# SOLVING VIA MODULAR METHODS

DEEBA AFZAL, FAIRA KANWAL, GERHARD PFISTER, AND STEFAN STEIDEL

Abstract. In this article we present a parallel modular algorithm to compute all solutions with multiplicities of a given zero-dimensional polynomial system of equations over the rationals. In fact, we compute a triangular decomposition using Möller's algorithm (cf. [Mö93]) of the corresponding ideal in the polynomial ring over the rationals using modular methods, and then apply a solver for univariate polynomials.

## 1. Introduction

One possible approach[1] to find the solutions of a zero-dimensional system of multivariate polynomials is the triangular decomposition of the corresponding ideal since triangular systems of polynomials can be solved using a univariate solver recursively. There are already several results in this direction including an implementation in *Maple* (cf. [AM99], [LMX05], [Ma00]). The technique to compute triangular sets has been refined (cf. [DMSWX05], [CLMPX07], [LMX06]), modularized and parallelized (cf. [MX07], [LM07]). We report about a modular and parallel version of the solver in Singular (cf. [DGPS12]) and focus mainly on a probabilistic algorithm to compute the solutions of a polynomial system with multiplicities.

## 2. Preliminary technicalities

We recall the definition and some properties of a triangular decomposition. For details we refer to [GP07], [Mö93] and [Mö97].

Let $K$ be a field, $X = \{x_1, \ldots, x_n\}$ a set of variables, $I \subseteq K[X]$ a zero-dimensional ideal, and we fix $>$ to be the lexicographical ordering induced by $x_1 > \ldots > x_n$. If $f \in K[X]$ is a polynomial, then we denote by $\mathrm{LE}(f)$ the leading exponent of $f$, by $\mathrm{LC}(f)$ the leading coefficient of $f$, by $\mathrm{LM}(f)$ the leading monomial of $f$, and by $\mathrm{LT}(f) = \mathrm{LC}(f) \cdot \mathrm{LM}(f)$ the leading term of $f$.

**Definition 2.1.** A set of polynomials $F = \{f_1, \ldots, f_n\} \subseteq K[X]$ is called a *triangular set* if $\mathrm{LT}(f_i) = x_{n-i+1}^{\alpha_i}$ for some $\alpha_i > 0$ and each $i = 1, \ldots, n$.

A list $\mathcal{F} = (F_1, \ldots, F_s)$ of triangular sets is called a *triangular decomposition* of a zero-dimensional ideal $I \subseteq K[X]$ if $\sqrt{I} = \sqrt{\langle F_1 \rangle} \cap \ldots \cap \sqrt{\langle F_s \rangle}$.

*Date*: June 7, 2013.

*Key words and phrases.* triangular sets, solving polynomial systems, modular computation, parallel computation.

[1]In Singular (cf. [DGPS12]) this approach is implemented in the library `solve.lib` on the basis of an univariate *Laguerre solver* (cf. [RR78, §8.9-8.13]).

*Remark* 2.2.

1. A triangular set is a Gröbner basis.
2. A minimal Gröbner basis of a maximal ideal is a triangular set, and the primary decomposition of $\sqrt{I}$ is a triangular decomposition of $I$.

The following two lemmata are the basis for the algorithm by Möller (cf. [Mö93]) which avoids primary decomposition.

**Lemma 2.3.** *Let $G = \{g_1, \ldots, g_m\}$ be a reduced Gröbner basis of the zero-dimensional ideal $I \subseteq K[X]$ such that $\mathrm{LM}(g_1) < \ldots < \mathrm{LM}(g_m)$. Moreover, for $i = 1, \ldots, m$, let $\alpha_i = \mathrm{LE}(g_i)$ in $(K[x_2, \ldots, x_n])[x_1]$, i.e. $g_i = \sum_{j=0}^{\alpha_i} g'_{ij} x_1^j$ for suitable $g'_{ij} \in K[x_2, \ldots, x_n]$. Then $G' = \{g'_{1\alpha_1}, \ldots, g'_{m-1\alpha_{m-1}}\}$ is a Gröbner basis of $\langle g_1, \ldots, g_{m-1}\rangle : g_m$, and we have $\langle G', g_m\rangle = \langle G', G\rangle$.*

*Proof.* The proof can be found in [Mö93, Lemma 7]. □

**Lemma 2.4.** *Let $I \subseteq K[X]$ be a zero-dimensional ideal, and $h \in K[X]$. Then the following hold.*

1. $\sqrt{I} = \sqrt{\langle I, h\rangle} \cap \sqrt{I : h}$.
2. $\dim_K \big(K[X]/I\big) = \dim_K \big(K[X]/\langle I, h\rangle\big) + \dim_K \big(K[X]/(I : h)\big)$.

