HAL open science

# A Fast Algorithm for Computing the Truncated Resultant

Guillaume Moroz, Éric Schost

## To cite this version:

Guillaume Moroz, Éric Schost. A Fast Algorithm for Computing the Truncated Resultant. ISSAC '16, Sergei A. Abramov; Eugene V. Zima, Jul 2016, Waterloo, Canada. pp.341-348, 10.1145/2930889.2930931 . hal-01366386

**HAL Id: hal-01366386**

**https://hal.science/hal-01366386v1**

Submitted on 14 Sep 2016

**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.

# A Fast Algorithm for Computing the Truncated Resultant

Guillaume Moroz
Inria Nancy Grand Est
guillaume.moroz@inria.fr

Éric Schost
University of Waterloo
eschost@uwaterloo.ca

## ABSTRACT

Let $P$ and $Q$ be two polynomials in $\mathbb{K}[x, y]$ with degree at most $d$, where $\mathbb{K}$ is a field. Denoting by $R \in \mathbb{K}[x]$ the resultant of $P$ and $Q$ with respect to $y$, we present an algorithm to compute $R \bmod x^k$ in $\mathcal{O}\tilde{}(kd)$ arithmetic operations in $\mathbb{K}$, where the $\mathcal{O}\tilde{}$ notation indicates that we omit polylogarithmic factors. This is an improvement over state-of-the-art algorithms that require to compute $R$ in $\mathcal{O}\tilde{}(d^3)$ operations before computing its first $k$ coefficients.

## 1. INTRODUCTION

Computing the resultant of two polynomials is an ubiquitous question in symbolic computation, with applications to polynomial system solving [17], computational topology [18, 2, 12, 7, 8, 5, 22, 6], Galois theory [28, 24, 1, 23], computations with algebraic numbers [4], etc.

From the complexity viewpoint, this question admits a satisfactory answer in the simplest case of polynomials $P, Q$ with coefficients in a field $\mathbb{K}$. Euclid's algorithm can be adapted to compute the resultant $R$ of $P$ and $Q$ in time $\mathcal{O}(d^2)$, assuming arithmetic operations in $\mathbb{K}$ are counted at unit cost (a thorough discussion of resultant algorithms based on Euclid's algorithm is in [14]). Using fast polynomial multiplication and divide-and-conquer techniques, the Knuth-Schönhage half-gcd algorithm [21, 27] allows one to compute $R$ in time $\mathcal{O}\tilde{}(d)$. This is optimal, up to logarithmic factors, since the input has size $\Theta(d)$.

However, no such quasi-linear result is known in the important case of bivariate polynomials over $\mathbb{K}$, or in the very similar case of univariate polynomial with integer coefficients (in which case one would be interested in bit complexity estimates). In the former situation, suppose we consider two polynomials $P$ and $Q$ in $\mathbb{K}[x, y]$, with degree at most $d$, and we want to compute their resultant $R$ with respect to $y$, so that $R$ is in $\mathbb{K}[x]$. The polynomial $R$ has degree at most $d^2$, so both input and output can be represented using $\Theta(d^2)$ elements in $\mathbb{K}$. However, the best known algorithms to compute $R$ take $\mathcal{O}\tilde{}(d^3)$ operations in $\mathbb{K}$, either by means of evaluation / interpolation techniques, or in a direct manner [26].

In this paper, we are interested in the computation of the resultant $R$ of such bivariate polynomials *truncated at order* $k$, that is of $R \bmod x^k$ for some given parameter $k$. This kind of question appears for instance in the algorithms of [17, 23], where we want two terms in the expansion, so that $k = 2$. A related example, in a slightly more involved setting, involves the evaluation of the second derivative of some subresultants, for input polynomials in $\mathbb{K}[x, y, z]$ [19].

Of course, one could simply compute $R$ itself and truncate it afterwards; however, it seems wasteful to compute all $d^2$ terms of $R$, incurring a cost of $\mathcal{O}\tilde{}(d^3)$, before discarding many of them. Now, for all but finitely many values $a$ in $\mathbb{K}$, it is possible to compute $R \bmod (x-a)^k$ using $\mathcal{O}\tilde{}(dk)$ operations in $\mathbb{K}$: indeed, as soon as the non-zero subresultants of $P$ and $Q$ do not vanish at $a$, we can run the Knuth-Schönhage algorithm with coefficients truncated modulo $(x-a)^k$, without attempting to invert an element that would vanish at $a$. The running time claimed above then follows from the fact that arithmetic operations in $\mathbb{K}[x]/(x-a)^k$ can be done using $\mathcal{O}\tilde{}(k)$ operations in $\mathbb{K}$ (for such standard complexity results, our reference is [13]).

If however we cannot choose the expansion point, as is the case here, there is no guarantee that all divisions remain feasible in $\mathbb{K}[x]/\langle x^k \rangle$; attempting to divide by an element of positive valuation would entail a loss of $x$-adic precision.

An obvious solution is to use a *division-free* algorithm over $\mathbb{K}[x]/\langle x^k \rangle$ (by contrast, so-called fraction-free algorithms often require the base ring to be a domain). The best result we are aware of is due to Kaltofen and Villard [20], with a cost of $\mathcal{O}(d^{2.698})$ ring operations to compute the determinant of a matrix of size $d$ over any ring, and thus $\mathcal{O}\tilde{}(d^{2.698}k)$ operations in $\mathbb{K}$ to solve our problem.

In [10], Caruso studies the phenomenon of loss of precision in the iterative version of the subresultant algorithm. He shows that on average, if the base field is finite, this loss of precision grows linearly with the degree of the inputs. In that same reference, he also shows how to modify this algorithm to reduce the loss of precision, resulting in a cost of $\mathcal{O}\tilde{}(d^2(k+\delta))$ operations in $\mathbb{K}$, where $\delta$ is the maximum of the $x$-adic valuations of the non-zero leading subresultants of the input polynomials (under the assumption that these leading subresultants all have valuation less than $k/2$). When $\mathbb{K}$ is finite, the expected value of $\delta$ is $\mathcal{O}(\log(d))$, so that the average running time becomes $\mathcal{O}\tilde{}(d^2k)$.

Our main result is a complexity estimate for the computation of $R \bmod x^k$, using the Knuth-Schönhage divide-and-

ACM ISBN ?.
DOI: ?

conquer algorithm. We show that we can compute $R \bmod x^k$ using $\mathcal{O}^{\sim}(dk)$ base field operations, when $\mathbb{K}$ has characteristic zero, or at least $k$.

We proceed in three steps. First, we compute cofactors $U, V$ and an integer $t$ such that $UV + PQ = x^t \bmod x^{t+1}$ holds in $\mathcal{R}_k = \mathbb{K}[x,y]/\langle x^k \rangle$; this is done in Section 3 by a suitable adaptation of the half-gcd algorithm. From this equality, we will able to deduce a first-order linear differential equation satisfied by the resultant $R$ (Section 5). Solving this differential equation is straightforward, once an initial condition is known; Section 4 shows how to compute the first non-zero term in the resultant.

## 2. PRELIMINARIES

To a polynomial $P$ of $\mathcal{R}_k$, we associate a valuation $\upsilon(P)$ defined as the smallest exponent on $x$ in the monomials of $P$ for $P$ non-zero; by convention, $\upsilon(0) = k$. By Gauss' Lemma, for any polynomials $P$ and $Q$ in $\mathcal{R}_k$, we have $\upsilon(PQ) = \upsilon(P) + \upsilon(Q)$ if the sum is less than $k$ (otherwise, $PQ = 0$). In all the algorithms of the paper, a polynomial $P$ of $\mathcal{R}_k$ is represented by the monomial $x^{\upsilon(P)}$ and the polynomial $P/x^{\upsilon(P)}$ of valuation 0. We do not take into account the boolean cost of exponent manipulations, in the sense that we assume that we can add two valuations in constant time. We define the valuation $\upsilon(P,Q)$ of a pair of polynomials $(P,Q) \in \mathcal{R}_k^2$ as the minimum of the valuations of $P$ and $Q$. We let $\mathcal{S}_k \subset \mathcal{R}_k^2$ be the set of all pairs of polynomials $(P,Q)$ with $\upsilon(P) < \upsilon(Q)$, to which we adjoin $(0,0)$.

For $n \in \mathbb{N}$, we define $\Pi_n$ as the function from $\mathcal{R}_k$ to itself, and by extension from $\mathcal{S}_k$ to itself, such that:

$$\Pi_n(P) = P \operatorname{rem}_x x^{\upsilon(P)+n}$$
$$\Pi_n(P,Q) = (P \operatorname{rem}_x x^{\upsilon(P,Q)+n}, Q \operatorname{rem}_x x^{\upsilon(P,Q)+n});$$

if $\upsilon(P) + n \geq k$ in this definition, we simply replace it by $k$. We denote by $\mathcal{U}(\mathcal{R}_k)$ the set of invertible elements of $\mathcal{R}_k$; they are exactly the polynomials $P$ such that $\upsilon(P) = 0$ and $\Pi_1(P) \in \mathbb{K}$.

We call *degree* of an element of $\mathbb{K}[x]/\langle x^k \rangle$ the degree of its canonical lift to $\mathbb{K}[x]$. We will often write expressions of the form $Q = P/x^{\upsilon(P)}$, for some $P$ in $\mathcal{R}_k$. Such a quotient is not unique; we remove ambiguities by requiring that all coefficients of $Q$ have degree in $x$ less than $k - \upsilon(P)$.

If $P$ is a polynomial in $\mathcal{R}_k$, then $[x^i]P \in \mathbb{K}[y]$ is the coefficient of $x^i$ in it. Then, we say that $P$ is *normal* if $\deg_y([x^{\upsilon(P)}]P) > \deg_y([x^i]P)$ for all $i > \upsilon(P)$. Equivalently, this is the case if the leading coefficient of $P$ has the form $cx^{\upsilon(P)}$, for some non-zero $c$ in $\mathbb{K}$.

Being normal is a useful property. For instance, it allows us to define quotient and remainder in a Euclidean division in many cases: if $P \in \mathcal{R}_k$ is normal and $Q \in \mathcal{R}_k$ has valuation at least $\upsilon(P)$, there exist $U$ and $R$ in $\mathcal{R}_k$ with $\deg(R) < \deg(P)$, and such that $Q = UP + R$; $U$ is uniquely defined modulo $x^{k-\upsilon(P)}$, whereas $R$ is unique. We will write $Q \operatorname{div}_y P$ for the unique quotient $U$ with degree in $x$ less than $k - \upsilon(P)$, and $R = Q \operatorname{rem}_y P$.

The following normalization property will be essential for our algorithms. Below, $\mathsf{M}(n)$ is a bound on the number of arithmetic operations in $\mathbb{K}$ for computing the product of two polynomials $f, g$ in $\mathbb{K}[x]$ of degree at most $n$; we assume that $\mathsf{M}$ satisfies the super-linearity conditions of [13, Chapter 8]. Using the Cantor-Kaltofen algorithm [9], we can take $\mathsf{M}(n)$ in $\mathcal{O}(n \log(n) \log\log(n))$. Using Kronecker's substitution as in [15], multiplication or Euclidean division in degree $d$ in $\mathcal{R}_k$ can then be done in $\mathcal{O}(\mathsf{M}(kd))$ operations in $\mathbb{K}$.

