How to solve multivariate polynomial systems with MSOLVE?

Contents

1 Introduction

MSOLVE is a C library for solving multivariate polynomial systems of equations. It relies on computer algebra, a.k.a. symbolic computation, algorithms to compute *algebraic* representations of the solution set from which many, if not all, informations can be extracted.

Solving polynomial systems with MSOLVE is *global* by contrast to *local* numerical routines. The use of computer algebra methods allow also the user to bypass classical numerical issues encountered by numerical methods for polynomial system solving such as those based on numerical homotopy continuation or semi-definite programming.

MSOLVE relies mainly on Gröbner bases algorithms (see below for a some basic definitions and properties). It is highly optimized, uses `AVX2` vectorization instructions and multi-threading.

It uses the `GMP` library (handling multi-precision integers) and the `FLINT` library (handling arithmetics of univariate polynomials).

MSOLVE can be downloaded from

`https://msolve.lip6.fr`

where binaries (for `x86` processors runing Linux operating systems) and source files are provided.

MSOLVE is designed for 64 bit architectures, with `AVX2` instructions.

MSOLVE allows you to:

- isolate all real solutions to polynomial systems with rational coefficients and finitely many complex solutions;
- compute Gröbner bases of polynomial systems with coefficients which are either rational numbers or in a prime field $\mathbb{Z}/p\mathbb{Z}$ with $p < 2^{31}$;
- compute parametrizations of the solutions of polynomial systems with coefficients which are either rational numbers or in a prime field $\mathbb{Z}/p\mathbb{Z}$ with $p < 2^{31}$ (assuming that the system has finitely many solutions with coordinates in an algebraic closure of the field generated by the input coefficients).

MSOLVE is based on Gröbner bases computations. When launching MSOLVE on an input polynomial system (see the file format in section 2), a Gröbner basis computation starts and allows MSOLVE to determine if the number of solutions to the system is infinite or finite in an algebraic closure of the base field (the complex numbers when the input coefficients are rational numbers).

When the number of solutions is finite, one says that the system (or the ideal generated by the input equations) has dimension zero at most. Else it has positive dimension. Section 3 shows how MSOLVE behaves when the input system has positive dimension or when there is no solution at all in an algebraic closure of the base field (over the complex numbers when the input coefficients are rational numbers).

When the system has dimension at most zero, MSOLVE can compute the real solutions or, as said above, compute a Gröbner basis (when the base field is a prime field) or compute a parametrization of the solutions. Section 4 shows how to use MSOLVE for solving polynomial systems over the reals when they have dimension at most zero. Section 5 shows how to use MSOLVE for computing Gröbner bases over prime fields (with some restriction on the bit size of the considered prime). Section 6 shows how to use MSOLVE for computing rational parametrizations of solutions to polynomial systems which have dimension at most zero. Finally, section 8 summarizes some options which can be used rational parametrizations of solutions to polynomial systems which have dimension at most zero. Finally, section 8 summarizes some options which can be used.

The MSOLVE library is described in [1] with implementation details on the algorithms used therein. All computations performed over the rational numbers (e.g. for computing real roots) are based on multi-modular computations with a probabilistic stopping criterion. Unless explicitely requested by the user (see the `-l` flag in section 8), all computations of Gröbner bases in prime fields use deterministic algorithms. Change of order algorithms which are used are deterministic when the input ideal is radical.

2 Input file format

MSOLVE allows you to solve polynomial systems either with coefficients which are either rational numbers or in a prime field $\mathbb{Z}/p\mathbb{Z}$ with $p < 2^{31}$. If you aim at solving polynomial systems with coefficients which are floating point numbers, you can just replace these floating point numbers with rational numbers. Further, we explain how the input files of MSOLVE should be.

Consider the following polynomial system of equations

$$\begin{array}{rcl} x + 2y + 2z - 1 & = & 0 \\ x^2 + 2y^2 + 2z^2 - x & = & 0 \\ 2xy + 2yz - y & = & 0 \end{array}$$

in $\mathbb{Q}[x, y, z]$.

