Decoupling multivariate fractions

François Lemaire, Adrien Poteaux

▶ **To cite this version:**

François Lemaire, Adrien Poteaux. Decoupling multivariate fractions. Computer Algebra in Scientific Computing, Sep 2021, Sochi, Russia. pp.209–231. hal-03285870

HAL Id: hal-03285870
https://hal.science/hal-03285870

Submitted on 13 Jul 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Decoupling multivariate fractions

François Lemaire[1] and Adrien Poteaux[1]

Univ. Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France
{francois.lemaire,adrien.poteaux}@univ-lille.fr

Abstract. We present a new algorithm for computing compact forms of multivariate fractions. Given a fraction presented as a quotient of two polynomials, our algorithm builds a tree where internal nodes are operators, and leaves are fractions depending on pairwise disjoint sets of variables. The motivation of this work is to obtain compact forms of fractions, which are more readable and meaningful for the user or the modeler, and better suited for interval arithmetic.

Keywords: Multivariate Fractions · Decoupling · Compact form

1 Introduction

This article presents a new algorithm decouple for computing compact forms of multivariate fractions. Informally, given a multivariate fraction given as a quotient P/Q, Algorithm decouple computes a (usually) more compact representative of P/Q in the form of a tree where internal nodes are operators $+$, $\times$ and $\div$, and where leaves are fractions depending on pairwise disjoint sets of variables. As a consequence, the fraction P/Q is usually written as a sum, product or quotient of expressions which may also contain fractions. As an example, our algorithm rewrites the fraction $\frac{a_0b_1a_3+a_0b_1b_2+a_0a_2+a_1a_3+a_1b_2}{b_1a_3+b_1b_2+a_2}$ as $a_0+\frac{a_1}{-\frac{a_2}{-a_3-b_2}+b_1}$, and rewrites the fraction $-\frac{x\left(d\,x^2+dxk_1+dxk_2+dk_1k_2+V_1x+V_2x+V_1k_2+V_2k_1\right)}{(k_1+x)(k_2+x)}$ as $-dx-\frac{V_1x}{k_1+x}-\frac{V_2x}{k_2+x}$. Our algorithm also works with polynomials, and in that case the expressions returned are free of quotients. For example, our algorithm rewrites $ab+ax+bx+cd+cx+dx+2x^2$ as $(x+b)\,(x+a)+(x+c)\,(x+d)$.

This work was mainly motivated by the following reasons. A compact expression is usually easier to read and understand for a user/modeler. Moreover, if the variables appear in the least places (ideally only once), the interval arithmetic should yield sharper results.

Computer algebra software are usually focused on polynomials rather than fractions. Extracting the numerator of a fraction can produce some expression swell, especially if the fraction is given as a sum of different fractions with different denominators. When fractions are reobtained after applying a polynomial-based method, our algorithm can help to recover different fractions with different denominators again.

In order to decompose a fraction into several terms, our method uses a decoupling technique on the variables. Roughly speaking, as the term decoupling

suggests, our method tries to split a fraction into different terms involving disjoint sets of variables. As a consequence, our method does nothing on a univariate fraction, even if the fraction can be written in a compact way using nested fractions.

Simplification of multivariate fractions has already been considered. The Leĭnartas decomposition is presented in [4] (see [8, Theorem 2.1] for an english presentation). It decomposes a fraction into a sum of fractions, using computations on the varieties associated to the irreducible factors of the fraction denominator. Also, [10] presents a partial decomposition for multivariate fractions, based on successive univariate partial decompositions. In both cases, multivariate fractions are rewritten in a more compact way, as a sum of several fractions (thus nested fractions are never produced).

Our method does not work the same way, and produces a different output. Our method can produce nested fractions such as $a_0+\frac{a_1}{-\frac{a_2}{-a_3-b_2}+b_1}$ mentioned earlier in the introduction. However, our method does no simplification on fractions which cannot be decoupled. For example, our algorithm performs no simplification on the fraction $F = \frac{x^2y+xy^2+xy+x+y}{xy(xy+1)}$ taken from [8, Example 2.5], whereas [4, 8] computes $F = \frac{1}{xy+1} + \frac{x+y}{xy}$, and [10] computes $F = \frac{1}{x} + \frac{1}{xy+1} + \frac{1}{y}$.

It is also worth mentioning [11] which provides "Ten commandments" around expression simplifications, especially Sections 3 and 4 which discuss some techniques for partially factoring the numerators and denominators of a fraction. Also, a method for computing Horner's schemes for multivariate polynomials is given in [3]. Finally, [9] presents a choice of nice functionalities a computer algebra software should provide for helping the user with expression manipulations.

We implemented our algorithm decouple in Maple 2020. All examples presented in the paper run under ten seconds (on a i7-8650U CPU 1.90GHz running Linux), and the memory footprint in under 180 Mbytes.

Organization of the paper. Section 2 defines the decouplings of fractions and the splittable fractions. Theorem 1 which characterizes the splittable fractions is presented, and the existence and uniqueness of the so-called finest partition (of variables) is proven. Section 3 presents our algorithm decouple, with elements of proofs. Section 4 presents some examples. Finally, Section 5 presents some complexity results and implementation remarks.

Notations. In this article, $\mathbb{K}$ denotes any field of characteristic zero[1]. Take a fraction F in $\mathbb{K}(X)$, where X contains n variables. For brevity, the partial derivatives $\frac{\partial F}{\partial x}$, $\frac{\partial^2 F}{\partial x \partial y}$ and $\frac{\partial^{j+k} F}{\partial x^j \partial y^k}$ are also written F_x, $F_{x,y}$ and F_{x^j,y^k}. We denote by $\mathrm{Supp}(F)$ the set $\{x \in X | F_x \neq 0\}$; it is simply the variables on which F really depends. We denote by $\mathrm{Def}(F)$ the domain of definition of F, which is the set of values of $\mathbb{K}^n$ which does not cancel the denominator of F.

[1] Fields of characteristic nonzero have not been considered by the authors, as they raise some difficulties. Indeed, most results and algorithms presented here rely on evaluation and differentiation, which are difficult to handle in nonzero characteristic.

For any subset $Y \subseteq X$ of size m, and any $Y^0 \in \mathbb{K}^m$, $F(Y = Y^0)$ designates the partial evaluation of F for the variables Y at Y^0. This partial evaluation is only defined if the denominator of F does not identically vanish at $Y = Y^0$. Partitions of a set X will usually be denoted $(X_i)_{1 \leq i \leq p}$ and $(Y_i)_{1 \leq i \leq q}$, or simply (X_i) and (Y_i).

2 Decoupled and splittable fractions

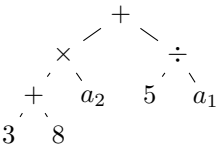
2.1 Definitions

Definition 1 (Expression Tree). *An* Expression Tree *in $a_1, \ldots, a_p$ over a field $\mathbb{K}$ is a finite tree satisfying:*

- *each internal node is a binary operator: either $+$, $\times$, or $\div$,*
- *each leaf is either a variable a_i, or an element of $\mathbb{K}$,*
- *if the tree contains two or more nodes, then any subtree encodes a nonzero fraction in the variables $a_1, \ldots, a_p$.*

Proposition 1 (Expression Tree and associated fraction). *The third item of Definition 1 ensures that no division by zero can occur. As a consequence, any Expression Tree encodes a fraction in the a_i variables. Moreover, any fraction can be encoded by an Expression Tree (note the Expression Tree is not unique). If A is an Expression Tree in $a_1, \ldots, a_p$, we simply denote its associated fraction by $A(a_1, \ldots, a_p)$. Please note that zero can still be encoded by the tree with only one root node equal to zero. Finally, the tree*  *is not an Expression Tree since it violates the third item of Definition 1.*

Definition 2 (Decoupled Expression Tree (DET)). *A* Decoupled Expression Tree *in the variables $a_1, \ldots, a_p$ is an Expression Tree where each a_i appears exactly once.*



is a **DET** encoding $11a_2 + 5/a_1$.

Proposition 2 (Interval arithmetic). *Assume $\mathbb{K}$ is $\mathbb{Q}$ or $\mathbb{R}$. Consider a fraction $F(a_1, \ldots, a_p)$ which can be represented as a **DET** A. If each a_i lies in some interval I_i, and if evaluating the tree A using interval arithmetic never inverses intervals containing zero, then the evaluation computes $F(I_1, \ldots, I_p)$.*

Proof. We prove it by induction on the number of nodes. The base case with one node is immediate. If the number of nodes is higher than 2, the induction hypothesis can be applied on both left and right subtrees, yielding two intervals I and J. The evaluation of the complete tree consists in evaluating either $I + J$, $I \times J$, or $I \div J$. In all these cases, interval arithmetic gives an exact interval image (i.e. not overestimated), since both subtrees involve distinct variables (A is a **DET**), and because (by assumption) no interval containing zero is inverted.

