Division-Free Computation of Subresultants Using Bezout Matrices

Michael Kerber

MPI–I–2006–1-006 May 2006

Authors' Addresses

Michael Kerber
Max-Planck-Institut für Informatik
Stuhlsatzenhausweg 85
66123 Saarbrücken
`mkerber@mpi-sb.mpg.de`

Acknowledgements

I thank Arno Eigenwillig for pointing me to this problem, for valuable discussions about it, and for carefully proofreading this report.

Abstract

We present an algorithm to compute the subresultant sequence of two polynomials that completely avoids division in the ground domain, generalizing an algorithm from Abdeljaoued et al. (see Abdeljaoed et al.: Minors of Bezout Matrices..., Int. J. of Comp. Math. 81, 2004). This is done by evaluating determinants of slightly manipulated Bezout matrices using an algorithm of Berkowitz. Experiments show that our algorithm is superior compared to pseudo-division approaches for moderate degrees if the domain contains indeterminates.

Keywords

Bezout matrix, Subresultant, Polynomial gcd

1 Introduction

The *subresultants* of two polynomials over a domain $\mathbb{D}$ correspond to the elements of a euclidean remainder sequence up to a scalar factor. Since the coefficients of the subresultants never leave the ground domain $\mathbb{D}$ and their size grows moderately compared to naive pseudo-division, they have become a standard tool in Computer Algebra, covered by textbooks ([GCL92], [Yap00], [BPR03]). The usual computation strategy for the subresultants is to iteratively perform pseudo-division, but by dividing out scalars in each iteration, the coefficient size is prevented from growing too fast.

Abdeljaoued et al. ([ADG04]) use a different approach to get the *principal subresultant coefficients*: They build a modified version of the *Bezout matrix* where the principal subresultant coefficients correspond to the determinants of the $k \times k$ upper-left submatrices (also known as *leading principal minors* or *corners*). The *Berkowitz algorithm* (or Samuelson-Berkowitz method, [Be84]) computes the determinant of a matrix without divisions, and all leading principal minors are obtained as a by-product. Consequently, only one determinant evaluation is necessary to get all principal subresultant coefficients.

The contribution of this work is to generalize the approach of [ADG04] such that the whole subresultant sequence is obtained. Therefore, for some k, matrices $M_1, \ldots, M_k$ of dimension $n, n-1, \ldots, n-k+1$ (where n is the maximal degree of the input polynomials) are created which arise from the hybrid Bezout matrix by only negating, swapping and deleting columns. These matrices contain all subresultant coefficients as leading principal minors, so evaluating the determinants of these k matrices with the Berkowitz method produces all needed coefficients.

Furthermore, an efficient method is given to compute the first s coefficients of the subresultants. More precisely, if the mth subresultant is $\sum_{k=0}^{m} a_k x^k$, the coefficients $a_m, \ldots, a_{m-s+1}$ are returned. Hence by choosing different values for s, only principal subresultant coefficients can be calculated as well as the entire subresultants, and also the computation of initial terms of the subresultant polynomials is possible (an application for the case $s = 2$ is described in Section 5).

If only the number of arithmetic operations is regarded, the discussed method is clearly asymptotically inferior to the division-based solutions. But counting operations in $\mathbb{D}$ might not be the right measure, for instance if $\mathbb{D}$ is itself a polynomial ring with a lot of variables. In such cases, a division can be a non-trivial issue in practice and the division-free approach seems to be the best choice.

The paper is structured as follows: In Section 2, the definition and the most important properties of subresultants are repeated very briefly and their relation to the Bezout matrices is discussed. Section 3 introduces the algorithm to compute the leading principal minors of a matrix. Then, Section 4 describes how to manipulate the Bezout matrix and proves the main results of this work. In Section 5, the algorithm is presented, and some applications are discussed. At the end, the running time is analysed and experimental results are presented (Section 6).

2 Subresultants and Bezout matrices

Throughout this article, let $\mathbb{D}$ be an integral domain and

$$f = \sum_{k=0}^{n} f_k x^k \qquad g = \sum_{k=0}^{m} g_k x^k$$

polynomials in $\mathbb{D}[x]$ with $n = \deg(f) \geq \deg(g) = m$.

2.1 Subresultants and the gcd

The *ith Sylvester submatrix* of f and g is defined to be the following $(n + m - 2i) \times (n + m - i)$ matrix, composed of $m - i$ rows of coefficients of f and $n - i$ rows of coefficients of g:

$$\mathrm{Syl}_i(f, g) = \begin{pmatrix} f_n & & \cdots & f_0 & & \\ & \ddots & & & \ddots & \\ & & f_n & & \cdots & f_0 \\ g_m & & \cdots & g_0 & & \\ & \ddots & & & \ddots & \\ & & g_m & & \cdots & g_0 \end{pmatrix}$$

$\mathrm{Syl}_0(f, g)$ is also called the *Sylvester matrix*.
The ith subresultant of f and g is a polynomial with minors of the Sylvester matrix as coefficients. These minors have a special form – we call them $M_{k,t}$*-minors*:

Definition 2.1. For any $p \times q$-matrix B with $p \leq q$, we define $M_{k,t}(B)$ with $1 \leq k \leq p$ and $0 \leq t \leq q - k$ to be the $k \times k$ minor of B extracted from the first k rows of B, the first $k - 1$ columns of B and the $(k + t)$th column of B. $M_{k,0}(B)$ is called *kth leading principal minor of B*.

