Extracting a Gröbner Basis from Inexact Input *

Kosaku Nagasaka

Kobe University †

E-mail: nagasaka@main.h.kobe-u.ac.jp

Abstract

Computing a Gröbner basis with inexact input is one of the challenging problems in symbolic-numeric computations, and has been studied by many researchers from the following points of view: 1) numerical stability and 2) mathematical correctness. However, it is not yet studied from the data mining point of view: how to extract a meaningful result from the given inexact input when the amount of noise is not small or we do not have enough information about the input. In this paper, we introduce a stabilization method for extracting a numerical Gröbner basis from the given inexact input, by basic numerical linear algebra, low rank approximations and known numerical Gröbner basis algorithms.

Key words: Approximate Gröbner Bases, Symbolic-Numeric Computations

1 Introduction

Gröbner basis is one of the most powerful tools in mathematical computations: solving algebraic equations, simplifying algebraic relations, automatic theorem proving and so on. Recently, computing a Gröbner basis for polynomials with inexact coefficients has been studied by several researchers ([17, 18, 10, 19, 20, 23, 25, 27]) since in practice we have to operate with empirical data. In Sasaki and Kako [17], this problem is classified into the first and second kinds of problem. The first kind is computing a Gröbner basis for the ideal generated by the given polynomials with exact coefficients by numerical arithmetic (e.g. floating-point arithmetic). The second kind is for the given polynomials with inexact coefficients having a priori errors. In this case, we have to operate with a priori errors whether we compute a basis by exact arithmetic or not. In this paper, we focus on the second kind of problem.

The research aspects for the second kind are mainly numerical stability and mathematical meaning, of the results. For the former aspect, several numerical analyses are given by Sasaki and Kako [18, 17] and an elaboration of the idea of a stabilized Gröbner basis computation for zero-dimensional cases is studied by Kondratyev and others [10]. For the latter aspect,

*This work was supported in part by MEXT KAKENHI (22700011).

†3-11 Tsurukabuto, Nada-ku, Kobe 657-8501 JAPAN.

Weispfenning introduced a solution by comprehensive Gröbner bases [27, 28, 26]. However, in general, a comprehensive Gröbner system has a huge number of segments and its computation time is quite slow (see [14] for example). Though Weispfenning [27] tried to decrease the time-complexity by using only a single parameter to represent the inexact parts, whose bounding error mechanism is very similar to interval arithmetic and Traverso and Zanoni [25] pointed out that an interval easily becomes too large when we compute a Gröbner basis by interval arithmetic. Moreover, Sasaki pointed out that it is very difficult to represent errors as parameters or intervals in practical situations, in personal conversations.

In this paper, from the data mining point of view which is different from these known results, we introduce a new approach for the second kind of problem by combining the following known methods.

- **Reduced row echelon form** ([11, 4, 6, 10, 15, 24]) by which we can compute a Gröbner basis by the well-known linear algebraic ways.
- **Structured low rank approximations** including **Structured total least squares** ([13, 12, 9, 3] and citations therein) by which we can compute a rank deficient structured matrix satisfying some useful properties.

After the introductory subsection below, we review some relevant results of computing a Gröbner basis by reduced row echelon form in Section **2**. Our definition for the second kind of numerical Gröbner basis and our new method to compute it are given in Sections **3** and **4**, respectively. In Section **5**, we give some useful remarks.

1.1 The Problem to be Solved

We show some examples of numerical difficulties to be solved in this paper. For example, suppose that we compute a Gröbner basis w.r.t. the graded lexicographic order for the ideal generated by the following polynomials.

$$F_{fp} = \{\ 1.01x^2-2.09y^2+0.002,\ 4.03x^2y+3.06xy,\ 2.04x^2y+0.504x^2+1.504xy-1.02y^2\ \}. \quad (1.1)$$

In this case, by the algorithm (appGröbner, [17]), we get the following Gröbner bases $G_{az,15}$ and $G_{az,2}$ that are computed with initial precision $\varepsilon_{Init} = 10^{-16}$ and $\varepsilon_{Init} = 10^{-3}$ and approximate-zero threshold $\varepsilon_Z = 10^{-15}$ and $\varepsilon_Z = 10^{-2}$, respectively. Please note that 1) we use our implementation[1] with Mathematica's error tracking system instead of effective floating-point numbers used in the original paper, 2) we show only limited leading figures (rounded) in the all examples in this paper, and 3) the resulting Gröbner basis would be similar to the following (though it is depending on the specified tolerance) if we use other algorithms with some approximate zero test (e.g. the present author [15]).

$$\begin{aligned} G_{az,15} &= \{1.0\}, \\ G_{az,2} &= \{1.0x^2 - 2.07y^2 + 0.00198,\ 1.0y^3 + 0.36xy\}. \end{aligned}$$

However, using any approximate zero test may make a difference between the ideal generated by the resulting Gröbner basis and the given ideal, and we do not have any method to confirm

[1]We use the Buchberger algorithm with the Gebauer and Möller criteria and the sugar strategy.

whether the resulting basis is suitable or not. This may cause a big problem. In fact, we have to choose a suitable result between $G_{az,15}$ and $G_{az,2}$. Moreover, there are so many possible combinations of initial precision and approximate-zero threshold though we use only two of them in this example.

On the other hand, we can use the stabilization techniques for algebraic algorithms ([19, 20, 21]) if we assume that the given polynomial set is exact up to their input precisions. Though we may be able to get some theoretical results, they just converge to $G_{st} = \{1\}$ which is just a result by ordinary exact arithmetic. Please note again that the stabilization techniques are introduced for exact inputs and not designed for inexact inputs.

