Term Cancellations in Computing Floating-point Gröbner Bases

Tateaki Sasaki[1] and Fujio Kako[2]

[1] Professor emeritus, University of Tsukuba,
Tsukuba-city, Ibaraki 305-8571, Japan
sasaki@math.tsukuba.ac.jp
[2] Department of Info. and Comp. Sci., Nara Women's University,
Nara-city, Nara 630-8506, Japan
kako@ics.nara-wu.ac.jp

Abstract. We discuss the term cancellation which makes the floating-point Gröbner basis computation unstable, and show that error accumulation is never negligible in our previous method. Then, we present a new method, which removes accumulated errors as far as possible by reducing matrices constructed from coefficient vectors by the Gaussian elimination. The method manifests amounts of term cancellations caused by the existence of approximate linearly dependent relations among input polynomials.

1 Introduction

Although floating-point Gröbner bases are indispensable in scientific computation, they have been scarcely used so far. The reason is that the computation is so unstable that it is very difficult to obtain desired results.

There are two kinds of floating-point Gröbner bases. The first kind is that the coefficients of the input polynomials are exact (say algebraic numbers or transcendental numbers) but we utilize the floating-point numbers for some reasons. The second kind is that the coefficients of input polynomials are inexact hence we inevitably express the coefficients by floating-point numbers. If the numerical errors increase during the computation, we can replay the computation with higher precision for the first kind, however, for the second kind, we must devise a method to preserve the initial accuracies of input polynomials as far as possible. In this paper, we deal with the second kind. We should emphasize that, since the errors in the input polynomials are unknown, the algorithm should return the bases which do not critically depend on the input errors.

The first kind of floating-point Gröbner bases were studied by Shirayanagi and Sweedler [16–18]. The second kind of floating-point Gröbner bases were studied by Stetter [19], Kalkbrener [6], Fortuna, Gianni and Trager [3], Traverso and Zanoni [24], Traverso [23], Weispfenning [25], Kondratyev, Stetter and Winkler [8], Gonzalez-Vega, Traverso and Zanoni [4], Stetter [21], Bodrato and Zanoni [1], and so on. Recently, Suzuki [22] and Nagasaka [11] proposed to compute Gröbner bases by reducing large numerical matrices by Gaussian elimination. In

spite of these studies, computation of floating-point Gröbner bases of the second kind has been a serious problem; the computation was so unstable in most cases if performed naively. This seriousness forced European researchers to study the so-called "border bases" [9, 10, 5], see also [2].

The key point of stabilizing the floating-point Gröbner basis computation is how to control the errors caused by term cancellation; see Sect. 2 for details. We classify the term cancellation into two classes, systematic and accidental. These four years, the present authors have studied this theme and presented the symbolic coefficient method [14] and the high-precision method [15]. The high-precision method is quite stable and useful so long as the accidental cancellations do not occur. However, the method has a weakness: although the coefficients are computed pretty accurately, the estimation of intrinsic errors occurred on the coefficients is very bad. Here, by "intrinsic errors" we mean errors caused by ill-conditionedness of the input basis, which cannot be reduced by the computational techniques. For example, suppose there exists an approximate linearly dependent relation among input polynomials, then, when the relation is computed, the errors of its coefficients will inevitably be increased. Knowing the amounts of intrinsic errors is very important for the Gröbner bases with inexact coefficients. Therefore, we want to develop a method which informs us the amounts of intrinsic errors. In this paper, we present such a method. The idea is to improve the estimation of errors of polynomials which are computed by our previous method, by reducing coefficient matrices by Gaussian elimination.

Since many readers will be unfamiliar with the phenomenon of term cancellation, we will explain our method as elementally as possible using examples.

2 Term cancellation and its monitoring

By F, G, etc., we denote polynomials in $\mathbb{F}[x, y, \ldots, z]$, where $\mathbb{F}$ denotes the floating-point numbers of a fixed precision. By $\|F\|$ we denote the *norm* of F; we employ the infinity norm in this paper. The *power product* is the product of powers of variables. By $\mathrm{supp}(F)$ we denote the *support* of F, i.e., the set of all the power products appearing in F. By $\mathrm{lt}(F)$ and $\mathrm{rt}(F)$, we denote the *leading term* and the *rest terms*, of F, respectively, with respect to a given order $\succ$; $F = \mathrm{lt}(F) + \mathrm{rt}(F)$. By $\mathrm{Spol}(F, G)$ and $\mathrm{Lred}(F, G)$, we denote the *S-polynomial* of F and G and the *leading-term reduction* of F by G, respectively. We express $\mathrm{Lred}(F, G)$ also as $F \xrightarrow{G}$. By $F \xrightarrow{G}\!\!\!\!\rightarrow \tilde{F}$, we denote successive leading-term reductions of F by G so that $\mathrm{lt}(\tilde{F})$ is no longer reducible by G.

If the error of a floating-point number f starts at the $(l{+}1)$-st bit then we say that the accuracy of f is $1/2^l$, and we call the leading l bits the significant ones. Let c_1T and c_2T be monomials, where $c_1, c_2 \in \mathbb{F}$. If leading l bits of c_1 and c_2 are the same but the $(l{+}1)$-st ones are not then the l bits are lost in the subtraction $c_1T - c_2T$. We call this *term cancellation* of amount 2^l, and the relative error of $c_1 - c_2$ increases by 2^l compared with c_1 and c_2. If the resulting $c_1 - c_2$ has no significant bit then we call the cancellation *exact*, otherwise *inexact*. If the term cancellation is exact, we call the resulting term $(c_1 - c_2)T$ a *fully-erroneous term*.

