Intervals, Syzygies, Numerical Gröbner Bases : A Mixed Study

Marco Bodrato and Alberto Zanoni

Dipartimento di Matematica “Leonida Tonelli” – Università di Pisa
Largo B. Pontecorvo 5 – 56127 Pisa, Italy
{bodrato, zanoni}@posso.dm.unipi.it

Abstract. In Gröbner bases computation, as in other algorithms in commutative algebra, a general open question is how to guide the calculations coping with numerical coefficients and/or not exact input data. It often happens that, due to error accumulation and/or insufficient working precision, the obtained result is not one expects from a theoretical derivation. The resulting basis may have more or less polynomials, a different number of solution, roots with different multiplicity, another Hilbert function, and so on. Augmenting precision we may overcome algorithmic errors, but one does not know in advance how much this precision should be, and a trial–and–error approach is often the only way to follow. Coping with initial errors is an even more difficult task. In this experimental work we propose the combined use of syzygies and interval arithmetic to decide what to do at each critical point of the algorithm.

AMS Subject Classification: 13P10, 65H10, 90C31

Keywords and phrases: Gröbner bases, numerical coefficients, syzygies

1 Introduction

For a general reference to Gröbner bases, we cite [1], [6], [5], [8] and [9]. Numerical stability for Gröbner bases computation has been studied by, among others, Stetter [14], [15] who considers coefficient sizes to influence term ordering. Shirayanagy [13] gives a theoretical basis to floating point computation for systems with exact coefficients, using a sufficiently high working precision. His stabilization technique is based on a clever rewriting rule concerning zero testing. Unfortunately, no upper bound on the initial sufficient precision is known. Migheli [12] uses Hybrids coefficients $H = (n, f)$ – where n is a number modulo a prime and f a floating point value – to measure the stabilization with respect to small variations. Zanoni [20], [21] mimics Migheli's approach using double–floats coefficients $F2 = (f_1, f_2)$, where two different precisions are used at the same time in the computation to control the behaviour of Buchberger algorithm. Traverso [16] presents an idea which is later partially developed in [3].

Being many concepts of algebra intrinsically discrete (dimensions of vector spaces, matrices rank, root multiplicity, etc.), when working with real or complex coefficients, usually the critical points arise when we have to decide if a coefficient c is zero or not. It's what we called the *zero test* (ZT). We are facing a *bifurcation*, and in the general case the algorithm under consideration can follow two completely different paths according to $c = 0$ or $c \neq 0$.

The weak point in easily definable ZTs is lack of flexibility. If we define a too strict one, we could keep too many things that should instead be thrown away. On the contrary, a ZT discarding too many things may equally lead to a not correct behaviour. For example, one may obtain $\langle 1 \rangle$ as final basis.

Moreover, there may be cases in which the *same* ZT is sometimes good, sometimes bad. Infact, it is usually based on some parameters (coefficient sizes, number of correct digits, initial precision, and so on), and to the best of our knowledge nowadays there is no automatic procedure to detect which are the best values (when they exist) for a system to be correctly treated. Some heuristics may be used, but, again, trial–and–error is still the only general method to analyze all the cases which are treatable with a ZT with fixed behaviour.

Let's now consider our case, the Buchberger algorithm. What one looks for is an *adaptive* test, defining a well–determined more general procedure to decide *case by case* the result of the ZT. In other words, we'd like that *the system itself* imposes the conditions that should be satisfied to fulfil, if possible, the ZT. This particularly in the case when initial coefficients are not exact, but known to belong to an interval of possible values, usually determined by available precision.

A zero value is a very particular case, indicating that the system has some hidden relations among its coefficients, which may be not clear from the beginning, because of the limited initial precision. The first problem is to distinguish between possible zeroes, moving initial coefficients in the interval, and values which seems to be zero just because of computational errors. Our philosophy would then be that of trying to (find and) impose these relations during the way, such that at each point of the algorithm in which a coefficient can be zero, it is forced to zero. In a certain sense we are looking for the most "degenerate" polynomial system, having coefficients compatible with initial precision.

With this in mind, in [16] the use of syzygies is proposed. Giving relations expressing the "history" (trace) of all the performed computations, they seem to be a good tool to analyze the current situation when a ZT has to be applied. A prototype implementation for the first experiments is being developed using the C++ `PoSSoLib` library, result of the FRISCO [10] project.

2 Syzygies

Let $\mathbb{K}$ be a field and $\mathcal{F} = \{f_1, \ldots, f_s\} \subset \mathbb{K}[X] = \mathbb{K}[x_1, \ldots, x_n]$ a list of polynomials representing the initial system. In this paper we consider the field of real numbers, $\mathbb{K} = \mathbb{R}$. Syzygies express polynomial relations among the f_i.

