PROJET DE THÈSE :
ALGORITHMIQUE DE BASE DES FONCTIONS D-FINIES, APPLICATIONS À L'ÉVALUATION GARANTIE

1. Présentation du domaine

Le calcul formel est l'étude des mathématiques effectives et de leur complexité. Cette discipline à l'interface des mathématiques et de l'informatique touche plusieurs millions d'utilisateurs par le biais de systèmes grand public comme Maple ou Mathematica. Le sujet de thèse développé ici est à la confluence de trois grandes directions de recherche actuelles en calcul formel :

- considérer les équations différentielles et les récurrences non plus comme des objets à résoudre, mais comme des structures de données représentant leur solutions (« D-finitude ») ;
- rechercher des algorithmes quasi-optimaux en utilisant l'analyse de complexité comme outil de conception d'algorithmes ;
- développer la complémentarité entre calcul numérique et calcul symbolique en exploitant les capacités d'approximation en précision arbitraire de ce dernier (« symbolique-numérique »).

1.1. **D-finitude.** Historiquement, d'importants efforts de recherche en calcul formel on été consacrés à donner des solutions *en forme close* de divers problèmes. Un exemple spectaculaire est le calcul de primitives. Les premiers logiciels mimaient les méthodes de calcul employées à la main. Puis l'algorithme de Risch, fondé sur des théorèmes de structure en algèbre différentielle, a fourni une solution générale (un algorithme de décision) pour les fonctions élémentaires. Plus largement, de nombreux travaux, notamment ceux de Singer, ont étendu la classe des équations différentielles que les logiciels de calcul formel permettent de résoudre explicitement en termes de fonctions liouvilliennes. Cependant, l'existence de telles solutions est l'exception plutôt que la règle, ce qui réduit l'application de ces algorithmes à des questions de décision, ou à l'initiation aux équations différentielles dans l'enseignement.

Ce projet de thèse s'inscrit dans un effort de développement d'une algorithmique *générale* et *efficace* sur les solutions d'équations différentielles linéaires à coefficients polynomiaux (fonctions dites différentiellement finies ou D-finies) *représentées implicitement*, par une équation différentielle et des conditions initiales. Une série est D-finie si et seulement si la suite de ses coefficients satisfait une relation de récurrence linéaire, également à coefficients polynomiaux (on dit qu'une telle suite est P-récursive) ; et là encore, celle-ci est une structure de données adéquate pour manipuler la suite.

Les principales propriétés mathématiques des fonctions D-finies sont connues depuis le XIX^e siècle. Leur développement moderne en combinatoire et en calcul formel est dû en grande partie à Stanley [Sta80] puis Zeilberger [Zei90], qui ont souligné leur intérêt comme classe de fonctions close par plusieurs opérations importantes et bénéficiant d'une algorithmique riche. L'étude systématique, du point de vue algorithmique, des fonctions D-finies et des suites P-récursives s'est poursuivie depuis. Sur le plan pratique, des logiciels comme le module Maple `gfun` [SZ94] fournissent déjà un grand nombre d'opérations importantes sur cette structure de données.

La notion de D-finitude admet des généralisations fructueuses à plusieurs variables [Lip89], et à des systèmes mixtes mélangeant dérivées partielles, récurrences multiples, q-analogues, etc. Les algorithmes associés permettent de manipuler des expressions complexes dans les systèmes de calcul formel, et notamment de prouver, voire de calculer, des identités comme

$$\frac{1}{2}J_0(z)^2 + \sum_{\nu=1}^{\infty} J_\nu(z)^2 = \frac{1}{2}, \tag{Ex}$$
$$\int_0^{+\infty} xJ_1(ax)I_1(ax)Y_0(x)K_0(x)\,dx = -\frac{\ln(1-a^4)}{2\pi a^2},$$

(où les J_ν, I_ν, Y_ν et K_ν sont des fonctions de Bessel) entre sommes et intégrales de solutions de systèmes D-finis [CS98, Chy98].

1.2. **Complexité.** La complexité des opérations sur les objets de base du calcul formel que sont les entiers ou les séries formelles est étudiée depuis longtemps comme en témoigne le volume 2 de *The Art of Computer Programming* [Knu97]. L'intérêt pour ces questions a connu un regain ces dernières années grâce à l'arrivée de machines de bureau permettant d'attaquer des problèmes de grande taille. Il s'avère que pour ces tailles, d'une part les analyses asymptotiques de complexité permettent des prédictions fiables sur les temps de calcul, et d'autre part les algorithmes rapides à base de FFT deviennent utilisables en pratique.