In order to solve it with MSOLVE one simply produces a file with the following content

```
x,y,z
0
x+2*y+2*z-1,
x^2+2*y^2+2*z^2-x,
2*x*y+2*y*z-y
```

Hence the structure of input files to MSOLVE is as follows:

1. the first line contains the variables of the input system, separated with a comma (no comma at end of line);
2. the second line contains the characteristic of the field over which computations are performed;
3. the next lines contain polynomials, in expanded form, separated by a comma and with a line break (no comma or line break for the last one).

In each given polynomial, MSOLVE expects a single occurrence of each monomial; if some monomial appears several times (e.g. as in `x+2*y+2*z-x`), the behaviour of MSOLVE's parser is undefined.

When one wants to solve this system over $\frac{\mathbb{Z}}{65521\mathbb{Z}}$ one just replaces 0 by 65521 in the second line. Note that in the positive characteristic case the coefficients used should be smaller or equal to $2^{31} - 1$.

```
x,y,z
65521
x+2*y+2*z-1,
x^2+2*y^2+2*z^2-x,
2*x*y+2*y*z-y
```

3 Computing the dimension

To make things explicit on the behaviour of MSOLVE when the input system does not have finitely many complex solutions, let us consider first the example below.

```
x, y
0
x*y-1,
x
```

Then, MSOLVE outputs

```
[-1]:
```

indicating that the dimension of the set of complex solutions is -1, hence it is empty.

If now, one considers the following example.

```
x, y, z
0
x^2-y^2,
x-y
```

Then, MSOLVE outputs

```
[1, 3, -1, []]:
```

The first integer 1 indicates that the complex solution is positive dimensional (note that the actual dimension of the complex solution set is 2).

4 Solving over the reals (finitely many solutions)

The basic functionality MSOLVE allows you to perform is real root isolation for polynomial systems with rational coefficients and *with finitely many complex solutions*. This latter requirement is automatically tested by MSOLVE.

For instance, consider the following input to MSOLVE written in a file `in.ms`.

```
x,y,z
0
x+2*y+2*z-1,
x^2+2*y^2+2*z^2-x,
2*x*y+2*y*z-y
```

Then, typing the following command line

```
    ./msolve -f in.ms -o out.ms
```

will display in the `out.ms` file the following content.

```
[0, [1,
[[[107291359935630315248585097660753910587 / 2^127, 268228399839075788121462744151884776477 / 2^125], [107291359935630315248585097660753910587 /
         2^128, 268228399839075788121462744151884776477 / 2^126], [-355532291286331190123863132844989723573 / 2^131,
        -142212916514532476049545253137995889429 1 / 2^133]], [[1, 1], [0, 0], [0, 0]], [[385439401733433119500040198100038943111 / 2^127,
        770878803466866239000080396200077886667 / 2^128], [963598504333582798750100495250097357 9 / 2^126, 481799252166791399375050247625048679 1
        / 2^125], [930533031137826078316792640958763170833 / 2^128, 372213212455130431326717056383505268333 / 2^130]],
        [[708921597751955132215363765482850440 5 / 2^124, 113427455640312821154458202477256070491 / 2^128], [-1 / 2^127, 1 / 2^127],
        [181483929024500513847133123963609712776 5 / 2^132, 907419645122502569235665619818048563883 / 2^131]]]
]]:
```

This is a list whose first element is the integer 0 indicating that the input polynomial system has finitely many complex solutions. The second element of this list is a list which provides the coordinates of the real solutions as follows:

- the first element is an integer ℓ indicating how many lists are given further; in the above example, the integer 1 tells that we have a single list (this will be the usual case);
- the next are ℓ lists $L_1, \ldots, L_\ell$ which encode the solutions to the input system; each of them containing boxes isolating a single real solution.

