Initiation au logiciel libre de calcul Sage

François DE MARÇAY
Département de Mathématiques d'Orsay
Université Paris-Sud, France

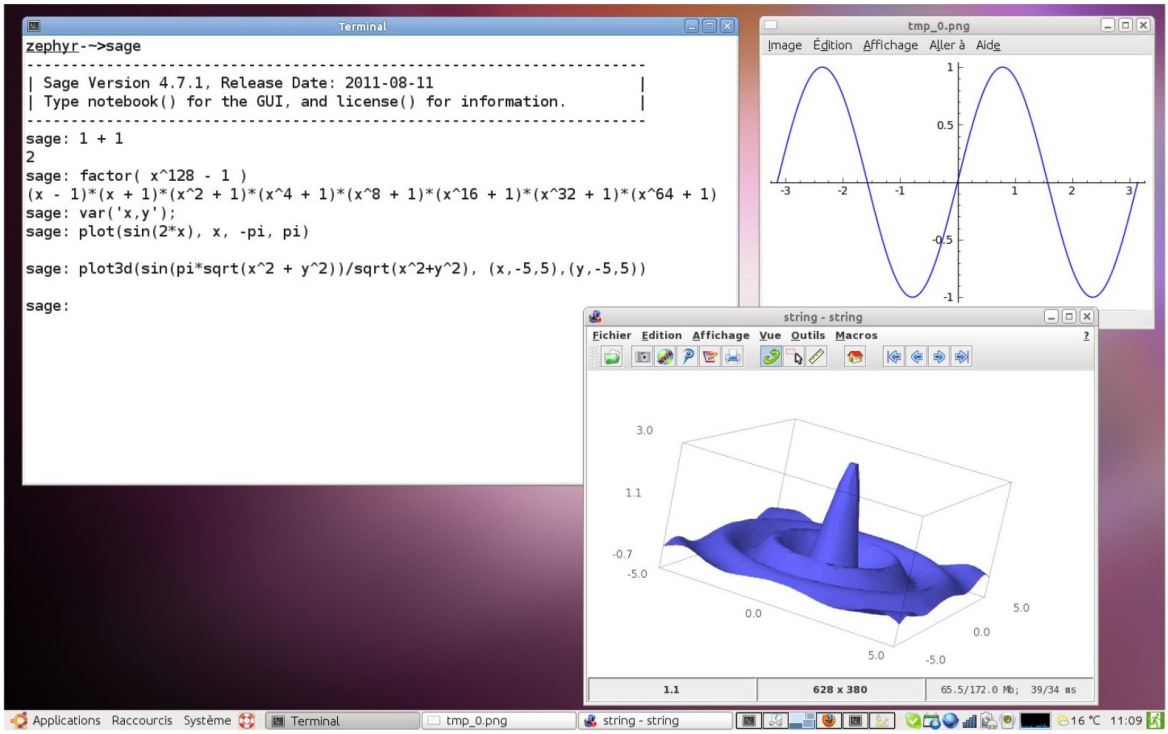
1. Accéder à Sage

Pour utiliser Sage, il suffit d'un navigateur web tel que Firefox. Le plus simple dans un premier temps est de se connecter sur un serveur de calcul Sage public comme `http://sagenb.org/`. On a alors accès à une interface utilisateur riche constituée d'un « bloc-notes » (*notebook* en anglais) permettant d'éditer et partager des feuilles de travail (*worksheets*, voir figure 1.2). De nombreuses universités et institutions mettent de tels serveurs à la disposition de leurs utilisateurs ; renseignez-vous autour de vous. Pour un usage plus intensif, on installe généralement Sage sur sa propre machine [1]. On peut alors toujours l'utiliser avec l'interface bloc-notes dans le navigateur web. Alternativement, on peut l'utiliser en ligne de commande, comme une calculatrice (voir figure 1.3).

Les figures 1.4 et 1.5 illustrent pas à pas les étapes pour accéder à un serveur Sage public et l'utiliser pour faire un calcul :

1. Ouvrez votre navigateur web et connectez-vous au site `http://sagenb.org` (figure 1.4). Vous pouvez dès à présent consulter les feuilles de travail publiques accessibles par le lien *Browse published Sage worksheets*.
2. Pour travailler sur vos propres feuilles de travail, vous devez vous authentifier ; pour cela `sagenb.org` utilise le système *OpenID* [2]. Si vous avez déjà, par exemple, un compte Google ou Yahoo, il vous suffit de suivre les liens correspondants. Sinon, vous devez d'abord créer un compte OpenID [3]. Ensuite, il ne restera qu'à revenir sur la page d'accueil, suivre le lien *OpenID*, et entrer votre identifiant OpenID (par exemple `http://votre-nom.myopenid.com/`).
3. Vous êtes maintenant sur la page principale de l'interface web de Sage, depuis laquelle vous pouvez gérer vos feuilles de travail. Créez-en une nouvelle (*Create worksheet*) et donnez-lui un nom (*Rename worksheet*).
4. Saisissez ensuite `1 + 1` dans la cellule de calcul, puis cliquez sur *evaluate* ou formez la combinaison de touches ⟨Alt⟩+⟨Entrée⟩ (figure 1.5).