*Proof.*

1. The proof is an exercise in [GP07, Exercise 4.5.3].
2. We consider two exact sequences. The first one
$$0 \longrightarrow (I : h)/I \longrightarrow K[X]/I \xrightarrow{\cdot h} K[X]/I \longrightarrow K[X]/\langle I, h\rangle \longrightarrow 0$$
yields $\dim_K \big((I : h)/I\big) = \dim_K \big(K[X]/\langle I, h\rangle\big)$, and the second one
$$0 \longrightarrow (I : h)/I \longrightarrow K[X]/I \longrightarrow K[X]/(I : h) \longrightarrow 0$$
yields $\dim_K \big(K[X]/I\big) = \dim_K \big((I : h)/I\big) + \dim_K \big(K[X]/(I : h)\big)$. Summarized, we obtain
$$\dim_K \big(K[X]/I\big) = \dim_K \big(K[X]/\langle I, h\rangle\big) + \dim_K \big(K[X]/(I : h)\big).$$

□

Consequently, for $h_1, \ldots, h_l \in K[X]$ and $h_{l+1} = 1$, we are able to apply Lemma 2.4 inductively and obtain

$$\begin{aligned}\sqrt{I} &= \sqrt{\langle I, h_1\rangle} \cap \sqrt{I : h_1} \\ &= \sqrt{\langle I, h_1, h_2\rangle} \cap \sqrt{\langle I, h_1\rangle : h_2} \cap \sqrt{I : h_1} \\ &= \ldots \\ &= \sqrt{\langle I, h_1, \ldots, h_l\rangle} \cap \left(\bigcap_{i=1}^{l+1} \sqrt{\langle I, h_1, \ldots, h_{i-1}\rangle : h_i}\right).\end{aligned}$$

Together with Lemma 2.3 we conclude the following corollary.

**Corollary 2.5.** *With the assumption of Lemma 2.3, let $G' \smallsetminus G = \{h_1, \ldots, h_l\}$. Then the following hold.*

1. *If $G' \smallsetminus G \neq \emptyset$, then $I \subsetneq \langle G, h_1\rangle$ and $I \subsetneq \langle G\rangle : h_1$.*
2. $\sqrt{I} = \sqrt{\langle G', g_m\rangle} \cap \big(\bigcap_{i=1}^{l+1} \sqrt{\langle G, h_1, \ldots, h_{i-1}\rangle : h_i}\big)$ *and, in addition,* $I \subseteq \langle G', g_m\rangle \cap \big(\langle G, h_1, \ldots, h_{i-1}\rangle : h_i\big)$.

(3) $\dim_K\big(K[X]/I\big) = \sum_{i=1}^{l+1} \dim_K\big(K[X]/(\langle G, h_1, \ldots, h_{i-1}\rangle : h_i)\big)$.

With regard to Corollary 2.5(2), especially $\langle G, h_1, \ldots, h_l\rangle = \langle G', g_m\rangle$ with $G' \subseteq K[x_2, \ldots, x_n]$ is predestined for induction since $\sqrt{\langle G', g_m\rangle} = \sqrt{\langle F_1', g_m\rangle} \cap \ldots \cap \sqrt{\langle F_s', g_m\rangle}$ if $\mathcal{F}' = (F_1', \ldots, F_s')$ is a triangular decomposition of $G'$. Referring to Corollary 2.5(3), the triangular decomposition obtained by iterating the approach of Corollary 2.5 respects the multiplicities of the zeros of $I$. Therefore the zero-sets of different triangular sets are in general not disjoint as the following example shows.