Moreover, we can compute a comprehensive Gröbner system for F_{fp} after rationalizing it and adding parameters on all the coefficients as follows.

$$\begin{aligned} F_\varepsilon = \{\ & (\tfrac{101}{100} + \varepsilon_1)x^2 + (-\tfrac{209}{100} + \varepsilon_2)y^2 + (\tfrac{1}{500} + \varepsilon_3),\ (\tfrac{403}{100} + \varepsilon_4)x^2y + (\tfrac{153}{50} + \varepsilon_5)xy, \\ & (\tfrac{51}{25} + \varepsilon_6)x^2y + (\tfrac{63}{125} + \varepsilon_7)x^2 + (\tfrac{188}{125} + \varepsilon_8)xy + (-\tfrac{51}{50} + \varepsilon_9)y^2\ \}. \end{aligned}$$

In this case, it is very difficult to get reasonable result and we have so many segments in resulting comprehensive Gröbner system (the number of segments is 263 even for very similar but easier problem). In addition to this problem, even if we can deal with the large number of segments, we are not able to determine which segment is preferable for this input since in general we do not have any information on a priori errors (that may be continuous as well) .

Therefore, roughly speaking, any known method can not extract a meaningful result from the given inexact input in both mathematical and practical aspects at once. To solve this problem, from the data mining point of view, we go back to the starting point: "When do we compute a Gröbner basis with inexact inputs?". The present author thinks that we compute it when it seems that there are some algebraic structures on inexact data. That is, we should find a Gröbner basis with lower entropy (unfortunately enlarged by some errors), which may be hidden by a priori errors. In this paper, we introduce an algorithm from this point of view. For example, by our algorithm, we have the following numerical Gröbner basis for F_{fp}.

$$G = \{1.00000x^2 - 2.06419y^2 + 1.32909 \times 10^{-14},\ 1.00000y^3 + 0.362958xy - 6.43879 \times 10^{-15}y\}.$$

A notable difference between G and $G_{az,2}$ is the constant term of the first basis polynomial, and our algorithm suggests that 0.00198 in $G_{az,2}$ is generated by a priori errors and should be zero or tiny.

2 Gröbner basis by RREF

Some researchers studied computing a Gröbner basis by using the reduced row echelon form (RREF for short, [11, 4]). However, this is not efficient since we have to operate with large matrices. Using matrix operations partially like F_4 and F_5 ([6], [7], [10]) may be the best choice if we want to decrease the computation time. However, it may be useful since we can use so many results from numerical linear algebra for the situation where we must inevitably operate with a priori errors.

We assume that we compute a Gröbner basis or its variants for the ideal $I \subseteq \mathbf{C}[\vec{x}]$ generated by a polynomial set $F = \{f_1, \ldots, f_k\} \subset \mathbf{C}[\vec{x}]$ where $\mathbf{C}[\vec{x}]$ is the polynomial ring in variables

$\vec{x} = x_1, \ldots, x_\ell$ over the complex number field $\mathbf{C}$. We note that we use the following definition though there are several equivalents (see [1] or other text books).

Definition 1 (Gröbner Basis) *$G = \{g_1, \ldots, g_{r_G}\} \subseteq I \backslash \{0\}$ is a Gröbner basis for I w.r.t. a fixed term order $\succ$ if for any $f \in I \setminus \{0\}$, there exists $g_i \in G$ such that $\mathrm{ht}(g_i)|\mathrm{ht}(f)$ where $\mathrm{ht}(p)$ denotes the head term of $p(\vec{x}) \in \mathbf{C}[\vec{x}]$ w.r.t. $\succ$.* $\triangleleft$

We consider the linear map $\phi_{\mathcal{T}} : \mathbf{C}[\vec{x}]_{\mathcal{T}} \to \mathbf{C}^{1 \times m_{\mathcal{T}}}$ such that $\phi_{\mathcal{T}}(t_i) = \overrightarrow{e_i}$ where $\mathbf{C}[\vec{x}]_{\mathcal{T}}$ is the submodule of $\mathbf{C}[\vec{x}]$ generated by an ordered set (the left-most element is the highest) of terms $\mathcal{T} = \{t_1, \ldots, t_{m_{\mathcal{T}}}\}_{\succ}$ and $\overrightarrow{e_i} (i = 1, \ldots, m_{\mathcal{T}})$ denotes the canonical basis of $\mathbf{C}^{1 \times m_{\mathcal{T}}}$. The coefficient vector $\overrightarrow{p}$ of $p(\vec{x}) \in \mathbf{C}[\vec{x}]$ is defined to be satisfying $\overrightarrow{p} = \phi_{\mathcal{T}}(p)$ and $p(\vec{x}) = \phi_{\mathcal{T}}^{-1}(\overrightarrow{p})$. With a fixed $\mathcal{T}$, we consider the following subset $F_{\mathcal{T}}$ of I.

$$F_{\mathcal{T}} = \left\{ \sum_{i=1}^{k} s_i(\vec{x}) f_i(\vec{x}) \;\middle|\; s_i(\vec{x}) f_i(\vec{x}) \in \mathbf{C}[\vec{x}]_{\mathcal{T}},\ s_i(\vec{x}) \in \mathbf{C}[\vec{x}] \right\}.$$

The Buchberger algorithm guarantees that $G \subseteq F_{\mathcal{T}}$ if $\mathcal{T}$ has a large enough number of elements (however, note that $\mathcal{T}$ must include some required elements depending on the term order). To compute a Gröbner basis for I, we construct the matrix $\mathcal{M}_{\mathcal{T}}(F)$ whose each row vector $\overrightarrow{p} = \phi_{\mathcal{T}}(p)$ is corresponding to each element of $p(\vec{x}) \in \mathcal{P}_{\mathcal{T}}(f)$ for each $f(\vec{x}) \in F$ where

$$\mathcal{P}_{\mathcal{T}}(f) = \{t_i \times f(\vec{x}) \in \mathbf{C}[\vec{x}]_{\mathcal{T}} \mid t_i = \phi_{\mathcal{T}}^{-1}(\overrightarrow{e_i}),\ i = 1, \ldots, m_{\mathcal{T}}\}.$$

By this definition, $F_{\mathcal{T}}$ and the linear space $\mathcal{V}_{\mathcal{T}}$ generated by the row vectors of $\mathcal{M}_{\mathcal{T}}(F)$ are isomorphic. We note that a matrix is said to be in the reduced row echelon form if it satisfies the following four conditions.

1. All nonzero rows appear above zero rows.
2. Each leading element of a row is in a column to the right of the leading element of the row above it.
3. The leading element in any nonzero row is 1.
4. Every leading element is the only nonzero element in its column.