Lemma 1 (Normalization). *For $P$ non-zero in $\mathcal{R}_k$, there exist polynomials $U$ and $T$ in $\mathcal{R}_k$ such that $T$ is normal, $U$ is a unit, with $\Pi_1(U) = 1$, and $P = UT$.*

*The polynomial $T$ is unique, whereas $U$ is uniquely defined modulo $x^{k-\upsilon(P)}$. Moreover, if $P$ has degree $d$, for $n \leq k - \upsilon(P)$, given $\Pi_n(P)$, $\Pi_n(U)$ and $\Pi_n(T)$ can be computed using $\mathcal{O}(\mathsf{M}(dn))$ operations in $\mathbb{K}$.*

Proof. Up to dividing $P$ by $x^{\upsilon(P)}$, we may assume that $P$ has valuation 0. Then, uniqueness and the corresponding algorithm are from Algorithm Q in [25]; the cost analysis is a straightforward extension of that given for Hensel step in [13, Chapter 15] using Kronecker substitution for the arithmetic operations. Once the result is known for $P/x^{\upsilon(P)}$, we recover our claim by multiplying $T$ by $x^{\upsilon(P)}$. □

Given $P$ in $\mathcal{R}_k$, we denote by respectively $\operatorname{Lc}(P)$ and $N(P)$ the unique $U$ and $T$ that satisfy the above conditions, with $U$ having degree in $x$ less than $k - \upsilon(P)$; these two polynomials have degree in $y$ at most $d$, and $\upsilon(N(P)) = \upsilon(P)$. We also define $\operatorname{Lc}_n(P) = \Pi_n(\operatorname{Lc}(P))$ and $N_n(P) = \Pi_n(N(P))$; by the previous lemma, if $P$ has degree $d$, for any $n$, $\operatorname{Lc}_n(P)$ and $N_n(P)$ can be computed in time $\mathcal{O}(\mathsf{M}(dn))$.

Using this result, we can reduce the problem of computing the resultant of two polynomials in $\mathcal{R}_k$ to the problem of computing the resultant of two normal polynomials.

Lemma 2. *Let $P$ and $Q$ be in $\mathcal{R}_k$ of degrees in $y$ at most $d$. Then there exist four monic polynomials $A, B, C, D$ in $\mathcal{R}_k$, with degrees at most $d$, and $u$ in $\mathbb{K}[x]/\langle x^k \rangle$ such that $\operatorname{Res}(P,Q) = u \operatorname{Res}(A,B) \operatorname{Res}(C,D)$. Moreover, $A, B, C, D, u$ can be computed in time $\mathcal{O}(\mathsf{M}(dk))$.*

Proof. By the multiplicative property of the resultant, $\operatorname{Res}(P,Q) = \operatorname{Res}(\operatorname{Lc}(P), Q) \operatorname{Res}(N(P), Q)$. Let $d_c, d_n, d_Q$ be the degrees in $y$ of $\operatorname{Lc}(P), N(P)$ and $Q$ respectively.

Taking the reciprocal polynomial of $\operatorname{Lc}(P)$ and $Q$ changes at most the sign of their resultant. Let us thus define $\tilde{P} = y^{d_c}\operatorname{Lc}(P)(1/y)$ and $\tilde{Q} = y^{d_Q}Q(1/y)$ and let $c_0 \in \mathbb{K}[x]/\langle x^k \rangle$ be the leading coefficient of $\tilde{P}$. By construction, $c_0$ is a unit, so we can define $A = \tilde{P}/c_0 \in \mathcal{R}_k$, and we have $\operatorname{Res}(\operatorname{Lc}(P), Q) = (-1)^{d_c d_Q} {c_0}^{d_Q} \operatorname{Res}(A, \tilde{Q})$, where $A$ is monic in $y$. Let $R$ be the remainder of the Euclidean division of $\tilde{Q}$ by $A$ and let $B = R + A$; since $A$ is monic, $\operatorname{Res}(A, \tilde{Q}) = \operatorname{Res}(A, B)$.

Similarly, because $N(P)$ is normal, we can write it as $N(P) = n_0 C$, where $n_0 \in \mathbb{K}[x]/\langle x^k \rangle$ is its leading coefficient and $C$ is monic in $y$; then, we have $\operatorname{Res}(N(P), Q) = {n_0}^{d_Q} \operatorname{Res}(C, Q)$. Defining as above $D = (Q \operatorname{rem}_y C) + C$, we deduce $\operatorname{Res}(N(P), Q) = {n_0}^{d_Q} \operatorname{Res}(C, D)$, and finally

$$\operatorname{Res}(P,Q) = (-1)^{d_c d_Q} (c_0 n_0)^{d_Q} \operatorname{Res}(A,B) \operatorname{Res}(C,D).$$

$A$ is monic of degree at most $d$, since $\operatorname{Lc}(P)$, has degree at most $d$. The remainder $R$ has degree less than the degree of $A$, so that $B = R + A$ is monic of the same degree as $A$. The same holds for $C$ and $D$.

In terms of complexity, after computing $\operatorname{Lc}(P)$ and $N(P)$, all other operations are $\mathcal{O}(d)$ inversions or multiplications of power series in $\mathbb{K}[x]/\langle x^k \rangle$, and $\mathcal{O}(1)$ Euclidean divisions by monic polynomials of degree at most $d$ in $\mathcal{R}_k$. Their total cost is $\mathcal{O}(\mathsf{M}(dk))$ operations in $\mathbb{K}$. □

## 3. COMPUTING PSEUDO INVERSES

In this section, we show that given two polynomials $P, Q$ in $\mathcal{R}_k$, we can compute a matrix $\mathbf{M} = \left(\begin{smallmatrix} U & V \\ X & Y \end{smallmatrix}\right) \in \mathscr{M}_2(\mathcal{R}_k)$ such that $UP + VQ$ is of the form $x^t \bmod x^{t+1}$, for some integer $t$; we call $U$ and $V$ *pseudo-inverses* of $P$ and $Q$.

To simplify notation, for $s = (P, Q)$ and $\mathbf{M}$ as above, we simply write $\mathbf{M} \cdot s$ for the matrix-vector product $\mathbf{M}\left(\begin{smallmatrix} P \\ Q \end{smallmatrix}\right)$.

### 3.1 The pseudo-division operator $\mathcal{Q}$

In this subsection, we define an operator $\mathcal{Q} : \mathcal{S}_k \to \mathcal{S}_k$ and study its properties. For $s = (0, 0)$, we define $\mathcal{Q}(s) = \mathrm{Id}$; otherwise, the construction involves three stages.

For a non-zero pair of polynomials $s = (P, Q)$ in $\mathcal{S}_k$, define

$$\eta(s) := \upsilon(Q \operatorname{rem}_y N(P)) - \upsilon(P).$$

This is well-defined, as $N(P)$ has the same valuation as $P$.

Lemma 3. *For an integer $n$ and $s$ in $\mathcal{S}_k$, given $\Pi_n(s)$, we can compute $\min(\eta(s), n)$ using $\mathcal{O}(\mathsf{M}(dn))$ operations in $\mathbb{K}$.*

Proof. We start by dividing both $P$ and $Q$ by $x^{\upsilon(P)}$; this does not involve any arithmetic operation. We can then compute the normalization $N_n(P)/x^{\upsilon(P)}$, and do the Euclidean division of $Q/x^{\upsilon(P)}$ by this polynomial, using coefficients taken modulo $x^n$, both in time $\mathcal{O}(\mathsf{M}(dn))$. The valuation of the remainder is precisely $\min(\eta(s), n)$. $\square$

The next lemma shows a more intrinsic characterization of $\eta$. We denote by $\sigma_0 : \mathcal{R}_k \to \mathbb{K}[y]$ the evaluation morphism that sends $x$ to 0.

Lemma 4. *For any integer $t \geq 0$, with $J_t(s) = \langle P, Q\rangle : x^{\upsilon(s)+t}$ and $I_t(s) = \sigma_0\left(J_t(s)\right)$, we have*

$$\begin{cases} I_t(s) = I_0(s) \text{ if } 0 \leq t < \eta(s) \\ I_t(s) \supsetneq I_0(s) \text{ if } t \geq \eta(s). \end{cases}$$

Proof. For $t \geq \upsilon(s)$, we have $I_0(s) \subset I_t(s)$. Then, let $Q' = Q \operatorname{rem}_y N(P)$. If $0 \leq t < \eta(s)$ let $W$ be such that $x^{\upsilon(s)+t}W = UP + VQ' \in \langle P, Q\rangle$. Since $x^{\upsilon(s)+\eta(s)}$ divides $Q'$ and $t + 1 \leq \eta(s)$ we have $x^{\upsilon(s)+t}W = UP \mod x^{\upsilon(s)+t+1}$, so $x^t$ divides $U$ and $x^{\upsilon(s)}W = U'P \mod x^{\upsilon(s)+1}$. Thus $I_t(s) \subset I_0(s)$ and $I_t(s) = I_0(s)$. If $t \geq \eta(s)$, $I_t(s)$ contains the residue class of the remainder of $\Pi_1(Q)/x^{\upsilon(Q)}$ by $\Pi_1(P)/x^{\upsilon(P)}$, that is a non-zero polynomial of degree less than $\deg_y(\Pi_1(P))$, and is not included in $I_0(s)$. $\square$

For $s = (P, Q)$ as above, perform a Euclidean division of $Q$ by the normal polynomial $N_{\eta(s)}(P)$, and define the matrix

$$\mathbf{D}_s := \begin{pmatrix} x^{\eta(s)} & 0 \\ -\left(Q \operatorname{div}_y N_{\eta(s)}(P)\right) \operatorname{rem}_x x^{\eta(s)} & \mathrm{Lc}_{\eta(s)}(P) \end{pmatrix}.$$

Then, the polynomial $\tilde{Q}$ defined by

$$\mathbf{D}_s \cdot \begin{pmatrix} P \\ Q \end{pmatrix} = \begin{pmatrix} x^{\eta(s)}P \\ \tilde{Q} \end{pmatrix}$$

has valuation $\upsilon(P) + \eta(s)$ and we have the inequality $\deg_y(\Pi_1(\tilde{Q})) < \deg_y(\Pi_1(P))$.

Given $P, Q \in \mathbb{K}[y]$ of degree at most $d$, and denoting by $G$ their monic gcd, there exists an invertible matrix $\mathbf{G}_{(P,Q)}$ with entries in $\mathbb{K}[y]$ of degree less than $d$ such that

$$\mathbf{G}_{(P,Q)} \cdot \begin{pmatrix} P \\ Q \end{pmatrix} = \begin{pmatrix} G \\ 0 \end{pmatrix}.$$

If $Q = 0$ then $G_{(P,Q)}$ is the identity matrix. By extension, for $s = (P, Q) \in \mathcal{R}_k^2$ with $\upsilon(P) = \upsilon(Q)$, we define $\mathbf{G}_s$ as $\mathbf{G}_{(\Pi_1(P)/x^{\upsilon(P)}, \Pi_1(Q)/x^{\upsilon(Q)})}$. By construction, the entries $(\tilde{P}, \tilde{Q})$ of $\mathbf{G}_s \cdot s$ satisfy $\upsilon(\tilde{P}) < \upsilon(\tilde{Q})$, or $\tilde{P} = \tilde{Q} = 0$; in other words, the pair $\mathbf{G}_s \cdot s$ belongs to $\mathcal{S}_k$.