Bravo, vous avez fait votre premier calcul avec Sage !



2. Sage comme calculatrice

2.1. Premier calcul :.

```
sage: 1+1
2
```

Le texte `sage:` au début de la première ligne est *l'invite* du système. L'invite (qui n'apparaît pas dans l'interface bloc-notes) indique que Sage attend une commande de l'utilisateur. La suite de la ligne correspond à la commande à saisir, puis à valider en appuyant sur ⟨Entrée⟩. Les lignes suivantes correspondent à la réponse du système, c'est-à-dire en général au résultat du calcul. Certaines instructions

2.2. Calculatrice simple :. Pour la multiplication, utiliser $\star$

```
sage: (1 + 2*(3 + 5))*2
34
```

2.3. Puissance :.

```
sage: 2^3
8
```

2.4. Division :.

```
sage: 20/6
10/3
```

2.5. Quotient de la division euclidienne entre deux entiers :.

```
sage: 20//6
3
```

2.6. Reste de la division euclidienne entre deux entiers :.

```
sage: 20 % 6
2
```

2.7. Plusieurs opérations sur une seule ligne :. Utiliser un point-virgule :

```
sage: 20//6;  20 % 6
3; 2
```

3. Quelques opérations usuelles

Opérateurs arithmétiques de base	
« quatre opérations »	`a+b`, `a-b`, `a*b`, `a/b`
puissance	`a^b` ou `a**b`
racine carrée	`sqrt(a)`
racine n-ième	`a^(1/n)`

Opérations sur les entiers	
division entière	`a // b`
modulo	`a % b`
factorielle $n!$	`factorial(n)`
coefficients binomiaux $\binom{n}{k}$	`binomial(n,k)`

Fonctions usuelles sur les réels, les complexes...	
partie entière	`floor(a)`
valeur absolue, module	`abs(a)`
fonctions élémentaires	`sin`, `cos`, ... (voir tableau 2.2)

Calcul formel et méthodes numériques

Un système de calcul formel est un logiciel qui a pour but de manipuler, de simplifier et de calculer des formules mathématiques en appliquant uniquement des transformations exactes. Le terme *formel* s'oppose ici à *numérique* ; il signifie que les calculs sont effectués sur des formules, de façon algébrique, en manipulant formellement des symboles. Pour cette raison l'expression *calcul symbolique* est parfois employée à la place de *calcul formel*. En anglais, on dit *computer algebra* ou parfois *symbolic computation*.

Les calculatrices manipulent de façon exacte les nombres entiers ayant moins d'une douzaine de chiffres ; les nombres plus grands sont arrondis, ce qui entraîne des erreurs. Ainsi une calculatrice numérique évalue de façon erronée l'expression suivante en obtenant 0 à la place de 1 :

$$(1 + 10^{50}) - 10^{50}.$$

De telles erreurs sont difficilement détectables si elles se produisent lors d'un calcul intermédiaire sans être prévues — ni facilement prévisibles — par une étude théorique. Les systèmes de calcul formel, au contraire, s'appliquent à repousser ces limites et à ne faire aucune approximation sur les nombres entiers pour que les opérations qui en découlent soient exactes : ils répondent 1 au calcul précédent.

Les méthodes d'analyse numérique approchent à une précision donnée (par une méthode des trapèzes, de Simpson, de Gauss, etc.) l'intégrale $\int_0^\pi \cos t \, \mathrm{d}t$ pour obtenir un résultat numérique plus ou moins proche de zéro (à 10^{-10} près par exemple) sans pouvoir affirmer si le résultat est le nombre entier 0 — de façon exacte — ou au contraire est proche de zéro mais est non nul.

Un système formel transforme par une manipulation de symboles mathématiques l'intégrale $\int_0^\pi \cos t \, \mathrm{d}t$ en la formule $\sin \pi - \sin 0$ qui est ensuite évaluée de façon exacte en $0 - 0 = 0$. Cette méthode prouve donc $\int_0^\pi \cos t \, \mathrm{d}t = 0$.

Cependant les transformations uniquement algébriques ont aussi des limites. La plupart des expressions manipulées par les systèmes formels sont des fractions rationnelles et l'expression a/a est automatiquement simplifiée en 1. Ce mode de calcul algébrique ne convient pas à la résolution des équations ; dans ce cadre, la solution de l'équation $ax = a$ est $x = a/a$ qui est simplifiée en $x = 1$ sans distinguer suivant la valeur de a par rapport à 0, valeur particulière pour laquelle tout nombre x est solution (voir aussi §2.1.5).

