Normaliz 3.10.0

Winfried Bruns   Max Horn   Ulrich von der Ohe

Former Normaliz 3 team members: Tim Römer, Richard Sieg and Christof Söger

Normaliz 2 team member: Bogdan Ichim

`https://normaliz.uos.de`   `https://github.com/Normaliz`

`mailto:normaliz@uos.de`

`https://hub.docker.com/u/normaliz/dashboard/`

`https://mybinder.org/v2/gh/Normaliz/NormalizJupyter/master`

Short reference: `NmzShortRef.pdf`

Contents

3 Affine monids and binomial ideals by examples

The role of binomials in the computation of affine monoids and their algebras is briefly explained in Sections A.8 and A.9. We assume that the user is familiar with them.

3.0.1 Input and default computation goals

Affine monoids are given to Normaliz by the input type

monoid

as in `monoid.in`:

```
amb_space 3
monoid 6
1 0 0
2 3 5
0 0 1
1 1 2
0 1 3
3 1 0
/* grading 1 -2 1*/
/*HilbertSeries*/
/*GroebnerBasis*/
/*Lex*/
/*MarkovBasis*/
/*gb_degree_bound 11*/
/*gb_min_degree 9*/
/*Multiplicity*/
/*SingularLocus*/
/*CodimSingularLocus*/
/*IsSerreR1*/
```

Positivity of the monoid does *not* mean that all components of the input vectors are nonnegative. It only means that $x = 0$ if both x and $-x$ belong to it.

Let us translate this exmple into multiplicative notation. We have binomials in $K[X_1, X_2, X_3]$, namely

$$M_1 = X_1, \qquad M_2 = X_1^2 X_2^3 X_3^5, \dots, \qquad M_6 = X_1^3 X_2.$$

In the output file we see

```
...
original monoid is not integrally closed in chosen lattice
...
6 Hilbert basis elements:
0 0 1
1 0 0
0 1 3
```

```
1 1 2
3 1 0
2 3 5

5 support hyperplanes:
0   0 1
0   1 0
1   0 0
2  -3 1
5 -15 7
```

The support hyprplanes re those of the cone generated by the monoid. They are used in aiuxiliary computations, for example in finding the Hilbert basis, i.e., the unique minimal system of generators of our monoid. In this case the input vectors are all in the Hilbert basis, but thins need not be the case. The Hilbert basis is ordered by degree and lexicographically within each degree. In fact, we have a grading

```
grading:
1 1 1
```

For the default choice of the grading we start from the standard grading on the ambient lattice. Then the grading, whether the default choice or an explicit grading in the input, is divided by the greatest common divisor of he degrees of the generators. In the context of monomial algebras it is the most natural choice. The division by the gcd can be suppressed by `NoGradingDenom`. In our example the gcd is 1.

Our monoid actually has another grading, in which all generators have degree 1: `grading 1 -2 1` in the input file. Activate it and study the changes.

Note: The input type `monoid` is close to `cone_and_lattice` if the monoid is normal.But there are two differences in the dafault choices: (1) The default computation goals and (2) the default grading. In fact, for `monoid` is is derived ftom the standard grading on the ambeint lattice, whereas or `cone_and_lattice` it gives degree 1 to the extreme integral generators, provided this is possible.

3.0.2 Markov and Gröbner bases, Representations

The purpoase of the computations in this section is to understand the defining ideal of the subalgebra A of $K[X_1,X_2,X_3]$ generfated by our binomials $M_1,\dots,M_6$ introduced above. To thias end we activate

`MarkovBasis`

in `monoid.in`, the Markov basis is computed and returned in the file with suffix

`mrk` file containing the Markov basis

In our case `monoid.mrk`:

```
7
```

```
6
1 0 -1 -1 1 0
-2 0 2 -1 0 1
-1 0 1 -2 1 1
2 1 0 -1 -1 -1
0 0 0 -3 2 1
1 1 1 -3 0 0
0 1 2 -2 -1 0
```

Each column corresponds to an input vector, and the rows aee indeed relations: the scalar product of a row listed in the Markov basis and a column of the matrix `monoid` is 0. The binomials in $P = K[Y_1, \dots, Y_6]$ corresponding to the rows in the Markov basis form a system of generators of the binomial ideal defining our monoid algebra as a residue class ring of P. The binomials are

$$b_1 = Y_1Y_5 - Y_3Y_4, \qquad b_2 = Y_3^2Y_6 - Y_1^2Y_3, \dots, \qquad b_7 = Y_2Y_3^2 - Y_4^2Y_5.$$

and indeed the binomials vanish if we subsritute M_i for Y_i, $i = 1, \dots, 7$. That they generate the defining ideal is claimed by Normaliz.

