SOMMES HYPERGÉOMÉTRIQUES : L'ALGORITHME DE GOSPER

1. Introduction

On s'intéresse aux sommes de la forme

$$s_n = \sum_{k=0}^{n-1} t_k. \tag{1}$$

où t_k est un terme hypergéométrique qui ne dépend pas de n. Cela signifie que le quotient de termes consécutifs

$$\frac{t_{k+1}}{t_k}$$

est une fraction rationnelle en k.

La borne supérieure de la somme $n-1$ dépend de l'indice n de s_n. On dit que la somme est *indéfinie*.

Nous voudrions si possible écrire s_n sous forme *close*, c'est-à-dire comme combinaison linéaire de r termes hypergéométriques, r étant un nombre constant, indépendant de n. On va en fait chercher à écrire s_n comme étant égal à un terme hypergéométrique, plus une constante.

On peut montrer que c'est en fait équivalent.

Nous avons : $s(n) - s(n-1) = t_n$. Donc, le problème est le suivant. **Etant donné un terme hypergéométrique t_n, existe-t-il un terme hypergéométrique z_n tel que**

$$z_{n+1} - z_n = t_n \ ? \tag{2}$$

C'est la question à laquelle répond l'algorithme de Gosper. Si nous pouvons trouver une telle suite z_n, alors nous obtiendrons immédiatement une forme simple pour (1) :

$$s_n = z_n - z_0,$$

c'est-à-dire un terme hypergéométrique plus une constante. Réciproquement, toute solution de (2) sera de la forme

$$z_n = z_0 + \sum_{k=0}^{n-1} t_k = s_n + c,$$

où $c = z_0$ est une constante.

Si la réponse est positive, alors s_n peut être écrit comme une somme d'un terme hypergéométrique plus une constante, et l'algorithme donne ce terme hypergéométrique. Dans ce cas, on dit que t_n est **Gosper-sommable**. Si par contre l'algorithme de Gosper donne une réponse négative, alors cela prouvera que (2) n'a pas de solution hypergéométrique.

2. De l'hypergéométrique au rationnel, puis au polynomial

Si z_n est un terme hypergéométrique qui satisfait (2), alors le quotient

$$\frac{z_n}{t_n} = \frac{z_n}{z_{n+1} - z_n} = \frac{1}{\frac{z_{n+1}}{z_n} - 1}$$

est clairement une fraction rationnelle de n. Soit

$$z_n = y(n)t_n,$$

où $y(n)$ est une fraction rationnelle en n (encore inconnue). Si l'on remplace z_n par $y(n)t_n$ dans (2), on trouve l'équation

$$r(n)y(n+1) - y(n) = 1, \tag{3}$$

où r est la fraction rationnelle

$$r(k) = \frac{t_{k+1}}{t_k}.$$

C'est une relation de récurrence du premier ordre avec coefficients rationnels. Le problème de trouver une solution hypergéométrique à (2) est réduit à trouver une solution rationnelle à (3). Gosper a trouvé un moyen ingénieux pour réduire ce problème à celui de trouver des solutions polynomiales à une autre relation de récurrence linéaire du premier ordre.

Supposons que nous puissions écrire

$$r(n) = \frac{a(n)}{b(n)}\frac{c(n+1)}{c(n)}, \tag{4}$$

où a, b et c sont des polynômes en n, et

$$\text{pgcd}(a(n), b(n+h)) = 1 \text{ pour tout entier naturel } h. \tag{5}$$

Nous verrons en fait au paragraphe suivant qu'une telle factorisation existe pour toute fraction rationnelle et indiquerons un algorithme pour le trouver. Suivant l'idée de Gosper, nous cherchons une solution rationnelle non nulle de (3) de la forme

$$y(n) = \frac{b(n-1)x(n)}{c(n)} \tag{6}$$

où $x(n)$ est une fraction rationnelle inconnue en n. Si l'on remplace r et y par leurs valeurs (données dans (4) et (6)) dans l'équation (3), on constate que $x(n)$ vérifie

$$a(n)x(n+1) - b(n-1)x(n) = c(n). \tag{7}$$

Et là, un miracle se produit.

Théorème 2.1. *Soient $a(n)$, $b(n)$ et $c(n)$ des polynômes en n tels que la condition (5) soit vérifiée. Si $x(n)$ est une fraction rationnelle en n satisfaisant (7), alors $x(n)$ est un polynôme en n.*

Trouver des solutions hypergéométriques à (2) revient donc à trouver des solutions des solutions polynomiales à (7). Le lien entre eux est que si x est une solution polynomiale de (7), alors

$$z_n = \frac{b(n-1)x(n)}{c(n)} t_n$$

est une solution hypergéométrique de (2) et vice versa. La question de savoir comment trouver une solution polynomiale à l'équation (7) est l'objet du paragraphe (4).