Definition 2.2. Set $S := \mathrm{Syl}_i(f, g)$

1. For $i \in \{0, \ldots, m-1\}$, the *ith subresultant* of f and g is defined as the polynomial
$$\mathrm{Sres}_i(f,g) := \sum_{j=0}^{i} M_{n+m-2i,i-j}(S) \cdot x^j.$$
2. The *ith principal subresultant coefficient (psc)* is given by
$$\mathrm{sres}_i(f,g) := \mathrm{coef}_i(\mathrm{Sres}_i(f,g)) = M_{n+m-2i,0}(S).$$

Some authors call the psc's "subresultants" (or *scalar subresultants*) which might confuse the reader. [GL03] contains an overview how the term "subresultants" is used in the literature.
The psc's arise naturally from the question whether there exist non-zero $u, v \in \mathbb{D}[x]$ with $\deg(u) \leq m-i-1$, $\deg(v) \leq n-i-1$ and and $\deg(uf + vg) < i$. The last condition is equivalent to the vanishing of the coefficients $x^i, \ldots, x^{n+m-i-1}$ of $uf + vg$. Considering the coefficients of u,v as indeterminates leads to a linear system whose matrix is built from the first $(n+m-2i)$ rows and columns of $\mathrm{Syl}_i(f,g)$. Thus such an solution exists if and only if $\mathrm{sres}_i(f,g)$ is zero.
Consequently, the psc's determine the degree of the gcd of f and g with the following well-known property:
$$\deg(\gcd(f,g)) \quad = \quad \min\{k \in \{0,\ldots,m-1\} \mid \mathrm{sres}_k(f,g) \neq 0\}. \quad (2.1)$$
and $\deg(\gcd(f,g)) = m$ if all psc's are zero (this happens iff g divides f).

The next equation expresses the subresultant as one single determinant. In [BT71], this is taken as the definition of the subresultant:
$$\mathrm{Sres}_i(f,g) \quad = \quad \det \begin{pmatrix} f_n & \cdots & \cdots & f_{2i-m+2} & x^{m-i-1}f \\ & \ddots & & \vdots & \vdots \\ & f_n & \cdots & f_{i+1} & f \\ g_m & \cdots & \cdots & g_{2i-n+2} & x^{n-i-1}g \\ & \ddots & & \vdots & \vdots \\ & g_m & \cdots & g_{i+1} & g \end{pmatrix} \quad (2.2)$$
with $f_j = 0 = g_j$ for negative indices j. In fact, using the linearity of the determinant in the last column proves the equality at once.
On the other hand, Laplace expansion on the last column in (2.2) yields:
$$\mathrm{Sres}_i(f,g) = uf + vg, \quad \deg(u) \leq m-i-1 \quad \deg(v) \leq n-i-1 \quad (2.3)$$
Let $k := \deg(\gcd(f,g))$ now. From (2.1), $\mathrm{Sres}_k(f,g)$ is a polynomial of degree k and with (2.3), it follows at once that $\mathrm{Sres}_k(f,g) = \gcd(f,g)$.

2.2 Bezout matrices

Although evaluating minors of the Sylvester matrix is possible, we won't use it for our computation because of its high dimension. Fortunately, there exists also a matrix of dimension n in which all subresultant coefficients appear (up to sign and powers of the leading coefficient) as $M_{k,t}$ minors:

Definition 2.3. The *(symmetric) Bezout matrix* of f and g is the symmetric matrix

$$\mathrm{Bez}(P, Q) = (c_{ij})_{0 \leq i,j \leq n-1}$$

whose entries are defined by the Cayley expression

$$\frac{f(x)g(y) - g(y)f(x)}{x - y} = \sum_{i,j=0}^{n-1} c_{ij} x^i y^j.$$

The algebraic relations between the Sylvester and the Bezout matrix are studied in [CGZ99] and [CGZ02]. The latter work also comes up with an efficient algorithm to compute the entries of the Bezout matrix.

Definition 2.4. Define

$$\begin{aligned} H_i \;:=\; & (f_p x^{i-1} + \ldots + f_{n-i+1})(g_{m-i} x^{n-i} + \ldots + g_0 x^{n-m}) \\ & -(f_{n-i} x^{n-i} + \ldots + f_0)(g_m x^{i-1} + \ldots + g_{m-i+1}). \end{aligned}$$

The *hybrid Bezout matrix* of f and g is defined as the $n \times n$ matrix

$$\mathrm{HBez}(f, g) = (c_{ij})_{1 \leq i,j \leq n}$$

with

$$c_{ij} = \begin{cases} \mathrm{coef}_{n-j}(H_{m-i+1}) & \text{for } 1 \leq i \leq m \\ \mathrm{coef}_{n-j}(x^{n-i} g) & \text{for } m+1 \leq i \leq n \end{cases}$$

In other words, the first m rows of $\mathrm{HBez}(f, g)$ correspond to polynomials k_i, whereas the last $n - m$ rows only depend on the coefficients of g and are equal to rows of the Sylvester matrix. Note that this is different from the hybrid resultant matrices defined in [CGZ99].

As an example, set $f := 3x^4 - x^2 + 6x + 1$, $g := 2x^2 + 2x + 3$. Then

$$\mathrm{Bez}(f, g) = \begin{pmatrix} 16 & -5 & 0 & 9 \\ -5 & -14 & 9 & 6 \\ 0 & 9 & 6 & 6 \\ 9 & 6 & 6 & 0 \end{pmatrix}, \quad \mathrm{HBez}(f, g) = \begin{pmatrix} 11 & -10 & -14 & -2 \\ 6 & 11 & -12 & -2 \\ 2 & 2 & 3 & 0 \\ 0 & 2 & 2 & 3 \end{pmatrix}.$$