For instance, from the above output, we deduce that the box defined by

$$\begin{cases} \frac{107291359935630315248585097660753910587}{2^{127}} \leqslant x \leqslant \frac{268228399839075788121462744151884776477}{2^{125}}, \\ \frac{107291359935630315248585097660753910587}{2^{128}} \leqslant y \leqslant \frac{268228399839075788121462744151884776477}{2^{126}}, \\ \frac{-355532291286331190123863132844989723573}{2^{131}} \leqslant z \leqslant \frac{-142212916514532476049545253137995889429 1}{2^{133}} \end{cases}$$

contains a single real solution to the input system.

Sometimes, it makes sense to increase the precision. To do that, we use the `-p` flag, followed by an integer monitoring the used precision, as follows.

```
    ./msolve -p 256 -f in.ms -o out.ms
```

We obtain in `out.ms` the following

```
[0, [1,
[[[36509357909062631536129668436573070012487487067100583303001175658946635606055 / 2^255,
        4563669738632828942016208554571633751560935883387572912875146957368329450757 / 2^252],
        [36509357909062631536129668436573070012487487067100583303001175658946635606055 / 2^256,
        4563669738632828942016208554571633751560935883387572912875146957368329450757 / 2^253],
        [-60490684797868661441895377475208744393359927205523538345094237255746825568573 / 2^258,
        -483925478382949291535163019801669955146879417644188306760753898045974604548583 / 2^261]], [[1, 1], [0, 0], [0, 0]],
        [[3278955798161077339921616998930436909728483733114914607048732943254033559905 / 2^253,
        26231646385288618719372935991443495277827869864919316856389863546032268479285 / 2^256],
        [6557911596322154679843233997860873819456967466229829214097465886508067119813 / 2^255,
        13115823192644309359686467995721747638913934932459658428194931773016134239637 / 2^256],
        [63328796466738957984825113025800917297614244935801930326677856915848592681411 / 2^257,
        126657592933477915969650226051601834595228489871603860653355713831697185362823 / 2^258]],
        [[2412335192444087404657728854347664746943124680534178417488699666831523534165 / 2^252,
        38597363079105398474523661669562635951089994888546854679819194669304376546651 / 2^256], [-1 / 2^255, 1 / 2^255],
        [617557809265686375592378586713002175217439918216749674877107114708870024746325 / 2^260,
        308778904632843187796189293356501087608719959108374837438553557354435012373163 / 2^259]]]
]]:
```

5 Computing Gröbner bases

MSOLVE relies on Gröbner bases algorithms which allow one to rewrite the input polynomial system as an equivalent system which reveals properties of the solution set (dimension, degree) and to compute “modulo” the input equations.

MSOLVE provides Gröbner bases computations for the so-called *grevlex* ordering (see e.g. [2]) when the coefficients either lie in the field of rational numbers or when they lie in the prime field case. For instance, assume the file `in.ms` contains the following:

```
z1, z2, z3
1073741827
7*z1*z2+5*z2*z3+z3^2+z1+5*z3+10,
7*z3^2-27*z1^2-15*z2^2+59*z3+3*z1,
8*z1^2+13*z1*z3+10*z3^2+z2+z1
```