Remark 1. When inverting an interval containing zero occurs during the evaluation of a **DET**, difficulties arise, as the following example shows. Take $F = \frac{x}{1+x}$, whose image on the interval $I = [0, 1]$ is $F(I) = [0, 1/2]$. The fraction F can be written as the **DET** $\frac{1}{1+\frac{1}{x}}$ whose evaluation is delicate because $\frac{1}{x}$ is not defined at $x = 0$. However, if F is written as the **DET** $1 - \frac{1}{1+x}$, Proposition 2 applies.

In order to generalize Proposition 2 for tackling intervals containing zero, multi-intervals and handling intervals containing infinity may be required.

Definition 3 (Decoupling of a fraction). *Let F a fraction of $\mathbb{K}(X)$. We call decoupling of F a triple $(A, (F_i)_{1\leq i\leq p}, (X_i)_{1\leq i\leq p})$ where:*

- *A is a **DET** in the variables $a_1, \ldots, a_p$ over $\mathbb{K}$,*
- *$(X_i)_{1\leq i\leq p}$ is a partition of $\mathrm{Supp}(F)$,*
- *each F_i is a fraction of $\mathbb{K}(X_i)$ with $\mathrm{Supp}(F_i) = X_i$,*
- *$F = A(F_1, \ldots, F_p)$ where $A(F_1, \ldots, F_p)$ designates the fraction associated to A evaluated on the F_i.*

In that case, we say that the partition (X_i) decouples the fraction F.

Remark 2. Any constant fraction F admits the decoupling $(F, \emptyset, \emptyset)$. Any non constant fraction $F \in \mathbb{K}(X)$ admits the (trivial) decoupling $(a_1, (F), (\mathrm{Supp}(F)))$.

Definition 4 (Splittable fraction). *A fraction F of $\mathbb{K}(X)$ is said* splittable *if there exists a partition $(X_i)_{1\leq i\leq p}$ of $\mathrm{Supp}(F)$ with $p \geq 2$, such that (X_i) decouples F. Otherwise, the fraction is said* nonsplittable.

Remark 3. Constant and univariate fractions are nonsplittable.

2.2 Characterization of splittable fractions

Theorem 1 below gives a characterization of splittable fractions. It is central for the decouple algorithm (Algorithm 1). Indeed the decouple algorithm checks the four different cases and either calls itself recursively if one case succeeds or concludes that the fraction is nonsplittable.

Lemma 1. *Take a splittable fraction F of $\mathbb{K}(X)$. Then for any nonzero constant c, the fractions $c + F$, $c \times F$, c/F, and F/c are splittable.*

Proof. For each fraction $c+F$, $c\times F$, c/F, and F/c, it suffices to adjust the tree A of a decoupling $(A, (F_i)_{1\leq i\leq p}, (X_i)_{1\leq i\leq p})$ of F.

Theorem 1 (Splittable characterization). *A fraction F of $\mathbb{K}(X)$ is splittable if and only if the fraction F can be written in one of the following forms:*

$$\textbf{C1}\quad G + H \qquad\qquad \textbf{C2}\quad c + GH$$

$$\textbf{C3}\quad c + \frac{1}{G + H} \qquad\qquad \textbf{C4}\quad c + \frac{d}{1 + GH}$$

where

- c *and* d *are in* $\mathbb{K}$*, and* $d \neq 0$*,*
- (Y, Z) *is a partition of* $\mathrm{Supp}(F)$*,*
- $G \in \mathbb{K}(Y)$ *and* $H \in \mathbb{K}(Z)$*, with* $\mathrm{Supp}(G) = Y$ *and* $\mathrm{Supp}(H) = Z$*.*

Proof. The right to left implication is immediate.

Let us prove the left to right implication. Assume F is splittable, and consider a decoupling $(A, (F_i)_{1\leq i\leq p}, (X_i)_{1\leq i\leq p})$ of F with $p \geq 2$. Since the fraction is splittable, it is necessarily non constant, and the root of the tree A is necessarily an operator. As a consequence, the tree A has the shape $\overset{o}{L\quad R}$. Substituting the F_i's in the L and R trees, one gets two fractions F_L and F_R. There are two cases:

Case 1. Both fractions F_L and F_R are nonconstant. They have by construction some disjoint supports. If the operator o is $+$, then F can be written as $F_L + F_R$ as in the case **C1**. If the operator is $\times$ (resp. $\div$), then F can be written as in the case **C2**, with $c = 0$, $G = F_L$, and $H = F_R$ (resp. $1/F_R$).

Case 2. Among the fractions F_R and F_R, one is constant, and the other one is nonconstant. The nonconstant fraction is splittable by Lemma 1. We consider the following scenario by induction: either the splittable fraction satisfies Case 1, concluding the induction, either we are once again in the Case 2. This process can only happen a finite number of times (since the tree A is finite). We can thus assume that the splittable fraction can be written in one of the four cases.

Assume first that F_R is constant and that the operator is $+$. If F_L has the form **C1** $G + H$, then $F = G + (H + F_R)$. If F_L has form **C2**, **C3** or **C4**, then F has the same form as F_L (by replacing c by $c + F_R$). By a similar argument, F has the same form as F_L is the operator is $\times$ or $\div$.

Assume now that F_L is constant. It is easy to show that F has the same form as F_R is the operator is $+$ or $\times$. If the operator is $\div$, some more computations are needed. If F_R has form **C1**, then F_L/F_R has form **C3** (with $c = 0$). If F_R has form **C2** with $F_R = c + GH$, then F_L/F_R has form **C2** if $c = 0$, and form **C4** otherwise. If F_R has form **C3** with $F_R = c + \frac{1}{G+H}$, then F_L/F_R has form **C1** if $c = 0$, and form **C3** otherwise. If F_R has form **C4** with $F_R = c + \frac{d}{1+GH}$, then F_L/F_R has form **C2** if either $c = 0$ or $c + d = 0$, and form **C4** otherwise. □

Remark 4. Anticipating Propositions 8 and 9, the constant c of Theorem 1 is unique for the cases **C2** and **C3**. Anticipating Proposition 10, the values of c and d in the case **C4** of Theorem 1 are not unique, because a fraction $c + \dfrac{d}{1+GH}$ can also be written as $(c + d) + \dfrac{-d}{1 + \frac{1}{G}\frac{1}{H}}$, which is also of the form **C4**.

2.3 Basic lemmas around fractions

This section gives some lemmas around the evaluation of fractions for some variables. Those lemmas would be quite obvious to prove for polynomials, but

fractions deserve special treatment because of the possible cancellations of the denominators at some evaluation points.

Lemma 2. *Consider a fraction F of $\mathbb{K}(X)$. If the fraction F cancels at any point $X^0 \in \mathrm{Def}(F)$, then the fraction F is the zero fraction.*

Proof. Write F as P/Q where P and Q are polynomials of $\mathbb{K}[X]$. Denote $X = \{x_1, \ldots, x_n\}$. Using a Kronecker substitution (see [5, exercise 8.4, page 247] and references therein), there exists a substitution ϕ of the form $x_1 \mapsto u^{a_1}$, $\ldots$, $x_n \mapsto u^{a_n}$, where u is a new variable and the a_i are positive integers, such that ϕ is injective on the sets of monomials occurring in P and Q.

The polynomial $\phi(Q)$ is nonzero and univariate, so there exists an integer u_0 such that $\phi(Q)(u) \neq 0$ for any integer $u \geq u_0$. As a consequence, the set of points $S = \{(u^{a_1}, \ldots, u^{a_n}) | u \in \mathbb{N}, u \geq u_0\}$ is included in $\mathrm{Def}(F)$.