4. Syntaxe générale

4.1. Traitement des instructions :. Il peut se faire ligne par ligne, mais pas nécessairement.

4.2. Plusieurs instructions par ligne :. Séparer par un point-virgule :

```
sage: 1+1; 2+2;
2
4
```

4.3. Commenter une ligne sans qu'elle soit lue par Sage :. Utiliser le caractère croisillon « # » :

```
sage: 2*3; 3*4; 4*5                # Trois résultats sont attendus
6
12
20
```

4.4. Effectuer plusieurs retours à la ligne avant d'exécuter une commande :. Utiliser le caractère « \ » :

```
sage: 123 + \
....: 345
468
```

5. Fonctions élémentaires

5.1. Fonctions trigonométriques :. Les calculs sont exacts, non numériques :

```
sage: sin(pi)
0
sage: tan(pi/3)
sqrt(3)
sage: arctan(1)
1/4*pi
sage: exp(2 * I * pi)
1
```

quitte à renvoyer des formules plutôt que des valeurs numériques :

```
sage: arccos(sin(pi/3))
arccos(1/2*sqrt(3))
sage: sqrt(2)
sqrt(2)
sage: exp(I*pi/6)
e^(1/6*I*pi)
```

5.2. Commande simple souvent utile :. `Simplify`

```
sage: simplify(arcos(sin(pi/3)))
(1/6)*pi
```

```
sage: simplify(exp(i*pi/6))
(1/2)*sqrt(3) + 1/2*I
```

5.3. Constantes prédéfinies :.

Quelques valeurs spéciales	
valeurs de vérité « vrai » et « faux »	`True`, `False`
unité imaginaire i	`I` *ou* `i`
infini ∞	`infinity` *ou* `oo`

Constantes mathématiques usuelles	
constante d'Archimède π	`pi`
base du logarithme naturel $e = \exp(1)$	`e`
constante d'Euler-Mascheroni γ	`euler_gamma`
nombre d'or $\varphi = (1+\sqrt{5})/2$	`golden_ratio`
constante de Catalan	`catalan`

TABLEAU 1.2 – Constantes prédéfinies.

6. Aide en ligne et Complétion automatique

6.1. Accéder directement à de l'aide sur une commande :. Placer un point d'interrogation juste après la commande :

```
sage: sin?
```

6.2. Utiliser la touche <Tab> :.

```
sage: arc<tab>
Possible completions are:
arc  arccos  arccosh  arccot  arccoth  arccsc  arccsch
arcsec  arcsech  arcsin  arcsinh  arctan  arctan2  arctanh
```

7. Affectations de variables et Égalité

7.1. Affecter une valeur à une variable se fait par un « = » simple, car Sage n'utilise pas « := ».

7.2. Tester, déclarer une égalité ou comparer deux expressions se fait par un « == » double.

7.3. Ne pas confondre :.

Assignation : Symbole égal simple sans deux points « = »
Comparaison : Symbole égal double « == »

7.4. Exemple d'affectation :. Déclarer qu'à la variable y est affectée la valeur $1+2$:

```
sage: y = 1 + 2
sage: y
3
sage: (2 + y)*y
15
```

Éventuellement, affecter et réaffecter y :

```
sage: y = 1 + 2; y
3
sage: y = 3*y + 1; y
10
```

7.5. Exemple d'égalité :.

```
sage: a = 1
sage: bool(a==1)
True
sage: bool(a==2)
False
```

8. Variables symboliques

8.1. Différence importante avec d'autres logiciels de calcul formel (Maple, Maxima) :. Les variables symboliques du mathématicien, telles que x, y, z, doivent être **déclarées expressément** avant d'être employées.

8.2. Déclaration standard de variable formelle :. Dans l'exemple qui suit, la commande `SR.var('z')` « construit » une variable symbolique `z`.

```
sage: y = SR.var('z')
sage: 2*y + 3
2*z + 3
```

8.3. Déclaration abrégée de variables formelles :. De même, la commande `var('x')` « construit » une variable symbolique `x`.

```
sage: var('x')
x
sage: 2*x+3
2*x+3
```

et pour plusieurs variables :

```
sage: var('a, b, c, x, y')
(a, b, c, x, y)
sage: a*x+b*y+c
a*x + b*y + c
```

8.4. Substituer une valeur à une variable symbolique :.

```
sage: var('x')
sage: expr = sin(x); expr
sin(x)
sage: expr(x=1)
sin(1)
```

8.5. Évaluer une expression :. Utiliser la commande `.subs`

```
sage: a, x = var('a, x'); y = cos(x+a) * (x+1); y
(x + 1)*cos(a + x)
sage: y.subs(a=-x); y.subs(x=pi/2, a=pi/3); y.subs(x=0.5, a=2.3)
x + 1
-1/4*(pi + 2)*sqrt(3)
-1.41333351100299
```

ou encore :

```
sage: y(a=-x); y(x=pi/2, a=pi/3); y(x=0.5, a=2.3)
x + 1
-1/4*(pi + 2)*sqrt(3)
-1.41333351100299
```

8.6. Remplacer une sous-expression plus complexe qu'une variable :. Utiliser la commande

```
sage: y, z = var('y, z'); f = x^3 + y^2 + z
sage: f.subs_expr(x^3 == y^2, z==1)
2*y^2 + 1
```