For a pair $s = (P, Q)$ in $\mathcal{S}_k$ with $P \neq 0$ we define $\mathbf{N}_s$ by

$$\mathbf{N}_s = \begin{pmatrix} \mathrm{Lc}(P)^{-1} & 0 \\ -\mathrm{Lc}(P)^{-1}(Q \operatorname{div}_y N(P)) & 1 \end{pmatrix}.$$

If $P = 0$, set $\mathbf{N}_s = \mathrm{Id}$. Else, $\upsilon(Q) > \upsilon(P)$ and there exists a matrix $\mathbf{R}$ such that $\mathbf{N}_s = \mathrm{Id} + x\mathbf{R}$; then, $\mathbf{N}_s$ is invertible.

Lemma 5. *For $s = (P, Q) \in \mathcal{S}_k^2$, define*

$$\mathcal{Q}(s) := \mathbf{N}_{\mathbf{G}_{\mathbf{D}_s \cdot s} \cdot \mathbf{D}_s \cdot s} \cdot \mathbf{G}_{\mathbf{D}_s \cdot s} \cdot \mathbf{D}_s.$$

*Then,*

1. *$\mathcal{Q}(s) \cdot s$ is in $\mathcal{S}_k$;*
2. *the following equality between ideals holds:*
$$\langle \mathcal{Q}(s) \cdot s\rangle = \langle s\rangle \cap \left\langle x^{\upsilon(s)+\eta(s)}\right\rangle;$$
3. *$\Pi_1(\mathcal{Q}(s) \cdot s) = \left(\begin{smallmatrix} x^{\upsilon(s)+\eta(s)}G \\ 0 \end{smallmatrix}\right)$, where $G$ is the gcd of $\Pi_1(P)/x^{\upsilon(s)}$ and $\Pi_1(\tilde{Q})/x^{\upsilon(s)+\eta(s)}$;*
4. *$\mathcal{Q}(s) \cdot s$ and $\mathcal{Q}(s') \cdot s'$ generate the same ideal, for any pair $s'$ that generates the same ideal as $s$;*
5. *the entries of $\mathcal{Q}(s) \cdot s$ have degree less than $\deg_y(\Pi_1(P))$.*

*For $s = (P, Q) \in \mathcal{S}_k$, where $P$ and $Q$ have degree at most $d$, given $\Pi_n(s)$ for some $n \geq \eta(s)$, $\Pi_{n-\eta(s)}(\mathcal{Q}(s) \cdot s)$ can be computed using $\mathcal{O}(\mathsf{M}(dn) + \mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$.*

Proof. We saw just above the lemma that $\mathbf{G}_{\mathbf{D}_s \cdot s} \cdot (\mathbf{D}_s \cdot s)$ belongs to $\mathcal{S}_k$. Since applying the matrix $\mathbf{N}_{\mathbf{G}_{\mathbf{D}_s \cdot s} \cdot \mathbf{D}_s \cdot s}$ to this vector does not change the valuations of its entries, we deduce that $\mathcal{Q}(s) \cdot s$ is in $\mathcal{S}_k$.

In order to prove the relation $\langle \mathcal{Q}(s) \cdot s\rangle = \langle s\rangle \cap \langle x^{\upsilon(s)+\eta(s)}\rangle$, it is sufficient to prove that $\langle \mathbf{D}_s \cdot s\rangle = \langle s\rangle \cap \langle x^{\upsilon(s)+\eta(s)}\rangle$, since the matrices $\mathbf{N}_{\mathbf{G}_{\mathbf{D}_s \cdot s} \cdot \mathbf{D}_s \cdot s}$ and $\mathbf{G}_{\mathbf{D}_s \cdot s}$ are invertible over $\mathcal{R}_k$. The two polynomials in the vector $\mathbf{D}_s \cdot s = (x^{\eta(s)}P, \tilde{Q})$ are divisible by $x^{\upsilon(P)+\eta(s)}$, thus $\langle \mathbf{D}_s \cdot s\rangle \subset \langle s\rangle \cap \langle x^{\upsilon(P)+\eta(s)}\rangle$. For the other inclusion, let $W$ be a polynomial in $\langle s\rangle \cap \langle x^{\upsilon(P)+\eta(s)}\rangle$. Since we have $s = \langle P, Q\rangle = \langle P, \tilde{Q}\rangle$, we can write $W = UP + V\tilde{Q}$, for some $U, V$ in $\mathcal{R}_k$. On the other hand, we know that $x^{\upsilon(P)+\eta(s)}$ divides $W$, and since it also divides $\tilde{Q}$, it divides $UP$. This implies that $x^{\eta(s)}$ divides $U$, which means that $W$ is in $\langle \mathbf{D}_s \cdot s\rangle$. This allows us to conclude for the equality of ideals, noting that $\upsilon(s) = \upsilon(P)$.

The third point comes from the fact that the matrix $\mathbf{N}_{\mathbf{G}_{\mathbf{D}_s \cdot s} \cdot \mathbf{D}_s \cdot s}$ can be written $\mathrm{Id} + x\mathbf{R}$, so that $\Pi_1(\mathcal{Q}(s) \cdot s) = \Pi_1(\mathbf{G}_{\mathbf{D}_s \cdot s} \cdot \mathbf{D}_s \cdot s) = \left(\begin{smallmatrix} x^{\upsilon(s)+\eta(s)}G \\ 0 \end{smallmatrix}\right)$.

If $s$ and $s'$ generate the same ideal, $\eta(s) = \eta(s')$ (Lemma 4); using the second item, this proves point 4.

The degree property follows from the fact that in the pair $(A, B) = \mathcal{Q}(s) \cdot s$, $A$ is normal of degree $\deg_y(G) < \deg_y(\Pi_1(P))$, and $B$ is a remainder modulo $N(A)$.

Computing $\mathbf{D}_s$ can be done in time $\mathcal{O}(\mathsf{M}(d\eta(s)))$ using Lemma 1 to compute $N_{\eta(s)}(P)$ and $\mathrm{Lc}_{\eta(s)}(P)$ in time $\mathcal{O}(\mathsf{M}(d\eta(s)))$. The matrix-vector product that gives $\tilde{Q}$ is done in degree $n$ in $x$ and $d$ in $y$, in time $\mathcal{O}(\mathsf{M}(dn))$. Then,

we saw that $\Pi_1(\tilde{Q})$ has degree less than $d$, so computing $\mathbf{G}_{\mathbf{D}_s\cdot s}$ is an extended gcd calculation in $\mathbb{K}[y]$ that can be done in time $\mathcal{O}(\mathsf{M}(d)\log(d))$. Applying $\mathbf{G}_{\mathbf{D}_s\cdot s}$ to $\mathbf{D}_s\cdot s$ takes $\mathcal{O}(n\mathsf{M}(d))$, and results in a matrix of degree $\mathcal{O}(d)$ in $y$. Finally, applying $\mathbf{N}_{\mathbf{G}_{\mathbf{D}_s\cdot s}\cdot\mathbf{D}_s\cdot s}$ to $\mathbf{G}_{\mathbf{D}_s\cdot s}\cdot\mathbf{D}_s\cdot s$ is again a normalization at precision $n$ along with arithmetic operations that can all be done in time $\mathcal{O}(\mathsf{M}(dn))$. $\square$

## 3.2 An extension of the half-gcd

Our goal is now to iterate the pseudo-division $\mathcal{Q}(s)$ until we reach an integer $t$ such that $\langle s\rangle\cap\langle x^t\rangle=\langle x^t\rangle$. We use a divide-and-conquer algorithm, inspired by the half-gcd algorithm. If we applied this idea directly, the increase in degree in $y$ of the transition matrices would prevent us from getting a softly linear bound in the degree of the input; we will thus work modulo an equivalence relation, to control the size of the intermediate polynomials.

Consider the following equivalence relation on $\mathcal{S}_k$: for any two pairs $(P,Q)$ and $(P',Q')$ in $\mathcal{S}_k$, we say that $(P,Q)\sim(P',Q')$ if and only if the ideals they generate are the same. In particular, this implies that $\upsilon(P,Q)=\upsilon(P',Q')$ and that $\Pi_1(P)$ and $\Pi_1(P')$ are equal up to a constant.

Let further $\mathcal{H},\mathcal{H}'$ be two functions from $\mathcal{S}_k$ to $\mathscr{M}_2(\mathcal{R}_k)$. Extending the equivalence property to the set of functions, we say that $\mathcal{H}$ and $\mathcal{H}'$ are equivalent if for all $s\in\mathcal{S}_k$ we have $\mathcal{H}(s)\cdot s\sim\mathcal{H}'(s)\cdot s$. We still write in this case $\mathcal{H}\sim\mathcal{H}'$.

Definition 1 (Euclidean function). *We say that $\mathcal{H}:\mathcal{S}_k\to\mathscr{M}_2(\mathcal{R}_k)$ is* Euclidean *if:*

- *for all $s,s'\in\mathcal{S}_k$, if $s\sim s'$, then $\mathcal{H}(s)\cdot s\sim\mathcal{H}(s')\cdot s'$*
- *for all $s\in\mathcal{S}_k$, if $s$ is non-zero, $\upsilon(\mathcal{H}(s)\cdot s)>\upsilon(s)$.*

*We denote $\upsilon(\mathcal{H}(s).s)-\upsilon(s)$ by $\eta_{\mathcal{H}}(s)$, and we say that $\mathcal{H}$ is* online *if for all $s\in\mathcal{S}_k$:*

- *$\eta_{\mathcal{H}}(\Pi_i(s))\geq i$ for all $i\leq\eta_{\mathcal{H}}(s)$*
- *$\mathcal{H}(s)\cdot s\sim\mathcal{H}(\Pi_i(s))\cdot s$ for all $i\geq\eta_{\mathcal{H}}(s)+1$.*

Lemma 6. *The function $\mathcal{Q}$ introduced in Lemma 5 is an online Euclidean function.*

Proof. The first point was proved in Lemma 5. Now, let $s=(P,Q)$ be a non-zero element of $\mathcal{S}_k$. By construction, we have $\upsilon(\mathcal{Q}(s)\cdot s)=\upsilon(s)+\eta(s)$. In particular, $\eta(s)=\upsilon(Q\ \mathrm{rem}_y\ N(P))-\upsilon(P)\geq\upsilon(Q)-\upsilon(P)>0$. Thus, $\mathcal{Q}$ is a Euclidean function.

