Programme

Chapitre I

I.1 C'est quoi Maple

Chapitre II : Analyse

II. 1 Calculs sur les entiers et les nombres réels
II. 2 Nombres Complexes
II. 3 Fonctions Numériques
II. 4 Expressions symboliques

Chapitre III: Analyse

III. 1 Matrices
III. 2 Fonction du package Linalg
III. 3 Les Vecteurs

Chapitre IV: Résolution des Equations

IV. 1 Equations et systèmes d'équations
IV. 2 équations différentielles
IV. 3 Manipulation des solutions

Chapitre V: Le Graphisme

V. 1 Courbes planes
V. 2 Surfaces dans l'espace
V. 3 Les options du graphisme

Chapitre VI: Les procédures et la programmation sous Maple

I-1 C'est quoi MAPLE

Maple est un logiciel commercial qui propose un environnement de **calcul formel**. Le calcul formel, encore appelé **calcul symbolique** (*computer algebra*) donne les moyens de manipuler les nombres, d'effectuer des calculs ou de réaliser des représentations graphiques sophistiquées, mais aussi et surtout de mener des calculs algébriques, de représenter symboliquement des objets mathématiques complexes ou élaborés comme des fonctions, des équations et leurs solutions algébriques. Les logiciels de calcul formel, tels **Maple, Mathematica** ou **Mupad**, ont connu une évolution consistant à intégrer aussi en leur sein des capacités de calcul numérique. C'est le cas de Maple qui incorpore désormais des logiciels spécifiquement dédiés au calcul numérique, afin de proposer à l'utilisateur un environnement permettant de traiter toute la gamme des situations que l'on peut rencontrer lors de la résolution d'un problème.

I-2 La puissance du calcul formel (symbolique)

La première série d'exemples montre la capacité à faire des calculs symboliques (par opposition à numériques) que tout utilisateur ayant acquis un certain niveau en mathématiques est capable de mener à bien lui-même.
Commençons par un calcul classique de limite :

```
> Limit((sin(tan(x))-tan(sin(x)))/x^7,x=0)=limit
  ((sin(tan(x))-tan(sin(x)))/x^7,x=0);
```

$$\lim_{x \to 0} \frac{\sin(\tan(x)) - \tan(\sin(x))}{x^7} = \frac{-1}{30}$$

Poursuivons par un calcul de dérivée tout aussi éloquent :

```
> Diff(arcsin(ln(x^3)),x)=diff(arcsin(ln(x^3)),x);
```

$$\frac{d}{dx} \arcsin(\ln(x^3)) = \frac{3}{x\sqrt{1 - \ln(x^3)^2}}$$

La puissance aussi de Maple est de définir de manière précise les expressions mathématiques avec ses constantes mathématiques célèbres (π dans l'exemple suivant au lieu d'une valeur décimale)

```
> Int(arctan(x),x=0..1)=int(arctan(x),x=0..1);
```

$$\int_0^1 \arctan(x)\, dx = \frac{\pi}{4} - \frac{1}{2}\ln(2)$$

Considérons enfin un exemple issu de l'algèbre, où il s'agit de déterminer les racines cubiques de l'unité :

```
> S:=solve(x^3-1=0);
```

$$S := 1, -\frac{1}{2} + \frac{1}{2} I \sqrt{3}, -\frac{1}{2} - \frac{1}{2} I \sqrt{3}$$

On remarque que Maple propose d'emblée une représentation exacte des trois racines cubiques de l'unité sous forme de nombres complexes (l'unité imaginaire étant notée I). Intéressons-nous à la deuxième racine trouvée :

```
> u:=S[2];
```

$$u := -\frac{1}{2} + \frac{1}{2} I \sqrt{3}$$

Pour en demander une valeur numérique approchée :

```
> v:=evalf(u);
```

$$v := -0.5000000000 + 0.8660254040\, I$$

On peut désormais comparer le résultat symbolique au résultat numérique en élevant chacun d'eux au cube :

```
> w:=evalc(u^3);
```