Since F cancels on $\mathrm{Def}(F)$ by assumption, P cancels on the set S, implying that $\phi(P)(u)$ cancels for any integer $u \geq u_0$. Since $\phi(P)$ is univariate, $\phi(P)$ is the zero polynomial. Since the transformation ϕ is injective on the monomials, P is also the zero polynomial, hence $F = 0$. □

Lemma 3. *Consider a nonzero fraction F of $\mathbb{K}(X)$ and a variable $x \in X$. There exists a finite subset S of $\mathbb{K}$ such that for any $x^0 \in \mathbb{K} \setminus S$, the partial evaluation $F(x = x^0)$ is well-defined and nonzero, and $\mathrm{Supp}(F(x = x^0)) = \mathrm{Supp}(F) \setminus \{x\}$.*

Proof. The lemma is immediate if x does not belong to the support of F. Now assume $x \in \mathrm{Supp}(F)$. Consider the fraction $H = F \prod_{y \in \mathrm{Supp}(F)} F_y$, which by construction is nonzero since F is nonzero. The fraction can be seen as a univariate fraction H of $\bar{\mathbb{K}}(x)$, where $\bar{\mathbb{K}} = \mathbb{K}(X \setminus \{x\})$. Consider the set $\bar{S} \subseteq \bar{\mathbb{K}}$ of elements $\bar{x}_0 \in \bar{\mathbb{K}}$ either canceling the numerator or the denominator of H. This set $\bar{S}$ is finite. Take $S = \bar{S} \cap \mathbb{K}$, which is also finite. Then for any element $x^0 \in \mathbb{K} \setminus S$, the fraction $H(x = x^0)$ is well-defined and nonzero. This ends the proof since $H(x = x^0) \neq 0$ implies $F(x = x^0) \neq 0$, and $F_y(x = x^0) \neq 0$ for any $y \in \mathrm{Supp}(X)$. □

The following lemma is a generalization of Lemma 3 for evaluating two different fractions simultaneously.

Lemma 4. *Consider two nonzero fractions F and G of $\mathbb{K}(X)$. For any variable $x \in X$, there exists a finite subset S of $\mathbb{K}$ such that for any $x^0 \in \mathbb{K} \setminus S$, the partial evaluations $F(x = x^0)$ and $G(x = x^0)$ are well-defined and nonzero, $\mathrm{Supp}(F(x = x^0)) = \mathrm{Supp}(F) \setminus \{x\}$, and $\mathrm{Supp}(G(x = x^0)) = \mathrm{Supp}(G) \setminus \{x\}$.*

Proof. The proof is similar to that of Lemma 3, simply replace the fraction H by $F(\prod_{y \in \mathrm{Supp}(F)} F_y)\, G(\prod_{y \in \mathrm{Supp}(G)} G_y)$. □

Lemma 5. *Consider a nonconstant univariate fraction F of $\mathbb{K}(x)$. For any finite set of values $S \subseteq \mathbb{K}$, there exists a value $x^0 \in \mathbb{K}$ such that $F(x^0)$ is well-defined and $F(x^0) \notin S$.*

Proof. Let us assume that the image of the fraction F is included in S. We prove that this leads to a contradiction. Since F is univariate, there exists an integer u_0 such that the denominator Q does not cancel on the set $D = \{u \in \mathbb{N} | u \geq u_0\}$. Since F is defined on D, and D is infinite, and S is finite, there exists a value v of S such that $F(u) = v$ for an infinite number of integers $u \geq u_0$. This implies that the numerator of $F - v$ cancels on an infinite number of integers, hence $F - v$ is the zero fraction. Contradiction since F is nonconstant. □

2.4 Finest decoupling partition

We prove in this section that for any fraction F, there exists a unique most refined partition decoupling F. The following definition is classical.

Definition 5 (Finer partition). *A partition $(X_1, \ldots, X_p)$ of some set X is finer than a partition $(Y_1, \ldots, Y_q)$ of X if each X_i is included in some Y_j. The finer-than relation is a partial order.*

Definition 6 (Partition deprived of one element). *Consider a partition $(X_i)_{1 \leq i \leq p}$ of some set X, and a variable $x \in X$. Up to a renaming of the X_i, assume that $x \in X_p$.*

Build a partition (Y_i) of $X \setminus \{x\}$ in the following way: if X_p is equal to $\{x\}$, then take the partition $(Y_i)_{1 \leq i \leq p-1}$ where $Y_i = X_i$ for $1 \leq i \leq p-1$. Otherwise take the partition $(Y_i)_{1 \leq i \leq p}$ where $Y_i = X_i$ for $1 \leq i \leq p-1$, and $Y_p = X_p \setminus \{x\}$.

The partition (Y_i) is called the partition (X_i) deprived of x.

The following proposition shows how to specialize a variable in a decoupling.

Proposition 3 (Specialization of a decoupling). *Let us consider a decoupling $(A, (F_i)_{1 \leq i \leq p}, (X_i)_{1 \leq i \leq p})$ of some fraction F of $\mathbb{K}(X)$ and a variable $x \in \mathrm{Supp}(F)$. Denote (Y_i) the partition (X_i) deprived of x.*

Then there exists an $x^0 \in \mathbb{K}$ such that the partition (Y_i) decouples $F(x = x^0)$.

Proof. Up to a renaming of the X_i, assume that $x \in X_p$. Assume that the set X_p equals $\{x\}$. Assigning a value a^0 to the variable a_p in the **DET** A may not yield a **DET** because of the third condition of Definition 1. However, there only exists a finite set S of "unlucky" values a^0 which break the third condition of Definition 1. By Lemma 5 on the univariate fraction $F_p(x)$ and S, there exists an x^0 such that $(A(a_p = F_p(x^0)), (F_i)_{1 \leq i \leq p-1}, (X_i)_{1 \leq i \leq p-1})$ is a decoupling of $F(x = x^0)$.

Now assume that the $\{x\}$ is strictly included in X_p. By Lemma 3, there exists a value x^0 such that $F_p(x = x^0)$ is well-defined, and $\mathrm{Supp}(F_p(x = x^0)) = X_p \setminus \{x\}$. Thus, replacing F_p by $F_p(x = x^0)$ and X_p by $X_p \setminus \{x\}$ in the decoupling $(A, (F_i)_{1 \leq i \leq p}, (X_i)_{1 \leq i \leq p})$ of F yield a decoupling for $F(x = x^0)$. □

The following proposition is a generalization of Proposition 3 for specializing two different decouplings of the same fraction F.

Proposition 4 (Simultaneous specialization of two decouplings). *Consider two decouplings* $(A, (G_i)_{1\leq i\leq p}, (X_i)_{1\leq i\leq p})$ *and* $(B, (H_i)_{1\leq i\leq q}, (U_i)_{1\leq i\leq q})$ *of the same fraction* F *of* $\mathbb{K}(X)$*, and a variable* $x \in \mathrm{Supp}(F)$*. Denote* (Y_i) *the partition* (X_i) *deprived of* x*, and* (V_i) *the partition* (U_i) *deprived of* x*.*

Then there exists a value $x^0 \in \mathbb{K}$ *such that both partitions* (Y_i) *and* (V_i) *decouples* $F(x = x^0)$*.*

Proof. The proof is similar to that of Proposition 3. The only difficulty is the choice of an x^0 which is suitable for both decouplings. Up to a renaming of the X_i and U_i, assume that $x \in X_p$ and $x \in U_q$. If both X_p and U_q are equal to $\{x\}$, then there is a finite number of values for x^0 to avoid, hence Lemma 5 concludes. Assume X_p is the singleton $\{x\}$ and U_q strictly contains x. By Lemmas 3 and 5, there is also a finite number of values for x^0 to avoid, which ends the proof. By symmetry, we need not consider the case where X_p strictly contains x and U_q is the singleton $\{x\}$. Finally, if x is strictly included in X_p and in U_q, Lemma 4 concludes. □

Lemma 6. *Consider a fraction* $F \in \mathbb{K}(X)$*, and a* ***DET*** C *in one variable* a_1*. Denote* $C(F)$ *the fraction obtained by evaluating* C *on* $a_1 = F$*. If the fraction* $C(F)$ *is splittable, then* F *is also splittable.*

Proof. Consider a **DET** C in one variable a_1. It can be shown by induction that the fraction $C(a_1)$ associated to C is an homography, i.e. $C(a_1) = \dfrac{u^0 + v^0\, a_1}{u^1 + v^1 a_1}$ where the u^0, v^0, u^1 and v^1 are in $\mathbb{K}$, and satisfy $u^0 v^1 - u^1 v^0 \neq 0$. Indeed, the variable a_1 is an homography, and adding a constant to an homography, multiplying an homography by a nonzero constant, or taking the inverse of an homography yield an homography. Since an homography is invertible, and its inverse is also an homography, $C(a_1)$ is invertible and its inverse $D(a_1)$ is an homography, which can easily be encoded by a **DET** in one variable.

Since $C(F)$ is splittable, adding D to the top of the tree of the decoupling of F yields a decoupling of $D(C(F))$ which is equal to F, hence F is splittable. □

Proposition 5 (Finest partition). *Consider a fraction* $F \in \mathbb{K}(X)$*. There exists a unique* finest partition (X_i) *of* $\mathrm{Supp}(F)$ *decoupling* F*. A decoupling* $(A, (F_i), (X_i))$ *of* F *is said* finest *if* (X_i) *is the finest partition decoupling* F*.*

Proof. The set S of partitions decoupling F is not empty by Remark 2, and is also finite. Since the finer-than relation is a partial order (Definition 5), the existence of finest partitions is guaranteed. We now prove that all finest partitions are in fact equal, which is the difficult part of the proof.