To prove that $\mathcal{Q}$ is online, notice that $\mathcal{Q}(s\ \mathrm{rem}_x\ x^{\upsilon(s)+i})\ \mathrm{rem}_x\ x^i=\mathcal{Q}(s)\ \mathrm{rem}_x\ x^i$. In particular for $i\leq\eta_{\mathcal{Q}}(s)$, $\mathcal{Q}(\Pi_i(s))\cdot\Pi_i(s)$ is given by

$$\begin{aligned}&\left(\mathcal{Q}(s)\ \mathrm{rem}_x\ x^i\right)\cdot\left(s\ \mathrm{rem}_x\ x^{\upsilon(s)+i}\right)\bmod x^{\upsilon(s)+i}\\&=\mathcal{Q}(s)\cdot s\mod x^{\upsilon(s)+i}=(0,0)\mod x^{\upsilon(s)+i},\end{aligned}$$

so that $\eta_{\mathcal{Q}}(\Pi_i(s))\geq i$.

If $i>\eta(s)$, we prove that there exists a matrix $\mathbf{R}$ such that $\mathcal{Q}(\Pi_i(s))=(\mathrm{Id}+x\mathbf{R})\cdot\mathcal{Q}(s)$. Indeed, if that holds, $(\mathrm{Id}+x\mathbf{R})$ is an invertible matrix over $\mathcal{R}_k$, so that $\langle\mathcal{Q}(\Pi_i(s))\cdot s\rangle=\langle\mathcal{Q}(s)\cdot s\rangle$. Moreover $\Pi_1(\mathcal{Q}(\Pi_i(s))\cdot s)=\Pi_1(\mathcal{Q}(s)\cdot s)$, which will be sufficient to conclude. Let $\mathbf{M}=\mathbf{G}_{\mathbf{D}_s\cdot s}\cdot\mathbf{D}_s$. If $i>\eta(s)$, we note by construction that $\mathbf{M}\ \mathrm{rem}_x\ x^i=\mathbf{M}$. Moreover there exists a matrix $\mathbf{R}$ such that $\mathbf{N}_{\mathbf{M}\cdot s}\ \mathrm{rem}_x\ x^i=(\mathrm{Id}+x\mathbf{R})\cdot\mathbf{N}_{\mathbf{M}\cdot s}$, so that $\mathcal{Q}(\Pi_i(s))=(\mathrm{Id}+x\mathbf{R})\cdot\mathcal{Q}(s)$. $\square$

Let $s$ be an element of $\mathcal{S}_k$ and define by recurrence the following sequence of elements of $\mathscr{M}_2(\mathcal{R}_k)$:

$$\begin{cases}\mathbf{Q}_0 &= \mathrm{Id}\\ \mathbf{Q}_{n+1} &= \mathcal{Q}(\mathbf{Q}_n\cdot s)\cdot\mathbf{Q}_n.\end{cases}\tag{1}$$

For $s$ in $\mathcal{S}_k$, the sequence $(\upsilon(\mathbf{Q}_i\cdot s))_{i\in\mathbb{N}}$ is increasing, until it reaches $\upsilon(\mathbf{Q}_i\cdot s)=k$. Thus given an integer $n$, we can define the function $\mathcal{Q}_n$ from $\mathcal{S}_k$ to $\mathscr{M}_2(\mathcal{R}_k)$ by

$$\begin{aligned}&\mathcal{Q}_n:\mathcal{S}_k\to\mathscr{M}_2(\mathcal{R}_k)\\&\qquad s\mapsto\mathbf{Q}_{i_0},\ i_0=\max\{i\in\mathbb{N}\mid\upsilon(\mathbf{Q}_i\cdot s)-\upsilon(s)\leq n\}\end{aligned}$$

for $s$ non-zero; for $s=(0,0)$, we set $\mathcal{Q}_n(s)=\mathrm{Id}$. In particular, for any $s$, $\mathcal{Q}_0(s)=\mathrm{Id}$ (since for $s$ non-zero, $\mathbf{Q}_1\cdot s=\mathcal{Q}(s)\cdot s$ has valuation greater than that of $s$).

Lemma 7. *For $n\geq 0$, if $s\sim s'$ in $\mathcal{S}_k$ then $\mathcal{Q}_n(s)\cdot s\sim\mathcal{Q}_n(s')\cdot s'$. Moreover, let $j$ be the minimal integer such that $I_j(s)=I_n(s)$; then, $\langle\mathcal{Q}_n(s)\cdot s\rangle=\langle s\rangle\cap\langle x^{\upsilon(s)+j}\rangle$.*

Proof. We use a recurrence on $n$. For $n=0$, it is clear that $\mathcal{Q}_0$ satisfies the desired properties. Then, given $n\geq 1$ and $s\sim s'$ two equivalent elements of $\mathcal{S}_k$ we know by recurrence assumption that $t:=\mathcal{Q}_{n-1}(s)\cdot s\sim t':=\mathcal{Q}_{n-1}(s')\cdot s'$.

If $\upsilon(\mathcal{Q}(t)\cdot t)>\upsilon(s)+n$, then $\upsilon(\mathcal{Q}(t')\cdot t')>\upsilon(s')+n$ and $\mathcal{Q}_n(s)=\mathcal{Q}_{n-1}(s)$ and $\mathcal{Q}_n(s')=\mathcal{Q}_{n-1}(s')$, so that $\mathcal{Q}_n(s)\cdot s\sim\mathcal{Q}_n(s')\cdot s'$. Moreover, in this case, let $j$ be the minimal integer such that $I_j(s)=I_{n-1}(s)$. For all $\ell<\eta(t)$ we have $I_\ell(t)=I_0(t)$. In particular, $\ell:=n-(\upsilon(t)-\upsilon(s))<\eta(t)$, and $I_\ell(t)=I_n(s)$ and $I_0(t)=I_{n-1}(s)$. Hence, $\langle\mathcal{Q}_n(s)\cdot s\rangle=\langle s\rangle\cap\langle x^{\upsilon(s)+j}\rangle$ and $j$ is the smallest integer such that $I_n(s)=I_{n-1}(s)=I_j(s)$.

Otherwise, $\upsilon(s)+n-1<\upsilon(\mathcal{Q}(t)\cdot t)\leq\upsilon(s)+n$, so that $\upsilon(\mathcal{Q}(t)\cdot t)=\upsilon(s)+n$. Then $\upsilon(\mathcal{Q}(t')\cdot t')=\upsilon(s')+n$. Thus $\mathcal{Q}_n(s)=\mathcal{Q}(t)\cdot\mathcal{Q}_{n-1}(s)$ and $\mathcal{Q}_n(s')=\mathcal{Q}(t')\cdot\mathcal{Q}_{n-1}(s')$. Since $\mathcal{Q}$ is a Euclidean function, $\mathcal{Q}(t)\cdot t\sim\mathcal{Q}(t')\cdot t'$ and this leads to $\mathcal{Q}_n(s)\cdot s\sim\mathcal{Q}_n(s')\cdot s'$. Moreover, let $\ell$ be the smallest integer such that $I_\ell(t)\neq I_0(t)$. We have $\langle\mathcal{Q}(t)\cdot t\rangle=\langle t\rangle\cap\langle x^{\upsilon(t)+\ell}\rangle$. According to Lemma 4 we have $\ell=\eta(t)=\upsilon(s)+n-\upsilon(t)$. In particular $n=\upsilon(t)+\ell-\upsilon(s)$, and we have $I_n(s)=I_\ell(t)$ and for all $i<n$ we have $I_i(s)\neq I_n(s)$. Thus, $n$ is the smallest integer $j$ such that $I_j(s)=I_n(s)$ and $\langle\mathcal{Q}_n(s)\cdot s)\rangle=\langle\mathcal{Q}(t)\cdot t\rangle=\langle t\rangle\cap\langle x^{\upsilon(t)+\eta(t)}\rangle$. Since $\langle t\rangle=\langle\mathcal{Q}_{n-1}(s)\cdot s\rangle=\langle s\rangle\cap\langle x^{\upsilon(s)+i}\rangle$ for some $i\leq n-1$, this leads to $\langle\mathcal{Q}_n(s)\cdot s)\rangle=\langle s\rangle\cap\langle x^{\upsilon(s)+n}\rangle$. $\square$

Our goal is to compute efficiently a mapping equivalent to $\mathcal{Q}_n$. To this effect, we introduce in Algorithm 1 a generalized version of the half-gcd algorithm [13, p. 320]. In order to control the degree in $y$ of the intermediate elements in Algorithm 1, we use a function $\varphi_{s,n}:\mathscr{M}_2(\mathcal{R}_k)\to\mathscr{M}_2(\mathcal{R}_k)$, that depends on an element $s\in\mathcal{S}_k$ and an integer $n$.

Let $s=(P,Q)$ be a pair of polynomials in $\mathcal{S}_k$, $n$ be an integer, and let $\mathbf{M}=\left(\begin{smallmatrix}A&B\\C&D\end{smallmatrix}\right)$ be a matrix of $\mathscr{M}_2(\mathcal{R}_k)$. If $Q=0$ or $\upsilon(\mathbf{M}\cdot s)=\upsilon(s)$ then we let $\varphi_{s,n}(\mathbf{M})=\mathrm{Id}$. Otherwise, neither $P$ nor $Q$ are zero and we define $\varphi_{s,n}(\mathbf{M})$ as the remainder of

$$\begin{pmatrix}L_Q(L_PA\ \mathrm{rem}_y\ N_u(Q)) & L_P(L_QB\ \mathrm{rem}_y\ N_u(P))\\ L_Q(L_PC\ \mathrm{rem}_y\ N_u(Q)) & L_P(L_QD\ \mathrm{rem}_y\ N_u(P))\end{pmatrix}$$

by $x^{n+1}$, where $L_P,L_Q$ are $\mathrm{Lc}(P),\mathrm{Lc}(Q)$ and $N_u(P),N_u(Q)$ are two polynomials with constant leading coefficients such that $N(P)=x^{\upsilon(P)}N_u(P)$ and $N(Q)=x^{\upsilon(Q)}N_u(Q)$.

Lemma 8 (Size control). *Let $s \in \mathcal{S}_k$ and assume $\mathbf{M}\cdot s$ is also in $\mathcal{S}_k$. If $\mathbf{M}\cdot s \sim \mathcal{Q}_n(s)\cdot s$, let $j = \upsilon(\mathbf{M}\cdot s) - \upsilon(s)$. Then $\varphi_{s,j}(\mathbf{M})\cdot s \in \mathcal{S}_k$ and $\varphi_{s,j}(\mathbf{M})\cdot s \sim \mathbf{M}\cdot s$.*

*Moreover the degree in $y$ of the entries in $\varphi_{s,j}(\mathbf{M})$ are bounded by $d_Q - 1$ for the first column and $d_P - 1$ for the second one, with $d_P = \deg_y(P)$ and $d_Q = \deg_y(Q)$.*

Proof. The degree bound follows from $\deg(L_P) + \deg(N_u(P)) = d_P$, and similarly for $Q$. If $Q = 0$ or $\upsilon(\mathbf{M}\cdot s) = \upsilon(s)$ then $\mathcal{Q}_n(s) = \mathrm{Id}$ for all $0 \le n < k$, and $\varphi_{s,j}(\mathbf{M}) = \mathrm{Id} = \mathcal{Q}_n(s)$.