Definition 1. *A* syzygy *for $\mathcal{F}$ is a s–tuple $\mathcal{H} = (h_1, \ldots, h_s) \subset \mathbb{K}[X]^s$ such that*

$$\mathcal{H} \cdot \mathcal{F} = \sum_{i=1}^{s} h_i(X) \cdot f_i(X) = 0$$

Let $\mathcal{G} = \{g_1, \ldots, g_t\} \subset \mathbb{K}[X]$ be a system obtained from $\mathcal{F}$ at a certain point of a Buchberger algorithm application. The idea is to keep track of the steps to derive $\mathcal{G}$ from $\mathcal{F}$, in a similar way as in the extended Euclid algorithm for Bezout's identity. In other words, we look for $k_{ij}(X) \in \mathbb{K}[X]$ with

$$g_j(X) = \sum_{i=1}^{s} k_{ij}(X) \cdot f_i(X) \qquad j = 1, ..., t \tag{1}$$

We can obtain syzygies and k_{ij} by using a variant of Buchberger algorithm itself (see [7]). Look at $f_1, \ldots, f_s \in \mathbb{K}[X]$ as vectors $(f_1, 1, 0, \ldots, 0), \ldots, (f_s, 0, \ldots, 0, 1) \in \mathbb{K}[X]^{s+1}$, considered as a $\mathbb{K}[X]$–module, with a term ordering in which comparisons are on pairs (t, i) – where t is a term and i is a position index – such that any term in initial position is greater than whatever term in any other position.

3 Multi-coefficients and Numerical Buchberger Algorithm

The fundamental idea for the `mCoeff` type is to take benefit from the combined use of the floating point and interval arithmetic. A `mCoeff` $m = (m_S, m_L, m_i, m_s)$ is an "enriched" representation of a real number. It has two almost equal floats with different precisions, called short (m_S) and long (m_L) part, respectively, and an interval $m_I = [m_i, m_s]$ containing both m_S and m_L.

Definition 2. *Let $x \in \mathbb{R} \setminus \{0\}$. The natural number $n =$ size(x) such that*

$$x = a \cdot 2^n \qquad \textit{with} \qquad \frac{1}{2} \leqslant |a| < 1.$$

is called the size *of x.*

In our implementation, the initial interval $m_I = [m_i, m_s]$ containing v is computed by default as follows: if $s(v)$ denotes the sign of v we have

$$\begin{array}{l} v > 0 : [v(1 - 2^{-\omega}), v(1 + 2^{-\omega})] \\ v < 0 : [v(1 + 2^{-\omega}), v(1 - 2^{-\omega})] \\ v = 0 : [0, 0] \end{array} \quad \text{or} \quad \begin{array}{l} v \neq 0 : [v(1 - s(v) \cdot 2^{-\omega}), v(1 + s(v) \cdot 2^{-\omega})] \\ v = 0 : [0, 0] \end{array}$$

In a future version, we will be possible to set interval width for each coefficient independently. In some cases a trivial interval $[v, v]$ is necessary: e.g. when we know in advance that a (initial) coefficient is exact, or when dividing a coefficient by itself. Even if in general it is difficult to detect such cases, there is one in which this is evident: when making a polynomial monic (leading coefficient becomes *exactly* 1).

To cope with easily detectable zeroes, we propose the below test. We plan to avoid using user-defined parameters or to change the definition when we'll have evidence of the effectiveness of the approach proposed in the following sections.

ZERO TEST : A `mCoeff` m is considered to be zero when (see [20], [21])

- its short or long part is exactly zero, or
- it is the result of an algebraic sum in which the size drops too much, or
- size(m_S) − size(m_L) grows too much (indicating that all the meaningful digits disappeared, and only garbage remained).

The "too much" quantities are controlled by user-defined integer parameters.

Definition 3. *An interval $I = [a, b] \subset \mathbb{R}$ is* dangerous *when $0 \in I$. A* `mCoeff` *m is dangerous when m_I is dangerous. A polynomial p with* `mCoeff`*s involved in the Buchberger algorithm is dangerous when its leading coefficient is dangerous and p is no more head–reducible with respect to the current basis.*

Let $\mathcal{F}$ be given, together with the finite precision determining the width of the initial intervals I_i for its coefficients. Any system $\mathcal{F}'$ obtained from $\mathcal{F}$ slightly moving the coefficients inside the corresponding I_i is considered as an equally valid representation of the problem to be solved, indistinguishable from $\mathcal{F}$ (we say it is *near* $\mathcal{F}$). This is the freedom we have in looking for the most interesting representative, that is the most *unstable* one, having presumably more interesting properties (such as positive root multiplicity, etc.) than all the near ones. The main point in the Numerical Buchberger Algorithm is the ZT in (IV). Details about NBA are explained in section 6.

Numerical Buchberger Algorithm (NBA)

I Construct the $\overline{\mathcal{F}}$ system with `mCoeff`icients, and start Buchberger algorithm.
II If there is a remaining S-polynomial, compute r, its complete reduction with respect to the current basis, otherwise go to step V.
III If $r = 0$ or its head coefficient c is not dangerous, update the data structures as usual and go to step II, otherwise to IV.
IV Decide if c can really be or is surely different from 0. Update data structures and in the first case modify $\mathcal{F}$ and go to I, otherwise continue from II.
V Extract the final polynomials g_i from the obtained basis, and output them.