Now, typing the following command:

```
    ./msolve -g 2 -f in.ms -o out.ms
```

where the `-g` flag indicates that one aims at computing a Gröbner basis. The value `2` tells MSOLVE to compute the Gröbner basis for the *grevlex order* with $z_1 > z_2 > z_3$. The computed Gröbner basis is then printed in the file `out.ms` as follows.

```
#Reduced Groebner basis data
#---
#field characteristic: 1073741827
#variable order: z1, z2, z3
#monomial order: graded reverse lexicographical
#length of basis: 6 elements sorted by increasing leading monomials
#---
[1*z2^2+832149913*z1^1*z3^1+876889156*z3^2+295279002*z1^1+724775733*z2^1+143165573*z3^1,
1*z1^1*z2^1+613566759*z2^1*z3^1+766958448*z3^2+766958448*z1^1+613566759*z3^1+153391691,
1*z1^2+134217730*z1^1*z3^1+268435458*z3^2+671088642*z1^1+671088642*z2^1,
1*z2^1*z3^2+722232944*z3^3+180778379*z1^1*z3^1+1027531442*z2^1*z3^1+173735741*z3^2+936498976*z1^1+702034498*z2^1+921316952*z3^1+59915395,
1*z1^1*z3^2+557357968*z3^3+911535897*z1^1*z3^1+419648179*z2^1*z3^1+96648475*z3^2+698659259*z1^1+282295066*z2^1+885328953*z3^1+769127629,
1*z3^4+250491376*z3^3+716275774*z1^1*z3^1+91652836*z2^1*z3^1+88303466*z3^2+855797860*z1^1+18642214*z2^1+728901227*z3^1+969918485]:
```

When one is interested only in the leading monomials of the Gröbner basis (which is a way smaller output), one simply uses the `-g 1` flag as follows

```
    ./msolve -g 1 -f in.ms -o out.ms
```

and we obtain:

```
#Leading ideal data
#---
#field characteristic: 1073741827
#variable order: z1, z2, z3
#monomial order: graded reverse lexicographical
#length of basis: 6 elements sorted by increasing leading monomials
#---
[z2^2,
z1^1*z2^1,
z1^2,
z2^1*z3^2,
z1^1*z3^2,
z3^4]:
```

Note that from this list of monomials, one can deduce the Hilbert series of the ideal generated by the input equations and then its dimension and its degree (see [2]).

Note that when the `-g 1` flag is used in characteristic 0, the result is always probabilistic: the leading monomials are deduced from two computations over a prime field, the first prime being chosen randomly.

For instance, in the above example, one can deduce that the ideal has dimension 0 (finitely many solutions with coordinates in an algebraic closure of $\frac{\mathbb{Z}}{1073741827\mathbb{Z}}$) since the basis of leading monomials contain pure powers of all variables. The degree of the ideal is also 8 since there are 8 monomials in z_1, z_2, z_3 which are not divisible by the above leading monomials.

MSOLVE also allows you to perform Gröbner bases computations using *one-block elimination monomial order* thanks to the `-e` flag. The following command

```
    ./msolve -e 1 -g 2 -f in.ms -o out.ms
```

on

```
t, w, x, y, z
1073741827
w^4,
x^4,
w*y^3+1073741826*x*z^3,
t*z+1073741826
```

will perform the Gröbner basis computation eliminating the first variable. The output is

```
#Reduced Groebner basis data
#---
#field characteristic: 1073741827
#variable order: t, w, x, y, z
#monomial order: eliminating first variable, blocks: graded reverse lexicographical
#length of basis: 7 elements sorted by increasing leading monomials
#---
[1*w^1*y^3+1073741826*x^1*z^3,
1*x^4,
1*w^1*x^3,
1*w^2*x^2,
1*w^3*x^1,
1*w^4,
1*t^1*z^1+1073741826]:
```

where we see that the first 6 polynomials are only in w, x, y, z, which corresponds to the elimination of the variable t. When the input coefficients lie in the field of rational numbers (hence, characteristic 0), the returned Gröbner basis is the one of the *elimination ideal*, i.e. they have partial degree 0 in the variables to eliminate.

More generally, using `-e k` will eliminate the k first variables. Thus

```
    ./msolve -e 2 -g 2 -f in.ms -o out.ms
```

will eliminate t and w, yielding

```
#Reduced Groebner basis data
#---
#field characteristic: 1073741827
#variable order: t, w, x, y, z
#monomial order: eliminating first 2 variables, blocks: graded reverse lexicographical
#length of basis: 7 elements sorted by increasing leading monomials
#---
[1*x^4,
1*w^1*y^3+1073741826*x^1*z^3,
1*w^1*x^3,
1*t^1*z^1+1073741826,
1*w^2*x^2,
1*w^3*x^1,
1*w^4]:
#Reduced Groebner basis for input in characteristic 1073741827
#for variable order t, w, x, y, z
#w.r.t. grevlex monomial ordering
#consisting of 7 elements:
[1*x^4,
1*w^1*y^3+1073741826*x^1*z^3,
1*w^1*x^3,
1*t^1*z^1+1073741826,
1*w^2*x^2,
1*w^3*x^1,
1*w^4]:
```

where we see that only the first polynomial is not in t and w.

