```
gap> g:= SymmetricGroup( 4 );
Sym( [ 1 .. 4 ] )
gap> tbl:= CharacterTable( g );;  HasIrr( tbl );
false
gap> tblmod2:= CharacterTable( tbl, 2 );
BrauerTable( Sym( [ 1 .. 4 ] ), 2 )
gap> tblmod2 = CharacterTable( tbl, 2 );
true
gap> tblmod2 = BrauerTable( tbl, 2 );
true
gap> libtbl:= CharacterTable( "M" );
CharacterTable( "M" )
gap> BrauerTable( libtbl, 2 );
fail
gap> CharacterTable( "Symmetric", 4 );
CharacterTable( "Sym(4)" )
gap> ComputedBrauerTables( tbl );
[ , BrauerTable( Sym( [ 1 .. 4 ] ), 2 ) ]
```

```
i5 : betti(t,Weights=>{1,0})

            0 1  2  3 4
o5 = total: 1 4 13 14 4
         0: 1 .  .  . .
         1: . 2  2  4 2
         2: . 2  5  6 .
         3: .  .  4  . 2
         4: .  .  .  4 .
         5: .  2  .  . .

o5 : BettiTally
i6 : betti(t,Weights=>{0,1})

            0 1  2  3 4
o6 = total: 1 4 13 14 4
         0: 1 .  .  . .
         1: . 2  2  4 2
         2: . 2  5  6 .
         3: .  .  4  . 2
         4: .  .  .  4 .
         5: .  2  .  . .

o6 : BettiTally
i7 : t1 = betti(t,Weights=>{1,1})

            0 1  2  3 4
o7 = total: 1 4 13 14 4
         0: 1 .  .  . .
         1: .  .  .  . .
         2: .  .  .  . .
         3: . 2  .  . .
         4: .  .  .  . .
         5: . 2  .  . .
         6: .  .  1  . .
         7: .  .  8  6 .
         8: .  .  4  8 4

o7 : BettiTally
i8 : peek t1

o8 = BettiTally{(0, {0, 0}, 0) => 1 }
                (1, {2, 2}, 4) => 2
                (1, {3, 3}, 6) => 2
                (2, {3, 7}, 10) => 2
                (2, {4, 4}, 8) => 1
                (2, {4, 5}, 9) => 4
                (2, {5, 4}, 9) => 4
                (2, {7, 3}, 10) => 2
                (3, {4, 7}, 11) => 4
                (3, {5, 5}, 10) => 6
                (3, {7, 4}, 11) => 4
                (4, {5, 7}, 12) => 2
                (4, {7, 5}, 12) => 2
```

```
ring r1 = 32003,(x,y,z),ds;
int a,b,c,t=11,5,3,0;
poly f = x^a+y^b+z^(3*c)+x^(c+2)*y^(c-1)+x^
         x^(c-2)*y^c*(y^2+t*x)^2;
option(noprot);
timer=1;
ring r2 = 32003,(x,y,z),dp;
poly f=imap(r1,f);
ideal j=jacob(f);
vdim(std(j));
==> 536
vdim(std(j+f));
==> 195
timer=0;  // reset timer
```

Journal of Software for Algebra and Geometry

vol 9 2019

JSAG **9** (2019), 1–9
dx.doi.org/10.2140/jsag.2019.9.1

The Journal of Software for
Algebra and Geometry

Strongly stable ideals and Hilbert polynomials

DAVIDE ALBERELLI AND PAOLO LELLA

ABSTRACT: The `StronglyStableIdeals.m2` package for Macaulay2 provides a method to compute all saturated strongly stable ideals in a given polynomial ring with a fixed Hilbert polynomial. A description of the main method and auxiliary tools is given.

INTRODUCTION. Strongly stable ideals are a key tool in commutative algebra and algebraic geometry. These ideals have nice combinatorial properties that make them well suited for both theoretical and computational applications. In the case of polynomial rings with coefficients in a field of characteristic zero, the notion of strongly stable ideals coincides with the notion of Borel-fixed ideals. Such ideals are fixed by the action of the Borel subgroup of triangular matrices and play a special role in the theory of Gröbner bases because initial ideals in generic coordinates are of this type [Galligo 1974].

In the context of parameter spaces of algebraic varieties, Galligo's theorem says that each component and each intersection of components of a Hilbert scheme contains at least one point corresponding to a scheme defined by a Borel-fixed ideal. Hence, these ideals are distributed throughout the Hilbert scheme and can be used to study its local structure. To this end, in recent years several authors [Lella and Roggero 2011; 2016; Cioffi and Roggero 2011; Bertone et al. 2013a; 2017a; 2017b] developed algorithmic methods based on the use of strongly stable ideals to construct flat families corresponding to special loci of the Hilbert scheme. In particular, a new open cover of the Hilbert scheme has been defined using strongly stable ideals and the action of the projective linear group [Bertone et al. 2013b; Brachat et al. 2016]. In this construction, the list of all points corresponding to Borel-fixed ideals in a given Hilbert scheme is needed. The main feature of the package `StronglyStableIdeals.m2` is a method to compute this set of points, i.e., the list of all saturated strongly stable ideals in a polynomial ring with a given

The second author is a member of GNSAGA .

MSC2010: primary 13P10; secondary 13P99.

Keywords: strongly stable ideal, Borel-fixed ideal, Hilbert polynomial, Gotzmann number, Hilbert scheme.

`StronglyStableIdeals.m2` version 1.1

Hilbert polynomial. The method has been theoretically introduced in [Cioffi et al. 2011] and improved in [Lella 2012]. Several other tools are developed and presented in the current paper.

1. STRONGLY STABLE IDEALS. Let us denote by $\mathbb{K}[\boldsymbol{x}]$ the polynomial ring in $n+1$ variables $\mathbb{K}[x_0, \dots, x_n]$ with coefficients in a field $\mathbb{K}$. We assume that $x_0 > x_1 > \cdots > x_n$. We use the multi-index notation to describe monomials, i.e., $\boldsymbol{x}^{\boldsymbol{\alpha}} := x_0^{\alpha_0} \cdots x_n^{\alpha_n}$ for every $\boldsymbol{\alpha} = (\alpha_0, \dots, \alpha_n) \in \mathbb{Z}_{\geqslant 0}^{n+1}$, and we denote by $\mathbb{T}_{n,s}$ the set of monomials of $\mathbb{K}[\boldsymbol{x}]$ of degree s. For any monomial $\boldsymbol{x}^{\boldsymbol{\alpha}}$, we denote by $\min \boldsymbol{x}^{\boldsymbol{\alpha}}$ and $\max \boldsymbol{x}^{\boldsymbol{\alpha}}$ the indices of the minimal and maximal variable dividing $\boldsymbol{x}^{\boldsymbol{\alpha}}$.

Following [Green 2010], *increasing* and *decreasing elementary moves* are defined as the multiplications

$$\mathrm{e}_i^+(\boldsymbol{x}^{\boldsymbol{\alpha}}) := \frac{x_{i-1}}{x_i} \cdot \boldsymbol{x}^{\boldsymbol{\alpha}}, \quad i > 0, \quad \text{and} \quad \mathrm{e}_j^-(\boldsymbol{x}^{\boldsymbol{\alpha}}) := \frac{x_{j+1}}{x_j} \cdot \boldsymbol{x}^{\boldsymbol{\alpha}}, \quad j < n.$$

We say that an elementary move $\mathrm{e}_i^{+/-}$ is *admissible* for a monomial $\boldsymbol{x}^{\boldsymbol{\alpha}}$ if $\alpha_i > 0$, that is, $\mathrm{e}_i^{+/-}(\boldsymbol{x}^{\boldsymbol{\alpha}})$ is a monomial of $\mathbb{K}[\boldsymbol{x}]$.

Definition 1.1. An ideal $I \subset \mathbb{K}[\boldsymbol{x}]$ is called *strongly stable* if

(i) I is a monomial ideal;

(ii) for every $\boldsymbol{x}^{\boldsymbol{\alpha}} \in I$ and for every admissible increasing move e_i^+, the monomial $\mathrm{e}_i^+(\boldsymbol{x}^{\boldsymbol{\alpha}})$ is contained in I.

We recall that a strongly stable ideal is a Borel-fixed ideal. We now summarize some properties holding in general for Borel-fixed ideals and useful in this context.