For the sake of completeness we present the following two lemmas with proofs introduced in [15] by the present author.

Lemma 1 *Let $\overline{\mathcal{M}_{\mathcal{T}}(F)}$ be the reduced row echelon form of $\mathcal{M}_{\mathcal{T}}(F)$. If $g_i(\vec{x}) \in F_{\mathcal{T}}$ for a fixed $i \in \{1, \ldots, r_G\}$, $\overline{\mathcal{M}_{\mathcal{T}}(F)}$ has a row vector $\overrightarrow{p}$ satisfying $\mathrm{ht}(g_i) = \mathrm{ht}(\phi_{\mathcal{T}}^{-1}(\overrightarrow{p}))$.* $\triangleleft$

Proof Since the linear map $\phi_{\mathcal{T}}$ is defined by the ordered set $\mathcal{T}$, each leading element of a row vector $\overrightarrow{p}$ of $\overline{\mathcal{M}_{\mathcal{T}}(F)}$ is corresponding to $\mathrm{ht}(\phi_{\mathcal{T}}^{-1}(\overrightarrow{p}))$. The lemma follows from the facts that $F_{\mathcal{T}}$ and $\mathcal{V}_{\mathcal{T}}$ are isomorphic and all the leading entries of nonzero rows are disjoint since $\overline{\mathcal{M}_{\mathcal{T}}(F)}$ is in the reduced row echelon form. $\square$

Lemma 2 *Let $\overline{\mathcal{M}_{\mathcal{T}}(F)}$ be the reduced row echelon form of $\mathcal{M}_{\mathcal{T}}(F)$. If $\mathcal{T}$ has a large enough number of elements, the following $G_{\mathcal{T}}$ is a Gröbner basis for I.*

$$G_{\mathcal{T}} = \left\{ \phi_{\mathcal{T}}^{-1}(\overrightarrow{p}) \;\middle|\; \overrightarrow{p} \text{ is a row vector of } \overline{\mathcal{M}_{\mathcal{T}}(F)} \right\}.$$

$\triangleleft$

Proof The Buchberger algorithm guarantees that $G \subseteq F_{\mathcal{T}}$ if $\mathcal{T}$ has a large enough number of elements. Therefore, $G_{\mathcal{T}}$ satisfies the condition of Definition 1 since we have $g_i(\vec{x}) \in G_{\mathcal{T}}$, $i = \{1, \ldots, r_G\}$ by Lemma 1. $\square$

Example 1 We compute the reduced Gröbner basis w.r.t. the graded lexicographic order for the ideal generated by the following polynomials.

$$F_{ex} = \{x^2 - 2y^2,\ 4x^2y + 3xy,\ 2x^2y + \frac{1}{2}x^2 + \frac{3}{2}xy - y^2\}. \tag{2.1}$$

We construct the following matrix $\mathcal{M}_{\mathcal{T}}(F_{ex})$ with $\mathcal{T} = \{x^4, x^3y, x^2y^2, xy^3, y^4, x^3, x^2y, xy^2, y^3, x^2, xy, y^2\}$ (this set is large enough and not the minimum set for F_{ex}) and compute $\overline{\mathcal{M}_{\mathcal{T}}(F_{ex})}$, the reduced row echelon form of $\mathcal{M}_{\mathcal{T}}(F_{ex})$.

$$\mathcal{M}_{\mathcal{T}}(F_{ex}) = \begin{pmatrix} 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -2 \\ 0 & 4 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 3 & 0 \\ 0 & 2 & 0 & 0 & 0 & \frac{1}{2} & \frac{3}{2} & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & \frac{1}{2} & \frac{3}{2} & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & \frac{1}{2} & \frac{3}{2} & -1 \end{pmatrix}, \quad \overline{\mathcal{M}_{\mathcal{T}}(F_{ex})} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{3}{2} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{9}{16} & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & \frac{3}{4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{9}{32} & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & \frac{3}{8} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & \frac{3}{4} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \frac{3}{8} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

We have $G_{\mathcal{T}} = \{x^4 + \frac{3}{2}xy^2, x^3y - \frac{9}{16}xy, x^2y^2 + \frac{3}{4}xy^2, xy^3 - \frac{9}{32}xy, y^4 + \frac{3}{8}xy^2, x^3 - 2xy^2, x^2y + \frac{3}{4}xy, y^3 + \frac{3}{8}xy, x^2 - 2y^2\}$ hence we obtain the following reduced Gröbner basis if we delete all the redundant elements.

$$G = \left\{x^2 - 2y^2, y^3 + \tfrac{3}{8}xy\right\}.$$

We note that in general we must check whether G is actually a Gröbner basis for I, or not, since Lemma 2 is valid only when $\mathcal{T}$ has a large enough number of elements. $\triangleleft$

2.1 Relation between Rank and Priori Errors

It seems that F_{fp} in (1.1) at the beginning of Section 1.1 is one of perturbed sets of F_{ex} in (2.1) at Example 1 by some errors. We note that we can not say this is correct in practice. In general, no one knows the actual polynomial set (with no errors) of F_{fp}. This means that F_{ex} is just one of (infinitely many) possible cases of F_{fp}. The aim of this paper is adding mathematical and practical meanings to the situation that we compute a Gröbner basis for F_{fp} with this property.

To make the situation much simpler, we consider the following polynomial set $\tilde{F}_{fp}$.

$$\tilde{F}_{fp} = \{x^2 - 2y^2, 4x^2y + 3xy, 2x^2y + \frac{1}{2}x^2 + 1.500001xy - y^2\}. \tag{2.2}$$

The difference between polynomial sets F_{ex} and $\tilde{F}_{fp}$ is just $0.000001xy$, and we have the following $\mathcal{M}_{\mathcal{T}}(\tilde{F}_{fp})$ which is very similar to $\mathcal{M}_{\mathcal{T}}(F_{ex})$. However, ranks of $\mathcal{M}_{\mathcal{T}}(F_{ex})$ and $\mathcal{M}_{\mathcal{T}}(\tilde{F}_{fp})$ are not the same: $\mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F_{ex})) = 9$ and $\mathrm{rank}(\mathcal{M}_{\mathcal{T}}(\tilde{F}_{fp})) = 11$.

$$\mathcal{M}_{\mathcal{T}}(\tilde{F}_{fp}) = \begin{pmatrix}
1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & -2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -2 \\
0 & 4 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 4 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 3 & 0 \\
0 & 2 & 0 & 0 & 0 & \frac{1}{2} & 1.500001 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 2 & 0 & 0 & 0 & \frac{1}{2} & 1.500001 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & \frac{1}{2} & 1.500001 & -1
\end{pmatrix}.$$

Moreover, the singular values of $\mathcal{M}_{\mathcal{T}}(\tilde{F}_{fp})$ are followings: {6.90335, 6.06098, 4.16844, 2.52968, 2.44446, 2.24685, 1.9732, 1.86836, 0.96183, 7.04525×10^{-7}, 2.89525×10^{-7}}.