For easier reference the input matrix is mirrored in the file with suffix

ogn file with the original generators

in our case `monoid.ogn`:

```
6
3
1 0 0
2 3 5
0 0 1
1 1 2
0 1 3
3 1 0
```

In order to compute a Gröbner basis of our binomial ideal, we activate

GroebnerBasis

and get the output file with suffix

grb containing the Gröbner basis.

For the Gröbnerb basis one has to choose a monomial otrder. The default choice is 'egree reverse lexicographic" In our case it yields

```
8
6
-1 0 1 1 -1 0
0 0 0 3 -2 -1
1 0 -1 2 -1 -1
2 0 -2 1 0 -1
0 1 2 -2 -1 0
```

```
1 1 1 0 -2 -1
2 1 0 -1 -1 -1
3 0 -3 0 1 -1
```

More precisely: the indeterminates in the polynomial ring housing the binomials are ordered $Y_1 > \dots Y_6$ and we take the degree reverse lexicographic extension, where‘ degree” means the total standard degree on the polynomial ring P. The file with suffix `ogn` is also created for the Gröbner basis. There is no output of the (minimal) Markov basis, unless you ask for it explicitly.

Despite of being the defualt choice, the degree reverse lexicographic order is in the list of perttaining computation goals:

RevLex degree reverse lexicographic order

Lex lexicographic order

DegLex gegree lexicographic order

Activate also `Lex` in our example and see what changes. `DegLex` is taken with respect to the total standard degree as well, and makes no difference in our case, since the generating binomials are homogeneous in this grading.

A grading of the monomial algebra induces a gtrading on the binomials in its definig ideal such that the latter are homogeneous polynomials. With respect to this grading the output of Markov and Gröbner bases can be restricted:

gb_degree_bound <n> sets upper degree bound <n> for binomials,

gb_min_degree <n> sets lower degree bound <n> for binomials.

There is one more computation goal for monoids that complements `HilbertBasis` (switched on by default):

Representations representations of reeducible elements in `monoid` in terms of the Hilbert basis

The outpur is a list of binomials in the file with suffix

rep reprdsentations of reducible elements in terms of the Hilbert basis.

Also the file with suffix `ogn` is written.

As a simple example we consider `representations.in`

```
amb_space 3
monoid 8
1 0 0
2 3 5
0 1 1
0 0 1
1 1 2
0 1 3
3 1 0
1 2 5
Representations
```

```
/*BinomialsPacked*/
```

with

```
4 Hilbert basis elements:
0 0 1
1 0 0
0 1 1
3 1 0
```

and representations of the other 4 elements in the input:

```
4
8
-1 0 -1 -1 1 0 0 0
0 0 -1 -2 0 1 0 0
-1 0 -2 -3 0 0 0 1
-2 1 -3 -2 0 0 0 0
```

The entries 1 in each row mark the reducible elements and the row should be read as a binomial vanishing on the input vectors (or monomials).

If you want to see computations that take longer than our toy example so far, run `A443monoid.in` and `Kwak80.in`.

3.0.3 Hilbert series and muliplicity

If we activate both (!) `HilbertSeries` and `Multiplicity` in `monoid.in`, the result is

```
multiplicity = 19/40
multiplicity (float) = 0.475

Hilbert series:
1 1 0 0 3 2 -2 -1 6 2 -4 0 6 1 -3 1 4 0 0 1 1
denominator with 3 factors:
1:1  2:1  20:1
...
```

followed by the representation with cyclotomic denominator and the Hilbert quasipolynomial.Activate `grading 1 -2 1` and observe the changes.

3.0.4 Binomial ideals from cone input

Defining binomial ideals can be computed not only for monoids defined by the input type `monoid`, but also for the monoids that defined by other input types as intersections of cones and lattices, for example `cone`, `cone_and_lattice`, `equations`, `inequalities` etc.In the case of generator input there are actually two monoids, the “original monoid” as discussed in Section 7.18, and its integral closure in the lattice defined by tghe input. So, if we ask for the Markov basis of the defining ideal, which monoid is taken? Answer: always the integral closure generated by its Hilbert basis, unless the property makes only sense for the original monoid: if we ask `IsIntegrallyClosed`, the answer is always ‘yes” for the integral closure.

