Premiers pas en Maple (Partie 2 de 5)

© Pierre Lantagne
Enseignant retraité du Collège de Maisonneuve

Pour voir le contenu des différentes sections, cliquer avec la souris sur le triangle ▶ précédant le titre. La section se déploiera ▼ et son contenu sera affiché.
(un clic gauche ou un clic droit sur le triangle permettra de rétracter la section et son contenu sera alors masqué).

Pour votre confort, vous pouvez ajuster la taille de l'affichage à l'aide de la commande *Facteur de zoom* du menu *Affichage*.

Bonne lecture à tous !

* Ce document Maple est exécutable avec la version 2020.2

Initialisation

```
> restart;
```

Objectif

Cette deuxième partie a comme objectif principal de présenter à l'étudiant l'importance en `Maple` de donner un nom aux différents objets pour ensuite mieux les opérer et pour mieux documenter la zone des résultats. L'élève sera donc amené à prendre connaissance des types en `Maple` pour mieux distinguer entre autre une fonction comme règle de correspondance d'avec la formule du calcul des images. Sur un autre plan, cette deuxième partie sera l'occasion d'aborder la résolution d'équations et d'inéquations. Finalement, il y aura un retour sur le mécanisme de la simplification de `Maple` pour illustrer que le domaine de simplification pour `Maple` est l'ensemble des nombres complexes.

Déroulement

Avant même d'exécuter les requêtes, l'élève doit faire d'abord une lecture attentive de la présentation des différentes transpositions en `Maple` des notions mathématiques. Pour profiter au maximum de la présentation des différents éléments fondamentaux présentés dans ce document, il ne faut surtout pas être un lecteur passif. Au contraire, au fil du déroulement de cette feuille `Maple`, l'élève devra, de plus en plus, anticiper le résultat qui sera affiché et de se convaincre, à chaque fois, que le résultat obtenu est le résultat attendu.

Une fonction en Maple

L'opérateur fonctionnelle flèche « -> » est à utiliser pour la création de fonctions. L'opérateur flèche est utilisé pour transposer en Maple la notion mathématique de règle de correspondance fonctionnelle.

```
> x->2*x-3;
```

$$x \mapsto 2 \cdot x - 3 \tag{4.1}$$

Une procédure est en quelque sorte la transposition informatique du concept mathématique de fonction.

Obtenons l'image du nombre 5 par la formule de calcul des images $2x-3$;

```
> (x->2*x-3)(5);
```

$$7 \tag{4.2}$$

Mais, comme en mathématique, il est commode de donner un nom à une fonction. Donnons-lui le nom f. Pour ce faire, assignons donc la fonction $x \rightarrow 2x-3$ à la variable f.

```
> f:=x->2*x-3;
```

$$f := x \mapsto 2 \cdot x - 3 \tag{4.3}$$

Ainsi, $f(x)$ désignera l'image de x par la fonction f.

```
> f(x);
```

$$2x-3 \tag{4.4}$$

```
> f(6);
```

$$9 \tag{4.5}$$

```
> f(t);
  f(x+h);
```

$$2t-3$$

$$2x+2h-3 \tag{4.6}$$

```
> f:='f':  # Rendons libre la variable f
```

C'est une très mauvaise habitude d'utiliser la syntaxe fonctionnelle pour donner un nom à une expression. La requête suivante est donc un exemple de requêtes à éviter.

```
> f(x):=2*x-3;
```

$$f(x) := 2x-3 \tag{4.7}$$

Voici pourquoi.

```
> f(6);
```

$$f(6) \tag{4.8}$$

```
> f(x):='f(x)';
```

$$f(x) := f(x) \tag{4.9}$$

Les types en Maple

Cette notion est fondamentale dans un logiciel de calcul formel. Bien que Maple puisse valider une expression donnée, ce n'est pas nécessairement celle que l'usager pense avoir créée.

a) Expression de type égalité (équation)