By Lemma 2 and the rank of $\mathcal{M}_{\mathcal{T}}(\tilde{F}_{fp})$, if $\mathcal{T}$ also has a large enough number of elements for $\tilde{F}_{fp}$, a Gröbner basis for the ideal generated by $\tilde{F}_{fp}$ has two extra polynomials against a Gröbner basis $G_{\mathcal{T}}$ for F_{ex}. These two polynomials must be generated by the tiny difference between the matrices and can be thought to be aftereffects of noise. This means that the difference ($0.000001xy$) hides the well structured result $G_{\mathcal{T}}$ and G for F_{ex}. Therefore, if we'd like to unveil the well structured result from the given inexact input, finding the rank deficient matrix $\mathcal{M}_{\mathcal{T}}(F_{ex})$ from $\mathcal{M}_{\mathcal{T}}(\tilde{F}_{fp})$ is the problem to be solved.

Moreover, it easily happens that the rank of a matrix will increase by some perturbations (errors) though it also happens but rarely that the rank of a matrix may decrease, since a rank increase or decrease corresponds to an increase or decrease in entropy, respectively. Hence, the difference of ranks can be thought to be aftereffects of a priori errors. This situation is very similar to several algorithms for approximate polynomial GCD, factorization and so on [9, 3, 29, 8]. Therefore, computing a Gröbner basis with inexact inputs also may be done by computing a rank deficient (structured) matrix and some known algorithms using floating point arithmetic, based on the conventional Buchberger algorithm.

Remark 1 One may think that there is a big difference between exact and numerical ranks: a small element in absolute value may be ignored even if its relative error is much smaller than others. Hence a resulting Gröbner basis can be far from the original. This possibly happens but inevitable for the second kind of problem since any relativeness of errors is easily lost even if we just do only a single addition of polynomials (see the discussion in [15]). Again, please note that the aim of this paper is finding a meaningful result hidden by a priori errors from the given inexact input, and in general it is difficult as shown in Section 1.1 that we get some practical results by known methods (e.g. CGS). $\triangleleft$

3 Structured Gröbner basis

For approximate GCD and factorization, we usually assume that inexact input polynomials (with a priori errors) and unknown exact polynomials (with no error) have the same degree at most. This means that input and hidden desired polynomials are supposed to have the same structure. In order to state this kind of constraint in its full generality, we define a structured polynomial set. Consider a mapping $\mathfrak{S}_i$ $(i = 1, \ldots, k)$ from a parameter space $\mathbf{C}^{n_i}$ to a set of polynomials $\mathbf{C}[\vec{x}]$. A polynomial set $F = \{f_1, \ldots, f_k\}$ is called $\mathfrak{S}$-structured if each element $f_i(\vec{x})$ of the set is in the image of $\mathfrak{S}_i$, i.e., if there exists a parameter $\vec{p_i} \in \mathbf{C}^{n_i}$, such that $f_i(\vec{x}) = \mathfrak{S}_i(\vec{p_i})$.

In this paper, we try to compute a numerical Gröbner basis by finding a rank deficient matrix as noted in the previous section. From this point of view, we introduce the following structured Gröbner basis (SGB for short).

Definition 2 (Structured Gröbner basis) *We say G is a $\mathfrak{S}$-structured Gröbner basis for F with tolerance $\varepsilon \in \mathbf{R}_{\geq 0}$, rank deficiency $d \in \mathbf{Z}_{\geq 0}$ and set of terms $\mathcal{T}$ if they satisfy the following conditions:*

1. *G is a Gröbner basis for the ideal generated by the following $F_{st} = \{f_{st,1}, \ldots, f_{st,k}\} \in \mathbf{C}[\vec{x}]$.*
2. *F and F_{st} are $\mathfrak{S}$-structured polynomial sets, i.e., there exists parameters $\vec{p_i}, \vec{p_{st}}_i \in \mathbf{C}^{n_i}$, such that $f_i(\vec{x}) = \mathfrak{S}_i(\vec{p_i})$ and $f_{st,i}(\vec{x}) = \mathfrak{S}_i(\vec{p_{st}}_i)$.*
3. *$\|(\vec{p_1} \ldots \vec{p_k}) - (\vec{p_{st}}_1 \ldots \vec{p_{st}}_k)\| = \varepsilon$ where $\|\cdot\|$ denotes a suitable vector norm.*
4. *$\mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F_{st})) = \mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F)) - d$.*

$\triangleleft$

By this definition, the aim of this paper becomes to find a method to compute a $\mathfrak{S}$-structured Gröbner basis for the given polynomial set F with the tolerance ε, the rank deficiency d, the set of terms $\mathcal{T}$ and $\mathfrak{S}$-structured polynomial set F_{st}.

