R1.07 TP 1

# Introduction au calcul formel avec *sagemath*

11 décembre 2025 – B. Colombel

Un compte-rendu de ce TP (réponses aux exercices et aux questions) est à rédiger sur le notebook *Jupyter* et votre compte rendu doit être déposé sous le nom

`prenom.nom-TP1.ipynb`

dans le dossier relatif à ce TP sur Ametice (R1.07, TP1).

## 1 Calcul symbolique

Pour l'instant, nous avons surtout utiliser *sagemath* pour écrire quelques algorithmes itératifs mais *sagemath* permet de faire bien autre chose.
En particulier, nous pouvons effectuer du calcul *formel*, ou calcul *symbolique*, c'est-à-dire que *sagemeth* permet de travailler directement avec des expressions algébriques.

Pour cela, il faut signifier au logiciel que la lettre `x` n'est plus une variable « numérique » mais une variable « symbolique » à l'aide de la commande `var()`.
Par exemple, la commande suivante ne génère pas d'erreur alors qu'aucune valeur n'a été affectée à `x`.

[1]:
```
x = var('x')
(x*x*x)^2
```

[1]:
```
x^6
```

**Exercice 1**. Tester les commandes `simplify()`, `expand()`, `factor()` sur $(x+3)(x+2)-2(x+2)$. Commenter.

## 2 Des mathématiques expérimentales

L'utilisation d'un système de calcul symbolique conduit le mathématicien à un type de démarche auquel il n'est traditionnellement pas habitué : l'expérimentation !

1. Je me pose des questions ;
2. j'écris un algorithme pour tenter d'y répondre ;
3. je trouve une conjecture sur la base des résultats expérimentaux ;
4. je prouve la conjecture.

**Exercice 2**. Comme dit précédemment, la commande `expand()` permet de développer une expression ; à l'inverse la commande `factor()` permet d'en factoriser une.

1. Demander à *sagemath* de développer $(x+y)^{20}$.
2. Réciproquement, demander-lui de factoriser $x^n + y^n$ pour $n$ allant de 1 à 17.
3. Quelles sont les expressions n'admettant aucune factorisation ? Conjecturez un résultat général.

**Attention**, ceci n'est pas une démonstration. Nous n'en sommes qu'à l'étape expérimentale !

Vous avez été quelques uns à m'expliquer lors de la dernière évaluation de R1.06 que si $n = 8k+6$ alors son reste dans la division euclidienne par 4 était égal à 2 car c'était le cas pour $k = 0$, $k = 1$, $k = 2$, et $k = 3$.

Ce raisonnement est faux car il ne permet pas d'affirmer que cela se passera de la même façon pour $k \geq 4$.

**Exercice 3.** Les nombres s'écrivant sous la forme $F_n = 2^{2^n} + 1 \quad (n \in \mathbb{N})$ sont appelés les *nombres de Fermat*. En 1640, Pierre de FERMAT émit la conjecture que tous ces nombres étaient premiers mais ne réussit pas à le démontrer.

1. Demander à *sagemath* de factoriser les nombres $2^{2^n} + 1$ pour $n$ allant 0 à 4.
   Cela confirme-t-il la conjecture de Fermat ?
2. Confirmer ou infirmer la conjecture en demandant à *sagemath* de factoriser les nombres de Fermat pour $n$ allant de 5 à 8.
3. Conclure.

## 3 Résolution d'équation

Nous abordons maintenant les équations et leur résolution. Les fonctions principales utilisées sont `solve()` et `find_root()`.

Considérons l'équation $x^2 - 2 = 0$ qui admet deux solutions égales à $\sqrt{2}$ et $-\sqrt{2}$.

[2]:
```
x = var('x')
eq = x^2 -2
solutions = solve(eq, x)  # On précise la variable inconnue
print(solutions)
```

[2]:
```
[
x == -sqrt(2),
x == sqrt(2)
]
```

L'ensemble des solutions est retourné sous forme de liste.

[3]:
```
solutions[0]
```

[3]:
```
x == -sqrt(2)
```

[4]:
```
solutions[1]
```

[4]:
```
x == sqrt(2)
```

Pour effectuer des résolutions numériques d'équations (c'est-à-dire donnant une valeur approchée d'une solution), on utilise la fonction `find_root()` qui prend en argument une fonction d'une variable ou une égalité symbolique, et les bornes de l'intervalle dans lequel il faut chercher une solution.

[5]:
```
expr = x^2-2
sol = find_root(expr, -2, 2)
sol
```

[5]:
```
1.4142135623731364
```

Quelles différence remarquez vous ?