*Example* 2.6. Let $G = \{x_2^{10}, x_1x_2^3 + x_2^5, x_1^{11}\} \subseteq \mathbb{Q}[x_1, x_2]$. Then we obtain $G' = \{x_2^{10}, x_2^3\}$, $G' \smallsetminus G = \{x_2^3\}$, and the triangular decomposition $\mathcal{F} = (F_1, F_2)$ of $\langle G\rangle$ with $F_1 = \langle G\rangle : x_2^3 = \langle x_2^7, x_1 + x_2^2\rangle$ and $F_2 = \langle G, x_2^3\rangle = \langle x_2^3, x_1^{11}\rangle$. Moreover, it holds $\dim_{\mathbb{Q}}(\mathbb{Q}[x_1, x_2]/\langle G\rangle) = 40$, $\dim_{\mathbb{Q}}\big(\mathbb{Q}[x_1, x_2]/(\langle G\rangle : x_2^3)\big) = 7$, and $\dim_{\mathbb{Q}}\big(\mathbb{Q}[x_1, x_2]/\langle G, x_2^3\rangle\big) = 33$. Note that $\langle G\rangle \subsetneq F_1 \cap F_2$.

Algorithm 1 shows the algorithm by Möller to compute the triangular decomposition of a zero-dimensional ideal.[2]

**Algorithm 1** Triangular Decomposition (`triangM`)

```
Input: $I \subseteq K[X]$, a zero-dimensional ideal .
Output: $\mathcal{F} = (F_1, \ldots, F_s)$, a triangular decomposition of $I \subseteq K[X]$ such that
    $\dim_K\big(K[X]/I\big) = \sum_{i=1}^{s} \dim_K\big(K[X]/\langle F_i\rangle\big)$.
1: compute $G = \{g_1, \ldots, g_m\}$, a reduced Gröbner basis of $I$ with respect to the
   lexicographical ordering $>$ such that $\mathrm{LM}(g_1) < \ldots < \mathrm{LM}(g_m)$;
2: compute $G' = \{g_1', \ldots, g_{m-1}'\} \subseteq K[x_2, \ldots, x_n]$ where $g_i'$ is the leading coefficient
   of $g_i$ in $(K[x_2, \ldots, x_n])[x_1]$;
3: $\mathcal{F}' = $ triangM$(\langle G'\rangle)$;
4: $\mathcal{F} = \{F' \cup \{g_m\} \mid F' \in \mathcal{F}'\}$;
5: for $1 \leq i \leq m-1$ do
6:     if $g_i' \notin G$ then
7:         $\mathcal{F} = \mathcal{F} \cup$ triangM$(\langle G\rangle : g_i')$;
8:         $G = G \cup \{g_i'\}$;
9: return $\mathcal{F}$;
```

Note that Algorithm 1 is only based on Gröbner basis computations, and does not use random elements. Hence, the result is uniquely determined which allows modular computations. In the following we fix Algorithm 1 to compute a triangular decomposition.

*Remark* 2.7. Replacing line 7 in Algorihtm 1 by $\mathcal{F} = \mathcal{F} \cup \texttt{triangM}(\langle G\rangle : g_i'^{\infty})$ we obtain a disjoint triangular decomposition (i.e. $F_i, F_j \in \mathcal{F}$ with $F_i \neq F_j$ implies $\langle F_i\rangle + \langle F_j\rangle = K[X]$). This decomposition does in general not respect the multiplicities (i.e. $\dim_K\big(K[X]/I\big) \neq \sum_{i=1}^{s} \dim_K\big(K[X]/\langle F_i\rangle\big)$).

[2]The corresponding procedure is implemented in SINGULAR in the library `triang.lib`.

## 3. Modular methods

One possible modular approach for solving a zero-dimensional ideal is to just replace each involved Gröbner basis computation by its corresponding modular algorithm as described in [IPS11]. Particularly, we can replace line 1 in Algorithm 1 by $G = \texttt{modStd}(I)$. In this case it is possible to apply the probabilistic variant since we can easily verify in the end by a simple substitution whether the solutions obtained from the triangular sets are really solutions of the original ideal. Nevertheless, we propose to compute the whole triangular decomposition via modular methods actually. The verification is then similar by just substituting the obtained result into the input polynomials.