6 Parametrizations of (finitely many) solutions

Assume that the input polynomials have coefficients in some field $\mathbb{K}$ with variables $x_1, \dots, x_n$. In the case of polynomial systems of dimension 0, MSOLVE computes by default a zero-dimensional parametrization of the solution set. The user can obtain such an encoding using the `-P` flag (see below).

Let us recall what a rational parametrization is. This is a couple $(\mathscr{P}, \ell)$ where ℓ is a linear form $\lambda_1 x_1 + \cdots + \lambda_n x_n$ with $\lambda_i \in \mathbb{K}$ (for $1 \leqslant i \leqslant n$), $\mathscr{P}$ is a sequence of polynomials $(w, w', v_1, \dots, v_n)$ in $\mathbb{K}[t]$ where t is a new variable such that:

- when $\mathbb{K}$ is a prime field, $w' = 1$ else $w' = \frac{\partial w}{\partial t}$;
- $\deg(v_i) < \deg(w)$ for $1 \leqslant i \leqslant n$;
- $\lambda_1 v_1 + \cdots + \lambda_n v_n = tw' \mod w$

and the solution set to the input polynomials coincides with the set:

$$\left\{ \left(-\frac{v_1(\vartheta)}{w'(\vartheta)}, \dots, -\frac{v_n(\vartheta)}{w'(\vartheta)} \right) \,\middle|\, w(\vartheta) = 0 \right\}.$$

In algebraic words, the polynomials $w' x_i + v_i$ belong to the radical of the ideal generated by the input equations and the form $\lambda_1 x_1 + \cdots + \lambda_n x_n + t$.

MSOLVE outputs univariate polynomials as an array `[deg,L]` where `deg` is the degree of the polynomial under consideration and `L` is the array of its coefficients in the monomial basis by increasing degree and `c` is a denominator to all coefficients. For instance, the polynomial $x^2 + 3x - 2$ is encoded by

```
[2, [-2, 3, 1]]
```

We first explain MSOLVE's output in the case where the input coefficients are rational numbers (the characteristic zero case). For an input in the file `in.ms`

```
z1, z2
0
z1^2+z2^2-1,
z1^2-z2^2
```

the command

```
./msolve -P 2 -f in.ms
```

MSOLVE outputs is

```
[0, [0,
3,
4,
['z1', 'z2', 'A'],
[-119/576,69/576,5/576],
[1,
[[4, [883600, 0, -18922, 0, 25]],
[3, [0, -37844, 0, 100]],
[
[[3, [223720, 0, -1190, 0]],
1],
[[3, [129720, 0, 690, 0]],
1]
]]]]:
```

and has the following structure

```
[0, [0, nvars, deg, vars, form, [1,[lw, lwp, param]]]]:
```

where

- the first 0 indicates that the input system has finitely many complex solutions (dimension at most 0);
- the second 0 is the characteristic;
- `nvars` is the number of variables used for the parametrization (it coincides with the number of input variables if the form ℓ is chosen as one of the variables else it is one more);
- `deg` is the number of solutions, *counted with multiplicities* (in other words the degree of the ideal generated by the input equations);

- `vars` is the list of variables following the ordering used for computing the parametrization (hence, with maybe with one more variable than the ones given as input).

  In our example, MSOLVE outputs:

  `['z1', 'z2', 'A']`

  where `A` is a new variable.