4 The zero test

Let $\alpha, \beta, \gamma, \dots \in \mathbb{N}^n$ be multindexes, $\mathbb{T} = \{X^\delta \mid |\delta| = 0, 1, ...\}$ the term basis, and $lt(r) = X^\rho$ the leading term of $r \in \mathbb{K}[X]$. We indicate with $\mathbf{1}$ the term with multiexponent $(0, \dots, 0)$. We consider relations (1) concerning r

$$r(X) = \sum_{\gamma} r_\gamma X^\gamma = \sum_{i=1}^{s} k_i(X) \cdot f_i(X)$$

Let K_i^α be the not zero coefficient of k_i in the monomial containing the term X^α, and F_i^β similarly for f_i. Abusing notation, we also introduce new variables $F = \{F_i^\beta \mid \beta \in B_i, i = 1, \dots, s\}$ and $K = \{K_i^\alpha \mid \alpha \in A_i, i = 1, \dots, s\}$ corresponding to these coefficients. Thanks to the `mCoeff` approach, interval limits for F, K unknowns are also available. If we expand the right hand side of this relation and equalize coefficients, we obtain the following system

$$S_{\rho,c} = \left\{ 0 = \sum_{\substack{i=1\\ \alpha+\beta=\gamma}}^{s} K_i^\alpha F_i^\beta \qquad \gamma > \rho \quad ; \quad r_\gamma = \sum_{\substack{i=1\\ \alpha+\beta=\gamma}}^{s} K_i^\alpha F_i^\beta \qquad \gamma \leqslant \rho \right\} \tag{2}$$

The zero test asks if:

$\boxed{\mathcal{P}_\rho}$: is there a set of values for K_i^α and F_i^β inside their definition intervals satisfying the "$= 0$" equations of $S_{\rho,c}$ *and* such that $lc(r) = r_\rho = c = 0$?

This means to detect an initial system near $\mathcal{F}$ and a set of syzygy values letting the computation trace up to now still be valid, but such that (iterating the process) we can force to zero as many coefficients as possible. We use the following notation for easiness of reference in the following:

1. $\underline{F}_i^\beta$, $\overline{F}_i^\beta$: whose entries are the initial intervals limits for F_i^β, for all i, β.
2. $\underline{K}_i^\alpha$, $\overline{K}_i^\alpha$: similarly for K_i^α, for all possible i and α.

By definition, 0 is never contained in the initial intervals for $\mathcal{F}$. This prevents the trivial null solution to be admissible. One could be tempted to write/solve

$$\mathcal{P}_\rho : \begin{cases} \textbf{min} \;\; c = \left| \displaystyle\sum_{\substack{i=1\\ \alpha+\beta=\rho}}^{s} K_i^\alpha F_i^\beta \right| & (O) \\ 0 = \displaystyle\sum_{\substack{i=1\\ \alpha+\beta=\gamma}}^{s} K_i^\alpha F_i^\beta \qquad \gamma > \rho & (V_1) \\ \left.\begin{array}{l} \underline{F}_i^\beta \leqslant F_i^\beta \leqslant \overline{F}_i^\beta \\ \underline{K}_i^\alpha \leqslant K_i^\alpha \leqslant \overline{K}_i^\alpha \end{array}\right\} \forall\, i, \alpha, \beta & (B) \end{cases} \tag{3}$$

From now on we will call r_ρ the *objective function* (o.f.) and the restrictions (B) for F, K the F- and K-box, respectively ($B = B_F \cup B_K$).

5 System solving

We have a quadratic optimization problem with quadratic restrictions. The general analysis can proceed following two main directions to nullify the o.f.:

Symbolic (S) : Extract the coefficient relations (polynomials in F) that were used in the algorithm. That is, make explicit w.r.t. F the conditions to fulfil for the trace to remain valid up to the current point and, in addition, the new relation (in F again) which represents the o.f., to be forced to zero.

Numeric (N) : Find numerically only some particular values for the K_i^α, F_i^β satisfying above relations.

Obtaining exact, symbolic relations in F is the best way to understand what's going on and set appropriately the initial values for $\mathcal{F}$. Tuning the F such that these relations are satisfied *exactly*, we force the critical dangerous c coefficients to be zero, skipping critical points. Note that each time that new F–relations are determined, they must be still verified in all of the following computations. The wish is that after some solving of $\mathcal{P}_\rho$ critical point systems, we have sufficient new relations in F variables forming a zero dimensional system, whose solution(s) correspond(s) to one (or many) distinguished system $\mathcal{F}'$ near $\mathcal{F}$ with more interesting properties.

We consider the system composed by (V_1) equations as living in $\mathbb{K}[F][K]$ instead than $\mathbb{K}[K, F]$, that is, with F variables considered as parameters. The system becomes then a sparse parametric linear one.

$$\mathcal{M}_{\mathcal{F}} \cdot K = \begin{pmatrix} \cdot & \cdot & \cdot \\ \cdot & F_{ij} & \cdot \\ \cdot & \cdot & \cdot \end{pmatrix} \begin{pmatrix} \cdot \\ K_{ij} \\ \cdot \end{pmatrix} = 0$$

$\mathcal{M}_{\mathcal{F}}$ entries are monic F-monomials. Regarding symbolic manipulations, we can consider the system as temporarily living in $\mathbb{Z}[F][K]$, and pass to $\mathbb{K}$ only if necessary. There is no predefined term ordering for F and K: this freedom will be fruitful.