Or l'utilisation d'équations différentielles et de récurrences comme structures de données mène naturellement à la production d'équations d'ordre et de degré élevés. C'est notamment le cas pour l'opération de *création télescopique* qui est au cœur du calcul d'identités comme (Ex) — en effet, l'élimination non-commutative sous-jacente subit des contraintes de dimensions similaires à celles de l'élimination dans les systèmes polynomiaux. Des problèmes de grande taille proviennent parfois aussi des utilisations en modélisation. Ainsi, H. Wilf nous posait récemment la question du calcul d'un terme d'indice environ 80000 d'une suite définie par une récurrence d'ordre à peu près 300.

Il faut donc revisiter une bonne partie des algorithmes du calcul formel et comprendre comment les faire bénéficier des opérations asymptotiquement rapides. Au-delà des entiers et des séries, on voit ainsi apparaître de nouveaux algorithmes de complexité *quasi-optimale* (c'est-à-dire optimale à des facteurs logarithmiques près par rapport à la taille de l'entrée et de la sortie) pour une grande variété de problèmes. Dans le cadre de la D-finitude, des progrès importants ont été réalisés (voir en particulier [Bos03]), mais de nombreux problèmes sont encore en quête d'un algorithme quasi-optimal ; quelques-uns d'entre eux seront décrits ci-dessous et feront l'objet d'une étude dans cette thèse.

1.3. **Symbolique-numérique.** De nombreuses fonctions spéciales classiques apparaissant en physique mathématique ou en mathématiques appliquées sont D-finies. Cependant, pour leur évaluation numérique, les développeurs de logiciels de calcul formel comme Maple, Mathematica ou Maxima agissent de manière schizophrène et les traitent pratiquement une par une, à l'aide de code *ad hoc*, en oubliant qu'ils disposent de capacités symboliques. Cela multiplie les problèmes d'efficacité, de correction et de maintenabilité des implémentations.

2. Sujet de thèse

Cette thèse vise à concevoir et valider des algorithmes efficaces et si possible quasi-optimaux dans le domaine de la D-finitude. Pour focaliser le choix des problèmes à étudier, deux objectifs de long terme serviront de guide :

- évaluer numériquement les fonctions D-finies de façon *générale*, *efficace* et *garantie* ;
- évaluer numériquement une intégrale simple ou multiple à paramètres, dont l'intégrande est défini par une équation ou un système d'équations différentielles linéaires.

Ce second but est largement inaccessible pour l'instant : grossièrement, les méthodes symboliques ne permettent de traiter que des cas très particuliers, et la grande dimension rend les méthodes de quadrature numérique inefficaces, à plus forte raison en présence de singularités.

Outre ces objectifs, des applications naturelles de ce travail sont le calcul explicite de matrices de monodromie ou de matrices de Stokes — problème d'actualité dans la communauté qui étudie la resommation des séries divergentes, comme en témoigne la thèse récente [Rem07].

La suite de ce document présente plus en détail les question algorithmiques et d'implémentation qui feront l'objet de cette thèse.

3. Algorithmique

Le principal résultat théorique de complexité sur l'évaluation des fonctions D-finies concerne la grande précision, et remonte aux travaux des frères Chudnovsky.

Théorème ([CC90, vdH99]). *Soit f une fonction D-finie, et soit U un domaine de sa surface de Riemann sur lequel f est bornée*[1]*. On peut calculer à précision 10^{-n} la valeur $f(z)$ que prend f en un point effectif $z \in U$ en $O\big(n(\log n)^3(\log\log n)^2\big)$ opérations binaires, uniformément en z.*

[1]Les résultats de [vdH01, vdH07] permettent dans une large mesure de s'affranchir de cette hypothèse. Ces travaux montrent par ailleurs comment appliquer ce résultat à l'évaluation asymptotique (possibilité qui n'est que mentionnée dans [CC90]), et à la resommation.

La complexité obtenue est quasi-optimale vis-à-vis de la précision des calculs et du point d'évaluation, mais plusieurs questions naturelles restent ouvertes.

3.1. **Scindage binaire.** L'algorithme précédent repose sur une méthode rapide, appelée *scindage binaire*, pour calculer un terme d'une suite récurrente. Soit $L \in \mathbb{Q}[n]\langle S\rangle$ un opérateur aux différences linéaire d'ordre s à coefficients polynomiaux, et soit $N \in \mathbb{N}$. Soient d le degré en n de L, et h une borne sur les tailles en bits de ses coefficients.