- `form` is the list of coefficients for the linear form ℓ when it does not coincide with one of the input variables (else it is an empty list);

  In our example, this is a list of three rational numbers, say `[-119/576,69/576,5/576]`, indicating that the linear form used to compute the rational parametrization is

  `-119/576*z1+69/576*z2+5/576*A`

- the next 1 indicates that a single parametrization is returned next (the one encoded by `[lw, lwp, param]`);

- `lw` is the encoding of the eliminating polynomial w;

- `lwp` is the encoding of the denominator used in the rational parametrization;

- `param` is the list of the output parametrizations, there are $n-1$ where n is the number of elements in `vars` ; they are encoded as follows `[[deg, L], c]` where `c` is an integer which divides the polynomial encoded by `[deg, L]`.

  The first one corresponds to the first variable in `vars`, the second parametrization corresponds to the second variable in `vars` and so on. Hence, the variable which is used to parametrize the solution set is always the last one.

We illustrate now how the output looks like on input file `in.ms`

```
z1, z2, z3
0
z1^2-z2^2+z3^2-4,
z1*z2+2*z2*z3-3*z3*z1-1,
z1+2*z2+3*z3-1
```

Using `./msolve -P 2 -f in.ms` the output is

```
[0, [0,
3,
4,
['z1', 'z2', 'z3'],
[0, 0, 1],
[1,
[[4, [-116, -210, 1484, -344, 53]],
[3, [-210, 2968, -1032, 212]],
[
[[3, [1894, 162, -1636, 192]],
1],
[[3, [-146, -620, -3118, 314]],
1]
]]]]:
```

On this example, all variables are parametrized by the variable `z3`.

The polynomial w is $-116 - 210z_3 + 1484z_3^2 - 344z_3^3 + 53z_3^4$. The polynomials v_1 and v_2 are respectively

$$v_1 = 1894 + 162z_3 - 1636z_3^2 + 192z_3^3 \quad \text{and} \quad v_2 = -146 - 620z_3 - 3118z_3^2 + 314z_3^3.$$

Note that we can get both the parametrization and the real roots. For instance, using the command `./msolve -P 1 -f in.ms`, one obtains

```
[0, [0,
3,
4,
['z1', 'z2', 'z3'],
[0, 0, 1],
[1,
[[4, [-116, -210, 1484, -344, 53]],
[3, [-210, 2968, -1032, 212]],
[
[[3, [1894, 162, -1636, 192]],
1],
[[3, [-146, -620, -3118, 314]],
1]
]]]],[1,
[[[6793756736462737050275302853303317150009 / 2^128, 3396878368231368525137651426651658575O5 / 2^127], [-60535166785954698124918883091179878673
         / 2^128, -3783447924122168632807430193198742417 / 2^124], [-1162789190151604508343028862486419979779 / 2^132,
        -5813945950758022541715144312432099898889 / 2^131]], [[-7566653066601039099672965716297915041 37 / 2^128,
        -756665306660103909967296571629791504135 / 2^128], [88745898258177294078528671940999802399 / 2^126,
        177491796516354588157057343881999604799 / 2^127], [2063895933416661444279689618845660247455 / 2^132,
        4127791866833322888559379237691320494911 / 2^133]]]
]]:
```

We end this section with the same example as above but seeing the coefficients in $\mathbb{Z}/65521\mathbb{Z}$.

```
z1, z2, z3
65521
z1^2-z2^2+z3^2-4,
z1*z2+2*z2*z3-3*z3*z1-1,
z1+2*z2+3*z3-1
```

The call `./msolve -P 2 -f in.ms` then outputs

```
[0, [65521,
3,
4,
['z1', 'z2', 'z3'],
[0, 0, 1],
[1,
[[4,
[16069, 9886, 28, 2466, 1]],
[0,
[1]],
[
[[3,
[6276, 37054, 57744, 4959]]],
[[3,
[29622, 14235, 36649, 30281]]]
]]]]]:
```