We consider the polynomial ring $\mathbb{Q}[X]$, fix a global monomial ordering $>$ on $\mathbb{Q}[X]$, and use the following notation: If $S \subseteq \mathbb{Q}[X]$ is a set of polynomials, then $\mathrm{LM}(S) := \{\mathrm{LM}(f) \mid f \in S\}$ denotes the set of leading monomials of $S$. If $f \in \mathbb{Q}[X]$ is a polynomial, $I = \langle f_1, \ldots, f_r\rangle \subseteq \mathbb{Q}[X]$ is an ideal, and $p$ is a prime number which does not divide any denominator of the coefficients of $f, f_1, \ldots, f_r$, then we write $f_p := (f \mod p) \in \mathbb{F}_p[X]$ and $I_p := \langle (f_1)_p, \ldots, (f_r)_p\rangle \subseteq \mathbb{F}_p[X]$.

In the following, $I = \langle f_1, \ldots, f_r\rangle \subseteq \mathbb{Q}[X]$ will be a zero-dimensional ideal. The triangular decomposition algorithm (Algorithm 1) applied to $I$ returns a list of triangular sets $\mathcal{F} = (F_1, \ldots, F_s)$ such that $\sqrt{I} = \bigcap_{i=1}^{s} \sqrt{\langle F_i\rangle}$ and $\dim_{\mathbb{Q}}(\mathbb{Q}[X]/I) = \sum_{i=1}^{s} \dim_{\mathbb{Q}}(\mathbb{Q}[X]/\langle F_i\rangle)$.

With respect to modularization, the following lemma obviously holds:

**Lemma 3.1.** *With notation as above, let p be a sufficiently general prime number. Then $I_p$ is zero-dimensional, $((F_1)_p, \ldots, (F_s)_p)$ is a list of triangular sets, and $\sqrt{I_p} = \bigcap_{i=1}^{s} \sqrt{\langle (F_i)_p\rangle}$.*

Relying on Lemma 3.1, the basic idea of the modular triangular decomposition is as follows. First, choose a set $\mathcal{P}$ of prime numbers at random. Second, compute triangular decompositions $\mathcal{F}_p$ of $I_p$ for $p \in \mathcal{P}$. Third, lift the modular triangular sets to triangular sets $\mathcal{F}$ over $\mathbb{Q}[X]$.

The lifting process consists of two steps. First, the set $\mathcal{FP} := \{\mathcal{F}_p \mid p \in \mathcal{P}\}$ is lifted to $\mathcal{F}_N$ with $(F_i)_N \subseteq (\mathbb{Z}/N\mathbb{Z})[X]$ and $N := \prod_{p\in\mathcal{P}} p$ by applying the Chinese remainder algorithm to the coefficients of the polynomials occurring in $\mathcal{FP}$. Second, we obtain $\mathcal{F}$ with $F_i \subseteq \mathbb{Q}[X]$ by lifting the modular coefficients occurring in $\mathcal{F}_p$ to rational coefficients via the Farey rational map[3]. This map is guaranteed to be bijective provided that $\sqrt{N/2}$ is larger than the moduli of all coefficients of elements in $\mathcal{F}$ with $F_i \subseteq \mathbb{Q}[X]$.

We now define a property of the set of primes $\mathcal{P}$ which guarantees that the lifting process is feasible and correct. This property is essential for the algorithm.

**Definition 3.2.** Let $\mathcal{F} = (F_1, \ldots, F_s)$ be the triangular decomposition of the ideal $I$ computed by Algorithm 1.

(1) A prime number $p$ is called *lucky* for $I$ and $\mathcal{F}$ if $((F_1)_p, \ldots, (F_s)_p)$ is a triangular decomposition of $I_p$. Otherwise $p$ is called *unlucky* for $I$ and $\mathcal{F}$.

[3]*Farey fractions* refer to rational reconstruction. A definition of *Farey fractions*, the *Farey rational map*, and remarks on the required bound on the coefficients can be found in [KG83].