Maple est un logiciel de calcul formel. Il n'y a donc aucun problème à ce qu'il puisse reconnaître une égalité.

```
> F=m*a;
```

$$F = ma \tag{5.1}$$

```
> whattype(F=m*a);
```

$$`=` \tag{5.2}$$

On peut alors donner un nom à cette égalité pour pouvoir s'y référer.

```
> Formule_de_physique:=F=m*a;
```

$$Formule\_de\_physique := F = ma \tag{5.3}$$

Même si c'est une mauvaise habitude d'utiliser la syntaxe fonctionnelle pour donner un nom à une expression, voyons comment reconnaîtra l'expression suivante.

```
> f(x)=2*x-3;
```

$$f(x) = 2x - 3 \quad \textbf{(5.4)}$$

```
> f(x);
```

$$f(x) \quad \textbf{(5.5)}$$

```
> f(5);
```

$$f(5) \quad \textbf{(5.6)}$$

Dans la notation fonctionnelle, f(*x*) aurait dû pointer vers la formule de calcul des images et f(5) vers la valeur calculée avec la formule,

Cela n'a pas été le cas car ce qui a été créé c'est une égalité.

En effet:

```
> whattype((5.4));
```

$$`=` \quad \textbf{(5.7)}$$

Avec la requête $f(x) = 2x - 3$, $f(x)$ n'est que le membre de gauche d'une égalité. f(*x*) est une variable libre.

b) Expression de type addition

```
> f(x):=2*x-3;
```

$$f(x) := 2x - 3 \quad \textbf{(5.8)}$$

```
> f(x);
```

$$2x - 3 \quad \textbf{(5.9)}$$

```
> f(5);
```

$$f(5) \quad \textbf{(5.10)}$$

Dans la notation fonctionnelle, f(*x*) aurait dû pointer vers la formule de la fonction et f(5) vers la valeur calculée avec la formule,

Cela n'a pas été le cas car ce qui a été créé avec la requête $f(x) := 2x - 3$, c'est une expression de type addition. En effet, on a assigné au nom $f(x)$ une expression de type addition.

Voici:

```
> whattype((5.8));
```

$$`+` \quad \textbf{(5.11)}$$

```
> f(x):='f(x)';
```

$$f(x) := f(x) \quad \textbf{(5.12)}$$

c) Expression de type fonctionnel

L'opérateur fonctionnel « -> » est utile pour la création de fonctions.

```
> f:=x->2*x-3;
```

$$f := x \mapsto 2 \cdot x - 3 \quad \textbf{(5.13)}$$

Une procédure est en quelque sorte la transposition informatique du concept mathématique de fonction.

```
> f(x);
```

$$2x - 3 \quad \textbf{(5.14)}$$

```
> f(5);
```

(5.15)

$$7 \tag{5.15}$$

C'est donc tout à fait le résultat attendu: nous venons de créer en Maple une fonction.
En effet,

```
> whattype((5.13));
```

$$\textit{procedure} \tag{5.16}$$

On peut alors précéder à d'autre évaluation.

```
> f(t);
  f(x+h);
```

$$2t-3$$
$$2x+2h-3 \tag{5.17}$$

Il est possible de créer une fonction à partir d'une expression à l'aide de la macro-commande **unapply()**.

```
> Expression:=(2*x+1)/(x^2-1);
```

$$\textit{Expression} := \frac{2x+1}{x^2-1} \tag{5.18}$$

```
> g:=unapply(Expression,x);
```

$$g := x \mapsto \frac{2 \cdot x + 1}{x^2-1} \tag{5.19}$$

```
> g(3+h);
```

$$\frac{7+2h}{(3+h)^2-1} \tag{5.20}$$

```
> f:='f':
  g:='g':
```

Résolution d'équations et notation indicielle

La résolution d'une équation est réalisée avec la macro-commande solve.

```
> Éq_1:=(x/(x-1))^2=6*(x/(x-1))+7;
```

$$\textit{Éq\_1} := \frac{x^2}{(x-1)^2} = \frac{6x}{x-1} + 7 \tag{6.1}$$

Avec la macro-commande `solve`, il y a deux syntaxes permise: avec ou sans l'utilisation des accolades autour de la variable de résolution.

```
> solve(Éq_1,{x});
```

$$\left\{x = \frac{1}{2}\right\}, \left\{x = \frac{7}{6}\right\} \tag{6.2}$$

```
> solve(Éq_1,x);
```

$$\frac{1}{2}, \frac{7}{6} \tag{6.3}$$

Observer la différence dans la présentation du résultat selon qu'on utilise ou non les accolades dans la macro-commande `solve`.