7 Saturation and colon ideals

MSOLVE also proposes algorithms for computing Gröbner bases of saturation and colon ideals. Given $m + 1$ polynomials $f_1, \ldots, f_m, \varphi$ over a field $\mathbb{K}$ with variables $x_1, \ldots, x_n$, the saturation ideal $\langle f_1, \ldots, f_m \rangle : \langle \varphi \rangle^\infty$ is the ideal of all polynomials h, such that there exists $k \in \mathbb{N}$ such that $h\varphi^k \in \langle f_1, \ldots, f_m \rangle$. The colon ideal $\langle f_1, \ldots, f_m \rangle : \langle \varphi \rangle$ is the ideal of all polynomials h, such that $h\varphi \in \langle f_1, \ldots, f_m \rangle$.

A Gröbner basis for the *grevlex order* can be computed in the former case with an input file containg $f_1, \ldots, f_m, \varphi$ and called with the flag `-S` to use the F4SAT algorithm. Note that this option is at the moment restricted to 32 bit prime fields.

For instance, consider the following input to MSOLVE written in a file `in.ms`.

```
w, x, y, z
1073741827
w^4,
x^4,
1073741826*x*z^3+w*y^3,
z
```

Then, typing the following command line

```
    ./msolve  -S -g 2 -f in.ms -o out.ms
```

will display in the `out.ms` file the following content.

```
#Reduced Groebner basis data
#---
#field characteristic: 1073741827
#variable order: w, x, y, z
#monomial order: graded reverse lexicographical
#length of basis: 6 elements sorted by increasing leading monomials
#---
[1*w^1*y^3+1073741826*x^1*z^3,
1*x^4,
1*w^1*x^3,
1*w^2*x^2,
1*w^3*x^1,
1*w^4]:
```

8 More flags and options

- The flag `-h` displays some documentation
- The flag `-v <int>` controls the verbosity

  `Default value: 0`

- The flag `-t <int>` controls the number of threads used

  `Default value: 1`

- The flag `-p <int>` controls the binary precision of the output of the univariate real root solver (default value may be automatically increased by MSOLVE when needed).

  `Default value:32`

- The flag `-g <int>` tells MSOLVE to output the leading monomial of the ideal generated by the input polynomials (when `<int>` is `1`) or the minimal reduced Gröbner basis (when `<int>` is `2` and a prime characteristic is indicated).

  `Default value:0`

- The flag `-P <int>` tells MSOLVE to output the rational parametrization computed for solving zero-dimensional polynomial systems (those with finitely many solutions in an algebraic closure of the base field). When `+-P 0` is set, such a parametrization is not returned, when `-P 1` is set, the parametrization is returned and, in the charactersitic zero case (rational coefficients), real solutions are returned, when `+-P 2` is set, only the rational parametrization is returned.

  `Default value:0`

- The flag `-c <int>` tells MSOLVE how to handle genericity requirements: when `<int>` is `0` MSOLVE quits when these requirements are not satisfied, when `<int>` is `1` MSOLVE is allowed to change the order of the variables if needed and quits if after these changes, the genericity requirements are not satisfied, when `<int>` is `2` MSOLVE is allowed to introduce a new variable and a linear form until the genericity requirements are satisfied.

`Default value:2`

- The flag `-d <int>` tells MSOLVE how to handle further genericity requirements when the staircase is not generic enough by computing some normal forms: `<int>` can go from `0` (no normal form computations are computed) to `4` (all the normal forms are computed).

`Default value:2`

9 Julia interface to MSOLVE

The Julia interface to MSOLVE is part of the official Julia package `Oscar.jl`. You can install the package via the following commands inside a Julia session:

```
using Pkg
Pkg.add("Oscar")
```

Once the package is loaded via `using Oscar` one can call the function `msolve()` which returns, if any, the rational parametrization and the solutions of the given input system of multivariate polynomials. The most common calling convention is as follows:

```
res = msolve(I).
```

where

- `I` is of type `ideal`.