$$w := 1$$

Premiers pas avec Maple

L'utilisateur doit donc saisir une instruction à côté de l'invite et pour signifier la fin de sa saisie, il lui faut terminer la ligne de commande par le caractère **";"** ou par le caractère **":"**. L'instruction saisie sera interprétée et exécutée (si la syntaxe est correcte) dès que l'utilisateur aura pressé la touche Entrée (ou Enter sur un clavier anglo-saxon).
Lorsqu'une instruction se finit par **";"**, l'interpréteur de Maple exécute l'instruction et, si elle est syntaxiquement valide15, affiche le résultat obtenu :

```
> 2+3;
```

$$5$$

```
> exp(1.0);
```

$$2.718281828$$

En revanche, si l'instruction est terminée par ":", Maple exécute l'instruction de façon muette, c'est-à-dire n'affiche pas le résultat :

```
> 2+3:
> exp(1.0):
```

Pour grouper plusieurs instruction, il suffit de les séparer par des **";"**

```
> 1+2;2.0^(1/2);(2*3)^2;
```

$$3$$

$$1.414213562$$

$$36$$

On peut aérer la présentation des instructions et passer à la ligne entre chacune d'elles plutôt que de les écrire sur la même ligne.(il faut presser simultanément **Shift/Enter**

```
> 1+2;
  2.0^(1/2);
  (2*3)^2;
```

$$3$$

$$1.414213562$$

$$36$$

Pour écrire une instruction dont la longueur est supérieure à celle d'une ligne sur la feuille de travail, on peut formater la saisie en plaçant un caractère "\" chaque fois que l'on va à la ligne. Maple utilise aussi ce caractère pour signaler des réponses qui tiennent sur plusieurs lignes :

```
> 1234567890\
  1234567890\
  1234567890\
  1234567890\
  1234567890\
  1234567890\
  1234567890\
  1234567890\
  1234567890+1;
```

$$12345678901234567890123456789012345678901234567890123456789012345678901234567891$$

1.4.4 Commentaires

Le caractère "#" permet d'introduire un commentaire dans une feuille de travail. Ce qui se trouve entre ce caractère et la fin de ligne n'est alors pas pris en compte par l'interpréteur, comme le montre l'exemple suivant :

```
> 1+2; # un premier calcul très simple
  # 2.0^(1/2); cette ligne n'est pas prise en compte
  (2*3)^2;
```

$$3$$

$$36$$

1.5.1 Noms de variables

Un nom de variable licite est un mot qui commence par une lettre suivie d'un nombre fini de caractères[18], lettres, chiffres ou "_" (*underscore*), autres que les caractères qui jouent un rôle particulier dans le langage ("%", "#", " ?", "&", ":", ";", "=", "$"...) de sorte que l'ensemble forme un mot n'apparaissant pas dans les mots réservés ou les mots protégés du langage Maple[19].

```
> abc; a1234; ABC; # sont des noms de variables licites
```

$$abc$$

$$a1234$$

$$ABC$$

sont des noms de variables licites, tandis que :

```
> 1abc; # n'est pas un nom de variable licite
Error, missing operator or ';'
```

est rejeté par l'interpréteur car le mot commence par un chiffre. De même, `cos` ne peut servir de nom de variable puisque ce mot est réservé par le langage pour désigner la fonction trigonométrique cosinus :

```
> cos:=25;
Error, attempting to assign to 'cos' which is protected
```

Affectation

```
> a:=10+2*4;
```

$$a := 18$$

```
> a;
```

$$18$$

```
> b:=x+y+2*x;
```

$$b := 3\,x + y$$

```
> b;
```

$$3\,x + y$$

Noms de variables fondés sur des lettres

```
> beta;Beta;delta;Delta;omega;Omega;
```

$$\beta$$
$$B$$
$$\delta$$
$$\Delta$$
$$\omega$$
$$\Omega$$

En ce qui concerne pi

```
> pi;Pi;PI;
```

$$\pi$$
$$\pi$$
$$\Pi$$

Voici ce qui se produit lorsqu'on essaie d'affecter une valeur à ces différents identificateurs :

```
> pi:=10;Pi:=10;PI:=10;
```