8.7. Fonctions symboliques :. Définir des *fonctions symboliques* afin de manipuler des expressions :

```
sage: f(x)=(2*x+1)^3; f(-3)
-125
sage: f.expand()
x |--> 8*x^3 + 12*x^2 + 6*x + 1
```

8.8. Déclarer une fonction :. Utiliser la commande `function`

```
sage: var('x, y'); u = sin(x) + x*cos(y)
sage: v = u.function(x,y); v
(x, y) |--> x*cos(y) + sin(x)
```

Alternative :

```
sage: var('x, y'); u = sin(x) + x*cos(y)
sage: w(x, y) = u; w
(x, y) |--> x*cos(y) + sin(x)
```

8.9. Développer une expression :. Utiliser la commande `expand`

```
sage: x, y = SR.var('x,y')
sage: p = (x+y)*(x+1)^2
sage: p2 = p.expand(); p2
x^3 + x^2*y + 2*x^2 + 2*x*y + x + y
```

8.10. Regrouper des termes :. Utiliser la commande `collect`

```
sage: x, y = SR.var('x,y')
sage: p = (x+y)*(x+1)^2
sage: p2 = p.expand();
sage: p3 = p2.collect(x); p3
(y+2)*x^2 + x^3 + (2*y + 1)*x + y
```

8.11. Exemple utilisant simultanément :. `expand` et `collect`

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

9. Polynômes et Fractions Rationnelles

9.1. Commandes très souvent utiles :.

`expand`

`collect`

`factor`

`combine`

9.2. Exemples nombreux :.

Polynôme $p = zx^2 + x^2 - (x^2 + y^2)(ax - 2by) + zy^2 + y^2$	
`p.expand().collect(x)`	$-ax^3 - axy^2 + 2\,by^3 + (2\,by + z + 1)x^2 + y^2z + y^2$
`p.collect(x).collect(y)`	$2\,bx^2y + 2\,by^3 - (ax - z - 1)x^2 - (ax - z - 1)y^2$
`p.expand()`	$-ax^3 - axy^2 + 2\,bx^2y + 2\,by^3 + x^2z + y^2z + x^2 + y^2$
`p.factor()`	$-\left(x^2 + y^2\right)(ax - 2\,by - z - 1)$
`p.factor_list()`	$\left[\left(x^2 + y^2, 1\right), (ax - 2\,by - z - 1, 1), (-1, 1)\right]$

Fraction rationnelle $r = \frac{x^3+x^2y+3\,x^2+3\,xy+2\,x+2\,y}{x^3+2\,x^2+xy+2\,y}$	
`r.simplify_rational()`	$\frac{(x+1)y+x^2+x}{x^2+y}$
`r.factor()`	$\frac{(x+1)(x+y)}{x^2+y}$
`r.factor().expand()`	$\frac{x^2}{x^2+y} + \frac{xy}{x^2+y} + \frac{x}{x^2+y} + \frac{y}{x^2+y}$

Fraction rationnelle $r = \frac{(x-1)x}{x^2-7} + \frac{y^2}{x^2-7} + \frac{b}{a} + \frac{c}{a} + \frac{1}{x+1}$	
`r.combine()`	$\frac{(x-1)x+y^2}{x^2-7} + \frac{b+c}{a} + \frac{1}{x+1}$

Fraction rationnelle $r = \frac{1}{\left(x^3+1\right)y^2}$	
`r.partial_fraction(x)`	$\frac{-(x-2)}{3\left(x^2-x+1\right)y^2} + \frac{1}{3\,(x+1)y^2}$

10. Fonctions mathématiques usuelles

Fonctions mathématiques usuelles	
Exponentielle et logarithme	`exp, log`
Logarithme de base a	`log(x, a)`
Fonctions trigonométriques	`sin, cos, tan`
Fonctions trigonométriques réciproques	`arcsin, arccos, arctan`
Fonctions hyperboliques	`sinh, cosh, tanh`
Fonctions hyperboliques réciproques	`arcsinh, arccosh, arctanh`
Partie entière, etc.	`floor, ceil, trunc, round`
Racine carrée et n-ième	`sqrt, nth_root`

Transformation des expressions trigonométriques	
Simplification	`simplify_trig`
Linéarisation	`reduce_trig`
Anti-linéarisation	`expand_trig`

10.1. Commandes spécifiques pour simplifier :.

`simplify_trig`

`simplify_exp`

`simplify_factorial`

10.2. Exemples :.

```
sage: f = (e^x-1) / (1 + e^(x/2))
sage: f.simplify_exp()
e^(1/2*x) - 1
```

```
sage: f = cos(x)^6 + sin(x)^6 + 3*sin(x)^2*cos(x)^2
sage: f.simplify_trig()
1
```

10.3. Factorielles :.

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

11. Résolutions d'équations

11.1. Résumé-anticipation :.

Équations numériques	
Résolution symbolique	`solve`
Résolution (avec multiplicité)	`roots`
Résolution numérique	`find_root`

Équations vectorielles et équations fonctionnelles	
Résolution d'équations linéaires	`right_solve, left_solve`
Résolution d'équations différentielles	`desolve`
Résolution de récurrences	`rsolve`

