Développements limités

I

Sans Python

1. L'application $g = [x \mapsto xe^{-x}]$ réalise une bijection de classe $\mathscr{C}^\infty$ de $[0,1]$ sur $[0,e^{-1}]$ et la bijection réciproque est de classe $\mathscr{C}^\infty$ sur $[0,e^{-1}[$.

1.1 Calculer le développement limité à l'ordre n au voisinage de 0 de g.

1.2 Au voisinage de $u = 0$,

$$g^{-1}(u) = u + u^2 + \frac{3}{2}u^3 + \frac{8}{3}u^4 + \frac{125}{24}u^5 + \frac{54}{5}u^6 + o(u^6).$$

2. Au voisinage de $x = 0$,

$$\tan x = x + \frac{x^3}{3} + \frac{2}{15}x^5 + \frac{17}{315}x^7 + \frac{62}{2835}x^9 + o(x^9).$$

(Comme quotient de $\sin x$ par $\cos x$ ou comme réciproque de Arctan.)

3.

$$\left(1 + \frac{1}{\sqrt{n}}\right)^n \underset{n\to+\infty}{\sim} \frac{e^{-\sqrt{n}}}{\sqrt{e}}$$

4. La fonction définie par

$$\forall\, 0 < x < 1, \quad \Phi(x) = \frac{1}{(1-x)^2}\left(1 + \frac{1-x}{\ell n\, x}\right)^2$$

peut être prolongée en une fonction continue sur $[0,1]$ en posant $\Phi(0) = 1$ et $\Phi(1) = 1/4$.
Par conséquent, la suite de terme général

$$S_n = \frac{1}{n}\sum_{k=2}^{n} \Phi(i/n)$$

converge.

II

Avec Python

5. On charge le module qui permet le calcul symbolique et on définit deux variables littérales : x pour les fonctions et n pour les suites.

```
from sympy import *

x, n = symbols("x n")
```

6. Les fonctions `limit` et `series` permettent de calculer la plupart des limites et des développements limités.

6.1 Calcul de limite

L'exécution de

```
limit(e, z, z0, dir='+')
```

calcule la limite de l'expression `e` lorsque la variable `z` tend vers la valeur de référence `z0`.
L'expression `e` peut contenir des expressions symboliques autres que `z` : elles sont alors traitées comme des constantes.
La valeur de référence `z0` peut être une expression numérique (0, 1...) ou symbolique : `sqrt(2)` pour $\sqrt{2}$, `pi/6` pour $\pi/6$, `oo` pour $+\infty$ et `-oo` pour $-\infty$.
L'option `dir` permet de calculer une limite à gauche (avec `dir='-'`) ou une limite à droite (avec `dir='+'`).

6.2 Calculs de développements

L'exécution de

```
series(expr, x=None, x0=0, n=6, dir='+')
```

calcule (lorsque c'est possible...) un développement limité ou asymptotique de l'expression `expr` lorsque la variable `x` tend vers la valeur de référence `x0`.
Par défaut, la variable est l'unique symbole qui figure dans l'expression `expr`.
Par défaut, la valeur de référence `x0` est nulle. C'est normal : pour calculer un développement limité, il faut d'abord définir un infiniment petit. Comme pour les calculs de limites, les valeurs prises par cet argument peuvent être numériques ou symboliques.
Le paramètre `n` donne l'ordre du développement. Par défaut, ce paramètre est égal à 6. Pour un développement limité normal, le résultat est donné à

$$\mathcal{O}\big((x - x_0)^n\big)$$

près. Pour un développement asymptotique, la précision du développement est moins facilement prévisible.
Pour les grandes occasions, on peut aussi calculer un développement limité à gauche ou à droite de la valeur de référence `x0` grâce à l'option `dir`.

TD 1

7. Exercice 7

7.1

```
f = (3*x**3+2*x-4)/(2*x**2-6*x+1)
DL = series(f, x, x0=oo, n=2)
# 55/(4*x) + 9/2 + 3*x/2 + O(x**(-2), (x, oo))
```

7.2

```
f = sqrt(x+4)-sqrt(x)
DL = series(f, x, x0=oo, n=2)
# 2*sqrt(1/x) - 2*(1/x)**(3/2) + O(x**(-2), (x, oo))
```

7.3

```
f = sqrt(3*x+2)-sqrt(2*x-4)
DL = series(f, x, x0=oo, n=2)
# (-sqrt(2) + sqrt(3))/sqrt(1/x)
# + (sqrt(3)/3 + sqrt(2))*sqrt(1/x)
# + (-sqrt(3)/18 + sqrt(2)/2)*(1/x)**(3/2)
# + O(x**(-2), (x, oo))
```