The proposition is immediate for constant fractions. Consider a non constant fraction $F \in \mathbb{K}(X)$, and two different finest partitions $(X_i)_{1\leq i\leq p}$ and $(Y_i)_{1\leq i\leq q}$ decoupling F, along with some corresponding decouplings $(A, (G_i), (X_i))$ and $(B, (H_i), (Y_i))$.

Since the two partitions (X_i) and (Y_i) are different, there exists a X_k intersecting at least two different sets of the (Y_i) partition. Without loss of generality,

let us assume that the set X_1 intersects the sets Y_1, Y_2, ..., Y_r with $r \geq 2$, and does not intersect the remaining sets Y_{r+1}, ..., Y_q. See figure 1 for an illustration. We prove below that the set X_1 can be further refined into $X_1 \cap Y_1$, ..., $X_1 \cap Y_r$, leading to a contradiction since (X_i) is finest.

Let us apply successively Proposition 4 on all variables of $X_2 \cup X_3 \cup \cdots \cup X_p$, thus obtaining some values X_2^0, ..., X_p^0. We obtain two different decouplings for $\bar{F} = F(X_2 = X_2^0, X_3 = X_3^0, \ldots, X_p = X_p^0)$. The first one (obtained from $(A, (G_i), (X_i))$ is $(C, (G_1), (X_1))$ where C is the (univariate) tree $A(a_2 = G_2(X_2^0), \ldots, a_p = G_p(X_p^0))$. The second one (obtained from $(B, (H_i), (Y_i)))$ is $(D, (R_i)_{1 \leq i \leq r}, (U_i)_{1 \leq i \leq r})$ where (U_i) is the partition $(X_1 \cap Y_1, \ldots, X_1 \cap Y_r)$ of X_1. As a consequence, the fraction $\bar{F}$ is splittable. Since $\bar{F} = C(G_1)$ is splittable, the fraction G_1 is also splittable by Lemma 6. This contradicts the fact that (X_i) is finest, since G_1 could be split in the decoupling $(A, (G_i), (X_i))$ of F. □

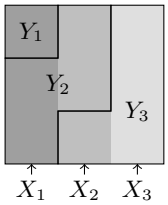


Fig. 1. Two partitions (X_i) and (Y_i) of X. The sets X_1, X_2 and X_3 are the rectangles in dark gray, gray and light gray. The set X_1 intersects the sets Y_1 and Y_2, but not Y_3.

2.5 Decomposition into a sum and product

In this section, we give necessary and sufficient conditions for decomposing a fraction $F \in \mathbb{K}(X)$ with $\mathrm{Supp}(F) = X$, into a sum $G + H$ or a product GH, where $G \in \mathbb{K}(Y)$, $H \in \mathbb{K}(Z)$, and (Y, Z) is a partition of X.

Over the reals or the complexes, the conditions are immediate to show using converging Taylor expansions for example. However, we show them in a more general context for any field $\mathbb{K}$ of characteristic zero, such as for example $\mathbb{Q}(a, b, c)$. We avoid here the use of converging Taylor expansions, which would require us to equip $\mathbb{K}$ with a norm. By the way, the authors of [1] required a normed vector space, which was in fact not required by using techniques presented here.

Proposition 6 (Decomposition into sum). *Let $F \in \mathbb{K}(X)$ with $\mathrm{Supp}(F) = X$, and (Y, Z) a partition of X. Then there exist two fractions $G \in \mathbb{K}(Y)$ and $H \in \mathbb{K}(Z)$ such that $F = G + H$ if and only if $F_{y,z} = 0$ for any $(y, z) \in (Y, Z)$.*

Proof. The left to right implication is immediate. To prove the right to left implication, we assume for simplicity that X only contains two elements y and z. Consider a point $X^0 = (y^0, z^0) \in \mathrm{Def}(F)$. Without loss of generality, using a shift on variables y and z, let us assume that $X^0 = (0, 0)$.

Take $G = F(z = 0)$ and $H = F(y = 0) - F(0, 0)$, and take $U = F - (G + H)$. Then $U_{y,z} = F_{y,z}$ is also the zero fraction. Moreover, U cancels on the varieties

$y = 0$ and $z = 0$, whenever U is well-defined. We now show that the fraction U is the zero fraction, which proves that $F = G + H$ as required.

Write U as P/Q with P and Q polynomials. Since $P = QU$ and using a classical Taylor expansion on P, one gets

$$P = \sum_{j \geq 0, k \geq 0} \frac{1}{j!k!} (QU)_{y^j, z^k}(0,0) y^j z^k \tag{1}$$

where only a finite terms are non zero since P is a polynomial (hence no convergence arguments are needed here).

Since $U_{y,z}$ is the zero fraction, then all terms with $j \geq 1$ and $k \geq 1$ in Equation (1) are zero. We finish the proof by showing (using the symmetry on y and z) that any term $(QU)_{y^j}(0,0)$ is zero, which proves that P is the zero polynomial.

By Lemma 2, the fraction $U(z = 0)$ is the zero fraction, hence the $(QU)(z = 0)$ is the zero polynomial. Using the fact that evaluating at $z = 0$ and differentiating w.r.t. y commute, $(QU)_{y^j}(0,0) = ((QU)(z = 0))_{y^j}(y = 0) = 0$. □

Remark 5. Proposition 6 can be interpreted using a graph. Indeed, with notations of Proposition 6, and writing $X = \{x_1, \ldots, x_n\}$, consider the undirected graph with nodes x_i and with edges the (x_i, x_j) such that $G_{x_i,x_j} \neq 0$. Then the fraction F can be written as a sum if and only if the graph admits at least two connected components.

The next proposition is the equivalent of Proposition 6 for decomposition a fraction as a product instead of a sum. From an analytical point of view, decomposing a fraction F as a product GH corresponds to decomposing $\ln F$ as the sum of $\ln G$ and $\ln H$. The differentiation conditions of Proposition 6 applied on $\ln F$, yield $(\ln F)_{y,z} = \left(\frac{F_y}{F}\right)_z = 0$. This condition, which does not involve a logarithm, is used in the following Proposition.

Proposition 7 (Decomposition into product). *Consider $F \in \mathbb{K}(X)$ with $\mathrm{Supp}(F) = X$, and (Y, Z) a partition of X. Then there exist two fractions $G \in \mathbb{K}(Y)$ and $H \in \mathbb{K}(Z)$ such that $F = GH$ if and only if $\left(\frac{F_y}{F}\right)_z = 0$ for any $(y, z) \in (Y, Z)$.*

Proof. The left to right implication is immediate. To prove the right to left implication, we assume for simplicity that X only contains two elements y and z. The fraction F is not constant since its support X is not empty. By contraposition of Lemma 2, there exists a point $X^0 \in \mathrm{Def}(F)$ such that $F(X^0) \neq 0$. Without loss of generality, using a shift on variables y and z, let us assume that $X^0 = (0,0)$.

Take $G = F(z = 0)$ and $H = F(y = 0)/F(0,0)$ and consider $U = F/(GH) - 1$. We prove below that the fraction U is equal to 0, thus showing that $F = GH$.

Write U as $\frac{P}{Q}$ with P and Q polynomials. Since U is zero on $z = 0$, then by Lemma 2, the fraction $U(z = 0)$ is the zero fraction. Using the commutativity

argument at the end of Proposition 6 proof, $U_{y^j}(0,0) = 0$ for any nonnegative integer j. By symmetry on y and z, $U_{z^k}(0,0) = 0$ for any nonnegative integer k.