THÉORÈME ([CC90, th. 2.2]). *Notons $u = (u_n)_{n\in\mathbb{N}}$ une suite (disons de rationnels) telle que $L \cdot u = 0$. On peut calculer la matrice de l'application $(u_0, \dots, u_{r-1}) \mapsto (u_N, \dots, u_{N+r-1})$ en $O\big(s^\omega(d+h)N(\log N)^3 \log\log N\big)$ opérations binaires (où ω désigne l'exposant du produit matriciel).*

En particulier, on peut calculer un terme u_N d'une suite donnée par une récurrence linéaire à coefficients polynomiaux en $O\big(N(\log N)^3 \log\log N\big)$ opérations, ce qui est quasi-optimal vis-à-vis de N. Mais il n'en va pas de même de l'ordre s de la récurrence : sa mauvaise dépendance en s suffit à rendre le scindage binaire inutilisable quand celui-ci dépasse quelques dizaines. L'algorithme d'évaluation des fonctions D-finies hérite naturellement de cette restriction.

QUESTION. *Soit (u_n) une suite d'entiers donnée par une relation récurrence linéaire d'ordre s à coefficients polynomiaux et des conditions initiales $u_0, \dots, u_{s-1}$. Comment calculer efficacement un terme u_N de (u_n) quand N et s sont grands ? Par exemple, est-il possible de calculer u_N en temps $\tilde{O}(s^2N)$? $\tilde{O}(sN)$?*

Il s'agit là d'un problème ouvert *a priori* difficile, dont la solution aurait des répercussions sur la complexité de nombreux algorithmes. Plus modestement, diminuer significativement la constante de la complexité du scindage binaire représenterait déjà un gain appréciable.

Une piste dans ce sens est fournie par les travaux récents de Cheng *et al.* [CHT+07] qui montrent comment éviter l'apparition de « gros » facteurs communs entre numérateurs et dénominateurs des rationnels calculés lors de l'évaluation par scindage binaire de certaines fonctions hypergéométriques. Les auteurs donnent également des arguments suggérant qu'il n'est pas possible de faire de même pour toute la classe des fonctions hypergéométriques.

QUESTION. *Ces résultats se généralisent-ils aux fonctions D-finies ?*

3.2. **Produit de matrices.** L'opération de base du scindage binaire est le produit de matrices d'ordre modéré dont les coefficients sont de (très) grands entiers. Les algorithmes asymptotiquement rapides de produit matriciel ne sont guère pertinents dans ce contexte, mais toute diminution du nombre de multiplications scalaires nécessaires pour calculer un produit matriciel se répercute sur la complexité du scindage binaire. Si le nombre précis de multiplications nécessaire au produit matriciel sur un anneau quelconque a été étudié assez systématiquement, les résultats dans le cas où l'anneau de base est supposé commutatif sont plus rares. Ainsi, pour le produit de matrices carrées à coefficients commutatifs de taille 4 à 10, l'algorithme de Waksman [Wak70] pour demande strictement moins de multiplications que le meilleur algorithme n'utilisant pas la commutativité que nous ayons recensé, et est apparemment le meilleur connu. En taille 2, le nombre minimal de multiplication possible est 7 dans les deux cas ; en taille 3, le meilleur algorithme connu demande 23 multiplications sur un anneau général et 22 sur un anneau non commutatif [Mak86].

QUESTION. *Soit $\mathbb{A}$ un anneau commutatif. Pour $s = 2, 3, 4, \dots$, combien de multiplications d'éléments non constants de $\mathbb{A}$ nécessite le produit de deux matrices de $\mathbb{A}^{s\times s}$ par un algorithme bilinéaire (ou quadratique) ?*

Là encore, il s'agit d'une question *a priori* difficile, et dont la portée dépasse le seul cadre des fonctions D-finies.

3.3. **Taille des équations.** Comme les équations sont souvent elles-mêmes résultats d'un calcul, il est parfois possible d'améliorer l'efficacité des algorithmes ultérieurs en revisitant les algorithmes en amont. Ceux-ci ont souvent été conçus pour produire les opérateurs d'ordre minimal, ce qui n'est pas nécessairement le bon critère à optimiser pour faciliter la suite des calculs. En particulier, les opérateurs d'ordre minimal sont souvent encombrés de *singularités apparentes*, qui augmentent considérablement le degré de leurs coefficients. Il est apparu récemment des gains d'ordre de grandeur en s'affranchissant de la minimalité [BCL+07].