The term "Bezout matrix" is used in the literature according to our Definition 2.3 ([CGZ02],[DG04],[ADG04]) as well as for the hybrid Bezout matrix ([HW00]). The MAPLE command "BezoutMatrix" also computes the hybrid version by default. Throughout this work, we consequently use the adjectives "symmetric" and "hybrid" to clarify which version is meant.
As for the Sylvester matrix, the determinant of the symmetric and hybrid Bezout matrix vanishes if and only if f and g have a non-constant common factor. This property was already known to Bezout who introduced the symmetric Bezout matrix for the same purpose. A recent result due to Diaz-Toca and Gonzalez-Vega ([DG04]) shows the relation between the Bezout matrices and the subresultants. Hou and Wang ([HW00]) provide another proof for the hybrid Bezout matrix:

Theorem 2.5. Define the matrices:

$$\begin{aligned} B &:= J_n \cdot \mathrm{Bez}(f,g) \cdot J_n \\ H &:= J_n \cdot \mathrm{HBez}(f,g) \end{aligned}$$

where J_n is the backward identity of dimension n. Then, for $k \in \{n-m+1,\ldots,n\}$, it holds that

$$\begin{aligned} (-1)^{k(k-1)/2} f_n^{n-m} \mathrm{Sres}_{n-k}(f,g) &= \sum_{t=0}^{n-k} M_{k,t}(B) x^{n-k-t} \\ (-1)^{k(k-1)/2} \mathrm{Sres}_{n-k}(f,g) &= \sum_{t=0}^{n-k} M_{k,t}(H) x^{n-k-t} \end{aligned}$$

Again, the subresultant coefficients appear as $M_{k,t}$-minors of special matrices. The formulas also show one advantage of the hybrid Bezout matrix: It provides the subresultants directly (up to sign) whereas the for the symmetric Bezout matrix, the polynomials need to be divided by the leading coefficient of f raised to the power of the degree difference of f and g.

3 The Berkowitz method

[HW00] already propose to compute the Bezout minors directly as an alternative to the known pseudo-division process. However, the question remains how to compute the minors efficiently. Berkowitz ([Be84]) presents an algorithm that computes the determinant of an n-dimensional matrix without divisions in the ground domain. Let $\mathrm{P}_A(\lambda) := \det(\lambda \mathbb{I}_r - A) \in \mathbb{D}[\lambda]$ denote the characteristic polynomial of a square matrix $A \in \mathbb{D}^{r\times r}$. Moreover, for

any polynomial $h = \sum_{i=0}^{n} h_i x^i$, define $\overline{h} = (h_n, \ldots, h_0)^T$ to be its coefficient vector.
The Berkowitz method is based on the following formula, due to Samuelson [Sa42]:

Theorem 3.1. Let A be an $r+1$-dimensional square matrix over the domain $\mathbb{D}$ partitioned as

$$A = \begin{pmatrix} A' & S \\ R & a_{r+1,r+1} \end{pmatrix}$$

with $A' \in \mathbb{D}^{r\times r}, R \in \mathbb{D}^{r\times 1}, S \in \mathbb{D}^{1\times r}$. Define the matrix $T_A \in \mathbb{D}^{(r+2)\times(r+1)}$ as:

$$T_A = \begin{pmatrix} 1 & 0 & \ldots & 0 \\ -a_{r+1,r+1} & 1 & \ldots & 0 \\ -RS & -a_{r+1,r+1} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -R(A')^{r-2}S & -R(A')^{r-3}S & \ldots & 1 \\ -R(A')^{r-1}S & -R(A')^{r-2}S & \ldots & -a_{r+1,r+1} \end{pmatrix}$$

Then

$$\overline{P_A} = T_A \cdot \overline{P_{A'}}.$$

Proofs can be found in [Sa42] and [ADG04]. [Ro01] gives an alternative combinatorial proof.

This theorem immediately implies an algorithm for the characteristic polynomial of A: Decompose A, compute the characteristic polynomial $P_{A'}$ of the submatrix A', compute the matrix T_A and return $T_A \cdot P_{A'}$. The recursion ends since for 1-dimensional matrices, the characteristic polynomial is simply $x - a_{1,1}$. In each recursion step, the characteristic polynomial of a $k \times k$ upper-left submatrix of A is calculated, and the characteristic polynomial contains the determinant as its constant factor (up to sign). Therefore, Theorem 3.1 in particular can be used to compute the leading principal minors of a matrix.
Note that the computation of the entries of T_A does not require to multiply two square matrices. Instead, one computes the vectors $A'S, (A')^2S, \ldots, (A')^{r-1}S$ and multiplies the results with R (following the notation of Theorem 3.1). So, the leading principal minors can be calculated with $O(n^4)$ additions and multiplications over $\mathbb{D}$.
For the sequence of psc's, the leading principal minors of one matrix suffice, as exhibitet in [ADG04]. However, for the other subresultant coefficients, more $M_{k,t}$-minors are necessary, according to Theorem 2.5. Is it possible to create matrices such that all subresultant coefficients can be obtained only with the Berkowitz method?

4 Computation of $M_{k,t}$-minors

Throughout this section, fix B to be a square matrix of dimension n.
Fixing a number $s \in \mathbb{N}$, matrices $(S_{i,s}(B))_{i=1}^{s}$ will be defined that contain each minor $M_{k,t}(B)$ with $k \in \{1, \ldots, n\}, t \leq \min\{s-1, n-k\}$ as a kth leading principal minor. The matrices $S_{i,s}(B)$ are built from B with three basic operations: swapping two columns, deleting the last row and column, and multiplying a column with -1. These operations can be represeneted by sequences of integers:

Definition 4.1. Let B be a $n \times n$-matrix.

1. For $i \in \{1, \ldots, n\}$, let B_i be the ith column of B and $B_{-i} := -B_i$.
2. For a sequence $[a_1, \ldots, a_k]$ in $\{\pm 1, \pm 2, \ldots, \pm n\}$, define $B[a_1, \ldots, a_k]$ to be the matrix extracted from the first k rows of the matrix $(B_{a_1}, \ldots, B_{a_k})$. If a matrix is of that form, we call the defining integer sequence the *column sequence* of the matrix.