Réponse: L'ensemble solution de l'équation est

$$\left\{\frac{1}{2}, \frac{7}{6}\right\}.$$

Voici un autre exemple.

```
> f:=x->2*x^2+5*x+6;
```

$$f := x \mapsto 2 \cdot x^2 + 5 \cdot x + 6 \tag{6.4}$$

```
> Racines:=solve(f(x)=0,x);
```

$$Racines := -\frac{5}{4} + \frac{I\sqrt{23}}{4}, -\frac{5}{4} - \frac{I\sqrt{23}}{4} \tag{6.5}$$

Observer que les deux zéros de la fonction f sont imaginaires.

En donnant un nom au résultat, il est plus commode ensuite de pointer vers l'un ou l'autre des éléments du résultat. En effet, Racines étant une séquence, il suffit d'utiliser la notation indicielle Racines[1] et Racines [2] pour pointer respectivement vers les premier et le second élément de Racines.

```
> Racines;
```

$$-\frac{5}{4} + \frac{I\sqrt{23}}{4}, -\frac{5}{4} - \frac{I\sqrt{23}}{4} \tag{6.6}$$

```
> Racines[1];
```

$$-\frac{5}{4} + \frac{I\sqrt{23}}{4} \tag{6.7}$$

```
> Racines[2];
```

$$-\frac{5}{4} - \frac{I\sqrt{23}}{4} \tag{6.8}$$

Vérifions si ce sont bien deux zéros de la fonction f.

```
> Verification1:=f(Racines[1]);
  Verification2:=f(Racines[2]);
```

$$Verification1 := 2\left(-\frac{5}{4} + \frac{I\sqrt{23}}{4}\right)^2 - \frac{1}{4} + \frac{5I\sqrt{23}}{4}$$

$$Verification2 := 2\left(-\frac{5}{4} - \frac{I\sqrt{23}}{4}\right)^2 - \frac{1}{4} - \frac{5I\sqrt{23}}{4} \tag{6.9}$$

Remarquons qu'il y a tout de même eu simplification automatique. Allons encore plus loin dans la simplification en opérant une simplification sur demande avec radnormal.

```
> radnormal(Verification1);
  radnormal(Verification2);
```

$$0$$

$$0 \tag{6.10}$$

Remarque importante: Il ne faut pas faire l'erreur suivante des débutants en voulant résoudre l'équation $S = \sqrt{2\pi r}$ pour r.

```
> S:=sqrt(2*Pi*r);
```

$$S := \sqrt{2}\sqrt{\pi r} \tag{6.11}$$

```
> solve(S,r);
```

$$0 \tag{6.12}$$

Il ne faut donc pas confondre l'opérateur `Maple` d'assignation d'avec le signe d'égalité définissant une équation.

```
> S:='S':  # À cause de l'assignation faite dans l'avant-dernière
  requête
> solve(S=sqrt(2*Pi*r),{r});
```

$$\left\{ r = \frac{S^2}{2\pi} \right\} \tag{6.13}$$

Résolution d'inéquations

```
> Inéquation:=x^2+6*x+5<0;
```

$$Inéquation := x^2 + 6x < -5 \tag{7.1}$$

```
> E_S:=solve(Inéquation,x);
```

$$E\_S := (-5, -1) \tag{7.2}$$

```
> E_S:=solve(Inéquation,{x});
```

$$E\_S := \{-5 < x, x < -1\} \tag{7.3}$$

Observer encore ici la différence dans la présentation du résultat de la macro-commande `solve` selon que les accolades sont utilisées ou non.

La notation $(-5, -1)$ est la notation anglosaxone d'un intervalle ouvert.

Dans les deux cas, on est amené à formuler la réponse suivante.
Réponse: L'ensemble solution est $]-5, -1[$.

L'ensemble solution obtenu de l'évaluateur est un objet Maple qu'il nous faut interpréter car c'est un objet Maple de type ensemble (set) qui contient deux objets de type `<` . L'ensemble-solution n'est donc pas, pour Maple, un ensemble de nombre réels.

```
> member(-3,E_S);
```

$$false \tag{7.4}$$

```
> member(-5<x,E_S);
```

$$true \tag{7.5}$$

La macro-commande solve est aussi utile pour résoudre un système d'équations ou un système d'inéquations.