Example 2 We show a $\mathfrak{S}$-structured Gröbner basis for the following $\tilde{F}_{fp}$ introduced in (2.2) at Section 2.1.

$$\tilde{F}_{fp} = \{x^2 - 2y^2, 4x^2y + 3xy, 2x^2y + \frac{1}{2}x^2 + 1.500001xy - y^2\}.$$

We take the following structure specification $\mathfrak{S}$ for example.

$$\begin{array}{rrcl} \mathfrak{S}_1: & (p_1\ p_2) & \mapsto & p_1x^2+p_2y^2, \\ \mathfrak{S}_2: & (p_3\ p_4) & \mapsto & p_3x^2y+p_4xy, \\ \mathfrak{S}_3: & (p_5\ p_6\ p_7\ p_8) & \mapsto & p_5x^2y+p_6x^2+p_7xy+p_8y^2. \end{array}$$

In this setting, the following G is a $\mathfrak{S}$-structured Gröbner basis for $\tilde{F}_{fp}$ w.r.t. the graded lexicographic order with tolerance $\varepsilon = 0.000001$ in 2-norm, rank deficiency $d = 2$ and set of terms $\mathcal{T} = \{x^4, x^3y, x^2y^2, xy^3, y^4, x^3, x^2y, xy^2, y^3, x^2, xy, y^2\}$.

$$G = \left\{x^2 - 2y^2, y^3 + \tfrac{3}{8}xy\right\}.$$

Moreover, for this idealized case, we have

$$F_{st} = \{x^2 - 2y^2,\ 4x^2y + 3xy,\ 2x^2y + \frac{1}{2}x^2 + \frac{3}{2}xy - y^2\}.$$

Note that $\mathfrak{S}$-structured Gröbner basis is not unique even for the fixed $\mathfrak{S}$, ε, d and $\mathcal{T}$. The above G is just one of them. ◁

4 Algorithm for computing SGB

In this section, we introduce an algorithm for computing $\mathfrak{S}$-structured numerical Gröbner basis. The algorithm is not for exact $\mathfrak{S}$-structured Gröbner basis but for numerical ones since in general it is difficult to compute a nearby exact rank deficient structured matrix. We give some remarks about this difficulty in Section 5.

To find a $\mathfrak{S}$-structured polynomial set F_{st} we compute a rank deficient structured matrix. This can be done by several known methods for the following problem (SLRA: Structured Low Rank Approximation).

Definition 3 (SLRA) *Given a structure specification $\mathcal{S} : \mathbf{R}^{n_\alpha} \to \mathbf{R}^{m\times n}$, a parameter vector $\vec{p} \in \mathbf{R}^{n_\alpha}$, a vector norm $\|\cdot\|$, and an integer r, $0 < r < \min\{m, n\}$, find a vector $\vec{p^*}$ such that*

$$\min_{\vec{p^*}} \|\vec{p} - \vec{p^*}\| \ \ \textit{and} \ \operatorname{rank}(\mathcal{S}(\vec{p^*})) \le r.$$

◁

The SLRA problem can be solved by the lift-and-project method or solvers for the STLS problem (Structured Total Least Squares) under some convergent conditions (see Chapter 5 in [2]) and easily extended to complex numbers (see Chapter 2 in [12]). We note that the difference between the SLRA and STLS problems is the objective rank deficiency. The STLS problem is a special case of the SLRA problem with $r = \min\{m, n\} - 1$, however in some cases we can convert the SLRA problem to the STLS problem though this conversion (see [16, 2]) is not easy task for our problem according to our experiments.

By the above discussions, we have the following algorithm for computing a $\mathfrak{S}$-structured numerical Gröbner basis.

Algorithm 1 ($\mathfrak{S}$-structured numerical Gröbner basis)

Input: $F = \{f_1(\vec{x}), \ldots, f_k(\vec{x})\} \subset \mathbf{C}[\vec{x}]$, a term order $\succ$ and a structure specification $\mathfrak{S}$.

Output: a $\mathfrak{S}$-structured Gröbner basis G for F with the tolerance ε, the rank deficiency d, the set of terms $\mathcal{T}$ and $\mathfrak{S}$-structured polynomial set F_{st}, or *failed*.

1. Compute a numerical Gröbner basis $\tilde{G}$ for the ideal generated by F, by some known algorithms and determine a suitable set of terms $\mathcal{T}$ based on the result.
2. Construct $\mathcal{M}_{\mathcal{T}}(F) \in \mathbf{C}^{m\times n}$, compute its non-zero singular values σ_i $(i = 1, \ldots, r_{org})$ and determine a suitable rank deficiency d (Take the largest d such that $\sigma_{r_{org}-d+1}/\sigma_{r_{org}-d} < 10^{-2}$ for example). If there is no such d found, output *failed*.
3. By a solver for the SLRA problem, find a $\mathfrak{S}$-structured polynomial set F_{st} with the rank deficiency d satisfying $\mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F_{st})) = \mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F)) - d$ and compute the tolerance ε. If there is no such F_{st} found, output *failed*.
4. Compute a numerical Gröbner basis G for the ideal generated by F_{st}, by some known algorithms and output $\{G,\ \varepsilon,\ d,\ \mathcal{T},\ F_{st}\}$.

$\triangleleft$

The above algorithm is very simple however it works as shown in the following examples and for F_{fp} in (1.1) at the beginning of Section 1.1.

Example 3 Suppose that we compute a $\mathfrak{S}$-structured numerical Gröbner basis w.r.t. the lexicographic order for the following polynomial set F.

$$F = \{x^3 + x^2y^2,\ x^2y^2 - y^3,\ -x^2y + 1.000001x^2 + xy^2 + 0.999999y^2\}.$$

This is the following polynomial set F_δ with $\delta = 10^{-6}$, which is introduced by Sasaki and Kako (Example 5,[17]).

$$F_\delta = \{x^3 + x^2y^2,\ x^2y^2 - y^3,\ -x^2y + (1+\delta)x^2 + xy^2 + (1-\delta)y^2\}.$$

In this case, by the algorithm (appGröbner, [17]) with initial precision[2] $\varepsilon_{Init} = 10^{-16}$ and approximate-zero threshold $\varepsilon_Z = 10^{-15}$, we get the following Gröbner basis and the set of terms $\mathcal{T}$.

$$G_{az} = \{1.0x^2 - 2.000001y^3 + 0.999998y^2,\ 1.0xy^2 + 1.000001y^3,\ 1.0y^4 - 1.000001y^3\},$$
$$\mathcal{T} = \{x^4y^4,\ x^4y^3,\ x^4y^2,\ x^4y,\ x^3y^5,\ x^3y^4,\ x^3y^3,\ x^3y^2,\ x^3y,\ x^3,\ x^2y^5,\ x^2y^4,\ x^2y^3,\ x^2y^2,\ x^2y,\ x^2,\ xy^6,\ xy^5,\ xy^4,\ xy^3,\ xy^2,\ y^6,\ y^5,\ y^4,\ y^3,\ y^2\}.$$

