Université Lyon 1 Année 2018-2019
M1G - UE Logiciels

TP1 – Sage

1 Pour commencer

1.1 Présentation du logiciel

Sage est un logiciel libre de mathématiques sous licence GPL qui combine la puissance de nombreux programmes libres dans une interface commune basée sur le langage de programmation **Python**.

Sage permet de faire des mathématiques générales et avancées, pures et appliquées. Il couvre une vaste gamme de mathématiques, dont l'algèbre, l'analyse, la théorie des nombres, la cryptographie, l'analyse numérique, l'algèbre commutative, la théorie des groupes, la combinatoire, la théorie des graphes, l'algèbre linéaire formelle, etc.

Sage a deux modes d'utilisation :

— un mode interactif (ou notebook), dont l'interface est un navigateur web disponible à l'adresse suivante :
`https://ipy.mecanique.univ-lyon1.fr`
et se connecter en utilisant son identifiant et son mot de passe Lyon 1 habituels. Il suffit ensuite d'aller dans l'onglet `New` et de choisir `SageMath 8.0`.
↪ *L'installation de Sage n'est pas obligatoire pour le mode bloc-note, le logiciel fonctionnant alors en mode client serveur. Ce mode permet de partager ou publier simplement feuilles de calcul, figures et graphiques.*

— un mode ligne de commande après installation du logiciel à l'adresse suivante :
`http://www.sagemath.org/download.html`
On utilise ensuite Sage en entrant la commande `sage` dans un terminal.
↪ *On peut ouvrir le mode interactif à partir du mode ligne de commandes en entrant la commande* `notebook()`.

1.2 Aide

La documentation en ligne peut être trouvée à l'adresse suivante : `http://www.sagemath.org/doc/index.html`

On peut accéder de manière interactive à la description d'une fonction, sa syntaxe et des exemples d'utilisation en la faisant suivre d'un point d'interrogation (exemple : `sin?`).

En cas de doute sur l'orthographe de la fonction, la touche tabulation <Tab> à la suite d'un début de mot montre quelles sont les commandes commençant par ces lettres.

1.3 Remarques

— Sage utilise des classes, c'est à dire que chaque objet est défini par ses attributs et un certain nombre de *méthodes*, c'est à dire de fonctions spécifiques à cet objet. Par exemple, le type `vector` est défini par un tableau d'entiers, de réels ou de complexes, et possède une méthode `degree` qui est appelée de la manière suivante :

```
v=vector([1,2,3])
v.degree()
```

Une fonction non liée à un objet spécifique sera appelée de manière classique

```
v=vector([1,2,3])
show(v)
```

— Une même méthode a souvent une variante spécifique adaptée à chaque type.
Exemple : `expand_trig()` est une version de la méthode `expand()` spécifique aux expressions trigonométriques.

— ATTENTION ! La numérotation des indices des vecteurs commence à 0.

1.4 Bibliographie

Un livre de référence pour débuter : *Calcul mathématique avec Sage* de P. Zimmerman *et al.*. Une version électronique est disponible librement à l'adresse suivante :

`http://dl.lateralis.org/public/sagebook/sagebook-web-20130530.pdf`

Une version physique peut être achetée sur Amazon.fr pour une dizaine d'euros.

2 Exercices préliminaires

Pour tous les exercices suivants, écrire les commandes suivantes et observer les réponses du logiciel.

Exercice 1 - Calculatrice, scalaires, vecteurs

```
5
2+5
factorial(100)
factorial(1000)
2345/34578
2345./34578
5^(1/3)
5.^(1/3)
5^(1./3)
1/3+2
1./3+2
sqrt(2)
sqrt(2.)
```

Que fait l'ajout du . ?

```
a=10
a
a+5
b=a+5
b
```

Que pensez-vous du résultat ?

```
c=[1,-2,7,0,10,8,3]
2*c
c=vector([1,-2,7,0,10,8,3])
2*c
c[3]
c[0]+c[1]+c[2]+c[3]+c[4]
c[0]=0
c
c.degree()
```

Que renvoie la méthode `degree()` ?

```
cos(2*pi)
exp(1)
sqrt(-1)
2+3*I
```

Exercice 2 - Fonctions usuelles

```
x=-1; y=2; z=3.4;
abs(x)
max(x,y)
min(x,y)
exp(x)
ln(y)
sin(y)
arctan(x)
ceil(z)
floor(z)
sqrt(y)
sqrt(y).n()
```

Que fait la dernière commande ?

