# Calcul formel sur des solutions d'équations différentielles linéaires

Encadrant : Marc Mezzarobba, LIP6 <marc.mezzarobba@lip6.fr>

Une fonction $y$ (par exemple une fonction $y\colon \mathbb{R} \to \mathbb{R}$ de classe $\mathcal{C}^\infty$, une fonction holomorphe $y\colon \mathbb{C} \to \mathbb{C}$, ou même une série formelle $y(z) = y_0 + y_1\, z + y_2\, z^2 + \cdots$) est appelée *différentiellement finie* si elle est solution d'une équation différentielle linéaire homogène dont les coefficients sont des polynômes en la variable de temps $z$ :

$$p_r(z)\, y^{(r)}(z) + \cdots + p_1(z)\, y'(z) + p_0(z)\, y(z) = 0.$$

Cette équation accompagnée de conditions initiales $y(z_0)$, $y'(z_0)$, ..., $y^{(r-1)}(z_0)$ en un point $z_0$ convenablement choisi définit complètement la fonction et permet de calculer sa valeur en n'importe quel point. Ainsi, bien qu'une fonction différentiellement finie soit définie en une infinité de points, il est possible de la représenter exactement sur ordinateur : il suffit de donner l'équation différentielle et les conditions initiales.

Par exemple, la fonction exponentielle est différentiellement finie puisqu'elle vérifie l'équation $\exp'(z) = \exp(z)$ pour tout $z \in \mathbb{C}$ et la condition initiale $\exp(0) = 1$. De même, la fonction $w(z) = \arctan(z)$ est différentiellement finie puisqu'elle satisfait

$$(z^2+1)\, w''(z) + 2\, z\, w'(z) = 0, \qquad w(0) = 0, \quad w'(0) = 0.$$

De nombreuses autres fonctions usuelles, ou fonctions moins usuelles mais utiles dans des applications spécifiques, sont elles aussi différentiellement finies.

Les fonctions différentiellement finies forment un anneau : on peut montrer que la somme $f + g$ et le produit $f\,g$ de deux fonctions différentiellement finies $f$ et $g$ sont encore des fonctions différentiellement finies. Par exemple, la fonction $y(z) = \exp(z) + \arctan(z)$ vérifie

$$(z+1)\, y'''(z) - (z-1)\,(z^2-3)\, y''(z) - 2\,(z^2+2\ z-1)\, y'(z) = 0, \quad y(0) = 1, y'(0) = 2, y''(0) = 1.$$

À partir de représentations de $f$ et $g$ par équation différentielle et conditions initiales, on peut calculer automatiquement une équation pour la somme ou le produit et les conditions initiales correspondantes. La même chose est vraie pour d'autres opérations plus compliquées.

L'objet de ce projet est de se servir de cette représentation pour faire des calculs sur les fonctions différentiellement finies dans SageMath (aussi simplement appelé Sage), un système de calcul formel libre écrit en langage Python utilisé par des milliers de mathématiciens. Vous devrez dans un premier temps vous familiariser avec les propriétés mathématiques des fonctions différentiellement finies et avec la bibliothèque Sage. Vous définirez ensuite à Sage un type d'objet « fonction différentiellement finie » qui devra rendre aussi naturels que possible pour l'utilisateur les calculs avec ces fonctions. Par exemple, si `u` et `v` sont des objets de type « fonction différentiellement finie », l'expression `u + v` devra calculer un objet « fonction différentiellement finie » représentant la somme des deux fonctions, `u(1)` devra calculer la valeur en 1 de $u$, et ainsi de suite. Si le temps disponible le permet, vous ajouterez ensuite des conversions d'autres types d'objets Sage (par exemple la fonction arctangente prédéfinie) en objets de type fonction différentiellement finie.

Vous vous appuierez pour cela sur la bibliothèque Sage `ore_algebra`, qui fournit des implémentations des algorithmes nécessaires sur les équations différentielles, mais sans suivre les conditions initiales. Suivant son état d'avancement, le code développé pourra être soumis pour inclusion dans `ore_algebra` à la fin du projet.

Ce projet demande une bonne connaissance du langage Python. Des connaissances de base en analyse complexe seraient un plus.

## Bibliographie

**[1]** Alexandre Casamayou, Nathann Cohen, Guillaume Connan, Thierry Dumont, Laurent Fousse, François Maltey, Matthias Meulien, Marc Mezzarobba, Clément Pernet, Nicolas M. Thiéry, and Paul Zimmermann. *Calcul mathématique avec Sage*. 2013. URL: http://sagebook.gforge.inria.fr/.

**[2]** Manuel Kauers, Maximilian Jaroschek, and Fredrik Johansson. `ore_algebra`, 2013–. URL: `http://kauers.de/software.html`.

**[3]** Manuel Kauers, Maximilian Jaroschek, and Fredrik Johansson. Ore polynomials in Sage. In Josef Schicho Gutierrez, Jaime and Martin Weimann, editors, *Computer Algebra and Polynomials*, pages 105–125. Springer, 2015. URL: `http://www.algebra.uni-linz.ac.at/people/mkauers/publications/kauers14b.pdf`.

**[4]** Manuel Kauers and Peter Paule. *The Concrete Tetrahedron: Symbolic Sums, Recurrence Equations, Generating Functions, Asymptotic Estimates*. Springer, 2011.

**[5]** Richard P. Stanley. *Enumerative combinatorics*, volume 2. Cambridge University Press, 1999.

**[6]** The SageMath Developers. SageMath mathematics software, 2005-. URL: `http://sagemath.org/`.