The symbolic approach (K variables ordering, system reduction, etc.) was described in [3]: we will refer here mainly to the numeric approach, apart from some initial symbolic management we report below, which can anyway be done.

5.1 Preprocessing

Let $lt(f_i) = X^{\delta_i}$: looking for a simplification of the system shape, we render the initial polynomials f_i monic. This gives the advantage, all leading coefficients being exactly equal to 1, that $F_i^{\delta_i}$ variables are now fixed, and there is then no reason at all even to introduce them. We therefore have *once and for all* reduced in a trivial way the research space dimension: $F_i^{\delta_i} = F_{i0}$ $(= 1)$ variables will never appear in V_1.

Definition 4. *A polynomial p (an equation $p = 0$) is* mute *if p is a linear binomial with its two coefficients equal to* 1.

Mute equations are the simplest ones that can appear in the system (we are working with monic f_i polynomials !). Due to their nothing more than "renaming" function, we can substitute (we use only one index for simplicity) as follows

$$K_i + K_j = 0 \qquad \Longrightarrow \qquad K_i = -K_j$$

deleting some K_i once and for all from the system. This helps in reducing the number of variables to be treated, saving thus space and elaboration time for data structures management representing polynomials on a computer.

It helps from the numerical point of view, too. Frequently the intervals of the two variables K_i and K_j are widely different, e.g one is dangerous and the other is not. The simple equality relation allows us to consider the *intersection* of the intervals, and store the new obtained ones for the forthcoming computations.

Generalizing the mute equations interval analysis, for equations like $(F_1)K_1 + (F_2)K_2 = 0$, using the two derived expressions

$$K_1 = -\frac{F_2}{F_1}K_2 \quad ; \quad K_2 = -\frac{F_1}{F_2}K_1$$

we obtain easy alternative formulae to recompute intervals for two K variables, and possibly refine them by intersection.

Definition 5. *A variable v is* single *for a linear system S (S-single, or simply* single *if S is clear from the context) if it appears only once in S (in polynomial p_v).*

If K_1 is V_1-single, we may consider a block term ordering (possibly changing the one we're currently using) such that it is the greatest variable. Then p_{K_1} will have it as leading term, and it will not be used at all, for there is nothing to reduce by it. We can then consider it as virtually discarded from V_1, and look now at $V_1' = V_1 \setminus \{p_{K_1}\}$. It may happen that some other K_2 variable appeared two times in the system, one in p_{K_1} and one elsewhere. Having deleted p_{K_1}, K_2 is now single for V_1', and we can discard p_{K_2}, too. Proceeding this way – possibly continuing changing ordering – until possible, we reduce the size of the really meaningful part of the system (less equations and variables), simulating non-effective reductions at practically no cost.

5.2 Looking for a minimum

Working numerically, we must look in the F-box for an optimal point π for which the o.f. computed absolute value c_π is zero or minimal.

Being ours a (numerical) point-wise approach, we must necessarily enter a cycle of o.f. evaluations to analyze behaviour. Various zero searching criteria – grid analysis, descent gradient, etc. – may be followed, but there may be problems for all of them passing from the continue to the discrete environment (local minima, saddles, etc.) In any case, for all of them we need a procedure to compute the o.f. pointwise for various points in the F-box.

Having performed all the computations with `MCoeffs`, we obtained the desired intervals for every coefficient, but also a distinguished value inside it (approximated by the m_S, m_L corresponding value). We indicate these values for K variables with $\sigma = (\sigma_{ij})$. In order to work mainly with F variables (which K ones really depend on), the body of the cycle may be composed by the following black–box procedure: given an admissible point $\pi = (\pi_{ij})$ *in the F-box*, the corresponding c_π value is returned.

- Specialize $\mathcal{M}_\mathcal{F}$ entries with π $(F_{ij} = \pi_{ij})$, obtaining $\mathcal{M}_\pi$.
- Solve the system $\mathcal{M}_\pi \cdot K = 0$, that is, find $\mathcal{K} = \ker(\mathcal{M}_\pi)$ in the form $K_v = N(\pi) \cdot K_p$, where K_p is the set of remaining "free parameters".
- Eliminate K_v variables in the o.f. by means of the found expressions in terms of K_p ones.
- Now the o.f. depends only on K_p variables: instantiate them with corresponding σ_{ij} values.

We note that, given $\mathcal{F}$, we always use the same σ values for K_p, for every π. When there is only one free parameter $K_p = \{\overline{K}\}$ we have

$$c_\pi = D(\pi) \cdot \overline{K}$$

where D is a rational function, indicating explicitly that and how c depends on initial coefficients, remembering the always present degree of freedom deriving from the possibility to multiply a polynomial for a not zero scalar, signed by the surviving $\overline{K}$ variable. In this case it is more evident that what really counts is essentially working on $D(\pi)$.