With that notation, one can also rewrite:

$$M_{k,t}(B) = \det B[1, \ldots, k-1, k+t]$$

Now, fix some $s \leq n$. The problem is to find s matrices that contain all $M_{k,t}$ minors of B with $t < s$ as leading principal minors. We will define such matrices by defining their column sequence. First of all, a upper triangular $n \times n$ matrix $Y = (y_{i,j})$ is defined by $y_{k,k+t} := M_{k,t}(B)$, that is

$$Y = \begin{pmatrix} \det B[1] & \det B[2] & \det B[3] & \ldots & \det B[n] \\ & \det B[1,2] & \det B[1,3] & \ldots & \det B[1,n] \\ & & \det B[1,2,3] & \ldots & \det B[1,2,n] \\ & & & \ddots & \vdots \\ & & & & \det B[1,\ldots,n] \end{pmatrix} \tag{4.1}$$

It is easy to see that the $M_{k,t}$-minors with $t < s$ are arranged in the main diagonal and the first $s-1$ superdiagonals of Y.
Consider the last column of Y: The minors of interest are the last s elements of the column, namely $y_{n,n} = \det B[1, \ldots, n-1, n], y_{n-1,n} = \det B[1, \ldots, n-2, n], \ldots, y_{n-s+1,n} = \det B[1, \ldots, n-s, n]$. It is possible to create a matrix with those as leading principal minors by a column sequence of B: Start with the matrix B which clearly has $y_{n,n}$ as nth leading principal minor, but none of the other column entries. Now, swap the two last columns of B, and negate the column that has become last, resulting in $B[1, \ldots, n-2, n, -(n-1)]$.

It is obvious that $y_{n-1,n}$ is the $(n-1)$st leading principal minor, and by the well-known equation

$$\det B[\ldots, a_i, a_{i+1}, \ldots] \quad = \quad \det B[\ldots, -a_{i+1}, a_i, \ldots]. \tag{4.2}$$

$y_{n,n}$ is still the nth leading principal minor. For the $(n-2)$nd minor, again shift the nth column of B to the left and negate the right-shifted column. After $s-1$ repetitions, the resulting matrix $S' := B[1, \ldots, n-s, n, -(n-s+1), \ldots, -(n-1)]$ contains $y_{n,n}, \ldots, y_{n-s+1,n}$ as its leading principle minors of sizes $n, \ldots, n-s+1$, as desired.
Now, we consider the $(n-s)$th column of Y: From this column, the entries $y_{n-s,n-s}, \ldots, y_{n-2s+1,n-s}$ must appear as leading principal minors of sizes $n-s, \ldots, n-2s+1$. The first of them, $y_{n-s,n-s}$, is already such a minor of S', and with the same trick as above, S' can be manipulated such that all desired entries in the column appear as leading principal minors. The result is

$$B[1, \ldots, n-2s, n-s, -(n-2s+1), \ldots, -(n-s-1), n, -(n-s+1), \ldots, -(n-1)].$$

Note that the leading principal minors of sizes $n, \ldots, n-s+1$ are not affected by this column operations because the entry $n-s$ can be shifted $s-1$ times to the right, eating up all minus signs on its way with (4.2). Iterating this construction, all columns j of Y are covered with $j \equiv n \mod s$. Furthermore, the structure of the column sequence of S consists of several blocks of length s (and one starting block which is possibly smaller). Every block starts with the biggest element, followed by a sequence of negated consecutive columns.

Definition 4.2. For two sequences x and y, $x \cdot y$ denotes the concatentation of x and y.

1. For $p, q \in \mathbb{N}$ define:

$$H_{p,q} = \begin{cases} [p, -1, -2, \ldots, -(p-1)] & \text{for } p < q \\ [p, -(p-q+1), -(p-q+2), \ldots, -(p-1)] & \text{for } p \geq q \end{cases}$$

2. Let B be a square matrix with dimension n.
   For $j, s \in \{1, \ldots, n\}$ with $j \leq s$ define

$$S_{j,s}(B) := B\left(\prod_{\lambda=0}^{N_{j,s}} H_{j+s\lambda,s}\right)$$

   where $N_{j,s} := \lfloor \frac{n-j}{s} \rfloor$.

Example: Let B be any square matrix of dimension 10. For $s = 4$, the matrices $S_{j,s}(B)$ are

$$\begin{aligned}
S_{1,4}(B) &= B[\underbrace{1}_{=H_{1,4}}, \underbrace{5,-2,-3,-4}_{=H_{5,4}}, \underbrace{9,-6,-7,-8}_{=H_{9,4}}],\\
S_{2,4}(B) &= B[\underbrace{2,-1}_{=H_{2,4}}, \underbrace{6,-3,-4,-5}_{=H_{6,4}}, \underbrace{10,-7,-8,-9}_{=H_{10,4}}],\\
S_{3,4}(B) &= B[\underbrace{3,-1,-2}_{=H_{3,4}}, \underbrace{7,-4,-5,-6}_{=H_{7,4}}],\\
S_{4,4}(B) &= B[\underbrace{4,-1,-2,-3}_{=H_{4,4}}, \underbrace{8,-5,-6,-7}_{=H_{8,4}}].
\end{aligned}$$

As a special case, observe that $S_{1,1}(B) = B$.

By construction, the matrices $S_{j,s}(B)$ are square matrices of dimension at most n. More precisely:

Lemma 4.3. $S_{j,s}(B)$ is a square matrix of dimension $j + s\lfloor\frac{n-j}{s}\rfloor$.