We may classify the term cancellation as follows.

$$\begin{cases} \textit{systematic} \begin{cases} \textit{exact cancellation} \\ \textit{inexact cancellation} \end{cases} \\ \textit{accidental} \begin{cases} \textit{exact cancellation} \\ \textit{inexact cancellation} \end{cases} \end{cases}$$

We explain the difference between systematic and accidental cancellations. Let a polynomial Q be contained in both P_1 and P_2 such as $P_i = P_i' + c_i Q$ $(i = 1, 2)$. If $c_1 \simeq c_2$ then $c_1 Q$ and $c_2 Q$ cancel in the subtraction $P_1 - P_2$. This is a typical systematic cancellation. The cancellation is exact if c_1 and c_2 cancel exactly, otherwise the cancellation is inexact. We also call the case $\|P_1 - P_2\| \ll \|P_1\|$ systematic cancellation. This case occurs, for example, if there exists an approximate linear dependence among polynomials in the initial or intermediate bases. Let P_1 and P_2 contain $c_1 T$ and $c_2 T$, respectively, where these terms originate from different initial terms. If $c_1 \simeq c_2$ then the term cancellation occurs in the subtraction $P_1 - P_2$. This is the accidental cancellation. The actual term cancellation is complicated because other term $c'T$ may be mixed before the subtraction of $c_1 T$ and $c_2 T$. We call $(c_1 + c')T - c_2 T$ and $c_1 T - c_2 T$ the term cancellation *with* and *without mixing*, respectively.

The systematic exact cancellation occurs frequently in polynomial linear algebra, such as the computation of determinants with polynomial entries and the polynomial remainder sequences. Similarly, it occurs frequently in the computation of Gröbner bases, as shown in [14, 15, 13]. If the systematic exact cancellation is caused by polynomials with small leading terms, we usually encounter large cancellation errors. On the other hand, the accidental cancellation occurs rarely, especially if the input polynomials are generated randomly or determined by experiments. In the rest of this section, we survey our previous work briefly.

If a fully-erroneous term appears as the leading term then the subsequent computation becomes meaningless. Therefore, we must remove the fully-erroneous terms completely. So far, two ideas have been proposed to do so. Shirayanagi [16–18] proposed to represent the input coefficients by intervals and remove the fully-erroneous terms by replacing the interval by 0 if it contains 0. The present authors devised the so-called *effective floating-point numbers*, or *efloats* in short [7], so as to detect the cancellation errors automatically but approximately. We represent the efloat number as $\#\mathrm{E}(f, e)$, where f is a floating-point number representing the value of this number, and e is a short floating-point number representing the error of f. We call f and e as *value-part* and *error-part*, respectively. For the arithmetic of efloats, see [7].

Let $\varepsilon_{\mathrm{cal}}$ be the precision of floating-point numbers employed; we have $\varepsilon_{\mathrm{cal}} \simeq 2 \times 10^{-16}$ in the double-precision arithmetic. In our algebra system, the error-part of efloat is set slightly larger than $\varepsilon_{\mathrm{cal}}|f|$: we set e to $10^{-15}|f|$ in the double-precision arithmetic, and to $100\,\varepsilon_{\mathrm{cal}}|f|$ in the high-precision arithmetic. If the input coefficient f contains relative error $\varepsilon_{\mathrm{init}}$, then we may set the error-part to $5\,\varepsilon_{\mathrm{init}}|f|$, say. Our algebra system sets the efloat $\#\mathrm{E}(f, e)$ to 0 if $|f| < e$.

The exact term cancellation with mixing is not so simple. Suppose we have $|c_1| \gg c'$ in $((c_1+c')-c_2)T$, then the cancellation of amount $|c_1/c'|$ occurs in the resulting $c_1+c'-c_2$. Fortunately, if the exact cancellation is systematic, we can preserve the initial accuracy in $c_1+c'-c_2$. The idea is to convert all the input coefficients to high precision floating-point numbers [15]. Then, since c_1 and c_2 originate from a single coefficient c of an input polynomial, their errors are the same initially and subsequent computation contaminates only some lower bits of them. Suppose ℓ lower bits of c_1 and c_2 are contaminated at most. Then, the significant part of c' is safe so long as the precision has been increased initially by more than 2^ℓ. This is the essence of the high-precision method.

3 Accidental cancellations and current problems

In this section, we consider accidental cancellations. For simplicity, we assume that the input coefficients are accurate to $\varepsilon_{\text{init}}$ at most; if the coefficients are of different accuracies, we set $\varepsilon_{\text{init}}$ to the maximum of the accuracies. It is natural to assume that $\varepsilon_{\text{init}} \gg \varepsilon_{\text{cal}}$.