If we find an admissible point π_1 such that $s(c_{\pi_1}) \neq s(c_{\pi_0})$ – where π_0 is the one we have obtained in our particular computation – we may easily recover with the desired precision a root of the o.f., because the admissible region (F-box) is *convex*. We consider the segment connecting the two points, reducing thus to the univariate case. Because of the zero theorem for continue functions, there exists $\pi_D = \pi_0 + t \cdot (\pi_1 - \pi_0)$ solving the problem, with $t \in (0, 1)$, and we can approximate it e.g. by successive bisections.

6 The procedure

In the general setting of the NBA, we detail here the things to be done. First of all, we attach a label (a natural number) to every reduced S-polynomial in the course of the computation: we indicate with S_i the i^{th} reduced S-polynomial.

We introduce two data structures (lists): $\mathcal{A}$ (the *agenda*), and $\mathcal{O}$ (the *restrictions*), in which we sign all the information we obtained up to now.

- ▷ $\mathcal{A}$ contains triples: $\mathcal{A} = \{a_j\} = \{(i_j, c_j, t_j)\}$ where $i_j \in \mathbb{N}$ are labels $(j < \ell \Rightarrow i_j < i_\ell)$ and c_j, t_j individuate the "actual" $lc(S_{i_j}), lt(S_{i_j})$, respectively (see below). We say a_j has label i_j.

▷ $\mathcal{O}$ contains triples: $\mathcal{O} = \{(o_j, \mathcal{V}_j, \sigma_j)\}$, where $o_j = \sum_{i,\alpha+\beta=\rho_j} K_i^\alpha F_i^\beta$ are o.f. expressions, $\mathcal{V}_j$ are the corresponding (V_1) equations and σ_j the found values for the K variables.

What does "actual" mean ? The idea is the following one. We may have obtained from precedent computations that, for a specific critical point, the head of the reduced S-polynomial r could effectively be set to zero. Because of numerical errors, however, the cancellations that should have taken place were not exact, and we obtained again the leading dangerous coefficient. But we know that it *must* be zero, because the actual F values were set such that it should. The same may happen for other monomials beyond the head. The "actual" head is the first monomial m, starting from the leading one, such that the answer to the corresponding $\mathcal{P}_\rho$ problem was not "$c_j = 0$" (that is, either it's $c_j \neq 0$ or $\mathcal{P}_\rho$ was still not solved). In both cases we record in the corresponding entry of the agenda the coefficient and term of the actual head. If all r coefficients can be set to zero at the same time, we use the default monomial $m_0 = 0 \cdot \mathbf{1} \Longrightarrow (c_j, t_j) = (0, \mathbf{1})$, where $\mathbf{1}$ is the constant term.

Let's see how $\mathcal{A}$,$\mathcal{O}$ are updated. At the beginning, they are both empty. When we find a dangerous polynomial, let i be the current label and $r = S_i$. Look in $\mathcal{A}$:

1. If no $a \in \mathcal{A}$ has label i (in particular, if $\mathcal{A} = \emptyset$), set $r' = r, c' = lc(r)$, go to **6**.
2. If $\mathcal{A} \ni a = (i, 0, \mathbf{1})$, set $r = 0$ and continue the algorithm with no updating.
3. Otherwise we must have $\mathcal{A} \ni a = (i, c, t)$ with $c \neq 0$, then remove from r all the monomials $m_i = c_i \cdot t_i$ with $t_i > t$, obtaining r'.
4. If c is not dangerous (it's an already discussed critical point), substitute with c the coefficient related to the leading term t of r' and continue with the algorithm.
5. (c is dangerous, that is, we must still decide) Let $c' = lc(r')$. If c' is not dangerous, update a with (i, c', t) and continue with the algorithm.
6. If c' is dangerous, set up and solve a $\mathcal{P}_\rho$–like problem (see details below). Let $\pi \in F$-box be the minimum point of the o.f. $c_\pi = \text{o.f.}(\pi)$.
6a. If $c_\pi = 0$ (the coefficient can be set to zero), do
⋆ update: set (or add to $\mathcal{A}$, if not present) $a = (i, \tilde{c}, \tilde{t})$, where $(\tilde{c}, \tilde{t})$ are respectively the coefficient and term in the successive monomial of r', or $(i, 0, \mathbf{1})$ if there are none left. Add $(o, \mathcal{V}_i, \sigma)$ to $\mathcal{O}$, where o is the o.f. equation and $\mathcal{V}_i, {}^iK$ its related data, as explained above.
⋆ set initial values (m_S, m_L) for $\mathcal{F}$ coefficients as π values, obtaining $\mathcal{F}'$.
⋆ clear all data structures except $\mathcal{A}$, $\mathcal{O}$: restart all the computation from $\mathcal{F}'$.
6b. If $c_\pi \neq 0$ (the coefficient is surely different from zero), we must refine c' interval $I' = [c'_i, c'_s]$:
⋆ if $c_\pi > 0$ set $c'_i = c_\pi$, otherwise set $c'_s = c_\pi$. Let c'' be the result, with $I'' = I_{c''}$
⋆ update (or add to $\mathcal{A}$) $a = (i, c'', t)$, substitute with c'' the coefficient related to the leading term t of r' and continue with the algorithm.