Question. *Comment effectuer la création télescopique en minimisant non pas l'ordre mais la taille totale de la sortie ?*

3.4. **Évaluation multi-points.** Il est classique que l'évaluation d'un polynôme sur de nombreux points peut tirer parti du fait que le même polynôme est considéré à chaque point, de sorte que l'efficacité gagne un facteur de l'ordre du degré du polynôme.

Question. *Comment calculer numériquement les valeurs d'une fonction D-finie d'une ou deux variables réelles ou complexes définie par une ou un système d'équations différentielles linéaires en de nombreux points plus efficacement qu'en l'évaluant séparément en chacun ?*

L'intérêt de cette question pour le tracé de graphes ou l'évaluation d'intégrales est évident.

3.5. **Précision et efficacité.** Les premières expériences sur une implémentation générale de la méthode des Chudnovsky semblent montrer que celle-ci devient compétitive, pour les fonctions usuelles, à partir de précisions allant de 10^{-1000} à $10^{-100000}$. Mais les problèmes d'évaluation des fonctions D-finies ne se limitent pas à la grande précision. Pour des précisions modérées et des points réels, on peut faire appel à des séries de Chebyshev qui peuvent elles aussi être calculées à partir des équations différentielles, pour des points proches des singularités, on peut parfois faire appel à des techniques de resommation de séries divergentes...

Question. *Comment détecter automatiquement et à faible coût, pour une fonction, un point et une précision, l'algorithme à choisir pour l'évaluation ?*

Par ailleurs, le contrôle de précision dans les travaux sur l'évaluation à grande précision est fait en valeur absolue. Une seconde question est donc d'adapter les méthodes de scindage binaire à la précision relative, et par là à l'évaluation en virgule flottante.

3.6. **Bornes.** Les algorithmes reposant sur l'évaluation de séries convergentes requièrent des estimations de bornes sur les restes des séries. Une manipulation assez brutale de séries majorantes à la Cauchy-Kovalevskaya permet d'obtenir les bons ordres de grandeurs pour la complexité dans une grande variété de situations [vdH03]. Il est cependant utile de disposer de bonnes bornes, qui font gagner des facteurs constants mais significatifs sur les temps de calcul, et peuvent par ailleurs trouver d'autres applications en combinatoire ou théorie des nombres.

Question. *Comment calculer des bornes qui approchent au plus près le comportement asymptotique des fonctions D-finies, y compris développées en des points singuliers ? La même idée s'adapte-t-elle aux suites P-récursives (notamment celles issues de la combinatoire) ?*

Nous avons déjà commencé à développer cette idée [Mez07] et obtenu des bornes ayant le bon ordre de croissance exponentiel sur les développements en série des fonctions D-finies en les points ordinaires.

4. Implémentation

Le développement d'implémentations efficaces des algorithmes étudiés constitue un aspect important de ce sujet. Il faudra s'efforcer de ne pas garder le code écrit à l'état de prototype, mais au contraire de le rendre accessible aux applications et comme sous-routines pour des développements ultérieurs.

Schématiquement, nous envisageons plusieurs types de débouchés pour les implémentations :

- les prototypes en Maple pourront être intégrées au module `gfun` développé au projet Algorithmes de l'Inria ;
- les développements dans un langage de plus bas niveau pourront fournir du code utilisable depuis le logiciel Mathemagix [vdHLMR]. Celui-ci, effort commun de plusieurs équipes de calcul formel françaises mené par Joris van der Hoeven à Orsay, dispose d'ores et déjà d'un embryon de bibliothèque consacrée aux fonctions D-finies ;
- pour ce qui concerne plus particulièrement l'évaluation à grande précision des fonctions spéciales usuelles, un effort sera fait pour produire du code utilisant la bibliothèque de multiprécision MPFR [FHL+07] développée aux projets Cacao et Arenaire de l'Inria ; il est possible à l'inverse qu'une partie du code développé puisse être utilisé par MPFR ;
- une partie des routines numériques pourra servir de moteur numérique au *Dynamic Dictionary of Mathematical Functions*, un « ouvrage de référence » interactif sur les fonctions

spéciales, généré automatiquement, développé depuis peu dans le cadre du laboratoire joint Inria-Microsoft.

Le développement logiciel suivra le modèle du logiciel libre, ce qui devrait lui faciliter visibilité et pérennité.