Proposition 1.2 [Green 2010, Section 2]. *Let $I \subset \mathbb{K}[\boldsymbol{x}]$ be a strongly stable ideal.*

(i) *The regularity of I is equal to the maximal degree of a generator.*

(ii) *Let $\mathfrak{m}$ be the irrelevant ideal of $\mathbb{K}[\boldsymbol{x}]$. Then, $(I : \mathfrak{m}) = (I : x_n)$, so that the ideal I is saturated if no generator involves the last variable x_n.*

(iii) *The last variable x_n is a regular element for I, i.e., the multiplication by x_n induces the short exact sequence*

$$0 \longrightarrow \frac{\mathbb{K}[\boldsymbol{x}]}{I}(t-1) \xrightarrow{\cdot x_n} \frac{\mathbb{K}[\boldsymbol{x}]}{I}(t) \longrightarrow \frac{\mathbb{K}[\boldsymbol{x}]}{(x_n, I)}(t) \longrightarrow 0.$$

2. HILBERT POLYNOMIALS. The Hilbert polynomial $p(t)$ of a homogeneous ideal $I \subset \mathbb{K}[\boldsymbol{x}]$ is the numerical polynomial such that for s sufficiently large

$$\dim_{\mathbb{K}}\left(\frac{\mathbb{K}[\boldsymbol{x}]}{I}\right)_s = \dim_{\mathbb{K}}\left(\frac{\mathbb{K}[\boldsymbol{x}]_s}{I_s}\right) = \binom{n+s}{n} - \dim_{\mathbb{K}} I_s = p(s).$$

Obviously, not every numerical polynomial is a Hilbert polynomial of some homogeneous ideal. Those being Hilbert polynomials have been completely described by Gotzmann [1978].

Gotzmann's decomposition. *A numerical polynomial $p(t) \in \mathbb{Q}[t]$ is a Hilbert polynomial if, and only if, it can be written as*

$$p(t) = \binom{n+a_1}{a_1} + \binom{n+a_2-1}{a_2} + \cdots + \binom{n+a_r-(r-1)}{a_r}, \quad a_1 \geqslant \cdots \geqslant a_r \geqslant 0. \tag{1}$$

This decomposition is strictly related to Macaulay's decomposition

$$p(t) = \sum_{k=0}^{d} \left[\binom{t+k}{k+1} - \binom{t+k-m_k}{k+1} \right],$$

where $d = \deg p(t)$. For all $n \geqslant d+1$ the saturated lexicographic ideal within $\mathbb{K}[x_0, \ldots, x_n]$ with Hilbert polynomial $p(t)$ is

$$(x_0, \ldots, x_{n-d-2}, x_{n-d-1}^{b_d+1}, x_{n-d-1}^{b_d} x_{n-d}^{b_{d-1}+1}, \ldots, x_{n-d-1}^{b_d} x_{n-d}^{b_{d-1}} \cdots x_{n-1}^{b_0}),$$

where

$$b_d = \#\{a_j \mid a_j = d\} = m_d \quad \text{and} \quad b_k = \#\{a_j \mid a_j = k\} = m_k - m_{k+1}, \quad 0 \leqslant k < d.$$

The description of the lexicographic ideal in terms of Gotzmann's decomposition gives an insight to the following theorem.

Gotzmann's regularity theorem. *The regularity of a saturated ideal $I \subset \mathbb{K}[\boldsymbol{x}]$ with Hilbert polynomial $p(t)$ is at most r, where r is the number of terms in the decomposition* (1) *and it is called the* ***Gotzmann number*** *of $p(t)$.*

Example 2.1. The package `StronglyStableIdeals.m2` provides the method `isHilbertPolynomial` to determine if a numerical polynomial is a Hilbert polynomial.

```
Macaulay2, version 1.11
with packages: ConwayPolynomials, Elimination, IntegralClosure,
               InverseSystems, LLLBases, PrimaryDecomposition,
               ReesAlgebra, TangentCone

i1 : loadPackage "StronglyStableIdeals";

i2 : QQ[t];

i3 : isHilbertPolynomial (4*t)

o3 = true

i4 : isHilbertPolynomial (5*t-6)

o4 = false
```

Gotzmann's and Macaulay's decompositions of a Hilbert polynomial can be computed using `gotzmannDecomposition` and `macaulayDecomposition`. These

methods return the list of terms in the decompositions. The summand $\binom{t+e}{c}$ is constructed with the command `projectiveHilbertPolynomial(c,c-e)`.

```
i5 : gotzmannDecomposition (4*t)
o5 = {P , - P  + P , - 2*P  + P , - 3*P  + P , P , P }
       1     0    1       0    1       0    1   0   0
o5 : List
i6 : macaulayDecomposition (4*t)
o6 = {- P  + P , 7*P  - P , - P  + P , - 10*P  + 5*P  - P }
         0    1     0    1     1    2        0      1    2
o6 : List
```

Finally, the saturated lexicographic ideal L with Hilbert polynomial $p(t)$ in the polynomial ring $\mathbb{K}[\boldsymbol{x}]$ can be computed with the method `lexIdeal` and its regularity is equal to the Gotzmann number of $p(t)$.

```
i7 : L = lexIdeal (4*t, QQ[x,y,z,w])
                    5   4 2
o7 = ideal (x, y , y z )
o7 : Ideal of QQ[x, y, z, w]
i8 : regularity L == gotzmannNumber (4*t)
o8 = true
```

3. THE MAIN ALGORITHM. In this section, we outline the strategy of the main algorithm. This algorithm was firstly described in [Cioffi et al. 2011] and then optimized in [Lella 2012]. The same problem has been previously discussed in [Reeves 1992] and an alternative algorithm was later presented in [Moore and Nagel 2014].

We need to relate the properties of a strongly stable ideal with its Hilbert polynomial. If I is a strongly stable ideal, for each $s \in \mathbb{N}$ the monomial basis of the homogeneous piece I_s of the ideal is a subset of $\mathbb{T}_{n,s}$ closed by increasing elementary moves. We call *Borel sets* such subsets of $\mathbb{T}_{n,s}$ (see Figure 1 for an example). Proposition 1.2(i) implies that the monomial basis of I_s for a saturated strongly stable ideal $I \subset \mathbb{K}[\boldsymbol{x}]$ with Hilbert polynomial $p(t)$ and regularity at most s is a Borel set with $q(s) := \binom{n+s}{n} - p(s)$ elements. Thus, we consider the map

$$\left\{ \begin{array}{c} \text{saturated strongly stable ideals in } \mathbb{K}[\boldsymbol{x}] \text{ with} \\ \text{Hilbert polynomial } p(t) \text{ and regularity } \leqslant s \end{array} \right\} \hookrightarrow \left\{ \begin{array}{c} \text{Borel sets of } \mathbb{T}_{n,s} \\ \text{with } q(s) \text{ elements} \end{array} \right\}. \quad (2)$$

Moreover, Gotzmann's regularity theorem suggests considering s equal to the Gotzmann number of $p(t)$ to determine all saturated strongly stable ideals with Hilbert polynomial $p(t)$. Obviously, there are many Borel sets in $\mathbb{T}_{n,s}$ with $q(s)$ elements not corresponding to an ideal with Hilbert polynomial $p(t)$. To identify the image of the previous map, we recall a definition and a proposition by Mall.

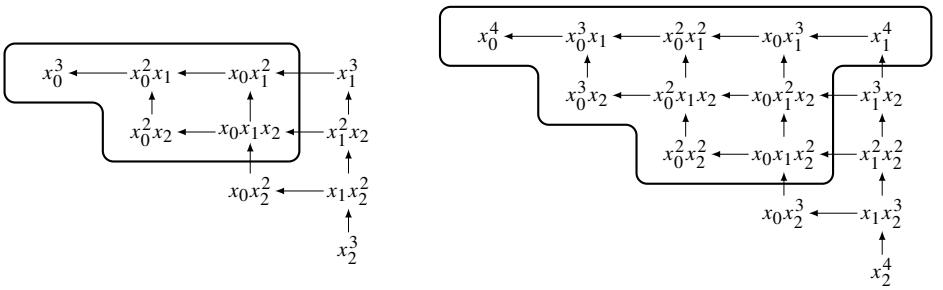

Figure 1. The Borel sets defined in $\mathbb{T}_{2,3}$ and $\mathbb{T}_{2,4}$ by the ideal $(x_0^2, x_0x_1, x_1^4) \subset \mathbb{K}[x_0, x_1, x_2]$.