We now specify the details for the $\mathcal{P}_\rho$-like problem in point **6**. If $\mathcal{O} = \emptyset$, we have the $\mathcal{P}_\rho$ problem introduced in section 4. If $\mathcal{O} \neq \emptyset$ we must take into account all the precedent added "$= 0$" conditions. Unfortunately, simply adding the corresponding precedent o.f. equations in (F, K) to the set of restrictions V_1 is generally not possible, because, given a point π in the F-box, we are not sure if it satisfies these added conditions. It would only if it lies on the variety associated to the set of polynomials expressing the dangerous coefficients in terms of the initial ones (this is the power of symbolic approach, that we would explicitly obtain these relations!). With the numerical approach it is practically impossible to consider exclusively points on this variety inside the F-box.

We propose a possible workaround: we consider a $\mathcal{P}_\rho$ problem with a modified o.f., that considers all the precedent obtained conditions ($\mathcal{O}$ entries). Let $\omega =^{\#} \mathcal{O}$: a modified o.f. may be

$$O = \sum_{i=1}^{\omega} o_i^2 + |o| \tag{4}$$

We squared the o_i to let O being differentiable (apart from the absolute value, difficulty solvable with a simple case distinction). Similar functions with analogous behaviour may be equivalently used. In this way, if a point π is found such that the sum is zero, then it satisfies *all* the restrictions, and therefore it is admissible. If the sum is not zero, then if the last term $|o_\pi|$ has a positive value v, then the coefficient under study is different from zero, and we can use v as new interval (first or second) extreme to exclude 0 from it. If $|o_\pi|$ is zero another π must be tested.

We again underline that after the first $\mathcal{P}_\rho$-like problem (really, a $\mathcal{P}_\rho$ problem), we cannot guarantee that V_1 equations have a solution inside the (F, K) box, because of new restrictions overlapping. This case must still be deeply investigated by the authors, but we underline that there is the possibility that some o.f. cannot be computed, and therefore the above procedure does not apply. In this case some other kind of ZT must be defined by the user.

7 Examples

We present here some preliminary partial results concerning easy system examples, to show what may happen in practical cases. For simplicity, we indicate only the floating point values of the coefficients. Intervals are determined as explained in section 3.

For each example, we use e.g. the [`M`(11,96,256,10,3,0,27) DRL] notation to indicate the use of `PL_MCoeffs` with 7 parameters – of which the first one (a prime number) is not used now – and DRL for the degree reverse lexicographic term ordering (L for lexicographic).

$\boxed{\boxed{1}}$ [M(32003,128,256,10,3,0,3) DRL] $E_1 = \begin{cases} z - 1000 & = 0 \\ x^2y + zx + x & = 0 \\ xy^2 + zy & = 0 \end{cases}$

In this simple example we have that the S-poly $s = S(x^2y+zx+x, xy^2+zy) = xy$. So where is the problem ? The algorithm added and subtracted $xy(z-1000)$, and therefore the head coefficient *seems* to be zero. The resulting system is:

$$O = (F_{1,2})K_{1,0} \quad ; \quad V_1 = \begin{cases} K_{1,0} + K_{2,0} & = 0 \\ (F_{1,1})K_{1,0} + (F_{2,1})K_{2,0} & = 0 \end{cases}$$

From this we can see that the objective function can not be zero (indeed, the interval for $K_{1,0}$ does not contain zero). Simplifying it, we obtain the relation $F_{1,1} - F_{2,1} = 0$ (which was quite easy to detect looking at the initial system). In this case, we have a critical point which is really a "false alarm", being due only to useless computations. Doing the same calculation using syzygies eliminates the uncertainty.

$\boxed{\boxed{2}}$ [M(32003,128,256,10,3,0,3) DRL] $\quad E_2 = \begin{cases} z - 10 & = 0 \\ z^5 + 20x^2y + 21xy & = 0 \\ z^5 + 21xy^2 + 20y^2 & = 0 \end{cases}$

Buchberger algorithm simplifies z^5 to $10z^4$ and so on till we obtain 10^5. This process gives us all the $K_{0,j}$ variables, which are completely independent from computation of critical heads. The first S-polynomial is

$$s = S(20x^2y + 21xy + 10^5, 21xy^2 + 20y^2 + 10^5) = (20 \cdot 20 - 21 \cdot 21)xy^2 + \ldots$$

Since these values are really intervals, it is possible to have that the obtained head coefficient equals zero. This is not the relation we see as an objective function, because the algorithm does continue, simplifying xy^2. That's why we have to study a function of the form $F \cdot K$, where K is dangerous.