11.2. Résolution explicite :. Considérons l'équation d'inconnue $z \in \mathbb{C}$ et de paramètre $\varphi \in \left]-\frac{\pi}{2}, \frac{\pi}{2}\right[$:

$$z^2 - \frac{2}{\cos\varphi}\, z + \frac{5}{\cos^2\varphi} - 4 = 0.$$

```
sage: z, phi = var('z,phi')
sage: eq = z^2 - 2*z/cos(phi) + 5/cos(phi)^2 - 4 == 0
z^2 - 2*z/cos(phi) + 5/cos(phi)^2 - 4 == 0
```

11.3. Extraction du membre de gauche et du membre de droite :.

```
sage: eq.lhs()
z^2 - 2*z/cos(phi) + 5/cos(phi)^2 - 4
sage: eq.rhs()
0
```

11.4. Résoudre :. Utiliser la commande `solve`

```
sage: solve(eq, z)
```

et obtenir :

$$\left[z=-2\,\frac{2\,\sqrt{\cos(\varphi)^2-1}-1}{\cos(\varphi)},\;\; z=-2\,\frac{2\,\sqrt{\cos(\varphi)^2-1}+1}{\cos(\varphi)}\right].$$

11.5. Exemple :.

```
sage: y = var('y'); solve(y^6==y, y)
[y == e^(2/5*I*pi), y == e^(4/5*I*pi), y == e^(-4/5*I*pi),
 y == e^(-2/5*I*pi), y == 1, y == 0]
```

11.6. Résoudre numériquement :. Lorsque Sage échoue :

```
sage: expt = sin(x) + sin(2*x) + sin(3*x)
sage: solve(expr, x)
[sin(3*x) == -sin(2*x) - sin(x)
```

utiliser la commande `find_root` qui prend en argument les bornes de l'intervalle dans lequel cherche une solution numérique :

```
sage: find_root(expr, 0.1, pi)
2.0943951023931957
```

On peut aussi transformer au préalable l'expression, si l'espoir de trouver une solution exacte persiste :

```
sage: f = expr.simplify_trig(); f
2*(2*cos(x)^2 + cos(x))*sin(x)
sage: solve(f, x)
[x == 0, x == 2/3*pi, x == 1/2*pi]
```

11.7. Résolution dans un corps déclaré :. La commande `roots` permet d'obtenir les solutions exactes d'une équation avec multiplicité, dans un corps donné.

11.8. Déclarer les deux corps standard $\mathbb{R}$ et $\mathbb{C}$:.

```
sage: ring=RR
```

```
sage: ring=CC
```

11.9. Exemple :. Considérer l'équation $x^3+2\,x+1=0$.

```
sage: (x^3+2*x+1).roots(x)
```

$$\left[\left(-\frac{1}{2}\left(I\sqrt{3}+1\right)\left(\frac{1}{18}\sqrt{3}\sqrt{59}-\frac{1}{2}\right)^{\left(\frac{1}{3}\right)}+\frac{-\left(I\sqrt{3}-1\right)}{3\left(\frac{1}{18}\sqrt{3}\sqrt{59}-\frac{1}{2}\right)^{\left(\frac{1}{3}\right)}},1\right),\right.$$
$$\left(-\frac{1}{2}\left(-I\sqrt{3}+1\right)\left(\frac{1}{18}\sqrt{3}\sqrt{59}-\frac{1}{2}\right)^{\left(\frac{1}{3}\right)}+\frac{-\left(-I\sqrt{3}-1\right)}{3\left(\frac{1}{18}\sqrt{3}\sqrt{59}-\frac{1}{2}\right)^{\left(\frac{1}{3}\right)}},1\right),$$
$$\left.\left(\left(\frac{1}{18}\sqrt{3}\sqrt{59}-\frac{1}{2}\right)^{\left(\frac{1}{3}\right)}+\frac{-2}{3\left(\frac{1}{18}\sqrt{3}\sqrt{59}-\frac{1}{2}\right)^{\left(\frac{1}{3}\right)}},1\right)\right]$$

Sur le corps $\mathbb{R}$:

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

$[(-0.453397651516404, 1)]$

Sur le corps $\mathbb{C}$:

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

$[(-0.453397651516404, 1), (0.226698825758202 - 1.46771150871022 * I, 1),$
$(0.226698825758202 + 1.46771150871022 * I, 1)]$

12. Sommes

12.1. Somme $1+2+3+\cdots+n$:.

```
sage: k, n = var('k, n')
sage: sum(k, k, 1, n).factor()
```

ce qui donne directement le résultat $\frac{1}{2}(n+1)n$.