We construct $\mathcal{M}_{\mathcal{T}}(F) \in \mathbf{C}^{30\times 26}$ and determine the rank deficiency $d = 4$ since we get the following singular values and $\sigma_{26-4+1}/\sigma_{26-4} = 1.47591 \times 10^{-6}$.

$$\{2.12971,\ 1.84588,\ \ldots\ \text{(18 elements snipped)}\ \ldots,\ 0.282599,\ 0.238185,\ 3.51538 \times 10^{-7},\ 3.04624 \times 10^{-7},\ 1.66608 \times 10^{-16},\ 1.66608 \times 10^{-16}\}.$$

[2]In this paper, we assume that we do not have any information on a priori errors hence we suppose that all the input coefficients uniformly have the machine-precision.

We take the following structure specification $\mathfrak{S}$.

$$\begin{array}{rrcl}
\mathfrak{S}_1: & (p_1\ p_2) & \mapsto & p_1x^3+p_2x^2y^2, \\
\mathfrak{S}_2: & (p_3\ p_4) & \mapsto & p_3x^2y^2+p_4y^3, \\
\mathfrak{S}_3: & (p_5\ p_6\ p_7\ p_8) & \mapsto & p_5x^2y+p_6x^2+p_7xy^2+p_8y^2.
\end{array}$$

By the lift-and-project method (a solver for the SLRA problem), we get the following $\mathfrak{S}$-structured polynomial set F_{st} with tolerance $\varepsilon = 7.2326\times10^{-7}$ in 2-norm, such that $\mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F_{st}))$ $= \mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F)) - d$.

$$\begin{aligned}
F_{st} = \{&1.00000023x^3+0.9999998x^2y^2,\ 1.00000018x^2y^2-0.9999998y^3,\\
&-0.99999968x^2y+1.00000073x^2+1.00000027xy^2+0.99999932y^2\}.
\end{aligned}$$

Finally, by the algorithm (appGröbner, [17]), we get the following $\mathfrak{S}$-structured numerical Gröbner basis for F.

$$\begin{aligned}
G = \{&1.0x^2-0.99999802y^5-0.99999908y^4+0.99999859y^2,\\
&1.0xy^2+0.99999954y^4,\ 1.0y^6-1.0000006y^3\}.
\end{aligned}$$

We note that this result is compatible to the following comprehensive Gröbner system for F if we think that δ represents a priori errors.

$$\begin{cases}
\{x^2-y^5-y^4+y^2,\ xy^2+y^4,\ y^6-y^3\} & (\delta=0),\\
\{2x^2+xy^2-y^3,\ xy^3+y^3,\ y^4-y^3\} & (\delta=1),\\
\{x^3+y^3,\ x^2y+y^3-2y^2,\ xy^2+y^3,\ y^4-y^3\} & (\delta=-1),\\
\{(1+\delta)^2x^2+2xy^2-2\delta y^3+(1-\delta^2)y^2,\ -(1-\delta)(xy^2+y^3),\ y^4-y^3\} & (\delta^3\neq\delta).
\end{cases}$$

$\triangleleft$

Example 4 Suppose that we compute a $\mathfrak{S}$-structured numerical Gröbner basis w.r.t. the graded lexicographic order for the following polynomial set F.

$$\begin{aligned}
F = \{&1.99974x^2yz+0.999742x^2,\ 3.00263xyz^2-2.00013yz,\\
&2.99826x^2z-2.00053x-2.00006yz-0.99943\}.
\end{aligned}$$

This is generated by adding small random perturbations to the following polynomial set F_{ex} introduced by Sasaki and Kako (Example 6,[17]).

$$F_{ex} = \{2x^2yz+x^2,\ 3xyz^2-2yz,\ 3x^2z-2x-2yz-1\}.$$

In this case, by the algorithm (appGröbner, [17]) with initial precision $\varepsilon_{Init} = 10^{-16}$ and approximate-zero threshold $\varepsilon_Z = 10^{-15}$, we get the following Gröbner basis and the set of terms $\mathcal{T}$.

$$G_{az} = \{1.0x-0.142550, 1.0y+0.106986, 1.0z-4.67292\},$$

$$\begin{aligned}
\mathcal{T} = \{&x^5yz^3, x^4y^2z^3, x^4yz^4, x^3y^2z^4, x^2y^2z^5, x^5yz^2, x^4y^2z^2,\\
&\ldots\ (42\ \text{elements snipped})\ \ldots,\\
&x^2z, xyz, xz^2, y^2z, yz^2, x^2, xy, xz, yz, z^2, x, y, z, 1\}.
\end{aligned}$$

We construct $\mathcal{M}_\mathcal{T}(F) \in \mathbf{C}^{75\times 63}$ and determine the rank deficiency $d = 7$ since we get the following singular values and $\sigma_{63-7+1}/\sigma_{63-7} = 0.00724885$.

$$\{2.14386,\ 2.08212,\ \ldots\ (52 \text{ elements snipped})\ \ldots,\ 0.04441,\ 0.0384606,\ 0.000278795,\ 0.000234535,\ 0.000181527,\ 0.000128085,\ 0.0000768688,\ 1.78766\times 10^{-16},\ 1.78766\times 10^{-16}\}.$$

We take the following structure specification $\mathfrak{S}$.

$$\begin{array}{rrcl} \mathfrak{S}_1: & (p_1\ p_2) & \mapsto & p_1x^2yz + p_2x^2, \\ \mathfrak{S}_2: & (p_3\ p_4) & \mapsto & p_3xyz^2 + p_4yz, \\ \mathfrak{S}_3: & (p_5\ p_6\ p_7\ p_8) & \mapsto & p_5x^2z + p_6x + p_7yz + p_8. \end{array}$$

We get the following $\mathfrak{S}$-structured polynomial set F_{st} with tolerance $\varepsilon = 0.00200393$ in 2-norm, such that $\text{rank}(\mathcal{M}_\mathcal{T}(F_{st})) = \text{rank}(\mathcal{M}_\mathcal{T}(F)) - d$, by the lift-and-project method (a solver for the SLRA problem).