Il est important de préciser la variable représentant l'inconnue, en particulier si l'équation admet un paramètre. Par exemple, si l'on veut résoudre l'équation

$$x^2 - \frac{2}{\cos\varphi}x + \frac{5}{\cos^2\varphi} - 4 = 0 \quad \text{avec} \quad \varphi \in \left]-\frac{\pi}{2}; \frac{\pi}{2}\right[$$

[6]:
```
x, phi = var('x, phi')
eq = x**2 - 2/cos(phi)*x + 5/cos(phi)**2 - 4 == 0
eq
```

[6]:

```
x^2 - 2*x/cos(phi) + 5/cos(phi)^2 - 4 == 0
```

On la résout également avec la fonction `solve()` :

[7]:

```
solve(eq, x)
```

[7]:

```
[x == -(2*sqrt(cos(phi)^2 - 1) - 1)/cos(phi), x == (2*sqrt(cos(phi)^2 - 1) + 1)/cos(phi)]
```

Dernier exemple :

[8]:

```
a, b, c = var('a, b, c')
eq2 = a*x^2 + b*x + c == 0
assume(b^2-4*a*c >0) # on fait l'hypothèse que la nombre b^2-4*a*c >0 est positif
solve(eq2, x)
```

[8]:

```
[x == -1/2*(b + sqrt(b^2 - 4*a*c))/a, x == -1/2*(b - sqrt(b^2 - 4*a*c))/a]
```

Que retrouve-t-on ?

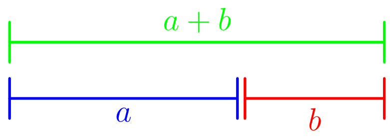
**Exercice 4.** On considère trois réels distincts $a$, $b$, $c$ et on définit le polynôme :

$$P(x) = \frac{(x-a)(x-b)}{(c-a)(c-b)} + \frac{(x-b)(x-c)}{(a-b)(a-c)} + \frac{(x-a)(x-c)}{(b-a)(b-c)} - 1$$

1. Définir trois variables *symboliques* par `a, b, c = var('a', 'b', 'c')` et créer une fonction `polynome(x)` qui renvoie $P(x)$.
2. Calculer $P(a)$, $P(b)$ det $P(c)$.
3. Comment un polynôme de degré 2 pourrait avoir 3 racines distinctes ? Expliquez ! (Vous pourrez appliquer la méthode `full_simplify()` à votre fonction ; n'hésitez pas à demander de l'aide et/ou des explications à votre charmant professeur !)

## 4 À vous de jouer ! ! !

**Exercice 5. (Le nombre d'or)**
Le nombre d'or est la proportion qui permet de diviser esthétiquement un segment en deux parties non égales :




La proportion vérifie :

$$\frac{a}{b} = \frac{a+b}{a}$$

Autrement dit « la grande longueur divisée par la petite longueur est égale à la longueur totale divisée par la grande longueur ».
On note cette proportion le nombre d'or :

$$\varphi = \frac{a}{b} = \frac{a+b}{a}$$

On peut montrer que $\phi$ vérifie l'équation :

$$\varphi^2 - \varphi - 1 = 0$$

1. En déduire que

$$\varphi = \frac{1+\sqrt{5}}{2}$$

et en donner une approximation numérique.
*fonctions utiles :* `solve()`, `n()`.

2. On définit la suite $u_0 = 1$, $u_1 = 1 + \frac{1}{1}$, $u_2 = 1 + \frac{1}{1+\frac{1}{1}}$, $u_3 = 1 + \frac{1}{\frac{1}{1+\frac{1}{1}}}$, etc.

   Plus généralement,

$$u_{n+1} = 1 + \frac{1}{u_n} \quad \text{pour} \quad n \geqslant 0$$

   Calculer les premiers termes de la suite et leurs valeurs numériques approchées. Quelle semble être la limite ?

3. On souhaite montrer que « pour tout $n$ positif, il existe deux suites $a_n$ et $b_n$ d'entiers telles que $\varphi^n = a_n\varphi + b_n$ ».
   Par exemple, $\varphi^5 = \frac{5\sqrt{5}}{2} + \frac{11}{2} = 5\varphi + 3$, donc $a_5 = 5$ et $b_5 = 3$.

   (a) Vérifier expérimentalement ce résultat pour les premières valeurs de $n$.

   (b) **Énigme**. Pour $n = 20$, $\varphi^{20} = a_{20}\varphi + b_{20}$. Que vaut $b_{20}$ ?

**Exercice 6**. **(Bonus)**
La suite de Fibonacci est définie par les relations suivantes :

$$F_0 = 0 \qquad F_1 = 1 \qquad F_n = F_{n-1} + F_{n-2} \quad (n \geqslant 2)$$

1. Écrire une fonction `Fibo(n)` qui retourne le liste des $n$ premiers termes de la suites de Fibonacci.
   Calculer les 10 premiers termes.
2. **Recherche d'une conjecture.** On cherche s'il peut exister des constantes $\phi$, $\psi$ et $c$ telles que :

$$F_n = c\,(\phi^n - \psi^n)$$

   Nous allons déterminer les constantes qui conviendraient.

   (a) On pose $G_n = \phi^n$. Quelle équation doit satisfaire $\phi$ afin que $G_n = G_{n-1} + G_{n-2}$ ?

   (b) Résoudre cette équation.

   (c) Calculer $c$, $\phi$ et $\psi$.

3. **Preuve.** On note $H_n = \frac{1}{\sqrt{5}}\,(\phi^n - \psi^n)$ où $\phi = \frac{1+\sqrt{5}}{2}$ et $\psi = \frac{1-\sqrt{5}}{2}$.

   (a) Vérifier que $H_n = F_n$ pour les premières valeurs de $n$.

   (b) Montrer à l'aide du calcul formel que $H_n = H_{n-1} + H_{n-2}$.
   Conclure.