$$\pi := 10$$

```
Error, attempting to assign to `Pi` which is protected
```

$$\Pi := 10$$

La lettre gamma

gamma désigne la constante d'Euler
Gamma est une constante licite
GAMMA désigne la fonction gamma d'Euler

```
> gamma;Gamma;GAMMA;
```

$$\gamma$$
$$\Gamma$$
$$\Gamma$$

Donc pour l'affectation

```
> gamma:=1;Gamma:=1;GAMMA:=1;
```

```
Error, attempting to assign to `gamma` which is protected
```

$$\Gamma := 1$$

```
Error, attempting to assign to `GAMMA` which is protected
```

Une nouvelle instruction, il s'agit de **evalf** pour évaluer une fonction, variable…

```
> evalf(gamma);evalf(Gamma);evalf(GAMMA);
```

$$0.5772156649$$
$$1.$$
$$\Gamma$$

Nettoyage de l'espace de travail

```
> a:=1;b:=2;a;b;
```

$$a := 1$$
$$b := 2$$
$$1$$
$$2$$

```
> restart:a;b;
> a;b;
```

$$a$$
$$b$$

Désaffecter une variable

```
> a:=5;
```

$$a := 5$$

```
> a:='a';
```

$$a := a$$

```
> a;
```

$$a$$

ANALYSE

```
>evalf(sqrt(2),20);
```

1.4142135623730950488

On aurait donc pu aussi imposer 20 chiffres à Maple en changeant la valeur de la variable **Digits** comme le montre l'exemple suivant :

```
>Digits:=20;
```

Digits := 20

```
>evalf(sqrt(2));
```

1.4142135623730950488

****

```
>whattype(2);
```

integer

alors que :

```
>whattype(2.);
```

float

****

```
>convert(2.5,fraction);
```

$$\frac{5}{2}$$

```
>abs(-5);
```

5

```
>abs(6);
```

6

On peut aussi chercher le minimum ou le maximum de plusieurs entiers à l'aide des fonctions **min** et **max** :

```
>min(1,2,6,-3);
```

-3

```
>max(1,2,6,-3);
```

6

****

La fonction **floor** permet d'obtenir la partie entière, et la fonction **ceil** l'entier immédiatement supérieur :

```
>floor(nombre);
>ceil(nombre);
```

Par exemple pour la valeur approchée de $\sqrt{2}$:

```
>floor(1.414);
```

1

```
>ceil(1.414);
```

2

On peut aussi réaliser une troncature ou un arrondi à l'aide des fonctions **trunc** et **round** :

```
>trunc(3.2);
```

3

```
>round(3.2);
```

3

1.5 *Sommation*

Maple permet de réaliser des sommes à l'aide de la fonction **sum**.

Ainsi, pour sommer l'expression de l'indice i pour i variant de a à b :

```
>sum(expression(i),i=a..b);
```

Par exemple on a :

```
>sum(i^2,i=1..100);
```

338350

```
>sum(1/k^2,k=1..infinity);
```

$$\frac{\pi^2}{6}$$

Pour obtenir une forme inerte, on utilise la fonction **Sum**, ainsi, on obtient :

```
>Sum(1/k^2,k=1..infinity)=sum(1/k^2,k=1..infinity);
```

$$\sum_{k=1}^{\infty}\frac{1}{k^2}=\frac{\pi^2}{6}$$

Si une série ne converge pas, Maple renvoie alors l'infini. Par exemple :

```
>sum(1/k,k=1..infinity);
```

$$\infty$$

1.6 *Produit*

De même que l'on vient de réaliser des sommes, Maple permet également de réaliser des produits à l'aide de la fonction **product**.

> Ainsi, pour réaliser le produit des expressions de l'indice *i* pour *i* variant de *a* à *b* :
>
> ```
> >product(expression(i),i=a..b);
> ```

Par exemple on a :

```
>product(i^2,i=1..100);
```

$$13168189440000$$

Comme pour la fonction somme, on obtient la forme inerte en entrant la fonction avec une majuscule :

```
>Product(k,k=1..n);
```

$$\prod_{k=1}^{n}k$$