Analyse complexe

TP 2 : Série entière

I. Généralités

On appelle série entière toute série de fonctions $\sum U_n$ pour laquelle il existe une suite (a_n) telle que :

$$\forall n \in \mathbb{N} \quad \forall z \in \mathbb{C} \quad u_n(z) = a_n z^n$$

Une telle série entière est notée $\sum a_n z^n$, appelée **série entière** ; les a_n sont les coefficients de la série entière.

II. Exercices

Exercice 1 :

Soit les deux séries suivantes :

F(x) = $\sum_{n=1}^{+\infty} \frac{x^n}{2n+1}$

G(x) = $\sum_{n=1}^{+\infty} \frac{x^n}{n}$

1- Ecrire un programme python qui calcule la somme de : F(n)+ G(n)

2- Ecrire un programme python qui calcule la somme de la série :

$$K(x) = \sum_{n=1}^{+\infty} \left(\frac{1}{n} + \frac{1}{2n+1}\right) x^n$$

3- Conclure.

Exercice 2 :

On pose la série entière suivante :

$$\sum \frac{x^{3n}}{(3n)!}$$

a. On note f(x) la somme de cette série. À l'aide de Python, donner une valeur approchée de f(1) puis tracer le graphe de f.

III. Les nombres complexes avec python

Un nombre complexe z = a +ib s'écrit avec PYTHON sous la forme a+bj ou complex(a,b).
Attention : le nombre complexe i se note complex(0,1) ou 0+1j ou plus simplement 1j.
Le module *cmath* est similaire au module *math*, mais définit les fonctions de manière appropriée pour le plan complexe.

- Tester dans une console les commandes suivantes :

```
z1=2+1j
z2=complex(1,2)
z1+z2
z1*z2
z1/z2
z1**2
abs(z1)
z1.real
z1.imag
z1.conjugate()
```

Pour la plupart des fonctions, nous avons besoin du module, par exemple **sqrt**

```
import cmath

cmath.sqrt(-1)
# Out: 1j
```

Naturellement, le comportement de **sqrt** est différent pour les nombres complexes et les nombres réels. Dans les *math* non complexes, la racine carrée d'un nombre négatif déclenche une exception :

```
import math

math.sqrt(-1)
# Exception: ValueError: math domain error
```

Exercice :

1- Définir les nombres complexes z1 = 1 + 2i et z2 = 3 – i et calculer :

z1 + z2
z1 z2
|z1|
1/ z1

2- Définir le nombre complexe $z = (3 - 4i)^2 (2 + i)$. Calculer la partie réelle de z, sa partie imaginaire et son conjugué.

IV. Calcul de la limite d'une fonction avec python

SymPy est une librairie Python de mathématiques symboliques et un outil de calcul formel. SymPy offre un ensemble riche de fonctions documentées qui facilite la modélisation de problèmes mathématiques (arithmétique, calcul symbolique, résolution d'équations, recherche de racines, dessin de fonctions, limites et séries, calcul différentiel et intégral, algèbre linéaire). La librairie offre aussi des fonctionnalités de mathématiques plus avancées (géométrie différentielle et algébrique, intégration numérique, théorie des catégories) et pour la physique (optique, mécanique classique, mécanique et informatique quantique).

A l'aide de la méthode sympy.limit (), nous pouvons trouver la limite de toute expression mathématique.

$$\lim_{x \to a} f(x)$$

Syntaxe : limit(expression, variable, valeur)

Paramèters:

- **Expression –** L'expression mathématique sur laquelle l'opération limite doit être effectuée, c.à.d., f (x).
- **Variable –** C'est la variable dans l'expression mathématique, c.à.d., x
- **Valeur–** C'est la valeur à laquelle la limite tend à, c.à.d., a.
- **Résultat :** renvoie la limite de l'expression mathématique dans des conditions données.

Exemple :

```
# import sympy
from sympy import *

x = symbols('x')
expr = sin(x)/x;

print("Expression : {}".format(expr))

# Use sympy.limit() method
limit_expr = limit(expr, x, 0)

print("Limit of the expression tends to 0 : {}".format(limit_expr))
```

```
Expression : sin(x)/x
Limit of the expression tends to 0 : 1
```

Pour la limite en l'infini :

```
>>> limit(x, x, oo)
oo

>>> limit(1/x, x, oo)
0
```

Exercice

Calculer les limites suivantes :

1. Lim $x \to 0$ $f(x) = (\sin(x) - x) / x^3$
2. Lim $x \to 5/2$ $f(x) = 2x + 1$
3. Lim $x \to +\infty$ $f(x) = 1/(5 + 2x)^2$
4. Lim $x \to +\infty$ $f(x) = x^3+2x^2-1$