Proof: By definition, the dimension of $S_{j,s}(B)$ equals the length of its sequence. Each block $H_{j+s\lambda,s}$ has length s, except the first one which has length j and there are $\lfloor\frac{n-j}{s}\rfloor + 1$ blocks all together. $\square$

In particular, for fixed s, the matrices $(S_{j,s}(B))_{j=1}^{s}$ have dimensions $(n - s + 1), (n - s + 2), \ldots, n$, but not necessarily in this order.

Now, what is the correct S-matrix to detect a given $M_{k,t}$ minor of B with $t < s$? Considering the matrix Y from (4.1), the minor appears in the $(k+t)$th column and the leading principal minors of $S_{j,s}$ contain the entries in all columns c with $j \equiv c \mod s$ as stated above. The next theorem gives a formal proof. For $p, s \in \mathbb{N}$, let $\overline{p}$ be the unique representative of p in $\{1, \ldots, s\}$ modulo s.

Theorem 4.4. For $s \in \{1, \ldots, n\}$. $k \in \{1, \ldots, n\}$ and $t \in \{0, \ldots, \min\{s - 1, n-k\}\}$, $M_{k,t}(B)$ is the kth leading principal minor of $S_{j,s}(B)$ for $j = \overline{k+t}$.

Proof. $k + t$ is the first element of some block H in the column sequence of $S_{j,s}(B)$. Let p denote the position of $k + t$ in that sequence. It is easy to see that $p = 1$, if H is the first block of the sequence, and $p = (k + t) - s + 1$ otherwise. In either case, $p \leq k$ holds.
It follows that $k + t$ is also contained in the column sequence $[c_1, \ldots, c_k]$ of the $k \times k$ upper-left submatrix of $S_{\overline{k+t},s}(B)$. By construction of $S_{j,s}(B)$, it

holds that $\{|c_1|, \ldots, |c_k|\} = \{1, \ldots, k-1, l\}$ for some $l \geq k$. Applying (4.2) in each block yields

$$\det B[c_1, \ldots, c_k] = \det B[1, \ldots, k-1, l].$$

Since $k+t$ is contained in the column sequnce and since $k+t > k-1$, the result follows. $\square$

Example: Let B be of dimension 9 and $s = 6$. We want to compute the minor $M_{5,4}(B) = \det B[1, 2, 3, 4, 9]$.
Since $\overline{t+k} = 3$, $S_{3,6}(B)$ is the right S-matrix according to Theorem 4.4, and indeed, the 5th leading principal minor of $S_{3,6}(B)$ is:

$$\det B[3, -1, -2, 9, -4] = \det B[1, 2, 3, 4, 9]$$

$\square$

As the matrices $(S_{j,s})_{j=1}^{s}$ have sizes $n-s+1, \ldots, n$, the total number of leading principal minors is $\sum_{i=n-s+1}^{n} i = ns - \frac{s(s-1)}{2}$. This equals the number of pairs (k,t) satisfying $k \in \{1, \ldots, n\}$ and $t \in \{0, \ldots, \min\{s-1, n-k\}\}$. Thus, each $M_{k,t}$-minor appears exactly once among the leading principal minors of the $S_{j,s}$ matrices.

5 Algorithms and applications

To link the results of Section 4 with the subresultants, consider the upper triangular $m \times m$-matrix Z with $z_{i,j} = \mathrm{coef}_{m-j}(\mathrm{Sres}_{m-i})$. With Theorem 2.5, Z can be written (up to sign) as

$$Z = \begin{pmatrix} M_{n-m,0}(X) & M_{n-m,1}(X) & M_{n-m,2}(X) & \ldots & M_{n-m,m}(X) \\ & M_{n-m+1,0}(X) & M_{n-m+1,1}(X) & \ldots & M_{n-m+1,m-1}(X) \\ & & M_{n-m+2,0}(X) & \ldots & M_{n-m+2,m-2}(X) \\ & & & \ddots & \vdots \\ & & & & M_{n,0}(X) \end{pmatrix}$$

where $X = B$ or $X = H$ using the notation of Theorem 2.5.
It is easy to see that this is exactly the lower-right $m \times m$-submatrix of Y from (4.1). So, for all coefficients of all subresultants, one has to compute the main diagonal and the first $m-1$ superdiagonals of Y, consequently, s must be set to m. In general, if $s \in \{1, \ldots, m\}$, only the main diagonal plus the $s-1$ first superdiagonals of X are computed. These entries correspond to $\mathrm{coef}_k(\mathrm{Sres}_k), \ldots, \mathrm{coef}_{k-s+1}(\mathrm{Sres}_k)$. Loosely speaking, the parameter

s controls how much of the subresultant is computed. Note that $s = 1$ leads to an algorithm for the principal subresultant coefficients. In fact, this specialisation gives precisely the "Bezout/Hybrid algorithm" from [ADG04] (depending on which matrix is chosen for X).
Algorithm 5.1 describes the details for the hybrid Bezout matrix, the symmetric one can be implemented analogously. Note that $\deg f \geq \deg g$ is supposed for shorter notation.

Algorithm 5.1 Subresultants without division

```
procedure SubresViaBezout(f, g, s)
    n ← deg f,   m ← deg g
    H ← J_n · HBez(f, g)                          ▷ J_n backward identity
    for j ∈ {1, ..., s} do
        S_j ← S_{j,s}(H)
        [S_j^k, ..., S_j^0] ←BERKOWITZ(S_j)
                          ▷ S_j^i is the i × i leading principal minor of S_j
    end for
    for k ∈ {n − m + 1, ..., n} do
        j := overline(k + t)      ▷ the representant of k + t in {1, ..., s} mod s
        Sres_{n−k} ← (−1)^{k(k−1)/2} Σ_{t=0}^{min{s−1,n−k}} S_j^k x^{n−k−t}
    end for
    return [Sres_0, ..., Sres_{m−1}]
end procedure
```

As investigated in Section 2, the case $s = 1$ is enough to compute the degree of the gcd, and $s = m$ computes the gcd itself. An application for the case $s = 2$ is the following: Suppose that f and g are polynomials with only one common root β over $\mathbb{C}$. The following lemma states the (non-surprising) fact that β is given by only two coefficients of $\gcd(f, g)$:

Lemma 5.1. Let $\gcd(f, g) = \sum_{i=0}^{k} g_i x^i$. If β is the only common root of f and g, then $\beta = -\frac{g_{k-1}}{k g_k}$.