En résumé, voici l'algorithme de Gosper.

Algorithme de Gosper

ENTREE : Un terme hypergéométrique t_n.

SORTIE : Un terme hypergéométrique z_n satisfaisant (1), s'il en existe ; $\sum_{k=0}^{n-1} t_k$ sinon.

1. Former le quotient $r(n) = t_{n+1}/t_n$.

2. Ecrire $r(n) = \frac{a(n)}{b(n)} \frac{c(n+1)}{c(n)}$, où $a(n)$, $b(n)$ et $c(n)$ satisfont (5).

3. Trouver un polynôme $x(n)$ solution de (7), s'il en existe. Sinon, retourner $\sum_{k=0}^{n-1} t_k$ et stopper.

4. Retourner $\frac{b(n-1)x(n)}{c(n)} t_n$ et stopper.

Ici, nous avons pris comme borne inférieure 0 dans la somme $\sum_{k=0}^{n-1} t_k$. On peut prendre une autre borne k_0. C'est nécessaire en particulier si t_n n'est pas défini pour un entier positif k, auquel cas il faut prendre k_0 plus grand que k.

3. Le pas 2

Dans cette section, nous montrons comment obtenir la factorisation (4) d'une fraction rationnelle donnée $r(n)$, qui vérifie la condition (5).

Soit $r(n) = f(n)/g(n)$, où $f(n)$ et $g(n)$ sont des polynômes premiers entre eux. Si $\operatorname{pgcd}(f(n), g(n+h)) = 1$ pour tout entier naturel h, on peut prendre $a(n) = f(n)$, $b(n) = g(n)$ et $c(n) = 1$. Nous obtenons tout de suite la factorisation désirée. Sinon, soit $u(n)$ un facteur commun de $f(n)$ et $g(n+h)$ pour un entier naturel h. L'idée est de "diviser $a(n)$ et $b(n)$" par ces facteurs et de les incorporer dans $c(n)$. Plus précisément, si l'on pose $f(n) = \overline{f}(n)u(n)$ et $g(n) = \overline{g}(n)u(n-h)$, alors

$$r(n) = \frac{f(n)}{g(n)} = \frac{\overline{f}(n)}{\overline{g}(n)} \frac{u(n)}{u(n-h)}.$$

La dernière fraction sur la droite peut être écrite comme un produit de fractions de la forme $c(n+1)/c(n)$ de la façon suivante.

$$\frac{u(n)}{u(n-h)} = \frac{u(n)u(n-1)\cdots u(n-h+1)}{u(n-1)u(n-2)\cdots u(n-h+1)u(n-h)}.$$

Ensuite, on répète cette procédure avec $\overline{f}(n)$ et $\overline{g}(n)$ à la place de $f(n)$ et de $g(n)$. En un nombre fini de pas, on doit obtenir la factorisation désirée (4).

Un problème reste à résoudre : comment savoir si la condition (5) est satisfaite, et sinon, comment trouver les entiers naturels h qui ne marchent pas ?

Voici une façon de le faire. Soit $R(h)$ le résultant de $f(n)$ et de $g(n+h)$, vus comme polynômes en n. Les valeurs de h qui contredisent (5) sont les racines positives entières de R.

Théorème 3.1. *Les polynômes a, b et c donnés par l'algorithme décrit dans ce paragraphe vérifient (4) et (5). De plus, si l'on traite les valeurs de h qui contredisent (5) dans l'ordre croissant, les propriétés suivantes sont elles aussi vérifiées.*
(i) *pgcd*$(a(n), c(n)) = 1$.
(ii) *pgcd*$(b(n), c(n+1)) = 1$.
Enfin, une telle factorisation de $r(n)$ vérifiant (5), (i) et (ii) est unique. C'est aussi la factorisation de $r(n)$ vérifiant (5) où le polynôme c a le plus bas degré.

Exemple. Soit $r(n) = (n+3)/(n(n+1))$. Alors le pas 2 de l'algorithme de Gosper donne $a(n) = 1$, $b(n) = n$, $c(n) = (n+1)(n+2)$. Remarquons que les conditions (4) et (5) seront également vérifiées par $a(n) = n-k$, $b(n) = n(n-k+1)$, $c(n) = (n+1)(n+2)(n-k)$ où k est un entier positif, mais là, aucune des propriétés (i) et (ii) du théorème 3.1 ne sont satisfaites.