12.2. Binôme de Newton :.

```
sage: n, k, y = var('n, k, y')
sage: sum(binomial(n,k)*x^k*y^(n-k), k, 0, n)
```

$(x+y)^n$

12.3. Trois autres exemples :.

```
sage: k, n = var('k, n')
sage: sum(binomial(n,k), k, 0, n), \
....: sum(k*binomial(n,k), k, 0, n), \
....: sum((-1)^k*binomial(n,k), k, 0, n), \
```

$\left(2^n,\ n\,2^{n-1},\ 0\right)$

12.4. Somme géométrique :.

```
sage: a, q, k, n = var('a, q, k, n')
sage: sum(a*q^k, k, 0, n)
```

$$\frac{a\,q^{n+1}-a}{q-1}$$

12.5. Calculer la série infinie :. Préciser que le module de la raison est strictement inférieur à 1 en valeur absolue :

```
sage: assume(abs(q) < 1)
sage: sum(a*q^k, k, 0, infinity)
```

$$-\frac{a}{q-1}$$

Exercice 1. [Calcul par récurrence des sommes de puissances p-èmes des n premiers entiers] Sans utiliser les algorithmes ou fonctions de Sage, calculer, pour $p=0,1,2,3,4,5$, les sommes :

$$S_n(p) \;=\; \sum_{k=1}^{n} k^p \;=\; 1^p+2^p+\cdots+n^p,$$

en utilisant la formule de récurrence suivantes, que l'on démontrera au préalable :

$$S_n(p) \;=\; \frac{1}{p+1}\left((n+1)^{p+1} \;-\; \sum_{j=0}^{p-1}\binom{p+1}{j} S_n(j)\right).$$

Indication: Pour obtenir cette formule, calculer de deux manières différentes la somme télescopique :

$$\sum_{0\leqslant k\leqslant n}\Big[(k+1)^{p+1}-k^{p+1}\Big].$$

13. Limites

13.1. Exemple :. Calculer les deux limites :

$a)\ \lim\limits_{x\to 8}\dfrac{\sqrt[3]{x}-2}{\sqrt[3]{x+19}-3}$;

$b)\ \lim\limits_{x\to\frac{\pi}{4}}\dfrac{\cos\left(\frac{\pi}{4}-x\right)-\tan x}{1-\sin\left(\frac{\pi}{4}+x\right)}$.

```
sage: limit((x**(1/3) - 2) / ((x + 19)**(1/3) - 3), x = 8)
9/4
sage: f(x) = (cos(pi/4-x)-tan(x))/(1-sin(pi/4 + x))
```

Obtenir :

```
sage: limit(f(x), x=pi/4)
Infinity
```

13.2. Limites à gauche et à droite :.

```
sage: limit(f(x), x = pi/4, dir='minus')
+Infinity
sage: limit(f(x), x = pi/4, dir='plus')
-Infinity
```

14. Suites

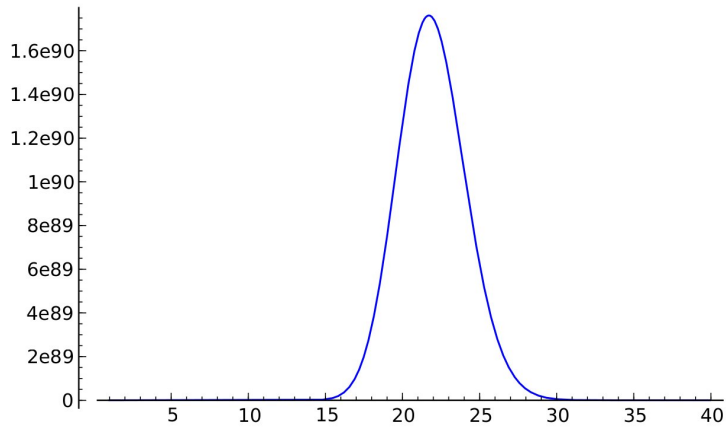
14.1. Exemple :. Étudier la suite :

$$u_n = \frac{n^{100}}{100^n}.$$

```
sage: u(n) = n^100 / 100^n
sage: u(2.);u(3.);u(4.);u(5.);u(6.);u(7.);u(8.);u(9.);u(10.)
1.26765060022823e26
5.15377520732011e41
1.60693804425899e52
7.88860905221012e59
6.53318623500071e65
3.23447650962476e70
2.03703597633449e74
2.65613988875875e77
1.00000000000000e80
```

14.2. Éviter de croire que $u_n \underset{n\to\infty}{\longrightarrow} \infty$:.

```
sage: plot(u(x), x, 1, 40)
```



14.3. Conjecturer que la suite commence à décroître à partir de $n = 22$:.

```
sage: v(x) = diff(u(x), x); sol = solve(v(x) == 0, x); sol
[x == 100/log(100), x == 0]
sage: floor(sol[0].rhs())
21
```

14.4. Calculer enfin la limite :.

```
sage: limit(u(n), n=infinity)
0
```

15. Développements limités

15.1. Commandes :.

`f(x).series(x==x0, n)`

`taylor(f(x), x, x0, n)`

15.2. Exemple 1 :. Calculer le développement en $x_0 = 0$ à l'ordre 3 de :

$$\left(1 + \arctan x\right)^{\frac{1}{x}}.$$

```
sage: taylor((1+arctan(x))**(1/x), x, 0, 3)
1/16*x^3*e + 1/8*x^2*e - 1/2*x*e + e
```