(2) A set $\mathcal{P}$ of lucky primes for $I$ and $\mathcal{F}$ is called *sufficiently large* for $I$ and $\mathcal{F}$ if

$$\prod_{p\in\mathcal{P}} p \geq \max\{2\cdot|c|^2 \mid c \text{ coefficient occuring in } \mathcal{F}\}.$$

From a theoretical point of view, the idea of the algorithm is now as follows: Consider a sufficiently large set $\mathcal{P}$ of lucky primes for $I$ and $\mathcal{F}$, compute the triangular decomposition of the $I_p$, $p \in \mathcal{P}$, via Algorithm 1, and lift the results to the triangular decomposition of $I$ as aforementioned.

From a practical point of view, we face the problem that we do not know in advance whether a prime number $p$ is lucky for $I$ and $\mathcal{F}$.

To handle this problem, we fix a natural number $t$ and an arbitrary set of primes $\mathcal{P}$ of cardinality $t$. Having computed $\mathcal{FP}$, we use the following test to modify $\mathcal{P}$ such that all primes in $\mathcal{P}$ are lucky with high probability:

DELETEUNLUCKYPRIMESTRIANG: *We define an equivalence relation on* $(\mathcal{FP},\mathcal{P})$ *by* $(\mathcal{F}_p,p) \sim (\mathcal{F}_q,q) :\Longleftrightarrow \big(\#\mathcal{F}_p = \#\mathcal{F}_q$ *and* $\{\mathrm{LM}(F_p) \mid F_p \in \mathcal{F}_p\} = \{\mathrm{LM}(F_q) \mid F_q \in \mathcal{F}_q\}\big)$. *Then the equivalence class of largest cardinality is stored in* $(\mathcal{FP},\mathcal{P})$*, the others are deleted.*

Since we do not know a priori whether the equivalence class chosen is indeed lucky and whether it is sufficiently large for $I$ and $\mathcal{F}$, we proceed in the following way. We lift the set $\mathcal{FP}$ to $\mathcal{F}$ over $\mathbb{Q}[X]$ as described earlier, and test the result with another randomly chosen prime number:

PTESTTRIANG: *We randomly choose a prime number* $p \notin \mathcal{P}$ *such that* $p$ *does not divide the numerator and denominator of any coefficient occuring in a polynomial in* $\{f_1,\dots,f_r\}$ *or* $\mathcal{F}$*. The test returns true if* $(F_p \mid F \in \mathcal{F})$ *equals the triangular decomposition* $\mathcal{F}_p$ *computed by the fixed Algorithm 1 applied on* $I_p$*, and false otherwise.*

If PTESTTRIANG returns false, then $\mathcal{P}$ is not sufficiently large for $I$ and $\mathcal{F}$ or the equivalence class of prime numbers chosen was unlucky. In this case, we enlarge the set $\mathcal{P}$ by $t$ new primes and repeat the whole process. On the other hand, if PTESTTRIANG returns true, then we have a triangular decomposition $\mathcal{F}$ of $I$ with high probability. In this case, we compute the solutions $S \subseteq \mathbb{C}^n$ with multiplicities of the triangular sets $F = \{f_1,\dots,f_n\} \in \mathcal{F}$ as follows. Solve the univariate polynomial $f_1(x_1)$ via a univariate solver counting multiplicities, substitute $x_1$ in $f_2(x_1,x_2)$ by these solutions of $f_1(x_1)$, solve the corresponding univariate polynomial, and continue inductively this way (call this step SOLVETRIANG). Finally, we verify the result $S \subseteq \mathbb{C}^n$ partially by testing whether the original ideal $I \subseteq \mathbb{Q}[X]$ is contained in every $F \in \mathcal{F}$, and whether the sum of the multiplicities equals the $\mathbb{Q}$-dimension of $\mathbb{Q}[X]/I$ (call this step TESTZERO).

We summarize modular solving in Algorithm 2.[4]

*Remark* 3.3. In Algorithm 2, the triangular sets $\mathcal{F}_p$ can be computed in parallel. Furthermore, we can parallelize the final verification whether $I \subseteq \mathbb{Q}[X]$ is contained in every $F \in \mathcal{F}$.

[4]The corresponding procedures are implemented in SINGULAR in the library `modsolve.lib`.