The condition $\left(\frac{F_y}{F}\right)_z = 0$ can be rewritten as $F_{y,z} = \frac{F_y F_z}{F}$. This implies that $U_{yz} = \frac{U_y U_z}{U+1}$. By an inductive argument on $k+l$, and using the fact that $U_{y^j}(0,0) = 0$ and $U_{z^k}(0,0) = 0$ for any nonnegative integers j and k, one proves that $U_{y^j,z^k}(0,0) = 0$ for any nonnegative integers j and k. Using Equation (1), the polynomial P is zero, hence $U = 0$ which ends the proof. □

3 Algorithm decouple

Algorithm 1: decouple(F, X)

Input: A fraction $F \in \mathbb{K}(X)$ and a list $X = [x_1, \ldots, x_n]$
Output: A finest decoupling $(A, [F_1, \ldots, F_p], [X_1, \ldots, X_p])$ of F

1 **if** F is constant **then return** $(F, [\,], [\,])$;
2 **else if** F does not depend on x_1 **then return** decouple$(F, [x_2, \ldots, x_n])$;
3 **else if** F only depends on x_1 **then return** $(a_1, [F], [\{x_1\}])$;
4 **else if** checkC1(F, X) returns G, Y, H, Z **then**
5 $\quad (A_G, \bar{G}, \bar{Y}) :=$ decouple(G, Y) ;
6 $\quad (A_H, \bar{H}, \bar{Z}) :=$ decouple(H, Z) ;
7 $\quad$ let r be the length of $\bar{G}$
8 $\quad$ shift the variables of A_H by r i.e. replace each a_i by a_{i+r} in A_H
9 $\quad$ **return** $(\ A_G + A_H\ , \bar{G} + \bar{H}, \bar{Y} + \bar{Z})$
10 $\quad$ `/* where` $\bar{G} + \bar{H}$ `and` $\bar{Y} + \bar{Z}$ `are list concatenations */`
11 **else if** checkC2(F, X) returns c, G, Y, H, Z **then**
12 $\quad$ proceed as in Lines 5 to 8 ;
13 $\quad$ **return** $(\ c + A_G \times A_H\ , \bar{G} + \bar{H}, \bar{Y} + \bar{Z})$
14 **else if** checkC3(F, X) returns c, G, Y, H, Z **then**
15 $\quad$ proceed as in Lines 5 to 8 ;
16 $\quad$ **return** $(\ c + 1 \div A_G + A_H\ , \bar{G} + \bar{H}, \bar{Y} + \bar{Z})$
17 **else if** checkC4(F, X) returns c, d, G, Y, H, Z **then**
18 $\quad$ proceed as in Lines 5 to 8 ;
19 $\quad$ **return** $(\ c + d \div 1 + A_G \times A_H\ , \bar{G} + \bar{H}, \bar{Y} + \bar{Z})$
20 **else return** $(a_1, [F], [\mathrm{Supp}(F)])$;

Algorithm decouple takes as input a fraction F and a list X, such that $F \in \mathbb{K}(X)$, and returns a finest decoupling of F. Unless F is a constant, or a univariate fraction, the four cases of Theorem 1 are sequentially checked by calling the four so-called functions checkC1 , ..., checkC4. If one of them succeeds (returning

some G and H plus other results), Algorithm 1 calls itself on G and H and builds the final result. Otherwise, if no case succeeds, the fraction is proved to be nonsplittable (by Theorem 1) and F is returned.

The main difficulty in the process is to prove that the four checkC1 , ..., checkC4 functions are correct, which is done in the four following subsections.

3.1 Algorithm checkC1 ($F = G + H$)

Following Proposition 6 and Remark 5, Algorithm checkC1 computes (using Algorithm 3) the connected component Y containing the node x_1 of the undirected graph with vertices the x_i, and with edges the (x_i, x_j) such that $F_{x_i,x_j} \neq 0$. If this component Y is strictly included in the support of F, then one can split F as a sum of two fractions $G + H$ as in the case **C1**, using some (random) evaluation to compute the (non-unique) G and H fractions.

Algorithm 2: checkC1(F, X)

Input: A fraction $F \in \mathbb{K}(X)$ s.t. $x_1 \in \mathrm{Supp}(F)$ and a list $X = [x_1, \ldots, x_n]$
Output: Succeeds by returning G, Y, H, Z if F can be written as $G + H$ (as in the case **C1**), and fails otherwise

```
1 Y := connectedVariablesSum(F, X) ;
2 Z := Supp(F) \ Y ;
3 if Z is empty then FAIL ;
4 else
5 |   G := F(Z = Z^0) where Z^0 is a random point such that F(Z = Z^0)
  |     is well-defined ;
6 |   H := F − G ;
7 |   succeed by returning G, Y, H, Z ;
```

Algorithm 3: connectedVariablesSum(F, X)

Input: A fraction $F \in \mathbb{K}(X)$ and a list $X = [x_1, \ldots, x_n]$
Output: Return the connected component containing x_1 of the undirected graph with nodes x_i, and with edges the (x_i, x_j) such that $F_{x_i,x_j} \neq 0$.

```
1 visited := {} ;
2 todo := {x_1} ;
3 while todo is not empty do
4 |   pick and extract some variable v from todo ;
5 |   visited := visited ∪ {v} ;
6 |   V := Supp(F_v) ;
7 |   todo := todo ∪ (V \ visited) ;
8 return visited
```

3.2 Algorithm checkC2 ($F = c + GH$)

Algorithm checkC2 proceeds similarly to Algorithm checkC1 but it first needs to compute a constant candidate c such that $F - c$ can be written as a product as in the case **C2**.

Proposition 8. *Take a fraction $F \in \mathbb{K}(X)$ of the form **C2** (i.e. $F = c + GH$) where $\mathrm{Supp}(G) = Y$, $\mathrm{Supp}(H) = Z$ and (Y, Z) is a partition of $\mathrm{Supp}(F)$. Then $c = F - \frac{F_y F_z}{F_{y,z}}$ for any $(y, z) \in (Y, Z)$.*

Moreover, if for some $(u, v) \in X^2$, the expression $F - \frac{F_u F_v}{F_{u,v}}$ is well-defined and constant, then it is equal to the c defined above.

Proof. The first part of the proposition is a simple computation (note that the formula for c is well-defined because $F_{y,z} = G_y H_z$, which is nonzero).

For the second part of the proposition, there is nothing to prove if $(u, v) \in (Y, Z)$, or $(u, v) \in (Z, Y)$. Assume (by symmetry) that both u and v lies in Y, and that $F - \frac{F_u F_v}{F_{u,v}}$ is well-defined and constant.

Simple computations show that $F - \frac{F_u F_v}{F_{u,v}} = c + (G - \frac{G_u G_v}{G_{u,v}})H$. Since this expression is constant, the term $G - \frac{G_u G_v}{G_{u,v}}$ is necessarily zero (otherwise, the expression would not be constant since the supports of G and H are distinct), which ends the proof. □

Remark 6. The case $G - \frac{G_u G_v}{G_{u,v}} = 0$ in the previous proof occurs for example if the fraction G can itself be written as a product of two fractions M and N of disjoint supports, with $u \in \mathrm{Supp}(M)$ and $v \in \mathrm{Supp}(N)$.

Proposition 8 ensures that Algorithm 5 can stop as soon as it finds a constant candidate c. Indeed, if F has form **C2**, then the constant candidate c is correct by Proposition 8. If F has not form **C2**, and if a constant candidate c is computed (which can happen by Remark 7), then the call to checkProd$(F - c)$ at Line 2 of Algorithm 4 will detect that F has not form **C2**.

Remark 7. Fix $X = \{x, y, z\}$. The following fraction $F = 3 + (x + z)/(y + z)$ yields $F - \frac{F_x F_y}{F_{x,y}} = 3$. However, it can be shown (using Algorithm decouple) that the fraction F cannot be written as **C2**. This does not contradict Proposition 8, since Algorithm 2 will detect that $F - 3$ cannot be decomposed as a non trivial product. Finally, note that F is splittable (of the form **C2**) if we take for example $X = \{x, y\}$ and $\mathbb{K} = \mathbb{Q}(z)$.

Algorithm 4: checkC2(F, X)

Input: A fraction $F \in \mathbb{K}(X)$ and a list $X = [x_1, \ldots, x_n]$
Output: Succeed by returning G, Y, H, Z if F can be written as $c + GH$ (as in the case **C2**), and fails otherwise

1 **if** findConstantCase2(F, X) returns a constant c **then**
2 **if** checkProd$(F - c, X)$ returns G, Y, H, Z **then**
3 **return** c, G, Y, H, Z
4 **else** FAIL ;
5 **else** FAIL ;

Algorithm 5: findConstantCase2(F, X)

Input: A fraction $F \in \mathbb{K}(X)$ and a list $X = [x_1, \ldots, x_n]$
Output: Either return some constant candidate c for case **C2**, or fails.