Exercice 3 - Commandes usuelles

Variable symbolique

Les variables symboliques sont déclarées à l'aide de la commande `var`.
En regardant l'aide, donner toutes les manières de déclarer des variables muettes x, y, z.

Sommes

```
var('i,n')
sum(i^3,i,1,20)
sum(i^3,i,1,n)
sum(i^3,i,1,n).factor()
sum(1/i^4,i,1,oo)
```

Limites

```
var('x')
((1+1/x)^x).limit(x=oo)
(sin(x)/x).limit(x=0)
(1/x).limit(x=0, dir='right')
```

Dérivées, primitives, intégrales

```
x=var('x')
sqrt(x).diff(x)
sin(x).integral(x)
sin(x).integral(x,0,1)
exp(x^2).integral(x,-1,2)
```

Exercice 4 - Graphiques

```
var('x y t')
plot(sin(x),x,-pi,pi)
plot([x,x^2],x,0,1)

p1=parametric_plot((cos(x),sin(x)),(x,0,2*pi))
p2=parametric_plot((cos(x),sin(x)^2),(x,0,2*pi))
p3=parametric_plot((cos(x),sin(x)^3),(x,0,2*pi))
show(p1+p2+p3, axes=false)

L=[[-1+cos(pi*i/100)*(1+cos(pi*i/100)),
2*sin(pi*i/100)*(1-cos(pi*i/100))]
for i in range(200)]
polygon(L).show()

p=plot3d(lambda x,y: x^2 + y^2, (-2,2), (-2,2))
p.show(viewer='tachyon')
```

Exercice 5 - Expressions

En mode interactif, il est possible d'afficher une expression `expr` de manière "mathématique" en utilisant les commandes `expr.show()` ou `show(expr)`.

Développement, factorisation, rangement

```
var('x y')
m=(x^2-1)*(x+3)
m.expand()
m.factor()
m.factor_list()

n=(x+y)*(x+1)^2
n.collect(x)
n.expand().collect(x)
n.expand().collect(y)
n.expand().collect(y).collect(x)

((x+y+sin(x))^2).expand().collect(sin(x))
```

Fractions rationnelles

```
var('x y')
r=(x^3+x^2*y+3*x^2+3*x*y+2*x
  +2*y)/(x^3+2*x^2+x*y+2*y)
show(r.simplify_rational())
show(r.factor())
show(r.factor().expand())

var('x y a b c')
r=((x-1)*x/(x^2-7)+y^2/(x^2-7)
  +b/a+c/a+1/(x+1))
show(r.combine())

r=1/((x+1)^2*(x+2))
show(r.partial_fraction())
show(r.partial_fraction(x))
show(r.partial_fraction(y))
```

Opérations trigonométriques

```
var('x')
(cos(x)^2+sin(x)^2).simplify()
(cos(x)^2+sin(x)^2).simplify_trig()

sin(2*x).expand_trig()
(sin(x)^2).reduce_trig()
```

Autres opérations

```
var('n x')
f=factorial(n+1)/factorial(n)
f.simplify_factorial()

var('a b')
f=(a-b)^2/(2*(sqrt(a)+sqrt(b))^2)
f.canonicalize_radical()
```

↪ *La commande* `simplify_full` *permet d'appliquer les fonctions (dans l'ordre)*

- `simplify_factorial`
- `simplify_trig`
- `simplify_rational`

Exercice 6 - Polynômes

Définition

Les polynômes peuvent être définis de deux manières :

- Comme des expressions polynômiales :
  ```
  var('x')
  p=(2*x+1)*(x+2)*(x^4-1)
  ```
- Comme des éléments de $\mathbb{Q}[X]$:
  ```
  x=polygen(QQ,'x')
  p=(2*x+1)*(x+2)*(x^4-1)
  ```

Essayer d'utiliser les fonctions `degree`, `factor`, `expand` et `collect` dans les deux cas.

Anneaux

Il est possible de définir les polynômes dans d'autres anneaux. Essayer les lignes de commandes suivantes en remplaçant successivement `QQ` par `RR` puis `CC`.

```
x=polygen(QQ,'x')
p=(2*x+1)*(x+2)*(x^4-1)
show(p.factor())
```

Dans quels anneaux sont alors définis les polynômes ?