As an example we take `cone_latt_markov.in` (`monoid.in` with a different input type):

```
amb_space 3
cone_and_lattice 6
1 0 0
2 3 5
0 0 1
1 1 2
0 1 3
3 1 0
MarkovBasis
SingularLocus
```

The output file contains

```
original monoid is not integrally closed in chosen lattice
...
codim singular locus = 2
18 Markov basis elements

***********************************************************************

9 lattice points in polytope (Hilbert basis elements of degree 1):
0 0 1
0 1 3
1 0 0
1 1 2
1 2 4
2 1 1
2 2 3
2 3 5
3 1 0

0 further Hilbert basis elements of higher degree:
```

The Markov basis is contained in `cone_latt_markov.mrk`:

```
18
9
-1 0 1 0 0 1 0 0 -1
0 0 0 -1 0 2 0 0 -1
...
0 -1 0 0 2 0 0 -1 0
```

The columns correspond to the Hilbert basis elements in the order they are listed above. Fpr completemess the file with suffix `ogn` is written also in this case. It conztains the Hilbert basis as listed in the `out` file.

The singular locus has codimension 2, the minimum for a normal monoid (algebra). The singular locus is stored in `cone_latt_markov.sng`.

We could equally well start from the inequalities defining the integral closure (the generators above generate the lattice $\mathbb{R}^3$) in `cone_latt_markov_supp.in`,

```
amb_space 3
inequalities 5
0   0 1
0   1 0
1   0 0
2  -3 1
5 -15 7
MarkovBasis
SingularLocus
```

with the same result as above, except that there is no originl monoid.

3.1 Monoids from binomials

As an example, we consider the binomial ideal generated by

$$Y_1^2Y_2 - Y_4Y_5Y_6, \;\; Y_1Y_4^2 - Y_3Y_5Y_6, \;\; Y_1Y_2Y_3 - Y_5^2Y_6.$$

in the polynomial ring $P = K[Y_1,\dots,Y_6]$. We want to find an embedding of the toric ring it defines. When we say "defines", then we do not claim that the residue ring P/I is a toric ring. But there is a unique smallest binomial ideal $J \supset I$ withn this propetry, and Normaliz finds the monoid and, if wanted, also a Markov (or GRöbner)basis of J. A priori $R = P/J$ is only defined as a residue class ring. It doesn't have a 'canonical' embedding into another polynomial ring, but Normaliz computes such an embedding if the monoid undrrlying R/J isd positive. As pointed out already, non-positive affine monoids an only be computed by Normaliz if they are normal.

3.1.1 Affine monoids from binomial ideals

The input type that asks for a toric ring from binomial input is

`toric_ideal`

The input vectors are obtained as the differences of the two exponent vectors in the binomials. So the input ideal `toric_ideal.in` for ur 3 binimials is

```
amb_space 6
toric_ideal 3
2 1  0 -1 -1  -1
```

```
1 0 -1  2 -1  -1
1 1  1  0 -2  -1
/* total_dgegree */
```

In order to avoid special input rules for this case in which our object is not defined as a subset of an ambient space, but as a quotient of type *generators/relations*, we abuse the name `amb_space`: it determines the space in which the input vectors live.

It is possible to define a `grading`. It must give positive degree to the unit vectors of the ambient space and degree 0 to the vectors representing the binomials so that the latter become homogeneous polynomials with respect to this grading.

In the output we get

```
6 original generators:
1 0 0
2 3 5
0 0 1
1 1 2
0 1 3
3 1 0
```

namely the residue classes of the indeterminates realized in an embedding. Test the binomials on the original generators! We know this monoid already from `monid.in`, and you can try the other computationn goals discussed for the latter.

We see

```
grading:
1 1 1
```

So Normaliz uses the standard grading on the ambient polynomoal ring into which R/J has been embedded. This is the default choice, as it is for the input type `monolid`. Our toric ring actually has its own standard tgrading: activate `total_degree` in the input file and look at the output. In fact, the binomials above are homogeneous in the standard garing on P, and `total_degree` sets this grading.

The generators are repeated (in this case) in a different order,m as wem know altready:

```
6 Hilbert basis elements:
0 0 1
1 0 0
0 1 3
1 1 2
3 1 0
2 3 5
```

Now they are sorted by degree and then lexicographically, as we always sort Hilbert bases.