4. Le pas 3

Dans ce paragraphe, nous expliquons comment trouver une solution polynomiale de (7) de façon systématique. Supposons que $x(n)$ soit un polynôme qui satisfait (7). On note d son degré. Si nous connaissions d, ou au moins une borne supérieure pour d, nous pourrions substituer un polynôme à coefficients indéterminés de degré d à x dans l'équation (7), identifier les coefficients des puissances successives de n et résoudre le système d'équations obtenues pour trouver les coefficients inconnus de $x(n)$. Remarquons que ces équations seraient linéaires, puisque l'équation (7) est linéaire en $x(n)$.

En fait, on peut trouver un nombre fini de candidats pour d (deux en fait). Pour tout polynôme P, on note cd P le coefficient du terme de plus haut degré de P. Nous distinguons deux cas.
Cas 1. deg $a(n) \neq$ deg $b(n)$ ou cd $a(n) \neq$ cd $b(n)$. Alors dans l'équation (7), les coefficients dominants dans le terme de gauche TG

$$TG = a(n)x(n+1) - b(n-1)x(n)$$

de l'équation (7) ne s'annulent pas (ces "coefficients dominants", ce sont les coefficents des termes de plus haut degré de $a(n)x(n+1)$ et de $b(n-1)x(n)$).

Donc le degré de ce terme TG est $d+\max\{\deg a(n), \deg b(n)\}$. Au côté droit, on a $c(n)$. Donc :

$$d = \deg c(n) - \max\{\deg a(n), \deg b(n)\}.$$

Cas 2. $\deg a(n) = \deg b(n)$ et $\mathrm{cd}\, a(n) = \mathrm{cd}\, b(n) = \lambda$. Alors dans l'équation (7), les coefficients dominants dans TG s'annulent. Il y a encore deux cas à considérer.

(2a) Les seconds termes dans l'ordre décroissant des puissances ne s'annulent pas, dans TG. Alors le degré de ce terme de gauche est $d+\deg a(n)-1$, donc

$$d = \deg c(n) - \deg a(n) + 1.$$

(2b) Les seconds termes dans l'ordre décroissant des puissances s'annulent dans TG. Posons

$$a(n) = \lambda n^k + A n^{k-1} + O(n^{k-2}), \tag{8}$$

$$b(n-1) = \lambda n^k + B n^{k-1} + O(n^{k-2}), \tag{9}$$

$$x(n) = C_0 n^d + C_1 n^{d-1} + O(n^{d-2}), \tag{10}$$

où $C_0 \neq 0$. Alors, en développant les termes de gauche de (7), on trouve

$$x(n+1) = C_0 n^d + (C_0 d + C_1) n^{d-1} + O(n^{d-2}),$$

$$a(n)x(n+1) = C_0 \lambda n^{k+d} + (\lambda(C_0 d + C_1) + AC_0) n^{k+d-1} + O(n^{k+d-2}),$$

$$b(n-1)x(n) = C_0 \lambda n^{k+d} + (BC_0 + \lambda C_1) n^{k+d-1} + O(n^{k+d-2}), \tag{11}$$

$$a(n)x(n+1) - b(n-1)x(n) = C_0(\lambda d + A - B) n^{k+d-1} + O(n^{k+d-2}). \tag{12}$$

Par hypothèse, le coefficient de n^{k+d-1} sur le coté droit de cette dernière équation (12) s'annule. Donc $C_0(\lambda d + A - B) = 0$ et

$$d = \frac{B - A}{\lambda}.$$

Donc, dans le cas 2, les seuls degrés possibles pour x sont $\deg c(n) - \deg a(n) + 1$ et $(B - A)/\lambda$, où A et B sont respectivement définis dans (8) et (9). Bien sûr, seuls les candidats positifs sont à considérer. Quand il y a deux candidats pour d, on peut prendre le plus grand des deux comme borne supérieure. Remarquons qu'en général, les cas (2a) et (2b) peuvent tous les deux se produire, puisque l'équation (7) peut avoir des solutions polynomiales de deux degrés distincts.

Exemple. Soit la somme $\sum_{k=1}^{n} 1/(k(k+1))$. Ici, $t_{n+1}/t_n = n/(n+2)$, donc $a(n) = n$, $b(n) = n + 2$, $c(n) = 1$ et l'équation (7) est

$$nx(n+1) - (n+1)x(n) = 1.$$

Sur cet exemple, le cas 1 ne s'applique pas. Le cas (2a) donne $d = 0$ et le cas (2b) $d = 1$, comme les seuls degrés possibles de solutions polynomiales. En effet, la solution générale de cette équation est $x(n) = Cn - 1$, et donc il y a une solution de degré 0 (quand $C = 0$) et des solutions de degré 1 (quand $C \neq 0$).