Now let $s = (P, Q)$ a pair in $\mathcal{S}_k$ with $Q \neq 0$ and $\upsilon(\mathbf{M}\cdot s) > \upsilon(s)$ and let $\left(\begin{smallmatrix} G \\ H \end{smallmatrix}\right) = \mathbf{M}\cdot\left(\begin{smallmatrix} P \\ Q \end{smallmatrix}\right)$. By assumption, $\langle G, H\rangle = \langle P, Q\rangle \cap \langle x^{\upsilon(s)+j}\rangle$. Moreover, $\Pi_1(G, H) = (\Pi_1(G), 0)$ and Lemma 5 shows that $\deg_y(\Pi_1(G)) < d_P$.

Assume first that $P$ and $Q$ are normal. In this case, $L_P = L_Q = 1$ and we have

$$\varphi_{s,j}(\mathbf{M})\cdot\begin{pmatrix} P \\ Q \end{pmatrix} = \begin{pmatrix} G + KN_u(P)N_u(Q) + Lx^{j+1} \\ H + MN_u(P)N_u(Q) + Nx^{j+1} \end{pmatrix},$$

where $K, M$ are polynomials of $\mathcal{R}_k$ and $L, N$ are in $\langle P, Q\rangle$, so that $\upsilon(L) \ge \upsilon(s)$ and $\upsilon(N) \ge \upsilon(s)$. In this case we also know that the degrees of $G + KN_u(P)N_u(Q) + Lx^{j+1}$ and $H + MN_u(P)N_u(Q) + Nx^{j+1}$ are lower than $d_P + d_Q$. On the other hand, $G \operatorname{rem}_x x^{\upsilon(s)+j+1}$ has a degree in $y$ less than $d_P$ and $H \operatorname{rem}_x x^{\upsilon(s)+j+1} = 0$. Also, $Lx^{j+1} \operatorname{rem}_x x^{\upsilon(s)+j+1} = Nx^{j+1} \operatorname{rem}_x x^{\upsilon(s)+j+1} = 0$. Hence, $K$ and $M$ have a valuation greater than or equal to $\upsilon(s)+j+1$ and $KN_u(P)N_u(Q)$ and $MN_u(P)N_u(Q)$ are in $\langle x^{j+1}P\rangle$.

Thus $\varphi_{s,j}(\mathbf{M})\cdot\left(\begin{smallmatrix} P \\ Q \end{smallmatrix}\right) = \left(\begin{smallmatrix} G+G' \\ H+H' \end{smallmatrix}\right)$, where $G'$ and $H'$ are two polynomials of $\langle x^{j+1}P, x^{j+1}Q\rangle$. In particular, by assumption this implies that $G'$ and $H'$ belongs to the ideal $\langle xG, xH\rangle$. Thus there is an invertible matrix that sends $\left(\begin{smallmatrix} G \\ H \end{smallmatrix}\right)$ to $\left(\begin{smallmatrix} G+G' \\ H+H' \end{smallmatrix}\right)$ and $\langle G, H\rangle = \langle G + G', H + H'\rangle$ and $\Pi_1(G+G', H+H') = \Pi_1(G, H)$. In particular $\varphi_{s,j}(\mathbf{M})\cdot s \in \mathcal{S}_k$.

If $P$ and $Q$ are not normal, let $\mathbf{L}$ be the matrix $\left(\begin{smallmatrix} L_P & 0 \\ 0 & L_Q \end{smallmatrix}\right)$. In this case, $\mathbf{M}\cdot s = \mathbf{M}\cdot\mathbf{L}\cdot\left(\begin{smallmatrix} N(P) \\ N(Q) \end{smallmatrix}\right)$. Hence, using the first part of the proof on $N(s) = (N(P), N(Q))$, we have

$$\begin{aligned}\varphi_{N(s),j}(\mathbf{M}\cdot\mathbf{L})\cdot\mathbf{L}^{-1}\cdot s &= \varphi_{N(s),j}(\mathbf{M}\cdot\mathbf{L})\cdot\left(\begin{smallmatrix} N(P) \\ N(Q) \end{smallmatrix}\right) \\ &\sim \mathbf{M}\cdot\mathbf{L}\cdot\left(\begin{smallmatrix} N(P) \\ N(Q) \end{smallmatrix}\right) \sim \mathbf{M}\cdot s.\end{aligned}$$

Then, $\varphi_{s,j}(\mathbf{M}) = \varphi_{N(s),j}(\mathbf{M}\cdot\mathbf{L})\cdot\left(\det(\mathbf{L})\mathbf{L}^{-1}\right) \mod x^{j+1}$. In particular, using the same argument as above, this implies that $\varphi_{s,j}(\mathbf{M})\cdot s = \varphi_{N(s),j}(\mathbf{M}\cdot\mathbf{L})\cdot\left(\det(\mathbf{L})\mathbf{L}^{-1}\right)\cdot s \mod \langle xG, xH\rangle$. Finally, we can factor out $\det(\mathbf{L})$ and since it is invertible, this implies that $\det(\mathbf{L})\varphi_{N(s),j}(\mathbf{M}\cdot\mathbf{L})\cdot\mathbf{L}^{-1}\cdot s$ generates the ideal $\langle G, H\rangle$. Finaly, using tthe same argument as above, this leads to $\varphi_{s,j}(\mathbf{M})\cdot s \sim \mathbf{M}\cdot s$. □

Lemma 9 (Generalized half-gcd). *For $n \in \mathbb{N}$, denote by $\mathcal{H}(n, .) : s \mapsto \mathcal{H}(n, s)$ the function computed by Algorithm 1. Then, $\mathcal{H}(n, .) \sim \mathcal{Q}_n$.*

Proof. We prove by recurrence on $n$ that $\mathcal{H}(n, .) \sim \mathcal{Q}_n$. For $n = 0$, and for any $s \in \mathcal{S}_k$, we have by construction $\mathcal{H}(0, s) = \mathrm{Id}$. On the other hand, we saw that for all $s$, $\mathcal{Q}_0(s) = \mathrm{Id}$, so our claim holds. Now assume that $\mathcal{H}(i, .) \sim \mathcal{Q}_i$ for $0 \le i < n$; we prove that the equivalence holds for $n$. Let $n_0 = \lfloor \frac{n}{2} \rfloor, n_1 = n - \left(\upsilon(\tilde{t}) - \upsilon(\tilde{s})\right)$.

**Algorithm 1** Generalized version of the half-gcd

```
function GenericHalfGcd(n, s)
    s̃ ← Π_{n+1}(s)
    if n = 0 or s̃ = (0, 0) then
        return Id
    end if
    R ← GenericHalfGcd(⌊n/2⌋, s̃)
    n' ← n − (υ(R · s̃) − υ(s̃))
    ũ ← Π_{n'+1}(R · s̃)
    η ← min(η(ũ), n' + 1)
    if η > n' then
        return R
    end if
    t̃ ← Π_{n'−η+1}(Q(ũ) · ũ)
    S ← GenericHalfGcd(n − (υ(t̃) − υ(s̃)), t̃)
    M ← φ_{s̃,υ(S·t̃)−υ(s̃)}(S · Q(ũ) · R)
    return M
end function
```

For any $s, s'$, if $\Pi_{n+1}(s) = \Pi_{n+1}(s')$, then $\mathcal{H}(n, s) = \mathcal{H}(n, s')$, since then $\tilde{s} = \tilde{s}'$. In particular, with the notation of the algorithm, $\Pi_{n_0+1}(s) = \Pi_{n_0+1}(\tilde{s})$, which implies that $\mathcal{H}(n_0, \tilde{s}) = \mathcal{H}(n_0, s)$. Hence, by recurrence assumption, we get that $\mathbf{R}\cdot s = \mathcal{H}(n_0, s)\cdot s \sim \mathcal{Q}_{n_0}(s)\cdot s$.

The definition of $\mathcal{Q}_{n_0}$ implies that $\upsilon(\mathcal{Q}_{n_0}(s)\cdot s) - \upsilon(s) \le \lfloor n/2 \rfloor$, and by the claim above, we get that $\upsilon(\mathbf{R}\cdot s) - \upsilon(s) \le \lfloor n/2 \rfloor$. This implies that $\upsilon(\mathbf{R}\cdot s) = \upsilon(\mathbf{R}\cdot\tilde{s})$, and, since $n' = n - (\upsilon(\mathbf{R}\cdot\tilde{s}) - \upsilon(\tilde{s}))$, that $\Pi_{n'+1}(\mathbf{R}\cdot s) = \Pi_{n'+1}(\mathbf{R}\cdot\tilde{s}) = \tilde{u}$.

To continue, we distinguish two cases. If $\eta(\mathbf{R}\cdot s) \ge n' + 1$ then the first part of $\mathcal{Q}$ being online shows that $\eta(\Pi_{n'+1}(\mathbf{R}\cdot s)) = \eta(\tilde{u}) \ge n' + 1$. Hence, in this case, the algorithm returns $\mathbf{R}$. On the other hand, we will prove that in this case, $\mathcal{Q}_n(s) = \mathcal{Q}_{n_0}(s)$; one this is established, this implies that $\mathcal{Q}_n(s)\cdot s \sim \mathbf{R}\cdot s$, so our correctness claim holds in this case. Indeed, $\eta(\mathbf{R}\cdot s) = \eta(\mathcal{Q}_{n_0}(s)\cdot s)$ (in view of the equivalence written above, and of Lemma 4), which can be rewritten as $\upsilon(\mathcal{Q}(\mathcal{Q}_{n_0}(s)\cdot s)\cdot\mathcal{Q}_{n_0}(s)\cdot s) - \upsilon(\mathcal{Q}_{n_0}(s)\cdot s)$. On the other hand, $n' = n - (\upsilon(\mathbf{R}\cdot\tilde{s}) - \upsilon(\tilde{s}))$ gives $n' = n - \upsilon(\mathcal{Q}_{n_0}(s)\cdot s) + \upsilon(s)$. Hence, $\eta(\mathbf{R}\cdot s) \ge n' + 1$ means that

$$\upsilon(\mathcal{Q}(\mathcal{Q}_{n_0}(s)\cdot s)\cdot\mathcal{Q}_{n_0}(s)\cdot s) - \upsilon(s) \ge n + 1,$$

which precisely implies that $\mathcal{Q}_{n_0}(s) = \mathcal{Q}_n(s)$.

If $n' + 1 > \eta(\mathbf{R}\cdot s)$, $\mathcal{Q}$ being online leads to the equivalence $\mathcal{Q}(\tilde{u})\cdot\mathbf{R}\cdot s \sim \mathcal{Q}(\mathbf{R}\cdot s)\cdot\mathbf{R}\cdot s$. We claim that the right-hand side has valuation $\upsilon(\tilde{t}) = \upsilon(\mathcal{Q}(\tilde{u})\cdot\tilde{u})$. Indeed, the proof of Lemma 6 establishes the existence of a matrix $\mathbf{K}$ such that $\mathcal{Q}(\tilde{u}) = (\mathrm{Id} + \mathbf{K})\mathcal{Q}(\mathbf{R}\cdot s)$; this implies that $\upsilon(\mathcal{Q}(\tilde{u})\cdot\tilde{u}) = \upsilon(\mathcal{Q}(\mathbf{R}\cdot s)\cdot\tilde{u})$. On the other hand, the inequality $n' + 1 > \eta(\mathbf{R}\cdot s)$ also implies that $\upsilon(\mathcal{Q}(\mathbf{R}\cdot s)\cdot\tilde{u})$ and $\upsilon(\mathcal{Q}(\mathbf{R}\cdot s)\cdot\mathbf{R}\cdot s)$ are the same, which proves our claim.