$$F_{st} = \{1.99978x^2yz + 0.99966x^2,\ 3.00199xyz^2 - 2.00109yz,\ 2.99915x^2z - 1.9992x - 1.99991yz - 0.999725\}.$$

Finally, by the algorithm (appGröbner, [17]), we get the following $\mathfrak{S}$-structured numerical Gröbner basis for F.

$$G = \{1.0yz + 0.499885,\ 1.0x + 1.33349y\}.$$

We note that the result is very similar to the following exact $\mathfrak{S}$-structured Gröbner basis which is the reduced Gröbner basis for the original polynomial set without perturbations though as noted in former sections we can not say that this is the unique correct answer. However, the result shows that we extracted a meaningful numerical Gröbner basis from the given inexact input.

$$G_{ex} = \{yz + \frac{1}{2},\ x + \frac{4}{3}y\}.$$

◁

As shown in the above examples, Algorithm 1 can compute a $\mathfrak{S}$-structured numerical Gröbner basis. However, we note that the algorithm works only as a preconditioning method and does not improve the numerical stability. It is the different problem. For example, the algorithm outputs *failed* and does not give us any new useful information on the following polynomial set used by Kondratyev and others [10].

$$\begin{cases} h_1 = 1.027748y^2 - 0.467871xy + 2.972252x^2 + 0.662026y + 0.0785252x - 3.888889, \\ h_2 = 3.958378y^2 + 0.701807xy + 1.041622x^2 - 0.0785252y + 0.662026x - 3.888889. \end{cases}$$

The reason is very simple. This polynomial set does not have any other $\mathfrak{S}$-structured polynomial set with any small tolerance ε and rank deficiency $d > 0$ if we take $\mathcal{T} = \{y^6, xy^5, y^5, x^2y^4, xy^4, y^4, x^3y^3, x^2y^3, xy^3, y^3, x^4y^2, x^3y^2, x^2y^2, xy^2, y^2, x^5y, x^4y, x^3y, x^2y, xy, y, x^6, x^5, x^4, x^3, x^2, x, 1\}$ for example. In fact, h_1 and h_2 are introduced by Stetter [22], by decimal approximations hence they do not have a priori errors from the data mining point of view in this paper.

5 Remarks

In general the SLRA problem is NP-hard except for a few special cases (see [13] for details). Our algorithm fully depends on the SLRA problem hence Algorithm 1 is not efficient in computation time. However, the lift-and-project method can find a local optimum in terms of the difference ε between F and F_{st} if each iteration constructs a distinct projection (see [5] for details). Our algorithm is to find a rank deficient matrix from the matrix having a large gap among its singular values hence this helps the lift-and-project method to start with a matrix near its local optimum.

One may be interested in the effectiveness of Algorithm 1 and think that the lift-and-project method converges only for toy examples. To see this, we generated 3 cases (A,B and C) of 100 polynomial sets in two variables and apply the algorithm with the graded lexicographic order. Each polynomial set is randomly generated as follows: 1) generate a Gröbner basis with 3 basis polynomials of total degree t_b at most having 2,3 or 4 terms with integer coefficients $\in [-10, 10]$, 2) generate a polynomial set with 4 syzygies of total degree t_s at most having 2,3 or 4 terms with integer coefficients $\in [-10, 10]$, and 3) perturb the all coefficients by relative noises $\in [-10^{-5}, 10^{-5}]$.

In Table 1, '$\#\vec{p}$' denotes the average of the number of parameters of structure specifications used in the algorithm, 'w/o' denotes the number of good results by our appGröbner implementation, 'with' denotes the number of good results by Algorithm 1, 'ε' denotes the average of tolerances of good results in 2-norm, and 'diff' denotes the average of differences between the resulting Gröbner basis by Algorithm 1 and the Gröbner basis for the exact polynomial set in 2-norm, where "good result" means that its resulting 'diff' is enough small.

It shows that our algorithm can extract a meaningful basis for the inexact input which the known algorithm can not work for. Note that our appGröbner implementation can work more if we have enough information on a priori errors (e.g. suitable initial precision and approximate-zero threshold). Moreover, for most of the failure polynomial sets, Algorithm 1 can compute meaningful results if we specify suitable rank deficiencies though this is difficult since in such cases the gaps among singular values are not large.

case	t_s	t_b	$\#\vec{p}$	w/o	with	ε	diff
A	1	2	33.9	0	96	0.000434	0.000043
B	1	3	40.7	3	45	0.000458	0.000034
C	2	2	42.9	15	89	0.000345	0.000040

Table 1: Experimental Results

For polynomials in more than two variables or polynomials of higher degrees, it becomes difficult to determine a suitable set of terms $\mathcal{T}$ since most of generated syzygies during computations in Step 1 becomes unreliable. This forces us to use much larger $\mathcal{T}$ hence the lift-and-project method has to operate with a huge matrix and its convergence speed becomes very slow. A simple workaround is replacing Step 4 with the following.

4. Compute a numerical Gröbner basis G for the ideal generated by F_{st}, by some known algorithms. Output $\{G,\ \varepsilon,\ d,\ \mathcal{T},\ F_{st}\}$ if the suitable set of terms $\mathcal{T}$ is unchanged, otherwise go to Step 2 with a new suitable set of terms $\mathcal{T}$.

However, this step tends to cause an infinite loop and not so efficient according to our experiments hence this does not resolve the problem.

Therefore, the followings are open questions for further work.

- Finding a method to determine a suitable set of terms $\mathcal{T}$ in terms of $\mathfrak{S}$-structured Gröbner basis for the input.
- Finding a better method for our SLRA problem.
- Finding a method to compute an optimal $\mathfrak{S}$-structured numerical Gröbner basis w.r.t. the tolerance ε.

Acknowledgements

The author wishes to thank Prof. Nabeshima and Mr. Kurata for their contributions in computing the large comprehensive Gröbner system (e.g. 263 segments). The information is very helpful to note the introductory section.

References

[1] T. Becker and V. Weispfenning. *Gröbner bases*, volume 141 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1993. A computational approach to commutative algebra, In cooperation with Heinz Kredel.