If accidental exact cancellation without mixing occurs then all the bits above $\varepsilon_{\text{init}}$ are lost. The resulting term is also fully-erroneous, and we must remove such terms, too. This removal can be done easily if we represent coefficients by efloats. Suppose the accidental exact cancellation occurs in c_1T-c_2T, hence c_1-c_2 is fully-erroneous. Since c_1 and c_2 originate from different coefficients, c_1-c_2 will be a number of relative error $\sim \varepsilon_{\text{init}}$, and we assumed that $\varepsilon_{\text{init}} \gg \varepsilon_{\text{cal}}$. We can detect this fully-erroneous term by monitoring the corresponding efloat. Thus, we can remove such a fully-erroneous term by replacing an efloat $\#\mathrm{E}[f, e]$ by 0 if we have $|f| < n_{\text{g}}\varepsilon_{\text{init}}\, e/(100\,\varepsilon_{\text{cal}})$, where n_{g} is a guard number, say 10.

Example 1 (Fully-erroneous term by accidental exact cancellation). We consider the following system with the lexicographic order.

$$\begin{cases} F_1 = x^2\,(2yz+1)/2.0, \\ F_2 = (x\,(3xz-2) - (2yz+1))/3.0, \\ F_3 = (yz\,(3xz-2))/2.0. \end{cases}$$

We first convert the coefficients into double-precision floats. Note that F_3 is input by dividing by 2.0, not by 3.0. This artificial trick introduces different errors into yz terms of F_2 and F_3. Computing the floating-point Gröbner basis by the high-precision method with 30 decimal precisions, we obtain $\{\,1\,\}$. Investigating the computation process, we found the following intermediate polynomial.

$$\begin{aligned} & \#\mathrm{E}(8.43749999999999945356\cdots\mathrm{e}{-1},\ 4.2\mathrm{e}{-28})\,x^2z \\ - \ & \#\mathrm{E}(3.70074341541718836536\cdots\mathrm{e}{-17},\ 6.7\mathrm{e}{-28})\,xy^2 \\ + \ & \#\mathrm{E}(1.12499999999999994795\cdots\mathrm{e}{-0},\ 3.0\mathrm{e}{-28})\,xyz \\ + \ & \quad\cdots\quad\cdots \end{aligned}$$

The above second term is fully erroneous, caused by the accidental cancellation. This term is very small but the result depends on it critically. □

We show a weakness of our previous method by an example.

Example 2 (Large errors in the result). We computed an unreduced Gröbner basis w.r.t. the total-degree order, of the above system, obtaining

$$\begin{cases} G_1 = \#\mathrm{E}(9.99999999999999999\cdots\mathrm{e}{-1},\ 1.9\mathrm{e}{-25})\,x \\ \quad + \#\mathrm{E}(1.33333333333333348136\cdots\mathrm{e}{-0},\ 2.5\mathrm{e}{-25})\,y, \\ G_2 = \#\mathrm{E}(9.99999999999999999\cdots\mathrm{e}{-1},\ 1.0\mathrm{e}{-28})\,yz \\ \quad + \#\mathrm{E}(5.81395348837208978910\cdots\mathrm{e}{-2},\ 3.7\mathrm{e}{-27})\,x \\ \quad + \#\mathrm{E}(7.75193798449612428019\cdots\mathrm{e}{-2},\ 4.9\mathrm{e}{-27})\,y \\ \quad + \#\mathrm{E}(5.00000000000000019525\cdots\mathrm{e}{-1},\ 6.7\mathrm{e}{-27}). \end{cases}$$

G_2 can be reduced by G_1 and we obtain the same Gröbner basis as that in Example 1. We, however, find that the errors of the above result are large compared with those in Example 1; error-parts of G_1 are about 10^3 larger than its initial values (the value-parts are accurate to $O(10^{-16})$ relatively.) The origin of these large errors can be understood by considering syzygies (a_{i1}, a_{i2}, a_{i3}) for G_i $(i=1,2)$: $G_i = a_{i1}F_1+a_{i2}F_2+a_{i3}F_3$. Normalizing the leading coefficients of G_1 to 1, we find that a_{11}, a_{12}, a_{13} contain 49, 54 and 40 terms, of norms 5109/5, 560 and 4439/10, respectively, and we have $\max\{\|a_{11}F_1\|, \|a_{12}F_2\|, \|a_{13}F_3\|\} = 10218/5$. The largeness of $\|a_{ij}\|$ $(1 \le i \le 2;\ 1 \le j \le 3)$ implies that, during Buchberger's procedure, corresponding polynomials are multiplied by large monomials but they will cancel later because the final polynomials are of norm $O(1)$. Therefore, big errors were induced in the final basis. □

Summarizing the above discussions, we have the following problems to solve.

1) Remove the errors caused by accidental exact cancellation with mixing.

2) Remove the errors caused by accidental inexact cancellation.

Furthermore, in our high-precision method, small cancellation errors accumulate gradually but steadily to error-parts of efloats, as Example 2 shows. Therefore, we must solve the following problem, too.

3) Reduce the gradually accumulating errors as far as possible.

4 Our idea: reduce the errors by a matrix method

In this paper, *we restrict the reduction of polynomials only to the leading-term reduction*: we compute the Gröbner bases by constructing S-polynomials and performing only the leading-term reductions successively. If we need the reduced Gröbner basis then we perform the reduction of non-leading terms after computing an unreduced Gröbner basis.