Using $\mathbf{R}\cdot s \sim \mathcal{Q}_{n_0}(s)\cdot s$, and $\mathcal{Q}$ being Euclidean, we get $\mathcal{Q}(\mathbf{R}\cdot s)\cdot\mathbf{R}\cdot s \sim \mathcal{Q}(\mathcal{Q}_{n_0}(s)\cdot s)\cdot\mathcal{Q}_{n_0}(s)\cdot s$. We claim that the right-hand side is equivalent to $\mathcal{Q}_{n-n_1}(s)$, which will prove $\mathcal{Q}(\tilde{u})\cdot\mathbf{R}\cdot s \sim \mathcal{Q}_{n-n_1}(s)\cdot s$. Indeed, by definition, $\mathcal{Q}_{n-n_1}(s)$ is the last element in the sequence $(\mathbf{Q}_i\cdot s)$ from (1) having valuation at most $\upsilon(\tilde{t})$. On the other hand, the previous paragraph proves that $\mathcal{Q}(\mathcal{Q}_{n_0}(s)\cdot s)\cdot\mathcal{Q}_{n_0}(s)\cdot s$, which belongs to the sequence $(\mathbf{Q}_i\cdot s)$, has valuation $\upsilon(\tilde{t})$; this is enough to conclude, since the valuations $\upsilon(\mathbf{Q}_i\cdot s)$ increase.

Then $\mathbf{S} = \mathcal{H}(n_1, \tilde{t})$; by recurrence assumption, $\mathcal{H}(n_1, .) \sim \mathcal{Q}_{n_1}$. Moreover, $\tilde{t} = \Pi_{n_1+1}(\mathcal{Q}(\tilde{u})\cdot\tilde{u}) = \Pi_{n_1+1}(\mathcal{Q}(\tilde{u})\cdot\mathbf{R}\cdot s)$,

and by construction $\mathcal{H}(n_1, \tilde{t}) = \mathcal{H}(n_1, \mathcal{Q}(\tilde{u}) \cdot \mathbf{R} \cdot s)$, so that

$$\begin{aligned}\mathbf{S} \cdot \mathcal{Q}(\tilde{u}) \cdot \mathbf{R} \cdot s &\sim \mathcal{Q}_{n_1}(\mathcal{Q}(\tilde{u}) \cdot \mathbf{R} \cdot s) \cdot \mathcal{Q}(\tilde{u}) \cdot \mathbf{R} \cdot s \\ &\sim \mathcal{Q}_{n_1}(\mathcal{Q}_{n-n_1}(s)) \cdot \mathcal{Q}_{n-n_1}(s) \cdot s.\end{aligned}$$

The definition of the sequence $(\mathcal{Q}_n)$ implies that the latter expression is equivalent to $\mathcal{Q}_n(s) \cdot s$.

Finally, since the function $\varphi_{s,n}$ satisfies $\varphi_{s,n}(\mathbf{S} \cdot \mathcal{Q}(\tilde{u}) \cdot \mathbf{R}) \cdot s \sim \mathbf{S} \cdot \mathcal{Q}(\tilde{u}) \cdot \mathbf{R} \cdot s$, we conclude that $\mathcal{H}(n, .) \sim \mathcal{Q}_n$. □

LEMMA 10. *Let $s = (P, Q)$ be in $\mathcal{S}_k$, of degrees at most $d$. For $n > 0$, Algorithm 1 computes $\mathcal{H}(n, s)$ in time $\mathcal{O}(\mathsf{M}(dn)\log(n) + \mathsf{M}(d)n\log(d))$.*

PROOF. For $n = 0$, for any $s$, computing $\mathcal{H}(0, s)$ takes constant time. For a higher value of $n$, remark that the recursive calls are made with arguments $n_0, n_1$ that are at most $\lfloor \frac{n}{2} \rfloor$: this is clear for $n_0$; for $n_1 = n - (\upsilon(\tilde{t}) - \upsilon(\tilde{s}))$, this is because $\upsilon(\tilde{t}) - \upsilon(\tilde{s}) = \upsilon(\mathcal{Q}(\tilde{u}) \cdot u) - \upsilon(\tilde{s})$ must be greater than $\lfloor n/2 \rfloor$, by definition of $\mathcal{Q}_{n_0}$.

The matrix $\mathbf{R}$ has entries of degree at most $d$ (because $\tilde{s}$ does), so computing $\tilde{u}$ takes time $\mathcal{O}(\mathsf{M}(dn))$, and its entries have degree $\mathcal{O}(d)$. More precisely, we saw in the proof of the previous lemma that $\mathbf{R} \cdot s \sim \mathcal{Q}_{n_0}(s) \cdot s$. Because $\mathcal{Q}_{n_0}$ is obtained by iterating $\mathcal{Q}$, the last item in Lemma 5 shows that the first entry of $\mathcal{Q}_{n_0}(s) \cdot s$ has degree less than $d$; as pointed out before, this implies the same property for $\mathbf{R} \cdot s$, and thus for $\tilde{u}$ (which is a truncation of it).

Computing $\eta$ takes time $\mathcal{O}(\mathsf{M}(dn))$ by Lemma 3, and the same holds for $\tilde{t}$ by Lemma 5, up to an extra term $\mathcal{O}(\mathsf{M}(d)\log(d))$. In addition, that lemma shows that the entries of $\tilde{t}$ have degree less than that of the first entry of $\tilde{u}$, and thus less than $d$.

After the last recursive call, it remains to compute $\mathbf{M}$. We first compute $\mathbf{Q}_1 = \mathcal{Q}(\tilde{u}) \mod (N_u(P), x^\ell)$ and $\mathbf{Q}_2 = \mathcal{Q}(\tilde{u}) \mod (N_u(Q), x^\ell)$, where $\ell = \upsilon(\mathbf{S} \cdot \tilde{t}) - \upsilon(\tilde{s}) \leq n$. This can be done in time $\mathcal{O}(\mathsf{M}(dn))$: $\mathcal{Q}(\tilde{u})$ is a product of three matrices, called $\mathbf{G}, \mathbf{D}, \mathbf{N}$ in Subsection 3.1. The first two have polynomial entries of degree at most $d$, and can be computed in time $\mathcal{O}(\mathsf{M}(dn))$; the last one involves a denominator of the form $\mathrm{Lc}(U)^{-1}$, for some polynomial $U$ of degree at most $d$. The inverse of $\mathrm{Lc}(U)$ may have degree $\Omega(dk)$, but one can directly compute it modulo $N_u(P)$ or $N_u(Q)$ using Newton iteration on $x$ in time $\mathcal{O}(\mathsf{M}(dn))$.

Then we compute $\mathbf{M}_1 = \varphi_{\bar{s},\ell}(\mathbf{S} \cdot \mathbf{Q}_1 \cdot \mathbf{R})$ and $\mathbf{M}_2 = \varphi_{\bar{s},\ell}(\mathbf{S} \cdot \mathbf{Q}_2 \cdot \mathbf{R})$ and we let $\mathbf{M}$ be the concatenation of the first column of $\mathbf{M}_2$ and the second column of $\mathbf{M}_1$. Thus $\mathbf{M}$ can be computed in time $\mathcal{O}(\mathsf{M}(dn))$.

Overall, the time spent on input $(n, s)$ is $\mathcal{O}(\mathsf{M}(dn) + \mathsf{M}(d)\log(d))$, plus two recursive calls with parameter at most $\lfloor n/2 \rfloor$, in degree at most $d$. The total is thus $\mathcal{O}(\mathsf{M}(dn)\log(n) + \mathsf{M}(d)n\log(d))$. □

## 3.3 Computing pseudo-inverses

Given two polynomials $P$ and $Q$ in $\mathcal{R}_k$, we will use Algorithm 1 to compute $U$ and $V$ such that $UP + VQ = x^t \mod x^{t+1}$, where $t$ is the smallest integer such that $\langle P, Q \rangle \cap \langle x^t \rangle = \langle x^t \rangle$. In particular, the valuation of the resultant of $P$ and $Q$ is greater than or equal to $t$.

COROLLARY 1. *Assume that $P, Q \in \mathcal{R}_k$ have degree at most $d$, and let $t$ be the minimal integer such that $\langle P, Q \rangle \cap \langle x^t \rangle = \langle x^t \rangle$. It is possible to compute in time $\mathcal{O}(\mathsf{M}(dt)\log(t) + \mathsf{M}(d)t\log(d))$ two polynomials $U, V \in \mathcal{R}_k$ such that $UP + VQ = x^t \mod x^{t+1}$, with $\deg_y(U) < \deg_y(Q)$ and $\deg_y(V) < \deg_y(P)$.*

PROOF. Without loss of generality, assume that $\upsilon(P) \leq \upsilon(Q)$. Suppose first that we actually have $\upsilon(P) < \upsilon(Q)$, and define $s = (P, Q) \in \mathcal{S}_k$. In the following we let $t' = t - \upsilon(s)$.

In particular we have $I_{t'}(s) = \langle 1 \rangle$ and since for all $i < t'$, $I_i(s) \subsetneq I_{t'}(s)$, the properties of $\mathcal{Q}_{t'}$ given in Lemma 7 ensure that $\mathcal{Q}_{t'}(s) \cdot s = \langle s \rangle \cap \langle x^t \rangle = \langle x^t \rangle$. Moreover, for any integer $i \geq t'$, we have $\mathcal{Q}_i(s) = \mathcal{Q}_{t'}(s)$. Thus, by the equivalence property of $\mathcal{H}$ given in Lemma 9, we have for any $i \geq t'$:

$$\mathcal{H}(i, s) \cdot s = \begin{pmatrix} x^t(a + xW) \\ x^{t+1}H \end{pmatrix},$$

where $s = (P, Q)$ and $W, H$ are in $\mathcal{R}_k$ and $a$ is a non-zero constant. Thus it is enough to compute $\mathcal{H}(i, s)$ for any $i \geq t'$ to recover $U$ and $V$ from the first row of the matrix $\mathcal{H}(i, s)$. On the other hand for any $i < t'$, the first coordinate of $\Pi_1(\mathcal{H}(i, s) \cdot s)$ has a degree greater or equal to 1. Thus we can apply Algorithm 1 to the input $(2^i, s)$, for $i$ from 1 to $\lceil \log k \rceil$ until the first polynomial of $\Pi_1(\mathcal{H}(2^i, s) \cdot s)$ has degree 0 in $y$. We will find $2^{i_0} \geq t'$ while calling $\mathcal{H}$ at most $\lceil \log t' \rceil$ times, and this will allow us to conclude.

Suppose now that $\upsilon(P) = \upsilon(Q)$, let $s = (P, Q)$ and $s' = \mathbf{G}_s \cdot s$, where $\mathbf{G}$ is the gcd matrix defined in Subsection 3.1. Then $s'$ satisfies the assumptions of the previous paragraph, we can compute polynomials $U', V'$ such that $U'P' + V'Q' = x^t \mod x^{t+1}$, with $s' = (P', Q')$. Remark that the value of $t$ is indeed the minimal possible one for $s$ as well, since $\mathbf{G}$ is a unit; note also that the degrees of $P', Q'$ are $\mathcal{O}(d)$, and thus so are those of $U'$ and $V'$.