1 **for** i from 2 to n **do**
2 **if** $F_{x_1,x_i} \neq 0$ **then**
3 $c := F - \dfrac{F_{x_1} F_{x_i}}{F_{x_1,x_i}}$;
4 **if** $c \in \mathbb{K}$ **then return** c ;
5 FAIL

Algorithm 6: checkProd(F, X)

Input: A fraction $F \in \mathbb{K}(X)$ with $x_1 \in \mathrm{Supp}(F)$, and a list $X = [x_1, \ldots, x_n]$
Output: Succeed by returning G, Y, H, Z if F can be written as GH (i.e. as in the case **C2** with $c = 0$), and fails otherwise

1 $Y :=$ connectedVariablesProd(F, X) ;
2 $Z := \mathrm{Supp}(F) \setminus Y$;
3 **if** Z is empty **then** FAIL ;
4 **else**
5 $G := F(Z = Z^0)$ where Z^0 is a random point such that $F(Z = Z^0)$ is well-defined and nonzero ;
6 $H := F/G$;
7 succeed by returning G, Y, H, Z

Algorithm 7: connectedVariablesProd(F, X)

Input: A fraction $F \in \mathbb{K}(X)$ and a list $X = [x_1, \ldots, x_n]$
Output: Return the connected component containing x_1 of the undirected graph with nodes x_i, and with edges the (x_i, x_j) s.t. $\left(\frac{F_{x_i}}{F}\right)_{x_j} \neq 0$.

1 Same algorithm as Algorithm 3 except F_v is replaced by $\dfrac{F_v}{F}$ in Line 6

3.3 Algorithm checkC3 ($F = c + 1/(G + H)$)

Algorithm checkC3 proceeds similarly to Algorithm checkC2 but with a different formula for computing c, G, and H. Once a candidate c is found, Algorithm checkC3 tries to decompose $1/(F-c)$ into a sum $G+H$ using Algorithm checkC1.

Proposition 9. *Take a fraction $F \in \mathbb{K}(X)$ of the form **C3** (i.e. $F = c+1/(G+H)$) where $\mathrm{Supp}(G) = Y$, $\mathrm{Supp}(H) = Z$ and (Y, Z) is a partition of $\mathrm{Supp}(F)$. Then $c = F - 2\frac{F_y F_z}{F_{y,z}}$ for any $(y, z) \in (Y, Z)$.*

Moreover, if for some $(u, v) \in X^2$, the expression $F - 2\frac{F_u F_v}{F_{u,v}}$ is well-defined and constant, then it is equal to c defined above.

Proof. The first part of the proof is once again a simple computation. For the second part, there is nothing to prove if $(u, v) \in (Y, Z)$ or $(u, v) \in (Z, Y)$. Assume (by symmetry) that both u and v lies in Y, and that the expression $F - 2\frac{F_u F_v}{F_{u,v}}$ is well-defined and constant.

Computations yields $F - 2\frac{F_u F_v}{F_{u,v}} = c + \dfrac{G_{u,v}}{G_{u,v}(G + H) - 2G_u G_v}$. This expression is equal to c when $G_{u,v} = 0$. If $G_{u,v} \neq 0$, the numerator of the expression is free of Z, but the support of denominator contains Z, hence a contradiction since the expression is constant. □

Algorithm 8: checkC3(F, X)

Input: A fraction $F \in \mathbb{K}(X)$ and a list $X = [x_1, \ldots, x_n]$
Output: Succeed by returning c, G, Y, H, Z if F can be written as $c + 1/(G + H)$ (as in the case **C3**), and fails otherwise

```
1 if findConstantCase3(F, X) returns a constant c then
2 |   U := 1/(F − c) ;
3 |   if checkC1(U, X) returns G, Y, H, Z then return c, G, Y, H, Z ;
4 |   else FAIL ;
5 else FAIL ;
```

Algorithm 9: findConstantCase3(F, X)

Input: A fraction $F \in \mathbb{K}(X)$ and a list $X = [x_1, \ldots, x_n]$
Output: Either return some constant candidate c for case **C3**, or fails.

1 Same algorithm as Algorithm 5 except Line 3 is replaced by

$$c := F - 2\frac{F_{x_1} F_{x_i}}{F_{x_1, x_i}}$$

3.4 Algorithm checkC4 ($F = c + d/(1 + GH)$)

Algorithm checkC4 proceeds similarly to Algorithms checkC2 and checkC3. Once candidates c and d are found, Algorithm checkC4 tries to decompose $d/(F-c)-1$ into a product GH using Algorithm checkProd.

However the computations for finding the candidates c and d are more difficult, because c and d are solutions of quadratic equations. As a consequence,

some fractions of $\mathbb{K}(X)$ are nonsplittable in $\mathbb{K}(X)$, but are splittable in $\bar{\mathbb{K}}(X)$ where $\bar{\mathbb{K}}$ is some extension of $\mathbb{K}$. Here is a quite easy example demonstrating this fact.

Example 1. The fraction $F = \frac{xy+a}{x+y} \in \mathbb{K}(x,y)$, where $a \in \mathbb{K}$, can be written as

$$F = -b + \frac{2b}{1 - \left(1 - \frac{2}{\frac{x}{b}+1}\right)\left(1 - \frac{2}{\frac{y}{b}+1}\right)}$$

if b satisfies $b^2 = a$. As a consequence, the fraction F is splittable in $\mathbb{K}(x,y)$ if a is a square in $\mathbb{K}$, and nonsplittable in $\mathbb{K}(x,y)$ otherwise.

Propositions 10 and 11 below explain the process used by Algorithm 11 for finding the candidates c and d.

Proposition 10. *Take a fraction $F \in \mathbb{K}(X)$ of the form **C4** (i.e. $c+d/(1+GH)$) where $\mathrm{Supp}(G) = Y$, $\mathrm{Supp}(H) = Z$, c and d constants, with $d \neq 0$.*

Then, for any $(y,z) \in (Y,Z)$, we have

$$\frac{1}{F-c} + \frac{1}{J-c} = \frac{2}{d} \tag{2}$$

where $J = F - 2\frac{F_y F_z}{F_{y,z}}$. Moreover, the couple (c,d) is unique up to the (involutive) transformation $(c,d) \mapsto (c+d,-d)$.

Proof. Take $(y,z) \in (Y,Z)$. Note that J is well-defined since $F_{yz} = \frac{2dG_y H_z}{(1+GH)^3}$, which is nonzero. Computations show that $J = c + \frac{d}{1-GH}$, and Equation (2) follows. Equation (2) can be rewritten as

$$-(c+d/2)S + c(c+d) + P = 0 \tag{3}$$

where $S = F + J$ and $P = FJ$. One proves that S is nonconstant. Indeed if S were constant, then $2c + d(\frac{1}{1+GH} + \frac{1}{1-GH})$ would be constant, $\frac{1}{1+GH} + \frac{1}{1-GH}$ would be constant, hence G^2H^2 would be constant, a contradiction since G and H are nonconstant with disjoint supports.

Since S is non constant, there exist X^0 and X^1 such that where $S(X^0) \neq S(X^1)$. Substituting those values in (3) yield a invertible linear system, hence unique values for $\bar{a} = -(c+d/2)$ and $\bar{b} = c(c+d)$.

Finally, d is solution of $d^2 = 4(\bar{a}^2 - \bar{b})$, and $c = -\bar{a} - d/2$. Hence the couple (c,d) is unique, up to the (involutive) transformation $(c,d) \mapsto (c+d,-d)$. □

Proposition 11. *Take the same hypotheses as in Proposition 10. Take $(u,v) \in X^2$ and assume $F_{u,v} \neq 0$. Consider $J = F - 2\frac{F_u F_v}{F_{u,v}}$, and $S = F + J$ and $P = FJ$. Assume S is not constant. Consider the (unique) solution $(\bar{a},\bar{b})$ of $aS+b+P = 0$, with $\bar{a}$ and $\bar{b}$ constants, if such a solution exists. Then, if $4(\bar{a}^2 - \bar{b})$ is nonzero and admits a squareroot $\bar{d}$ in $\mathbb{K}$, then $(\bar{c},\bar{d})$, where $\bar{c} = -\bar{a} - \bar{d}/2$, is either (c,d) or $(c+d,-d)$.*

Proof. If $(u,v) \in (Y,Z)$ or $(u,v) \in (Z,Y)$, Proposition 10 concludes. Assume (by symmetry) that both u and v lies in Y. Computations show that $J = c + d\frac{G_{u,v}}{G_{u,v}+(GG_{u,v}-2G_uG_v)H}$. If $G_{u,v} = 0$, then $J = c$. It implies that $S = F + c$, $P = cF$. As a consequence, S is not constant, and $(\bar{a},\bar{b}) = (-c, c^2)$ is the unique solution of $aS+b+P = 0$. In that case, $4(\bar{a}^2-\bar{b}) = 0$, which ends the proof. Now assume that $G_{u,v} \neq 0$. The fraction J can then be rewritten as $J = c+d\dfrac{1}{1+\bar{G}H}$ where $\bar{G} = G - 2\frac{G_uG_v}{G_{u,v}}$.