5. Vers l'algorithme de Zeilberger

L'algorithme de Gosper ne peut trouver des formes closes que pour un petit nombre de sommes, parmi toutes celles qu'on rencontre en pratique. Doron Zeilberger a appliqué les résultats de Gosper, pour trouver une forme close à des expressions de la forme $\sum_k F(n,k)$, où F est doublement hypergéométrique (c'est-à-dire,

$$F(n+1,k)/F(n,k) \text{ et } F(n,k)/F(n,k+1)$$

sont des fractions rationnelles en n et k). Dans cette somme, k varie entre les bornes naturelles de F. Par exemple,

$$\sum_k \binom{n}{k} = \sum_{k=0}^{n} \binom{n}{k}.$$

En fait, Zeilberger a donné un algorithme permettant de trouver une relation de récurrence pour de telles expressions, qui peuvent permettre ensuite d'en donner une forme close. Nous n'allons pas détailler ici cet algorithme, mais l'expliquer sur des exemples.

5.1. **Un échauffement.** Commençons, pour voir, par une expression très simple. Calculons $\sum_k \binom{n}{k} z^k$ par cette méthode. Si l'on applique l'algorithme de Gosper à $t(n,k) = \binom{n}{k} z^k$, on voit que cette expression n'est pas Gosper-sommable. Essayons donc avec

$$u(n,k) = b_0(n)t(n,k) + b_1(n)t(n+1,k),$$

et cherchons des valeurs de b_0 et b_1 pour lesquelles u est Gosper-sommable. En évaluant $t(n+1,k)/t(n,k)$, on trouve

$$u(n,k) = p(n,k)\frac{t(n,k)}{n+1-k},$$

où

$$p(n,k) = (n+1-k)b_0 + (n+1)b_1.$$

Maintenant, appliquons l'algorithme de Gosper à $u(n,k)$. On cherche a, b et c tels que

$$\frac{u(n,k+1)}{u(n,k)} = \frac{c(n,k+1)}{c(n,k)}\frac{a(n,k)}{b(n,k)},$$

où a et b vérifient (5). On trouve : $c = p$, $a = (n+1-k)z$ et $b = k+1$.

Il faut maintenant résoudre

$$c(n,k) = a(n,k)x(n,k+1) - b(n,k-1)x(n,k),$$

où le degré en k du polynôme x est égal à 0. On trouve que $b_0 = z+1$, $b_1 = -1$ et $x = 1$. Donc

$$u(n,k) = (z+1)t(n,k) - t(n+1,k)$$

est Gosper-sommable. Il existe donc un terme $T(n,k)$ hypergéométrique en k tel que

$$u(n,k) = T(n,k+1) - T(n,k).$$

On peut calculer $T(n,k)$: il est égal à $\binom{n}{k-1}z^k$, mais en fait, on n'en a pas besoin. Si en effet on somme $u(n,k) = T(n,k+1) - T(n,k)$ sur k, on trouve que $\sum_k u(n,k) = 0$, et donc, si $S(n) = \sum_k t(n,k)$, on obtient $(z+1)S(n) = S(n+1)$, et donc, par récurrence, $S(n) = (z+1)^n$.

5.2. **Un exemple plus convaincant.** Soit

$$f(n) = \sum_k (-1)^k \binom{2n}{k}^3.$$

Alors par la même méthode, on trouve que

$$f(n) = (-1)^n \frac{(3n)!}{n!^3}.$$

5.3. **La suite d'Apéry.** Soit la suite

$$A(n) = \sum_k \binom{n}{k}^2 \binom{n+k}{k}^2.$$

En utilisant la même méthode que ci-dessus, on trouve la relation de récurrence

$$(n+1)^3 A(n) + (n+2)^3 A(n+2) = (2n+3)(17n^2+51n+39)A(n+1,k).$$

Remarquons que l'ordre de la récurrence est ici égal à deux. Si l'on essaie l'ordre 1, on tombe en effet sur une impossibilité.

Là, on peut se demander si grâce à cette relation de récurrence, on ne pourrait pas trouver une forme close pour $A(n)$. En fait, il existe des algorithmes pour répondre à cette question, et pour trouver la forme close en question. Dans ce cas, la réponse est “non”.

Références

$A = B$, M. Petkovšek, H. S. Wilf, D. Zeilberger.

Mathématiques concrètes, R. L. Graham, D. E. Knuth, O. Patashnik.