7.4

```
f = (sqrt(x)-2)/(x-4)
DL = series(f, x, x0=4, n=3)
# 5/16 + (x − 4)**2/512 − x/64 + O((x − 4)**3, (x, 4))
```

7.5

```
f = (x*sqrt(x)-27)/(x-9)
DL = series(f, x, x0=9, n=3)
# 27/8 − (x − 9)**2/432 + x/8 + O((x − 9)**3, (x, 9))
DL = series(f, x, x0=4, n=3)
# 79/25 − 11*(x − 4)**2/2000 + 4*x/25
# + O((x − 4)**3, (x, 4))
```

7.6

```
f = (cos(x)-sin(x))/(x-pi/4)
DL = series(f, x, x0=pi/4, n=3)
# −sqrt(2) + sqrt(2)*(x − pi/4)**2/6
# + O((x − pi/4)**3, (x, pi/4))
DL = series(f, x, x0=0, n=3)
# −4/pi + x*(−16/pi**2 + 4/pi)
# + x**2*(−64/pi**3 + 2/pi + 16/pi**2) + O(x**3)
```

8. Exercice 8

8.1

```
f = sqrt(x**2-x+3)
DL = series(f, x, x0=oo, n=2)
# 11/(8*x) − 1/2 + x + O(x**(−2), (x, oo))
```

8.2

```
f = 1/log(x**2+x+1)
DL = series(f, x, x0=oo, n=2)
# −1/(4*x*log(1/x)**2) − 1/(2*log(1/x))
# + O(x**(−2), (x, oo))
```

8.3

```
f = x*sqrt(2/(x+1))
DL = series(f, x, x0=oo, n=2)
# sqrt(2)/sqrt(1/x) − sqrt(2)*sqrt(1/x)/2
# + 3*sqrt(2)*(1/x)**(3/2)/8
# + O(x**(−2), (x, oo))
```

9. Exercice 9

```
f = log(x+sqrt(x**2+1))
DL = series(f, x, x0=oo, n=5)
# −3/(32*x**4) + 1/(4*x**2) + log(2) − log(1/x)
# + O(x**(−5), (x, oo))
```

10. Exercice 23

```
f = log((1-x)/(1+x))
DL = series(f, x, x0=0)
# −2*x − 2*x**3/3 − 2*x**5/5 + O(x**6)
```

TD 3

11. Exercice 3

```
f = x*log(x)/(x**2-1)
X0 = [ 0, 1, oo ]
DL = []
for x0 in X0:
DL.append(series(f, x, x0, 3))
# [ −x*log(x) − x**3*log(x) + O(x**3),
  #   1/2 − (x − 1)**2/12 + O((x − 1)**3, (x, 1)),
  #   −log(1/x)/x + O(log(1/x)/x**3, (x, oo)) ]
```

12. Exercice 4

12.1

```
f = log(1+x**2)/2-log(x)
DL = series(f, x, 0)
# −log(x) + x**2/2 − x**4/4 + O(x**6)
```

12.2

```
f = log(1+2*x**2)/log(2*x+3)
DL = series(f, x, 0, 4)
# 2*x**2/log(3) − 4*x**3/(3*log(3)**2) + O(x**4)
```

13. Exercice 6

13.1

```
f = log(1+3*x)/sin(5*x)
DL = series(f, x, 0, 4)
# 3/5 − 9*x/10 + 43*x**2/10 − 39*x**3/5 + O(x**4)
```

13.2

```
f = (exp(2*x)-1)/(sin(3*x))
DL = series(f, x, 0, 4)
# 2/3 + 2*x/3 + 13*x**2/9 + 11*x**3/9 + O(x**4)
```

13.3

```
f = ((1+x)/x)**sqrt(x)
DL = series(f, x, oo, 2)
# 1/(2*x) + 1 + sqrt(1/x) − (1/x)**(3/2)/3 + O(x**(−2), (x,
```

TD 9

14. Exercice 1

14.1

```
u = sqrt(n**2+n+1)-sqrt(n**2-n+1)
DL = series(u, n, oo, 5)
# 15/(128*n**4) − 3/(8*n**2) + 1 + O(n**(−5), (n, oo))
```

14.2

```
u = sqrt(n+sin(n))-sqrt(n)
# DL = series(u, n, oo, 3)
# PoleError: Cannot expand ...
```

TD 11

15. Exercice 8

```
f = x*log(x)/(x**2-1)
DL = series(f, x, 1, n=4)
# 15/(128*n**4) - 3/(8*n**2) + 1 + O(n**(-5), (n, oo))

g = x/(x**2-1)
DL = series(g, x, oo, n=6)
# x**(-5) + x**(-3) + 1/x + O(x**(-6), (x, oo))
```