Definition 3.1 [Mall 1997, Definition 2.7]. Let $B \subset \mathbb{T}_{n,s}$ be a Borel set. The set $B^{(i)} := \{\boldsymbol{x}^{\boldsymbol{\alpha}} \in B \mid \min \boldsymbol{x}^{\boldsymbol{\alpha}} = n - i\}$ is called the *i-growth class* of B. The sequence $\mathrm{gv}(B) := (|B^{(0)}|, \dots, |B^{(n)}|)$ is called the *growth vector* of B.

Proposition 3.2 [Mall 1997, Proposition 3.2]. *Let $I \subset \mathbb{K}[\boldsymbol{x}]$ be a strongly stable ideal generated by the monomials of a Borel set $B \subset \mathbb{T}_{n,s}$ and let $p(t)$ be its Hilbert polynomial. Then*,

$$p(t) = \binom{n+t}{n} - \sum_{k=0}^{n} |B^{(k)}| \binom{k+t-s}{k}, \qquad \text{for all } t \geqslant s. \tag{3}$$

We can use this result to determine the growth vector of a Borel set $B \subset \mathbb{T}_{n,s}$ starting from the Hilbert polynomial. The i-th difference polynomial of $p(t)$ is

$$(\Delta^i p)(t) = (\Delta^{i-1} p)(t) - (\Delta^{i-1} p)(t-1) = \binom{n+t-i}{n-i} - \sum_{k=i}^{n} |B^{(k)}| \binom{k+t-s-i}{k-i}.$$

Evaluating these identities at $t = s$, we obtain the linear system

$$\begin{cases} \sum_{k=0}^{n} |B^{(k)}| = \binom{n+s}{n} - p(s), \\ \quad \vdots \\ \sum_{k=i}^{n} |B^{(k)}| = \binom{n+s-i}{n-i} - (\Delta^i p)(s), \\ \quad \vdots \\ |B^{(n)}| = \binom{s}{0} - (\Delta^n p)(s), \end{cases} \tag{4}$$

whose solution is

$$|B^{(i)}| = \sum_{k=i}^{n} |B^{(k)}| - \sum_{k=i+1}^{n} |B^{(k)}| = \binom{n+s-i-1}{n-i} - (\Delta^i p)(s) + (\Delta^{i+1} p)(s), \quad i < n,$$

and $|B^{(n)}| = 1$ (recall that $(\Delta^i p)(t) \equiv 0$ for $i > \deg p(t)$ and $\deg p(t) < n$). Let us call the *growth vector* of $p(t)$ in degree s the solution of the linear system (4) and let us denote it by $\mathrm{gv}_s(p(t))$.

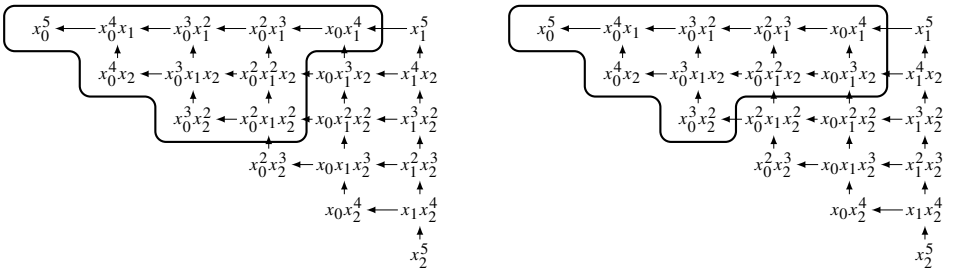

Figure 2. Borel sets in $\mathbb{T}_{2,5}$ corresponding to the saturated strongly stable ideals $(x_0^3, x_0^2x_1, x_0x_1^4)$ (on the left) and $(x_0^3, x_0^2x_1^2, x_0x_1^3)$ (on the right) in 3 variables with Hilbert polynomial $t+6$ and regularity at most 5.

Proposition 3.3 (cf. [Lella 2012, Theorem 3.3]). *Let $p(t)$ be a Hilbert polynomial. There is a bijective map*

$$\left\{\begin{array}{c}\textit{saturated strongly stable ideals}\\ \textit{in } \mathbb{K}[\boldsymbol{x}] \textit{ with Hilbert polynomial}\\ p(t) \textit{ and regularity } \leqslant s\end{array}\right\} \overset{1:1}{\longleftrightarrow} \left\{\begin{array}{c}\textit{Borel sets of } \mathbb{T}_{n,s}\\ \textit{with } q(s) \textit{ elements and}\\ \textit{growth vector } \operatorname{gv}_s(p(t))\end{array}\right\}, \tag{5}$$

$$\begin{array}{ccc} I & \longrightarrow & \textit{monomial basis of } I_s,\\ \textit{saturation of } (B) & \longleftarrow & B.\end{array}$$

In order to determine the Borel sets of Proposition 3.3, we use a recursive algorithm based on Proposition 1.2(iii). Indeed, if $I \subset \mathbb{K}[x_0, \dots, x_n]$ is a strongly stable ideal with Hilbert polynomial $p(t)$ and B is the associated Borel set in $\mathbb{T}_{n,s}$, then the subset $B' = \{\boldsymbol{x}^{\boldsymbol{\alpha}} \in B \mid \min \boldsymbol{x}^{\boldsymbol{\alpha}} > n\} \subset B$ is a Borel set in $\mathbb{T}_{n-1,s}$ corresponding to the strongly stable ideal $I' = (x_n, I) \cap \mathbb{K}[x_0, \dots, x_{n-1}] \subset \mathbb{K}[x_0, \dots, x_{n-1}]$ with Hilbert polynomial $(\Delta p)(t)$.

Example 3.4. We want to determine the set of strongly stable ideals in the polynomial ring $\mathbb{K}[x_0, x_1, x_2]$ with regularity at most 5 defining schemes with Hilbert polynomial $p(t) = t+6$. The Gotzmann number of $p(t)$ is 6 and its growth vector in degree 5 is $\operatorname{gv}_5(t+6) = (5, 4, 1)$. We start considering the set of strongly stable ideals in $\mathbb{K}[x_0, x_1]$ with Hilbert polynomial $\Delta p(t) = 1$ and regularity at most 5 corresponding to Borel sets with growth vector $\operatorname{gv}_5(\Delta p(t)) = (4, 1)$. There is a unique Borel set

$$B' = \left\{x_0^5, x_0^4x_1, x_0^3x_1^2, x_0^2x_1^3, x_0x_1^4\right\}.$$

Since x_1^5 is not contained in B', a Borel set $B \subset \mathbb{T}_{2,5}$ with growth vector $(5, 4, 1)$ does not contain monomials obtained from x_1^5 by applying decreasing elementary moves, i.e., $x_1^4x_2$, $x_1^3x_2^2$, $x_1^2x_2^3$, $x_1x_2^4$ and x_2^5. Hence, we need to select five monomials divisible by both x_0 and x_2 producing a set closed by increasing elementary moves (see Figure 2).

Our package provides the method `stronglyStableIdeals` to compute the set of strongly stable ideals of a given polynomial ring with fixed Hilbert polynomial and bounded regularity.

```
i9 : stronglyStableIdeals (4*t, QQ[x,y,z,w])
                   5   4 2                        2   4    5
o9 = {ideal (x, y , y z ), ideal (x*z, x*y, x , y z, y ),
                    2     2   4                  2   3
      ideal (x*y, x , x*z , y ), ideal (x*y, x , y )}
o9 : List
i10 : stronglyStableIdeals (4*t, QQ[x,y,z,w], MaxRegularity => 4)
                    2     2   4                  2   3
o10 = {ideal (x*y, x , x*z , y ), ideal (x*y, x , y )}
o10 : List
```

4. SEGMENT IDEALS. The transitive closure of the order relation

$$\boldsymbol{x}^{\boldsymbol{\alpha}} >_B \boldsymbol{x}^{\boldsymbol{\beta}} \quad \Longleftrightarrow \quad \boldsymbol{x}^{\boldsymbol{\beta}} = \mathrm{e}_i^-(\boldsymbol{x}^{\boldsymbol{\alpha}}) \tag{6}$$

induces a partial order on the set of monomials of any degree called the *Borel order*. Every graded term ordering is a refinement of this partial order. Since a Borel set B is closed with respect to the Borel order, i.e., $\boldsymbol{x}^{\boldsymbol{\alpha}} >_B \boldsymbol{x}^{\boldsymbol{\beta}},\ \boldsymbol{x}^{\boldsymbol{\beta}} \in B \Rightarrow \boldsymbol{x}^{\boldsymbol{\alpha}} \in B$, it is natural to ask whether there exists a term ordering $\prec$ with the same property. For instance, for the lexicographic ideal, the graded lexicographic order separates, in each degree, monomials contained in the ideal from those outside. In [Cioffi et al. 2011], several notions of segment ideals are introduced.