The most common options for calling `msolve()` in Julia are:

- `info_level` with values `0` (no information printing; default), `1` (slight information printing on comptutational status) or `2` (full information printing also on intermediate steps),
- `la_option` for the linear algebra variant to be chosen inside F4: `2` for exact linear algebra and tracing multi modular computations (default) or `44` for probabilistic linear algebra with independent modular computations;
- `precision` for the bit precision with which the solutions are computed from the rational parametrization. Default is `64`.

So using `msolve()` with probabilistic linear algebra, the most verbose information printout and a precision of 80 one would call

```
res = msolve(I, la_option=44, info_level=2, precision=80).
```

You can get a full list of options for `msolve()` in Julia by typing inside Julia

```
?  msolve()
```

10 Maple interface to MSOLVE

The Maple interface to MSOLVE is a file interface which can be found on the MSOLVE interface page or in the MSOLVE binary package. Having loaded the interface one can call the function `MSolveRealRoots()` in the following way:

```
results = MSolveRealRoots(F, vars)
```

where `F` denotes a polynomial system in variables `vars`,

In order to compute Gröbner bases, you can also the function `MSolveGroebner`.

You may consult the source code for optional arguments which allow you to better control the output format, the names of used files, verbosity, etc.

11 Sage interface to MSOLVE

There is now an interface between SageMath and MSOLVE.

You can have a look at `https://github.com/sagemath/sage/blob/develop/src/sage/rings/polynomial/msolve.py` and `https://github.com/sagemath/sage/blob/develop/src/sage/rings/polynomial/multi_polynomial_ideal.py`

Many thanks to the SageMath development team, in particular to Marc Mezzarobba who initiated this interface.

12 Credits

The main developers of MSOLVE are Jérémy Berthomieu, Christian Eder and Mohab Safey El Din. It relies on original implementations of Faugère's F_4 algorithm [3] as well as Faugère and Mou's Sparse-FGLM algorithm [4]. We are grateful to Huu Phuoc Le and Jorge García Fontán for a preliminary version of the Maple interface as well as Rémi Prébet for a preliminary version of the Sage interface.

If you use MSOLVE, you may cite:

```
@inproceedings{msolve,
  TITLE = {{msolve: A Library for Solving Polynomial Systems}},
  AUTHOR = {Berthomieu, J{\'e}r{\'e}my and Eder, Christian and {Safey El Din}, Mohab},
  URL = {https://hal.sorbonne-universite.fr/hal-03191666},
  BOOKTITLE = {{2021 International Symposium on Symbolic and Algebraic Computation}},
  ADDRESS = {Saint Petersburg, Russia},
  SERIES = {46th International Symposium on Symbolic and Algebraic Computation},
  YEAR = {2021},
  MONTH = Jul,
  DOI = {10.1145/3452143.3465545},
  PDF = {https://hal.sorbonne-universite.fr/hal-03191666v2/file/main.pdf},
  HAL_ID = {hal-03191666},
  HAL_VERSION = {v2},
}
```

References

[1] J. Berthomieu, C. Eder, and M. Safey El Din. “msolve: A Library for Solving Polynomial Systems”. In: *2021 International Symposium on Symbolic and Algebraic Computation*. 46th International Symposium on Symbolic and Algebraic Computation. Saint Petersburg, Russia, July 2021.

[2] D. A. Cox, J. Little, and D. O’Shea. *Ideals, varieties, and algorithms*. Fourth. Undergraduate Texts in Mathematics. An introduction to computational algebraic geometry and commutative algebra. Springer, Cham, 2015, pp. xvi+646.

[3] J.-C. Faugère. “A new efficient algorithm for computing Gröbner bases (F4)”. In: *Journal of Pure and Applied Algebra* 139.1-3 (1999), pp. 61–88.

[4] J.-C. Faugère and C. Mou. “Sparse FGLM algorithms”. In: *Journal of Symbolic Computation* 80 (2017), pp. 538–569.