As a trivial example in which the Hilbert basis does not simply repeat the original generators in a different order, compute `lin_bin.in`:

```
amb_space 2
toric_ideal 1
1 -1
```

The output contains

```
2 original generators:
1
1

1 Hilbert basis elements:
1
```

3.1.2 Normalization of monoids from binomials

One go can a step further, using the input type

normal_toric_ideal

It asks for the *normalization* of the toric ring defined by the binomials. In `normal_toric_ideral.in` we take the same binoials as above:

```
amb_space 6
normal_toric_ideal 3
2 1  0 -1 -1  -1
1 0 -1  2 -1  -1
1 1  1  0 -2  -1
```

In the output file we find

```
6 original generators:
1 0 0
2 3 5
0 0 1
1 1 2
0 1 3
3 1 0

2 lattice points in polytope (Hilbert basis elements of degree 1):
0 0 1
1 0 0

7 further Hilbert basis elements of higher degree:
0 1 3
1 1 2
2 1 1
3 1 0
1 2 4
```

```
2 2 3
2 3 5
```

The “original generetors" arev the same as above, as they should be. Also the default grading is the same, aned the default computation goals are identical as well. But the Hilbert series, Markov basis, Gröbner basis etc. are computed for the normalization, as the user can see by playing with the commnted out computation goals.

Note: Until version 3.9.4 the input type `normal_toric_ideal` was called `lattice_ideal`, which has a different meaning now and is discussed in the next subsection.

3.2 Lattice ideals

A lttice ideal I in a polynomial ring P is a binomial ideal modulo which all monomials are nonzerodivisors. This implies that P/J is a monoid ring whose underlying monoid is the natural image of the monoid of monomials in P. Moreover, it is a cancellative monoid, but not necessarily affine—the latter property requires torsion freeness additionally. The input type is

`lattice_ideal`

Normaliz tests whether the lattice ideral is toric and indicates it in the input file, but does not automatically treat the input like `toric_ideal` in the positive case.

A soimple example of a non-toric lattice ideal is `non_toric.in`

```
amb_space 4
lattice_ideal 4
2 -2 0 0
1 1 -1 -1
2 -1 1 -2
-1 -1 1 -1
/*GroebnerBasis
DegLex*/
```

The default computation goal is `MarkovBsis`. In our case the result is

```
4
4
-4 -1 1 0
-3 0 0 1
-2 2 0 0
6 0 0 0
```

Attention: the last binomial is $x_1^6 - 1$ so that the residue class of x_1 is a torsion element in the monoid of residue classes.

For internal reasons and the exchnge of data with external programs we can ask

```
IsLatticeIdealToric
```

There is only one more allowed computation goal, namely `GroebnerfBasis`. There are no monoid generators for `lattice_ideal`.

Note: In versions until 3.9.4 `lattice_ideal` had the meaning of `normal_toric_ideal`.

References

[1] 4ti2 team. 4ti2-A software package for algebraic, geometric and combinatorial problems on linear spaces. Available at `https://github.com/4ti2/4ti2`.

[2] J. Abbott, A. M. Bigatti and G. Lagorio, *CoCoA-5: a system for doing Computations in Commutative Algebra*. Available at `http://cocoa.dima.unige.it`.

[3] V. Almendra and B. Ichim, *jNormaliz 1.7*. Available at `https://normaliz.uos.de`.

[4] V. Baldoni, N. Berline, J. A. De Loera, B. Dutra, M. Köppe, S. Moreinis, G. Pinto, M. Vergne and J. Wu, *A User's Guide for LattE integrale v1.7.2, 2013*. Software package LattE is available at `https://www.math.ucdavis.edu/~latte/`.

[5] D. Bremner, M. D. Sikirić, D. V. Pasechnik, Th. Rehn and A. Schürmann, *Computing symmetry groups of polyhedra*. LMS J. Comp. Math. 17 (2014), 565–581.

[6] St. Brumme, *Hash library*. Package available at `https://create.stephan-brumme.com/`.

[7] W. Bruns, *Automorphism groups and normal forms in Normaliz.* Res. Math. Sci. 9 (2022), no. 2, Paper No. 20, 15 pp.

[8] W. Bruns, *Polytope volume in Normaliz.* São Paulo J. Math. Sci. `https://doi.org/10.1007/s40863-022-00317-9`

[9] W. Bruns, P. Garcia-Sanchez, C. O'Neill and D. Wilburne, *Wilf's conjecture in fixed multiplicity*. Int. J. Algebra Comp. 30 (2020), 861–882.