The problems given in Sect. 3 are quite difficult to solve algebraically. For example, the problem 3) is inevitable so long as Buchberger's procedure is employed. We note that problems 1) and 2) are common in numerical computation, and numerical analysts developed excellent techniques to suppress the increase of errors. In the case of matrix computation, the QR-decomposition is one of such techniques. On the other hand, our high-precision method can compute

floating-point Gröbner bases stably. Therefore, we will reduce numerically the errors of the results computed by the high-precision method.

Let $\Phi_k = \{F_1^{(k)}, \ldots, F_r^{(k)}\}$ and $\Phi_l = \{F_1^{(l)}, \ldots, F_s^{(l)}\}$ be intermediate bases appearing in the computation of Gröbner basis by Buchberger's procedure, where Φ_k is a basis computed before Φ_l (Φ_k may be the initial basis). Then, each polynomial $F_i^{(l)}$ in Φ_l can be expressed in terms of the elements of Φ_k, as follows:

$$F_i^{(l)} = a_{i1}F_1^{(k)} + \cdots + a_{ir}F_r^{(k)}. \tag{1}$$

We call the tuple $(a_{i1}, \ldots, a_{ir})$ a syzygy for $F_i^{(l)}$. In order to reduce the errors occurred during the computation from Φ_k to Φ_l, we utilize the syzygies.

Let $\mathcal{R}_i$ $(1 \leq i \leq s)$ and $\mathcal{C}_{ij}$ $(1 \leq j \leq r)$ be sets of power-products defined as follows.

$$\begin{aligned} &\mathcal{R}_i = \cup_{j=1}^{r} \mathrm{supp}(a_{ij}F_j^{(k)}) \stackrel{\mathrm{def}}{=} \{T_1, T_2, \ldots, T_{\bar{n}}\}, \quad T_1 \succ T_2 \succ \cdots \succ T_{\bar{n}}, \\ &\mathcal{C}_{ij} = \mathrm{supp}(a_{ij}) \stackrel{\mathrm{def}}{=} \{S_{ij,1}, S_{ij,2}, \ldots, S_{ij,m_j}\}, \quad S_{ij,1} \succ S_{ij,2} \succ \cdots \succ S_{ij,m_j}. \end{aligned} \tag{2}$$

We express $F_i^{(l)}$ and $F_j^{(k)}$ as follows.

$$\begin{aligned} F_i^{(l)} &= f_{i1}^{(l)}T_{i1} + f_{i2}^{(l)}T_{i2} + \cdots, \quad T_{i1} \succ T_{i2} \succ \cdots, \\ F_j^{(k)} &= f_{j1}^{(k)}T_{j1} + f_{j2}^{(k)}T_{j2} + \cdots. \; T_{j1} \succ T_{j2} \succ \cdots. \end{aligned} \tag{3}$$

Since $\{S_{ij,m}T_{j1}, \ldots, S_{ij,m}T_{jn_j}\} \subset \{T_1, T_2, \ldots, T_{\bar{n}}\}$ for any $j \in \{1, \ldots, r\}$ and $m \in \{1, \ldots, m_j\}$, (1) assures that we can express the coefficient vector of $F_i^{(l)}$ in terms of a sum of coefficient vectors of $S_{ij,m}F_j^{(k)}$ $(1 \leq j \leq r;\ 1 \leq m \leq m_j)$, expanded in power-products $T_1, T_2, \ldots, T_{\bar{n}}$.

We put $\bar{m} = m_1 + \cdots + m_j$. We define an $\bar{m} \times \bar{n}$ matrix $\mathcal{M}_i$, which we call *reduction matrix*, as follows.

$$\mathcal{M}_i \;=\; \begin{pmatrix} \text{coefficient vector of } S_{i1,1}F_1^{(k)} \\ \quad \text{coefficient vector of } S_{i1,2}F_1^{(k)} \\ \qquad \ddots \quad \ddots \quad \ddots \quad \ddots \\ \text{coefficient vector of } S_{i2,1}F_2^{(k)} \\ \quad \text{coefficient vector of } S_{i2,2}F_2^{(k)} \\ \qquad \ddots \quad \ddots \quad \ddots \quad \ddots \\ \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \end{pmatrix} \tag{4}$$

Let $\mathcal{M}_i^{(\mathrm{L})}$ be the submatrix of $\mathcal{M}_i$, composed of the columns located in the left side of the power product T_{i1}. We eliminate the columns of $\mathcal{M}_i^{(\mathrm{L})}$, by transforming the rows of $\mathcal{M}_i$ simultaneously. This elimination gives us a row of $\mathcal{M}_i$, of the form $(0, \ldots, 0, \star, *, \ldots, *)$, where $\star$ is not zero and it stands at T_{i1}, while $*$ may be zero. Then, $(\star, *, \ldots, *)$ is a coefficient vector of $F_i^{(l)}$.