Références

[BCL+07] Alin Bostan, Frédéric Chyzak, Grégoire Lecerf, Bruno Salvy, and Éric Schost. *Differential equations for algebraic functions.* In C. W. Brown (editor): *ISSAC 2007: Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation*, 2007.

[Bos03] Alin Bostan. *Algorithmique efficace pour des opérations de base en calcul formel.* Thèse de doctorat, École polytechnique, décembre 2003.

[CC90] David V. Chudnovsky and Gregory V. Chudnovsky. *Computer algebra in the service of mathematical physics and number theory.* In *Computers in mathematics (Stanford, CA, 1986)*, page 109–232, New York, 1990. Dekker.

[CHT+07] Howard Cheng, Guillaume Hanrot, Emmanuel Thomé, Eugene Zima, and Paul Zimmermann. *Time- and space-efficient evaluation of some hypergeometric constants*, 2007.

[Chy98] Frédéric Chyzak. *Groebner bases, symbolic summation and symbolic integration.* In B. Buchberger and F. Winkler (editors): *Groebner Bases and Applications (Proc. of the Conference 33 Years of Gröbner Bases)*, volume 251 of *London Mathematical Society Lecture Notes Series*, pages 32–60. Cambridge University Press, 1998.

[CS98] Frédéric Chyzak and Bruno Salvy. *Non-commutative elimination in Ore algebras proves multivariate holonomic identities.* Journal of Symbolic Computation, 26(2):187–227, August 1998.

[FHL+07] Laurent Fousse, Guillaume Hanrot, Vincent Lefèvre, Patrick Pélissier, and Paul Zimmermann. *MPFR: A multiple-precision binary floating-point library with correct rounding.* ACM Transactions on Mathematical Software, 33(2):13, June 2007. 15 pages.

[Knu97] Donald E. Knuth. *Seminumerical Algorithms*, volume 2 of *The Art of Computer Programming.* Addison-Wesley, third edition, 1997.

[Lip89] Leonard Lipshitz. *D-finite power series.* Journal of Algebra, 122(2):353–373, 1989.

[Mak86] Oleg M. Makarov. *An algorithm for multiplying 3x3 matrices.* U.S.S.R. Computational Mathematics and Mathematical Physics, 26(1):179–180, 1986.

[Mez07] Marc Mezzarobba. *Génération automatique de procédures numériques pour les fonctions D-finies.* Rapport de stage, Master parisien de recherche en informatique, 2007. 89 pages.

[Rem07] Pascal Remy. *Résurgence des systèmes différentiels linéaires et calcul des matrices de Stokes.* Thèse de Doctorat, École doctorale d'Angers, septembre 2007.

[Sta80] Richard P. Stanley. *Differentiably finite power series.* European Journal of Combinatorics, 1(2):175–188, 1980.

[SZ94] Bruno Salvy and Paul Zimmermann. *Gfun: a Maple package for the manipulation of generating and holonomic functions in one variable.* ACM Transactions on Mathematical Software, 20(2):163–177, 1994.

[vdH99] Joris van der Hoeven. *Fast evaluation of holonomic functions.* Theoretical Computer Science, 210(1):199–216, 1999. `http://www.math.u-psud.fr/~vdhoeven/Publs/1997/TCS.ps.gz`.

[vdH01] Joris van der Hoeven. *Fast evaluation of holonomic functions near and in regular singularities.* Journal of Symbolic Computation, 31(6):717–743, 2001. `http://www.math.u-psud.fr/~vdhoeven/Publs/2000/singhol.ps.gz`.

[vdH03] Joris van der Hoeven. *Majorants for formal power series.* Technical Report 2003-15, Université Paris-Sud, Orsay, France, 2003. `http://www.math.u-psud.fr/~vdhoeven/Publs/2003/maj.ps.gz`.

[vdH07] Joris van der Hoeven. *Efficient accelero-summation of holonomic functions.* Journal of Symbolic Computation, 2007. `http://www.math.u-psud.fr/~vdhoeven/Publs/2005/reshol.ps.gz`.

[vdHLMR] Joris van der Hoeven, Grégoire Lecerf, Bernard Mourrain, and Olivier Ruatta. *Mathemagix.* `http://mathemagix.org/`.

[Wak70] Abraham Waksman. *On Winograd's algorithm for inner products.* IEEE Transactions on Computers, C-19(4):360–361, 1970.

[Zei90] Doron Zeilberger. *A holonomic systems approach to special functions identities.* Journal of Computational and Applied Mathematics, 32(3):321–368, 1990.