Résolvons le système d'équations suivant:

$$\begin{aligned} -2x + y + 2z + w &= 12 \\ y - 2z - 3w &= -20 \\ 2x + y + 3z + 2w &= 12 \\ -x - y - 3z + w &= 28 \end{aligned}$$

```
> Éq1:=-2*x + y + 2*z + w   =12;
  Éq2:= y - 2*z  - 3*w = -20;
```

```
Éq3:= 2*x + y + 3*z + 2*w  = 12;
Éq4:= -x - y - 3*z + w  = 28;
```

$$Éq1 := -2x + y + 2z + w = 12$$

$$Éq2 := y - 2z - 3w = -20$$

$$Éq3 := 2x + y + 3z + 2w = 12$$

$$Éq4 := -x - y - 3z + w = 28 \quad \textbf{(7.6)}$$

```
> solve({Éq1,Éq2,Éq3,Éq4},{x,y,z,w});
```

$$\left\{w = \frac{376}{27}, x = -\frac{16}{9}, y = \frac{220}{27}, z = -\frac{184}{27}\right\} \quad \textbf{(7.7)}$$

Remarque: Attention à l'ordre alphabétique des variables.

Réponse: L'ensemble solution est $\left\{\left(-\frac{16}{9}, \frac{220}{27}, -\frac{184}{27}, \frac{376}{27}\right)\right\}$.

Fonctions rationnelles

Il est **possible** de créer une fonction rationnelle avec des facteurs communs explicitement au numérateur et au dénominateur. Le mécanisme de la simplification automatique le permet.

```
> f:=x->(x+2)*(x^2+2*x+3)/(x+2);
```

$$f := x \mapsto \frac{(x+2)\cdot(x^2+2\cdot x+3)}{x+2} \quad \textbf{(8.1)}$$

```
> 'f'(x)=f(x);
```

$$f(x) = x^2 + 2x + 3 \quad \textbf{(8.2)}$$

Par contre,

```
> f(-2);
Error, (in f) numeric exception: division by zero
```

Bien qu'il y a affichage de $f(x) = x^2 + 2x + 3$, Maple a mémorisé que f: $x \to \frac{(x+2)\,(x^2+2x+3)}{x+2}$.

Macro-commande `piecewise`

La macro-commande piecewise est utile pour créer des formules de calculs par morceaux.

```
> Formule:=piecewise(x<2,x^3+1,x>3 and x<5,4,x>5,1/x);
```

$$Formule := \begin{cases} x^3 + 1 & x < 2 \\ 4 & 3 < x < 5 \\ \frac{1}{x} & 5 < x \end{cases} \quad \textbf{(9.1)}$$

```
> eval(Formule,x=10);
```

$$\frac{1}{10} \quad \textbf{(9.2)}$$

```
> eval(Formule,x=0);
```

$$1 \tag{9.3}$$

```
> eval(Formule,x=3);
```

$$0 \tag{9.4}$$

Ce dernier résultat vous semble-t-il inattendu ? Ben Oui !

Justifier votre résultat.

Incapable de le justifier sans approfondire cette macro-commande.

Simplifions Formule, cela montrera comment `Maple` opère l'évaluation dans ce cas. Avec une vérification systématique des conditions une à une du haut vers le bas, lorsque qu'une condition est vérifiée, il y a alors aussitôt évaluation de l'expression associée à cette condition. Sachons que Maple donne par défaut, la valeur 0 associée aux valeurs de x qui ne sont pas conditionnées (pas dans le domaine). Ce qui n'est pas pratique.

```
> simplify(Formule);
```

$$\begin{cases} x^3+1 & x<2 \\ 0 & x\le 3 \\ 4 & x<5 \\ 0 & x=5 \\ \frac{1}{x} & 5<x \end{cases} \tag{9.5}$$

Heureusement, on peut préciser, en option, dans la macro-commande `piecewie`, une "valeur" qui sera pris en compte pour toutes les valeurs de x qui ne sont pas conditionnées. Si on veut que l'expression ne soit pas définie, il suffit de précisiser comme dernier argument dans la macro-commande `piecewise`, une variable libre, la variable "non_définie" par exemple.

```
> Formule:=piecewise(x<2,x^3+1,x>3 and x<5,4,x>5,1/x,non_définie);
```

$$Formule := \begin{cases} x^3+1 & x<2 \\ 4 & 3<x<5 \\ \frac{1}{x} & 5<x \\ non\_définie & otherwise \end{cases} \tag{9.6}$$

```
> simplify(Formule);
```

$$\begin{cases} x^3+1 & x<2 \\ non\_définie & x\le 3 \\ 4 & x<5 \\ non\_définie & x=5 \\ \frac{1}{x} & 5<x \end{cases} \tag{9.7}$$