Example 3 (Reduction matrix $\mathcal{M}_2$ for G_2 in example 1). The G_2 was computed as $F_1 \xrightarrow{F_2} F_1',\ F_2 \xrightarrow{F_1'} F_2',\ F_3 \xrightarrow{F_2'} F_3',\ F_2' \xrightarrow{F_3'} G_2$. The corresponding syzygy is $(a_{2,1} = -3yz^2 + 3/2z,\ a_{2,2} = 2y^2z^2 - 1/2,\ a_{2,3} = 4/3y)$. By this, we can easily obtain the following minimal power-product set for G_2:

$$\mathcal{R}_2 = (x^2y^2z^3,\ x^2yz^2,\ x^2z,\ xy^2z^2,\ x,\ y^3z^3,\ y^2z^2,\ y^2z,\ yz,\ 1)$$

Then, the reduction matrix $\mathcal{M}_2$ for G_2, expressed by F_1, F_2, F_3, is as follows.

$$\left(\begin{array}{r|cccc|ccccccc}
 & x^2y^2z^3 & x^2yz^2 & x^2z & xy^2z^2 & x & y^3z^3 & y^2z^2 & y^2z & yz & 1 \\ \hline
yz^2F_1 & 1.0 & 0.5 & & & & & & & & \\
zF_1 & & 1.0 & 0.5 & & & & & & & \\
y^2z^2F_2 & 1.0 & & & -2/3. & & -2/3. & -1/3. & & & \\
1F_2 & & & 1.0 & & -2/3. & & & & -2/3. & -1/3. \\
yF_3 & & & & 3/2. & & & & -1.0 & &
\end{array}\right)$$

Here, $\mathcal{M}_2^{(\mathrm{L})}$ is the matrix composed of the left four columns. □

Remark 1. It is obvious that the rows in $\mathcal{M}_i^{(\mathrm{L})}$ are linearly dependent. However, the dimension of null space may be greater than 1. Even if the dimension is 1, the corresponding vector in $\mathcal{M}_i$ may be different from the coefficient vector of $F_i^{(l)}$, because the matrix reduction is rarely the same as the corresponding reductions in Buchberger's method. □

Our idea solves problem 3). The systematic exact cancellation is analogous to that occurs in naive Gaussian elimination of numerical matrices: if a row of small pivot is used to eliminate columns then there occur large cancellations in subsequent eliminations. We can avoid such cancellations almost completely by pivoting. We show this by Example 4, below. Furthermore, the matrix method mentioned above decreases the risk of accidental cancellations largely.

Example 4 (Reducing the errors in G_1, G_2 in Example 2). Eliminating reduction matrices for G_1, G_2, we obtained the following improved G_1, G_2.

$$\begin{cases}
G_1 = x + \#\mathrm{E}(1.33333333333333340734820164167 7\mathrm{e}{-0},\ 2.7\mathrm{e}{-28})\, y \\
G_2 = yz + \#\mathrm{E}(4.99999999999999972244424384371 0\mathrm{e}{-1},\ 1.0\mathrm{e}{-28})\, x \\
\qquad + \#\mathrm{E}(6.66666666666666666666666666666 6\mathrm{e}{-1},\ 1.3\mathrm{e}{-28})\, y \\
\qquad + \#\mathrm{E}(5.00000000000000000000000000000 0\mathrm{e}{-1},\ 1,5\mathrm{e}{-29})\,.
\end{cases}$$

Almost no cancellation occurred in the computation of the above G_1, G_2. □

Remark 2. The above G_2 is different from that in Example 2. The matrix $\mathcal{M}_2$ for G_2 is of size 112×88, and 84 columns are eliminated. When the elimination finished, we have 21 nonzero rows (the left 84 elements are zero). These 21 rows are linear combinations of the following two vectors

$$\begin{array}{llllllll}
(0 & \cdots & 0 & 3.30\cdots & 0.00 & 0.00 & 1.65\cdots), \\
(0 & \cdots & 0 & 0.00 & 3.60\cdots & 4.80\cdots & 0.00 &).
\end{array}$$

These vectors correspond to $yx + 1/2$ and $x + 4/3y$, respectively. Hence, the ideal itself is the same. □

Example 5 (Case of large systematic exact cancellation). We compute an unreduced Gröbner basis w.r.t. the total-degree order, of the following system.

$$\begin{cases} F_1 = x^3/30.0 + x^2y + y^2/3.0, \\ F_2 = x^2y^2/3.0 - xy^2 - xy/3.0, \\ F_3 = y^3/20.0 + x^2. \end{cases}$$

The high-precision method gives us the following unreduced Gröbner basis.

$$\begin{cases} G_2 = \#\mathrm{E}(9.99999999999999999\cdots\mathrm{e}{-1},\ 1.7\mathrm{e}{-18})\, y^2, \\ G_4 = \#\mathrm{E}(9.99999999999999999\cdots\mathrm{e}{-1},\ 2.2\mathrm{e}{-19})\, xy \\ \quad + \#\mathrm{E}(8.44022550452195867628\cdots\mathrm{e}{-2},\ 1.8\mathrm{e}{-20})\, y^2, \\ G_5 = \#\mathrm{E}(9.99999999999999999\cdots\mathrm{e}{-1},\ 1.0\mathrm{e}{-28})\, x^2 \\ \quad + \#\mathrm{E}(7.14849689746270700639\cdots\mathrm{e}{-0},\ 1.7\mathrm{e}{-18})\, xy \\ \quad + \#\mathrm{E}(5.73716139524645722576\cdots\mathrm{e}{-1},\ 1.3\mathrm{e}{-19})\, y^2. \end{cases}$$

We see that there occurred large cancellations of amounts $O(10^{10})$. Applying the matrix method, we obtained the following result (we omit G_2).