16. Exercice 27

16.1

```
f = (3*x**2-5*x+1)/(log(x)**3+x**2-5)
DL = series(f, x, oo, n=4)
# -5*log(1/x)**3/x**3 + (3*log(1/x)**3 + 1)/x**2
# - 5/x + 3 + O(x**(-4), (x, oo))
```

16.2

```
f = sin(x*log(x))/x
# DL = series(f, x, 0, n=5)
# PoleError: Cannot expand...
```

16.3

```
f = 2/sin(x)**2-1/(1-cos(x))
DL = series(f, x, 0, 5)
# 1/2 + x**2/8 + x**4/48 + O(x**5)
```

16.4

```
f = (x**3+3*x)/(x**4-5*x**2)
DL = series(f, x, 0, 6)
# -3/(5*x) - 8*x/25 - 8*x**3/125 - 8*x**5/625
# + O(x**6)
```

16.5

```
f = log(1+2*x**2)/(3*x**2+5*x)
DL = series(f, x, 0, 5)
# 2*x/5 - 6*x**2/25 - 32*x**3/125 + 96*x**4/625
# + O(x**5)
```

16.6

```
f = sin(3*x)*tan(2*x)/(x*(exp(2*x)-1))
DL = series(f, x, 0, 5)
# 3 - 3*x + x**2/2 + x**3/2 + 263*x**4/120 + O(x**5)
```

16.7

```
f = (log(1+x))**2-log(x)**2
DL = series(f, x, 0, 5)
#-log(x)**2 + x**2 - x**3 + 11*x**4/12 + O(x**5)
DL = series(f, x, oo, 4)
# (-2/(3*log(1/x)) - 1/log(1/x)**2)*log(1/x)**2/x**3
# + (1/log(1/x) + log(1/x)**(-2))*log(1/x)**2/x**2
# - 2*log(1/x)/x + O(x**(-4), (x, oo))
```

16.8

```
f = sin(2*x)/sqrt(1-cos(x))
DL = series(f, x, 0, 5)
# 2*sqrt(2) - 5*sqrt(2)*x**2/4 + 41*sqrt(2)*x**4/192
# + O(x**5)
```

16.9

```
f = cos(3*x)/(1-2*sin(x))
DL = series(f, x, pi/6, 4)
# sqrt(3) - pi/12 - 5*sqrt(3)*(x - pi/6)**2/4
# - 7*(x - pi/6)**3/12 + x/2
# + O((x - pi/6)**4, (x, pi/6))
```

16.10

```
f = ((1+x)**(1/3)-1)/(sqrt(1-x)-1)
DL = series(f, x, 0, 4)
# -0.666667 + 0.388889*x - 0.137345*x**2
# + 0.120113*x**3 + O(x**4)
```

16.11

```
f = log(sin(x)**2)/(x-pi/2)
DL = series(f, x, pi/2, 7)
# pi/2 - (x - pi/2)**3/6 - 2*(x - pi/2)**5/45 - x
# + O((x - pi/2)**7, (x, pi/2))
```

TD 18

17. Exercice 2

17.1

```
f = 1/x
DL = series(f, x, 3, 4)
# 2/3 + (x - 3)**2/27 - (x - 3)**3/81 - x/9
# + O((x - 3)**4, (x, 3))
```

17.2

```
f = sqrt(x)
DL = series(f, x, 4, 3)
# 1 - (x - 4)**2/64 + x/4 + O((x - 4)**3, (x, 4))
```

17.3

```
f = log(x)/x
DL = series(f, x, 2, 3)
# log(2)/2 + (1/4 - log(2)/4)*(x - 2)
# + (-3/16 + log(2)/8)*(x - 2)**2
# + O((x - 2)**3, (x, 2))
```

17.4

```
f = sin(x)**x
DL = series(f, x, pi/2, 4)
# 1 - pi*(x - pi/2)**2/4 - (x - pi/2)**3/2
# + O((x - pi/2)**4, (x, pi/2))
```

18. Exercice 6

18.1

```
f = 1/x-1/sin(x)
DL = series(f, x, 0, 4)
# -x/6 - 7*x**3/360 + O(x**4)
```

18.2

```
f = log(1+sin(x))-sin(log(1+x))
DL = series(f, x, 0, 7)
# -x**4/12 + x**5/8 - 53*x**6/360 + O(x**7)
```

Mon pire cauchemar

19.

```
f = sin(tan(x))-tan(sin(x))
DL = series(f, x, 0, 8)
# -x**7/30 + O(x**8)
# quelques ingrédients
g = tan(x)
# x + x**3/3 + 2*x**5/15 + 17*x**7/315 + O(x**8)
h = sin(tan(x))
# x + x**3/6 - x**5/40 - 55*x**7/1008 + O(x**8)
```