[10] W. Bruns and J. Gubeladze, *Polytopes, rings, and K-theory*. Springer, 2009.

[11] W. Bruns, R. Hemmecke, B. Ichim, M. Köppe and C. Söger, *Challenging computations of Hilbert bases of cones associated with algebraic statistics*. Exp. Math. 20 (2011), 25–33.

[12] W. Bruns and B. Ichim, *Normaliz: algorithms for rational cones and affine monoids*. J. Algebra 324 (2010) 1098–1113.

[13] W. Bruns and B. Ichim, *Polytope volume by descent in the face lattice and applications in social choice*. Math. Prog. Comp. 113 (2020), 415–442.

[14] W. Bruns, B. Ichim and C. Söger, *The power of pyramid decomposition in Normaliz*. J. Symb. Comp. 74 (2016), 513–536.

[15] W. Bruns, B. Ichim and C. Söger, *Computations of volumes and Ehrhart series in four candidates elections*. Ann. Oper. Res. 280 (2019), 241–265.

[16] W. Bruns and R. Koch, *Computing the integral closure of an affine semigroup*. Univ. Iagell. Acta Math. 39 (2001), 59–70.

[17] W. Bruns, R. Sieg and C. Söger, *Normaliz 2013–2016*. In G. Böckle, W. Decker and G. Malle, editors, *Algorithmic and Experimental Methods in Algebra, Geometry, and Number Theory*, pages 123–146. Springer, 2018.

[18] W. Bruns and C. Söger, *The computation of weighted Ehrhart series in Normaliz*. J. Symb. Comp. 68 (2015), 75–86.

[19] B. Büeler and A. Enge, *Vinci.* Package available from `https://www.math.u-bordeaux.fr/~aenge/`

[20] B. Büeler, A. Enge, K. Fukuda, *Exact volume computation for polytopes: a practical study.* In: Polytopes - combinatorics and computation (Oberwolfach, 1997), pp. 131 – 154, DMV Sem. 29,

Birkhäuser, Basel, 2000.

[21] J. A. De Loera, R. Hemmecke and M. Köppe. Algebraic and geometric ideas in the theory of discrete optimization. MOS-SIAM Series on Optimization, 14. Society for Industrial and Applied Mathematics (SIAM), Philadelphia 2013.

[22] V. Delecroix, *embedded algebraic number fields (on top of antic)*, package available at `https://github.com/flatsurf/e-antic`.

[23] P. Filliman, *The volume of duals and sections of polytopes.* Mathematika 37 (1992), 67–80.

[24] S. Gutsche, M. Horn and C. Söger, *NormalizInterface for GAP*. Available at `https://github.com/gap-packages/NormalizInterface`.

[25] S. Gutsche and R. Sieg, *PyNormaliz - an interface to Normaliz from python*. Available at `https://github.com/Normaliz/PyNormaliz`.

[26] W. B. Hart, *Algebraic Number Theory In C*. Package available at `https://github.com/wbhart/antic`.

[27] W. B. Hart, F. Johansson and S. Pancratz, *FLINT: Fast Library for Number Theory*. Available at `https://flintlib.org`.

[28] Hemmecke and P. N. Malkin. Computing generating sets of lattice ideals and Markov bases of lattices. J. Symb. Comp. 44, 1463–1476 (2009).

[29] F. Johansson, *Arb - a C library for arbitrary-precision ball arithmetic*. Available at `https://arblib.org/`.

[30] J. Lawrence, *Polytope volume computation.* Math. Comp. 57 (1991), 259–271.

[31] M. Köppe and S. Verdoolaege, *Computing parametric rational generating functions with a primal Barvinok algorithm*. Electron. J. Comb. 15, No. 1, Research Paper R16, 19 p. (2008).

[32] B. D. McKay and A. Piperno, *Practical graph isomorphism, II*. J. Symbolic Comput. 60 (2014), 94–112.

[33] L. Pottier, *The Euclide algorithm in dimension n*. Research report, ISSAC 96, ACM Press 1996.

[34] A. Schürmann, *Exploiting polyhedral symmetries in social choice*. Social Choice and Welfare 40 (2013), 1097–1110.

[35] B. Sturmfels, *Gröbner baes and convex polytopes*. American Mathematical Society 1996.

Index of keywords