$$\begin{cases} G_4 = xy + \#\mathrm{E}(8.44022550452195867628\cdots\mathrm{e}{-2},\ 3.7\mathrm{e}{-29})\, y^2, \\ G_5 = x^2 + \#\mathrm{E}(7.14849689746270700639\cdots\mathrm{e}{-0},\ 3.6\mathrm{e}{-26})\, xy \\ \qquad + \#\mathrm{E}(5.73716139524645722576\cdots\mathrm{e}{-1},\ 2.7\mathrm{e}{-27})\, y^2. \end{cases}$$

We see that the error-parts of the efloats are improved drastically. (The value-parts are the same up to the double-precision, showing that the high-precision method is trusty). □

We next consider intrinsic errors. From the viewpoint of matrix method, the intrinsic errors are regarded as errors occurring in reducing ill-conditioned matrices, hence they cannot be reduced by the matrix method; we need some pre-conditioning operation to reduce the errors.

Example 6 (Case of intrinsic errors). We compute an unreduced Gröbner basis w.r.t. the total-degree order, of the following system. Note that we have the relation $\|56/57\, yzF_1 - 57/56\, xzF_3 - 2F_2\| = 0.000041$.

$$\begin{cases} F_1 = 57/56\, x^2y + 68/67\, xz^2 - 79/78\, xy + 89/88\, x, \\ F_2 = \qquad\qquad xyz^3 - \quad xy^2z + \quad xyz, \\ F_3 = 56/57\, xy^2 - 67/68\, yz^2 + 78/79\, y^2 - 88/89\, y. \end{cases}$$

Using the high-precision method of 30 decimal precision, we found that the following polynomials were generated in the computation.

$$\begin{aligned} G_6 = {} & \#\mathrm{E}(9.99999999999999999\cdots\mathrm{e}{-1},\ 1.23\mathrm{e}{-23})\, x^2y^2 \\ & - \#\mathrm{E}(2.99543694773255264453\cdots\mathrm{e}{-0},\ 3.68\mathrm{e}{-23})\, xy^2 \\ & - \#\mathrm{E}(1.00207821651237482576\cdots\mathrm{e}{-0},\ 1.23\mathrm{e}{-23})\, y^3 \\ & + \#\mathrm{E}(1.99832546917372451401\cdots\mathrm{e}{-0},\ 2.45\mathrm{e}{-23})\, xy \\ & + \#\mathrm{E}(1.00352171725641447536\cdots\mathrm{e}{-0},\ 1.23\mathrm{e}{-23})\, y^2, \\ F' = {} & \#\mathrm{E}(9.99999999999999999\cdots\mathrm{e}{-1},\ 6.33\mathrm{e}{-13})\, xy^2 \\ & - \#\mathrm{E}(568.429046071616395538\cdots\mathrm{e}{-0},\ 3.60\mathrm{e}{-10})\, xz^2 \\ & + \#\mathrm{E}(565.429585271231326003\cdots\mathrm{e}{-0},\ 3.58\mathrm{e}{-10})\, xy \\ & - \#\mathrm{E}(566.434224887207346419\cdots\mathrm{e}{-0},\ 3.59\mathrm{e}{-10})\, x. \end{aligned}$$

G_6 is changed only a little by the matrix method: the error-parts of the coefficients of G_6 are changed only in the third digits.

The F' does not appear in the final basis, but if we discard F' just when it is generated, the resulting basis becomes very different from the basis over $\mathbb{Q}$. The F' suggests that, once the intrinsic errors of considerable amounts occur, the computation will become unstable. □

We mentioned in **1** that we want to know the amounts of intrinsic errors. If the errors due to the systematic exact cancellations are eliminated completely, then the resulting errors are intrinsic. We can expect that the matrix method will avoid the systematic exact cancellations almost completely. Then, we may say that the matrix method informs us the amounts of intrinsic errors.

5 Actual implementation and discussions

In this section, by "quality improvement of a polynomial" we mean reducing the error-parts of the polynomial coefficients by the matrix method.

In examples in **4**, qualities of final polynomials were improved directly from the initial ones. In this approach, we must often handle matrices of very large sizes. For example, in Example 4, we handled matrices of sizes 143×109 and 112×88, and the cost of matrix reduction is about twice of that of the Gröbner basis computation itself. Furthermore, in this approach, we know the amounts of intrinsic errors only at the final stage, where initial accuracies of some polynomials may be lost at all. Therefore, we improve the qualities of polynomials not only in the final but also in intermediate bases.

We divide the whole computation into many stages, the initial stage, the 1-st stage, the 2-nd stage, and so on. In the beginning of the k-th stage, we have a set of starting polynomials $\{F_1^{(k)}, \ldots, F_{r_k}^{(k)}\}$. At the end of the k-th stage, we improve the qualities of all the existing polynomials, and the improved polynomials are used as the starting polynomials of the $(k+1)$-st stage:

$$\Phi_0 = \{F_1^{(0)}, \ldots, F_{r_0}^{(0)}\} \longrightarrow \cdots \longrightarrow \Phi_k = \{F_1^{(k)}, \ldots, F_{r_k}^{(k)}\} \longrightarrow \cdots . \tag{5}$$

Each stage is closed and the next stage begins when a systematic cancellation of amount C_{small} or more is detected or when the cancellations accumulate to C_{med} or more ($C_{\text{small}} = 100$ and $C_{\text{med}} = 1000$ in the current program). The quality improvement is performed by the following two procedures.