Proof. Follows from the equality $\sum_{i=0}^{k} g_i x^i = g_k(x - \beta)^k$ by comparing the coefficients of x^{k-1}. □

To compute the gcd in this situation, one can proceed as follows: The psc sequence reveals the degree k of the greatest common divisor (2.1). By the preceeding lemma, only $\mathrm{coef}_k(\mathrm{Sres}_k) = \mathrm{sres}_k$ and $\mathrm{coef}_{k-1}(\mathrm{Sres}_k)$ are necessary to compute the gcd. Hence, it suffices to call Algorithm 5.1 with $s = 2$.

6 Running times

6.1 Complexity

The best known algorithms for subresultants, based on pseudo-division, perform $O(n^2)$ arithmetic operations in the ground domain, n being the maximal degree of the input polynomials ([Du00],[LRS00]). Compared to that, the presented approach looks quite poor at a first sight:

Lemma 6.1. Counting arithmetic operations in the ground domain, the complexity of *SubresViaBezout(f,g,s)* is $O(s \cdot n^4)$.

Proof. The algorithm is clearly dominated by the s calls of the *Berkowitz algorithm* (the construction of the hybrid Bezout matrix and the S-matrices is negligible). As mentioned in Section 3, the complexity is $O(n^4)$, where n is the dimension of the matrix. $\square$

Consequently, for high degrees, there is no hope that *SubresViaBezout* is competitive to division-based approaches.
On the other hand, the costs of a single division might be significantly greater than those of a multiplication or addition in the ground domain. This is particulary true if the domain contains indeterminates, for instance $\mathbb{D} = \mathbb{Z}[a, b, c]$.

6.2 Experiments

Algorithm 5.1 was implemented using Maple 10 on a Linux Workstation with a Pentium IV 2.8 GHz processor and 512 MB RAM. Throughout the tests, it was always applied with $s = m$ to obtain the full subresultant polynomials. For comparision, the *Classical Subresultant algorithm (SubresClass)* ([Lo82],[Du00]) and *Ducos algorithm (SubresDucos)* ([Du00]) were implemented on the same platform. Ducos algorithm has the same asymptotic bound as the classical algorithm: The complexity is $O(n^2)$, counting the arithmetic operations in $\mathbb{D}$. However, it reduces also the size of the intermediate results during the computation, leading to a better performance for certain polynomials. The algorithm is also competitive to the Flipflop algorithm from [LRS00], as the authors point out in that article.
The first tests (Table 1) show the behaviour for polynomials in $\mathbb{Z}[x]$. Thereby, the parameters (n, m, c) mean a randomly generated pair of (dense) polynomials (f, g) with $\deg(f) = n,\ \deg(g) = m$, and all coefficients of bitsize c.

It is clearly visible that the Bezout approach is much worse than the other

Parameters	SubresViaBezout	SubresClass	SubresDucos
(25,15,32)	3.436	0.009	0.035
(35,25,32)	19.608	0.040	0.055
(45,35,32)	74.028	0.062	0.145
(55,45,32)	217.137	0.121	0.280
(7,6,2^8)	0.053	0.003	0.005
(7,6,2^{10})	0.071	0.013	0.021
(7,6,2^{12})	0.260	0.143	0.210
(7,6,2^{14})	1.572	1.744	2.781

Table 1: Running times for polynomials in $\mathbb{Z}[x]$. All times are measured in seconds using Maple's `time()` command

two algorithms for high degrees. The second part shows that our algorithm behaves better than the others for small degrees and very big coefficients. What happens if some indeterminates are added? To study the effect, randomly generated pairs of (dense) polynomials (f, g) were created using parameters (n, m, r, d, c), with $f, g \in \mathbb{Z}[a_1, \dots, a_r][x]$, $\deg_x(f) = n$, $\deg_x(g) = m$, c as above and each coefficient of f and g has total degree at most d. Table 2 shows the parameters used and the running times for the same random polynomial.

Parameters	SubresViaBezout	SubresClass	SubresDucos
$(6, 6, 3, 1, 32)$	0.858	14.294	4.328
$(6, 6, 4, 1, 32)$	5.532	816.292	120.684
$(6, 6, 5, 1, 32)$	34.358	?	3294.105
$(6, 6, 6, 1, 32)$	160.145	?	?
$(6, 5, 2, 3, 32)$	4.391	14.548	7.405
$(6, 5, 2, 5, 32)$	30.426	140.624	69.442
$(6, 5, 2, 7, 32)$	130.344	704.525	334.382
$(6, 5, 2, 9, 32)$	413.366	2452.121	1066.110

Table 2: Running times for polynomials in $\mathbb{Z}[x]$. The entry "?" means that the computation did not finish after one hour.

The first half of the table shows the effect of increasing the number of indeterminates without changing the total degree (all a_i's only appear linearly). The second part deals with trivariate polynomials of increasing total degree. Obviously, the division-free approach behaves much better in both situations. The experiments suggest that the coefficient size, the total degree and the number of indeterminates affect the running time for the division-based methods in a more dramatic way than for the division-free approach.