More precisely, we have that for 6^{th} critical pair we find a dangerous situation:

$$O = (F_{2,2})K_{2,1}$$

$$V_1 = \begin{cases} K_{0,0} + K_{2,0} = 0 \ (1) & (F_{0,1})K_{0,2} + K_{0,5} = 0 \ \ (7) \\ K_{0,1} + K_{1,0} = 0 \ (2) & (F_{0,1})K_{0,3} + K_{0,6} = 0 \ \ (8) \\ (F_{0,1})K_{0,0} + K_{0,2} = 0 \ (3) & (F_{0,1})K_{0,4} + K_{0,7} = 0 \ \ (9) \\ (F_{0,1})K_{0,1} + K_{0,3} = 0 \ (4) & (F_{0,1})K_{0,5} + K_{0,8} = 0 \ (10) \\ K_{0,4} + K_{2,1} = 0 \ (5) & (F_{0,1})K_{0,6} + K_{0,9} = 0 \ (11) \\ (F_{1,1})K_{1,0} + (F_{2,1})K_{2,0} = 0 \ (6) & (F_{0,1})K_{0,7} + K_{0,10} = 0 \ (12) \\ (F_{1,2})K_{1,0} + (F_{2,2})K_{2,0} + (F_{2,1})K_{2,1} & = 0 \ (13) \end{cases}$$

We show here in some detail the effect of the preprocessing: to do this we numbered V_1 equations for clarity. Equations (1), (2) and (5) are mute. $K_{0,8}$ is single: after having removed (10), $K_{0,5}$ becomes single, and (7) is removed, too. Continuing in this way we see that the variables (and equations in which they appear) we can avoid to consider are

$$\{ \ K_{0,8}, \ K_{0,5}, \ K_{0,2}, \ K_{0,9}, \ K_{0,6}, \ K_{0,3}, \ K_{0,10}, \ K_{0,7} \ \}$$

Performing all the simplifications, the new "initial" system is surprisingly simple

$$V_1 = \begin{cases} (F_{1,1})K_{1,0} + (F_{2,1})K_{2,0} & = 0 \\ (F_{1,2})K_{1,0} + (F_{2,2})K_{2,0} + (F_{2,1})K_{2,1} & = 0 \end{cases}$$

from the computation, we have that $K_{1,0}$ and $K_{2,0}$ are not dangerous, while $K_{2,1}$ is. We then change ordering, with $K_{2,1}$ as greatest variable. We finally have

$$\overline{\mathcal{M}}_{\mathcal{F}} \cdot K = \left(\begin{array}{cc|c} F_{1,1}F_{2,1} & 0 & F_{1,1}F_{2,2} - F_{1,2}F_{2,1} \\ 0 & F_{1,1} & F_{2,1} \end{array}\right) \begin{pmatrix} K_{2,1} \\ K_{1,0} \\ K_{2,0} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

and the o.f. O becomes $O = \dfrac{N(F)}{D(F)}\ K_{2,0} = \dfrac{F_{1,1}F_{2,2} - F_{1,2}F_{2,1}}{F_{1,1}F_{2,1}}\ K_{2,0}$.

As expected, $N(F)$ represents a determinant. We translated the uncertainty problem from K space ($K_{2,1}$ is dangerous) to a convex 4-dimensional F subspace.

$\boxed{\mathbf{3}}$ [M(32003,64,128,10,3,0,22) L] Windsteiger's system: the exact version is

$$E_5 = \begin{cases} -4 + 3\left(\dfrac{172966043}{174178537}x - \dfrac{42176556}{358072327}y\right)^2 + \left(\dfrac{1}{3} + \dfrac{42176556}{358072327}x + \dfrac{172966043}{174178537}y\right)^2 = 0 \\ -4 + \left(\dfrac{1}{3} - \dfrac{42176556}{358072327}y + \dfrac{172966043}{174178537}x\right)^2 + 4\left(\dfrac{172966043}{174178537}y + \dfrac{42176556}{358072327}x\right)^2 = 0 \end{cases}$$

and the approximated one (which we use) is

$$E_5' = \begin{cases} 10277480y^2 - 4678710xy + 29722520x^2 + 6620260y + 785252x - 38888890 = 0 \\ 39583780y^2 + 7018070xy + 10416220x^2 - 785252y + 6620260x - 38888890 = 0 \end{cases}$$

We report the obtained condition (partially factorized) and after substituting the exact values for $F_{i,j}$: we see that

$$\begin{aligned} O = {} & (F_{1,5} - F_{0,5})(F_{0,1} - F_{1,1})^2 + (F_{0,4} - F_{1,4})(F_{0,1} - F_{1,1})(F_{0,2} - F_{1,2}) + \\ & (F_{1,3} - F_{0,3})(F_{0,2} - F_{1,2})^2 \simeq 3.4125131480145708084595190432555 \cdot 10^{-16} \end{aligned}$$

A very small value which justifies the point to be critical.

8 Conclusions

In this paper we presented a possible approach to zero testing in numerical Gröbner bases computations with not exact initial coefficients. The combined use of the ad–hoc introduced *multi-component* coefficients and of syzygies permitted to obtain equations to be fulfilled by the coefficients of the initial polynomials and syzygies. If, within interval tolerances, the doubtful coefficient can be zero, a new equation/restriction must be taken care of, otherwise intervals can be refined and computations proceed.

Both symbolic and numeric analysis are possible, the former slow but giving more information, the latter computationally faster but giving only punctual results. A deeper numerical analysis behaviour on real–life examples is planned.