Definition 4.1 [Cioffi et al. 2011, Definitions 3.1 and 3.7]. A Borel set $B \subset \mathbb{T}_{n,s}$ is called a *segment* if there exists a term ordering $\prec$ such that $\boldsymbol{x}^{\boldsymbol{\alpha}} \succ \boldsymbol{x}^{\boldsymbol{\beta}}$, for all $\boldsymbol{x}^{\boldsymbol{\alpha}} \in B$ and $\boldsymbol{x}^{\boldsymbol{\beta}} \in \mathbb{T}_{n,s} \setminus B$.

Let $I \subset \mathbb{K}[\boldsymbol{x}]$ be a saturated strongly stable ideal.

(i) I is called a *hilb-segment* if the Borel set $I \cap \mathbb{T}_{n,r}$ is a segment, where r is the Gotzmann number of the Hilbert polynomial of I.

(ii) I is called a *reg-segment* if the Borel set $I \cap \mathbb{T}_{n,m}$ is a segment, where m is the regularity of I.

(iii) I is called a *gen-segment* if there exists a term ordering $\prec$ such that $\boldsymbol{x}^{\boldsymbol{\alpha}} \succ \boldsymbol{x}^{\boldsymbol{\beta}}$ for each minimal generator $\boldsymbol{x}^{\boldsymbol{\alpha}}$ of degree s of I and for all $\boldsymbol{x}^{\boldsymbol{\beta}} \in \mathbb{T}_{n,s} \setminus I_s$.

These notions are very important in the construction of flat families based on properties of Gröbner bases and in general for the study of the Hilbert scheme. The `StronglyStableIdeals.m2` package provides three methods for determining whether a strongly stable ideal may be some type of segment (and, in case, gives the term ordering). These methods use tools of the package `gfanInterface.m2` and the term ordering is given as a weight vector.

```
i11 : sevenPointsP2 = stronglyStableIdeals (7, 3, MaxRegularity => 5)

                 2     2   5           2   4     3
o11 = {ideal (x , x x , x ), ideal (x , x , x x ),
               0   0 1   1           0   1   0 1

                  2   2     3   4
       ideal (x x , x x , x , x )}
               0 1   0 1   0   1

o11 : List

i12 : for J in sevenPointsP2 list isHilbSegment J

o12 = {(true, {7, 3, 1}), (false, ), (true, {4, 3, 1})}

o12 : List

i13 : for J in sevenPointsP2 list isRegSegment J

o13 = {(true, {7, 3, 1}), (false, ), (true, {4, 3, 1})}

o13 : List

i14 : for J in sevenPointsP2 list isGenSegment J

o14 = {(true, {6, 3, 1}), (true, {4, 3, 1}), (true, {4, 3, 1})}

o14 : List
```

SUPPLEMENT. Version 1.1 of `StronglyStableIdeals.m2` is contained in the online supplement.

REFERENCES.

[Bertone et al. 2013a] C. Bertone, F. Cioffi, P. Lella, and M. Roggero, "Upgraded methods for the effective computation of marked schemes on a strongly stable ideal", *J. Symbolic Comput.* **50** (2013), 263–290. MR Zbl

[Bertone et al. 2013b] C. Bertone, P. Lella, and M. Roggero, "A Borel open cover of the Hilbert scheme", *J. Symbolic Comput.* **53** (2013), 119–135. MR Zbl

[Bertone et al. 2017a] C. Bertone, F. Cioffi, and M. Roggero, "Double-generic initial ideal and Hilbert scheme", *Ann. Mat. Pura Appl.* **196**:1 (2017), 19–41. MR Zbl

[Bertone et al. 2017b] C. Bertone, F. Cioffi, and M. Roggero, "Macaulay-like marked bases", *J. Algebra Appl.* **16**:5 (2017), 1750100, 36. MR Zbl

[Brachat et al. 2016] J. Brachat, P. Lella, B. Mourrain, and M. Roggero, "Extensors and the Hilbert scheme", *Ann. Sc. Norm. Super. Pisa Cl. Sci.* **16**:1 (2016), 65–96. MR Zbl

[Cioffi and Roggero 2011] F. Cioffi and M. Roggero, "Flat families by strongly stable ideals and a generalization of Gröbner bases", *J. Symbolic Comput.* **46**:9 (2011), 1070–1084. MR Zbl

[Cioffi et al. 2011] F. Cioffi, P. Lella, M. G. Marinari, and M. Roggero, "Segments and Hilbert schemes of points", *Discrete Math.* **311**:20 (2011), 2238–2252. MR Zbl

[Galligo 1974] A. Galligo, "À propos du théorème de-préparation de Weierstrass", pp. 543–579 in *Fonctions de plusieurs variables complexes*, edited by F. Norguet, Lecture Notes in Math. **409**, Springer, 1974. MR Zbl

[Gotzmann 1978] G. Gotzmann, "Eine Bedingung für die Flachheit und das Hilbertpolynom eines graduierten Ringes", *Math. Z.* **158**:1 (1978), 61–70. MR Zbl

[Green 2010] M. L. Green, "Generic initial ideals", pp. 119–186 in *Six lectures on commutative algebra*, edited by J. Elias et al., Birkhäuser Verlag, Basel, 2010. MR

[Lella 2012] P. Lella, "An efficient implementation of the algorithm computing the Borel-fixed points of a Hilbert scheme", pp. 242–248 in *ISSAC 2012—Proceedings of the 37th International Symposium on Symbolic and Algebraic Computation*, edited by J. van der Hoeven and M. van Hoeij, ACM, New York, 2012. MR Zbl

[Lella and Roggero 2011] P. Lella and M. Roggero, "Rational components of Hilbert schemes", *Rend. Semin. Mat. Univ. Padova* **126** (2011), 11–45. MR Zbl

[Lella and Roggero 2016] P. Lella and M. Roggero, "On the functoriality of marked families", *J. Commut. Algebra* **8**:3 (2016), 367–410. MR Zbl

[Mall 1997] D. Mall, "Betti numbers, Castelnuovo Mumford regularity, and generalisations of Macaulay's theorem", *Comm. Algebra* **25**:12 (1997), 3841–3852. MR Zbl

[Moore and Nagel 2014] D. Moore and U. Nagel, "Algorithms for strongly stable ideals", *Math. Comp.* **83**:289 (2014), 2527–2552. MR Zbl

[Reeves 1992] A. A. Reeves, *Combinatorial structure on the Hilbert scheme*, Ph.D. thesis, Cornell University, Ann Arbor, MI, 1992, Available at https://search.proquest.com/docview/303996863. MR

RECEIVED: 26 Jun 2014 REVISED: 22 Jun 2018 ACCEPTED: 4 Nov 2018

DAVIDE ALBERELLI:

davide.alberelli@gmail.com

PAOLO LELLA:

paolo.lella@polimi.it
Dipartimento di Matematica, Politecnico di Milano, Milano, Italy

JSAG **9** (2019), 11–17
dx.doi.org/10.2140/jsag.2019.9.11

The Journal of Software for
Algebra and Geometry

DiffAlg: a Differential algebra package

MANUEL DUBINSKY, CÉSAR MASSRI,
ARIEL MOLINUEVO AND FEDERICO QUALLBRUNN

ABSTRACT: In this article we present `DiffAlg.m2`, a differential algebra package for Macaulay2. It can perform the following operations: wedge products and exterior differentials of differential forms, contraction and Lie derivatives of differential forms with respect to a vector field and Lie brackets between vector fields.

Given a homogeneous differential operator of degree one D, the lack of an algebraic module structure attached to the kernel or image of D hinders the study of D. The main purpose of `DiffAlg.m2` is to handle these spaces degree-wise.

MOTIVATION AND DESCRIPTION OF THE PACKAGE. Algebraic and differential operations arise naturally when working with differential forms and vector fields, e.g., wedge products and exterior differentials of differential forms, contraction and Lie derivatives of differential forms with respect to a vector field and Lie brackets between vector fields. Some important statements involving these operations include the following:

(a) A differential r-form ω in the affine space $\mathbb{K}^{n+1}$ *descends to the projective space* $\mathbb{P}^n_{\mathbb{K}}$ if it satisfies the equation

$$i_R\omega = 0,$$

where $\mathbb{K}$ is a field, R is the radial vector field $R = \sum x_i \frac{\partial}{\partial x_i}$ and i_R denotes the contraction; see [Hartshorne 1977, Theorem 8.13, p. 176].