References

[ADG04] J.Abdeljaoued, G.M.Diaz-Toca, L.Gonzalez-Vega: Minors of Bezout Matrices, Subresultants and the Parameterization of the Degree of the Polynomial Greatest Common Divisor. *Int. Journal of Computer Mathematics* **81** (2004) 1223-1238

[Be84] S.J.Berkowitz: On Computing the Determinant in Small Parallel Time Using a Small Number of Processors. *Information Processing Letters* **18** (1984) 147-150

[BPR03] S.Basu, R.Pollack, M.-F.Roy: Algorithms in Real Algebraic Geometry. Springer 2003

[BT71] J.F.Traub, W.S.Brown: On Euclid's Algorithm and the Theory of Subresultants. *J. Ass. for Comp. Machinery* **18** (1971) 504-514

[CGZ99] E.Chionh, M.Zhang, R.N.Goldman: Transformations and Transitions from the Sylvester to the Bezout Resultant. Technical Report TR99-343, Department of Computer Science, Rice University, 1999.

[CGZ02] E.Chionh, M.Zhang, R.N.Goldman: Fast Computation of the Bezout and Dixon Resultant Matrices. *J. Symbolic Comp.* **33** (2002) 13-29

[DG04] G.M.Diaz-Toca,L.Gonzalez-Vega: Various New Expressions for Subresultants and Their Applications. *Appl. Algebra in Engin., Communic. and Comp.* **15** (2004) 233-266

[Du00] L.Ducos: Optimization of the subresultant algorithm. *J. Pure Appl. Algebra* **145** (2000) 149-163

[GCL92] K.Geddes, S.Czapor, G.Labahn: Algorithms for Computer Algebra. Kluwer Academic Publishers, 1992

[GL03] J.von zur Gathen, T.Lücking: Subresultants revisited. *Theo. Comp. Science* **297** (2003) 199-239

[HW00] X.Hou, D.Wang: Subresultants with the Bezout Matrix. *Proc. of the Fourth Asian Symp. on Computer Mathematics* (ASCM 2000) 19-28

[Lo82] R.Loos: Generalized polynomial remainder sequences. In: B.Buchberger, G.E. Collins, R.Loos (editors): Computer Algebra – Symbolic and Algebraic Computation. Springer (1982) 115-138

[LRS00] H.Lombardi, M.-F.Roy, M.Safey El Din: New Structure Theorem for Subresultants. *Journal of Symbolic Computation* **29** (2000) 663-689

[Ro01] G.Rote: Division-Free Algorithms for the Determinant and the Pfaffian: Algebraic and Combinatorial Approaches. In: H.Alt (Ed.): Computational Discrete Mathematics, 119-135, Springer LNCS 2122, 2001

[Sa42] P.A.Samuelson: A Method of Determining Explicitly the Coefficients of the Characteristic Equation. *Ann. of Math. Statistics* **13** (1942) 424-429

[Yap00] C.K.Yap: Fundamental Problems of Algorithmic Algebra. Oxford University Press, New York 2000

Below you find a list of the most recent technical reports of the Max-Planck-Institut für Informatik. They are available by anonymous ftp from `ftp.mpi-sb.mpg.de` under the directory `pub/papers/reports`. Most of the reports are also accessible via WWW using the URL `http://www.mpi-sb.mpg.de`. If you have any questions concerning ftp or WWW access, please contact `reports@mpi-sb.mpg.de`. Paper copies (which are not necessarily free of charge) can be ordered either by regular mail or by e-mail at the address below.

Max-Planck-Institut für Informatik
Library
attn. Anja Becker
Stuhlsatzenhausweg 85
66123 Saarbrücken
GERMANY
e-mail: `library@mpi-sb.mpg.de`

MPI-I-2006-5-004	F. Suchanek, G. Ifrim, G. Weikum	Combining Linguistic and Statistical Analysis to Extract Relations from Web Documents
MPI-I-2006-5-003	V. Scholz, M. Magnor	Garment Texture Editing in Monocular Video Sequences based on Color-Coded Printing Patterns
MPI-I-2006-5-002	H. Bast, D. Majumdar, R. Schenkel, M. Theobald, G. Weikum	IO-Top-k: Index-access Optimized Top-k Query Processing
MPI-I-2006-5-001	M. Bender, S. Michel, G. Weikum, P. Triantafilou	Overlap-Aware Global df Estimation in Distributed Information Retrieval Systems
MPI-I-2006-4-001	R. Mantiuk	?
MPI-I-2006-2-001	T. Wies, V. Kuncak, K. Zee, A. Podelski, M. Rinard	On Verifying Complex Properties using Symbolic Shape Analysis
MPI-I-2006-1-005	I. Albrecht	?
MPI-I-2006-1-001	M. Dimitrios	?
MPI-I-2006-	M. Kerber	?
MPI-I-2005-5-002	S. Siersdorfer, G. Weikum	Automated Retraining Methods for Document Classification and their Parameter Tuning
MPI-I-2005-4-006	C. Fuchs, M. Goesele, T. Chen, H. Seidel	An Emperical Model for Heterogeneous Translucent Objects
MPI-I-2005-4-005	G. Krawczyk, M. Goesele, H. Seidel	Photometric Calibration of High Dynamic Range Cameras
MPI-I-2005-4-004	C. Theobalt, N. Ahmed, E. De Aguiar, G. Ziegler, H. Lensch, M.A.,. Magnor, H. Seidel	Joint Motion and Reflectance Capture for Creating Relightable 3D Videos
MPI-I-2005-4-003	T. Langer, A.G. Belyaev, H. Seidel	Analysis and Design of Discrete Normals and Curvatures
MPI-I-2005-4-002	O. Schall, A. Belyaev, H. Seidel	Sparse Meshing of Uncertain and Noisy Surface Scattered Data
MPI-I-2005-4-001	M. Fuchs, V. Blanz, H. Lensch, H. Seidel	Reflectance from Images: A Model-Based Approach for Human Faces
MPI-I-2005-2-004	Y. Kazakov	A Framework of Refutational Theorem Proving for Saturation-Based Decision Procedures
MPI-I-2005-2-003	H.d. Nivelle	Using Resolution as a Decision Procedure
MPI-I-2005-2-002	P. Maier, W. Charatonik, L. Georgieva	Bounded Model Checking of Pointer Programs
MPI-I-2005-2-001	J. Hoffmann, C. Gomes, B. Selman	Bottleneck Behavior in CNF Formulas
MPI-I-2005-1-008	C. Gotsman, K. Kaligosi, K. Mehlhorn, D. Michail, E. Pyrga	Cycle Bases of Graphs and Sampled Manifolds
MPI-I-2005-1-008	D. Michail	?
MPI-I-2005-1-007	I. Katriel, M. Kutz	A Faster Algorithm for Computing a Longest Common Increasing Subsequence
MPI-I-2005-1-003	S. Baswana, K. Telikepalli	Improved Algorithms for All-Pairs Approximate Shortest Paths in Weighted Graphs
MPI-I-2005-1-002	I. Katriel, M. Kutz, M. Skutella	Reachability Substitutes for Planar Digraphs