**Algorithm 2** Modular Solving (`modSolve`)

**Input:** $I \subseteq \mathbb{Q}[X]$, a zero-dimensional ideal.
**Output:** $S \subseteq \mathbb{C}^n$, a set of points in $\mathbb{C}^n$ such that $f(P) = 0$ for all $f \in I, P \in S$.

```
1: choose $\mathcal{P}$, a list of random primes;
2: $\mathcal{FP} = \emptyset$;
3: loop
4:     for $p \in \mathcal{P}$ do
5:         $\mathcal{F}_p = \texttt{triangM}(I_p)$, the triangular decomposition of $I_p$ via Algorithm 1;
6:         $\mathcal{FP} = \mathcal{FP} \cup \{\mathcal{F}_p\}$;
7:     $(\mathcal{FP}, \mathcal{P}) = \text{deleteUnluckyPrimesTriang}(\mathcal{FP}, \mathcal{P})$;
8:     lift $(\mathcal{FP}, \mathcal{P})$ to $\mathcal{F}$ over $\mathbb{Q}[X]$ by applying Chinese remainder algorithm and
       Farey rational map;
9:     if $\text{pTestTriang}(I, \mathcal{F}, \mathcal{P})$ then
10:        $S = \text{solveTriang}(\mathcal{F})$;
11:        if $\text{testZero}(I, \mathcal{F}, S)$ then
12:            return $S$;
13:    enlarge $\mathcal{P}$;
```

## 4. Examples and timings

In this section we provide examples on which we time the algorithm `modSolve` (cf. Algorithm 2) and its parallel version as opposed to the algorithm `solve` (the procedure `solve` is implemented in Singular in the library `solve.lib` and computes all roots of a zero-dimensional input ideal using triangular sets). Timings are conducted by using Singular 3-1-6 on an AMD Opteron 6174 machine with 48 CPUs, 2.2 GHz, and 128 GB of RAM running the Gentoo Linux operating system. All examples are chosen from The SymbolicData Project (cf. [G13]).

*Remark* 4.1. The parallelization of the modular algorithm is attained via multiple processes organized by Singular library code. Consequently, a future aim is to enable parallelization in the kernel via multiple threads.

We choose the following examples to emphasize the superiority of modular solving and especially its parallelization:

*Example* 4.2. `Cyclic_7.xml` (cf. [G13]).

*Example* 4.3. `Verschelde_noon6.xml` (cf. [G13]).

*Example* 4.4. `Pfister_1.xml` (cf. [G13]).

*Example* 4.5. `Pfister_2.xml` (cf. [G13]).

Table 1 summarizes the results where `modSolve`$(c)$ denotes the parallelized version of the algorithm applied on $c$ cores. All timings are given in seconds.

*Remark* 4.6. Various experiments reveal that a sensitive choice of $\#\mathcal{P}$, the number of random primes in lines 1 and 13 in Algorithm 2, can decrease the running time enormously. To sum up, it is recommendable to relate $c$, the number of available cores, to $\#\mathcal{P}$. Particularly, in case of having more than ten cores to ones's disposal it is reasonable to set $c = \#\mathcal{P}$.

| Example | `solve` | `modSolve` | `modSolve`(10) | `modSolve`(20) |
|---|---|---|---|---|
| 4.2 | $> 18$h | 692 | 217 | 152 |
| 4.3 | 517 | 1223 | 522 | 371 |
| 4.4 | 526 | 800 | 288 | 165 |
| 4.5 | 2250 | 1276 | 323 | 160 |

Table 1. Total running times in seconds for computing all roots of the considered examples via `solve`, `modSolve` and its parallelized variant `modSolve`($c$) for $c = 10, 20$.

## References

[AM99] Aubry, P.; Moreno Maza, M.: *Triangular Sets for Solving Polynomial Systems: a Comparative Implementation of Four Methods.* Journal of Symbolic Computation 28, 125–154 (1999).

[CLMPX07] Chen, C.; Lemaire, F.; Moreno Maza, M.; Pan, W.; Xie, Y.: *Efficient Computations of Irredundant Triangular Decompositions with the RegularChains Library.* Proceedings of Computer Algebra Systems and Their Applications '07. In: Shi, Y. et al. (Eds.): ICCS 2007, Part II, Lecture Notes in Computer Science 4488, 268–271 (2007).