[2] P. Boito. *Structured Matrix Based Methods for Approximate GCD*. Ph.D. Thesis. Department of Mathematics, University of Pisa, Italia, 2007.

[3] B. Botting. *Structured total least squares for approximate polynomial operations*. Master's Thesis. School of Computer Science, University of Waterloo, 2004.

[4] M. Byröd, K. Josephson, and K. Åström. Fast optimal three view triangulation. In Y. Yagi, I. S. Kweon, S. B. Kang, and H. Zha, editors, *Asian Conference on Computer Vision*, 2007.

[5] M. T. Chu, R. E. Funderlic, and R. J. Plemmons. Structured low rank approximation. *Linear Algebra Appl.*, 366:157–172, 2003. Special issue on structured matrices: analysis, algorithms and applications (Cortona, 2000).

[6] J.-C. Faugére. A new efficient algorithm for computing Gröbner bases (F_4). *J. Pure Appl. Algebra*, 139(1-3):61–88, 1999.

[7] J.-C. Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, pages 75–83 (electronic), New York, 2002. ACM.

[8] S. Gao, E. Kaltofen, J. May, Z. Yang, and L. Zhi. Approximate factorization of multivariate polynomials via differential equations. In *ISSAC 2004*, pages 167–174. ACM, New York, 2004.

[9] E. Kaltofen, Z. Yang, and L. Zhi. Approximate greatest common divisors of several polynomials with linearly constrained coefficients and singular polynomials. In *ISSAC '06: Proceedings of the 2006 international symposium on Symbolic and algebraic computation*, pages 169–176, New York, NY, USA, 2006. ACM Press.

[10] A. Kondratyev, H. J. Stetter, and S. Winkler. Numerical computation of gröbner bases. In *Proceedings of CASC2004 (Computer Algebra in Scientific Computing)*, pages 295–306, 2004.

[11] D. Lazard. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In *Computer algebra (London, 1983)*, volume 162 of *Lecture Notes in Comput. Sci.*, pages 146–156. Springer, Berlin, 1983.

[12] P. Lemmerling. *Structured total least squares: analysis, algorithms and applications.* Ph.D. Thesis. Faculty of Applied Sciences, K.U. Leuven, Belgium, 1999.

[13] I. Markovsky. Structured low-rank approximation and its applications. *Automatica J. IFAC*, 44(4):891–909, 2008.

[14] K. Nabeshima. A speed-up of the algorithm for computing comprehensive gröbner systems. In *ISSAC 2007: Proceedings of the 2007 international symposium on Symbolic and algebraic computation*, pages 299–306, New York, NY, USA, 2007. ACM.

[15] K. Nagasaka. A study on gröbner basis with inexact input. In *Proceedings of CASC 2009*, volume 5743 of *Lecture Notes in Comput. Sci.*, pages 247–258. Springer, Berlin, 2009.

[16] H. Park, L. Zhang, and J. B. Rosen. Low rank approximation of a Hankel matrix by structured total least norm. *BIT*, 39(4):757–779, 1999.

[17] T. Sasaki and F. Kako. Computing floating-point gröbner bases stably. In *Proceedings of SNC 2007*, pages 180–189. ACM, New York, 2007.

[18] T. Sasaki and F. Kako. Floating-point gröbner basis computation with ill-conditionedness estimation. In *Proceedings of ASCM 2007*, volume 5081 of *Lecture Notes in Comput. Sci.*, pages 278–292. Springer, Berlin, 2008.

[19] K. Shirayanagi. An algorithm to compute floating point gröbner bases. In *Proceedings of the Maple summer workshop and symposium on Mathematical computation with Maple V : ideas and applications*, pages 95–106, Cambridge, MA, USA, 1993. Birkhauser Boston Inc.

[20] K. Shirayanagi. Floating point gröbner bases. In *Selected papers presented at the international IMACS symposium on Symbolic computation, new trends and developments*, pages 509–528, Amsterdam, The Netherlands, The Netherlands, 1996. Elsevier Science Publishers B. V.

[21] K. Shirayanagi and M. Sweedler. A theory of stabilizing algebraic algorithms. *Technical Report 95-28*, pages 1–92, 1995. http://www.ss.u-tokai.ac.jp/s̃hirayan/msitr95-28.pdf.

[22] H. J. Stetter. Stabilization of polynomial systems solving with Groebner bases. In *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation (Kihei, HI)*, pages 117–124 (electronic), New York, 1997. ACM.

[23] H. J. Stetter. Approximate gröbner bases – an impossible concept? In *Proceedings of SNC 2005 (Symbolic-Numeric Computation)*, pages 235–236, 2005.

[24] A. Suzuki. Computing gröbner bases within linear algebra. In *Proceedings of CASC 2009*, volume 5743 of *Lecture Notes in Comput. Sci.*, pages 310–321. Springer, Berlin, 2009.

[25] C. Traverso and A. Zanoni. Numerical stability and stabilization of groebner basis computation. In *ISSAC 2002: Proceedings of the 2002 international symposium on Symbolic and algebraic computation*, pages 262–269, New York, NY, USA, 2002. ACM.

[26] V. Weispfenning. Comprehensive Gröbner bases. *J. Symbolic Comput.*, 14(1):1–29, 1992.

[27] V. Weispfenning. Gröbner bases for inexact input data. In *Proceedings of CASC 2003 (Computer Algebra in Scientific Computing)*, pages 403–411, 2002.

[28] V. Weispfenning. Canonical comprehensive Gröbner bases. *J. Symbolic Comput.*, 36(3-4):669–683, 2003. International Symposium on Symbolic and Algebraic Computation (ISSAC 2002) (Lille).

[29] L. Zhi. Displacement structure in computing approximate GCD of univariate polynomials. In *Computer mathematics*, volume 10 of *Lecture Notes Ser. Comput.*, pages 288–298. World Sci. Publ., River Edge, NJ, 2003.

The preliminary implementation of Algorithm 1 on *Mathematica* 7.0, used in this paper can be found at “ `http://wwwmain.h.kobe-u.ac.jp/~nagasaka/research/snap/sgb.nb` ”.