Multiplying $U', V'$ by $\mathbf{G}$, we obtain polynomials $U'', V''$ of degree $\mathcal{O}(d)$ such that $U''P + V''Q = x^t \mod x^{t+1}$. Then, define $U = L_Q(L_P U'' \mod N_u(Q))$ and $V = L_P(L_Q V'' \mod N_u(P))$, with $L_P, L_Q, N_u(P), N_u(Q)$ defined as in Lemma 8. These polynomials have prescribed degrees, can be computed modulo $x^{t+1}$ in time $\mathcal{O}(\mathsf{M}(dt))$, and the same proof as in Lemma 8 shows that $UP + VQ = x^t \mod x^{t+1}$. □

# 4. THE FIRST NON-ZERO COEFFICIENT

Before computing the first $k$ coefficients of the resultant $R(x) \in \mathbb{K}[x]/\langle x^k \rangle$ of two polynomials $P, Q \in \mathcal{R}_k$, we focus on computing the first non-zero coefficient of $R$. The following lemma will allow us to compute it by recurrence.

LEMMA 11. *Let $M, N \in \mathcal{R}_k$ and $U \in \mathcal{U}(\mathcal{R}_k)$ and an integer $t \leq k$ be such that $MP + NQ = x^t U$ with $\deg_y(N) < \deg_y(P)$, and such that $P$ and $N$ are normal and $\Pi_1(U) = 1$. Furthermore, assume that $\upsilon(P) = \upsilon(N) = 0$.*

*Denote by $d_P, d_Q, d_M, d_N$ the degrees in $y$ of $P, Q, M, N$ respectively, by $p_0 \in \mathbb{K}$ the coefficient of $y^{d_P}$ in $P$, and by $n_0 \in \mathbb{K}$ the coefficient of $y^{d_N}$ in $N$. Then, there exists $V$ unit in $\mathbb{K}[x]/\langle x^k \rangle$, with $\Pi_1(V) = 1$, such that:*

$$\mathrm{Res}\,(P, Q) = x^{t(d_P - d_N)}(-1)^{d_N d_P} \frac{p_0^{d_N + d_Q}}{n_0^{d_M + d_P}} V \,\mathrm{Res}\,(N, M).$$

PROOF. By the multiplication rules of the resultant,

$$\begin{aligned}\mathrm{Res}\,(P, N)\,\mathrm{Res}\,(P, Q) &= \mathrm{Res}\,(P, NQ) \\ \mathrm{Res}\,(N, P)\,\mathrm{Res}\,(N, M) &= \mathrm{Res}\,(N, MP)\end{aligned}$$

Replacing in the first equality $NQ$ by $NQ + MP = x^t U$, we get $\mathrm{Res}\,(P, NQ) = p_0^{d_N + d_Q - d_U}\,\mathrm{Res}\,(P, x^t U)$ (see for example [16, 11]). Finally, $\mathrm{Res}\,(P, x^t U) = x^{t d_P}\,\mathrm{Res}\,(P, U)$. Then since $P$ is normal of valuation zero and $\Pi_1(U) = 1$, there exists $W = 1 + x\tilde{W} \in \mathbb{K}[x]/\langle x^k \rangle$ such that $\mathrm{Res}\,(P, U) = p_0^{d_U} W$.

Applying these arguments to $\operatorname{Res}(N, MP)$, we conclude:

$$\operatorname{Res}(P,N)\operatorname{Res}(P,Q) = p_0^{d_N+d_Q} x^{td_P} W$$
$$\operatorname{Res}(N,P)\operatorname{Res}(N,M) = n_0^{d_M+d_P} x^{td_N} W'$$

Note that the symmetry formula for the resultant implies that $\operatorname{Res}(P,N) = (-1)^{d_P d_N}\operatorname{Res}(N,P)$. Then dividing the two equalities, we recover the desired result. □

LEMMA 12 (FIRST NON-ZERO COEFFICIENT). *Let $d$ be a bound on the degrees in $y$ of $P$ and $Q$. One can determine whether $R = \operatorname{Res}(P,Q)$ vanishes in $\mathbb{K}[x]/\langle x^k\rangle$, and if not compute its first non-zero coefficient and its valuation in $\mathcal{O}(\mathsf{M}(dk)\log(k) + \mathsf{M}(d)k\log(d))$ arithmetic operations.*

PROOF. Lemma 2 allows us to reduce to the case where $P$ and $Q$ are normal polynomials; the cost of this reduction is $\mathcal{O}(\mathsf{M}(dk))$. Starting from normal $P$ and $Q$, we prove the result by induction; the proof actually only uses the fact that one polynomial, say $P$, is normal. We use an integer argument $\tau$, which gives us a (strict) upper bound on the valuation of the resultant; initially, it is set to $k$.

Dividing by powers of $x$, we can assume that $v(P) = v(Q) = 0$; the upper bound $\tau$ remains valid. We then compute the resultant of $\Pi_1(P)$ and $\Pi_1(Q)$ in $\mathbb{K}[y]$, in time $\mathcal{O}(\mathsf{M}(d)\log(d))$. If it is non-zero, we are done.

Else, let $t$ be the smallest integer such that $\langle P,Q\rangle \cap \langle x^t\rangle = \langle x^t\rangle$; hence, $t \le \tau$, but we also have $t > 0$. Define function $F(d,n) = \mathsf{M}(dn)\log(n) + \mathsf{M}(d)n\log(d)$. Using Corollary 1, we see that there exists a universal constant $c_1$ such that we can compute in $c_1F(d,t)$ operations two polynomials $U$ and $V$ in $\mathcal{R}_k$ of degree less than $d$ such that $UP + VQ = x^tW$ with $W \in \mathcal{U}(\mathcal{R}_k)$, with more precisely $\deg_y(U) < \deg_y(P)$. Since $t$ is minimal, this implies $v(U,V) = 0$ and since $v(P) = 0$, this implies that $v(V) = 0$.

If $t = \tau$, we are done. Else, let $A = \Pi_{t+1}(\operatorname{Lc}(V))$, $N = \Pi_{t+1}(N_u(V))$ and $M = \Pi_{t+1}(U/A \operatorname{rem}_y N_u(Q))$. Since $A$ is a unit, these definitions imply the equality $MP + NQ = x^t(1+xY) + ZN_u(Q)P$, for some polynomials $Y$ and $Z$. The degree of the left-hand side is less than $\deg_y(P) + \deg_y(Q)$, whereas $N_u(Q)P$ is monic of degree $\deg_y(P)+\deg_y(Q)$. Hence, the previous equality shows that $ZN_u(Q)P$ vanishes modulo $x^{t+1}$. The assumptions of Lemma 11 are satisfied; we can thus do a recursive call on $N$ and $M$, with upper bound $\tau - t$, from which we can recover our output using the formula in that lemma.

In terms of complexity, all calculations giving $A, N, M$ can be done in $c_2\mathsf{M}(dt)$ operations (the only non-trivial point is the computation of $1/A \operatorname{rem}_y N_u(Q)$, which is done by Newton iteration on $x$). Hence, the runtime $G(d,\tau)$ satisfies $G(d,\tau) \le c_0\mathsf{M}(d)\log(d) + c_1F(d,t) + c_2\mathsf{M}(dt) + G(d,\tau-t)$. Using the super-linearity of $F$ in $t$ and of $\mathsf{M}$, and the definition of $F$, we deduce the overall cost $G(d,k) = \mathcal{O}(F(d,k))$. □

## 5. A DIFFERENTIAL EQUATION

Let $P$ and $Q$ be in $\mathbb{K}[x,y]$, and let $R \in \mathbb{K}[x]$ be their resultant with respect to $y$. We now prove our main result:

THEOREM 1. *If $P$ and $Q$ have degree at most $d$ and $\mathbb{K}$ has characteristic zero, or at least $k$, one can compute $R \operatorname{rem}_x x^k$ using $\mathcal{O}(\mathsf{M}(dk)\log(k) + \mathsf{M}(d)k\log(d))$ operations in $\mathbb{K}$.*

**First reduction:** If the degree in $x$ of $P$ or $Q$ is greater than or equal to $k$, let $P_k = P \operatorname{rem}_x x^k$ and $Q_k \operatorname{rem}_x x^k$ and let $d_P, d_Q$ be the degrees in $y$ of $P$ and $Q$ respectively. Then,

$$\operatorname{Res}_{d_P,d_Q}(P_k,Q_k) = \operatorname{Res}(P,Q) \mod x^k,$$

where $\operatorname{Res}_{d_P,d_Q}$ denotes the determinant of the Sylvester matrix associated to the degrees $(d_P,d_Q)$. If both leading coefficients of $P$ and $Q$ have a valuation less than $k$, then $\operatorname{Res}_{d_P,d_Q}(P_k,Q_k) = \operatorname{Res}(P_k,Q_k)$. If both leading coefficients have a valuation greater than or equal to $k$ then $\operatorname{Res}(P,Q) = 0 \bmod x^k$. Finally, if only the leading coefficient of say $Q$ has a valuation greater or equal to $k$, then we have $\operatorname{Res}_{d_P,d_Q}(P_k,Q_k) = p_0^{d_Q-\deg_y(Q_k)}\operatorname{Res}(P_k,Q_k)$, where $p_0$ is the leading coefficient of $P$ in $y$. Thus, in any case, we can recover the resultant of $P$ and $Q$ modulo $x^k$ from that of $P_k$ and $Q_k$, in a time that fits in our runtime bound.

**Second reduction:** Assume that $P$ and $Q$ have degree at most $d$ in $y$, with coefficients of degree less than $k$. Using Lemma 2, in time $\mathcal{O}(\mathsf{M}(dk))$, we can reduce the problem of computing $\operatorname{Res}(P,Q) \operatorname{rem}_x x^k$ to a similar problem with $P$ and $Q$ both monic in $y$, reduced modulo $x^k$, and with no degree increase in $y$ (that lemma proves the existence of suitable polynomials in $\mathcal{R}_k$, so we take their canonical lifts to $\mathbb{K}[x,y]$). Hence, below, we suppose we are in this case.

With the results of the previous section, we can test if $R = \operatorname{Res}(P,Q)$ vanishes modulo $x^k$, and if not, find its valuation $\mu \le k$ and the coefficient $c$ of $x^\mu$, in time $\mathcal{O}(\mathsf{M}(dk)\log k + \mathsf{M}(d)k\log d)$. We thus assume that $R \operatorname{rem}_x x^k$ is non-zero, as otherwise we are done. The key to our algorithm is the following differential equation satisfied by $R$ over $\mathbb{K}(x)[y]$; below, we write $d_P = \deg_y(P)$ and $d_Q = \deg_y(Q)$.