Exercice 7 - Résolution

```
var('x y z')
solve([x+y==1, 2*x-y==3], x, y)
solve([x+y+z==1, 3*x-2*y-z==2,
      -x+y-z==0], x,y,z)
solve([x^2+y^2==1, y^2==x^3+x+1], x, y)
```

Pour ce dernier système, récupérer la valeur de la troisième solution de x en utilisant la méthode `.rhs()`.

```
solve([x+y==3, 2*x+2*y==6], x, y)
solve([cos(x)*sin(x)==1/2, x+y==0], x, y)
```

Que représentent les inconnues `r1` et `z36` (ou `z60`) qui apparaissent dans les deux dernières réponses du logiciel ?

```
solve(x^2+1==0,x)
solve(x^2+1,x)
(x^2+1).roots()
```

Quelle est la différence entre `root` et `solve` ?

```
(x^3+2*x+1).roots(x)
(x^3+2*x+1).roots(x, ring=RR)
(x^3+2*x+1).roots(x, ring=CC)

q=sin(x)+sin(2*x)+sin(3*x)
solve(q,x)
```

Que se passe-t-il pour la dernière équation ? Deux solutions sont possibles dans ce cas :

- Essayer de transformer l'expression pour obtenir une forme plus facile à résoudre :
  ```
  f=q.simplify_trig()
  solve(f,x)
  ```
- Chercher une solution numérique :
  ```
  find_root(q,0.1,pi)
  ```

---

3 Exercices d'application

Exercice 1

Le plan est muni d'un repère orthonormé d'axes Ox et Oy. Déterminer les points d'intersection de deux cercles $C1$ et $C2$ (de rayons respectifs $R1$ et $R2$) d'équations

$$(C1) : x^2 + y^2 = R_1^2$$
$$(C2) : (x - R_1)^2 + y^2 = R_2^2$$

Exercice 2

Trouver un contre-exemple à une conjecture de Fermat : "Tout entier de la forme $2^{2^n} + 1$, avec $n \in \mathbb{N}$ est premier."

↪ *Utiliser par exemple les méthodes* `is_prime` *et* `factor`.

Exercice 3

Factoriser les expressions suivantes :

1. $49x^5 + 294x^4 - 56x^3 - 336x^2 + 16x + 96$
2. $x^6 - \sin(y)^3$

Exercice 4

On considère le polynôme de Tchebychev d'ordre n

$$P(t) = \cos(n \arccos t)$$

1. Vérifier pour quelques valeurs de n que $P(t)$ est un polynôme en t de degré n.
2. Donner la valeur exacte prise en $t = 2$ par le polynôme de Tchebychev d'ordre 5.

Exercice 5

Pour $j = e^{2\pi i/3}$ (racine cubique de l'unité), vérifier que

$$j^3 = 1 \ \text{ et } \ 1 + j + j^2 = 0.$$

Exercice 6

Vérifier la formule du binôme de Newton

$$\sum_{0 \le k \le n} C_n^k \, x^k \, y^{n-k} = (x + y)^n$$

↪ *Utiliser la fonction* `binomial`.

Exercice 7

On considère la formule de Ramanujan suivante :

$$\frac{1}{\pi} = \frac{2\sqrt{2}}{9801} \sum_{k \ge 0} \frac{(4k)! \, (1103 + 26390k)}{(k!)^4 \, 396^{4k}}$$

À combien de chiffres significatifs cette formule est-elle vraie en utilisant les 12 premiers termes de la série ?

↪ *Utiliser la méthode* `n(digits=...)` *pour imposer une précision de calcul (voir l'aide).*

Exercice 8

On considère la fonction suivante définie pour tout $(x, y) \neq (0, 0)$

$$f(x, y) = \frac{1}{2} \ln(x^2 + y^2),$$

Vérifier que f est harmonique pour tout $(x, y) \neq (0, 0)$, c'est à dire que l'on a

$$\Delta f(x, y) = 0, \ (x, y) \neq (0, 0).$$

Exercice 9

Calculer l'intégrale suivante :

$$\varphi(x) = \int_0^{+\infty} \frac{x \cos(u)}{u^2 + x^2} \, \mathrm{d}u$$

↪ *Utiliser les fonctions* `assume` *et* `forget`.

Exercice 10

Vérifier que

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

et que

$$\int_1^{+\infty} \frac{1}{x^2} \, \mathrm{d}x = 1.$$

Exercice 11

On considère u_n tel que

$$u_n = \sum_{0 \le k \le n} a \, q^k$$

Calculer la limite de u_n lorsque n tend vers ∞ en fonction de la valeur de q.