Let us first consider the case $\bar{G} = -G$ (which happens for example if G can be written as a product, see Remark 6). This implies $J = c + d\frac{1}{1-GH}$, and the proof ends by following the proof of Proposition 10.

Now consider that $\bar{G} \neq -G$. By taking the numerator of the equation $aS + b + P = 0$, tedious computations[2] yield:

$$G\bar{G}(2ac + c^2 + b)\,H^2 + (2ac + c^2 + b + (a + c)d)(\bar{G} + G)\,H + (2ac + c^2 + b + 2(a + c)d + d^2) = 0. \quad (4)$$

Since the supports of G and H are disjoint, and H is nonconstant, the previous equation can only hold if the three coefficients in H are the zero fractions. This implies that the three constants $u_2 = 2ac + c^2 + b$, $u_1 = 2ac + c^2 + b + (a + c)d$ and $u_0 = 2ac + c^2 + b + 2(a + c)d + d^2$ are zero. Expanding $u_2 - 2u_1 + u_0 = 0$ yields $d^2 = 0$, a contradiction, which ends the proof. □

Algorithm 10: checkC4(F, X)

Input: A fraction $F \in \mathbb{K}(X)$ and a list $X = [x_1, \ldots, x_n]$
Output: Succeed by returning c, d, G, Y, H, Z if F can be written as $c + d/(1 + GH)$ (as in the case **C4**), and fails otherwise

```
if findConstantsCase4(F, X) returns a couple (c, d) then
    U := d/(F − c) − 1 ;
    if checkProd(U, X) returns G, Y, H, Z then
        return c, d, G, Y, H, Z
    else FAIL ;
else FAIL ;
```

[2] use your favorite computer algebra system!

Algorithm 11: findConstantsCase4(F, X)

Input: A fraction $F \in \mathbb{K}(X)$ and a list $X = [x_1, \ldots, x_n]$
Output: Either return some couple candidate $(c, d) \in \mathbb{K}^2$ for case **C4**, or fails.

for i from 2 to n **do**
  if $F_{x_1,x_i} \neq 0$ **then**
    $J := F - 2\frac{F_{x_1} F_{x_i}}{F_{x_1,x_i}}$;
    $S := F + J$;
    $P := F \times J$;
    if S is not constant **then**
      find two random points X^0 and X^1 non canceling the denominators of F and J, such that $S(X^0) \neq S(X^1)$;
      find the solution $(\bar{a}, \bar{b})$ of the linear system $aS(X^0) + b + P(X^0) = 0$, $aS(X^1) + b + P(X^1) = 0$;
      if (the fraction $\bar{a}S + \bar{b} + P$ is the zero fraction, and if $4(\bar{a}^2 - \bar{b})$ is **nonzero** and admits a squareroot $\bar{d} \in \mathbb{K}$, **then**
        $\bar{c} := -\bar{a} - \bar{d}/2$;
        succeed by returning $(\bar{c}, \bar{d})$
FAIL

4 Examples

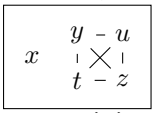
Example 2. The polynomial $p = ab+ax+bx+cd+cx+dx+2x^2$ can be decoupled in $\mathbb{K}(a, b, c, d)$ with $\mathbb{K} = \mathbb{Q}(x)$ into $(x + b)(x + a) + (x + c)(x + d)$.

Note that p is not splittable in $\mathbb{Q}(a, b, c, d, x)$.

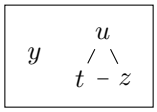
Example 3. We present here a worked out example to illustrate how the decouple algorithm works. To make the walkthrough readable, we do not fully detail all values returned by the algorithms. Consider the fraction $F = x^2+x+4+\frac{y+1}{z+\frac{2}{1+tu}}$ taken in $\mathbb{Q}(x, y, z, t, u)$, whose expanded form is

$$\frac{tux^2z + tuxz + tuy + 4tuz + x^2z + tu + 2x^2 + xz + 2x + y + 4z + 9}{tuz + z + 2}.$$

When calling decouple$(F, [x, y, z, t, u])$, the call to checkC1 succeeds. Indeed, the graph considered during the call to connectedVariablesSum is



which admits two connected components. The connected component containing the vertex x is simply $\{x\}$, so F can be written as the sum $Q + R$, with $Q = F(y = -2, z = 0, t = 1, u = 1) = x^2 + x + 3 \in \mathbb{K}(x)$ and $R = F - Q = \frac{tuy+tuz+tu+y+z+3}{tuz+z+2} \in \mathbb{K}(y, z, t, u)$. Please note that the previous evaluation is chosen randomly in checkC1, we just picked a possible one for the example.

Then the function decouple performs recursive calls on Q and R. The call to decouple$(Q,[x])$ simply returns Q itself. When calling decouple$(R,[y,z,t,u])$, the call to checkC2 succeeds. Indeed, the constant $c = R - \frac{R_y R_z}{R_{y,z}} = 1$ is computed by findConstantCase2. Then checkProd$(S,[y,z,t,u])$ where $S = R - 1$ is able to split S as a product TU, where $T = S(z=0,t=1,u=1) = y+1 \in \mathbb{K}(y)$ and $U = S/T = \frac{tu+1}{tuz+z+2} \in \mathbb{K}(z,t,u)$. Indeed, the graph considered in connectedVariablesProd$(S,[y,z,t,u])$ is  which admits two connected components.

Then the function decouple performs recursive calls on T and U. The call to decouple$(T,[y])$ simply returns T. When calling decouple$(U,[z,t,u])$, the call to checkC3 succeeds. Indeed, the constant $c = U - \frac{U_z U_t}{U_{z,t}} = 0$ is computed by findConstantCase3. Then checkC1$(1/U,[z,t,u])$ is able to split $1/U$ as a sum $V+W$ where $V = 1/U(t=1,u=0) = z+2 \in \mathbb{K}(z)$ and $W = 1/U - V = -\frac{2tu}{1+tu}$.

Then the function decouple performs recursive calls on V and W. The call to decouple$(V,[z])$ simply returns V. Finally, when calling decouple$(W,[t,u])$, checkC4 succeeds. Algorithm findConstantsCase4 computes $c = 0$ and $d = -2$ and checkProd$(-d/(W-c)-1,[t,u])$ decomposes $-d/(W-c)-1 = \frac{1}{tu}$ as a product $\frac{1}{t}$ times $\frac{1}{u}$.

Putting everything together, decouple$(F,[x,y,z,t,u])$ returns a tree encoding the fraction $(x^2+x+3)+(1+\frac{y+1}{z+2-\frac{2}{1+\frac{1}{tu}}})$.

Please note that the output would be different (and slightly more compact) if Algorithm findConstantsCase4 were returning $c = -2$ and $d = 2$, instead of $c = 0$ and $d = -2$ (see Proposition 10).

Example 4. The main point of this example is to show how Equations (5) below can be rewritten into Equations (6). We however quickly explain how to derive Equations (5) using [2, 7].

Consider the following reactions, which simulate a gene G regulated by the protein P it produces: $G + P \underset{b}{\overset{a}{\rightleftharpoons}} H$, $G \xrightarrow{e} G + M$, $M \xrightarrow{f} M + P$, $M \xrightarrow{V_m,k_m} \emptyset$, $P \xrightarrow{V_p,k_p} \emptyset$. The three first reactions follow the classical mass action law, and the two last are Michaelis-Menten degradations. Assuming the binding/unbinding of the protein is fast, the following dynamical system can be obtained:

$$\begin{aligned} H'(t) = -G'(t) &= \frac{a((fM(t)-V_p)P(t)+fk_pM(t))G(t)}{(aG(t)+aP(t)+b)(k_p+P(t))}, \\ M'(t) &= \frac{(eG(t)-V_m)M(t)+ek_mG(t)}{k_m+M(t)}, \\ P'(t) &= \frac{((fM(t)-V_p)P(t)+fk_pM(t))(aP(t)+b)}{(aG(t)+aP(t)+b)(k_p+P(t))}. \end{aligned} \tag{5}$$

The right hand sides are quite compact, and one clearly sees some denominators like $k_p+P(t)$ and $k_m+M(t)$ coming for the Michaelis-Menten degradations, and $aG(t)+aP(t)+b$ coming from the fast binding/unbinding hypothesis.