LEMMA 13. *The following equality holds:*

$$\frac{dR}{dx} = R\left(\operatorname{coeff}\left(\frac{1}{P}\frac{dP}{dx}\frac{dQ}{dy} \operatorname{rem}_y Q, y^{d_Q-1}\right) + \operatorname{coeff}\left(\frac{1}{Q}\frac{dQ}{dx}\frac{dP}{dy} \operatorname{rem}_y P, y^{d_P-1}\right)\right).$$

PROOF. Let $\mathbf{A}$ be the Sylvester matrix of $P$ and $Q$ with respect to the variable $y$, so that $R = \det(\mathbf{A})$. Since $R \operatorname{rem}_x x^k$ is non-zero, $\mathbf{A}$ is a unit over $\mathbb{K}(x)$.

Differentiating this equality with respect to $x$, we obtain $\frac{dR}{dx} = R\operatorname{trace}\left(\mathbf{A}^{-1}\frac{d\mathbf{A}}{dx}\right)$. Since $\mathbf{A}$ is the matrix of the mapping $(F,G) \mapsto FP + GQ$ (in the canonical monomial bases), $\frac{d\mathbf{A}}{dx}$ represents $(F,G) \mapsto F\frac{dP}{dx} + G\frac{dQ}{dx}$, in the same bases. Similarly, the inverse $\mathbf{A}^{-1}$ represents the mapping $S \mapsto (S/P \operatorname{rem}_y Q, S/Q \operatorname{rem}_y P)$. Due to the block structure of the matrix $\mathbf{A}^{-1}\frac{d\mathbf{A}}{dx}$, its trace is the trace of the block-diagonal operator

$$(F,G) \mapsto \left(F\frac{1}{P}\frac{dP}{dx} \operatorname{rem}_y Q,\ G\frac{1}{Q}\frac{dQ}{dx} \operatorname{rem}_y P\right).$$

This mapping being block-diagonal, its trace is the sum of the traces of its two components, $F \mapsto F\frac{1}{P}\frac{dP}{dx} \operatorname{rem}_y Q$ and $G \mapsto G\frac{1}{Q}\frac{dQ}{dx} \operatorname{rem}_y P$. To conclude, remark that these traces are respectively $\operatorname{coeff}\left(\frac{1}{P}\frac{dP}{dx}\frac{dQ}{dy} \operatorname{rem}_y Q, y^{d_Q-1}\right)$ and $\operatorname{coeff}\left(\frac{1}{Q}\frac{dQ}{dx}\frac{dP}{dy} \operatorname{rem}_y P, y^{d_P-1}\right)$. □

Using Corollary 1, we can compute in time $\mathcal{O}(\mathsf{M}(dk)\log(k) + \mathsf{M}(d)k\log(d))$ we know two cofactors $U$ and $V$ in $\mathbb{K}[x,y]$, of degree less than $k$ in $x$ such that $\deg(U,y) < d_Q$, $\deg(V,y) < d_P$ and

$UP + VQ = x^t \bmod x^{t+1}$, with $t$ chosen minimal, so that $t \le \mu$. Then, $t \le k$, since otherwise $R = 0 \mod x^k$.

From this, we deduce $U', V'$ with the same degree constraints on $y$ and degree less than $2t+k-1$ in $x$, such that

$$U'P + V'Q = x^t \bmod x^{2t+k-1}:$$

compute $W$ such that $UP+VQ = x^t(1+xW) \mod x^{2t+k-1}$, then the inverses $A$ (resp. $B$) of $1+xW$ modulo $\langle Q, x^{2t+k-1}\rangle$ (resp. modulo $\langle P, x^{2t+k-1}\rangle$), and let $U' = UA \operatorname{rem}_y Q$ and $V' = VB \operatorname{rem}_y P$. The only non-trivial point is the inversions, which are done by Newton iteration with respect to $x$; overall, the cost of this step is $\mathcal{O}(\mathsf{M}(dk))$.

The defining equality for $U'$ and $V'$ can be rewritten as

$$U'P + V'Q = x^t(1 + x^{t+k-1}S),$$

for some $S$ in $\mathbb{K}[x, y]$. Their degree constraints then show that the inverse of $P$ modulo $Q$ is $U'/(x^t(1 + x^{t+k-1}S)) \operatorname{rem}_y Q$; similarly, the inverse of $Q$ modulo $P$ is $V'/(x^t(1 + x^{t+k-1}S)) \operatorname{rem}_y P$. This further implies that $x^t \frac{dR}{dx}$ is equal to

$$R\left(\operatorname{coeff}\left(\frac{U'}{1+x^{t+k-1}S}\frac{dP}{dx}\frac{dQ}{dy} \operatorname{rem}_y Q, y^{d_Q-1}\right) + \operatorname{coeff}\left(\frac{V'}{1+x^{t+k-1}S}\frac{dQ}{dx}\frac{dP}{dy} \operatorname{rem}_y P, y^{d_P-1}\right)\right).$$

Taking this equality modulo $x^{t+k-1}$, we obtain a relation of the form $x^t \frac{dR}{dx} = RF \bmod x^{t+k-1}$, with

$$F = \operatorname{coeff}\left(U'\frac{dP}{dx}\frac{dQ}{dy} \operatorname{rem}_y Q, y^{d_Q-1}\right) + \operatorname{coeff}\left(V'\frac{dQ}{dx}\frac{dP}{dy} \operatorname{rem}_y P, y^{d_P-1}\right) \bmod x^{t+k-1}.$$

Because $P$ and $Q$ are both monic in $y$, once $U'$ and $V'$ are known, we can compute $F$ using $\mathcal{O}(\mathsf{M}((t+k)d))$ operations in $\mathbb{K}$, which is $\mathcal{O}(\mathsf{M}(kd))$.

Recall that $R$ is has the form $cx^\mu + \cdots$, for some $\mu < k$, so that $\frac{dR}{dx}$ has the form $\mu c x^{\mu-1} + \cdots$. Thus, the Laurent series $\frac{1}{R}\frac{dR}{dx}$ has valuation at least $-1$, so that $F$ has valuation at least $t-1$. Dividing by $x^{t-1}$ on both sides, we obtain $x\frac{dR}{dx} = R\tilde{F} \bmod x^k$, with $\tilde{F} = F/x^{t-1}$. From now on, let us assume that the characteristic $p$ of the base field is at least equal to $k$, or zero. Then, this relation determines $R \bmod x^k$ up to a constant factor, and knowing the initial condition $c$ allows us to deduce $R \bmod x^k$ unambiguously. Given $\tilde{F}$, this is done by means of [3, Theorem 2], which allows us to compute $R \bmod x^k$ in $\mathcal{O}(\mathsf{M}(k))$ operations in $\mathbb{K}$. Summing all the costs seen so far concludes the proof of our theorem.

**Acknowledgements.** The authors sincerely thank the reviewers for their careful reading and suggestions. Moroz is supported by project Singcast (ANR-13-JS02-0006); Schost is supported by NSERC.

# 6. REFERENCES

[1] P. Aubry and A. Valibouze. Using Galois ideals for computing relative resolvents. *J. Symb. Comp.*, 30(6):635–651, 2000.

[2] E. Berberich, P. Emeliyanenko, and M. Sagraloff. An elimination method for solving bivariate polynomial systems: Eliminating the usual drawbacks. In *ALENEX*, pages 35–47. SIAM, 2011.

[3] A. Bostan, M. F. I. Chowdhury, R. Lebreton, B. Salvy, and É. Schost. Power series solutions of singular $(q)$-differential equations. In *ISSAC'12*, pages 107–114. ACM, 2012.

[4] A. Bostan, P. Flajolet, B. Salvy, and É. Schost. Fast computation of special resultants. *Journal of Symbolic Computation*, 41(1):1–29, 2006.

[5] Y. Bouzidi, S. Lazard, G. Moroz, M. Pouget, and F. Rouillier. Improved algorithm for computing separating linear forms for bivariate systems. In *ISSAC'14*, pages 75–82. ACM, 2014.

[6] Y. Bouzidi, S. Lazard, G. Moroz, M. Pouget, F. Rouillier, and M. Sagraloff. Improved algorithms for solving bivariate systems via Rational Univariate Representations. Research report, Inria, 2015.

[7] Y. Bouzidi, S. Lazard, M. Pouget, and F. Rouillier. Rational univariate representations of bivariate systems and applications. In *ISSAC'13*, pages 109–116. ACM, 2013.

[8] Y. Bouzidi, S. Lazard, M. Pouget, and F. Rouillier. Separating linear forms for bivariate systems. In *ISSAC'13*, pages 117–124. ACM, 2013.

[9] D. G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Informatica*, 28(7):693–701, 1991.

[10] X. Caruso. Resultants and subresultants of $p$-adic polynomials. abs/1507.06502, 2015.

[11] D. Cox, J. Little, and D. O'Shea. *Using algebraic geometry.* Graduate Texts in Mathematics. Springer-Verlag, 1998.

[12] P. Emeliyanenko and M. Sagraloff. On the complexity of solving a bivariate polynomial system. In *ISSAC'12*, pages 154–161. ACM, 2012.

[13] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra.* Cambridge University Press, third edition, 2013.

[14] J. von zur Gathen and T. Lücking. Subresultants revisited. *Theoretical Computer Science*, 297(1–3):199–239, 2003.

[15] J. von zur Gathen and V. Shoup. Computing Frobenius maps and factoring polynomials. *Computational Complexity*, 2(3):187–224, 1992.

[16] I. Gelfand, M. Kapranov, and A. Zelevinsky. *Discriminants, Resultants, and Multidimensional Determinants.* Modern Birkhäuser Classics. Birkhäuser Boston, 2008.

[17] M. Giusti, G. Lecerf, and B. Salvy. A Gröbner free alternative for polynomial system solving. *J. Complexity*, 17(2):154–211, 2001.

[18] L. González-Vega and M. E. Kahoui. An improved upper complexity bound for the topology computation of a real algebraic plane curve. *J. Complexity*, 12(4):527 – 544, 1996.

[19] R. Imbach, G. Moroz, and M. Pouget. Numeric certified algorithm for the topology of resultant and discriminant curves. abs/1412.3290, 2014.

[20] E. Kaltofen and G. Villard. On the complexity of computing determinants. *Comput. Complexity*, 13(3-4):91–130, 2004.

[21] D. E. Knuth. The analysis of algorithms. In *Congrès int. Math., Nice, France*, volume 3, pages 269–274, 1970.

[22] A. Kobel and M. Sagraloff. On the complexity of computing with planar algebraic curves. *Journal of Complexity*, 31(2):206–236, 2015.

[23] R. Lebreton and É. Schost. Algorithms for the universal decomposition algebra. In *ISSAC'12*, pages 234–241. ACM, 2012.

[24] F. Lehobey. Resolvent computations by resultants without extraneous powers. In *ISSAC'97*, pages 85–92. ACM, 1997.

[25] D. R. Musser. Multivariate polynomial factorization. *J. Assoc. Comput. Mach.*, 22:291–308, 1975.

[26] D. Reischert. Asymptotically fast computation of subresultants. In *ISSAC'97*, pages 233–240. ACM, 1997.

[27] A. Schönhage. Schnelle Berechnung von Kettenbruchentwicklungen. *Acta Inform.*, 1:139–144, 1971.

[28] L. Soicher. The computation of Galois groups. Master's thesis, Concordia University, 1981.