15.3. Exemple 2 :. Calculer le développement en $x_0 = \frac{\pi}{6}$ à l'ordre 3 de :

$$\log\left(2\sin x\right).$$

```
sage: (ln(2*sin(x))).series(x==pi/6, 3)
(sqrt(3))*(-1/6*pi + x) + (-2)*(-1/6*pi + x)^2 + Order(-1/216*(pi - 6*x)^3)
```

15.4. Extraire :. Utiliser la fonction `truncate`

```
sage: (ln(2*sin(x))).series(x==pi/6, 3).truncate()
-1/18*(pi - 6*x)^2 - 1/6*(pi - 6*x)*sqrt(3)
```

15.5. Calculer des développements asymptotiques :. Utiliser la fonction `taylor`, à l'infini. Par exemple, trouver pour $x \to \infty$ un équivalent de :

$$\left(x^3 + x\right)^{\frac{1}{3}} - \left(x^3 - x\right)^{\frac{1}{3}}.$$

```
sage: taylor( (x^3+x)^(1/3) - (x^3-x)^(1/3), x, infinity, 2)
2/3/x
```

15.6. Exemple : Formule de Machin 1706 :. L'astronome John Machin (1680–1752) calcula plus d'une centaine de décimales de π en utilisant la formule dite d'*arc-tangente* qui lui est due :

$$\frac{\pi}{4} = 4\arctan\frac{1}{5} - \arctan\frac{1}{239}.$$

formule qui connut de nombreuses modifications et améliorations.

• Utiliser la fonction `simplify_trig` :

```
sage: tan(4*arctan(1/5)).simplify_trig
120/119
sage: tan( pi/4 + arctan(1/239) ).simplify_trig
120/119
```

• Obtenir une valeur approchée :

```
sage: f = arctan(x).series(x, 10); f
1*x + (-1/3)*x^3 + 1/5*x^5 + (-1/7)*x^7 + 1/9*x^9 + Order(x^10)
sage: (16*f.subs(x==1/5) - 4*f.subs(x==1/239)).n(); pi.n()
3.14159268240440
3.14159265358979
```

Exercice 2. Soit la formule plus longue mais plus efficace due à Gauss :

$$\frac{\pi}{4} = 12\arctan\frac{1}{38} + 20\arctan\frac{1}{57} + 7\arctan\frac{1}{239} + 24\arctan\frac{1}{268}.$$

(a) Poser :

$$\theta = 12\arctan\frac{1}{38} + 20\arctan\frac{1}{57} + 7\arctan\frac{1}{239} + 24\arctan\frac{1}{268},$$

et vérifier avec Sage que $\tan\theta = 1$.

(b) Montrer que pour tout $x \in [0, \pi/4]$, on a :

$$\tan x \leqslant \tfrac{4}{\pi}\, x$$

(c) En approchant la fonction arctan par son polynôme de Taylor d'ordre 21 en 0, donner une nouvelle approximation, meilleure, de π.

16. Séries

16.1. Exemple :. Calculs de séries de Riemann :

```
sage: k = var('k')
sage: sum(1/k^2, k, 1, infinity),\
....: sum(1/k^4, k, 1, infinity),\
....: sum(1/k^5, k, 1, infinity)
(1/6*pi^2, 1/90*pi^4, zeta(5))
```

16.2. Exemple :. Formule due à Ramanujan :

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

• Calculer la somme partielle des 12 premiers termes, et obtenir 95 décimales correctes de π :

```
sage: s = 2*sqrt(2)/9801*(sum((factorial(4*k)) * (1103+26390*k) /
....:     ((factorial(k)) ^ 4 * 396 ^ (4 * k)) for k in (0..11)))
sage: (1/s).n(digits=100)
3.1415926535897932384626433832795028841971693993751058209 74...
sage: (pi-1/s).n(digits=100).n()
-4.36415445739398e-96
```

17. Dérivées

17.1. Dériver une fonction d'une variable :. Utiliser la fonction `derivative` ou son alias (équivalent) `diff`

```
sage: diff(sin(x^2), x)
2*x*cos(x^2)
```

17.2. Dérivée d'une fonction composée :.

```
sage: function('f', x); function('g', x);
f(x)
g(x)
sage: diff(f(g(x)), x)
D[0](f)(g(x))*D[0](g)(x)
```

```
sage: diff(ln(f(x)), x)
D[0](f)(x)/f(x)
```

17.3. Dérivées partielles :. Utiliser aussi la fonction `derivative` ou `diff`

```
sage: f(x,y) = x*y + sin(x^2) + e^(-x); derivative(f, x)
(x, y) |--> 2*x*cos(x^2) + y - e^(-x)
sage: derivative(f,y)
(x, y) |--> x
```

Exercice 3. On dit qu'une fonction $f\colon \Omega \longrightarrow \mathbb{R}$ définie dans un ouvert $\Omega \subset \mathbb{R}^2$ muni de coordonnées réelles $(x,y) \in \Omega$ est *harmonique* lorsque son *laplacien* s'annule :

$$0 = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}.$$

(a) Pour la fonction :

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

définie sur l'ouvert $\Omega = \mathbb{R}^2\backslash\{(0,0)\}$, calculer à la main ces deux dérivées partielles et vérifier que leur somme s'annule identiquement.