`improvePols`(Φ): this procedure is called at the end of each stage, and it improves the quality of every polynomial in the basis Φ, intermediate or final. Each polynomial is then replaced by the improved one.

`improvePol`(P): this procedure improves a single polynomial P and checks whether P is actually reduced to 0. This procedure is called when the cancellation of C_{big} or more is detected in $\text{Spol}(P_1, P_2)$ or $P' \xrightarrow{Q}\!\!\!\!\rightarrow P$ ($C_{\text{big}} = 10^6$ in the current program).

One important notice in this approach is that the systematic exact cancellation may not be removed inside a single stage. A typical mechanism of systematic exact cancellation is as follows. Suppose polynomials P_1 and P_2 are reduced by Q the leading term of which is small: $P_i \xrightarrow{Q} \tilde{P}_i$ $(i=1,2)$ (P_1 and P_2 may be reduced by $Q_1, \ldots, Q_k$). Then, $P_i \approx -(\mathrm{lt}(P_i)/\mathrm{lt}(Q))\,\mathrm{rt}(\cdots \mathrm{rt}(Q)\cdots)$, and the multiples of $\mathrm{rt}(\cdots \mathrm{rt}(Q)\cdots)$ cancel exactly in subsequent $\mathrm{Spol}(\tilde{P}_1, \tilde{P}_2)$ or $\mathrm{Lred}(\tilde{P}_1, \tilde{P}_2)$. In [14, 15], we called $\tilde{P}_1$ and $\tilde{P}_2$ *clones* of Q, and the subsequent cancellation *self-reduction*. Suppose the clones are generated in the k-th stage and the self-reduction of the clones occurs in the l-th stage. If $k = l$ then the quality improvement at the end of the l-th stage will remove the systematic exact cancellation. If, however, $k < l$ then we must back to the k-th stage so as to remove the cancellation occurred in the l-th stage. On the other hand, for the systematic inexact cancellation, we need not back to previous stages.

Another important notice is on how to express polynomials in the l-th stage by the starting polynomials in the k-th stage when $l > k{+}1$. One may think that we can do this by connecting syzygies in the k-th to l-th stages. This is true if no quality improvement is made in the stages. However, in many cases, the quality improvement changes the polynomial structure; some terms may be missing and some terms may be added by the improvement. Therefore, we compute the polynomials in the l-th stage as follows: first, compute the starting polynomials of the $(k{+}1)$-st stage by using the syzygies, then compute the polynomial in the l-th stage by applying Buchberger's procedure to the starting polynomials in the $(k{+}1)$-st stage.

So far, we have tested only a preliminary version which does not back stages. We show some timing data on Lenovo IdeaPad U450p (1.2GHz, 1MB), where we used efloats of 30 decimal precision. Each Gröbner basis computed is the same as that over $\mathbb{Q}$ up to double precision.

(unit: milliseconds)
(datum 1180 for Katsura-4 is by the computation over $\mathbb{Q}$)

sample	high-prec. method	with quality-improve
Example-4	3.6	10.4
Example-5	1.8	7.6
Example-6	2.4	28.0
Katsura-4	[1180 (+320GC)]	824 (+112GC)

Katsura-4 contains 4 polynomials of total-degree 2 and 1 polynomial of total-degree 1, in 5 variables. The Gröbner basis w.r.t. the total-degree order is composed of 16 polynomials of total-degrees 1 to 5.

We found the following features of our method.

1. Gaussian elimination with partial pivoting which selects a row is often fail to reduce the errors well, but the full pivoting which selects the maximum magnitude element works well. We also tested matrix reduction by the Givens rotation, and found that this reduction method is much more expensive than the Gaussian elimination.

2. Procedure `improvePol`(P) is quite effective for detecting whether P should be regarded as 0.
3. If a considerable amounts of intrinsic errors occur on a polynomial P, then the errors propagate to other polynomials via $\mathrm{Spol}(P, P')$ and $F \xrightarrow{P}\!\!\!\!\rightarrow \tilde{F}$, and the computation becomes unstable soon. (In Katsura-4, we encountered no intrinsic cancellation.)

We must comment on the above point 3. One may think that the point 3. implies a severe limitation of our method. Our opinion is completely different. We note that we have tested polynomials of limited accuracies: we have converted the coefficients of given polynomials into double-precision floating-point numbers, which introduces $O(10^{-16})$ relative errors, then converted them into efloats of 30 decimal precision. When handling polynomials of limited accuracies, the bases computed have no meaning if the initial accuracies are lost during the computation. Therefore, in order to obtain some meaningful results, we must discard polynomials the accuracies of which were lost largely. That is, we must define "approximate Gröbner bases" so that the polynomials of very low accuracies are discarded by some criteria derived theoretically.

In [14], we have tried to define approximate Gröbner bases by using syzygies. However, the definition in [14] is too immature. We explain this by considering G_2 in Example 2. The syzygy computed by Buchberger's procedure contains a term of magnitude $O(10^4)$, which does not mean that the intrinsic cancellations of magnitude $O(10^4)$ occur in the computation of G_2. In fact, Example 4 shows that no intrinsic cancellation occurred on G_2. Therefore, we must define approximate Gröbner bases without using syzygies. We are now trying to define approximate Gröbner bases more appropriately.