[DMSWX05] Dahan, X.; Moreno Maza, M.; Schost, É.; Wu, W.; Xie, Y.: *Lifting Techniques for Triangular Decompositions.* Proceedings of ISSAC '05, Beijing, China, ACM Press, 108–115 (2005).

[DZ05] Dayton, B. H.; Zeng, Z.: *Computing the multiplicity structure in solving polynomial systems.* Proceedings of ISSAC '05, Beijing, China, ACM Press, 116–123 (2005).

[DGPS12] Decker, W.; Greuel, G.-M.; Pfister, G.; Schönemann, H.: Singular *3-1-6 — A computer algebra system for polynomial computations.* http://www.singular.uni-kl.de (2012).

[G13] Gräbe, H.-G.: *The SymbolicData Project — Tools and Data for Testing Computer Algebra Software.* http://www.symbolicdata.org (2013).

[GP07] Greuel, G.-M.; Pfister, G.: *A* Singular *Introduction to Commutative Algebra.* Second edition, Springer (2007).

[IPS11] Idrees, N.; Pfister, G.; Steidel, S.: *Parallelization of Modular Algorithms.* Journal of Symbolic Computation 46, 672–684 (2011).

[KG83] Kornerup, P.; Gregory, R. T.: *Mapping Integers and Hensel Codes onto Farey Fractions.* BIT Numerical Mathematics 23(1), 9–20 (1983).

[L92] Lazard, D.: *Solving Zero-dimensional Algebraic Systems.* Journal of Symbolic Computation 13, 117–131 (1992).

[LMX05] Lemaire, F.; Moreno Maza, M.; Xie, Y.: *The RegularChains library in Maple 10.* Maplesoft, Canada (2005).

[LMX06] Lemaire, F.; Moreno Maza, M.; Xie, Y.: *Making a Sophisticated Symbolic Solver Available to Different Communities of Users.* Proceedings of Asian Technology Conference in Mathematics '06, Polytechnic University of Hong Kong (2006).

[LM07] Li, X.; Moreno Maza, M.: *Multithreaded Parallel Implementation of Arithmetic Operations modulo a Triangular Set.* Proceedings of Parallel Symbolic Computation '07, London, Canada, ACM Press, 53–59 (2007).

[Mö93] Möller, H. M.: *On Decomposing Systems of Polynomial Equations With Finitely Many Solutions.* Applicable Algebra in Engineering, Communication and Computing 4, 217–230 (1993).

[Mö97] Möller, H. M.: *Solving of Algebraic Equations – An Interplay of Symbolical and Numerical Methods.* In: Multivariate Approximation, Recent Trends and Results (eds.: Haußmann, W.; Jetter, K.; Reimer, M.), Mathematical Research 101, Akademie Verlag, 161–176 (1997).

[Ma00] Moreno Maza, M.: *On Triangular Decompositions of Algebraic Varieties.* Presented at the MEGA 2000 conference, Bath, United Kingdom (2000).

[MX07] Moreno Maza, M.; Xie, Y.: *Component-level Parallelization of Triangular Decompositions.* Proceedings of Parallel Symbolic Computation '07, London, Canada, ACM Press, 69–77 (2007).

[RR78] Ralston, A.; Rabinowitz, P.: *A First Course in Numerical Analysis.* Second edition, McGraw-Hill (1978).

Deeba Afzal, Abdus Salam School of Mathematical Sciences, GC University, Lahore, 68-B, New Muslim Town, Lahore 54600, Pakistan
*E-mail address*: deebafzal@gmail.com

Faira Kanwal, Abdus Salam School of Mathematical Sciences, GC University, Lahore, 68-B, New Muslim Town, Lahore 54600, Pakistan
*E-mail address*: fairakanwaljanjua@gmail.com

Gerhard Pfister, Department of Mathematics, University of Kaiserslautern, Erwin-Schrödinger-Str., 67663 Kaiserslautern, Germany
*E-mail address*: pfister@mathematik.uni-kl.de

Stefan Steidel, Department of Mathematical Methods in Dynamics and Durability, Fraunhofer Institute for Industrial Mathematics ITWM, Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany
*E-mail address*: stefan.steidel@itwm.fraunhofer.de