(b) If ω is a differential 1-form, then ω defines a foliation in $\mathbb{K}^{n+1}$ if it satisfies the *Frobenius integrability condition* given by the equation

$$\omega \wedge d\omega = 0;$$

see [Suwa 1995, Definition 2.2, p. 823].

The authors were supported by CONICET, Argentina.
MSC2010: 14-04, 53-04.
Keywords: differential operators, exterior algebra, vector fields.
`DiffAlg.m2` version 1.5

(c) Let ω be an integrable 1-form and $L_X\omega$ be the Lie derivative of ω with respect to a vector field X. Then, the solutions of the equation

$$L_X\omega = 0$$

define all the *infinitesimal automorphisms* of the foliation given by ω; see [Suwa 1995, Proposition 7.7, p. 845].

(d) Let ω be an integrable 1-form. Then the *tangent space of the space of foliations* at ω is given by the differential 1-forms η that satisfy the equation

$$\omega \wedge d\eta + d\omega \wedge \eta = 0;$$

see [Cukierman et al. 2009, Section 2.1. p. 709].

(e) Let D be a *bracket generating distribution*. Some important invariants of D are the ranks of the derived sequence

$$a(p) := \operatorname{rank} D^{(p)} = \operatorname{rank}(D^{(p-1)} + [D, D^{(p-1)}]);$$

see [Tanaka 1970, §1, pp. 8–9].

(f) A *symplectic structure* in a variety of dimension $2r$ is given by a 2-form ω such that $d\omega = 0$ and $\omega^r \neq 0$; see [Bryant et al. 1991, p. 41].

For a clear understanding of how `DiffAlg.m2` deals with such equations, let us fix some notation.

Let $S = \mathbb{K}[x_0, \ldots, x_n]$ be the polynomial ring in $n+1$ variables and let $\Omega = \bigoplus_{r\geq 0} \Omega^r$ be the exterior algebra of differential forms of S over $\mathbb{K}$. Let $\Omega^r(d)$ denote the space of r-forms with polynomial coefficients of homogeneous degree d, where we assign degree 1 to each x_i and degree 0 to each dx_i.

Therefore $\omega \in \Omega^r(d)$ can be written as

$$\omega = \sum_{\substack{I\subset\{0,\ldots,n\}\\ \#I=r}} \sum_{\substack{\alpha\in\mathbb{N}^{n+1}\\ |\alpha|=d}} a_{\alpha,I}\, x^\alpha\, dx_I, \qquad a_{\alpha,I} \in \mathbb{K}, \tag{1}$$

where, for each I of the form $I = \{i_1, \ldots, i_r\} \subset \{0, \ldots, n\}$, we let dx_I denote $dx_{i_1} \wedge \cdots \wedge dx_{i_r}$ and for each $\alpha = (\alpha_0, \ldots, \alpha_n) \in \mathbb{N}^{n+1}$ we denote $|\alpha| := \sum_{i=0}^n \alpha_i$ and $x^\alpha := x_0^{\alpha_0} \cdot \ldots \cdot x_n^{\alpha_n}$.

Let T be the module of vector fields with coefficients in S. Let $T(e)$ denote the homogeneous vector fields with polynomial coefficients of degree e, where we assign degree 1 to each x_i and degree 0 to each $\frac{\partial}{\partial x_i}$ and, analogously to (1), $X \in T(e)$ can be written as

$$X = \sum_{i=0}^{n} \sum_{\substack{\beta\in\mathbb{N}^{n+1}\\ |\beta|=e}} b_{\beta,i}\, x^\beta\, \frac{\partial}{\partial x_i}, \qquad b_{\beta,i} \in \mathbb{K}.$$

Current algebraic software systems implement functionality to deal with differential forms and vector fields, but usually scalar parameters $a_{\alpha,I}$ and $b_{\beta,i}$ must be specified as fixed elements in $\mathbb{K}$. Instead, `DiffAlg.m2` treats homogeneous forms and vector fields in a completely symbolic environment by considering the scalar coefficient rings

$$\mathbb{K}[a_{\alpha,I}] \quad \text{and} \quad \mathbb{K}[b_{\beta,i}].$$

Scalar coefficients can be systematically obtained by looking at the coordinates of differential forms and vector fields written in the standard bases

$$\mathcal{B}_{r,d} = \{x^\alpha dx_I\}_{\substack{\#I=r \\ |\alpha|=d}} \quad \text{and} \quad \mathcal{B}_e = \left\{x^\beta \tfrac{\partial}{\partial x_i}\right\}_{|\beta|=e}$$

of the spaces $\Omega^r(d)$ and $T(e)$, respectively.

Importantly, when using `DiffAlg.m2`, each object is expected to be defined in its own coefficient ring. Then, certain operations, such as contraction or computing the wedge product, involve different input and output rings, producing a modification of the coefficients rings $\mathbb{K}[a_{\alpha,I}]$ or $\mathbb{K}[b_{\beta,i}]$. For greater clarity, consider the following example. Fix $\omega \in \Omega^r(d)$ and $X \in T(e)$ and consider the contraction $i_X\omega \in \Omega^{r-1}(d+e)$. Then, the following will be taking place:

	(ω, X)	$\mapsto$	$i_X\omega$
Ring	$\mathbb{K}[a_{\alpha,I}] \times \mathbb{K}[b_{\beta,i}]$		$\mathbb{K}[a_{\alpha,I}, b_{\beta,i}]$
Basis	$\mathcal{B}_{r,d} \times \mathcal{B}_e$		$\mathcal{B}_{r-1,d+e}$

As mentioned before, the main purpose of `DiffAlg.m2` is to find algebraic solutions to equations in the context of differential algebra. Equations are treated differently in the linear and nonlinear cases:

(a) In the linear case, for example $i_R\omega = 0$, `DiffAlg.m2` can compute a basis of the solutions of the equation. Once this is done, it can also compute a generic linear combination of the elements of the basis; see Example 1.

(b) In the nonlinear case, for example $\omega \wedge d\omega = 0$, the coordinates will be polynomial. In this case, `DiffAlg.m2` would compute the ideal generating the space of solutions. This ideal can be obtained in two different ways: taking coordinates in the basis $\mathcal{B}_{r,d}$ or $\mathcal{B}_e$, or taking coordinates in the basis $\{dx_I\}$ or $\left\{\frac{\partial}{\partial x_i}\right\}$; see Examples 2 and 4.

`DiffAlg.m2` can also be a valuable tool for studying differential operators. The lack of algebraic theory to deal with such objects can be mitigated by nonconclusive computations easily made by `DiffAlg.m2`. As a first example, one could consider computing solutions of a differential operator degree-wise for low degrees.

SOME EXAMPLES.

Example 1. In the following example we obtain a basis of the space of projective differential 2-forms in $\mathbb{P}^3_{\mathbb{K}}$. Then, we define a generic projective differential 2-form to be possibly used in further computations.

```
i1 : loadPackage "DiffAlg";

i2 : R = radial 3

o2 = x ax  + x ax  + x ax  + x ax
      0  0    1  1    2  2    3  3

o2 : DiffAlgField

i3 : w = newForm(3,2,1,"a");

o3 = (a x +a x +a  x +a  x )dx dx +(a x  + a x  + a  x  + a  x )dx dx
       0 0  6 1  12 2  18 3   0  1   1 0    7 1    13 2    19 3   0  2

     + (a x  + a x  + a  x  + a  x )dx dx  + (a x  + a x  + a  x  +
        3 0    9 1    15 2    21 3   1  2     2 0    8 1    14 2

     a  x )dx dx  + (a x  + a  x  + a  x  + a  x )dx dx +(a x +a  x  +
      20 3   0  3     4 0    10 1    16 2    22 3   1  3   5 0  11 1

     a  x  + a  x )dx dx
      17 2    23 3   2  3

o3 : DiffAlgForm

i4 : pretty ring w

       QQ[i]
o4 = ------[][a , a , a , a , a , a , a , a , a , a , a  , a  , a  ,
      2        0   1   2   3   4   5   6   7   8   9   10   11   12
     i  + 1

     a  , a  , a  , a  , a  , a  , a  , a  , a  , a  , a  ][x , x ,
      13   14   15   16   17   18   19   20   21   22   23   0   1

     x , x ][dx , dx , dx , dx ]
      2   3    0    1    2    3

i5 : K = genKer (R _ w, w);

i6 : length K

o6 = 4

i7 : v = linearComb(K,"a")

o7 = (a x  - a x )dx dx +(- a x  + a x )dx dx  + (a x  + a x )dx dx +
       0 2    1 3   0  1     0 1    2 3   0  2     0 0    3 3   1  2

     (a x  - a x )dx dx  + (-a x  - a x )dx dx  + (a x  + a x )dx dx
       1 1    2 2   0  3      1 0    3 2   1  3     2 0    3 1   2  3

o7 : DiffAlgForm

i8 : pretty ring v

       QQ[i]
o8 = ------[][a , a , a , a ][x , x , x , x ][dx , dx , dx , dx ]
      2        0   1   2   3   0   1   2   3    0    1    2    3
     i  + 1
```

