contient via la commande `with(nom du package)` ;. Exemple : pour tracer une animation de courbes, on utilise la fonction `animate`. Avant d'utiliser cette fonction, on doit exécuter la commande `with(plots)`.

#### 2.1.4 Opérations usuelles

+, -, /, ^	opérations standards
*	multiplication : pas de raccourci d'écriture!
sqrt	racine carrée
@	composition de fonctions
@@	puissance de composition

La puissance s'écrit également \*\* :

> 3\*\*2;

La fonction `evalf` donne un arrondi du nombre considéré :

```
> evalf(ln(2));
```

0.6931471806

La commande `Digits` permet de définir le nombre de chiffres après la virgule que Maple doit considérer.

Le logiciel MAPLE offre des possibilités de calcul numérique et littéral :

– `expand` : développe une expression.

– `factor` : factorise une expression.

– `simplify` : réduit une expression.

```
> expand((a+b)^3);
```

$$a^3 + 3a^2b + 3ab^2 + b^3$$

```
> factor(3*x + 3*x^2);
```

$$3x(1 + x)$$

### 2.1.5 Variables, affectation et égalité

Pour donner une valeur à une variable on utilise la commande `:=`. Exemple :

```
> x:=4+a;
```

$$x := 4 + a$$

```
> x;
```

$$4 + a$$

La commande `x :=4+a` change la valeur de `x`.

Il ne faut pas confondre l'opération d'affectation avec l'égalité `x=4+a` qui peut s'évaluer à l'aide de `true` ou `false`.

```
> is(x=a+4);
```

*true*

```
> is(x=1);
```

*false*

**IMPORTANT** : La commande `restart` permet d'annuler toutes les affectations antérieures.

### 2.1.6 Résoudre une équation

La fonction `solve` permet de résoudre des équations :

```
> eq := x^2 + a*x + 1 = 0;
```

```
> solve(eq,x);
```

$$-\frac{a}{2} + \frac{\sqrt{a^2 - 4}}{2}, -\frac{a}{2} - \frac{\sqrt{a^2 - 4}}{2}$$

La commande `dsolve` sert à la résolution d'équations différentielles.

### 2.1.7 Constantes et fonctions usuelles

MAPLE connaît les constantes et fonctions usuelles :  $\pi$  (noté `Pi`),  $i$  (noté `I`), `ln`, `sin`...). Exemples :

> `I^2+1;`

0

> `exp(I*Pi);`

-1

> `cos(2*Pi);`

1

### 2.1.8 Le calcul différentiel

Pour définir une fonction  $f$  de variable  $x$ , on utilise la syntaxe suivante : `f := x -> f(x)` ;. On dispose des commandes suivantes :

- `diff` : dériver
- `int` : intégrer formellement, déterminer une primitive.
- `Int` : intégrer numériquement.
- `plot` : tracer.
- `limit` : calculer une limite.

Exemples :

> `c:=x-> x^2 + 1/x ;`

$$c := x \rightarrow x^2 + \frac{1}{x}$$

> `diff(c(x),x);`

$$2x - \frac{1}{x^2}$$

## 2.2 Les objets Maple

### 2.2.1 Les types

MAPLE associe automatiquement à toute expression un *type*. Donnons-en une liste non exhaustive :

- `integer` : entiers relatifs.
- `rational` : rationnels
- `float` : nombre à virgule flottante.
- `list` : liste
- `set` : ensemble

Il faut apprendre à distinguer les différents types. La commande `whattype` renvoie le type de la variable.

> `whattype(1);`

*integer*

> `whattype(3.1415);`

*float*

```
> X:=a+b;
```

$$X := a + b$$

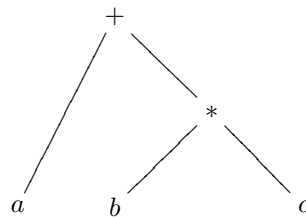
```
> whattype(X);
```

+

### 2.2.2 Représentation arborescente

Maple représente les expressions mathématiques sous forme d'*arbres*. Les noeuds de cet arbre sont des opérateurs (comme +, \*, cos) et les feuilles sont des expressions simples.

Exemple : représentation arborescente de l'expression :



**Exercice 1.** Déterminer la représentation arborescente de l'expression suivante :

```
> cos(a^2 + b*c);
```

La fonction `op` permet de "naviguer" dans la représentation arborescente des expressions.

```
> L:=[a,b,c+d];
```

$$L := [a, b, c + d]$$

```
> op(L);
```

$$a, b, c + d$$