Using Algorithm decouple on the right hand sides of Equations (5) seen as fractions of $\mathbb{K}(V_m, k_m, V_p, k_p, a, b, e, f, G, H, M)$ with $\mathbb{K} = \mathbb{Q}(P)$, one gets

$$H'(t) = -G'(t) = \frac{fM(t) - \dfrac{V_p P(t)}{k_p + P(t)}}{1 + \dfrac{\frac{b}{a} + P(t)}{G(t)}} \tag{6}$$

$$M'(t) = -\frac{V_m}{\dfrac{k_m}{M(t)} + 1} + eG(t) \qquad P'(t) = \frac{fM(t) - \dfrac{V_p P(t)}{k_p P(t)}}{1 + \dfrac{G(t)}{P(t)\left(1 + \frac{b}{aP(t)}\right)}}.$$

Equations (6) might be of interest for a modeler. For example, the expression of $M'(t)$ in (6) clearly shows that $M'(t)$ is the contribution of two reactions (the degradation of M and the production of M by the gene), whereas it is quite hidden in (5). Expression of $M'(t)$ in (6) clearly indicates a contribution $fM(t) - \frac{V_p P(t)}{k_p+P(t)}$ (the production of P minus the degradation of P) divided by the special correction term $-\left(1 + \frac{\frac{b}{a}+P(t)}{G(t)}\right)$ that comes from the fast binding/unbinding hypothesis.

Please remark that Equations (6) are not obtained anymore if one considers P as a variable instead of putting it in the base field. The reason is that P appears in too many places, which prevents a "nice" decoupling.

Also, note the term $\frac{V_m}{\frac{k_m}{M(t)}+1}$ in the expression of $M'(t)$ in (6). This term is probably a bit odd for a modeler, who would rather prefer the more classical form $\frac{V_m M(t)}{k_m+M(t)}$.

As a last comment, using the `intpakX` package [6], here are the intervals obtained by using either Equations (5) or (6), on the intervals $G = [0.4, 0.7]$, $M = [10.0, 15.0]$, $P = [50.0, 100.0]$, $V_m = [130.0, 250.0]$, $k_m = [100.0, 200.0]$, $V_p = [80.0, 160.0]$, $k_p = [100.0, 200.0]$, $a = [10.0, 20.0]$, $b = [5.0, 10.0]$, $e = [3.1, 4.5]$, $f = [7.8, 11.6]$:

Value for	$H'(t)$	$G'(t)$	$M'(t)$	$P'(t)$
Equations (5)	[-0.07, 8.10]	[-8.10, 0.07]	[-32.79, -2.97]	[-10.53, 1163.61]
Equations (6)	[-0.39, 2.21]	[-2.21, 0.39]	[-31.37, -3.04]	[-28.55, 160.04]

Except for the interval for $P'(t)$, the differences between the intervals using Equations (5) or (6) is here rather minor.

Example 5. The following example is a bit artificial but illustrates how compact fractions can become big when being developed. Consider the fraction

$$F = \frac{a_0 + \frac{a_1}{b_1 + \frac{a_2}{b_2+a_3}}}{c_0 + \frac{c_1}{d_1 + \frac{c_2}{d_2+c_3}}} + \frac{e_0 + \frac{e_1}{f_1 + \frac{e_2}{f_2+e_3}}}{g_0 + \frac{g_1}{h_1 + \frac{g_2}{h_2+g_3}}}$$

which is in a completely decoupled form since each variable only appears once.

Developing F as a reduced fraction P/Q yields a polynomial P of degree 10 with 450 monomials, and a polynomial Q of degree 10 with 225 monomials. Our algorithm decouple recovers from P/Q the fraction F with some minor sign differences

$$\frac{-a_0 - \frac{a_1}{-\frac{a_2}{-a_3-b_2}+b_1}}{-c_0 - \frac{c_1}{-\frac{c_2}{-d_2-c_3}+d_1}} + \frac{-e_0 - \frac{e_1}{f_1+\frac{e_2}{f_2+e_3}}}{-g_0 + \frac{g_1}{\frac{g_2}{-g_3-h_2}-h_1}}.$$

Finally, if each variables is replaced by the interval $[1.0, 5.0]$, the reduced fraction P/Q yields the interval $[0.140\times10^{-5}, 0.284\times10^{7}]$, whereas the decoupled form yields $[0.237, 16.8]$.

5 Implementation and complexity

5.1 Complexity

Algorithm decouple performs $O(n^2)$ operations over $\mathbb{K}(X)$[3], where n is the number of variables of X. Indeed, Algorithm decouple performs at most two recursive calls on a partition of X, plus at most $O(n)$ arithmetic operations: each "check" algorithm performs a linear number of operations over $\mathbb{K}(X)$ (including their subalgorithms), and there is no other operation in $\mathbb{K}(X)$ in Algorithm decouple.

Note that we are not considering the complexity over $\mathbb{K}$, which is much higher. Indeed, differentiations, additions, are intensively used and may produce very large fractions. Also, reduced forms of fractions are computed intensively, causing a lot of gcd computations. Some techniques are mentioned in the next section, in order to limit this problem. Finally, if we consider operations over $\mathbb{K}$, our algorithm is Las Vegas, as there are some random evaluations in Algorithms 2 and 11.

5.2 Implementation

Algorithm decouple has been coded in the Maple 2020 computer algebra system. All examples presented in the paper run under ten seconds (on a i7-8650U CPU 1.90GHz running Linux), and the memory footprint in under 180 Mbytes.

Our implementation has also been intensively stress-tested in the following way. It is easy to compute splittable fractions by building a tree representing a decoupling. Expanding this tree yields a (usually large) fraction which is given to Algorithm decouple, which then recovers the initial decoupling.

Some heuristics have been used in our code to limit potential costly computations. For example, when testing that a fraction is zero, some evaluations are first performed, and the fraction is only developed if all evaluations return zero.

One difficulty in Algorithm 11 is to decide whether the equation $d^2 = 4(\bar{a}^2 - \bar{b})$ admits solution for d in the field $\mathbb{K}$. For the moment, we simply chose to use an expression with a radical for d, thus assuming d exists.

[3] We count here only arithmetic operations and differentiations. Note that complexity is not the main point of this paper, so that we do not go into much details here.

Acknowledgments

This work has been supported by the interdisciplinary bilateral project ANR-17-CE40-0036 and DFG-391322026 SYMBIONT. We also want to thank the reviewers for their useful comments and suggestions which helped improve this article.

References

[1] François Boulier and François Lemaire. "Finding First Integrals Using Normal Forms Modulo Differential Regular Chains". In: *Computer Algebra in Scientific Computing 2015*. Computer Algebra in Scientific Computing. Aachen, Germany: Springer, Sept. 2015, pp. 101–118. URL: https://hal.archives-ouvertes.fr/hal-01234982.

[2] François Boulier et al. "Model Reduction of Chemical Reaction Systems using Elimination". In: *Mathematics in Computer Science* 5 (2011). Presented at the international conference MACIS 2007, submitted to Mathematics in Computer Science, Special Issue on Polynomial System Solving in July 2008, pp. 289–301. URL: http://hal.archives-ouvertes.fr/hal-00184558.

[3] Martine Ceberio and Vladik Kreinovich. "Greedy algorithms for optimizing multivariate Horner schemes". In: *SIGSAM Bull.* 38.1 (2004), pp. 8–15. ISSN: 0163-5824. DOI: 10.1145/980175.980179.

[4] E. K. Leĭnartas. "Factorization of rational functions of several variables into partial fractions". In: *Soviet Math. (Iz. VUZ)* 22.10 (1978), pp. 35–38.

[5] Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. 3rd ed. Cambridge University Press, 2013. DOI: 10.1017/CBO9781139856065.

[6] Ilse Geulig, Walter Krämer, and Markus Grimmer. *The intpakX V1.2 package*. http://www2.math.uni-wuppertal.de/wrswt/software/intpakX/. 2005.

[7] François Lemaire and Aslı Ürgüplü. "MABSys: Modeling and Analysis of Biological Systems". In: *Proceedings of ANB2010*. 2010, pp. 57–72.

[8] Alexander Raichev. *Leĭnartas's partial fraction decomposition*. 2012. arXiv: 1206.4740v3 [math.AC].

[9] David R. Stoutemyer. *A computer algebra user interface manifesto*. 2013. arXiv: 1305.3215v1 [cs.SC].

[10] David R. Stoutemyer. "Multivariate partial fraction expansion". In: *ACM Commun. Comput. Algebra* 42.4 (Feb. 2009), pp. 206–210. ISSN: 1932-2240. DOI: 10.1145/1504341.1504346.

[11] David R. Stoutemyer. "Ten commandments for good default expression simplification". In: *Journal of Symbolic Computation* 46.7 (2011). Special Issue in Honour of Keith Geddes on his 60th Birthday, pp. 859–887. ISSN: 0747-7171. DOI: 10.1016/j.jsc.2010.08.017.