Let us explain part of the code:

i2. Creates the radial vector field in four variables. We are denoting the basic field $\partial/\partial x_i$ as ax_i.

i3. Creates a generic linear 2-form in $\Omega^2_{\mathbb{K}^4}(1)$, with the coefficients indexed as a_i.

i4. Shows the ring of definition of w.

i5. Gets a basis (as a Macaulay2 list) of forms in $\Omega^2_{\mathbb{K}^4}(1)$ that descend to projective space. The operation R_w computes the contraction of the differential form w with the vector field R.

i6. Gets the dimension of $\Omega^2_{\mathbb{P}^3}(1)$ in projective 3-space.

i7. Defines a generic projective form with coefficients a_i.

i8. Shows the ring of definition of v.

Example 2. In the finite-dimensional $\mathbb{K}$-vector space $\Omega^1(d)$, the solutions of the equation $\omega \wedge d\omega = 0$ determine an algebraic variety; its points are the integrable differential 1-forms of degree d. In the following example, we compute the equations of the variety of integrable 1-forms of degree 1 in 3-dimensional space.

It is worth mentioning that, for $n \geq 3$ and $d > 5$, it is an open problem to classify the irreducible components of this varieties; see [Cukierman et al. 2009].

```
i1 : loadPackage "DiffAlg";

i2 : w = newForm (2,1,1,"a")

o2 = (a x + a x  + a x )dx  + (a x  + a x  + a x )dx +(a x  + a x + a x )dx
      0 0   3 1    6 2   0     1 0    4 1    7 2   1   2 0    5 1   8 2   2

o2 : DiffAlgForm

i3 : moduliIdeal (w ^ (diff w))

o3 = ideal (- a a  + a a  + a a  - a a , - a a  + a a  + a a  - a a ,
              2 3    0 5    1 6    0 7    2 4    1 5    4 6    3 7

           a a  - a a  + a a  - a a )
            5 6    2 7    1 8    3 8

               QQ[i]
o3 : Ideal of ------[][a , a , a , a , a , a , a , a , a ]
               2       0   1   2   3   4   5   6   7   8
              i  + 1
```

About the code:

i2. Creates a generic linear 1-form in $\Omega^1_{\mathbb{K}^3}(1)$, with coefficients a_i.

i3. Returns the ideal of the scalar coefficients given by $w \wedge dw = 0$.

Example 3. Let D be a 2-dimensional distribution generated by vector fields X and Y in 5-dimensional space. In the following example we compute the ranks of the derived distributions $D^{(p)}$. We verify that this derived series eventually spans

the entire tangent space. A distribution D satisfying this condition is called *bracket-generating*.

```
i2 : X = newField "x_0^2*ax_0+x_1^2*ax_1+x_2^2*ax_2+x_3^2*ax_3";

i3 : Y = newField "x_5*ax_0+x_4*ax_1+x_3*ax_2+x_2*ax_3+x_1*ax_4+x_0*ax_5";

i4 : D_0 = {X,Y};

i5 : for b in 1..3 do (
      for a in D_(b-1) do (
       D_b = join(D_(b-1),{a|Y,a|X})
      )
     );

i6 : {rank dist D_0, rank dist D_1, rank dist D_2, rank dist D_3}

o6 = 2, 3, 5, 6

o6 : List
```

About the code:

`i5.` Computes the derived sequence.

`i6.` Prints the ranks of the derived series.

Example 4. In the following example, we generate a random rational 1-form of type (1,2) in $\mathbb{P}^2_{\mathbb{K}}$. First, we compute (the dimension of) the space of its integrating factors; see [Suwa 1995, pp. 828–829]. Then, we compute the ideal of its singular locus (the ideal where it vanishes).

```
i2 : w = random logarithmicForm (2,{1,2},"a",Projective => true);

i3 : f = newForm (2,0,3,"a");

i4 : length genKer(w^(diff f) + f*(diff w), f)

o4 = 2

i5 : I = singularIdeal w

                          2                                 2    2
o6 = ideal (- 9x x  + 63x  - 36x x  - 54x x  - 54x , 9x  - 63x x  -
               0 1      1      0 2      1 2      2    0      0 1

                  2      2               2
     27x x  -45x x  - 54x , 36x  + 81x x  + 45x  + 54x x  + 54x x )
        0 2     1 2      2     0      0 1      1      0 2      1 2

              QQ[i]
o6 : Ideal of ------[][x , x , x ]
              2        0   1   2
             i  + 1
```

SUPPLEMENT. The online supplement contains version 1.5 of `DiffAlg.m2`.

REFERENCES.

[Bryant et al. 1991] R. L. Bryant, S. S. Chern, R. B. Gardner, H. L. Goldschmidt, and P. A. Griffiths, *Exterior differential systems*, Mathematical Sciences Research Institute Publications **18**, Springer, 1991. MR Zbl

[Cukierman et al. 2009] F. Cukierman, J. V. Pereira, and I. Vainsencher, "Stability of foliations induced by rational maps", *Ann. Fac. Sci. Toulouse Math.* (6) **18**:4 (2009), 685–715. MR Zbl

[Hartshorne 1977] R. Hartshorne, *Algebraic geometry*, Graduate Texts in Mathematics **52**, Springer, 1977. MR Zbl

[Suwa 1995] T. Suwa, "Unfoldings of codimension one complex analytic foliation singularities", pp. 817–865 in *Singularity theory* ((Trieste, 1991)), edited by D. T. Lê et al., World Sci. Publ., River Edge, NJ, 1995. MR Zbl

[Tanaka 1970] N. Tanaka, "On differential systems, graded Lie algebras and pseudogroups", *J. Math. Kyoto Univ.* **10** (1970), 1–82. MR

RECEIVED: 27 May 2015 REVISED: 26 Oct 2018 ACCEPTED: 19 Nov 2018

MANUEL DUBINSKY:

manudubinsky@gmail.com
Departamento de Computación, Universidad de Avellaneda, Avellaneda, Argentina

CÉSAR MASSRI:

cmassri@caece.edu.ar
Departamento de Matemática, Universidad CAECE, Buenos Aires, Argentina

ARIEL MOLINUEVO:

amoli@dm.uba.ar
Instituto de Matemática, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil

FEDERICO QUALLBRUNN:

fquallb@dm.uba.ar
Departamento de Matemática, Universidad de Buenos Aires, Pabellón I, Ciudad Universitaria, Buenos Aires, Argentina

JSAG **9** (2019), 19–27
dx.doi.org/10.2140/jsag.2019.9.19

The Journal of Software for
Algebra and Geometry

Matroids: a Macaulay2 package

JUSTIN CHEN

ABSTRACT: We give an overview of the Macaulay2 package `Matroids.m2`, which introduces functionality to create and compute with matroids into Macaulay2. Examples highlighting the use of many functions in the package are provided, including applications of matroids to other areas.

INTRODUCTION. A matroid is a combinatorial object which abstracts the notions of (linear algebraic, graph-theoretic) independence. Since their introduction by Whitney [1935], matroids have found diverse applications in combinatorics, graph theory, optimization, and algebraic geometry, in addition to being studied as interesting objects in their own right.

We describe here the Macaulay2 package `Matroids.m2`, which is available at https://github.com/jchen419/Matroids-M2. For the reader already familiar with matroids, it provides capabilities to form matroids from a matrix, graph, or ideal; convert between various representations of matroids; create and detect existence of minors; compute Tutte polynomials and Chow rings; as well as applications of matroids to polyhedral and algebraic geometry, commutative algebra, optimization, and even group theory. Each will in turn be illustrated with examples. Virtually all notation and results mentioned below can be found in [Oxley 2011].