(b) En trois lignes de Sage, exécuter ce calcul.

Exercice 4. Soit la fonction de $\mathbb{R}^2$ à valeurs dans $\mathbb{R}$ définie par :

$$f(x,y) = \begin{cases} xy\,\dfrac{x^2-y^2}{x^2+y^2} & \text{lorsque } (x,y) \neq (0,0), \\ 0 & \text{lorsque } (x,y) = (0,0). \end{cases}$$

Montrer, d'abord à la main et ensuite sous Sage, que les dérivées partielles de f par rapport à x et à y ne commutent *pas* à l'origine :

$$\frac{\partial}{\partial y}\left(\frac{\partial}{\partial x} f\,\Big|_{x=0}\right)\Big|_{y=0} \neq \frac{\partial}{\partial x}\left(\frac{\partial}{\partial y} f\,\Big|_{y=0}\right)\Big|_{x=0}.$$

18. Intégration

18.1. Trouver une primitive et calculer une intégrale définie :. Utiliser la commande `integrate`, ou son alias (équivalent) `integral`

```
sage: sin(x).integral(x, 0, pi/2)
1
```

```
sage: integrate(1/(1+x^2), x)
arctan(x)
```

```
sage: integrate(1/(1+x^2), x, -infinity, infinity)
pi
```

```
sage: integrate(exp(-x^2), x, 0, infinity)
1/2*sqrt(pi)
```

18.2. Calculer une intégrale dépendant d'un paramètre :. Soit à calculer, pour un paramètre réel $x > 0$, l'intégrale définie :

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

• Utiliser la fonction `assume`

```
sage: u = var('u'); f = x*cos(u)/(u^2+x^2)
sage: assume(x>0); f.integrate(u, 0, infinity)
1/2*pi*e^(-x)
```

18.3. Intégrer *numériquement* sur un intervalle numérique donné :. Utiliser la fonction `integral_numerical`, qui renvoie une valeur approchée de l'intégrale en premier argument et une estimation d'erreur en deuxième argument :

```
sage: integral_numerical(sin(x)/x, 0, 1)
(0.94608307036718287, 1.0503632079297086e-14)
```

```
sage: g = integrate(exp(-x^2), x, 0, infinity)
sage: g, g.n()
(1/2*sqrt(pi), 0.886226925452758)
```

```
sage: valeur-approchee = integral_numerical(exp(-x^2), 0, infinity)
sage: valeur-approchee
(0.88622692545275705, 1.7147744320162414e-08)
```

Exercice 5. [Formule de Bailey-Borwein-Plouffe] Est-il possible de calculer des décimales très éloignées de π sans avoir à calculer toutes les décimales qui précèdent ? Oui, grâce à une formule spécialement construite pour cela en 1995, qui a permis d'atteindre en 2001 par exemple le $4\,000\,000\,000\,000\,000^{\text{ème}}$ chiffre de π en base 2.

(a) Pour tout entier $N \geqslant 1$, on introduit :

$$S_N = \sum_{n=0}^{N} \left(\frac{4}{8n+1} - \frac{2}{8n+4} - \frac{1}{8n+5} - \frac{1}{8n+6} \right) \left(\frac{1}{16} \right)^n.$$

Montrer que $\lim\limits_{N\to\infty} S_N$ existe et vaut :

$$\lim_{N\to\infty} S_N = \int_0^{1/\sqrt{2}} \frac{f(t)}{1-t^8}\, dt.$$

(b) Montrer la formule de Bailey-Borwein-Plouffe :

$$\sum_{n=0}^{N} \left(\frac{4}{8n+1} - \frac{2}{8n+4} - \frac{1}{8n+5} - \frac{1}{8n+6} \right) \left(\frac{1}{16} \right)^n = \pi.$$

19. Tableau récapitulatif des fonctions utiles en Analyse

Fonctions et opérateurs	
Dérivation	`diff(f(x), x)`
Dérivée n-ième	`diff(f(x), x, n)`
Intégration	`integrate(f(x), x)`
Intégration numérique	`integral_numerical(f(x), a, b)`
Somme symbolique	`sum(f(i), i, imin, imax)`
Limite	`limit(f(x), x=a)`
Polynôme de Taylor	`taylor(f(x), x, a, n)`
Développement limité	`f.series(x==a, n)`
Tracé d'une courbe	`plot(f(x), x, a, b)`

RÉFÉRENCES

[1] Casamayou, A. ; Cohen, N. ; Connan, G. ; Dumont, T. ; Fousse, L. ; Maltey, F. ; Meulien, M. ; Mezzarobba, M. ; Pernet, C. ; Thiéry, N. ; Zimmermann, P. : *Calcul mathématique avec Sage*, Creative Commons, 2014, xii+460 pp.