Évaluons maintenant Formule avec $x=3$.

```
> eval(Formule,x=3);
```

$$non\ \ définie \tag{9.8}$$

Pouvez-vous expliquer les deux évaluations suivantes.

```
> eval(Formule,x=infinity);
```

$$0 \tag{9.9}$$

```
> eval(Formule,x=-infinity);
```

$$-\infty \tag{9.10}$$

Votre réponse.

Retour sur le mécanisme de la simplification automatique

Dans les nombres réels, l'expression $\sqrt{x^2} = |x|$. Qu'en est-il de cette simplication avec `Maple`.

```
> Expression:=sqrt(x^2);
```

$$Expression := \sqrt{x^2} \tag{10.1}$$

```
> simplify(Expression);
```

$$\mathrm{csgn}(x)\, x \tag{10.2}$$

Ce qui nous indique qu'en général, nous n'avons pas $\sqrt{x^2} = x$ ni même $\sqrt{x^2} = |x|$.

Afin d'obtenir la simplification attendue, soit $|x|$, il faut spécifier à Maple de faire la simplification demandée en tenant compte que la variable x représente un nombre réel.

```
> simplify(Expression, assume=real);
```

$$|x| \tag{10.3}$$

Montrons, en effet, que généralement dans $\mathbb{C}$, $\sqrt{x^2} \neq x$ et $\sqrt{x^2} \neq |x|$. Considérons le nombre imaginaire $-1 - i$.

```
> x:=-1-I;
```

$$x := -1 - \mathrm{I} \tag{10.4}$$

```
> 'sqrt(x^2)'=sqrt(x^2);
```

$$\sqrt{x^2} = 1 + \mathrm{I} \tag{10.5}$$

```
> evalb(sqrt(x^2)=x);
```

$$false \tag{10.6}$$

On voit bien que dans $\mathbb{C}$, en général $\sqrt{x^2} \neq x$.

Montrons ensuite que dans $\mathbb{C}$, $\sqrt{x^2}$ ne se simplifie pas par $|x|$. Nous avons montrer que $\sqrt{x^2} = 1 + \mathrm{I}$. Simplifions $|x|$.

```
> abs(x);
```

$$\sqrt{2} \tag{10.7}$$

```
> evalb(sqrt(x^2)=abs(x));
```

$$false \tag{10.8}$$

```
> x:='x': # Rendons x libre pour le prochain exemple
```

Présentons une autre situation de simplification automatique inattendue.

Dans $\mathbb{R}$, on a, bien sûr, $e^{\ln(x)} = x$ et $\ln(e^x) = x$. Mais, qu'en est-il avec Maple. Ne pas oublier que pour Maple, le référentiel est l'ensemble des nombres complexes.

```
> exp(ln(x));
```

$$x \tag{10.9}$$

```
> ln(exp(x));
```

$$\ln(e^x) \tag{10.10}$$

Avec Maple, $\ln(e^x)$ n'a pas été automatiquement simplifiée par x. Pourquoi ? Tout simplement parce que ce n'est pas toujours vraie. C'est-à-dire que l'égalité $\ln(e^x) = x$ n'est pas une identité dans $\mathbb{C}$.

```
> ln(exp(-3-5*I));
```

$$\ln(e^{-3-5I}) \tag{10.11}$$

```
> evalc((10.11));
```

$$-3 + I(-5 + 2\pi) \tag{10.12}$$

Ce qui montre que $\ln(e^{-3-5I}) \neq -3 - 5I$.

Pour terminer cette section, simplifions $\ln(e^x)$ sur la base que $x \in \mathbb{R}$. Ce qui montrera que nous avons $\ln(e^x) = x$ dans $\mathbb{R}$ seulement.

```
> simplify(ln(exp(x)),assume=real);
```

$$x \tag{10.13}$$