One striking feature of matroids is the multitude of distinct ways to define them. This variety of equivalent — or *cryptomorphic* — ways to characterize matroids is a great strength of matroid theory, and one of the reasons for its ubiquity. From the perspective of this package, the key definition is via bases:

Definition. Let E be a finite set, and $\varnothing \neq \mathcal{B} \subseteq 2^E$ a set of subsets of E. The pair $(E, \mathcal{B})$ is a *matroid* if for any $B_1, B_2 \in \mathcal{B}$ and $b_1 \in B_1 \setminus B_2$, there exists $b_2 \in B_2 \setminus B_1$ with $B_1 \setminus \{b_1\} \cup \{b_2\} \in \mathcal{B}$.

The set E is called the *ground set* of the matroid $M = (E, \mathcal{B})$, and $\mathcal{B}$ is the set of *bases* of M. All bases have the same cardinality, called the *rank* of M. Any subset of a basis is an *independent* set. A subset of E that is not independent is *dependent*.

MSC2010: primary 05-04, 05B35; secondary 05C31, 52B40.
Keywords: matroids, circuits, Tutte polynomial.
`Matroids.m2` version 0.9.7

The minimal (with respect to inclusion) dependent sets are *circuits*. It is easy to see that any of bases, independent sets, dependent sets, and circuits determines the others.

As any subset of an independent set is independent, the set of independent sets of a matroid forms a simplicial complex on E, called the *independence complex* of M, denoted by Δ_M. Via Stanley–Reisner theory, Δ_M corresponds to a squarefree monomial ideal $I_{\Delta_M} := \langle \prod_{i \in C} x_i \mid C \text{ circuit} \rangle$, inside a polynomial ring $k[x_i \mid i \in E]$ (since faces of Δ_M are independent sets, the minimal nonfaces are precisely the minimal dependent sets, i.e., circuits). We call I_{Δ_M} the (circuit) ideal of M: internally, many algorithms in this package work directly with this ideal, to exploit Macaulay2's facility with monomial ideals.

A FIRST EXAMPLE. The most basic way to create a matroid is by specifying the ground set and list of bases:

```
i1 : needsPackage "Matroids";
i2 : M = matroid({a,b,c,d},{{a,b},{a,c}})
o2 = a matroid of rank 2 on 4 elements
o2 : Matroid
```

This creates a matroid of rank 2 on the ground set $\{a, b, c, d\}$ with two bases. We can peek at the matroid to see more of its internal structure:

```
i3 : peek M
o3 = Matroid{bases => {set {0, 1}, set {0, 2}}}
             cache => CacheTable{...2...}
             groundSet => set {0, 1, 2, 3}
             rank => 2
```

Two things should be noticed: first, `groundSet` is a set of integers $\{0, \ldots, 3\}$ (instead of the given list $\{a, b, c, d\}$). Second, the bases consist of a list of subsets of `groundSet`. This convention is by design: internally, the ground set is always identified with the set $\{0, \ldots, |E| - 1\}$, and all sets associated to the structure of the matroid are subsets of the ground set. One should think of the integers in `groundSet` as *indices* of the actual elements, so 0 is the index of the first element (in this case a), 1 is the index of the second element, etc.

The actual elements of the user-inputted ground set are not lost though; they have been cached in the `CacheTable`, and can be accessed by using indices as subscripts on M, or all at once with an asterisk:

```
i4 : (M_3, M_{0,1}, M_(set{1,2}), M_*)
o4 = (d, {a, b}, {b, c}, {a, b, c, d})
```

So far, no attempt has been made to check that M is actually a matroid. We verify this now using the method `isWellDefined` (which internally checks the circuit elimination axiom), and also give a nonexample.

```
i5 : (isWellDefined M, isWellDefined matroid({a,b,c,d},{{a,b},{c,d}}))
o5 = (true, false)
```

We can obtain plenty of matroid-theoretic information for this example. Recall:

Definition. A *loop* in M is a 1-element circuit, and a *coloop* in M is an element contained in every basis. For $A \subseteq E$, the *rank* of A is the size of the largest independent subset of A, and the *closure* of A is $\bar{A} := \{x \in E \mid \text{rank}(A) = \text{rank}(A \cup \{x\})\}$. A *flat* of M is a closed subset, i.e., $A = \bar{A}$. A *hyperplane* of M is a flat of rank equal to rank $M - 1$.

```
i6 : (rank M, rank(M, set{0,3}))
o6 = (2, 1)
i7 : (circuits M, independentSets(M, 1))
o7 = ({set {1, 2}, set {3}}, {set {0}, set {1}, set {2}})
i8 : (loops M, coloops M, closure(M, set{2,3}), hyperplanes M)
o8 = ({3}, {0}, set {1, 2, 3}, {set {0, 3}, set {1, 2, 3}})
i9 : flats M -- sorted by increasing size
o9 = {set {3}, set {0, 3}, set {1, 2, 3}, set {0, 1, 2, 3}}
i10 : fVector M -- number of flats of rank i, for 0 <= i <= rank M
o10 = HashTable{0 => 1}
                1 => 2
                2 => 1
```

CONSTRUCTING TYPES OF MATROIDS. The simplest family of matroids is the family of *uniform* matroids, where the set of bases equals all subsets of a fixed size:

```
i11 : U = uniformMatroid(2,4); bases U
o12 = {set {0, 1}, set {0, 2}, set {1, 2}, set {0, 3}, set {1, 3}, set {2, 3}}
```

Another family of fundamental importance is the class of *linear* matroids, which arise naturally from a matrix. The columns of the matrix form the ground set, and a set of column vectors is declared independent if they are linearly independent in the vector space spanned by the columns.

```
i13 : A = matrix{{0,4,-1,6},{0,2/3,7,1}},; MA = matroid A; representationOf MA
o15 = | 0 4   -1 6 |
      | 0 2/3 7  1 |
```

An abstract matroid M is called *representable* or *realizable* over a field k if M is *isomorphic* to a linear matroid over k, where an *isomorphism* of matroids is a bijection between ground sets that induces a bijection on bases. We verify that the matroid M we started with is isomorphic to MA, hence is representable over $\mathbb{Q}$:

```
i16 : areIsomorphic(M, MA)
o16 = true
```

A matroid can also be constructed by specifying its circuit ideal, which we do for the same M above. Here two matroids are considered equal if they have the same set of bases and same size ground sets; or, equivalently, the identity permutation is an isomorphism between them.

```
i17 : R = QQ[x,y,z,w]; MI = matroid ideal(y*z, w)
o18 = a matroid of rank 2 on 4 elements
i19 : M == MI
o19 = true
```

An important class of representable matroids (over any field) is the class of *graphic* matroids, derived from a graph. If G is an (undirected) graph, then the graphic matroid $M(G)$ has ground set equal to the edge set of G, and circuits given by cycles in G, including loops and parallel edges.

```
i20 : K5 = completeGraph 5; M5 = matroid K5
o21 = a matroid of rank 4 on 10 elements
i22 : #bases M5 -- == n^(n-2) for M(K_n), by Cayley's theorem
o22 = 125
```

In this package, the graphic matroid is created by specifying circuits. This can be done for an abstract matroid as well, using the optional argument `EntryMode => "circuits"` in the constructor function. Regardless of the value of `EntryMode`, the bases are automatically computed upon creation. We recreate the matroid M from before, by specifying its circuits (note the similarity with specifying the circuit ideal):

```
i23 : M == matroid({a,b,c,d},{{b,c},{d}}, EntryMode => "circuits")
o23 = true
```

Certain common matroids are close to uniform, in the sense that relatively few subsets of size rank M are dependent, so the set of *nonbases* (= dependent sets of size rank M) can also be specified:

```
i24 : nb = {{0,2,4},{1,3,4},{1,2,5},{0,3,5},{0,1,6},{2,3,6},{4,5,6}}/set;
i25 : F7 = matroid(toList(0..6), nb, EntryMode => "nonbases")
o25 : a matroid of rank 3 on 7 elements
i26 : (#bases F7, #circuits F7)
o26 = (28, 14)
```

A few specific matroids of theoretical importance are also built-in. Currently these are F_7, F_7^-, V_8, V_8^+, $AG(3,2)$, R_{10}, and the Pappus and non-Pappus matroids. A library of all matroids on up to eight elements is included as well:

```
i27 : F7 == specificMatroid "fano"
o27 = true
i28 : L7 = allMatroids 7 -- non-isomorphic matroids on 7 elements
o28 = {a matroid of rank 0 on 7 elements, a matroid of rank 1 on 7 elements, ...
i29 : (#L7, #flatten apply(6, allMatroids))
o29 = (306, 70)
```