MPI-I-2005-1-001	D. Michail	Rank-Maximal through Maximum Weight Matchings
MPI-I-2004-NWG3-001	M. Magnor	Axisymmetric Reconstruction and 3D Visualization of Bipolar Planetary Nebulae
MPI-I-2004-NWG1-001	B. Blanchet	Automatic Proof of Strong Secrecy for Security Protocols
MPI-I-2004-5-001	S. Siersdorfer, S. Sizov, G. Weikum	Goal-oriented Methods and Meta Methods for Document Classification and their Parameter Tuning
MPI-I-2004-4-006	K. Dmitriev, V. Havran, H. Seidel	Faster Ray Tracing with SIMD Shaft Culling
MPI-I-2004-4-005	I.P. Ivrissimtzis, W.-. Jeong, S. Lee, Y.a. Lee, H.-. Seidel	Neural Meshes: Surface Reconstruction with a Learning Algorithm
MPI-I-2004-4-004	R. Zayer, C. Rssl, H. Seidel	r-Adaptive Parameterization of Surfaces
MPI-I-2004-4-003	Y. Ohtake, A. Belyaev, H. Seidel	3D Scattered Data Interpolation and Approximation with Multilevel Compactly Supported RBFs
MPI-I-2004-4-002	Y. Ohtake, A. Belyaev, H. Seidel	Quadric-Based Mesh Reconstruction from Scattered Data
MPI-I-2004-4-001	J. Haber, C. Schmitt, M. Koster, H. Seidel	Modeling Hair using a Wisp Hair Model
MPI-I-2004-2-007	S. Wagner	Summaries for While Programs with Recursion
MPI-I-2004-2-002	P. Maier	Intuitionistic LTL and a New Characterization of Safety and Liveness
MPI-I-2004-2-001	H. de Nivelle, Y. Kazakov	Resolution Decision Procedures for the Guarded Fragment with Transitive Guards
MPI-I-2004-1-006	L.S. Chandran, N. Sivadasan	On the Hadwiger's Conjecture for Graph Products
MPI-I-2004-1-005	S. Schmitt, L. Fousse	A comparison of polynomial evaluation schemes
MPI-I-2004-1-004	N. Sivadasan, P. Sanders, M. Skutella	Online Scheduling with Bounded Migration
MPI-I-2004-1-003	I. Katriel	On Algorithms for Online Topological Ordering and Sorting
MPI-I-2004-1-002	P. Sanders, S. Pettie	A Simpler Linear Time 2/3 - epsilon Approximation for Maximum Weight Matching
MPI-I-2004-1-001	N. Beldiceanu, I. Katriel, S. Thiel	Filtering algorithms for the Same and UsedBy constraints
MPI-I-2003-NWG2-002	F. Eisenbrand	Fast integer programming in fixed dimension
MPI-I-2003-NWG2-001	L.S. Chandran, C.R. Subramanian	Girth and Treewidth
MPI-I-2003-4-009	N. Zakaria	FaceSketch: An Interface for Sketching and Coloring Cartoon Faces
MPI-I-2003-4-008	C. Roessl, I. Ivrissimtzis, H. Seidel	Tree-based triangle mesh connectivity encoding
MPI-I-2003-4-007	I. Ivrissimtzis, W. Jeong, H. Seidel	Neural Meshes: Statistical Learning Methods in Surface Reconstruction
MPI-I-2003-4-006	C. Roessl, F. Zeilfelder, G. Nrnberger, H. Seidel	Visualization of Volume Data with Quadratic Super Splines
MPI-I-2003-4-005	T. Hangelbroek, G. Nrnberger, C. Roessl, H.S. Seidel, F. Zeilfelder	The Dimension of C^1 Splines of Arbitrary Degree on a Tetrahedral Partition
MPI-I-2003-4-004	P. Bekaert, P. Slusallek, R. Cools, V. Havran, H. Seidel	A custom designed density estimation method for light transport
MPI-I-2003-4-003	R. Zayer, C. Roessl, H. Seidel	Convex Boundary Angle Based Flattening
MPI-I-2003-4-002	C. Theobalt, M. Li, M. Magnor, H. Seidel	A Flexible and Versatile Studio for Synchronized Multi-view Video Recording
MPI-I-2003-4-001	M. Tarini, H.P.A. Lensch, M. Goesele, H. Seidel	3D Acquisition of Mirroring Objects