The addition of a modular part to multi-component coefficients might also be used to apply the Gröbner trace algorithm proposed in [17]. This could lead to further improvements for guided floating point computations/analysis.

References

1. Adams, W. W., Loustaunau, P.: *An Introduction to Gröbner bases*, Graduate Studies in Mathematics, Volume 3, AMS (1994)
2. Alefeld, G., Herzberger, J.: *Introduction to Interval Computations*, Academic Press, New York (1983)
3. Bodrato, M., Zanoni, A.: *Numerical Gröbner bases and syzygies: an interval approach*. Proceedings of the 6^{th} SYNASC Symposium. T. Jebelean, V. Negru, D. Petcu, D. Zaharie ed. Mirton, Timisoara, Romania, (2004) 77 - 89
4. Bonini, C., Nischke, K.–P., Traverso, C.: *Computing Gröbner bases numerically: some experiments*, Proceedings SIMAI (1998)
5. Buchberger, B.: *Introduction to Gröbner Bases* Gröbner Bases and Applications (B. Buchberger, F. Winkler eds.), London Mathematical Society Lecture Notes Series 251, Cambridge University Press, (1998) 3–31
6. Becker, T., Weispfenning, V.: *Gröbner Bases: A Computational Approach to Commutative Algebra*, Graduate Studies in Mathematics, Volume 141, Springer Verlag, (1993) (second edition, 1998).
7. Caboara, M., Traverso, C.: *Efficient Algorithms for ideal operations*, Proceedings ISSAC 98, ACM Press (1998) 147–152
8. Cox, D., Little, J., O'Shea, D.: *Ideals, Varieties, and Algorithms*, Springer–Verlag, (1991) (second corrected edition, 1998).
9. Cox, D., Little, J., O'Shea, D.: *Using algebraic geometry*, Springer-Verlag (1998)
10. FRISCO: *A Framework for Integrated Symbolic/Numeric Computation*, ESPRIT Project LTR 21024, European Union (1996–1999)
11. FRISCO test suite: `http://www.inria.fr/saga/POL/`
12. Migheli, L.: *Basi di Gröbner e aritmetiche approssimate*, Tesi di Laurea, Università di Pisa (in italian) (1999)
13. Shirayanagi, K.: *Floating Point Gröbner Bases*, Journal of Mathematics and Computers in Simulation 42, (1996) 509–528
14. Stetter, H.J.: *Stabilization of polynomial system solving with Gröbner bases*, Proceedings ISSAC (1997) 117–124
15. Stetter, H. J.: *Numerical Polynomial Algebra* SIAM, (2004)
16. Traverso, C.: *Syzygies, and the stabilization of numerical buchberger algorithm* Proceedings LMCS, RISC-Linz (2002), 244–255
17. Traverso, C.: *Gröbner trace algorithms* Proceedings ISSAC 88, LNCS 358, Springer Verlag, (1988) 125–138
18. Traverso, C., Zanoni, A.: *Numerical Stability and Stabilization of Groebner Basis Computation*, Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation, Université de Lille, France, ACM Press, Teo Mora editor, (2002) 262–269
19. Weispfenning, V.: *Gröbner Bases for Inexact Input Data* Computer Algebra in Scientific Computation - CASC 2003, Passau, TUM, (2003) 403–412
20. Zanoni, A.: *Numerical stability in Gröbner bases computation*, Proceedings of the 8^{th} Rhine Workshop on Computer Algebra. H. Kredel, W. K. Seidler, ed. (2002) 207–216
21. Zanoni, A.: *Numerical Gröbner bases*, PhD thesis, Università di Firenze, Italy (2003)

THIS PAGE IS NOT PART OF THE ARTICLE[1]

BibTeX entry

```
@InProceedings{Bodrato:CASC2006,
  author =   {Marco Bodrato and Alberto Zanoni},
  title =   {Intervals, Syzygies, Numerical {Gr\"obner} Bases : A Mixed Study},
  crossref =   {CASC2006},
  pages =   {64--76},

  booktitle =   {{CASC} 2006 Proceedings},
  year =   {2006},
  editor =   {Victor G. Ganza and Ernst W. Mayer and Evgenii V. Vorozhtsov},
  volume =   {4194},
  series =   {LNCS},
  month =   {September},
  publisher =   {Springer}
}

@Proceedings{CASC2006,
  title =   {{C}omputer {A}lgebra in {S}cientific {C}omputing,
  9th International Workshop, {CASC2006}, {C}hi{\,s}inau, {M}oldova,
  {S}eptember 2006, Proceedings},
  location =     {Chi\,sinau, Moldova},

  booktitle =   {{CASC} 2006 Proceedings},
  year =   {2006},
  editor =   {Victor G. Ganza and Ernst W. Mayer and Evgenii V. Vorozhtsov},
  volume =   {4194},
  series =   {Lecture Notes in Computer Science},
  month =   {September},
  publisher =   {Springer}
}
```

[1] Paper edited with Emacs-21 and LaTeX-3.0 on a Debian GNU/Linux box.