References

1. M. Bodrato and A. Zanoni, Intervals, syzygies, numerical Gröbner bases: a mixed study. *Proceedings of CASC2006 (Computer Algebra in Scientific Computing)*: Springer-Verlag LNCS 4194, 64-76, 2006.
2. Y. Chen and X. Meng, Border bases of positive dimensional polynomial ideals. *Proceedings of SNC2007 (Symbolic Numeric Computation)*, 65-71, London, Canada, 2007.
3. E. Fortuna, P. Gianni and B. Trager, Degree reduction under specialization. *J. Pure Appl. Algebra*, **164**, 153-164, 2001.
4. L. Gonzalez-Vega, C. Traverso and A. Zanoni. Hilbert stratification and parametric Gröbner bases. *Proceedings of CASC2005 (Computer Algebra in Scientific Computing)*: Springer-Verlag LNCS 3718, 220-235, 2005.
5. M. Kreuzer and A. Kehrein, Computing border bases. *J. Pure Appl. Alg.*, **200**, 279-295, 2006.
6. M. Kalkbrener, On the stability of Gröbner bases under specialization. *J. Symb. Comput.*, **24**, 51-58, 1997.
7. F. Kako and T. Sasaki, Proposal of "effective" floating-point number. Preprint of Univ. Tsukuba, May 1997 (unpublished).

8. A. Kondratyev, H.J. Stetter and S. Winkler, Numerical computation of Gröbner bases. *Proceedings of CASC2004 (Computer Algebra in Scientific Computing)*, 295-306, St. Petersburg, Russia, 2004.
9. B. Mourrain, A new criterion for normal form algorithms. *Lec. Notes Comp. Sci.*, **179**, 430-443, Springer, 1999.
10. B. Mourrain, Pythagore's dilemma, symbolic-numeric computation, and the border basis method. *Symbolic-Numeric Computations (Trends in Mathematics)*, 223-243, Birkhäuser Verlag, 2007.
11. K. Nagasaka, A Study on Gröbner basis with inexact input. *Lec. Notes Comp. Sci.: Proceedings of CASC2009 (Computer Algebra in Scientific Computing)*, **5743**, 248-258, Kobe, Japan, 2009.
12. T. Sasaki, A practical method for floating-point Gröbner basis computation. *Proceedings of ASCM2009 (Asian Symposium on Computer Mathematics)*; Math-for-industry series (Kyushu Univ.), Vol. 22, 167-176, Fukuoka, Japan, 2009.
13. T. Sasaki, A subresultant-like theory for Buchberger's procedure. Preprint of Univ. Tsukuba, March, 2010 (17 pages).
14. T. Sasaki and F. Kako, Computing floating-point Gröbner base stably. *Proceedings of SNC2007 (Symbolic Numeric Computation)*, 180-189, London, Canada, 2007.
15. T. Sasaki and F. Kako, Floating-point Gröbner basis computation with ill-conditionedness estimation. *Proceedings of ASCM2007 (Asian Symposium on Computer Mathematics)*, Singapore, Dec. 2007; see also LNAI 5081, 278-292, Deepak Kapur (Ed.), Springer, 2008.
16. K. Shirayanagi, An algorithm to compute floating-point Gröbner bases. *Mathematical Computation with Maple V. Ideas and Applications*, Birkhäuser, 95-106, 1993.
17. K. Shirayanagi, Floating point Gröbner bases. *Mathematics and Computers in Simulation*, **42**, 509-528, 1996.
18. K. Shirayanagi and M. Sweedler, Remarks on automatic algorithm stabilization. *J. Symb. Comput.*, **26**, 761-765, 1998.
19. H.J. Stetter, Stabilization of polynomial systems solving with Gröbner bases. *Proceedings of ISSAC'97 (Intern'l Symposium on Symbolic and Algebraic Computation)*, 117-124, ACM Press, 1997.
20. H.J. Stetter. *Numerical Polynomial Algebra*. SIAM Publ., Philadelphia, 2004.
21. H.J. Stetter, Approximate Gröbner bases – an impossible concept?. *Proceedings of SNC2005 (Symbolic-Numeric Computation)*, 235-236, Xi'an, China, 2005.
22. A. Suzuki, Computing Gröbner bases within linear algebra. *Lec. Notes Comp. Sci.: Proceedings of CASC2009 (Computer Algebra in Scientific Computing)*, **5743**, 310-321, Kobe, Japan, 2009.
23. C. Traverso, Syzygies, and the stabilization of numerical Buchberger algorithm. *Proceedings of LMCS2002 (Logic, Mathematics and Computer Science)*, 244-255, RISC-Linz, Austria, 2002.
24. C. Traverso and A. Zanoni, Numerical stability and stabilization of Gröbner basis computation. *Proceedings of ISSAC2002 (Intern'l Symposium on Symbolic and Algebraic Computation)*, 262-269, ACM Press, 2002.
25. V. Weispfenning, Gröbner bases for inexact input data. *Proceedings of CASC2003 (Computer Algebra in Scientific Computing)*, 403-411, Passau, Germany, 2003.