One can also construct a new matroid from smaller ones by taking *direct sums*: if $M_1 = (E_1, \mathcal{B}_1)$, $M_2 = (E_2, \mathcal{B}_2)$ are matroids, then their direct sum is

$$M_1 \oplus M_2 := (E_1 \sqcup E_2, \{B_1 \sqcup B_2 \mid B_1 \in \mathcal{B}_1, B_2 \in \mathcal{B}_2\}).$$

A matroid that cannot be written as a direct sum of nonempty matroids is called *connected*. Every matroid is a direct sum of connected matroids, its *connected components*, which are unique up to rearrangement:

```
i30 : S = U ++ matroid completeGraph 3
o30 = a matroid of rank 4 on 7 elements
```

```
i31 : C = components S
o31 = {a matroid of rank 2 on 4 elements, a matroid of rank 2 on 3 elements}
i32 : S == C#0 ++ C#1 and C#0 == U and C#1 == matroid completeGraph 3
o32 = true
```

DUALITY AND MINORS. One of the most important features of matroid theory is the existence of a duality. It is straightforward to check that if $M = (E, \mathcal{B})$ is a matroid, then $\{E \setminus B \mid B \in \mathcal{B}\}$ is the set of bases of a matroid on E, called the *dual matroid* of M, denoted by M^*.

```
i33 : D = dual M; (bases M, bases D)
o34 = ({set {0, 1}, set {0, 2}}, {set {2, 3}, set {1, 3}})
i35 : M == dual D
o35 = true
```

Virtually any matroid-theoretic property or operation can be enriched by considering its dual version — for instance, loops of M^* are coloops of M, and circuits of M^* are complements of hyperplanes of M (this is in fact how the method `hyperplanes` works). Another operation is deletion, which dualizes to contraction:

Definition. Let $M = (E, \mathcal{B})$ be a matroid, and $S \subseteq E$. The *restriction* of M to S, denoted $M|_S$, is the matroid on S with bases $\{B \cap S \mid B \in \mathcal{B}, |B \cap S| = \operatorname{rank} S\}$. The *deletion* of S, denoted $M \setminus S$, is the restriction of M to $E \setminus S$. The *contraction* of M by S, denoted M/S, is defined as $(M^* \setminus S)^*$.

```
i36 : N1 = M \ set{3}; (N1_*, bases N1)
o37 = ({a, b, c}, {set {0, 1}, set {0, 2}})
i38 : N2 = M / set{1}; (N2_*, bases N2)
o39 = ({a, c, d}, {set {0}})
```

A *minor* of M is any matroid which can be obtained from M by a sequence of deletions and contractions. It is a fact that any minor of M is of the form $(M/X) \setminus Y$ for disjoint subsets $X, Y \subseteq E$.

```
i40 : minorM5 = minor(M5, set{9}, set{3,5,8}) -- contracts {9}, then deletes {3,5,8}
o40 = a matroid of rank 3 on 6 elements
i41 : (minorM5_*, #bases minorM5)
o41 = ({set {0, 1}, set {0, 2}, set {0, 3}, set {1, 2}, set {1, 4}, set {2, 3}}, 16)
```

Minors can be used to describe many important classes of matroids. For example, a class $\mathcal{M}$ of matroids is said to be *minor-closed* if every minor of a matroid in $\mathcal{M}$ is again in $\mathcal{M}$. The classes of uniform, k-representable (for any field k), and graphic matroids are all minor-closed. Various classes of matroids can be characterized by their *forbidden* or *excluded* minors: namely the matroids not in the class, but with every proper minor in the class.

Theorem 1 (Tutte 1958a; 1958b; 1959). *Let M be a matroid. We denote by $U_{2,4}$ the uniform matroid of rank 2 on 4 elements, and by F_7 the Fano matroid.*

(i) *M is binary (= representable over $\mathbb{F}_2$) if and only if M has no $U_{2,4}$ minor (i.e., no minor of M is isomorphic to $U_{2,4}$).*

(ii) *M is regular (= representable over any field) if and only if M has no $U_{2,4}$, F_7, or F_7^* minor.*

(iii) *M is graphic if and only if M has no $U_{2,4}$, F_7, F_7^*, $M(K_5)^*$, or $M(K_{3,3})^*$ minor.*

We illustrate this by verifying that $M(K_5)$ is regular (alternatively, note that for any graph G, the signed incidence matrix of any orientation of G represents $M(G)$ over any field):

```
i42 : any({U, F7, dual F7}, forbidden -> hasMinor(M5, forbidden))
o42 = false
```

Every minor of M is in fact of the form $(M/I) \setminus I^*$, where I, I^* are disjoint, I is independent, and I^* is *coindependent* (= independent in M^*). Such a minor has rank equal to that of M/I, which is equal to rank $M - |I|$. Thus checking existence of a minor N in M can be realized as a two-step process, where the first step contracts independent sets of M of a fixed size down to the rank of N, and the second step deletes coindependent sets down to the size of N.

```
i43 : M4 = matroid completeGraph 4; hasMinor(M5, M4)
o44 = true
i45 : minorM5 == M4
o45 = true
```

Finally, the *Tutte polynomial* $T_M(x, y)$ of a matroid is an invariant which is universal with respect to satisfying a *deletion-contraction recurrence*. It is a bivariate polynomial over $\mathbb{Z}$ which can be defined by the relation

$$T_M(x, y) = T_{M\setminus e}(x, y) + T_{M/e}(x, y), \quad e \in E \text{ not a loop or coloop},$$

with the initial condition $T_M(x, y) = x^a y^b$ if M consists of a coloops and b loops. Any numerical invariant of matroids which satisfies a (weighted) deletion-contraction recurrence is an evaluation of the Tutte polynomial, up to a scale factor. For instance, the number of bases is equal to $T_M(1, 1)$:

```
i46 : tuttePolynomial M5
       6     5    4       3      4     3      2        2      3      2        2
o46 = y  + 4y  + x  + 5x*y  + 10y  + 6x  + 10x y + 15x*y  + 15y  + 11x  + 20x*y + 15y ···
i47 : tutteEvaluate(M5, 1, 1)
o47 = 125
```

For graphic matroids, the Tutte polynomial contains a wealth of information about the graph; e.g., the Tutte polynomial specializes to the chromatic polynomial. Even evaluations at specific points contain nontrivial information: e.g., $T_{M(G)}(2, 1)$ counts the number of spanning forests in G, and $T_{M(G)}(2, 0)$ counts the number of acyclic orientations of G.

```
i48 : (tutteEvaluate(M5, 2, 1), tutteEvaluate(M5, 2, 0), factor chromaticPolynomial K5)
o48 = (291, 120, (x)(x - 4)(x - 3)(x - 2)(x - 1))
```

CONNECTIONS. We now present some connections of matroids to other areas of mathematics. First, polyhedral geometry: let $M = ([n], \mathcal{B})$ be a matroid on $\{1, \ldots, n\}$. In Euclidean space $\mathbb{R}^n$ with standard basis $\{e_1, \ldots, e_n\}$, define the matroid polytope P_M by taking the convex hull of the indicator vectors of the bases of M:

$$P_M := \mathrm{conv}\left(\sum_{i \in B} e_i \mid B \in \mathcal{B}\right).$$

The matroid polytope can be created as follows:

```
i49 : needsPackage "Polyhedra"; P = convexHull basisIndicatorMatrix M4
o50 = {ambient dimension => 6           }
       dimension of lineality space => 0
       dimension of polyhedron => 5
       number of facets => 16
       number of rays => 0
       number of vertices => 16
o50 : Polyhedron
```

A theorem of Gelfand, Goresky, MacPherson, and Serganova [Gelfand et al. 1987] classifies the subsets $\mathcal{B} \subseteq 2^{[n]}$ which are the bases of a matroid on $[n]$ in terms of the polytope P_M.

Next is optimization: let E be a finite set, and $\mathcal{I} \subseteq 2^E$ a set of subsets that is downward closed: if $X \in \mathcal{I}$ and $Y \subseteq X$, then $Y \in \mathcal{I}$. Let w be a weight function on E, i.e., a function $w : E \to \mathbb{R}$, extended to $w : 2^E \to \mathbb{R}$ by setting $w(X) := \sum_{x \in X} w(x)$. Consider the optimization problem $(*)$ of finding a maximal member of $\mathcal{I}$ of maximum weight, with respect to w. One attempt to solve $(*)$ is to apply the greedy algorithm: namely, after having already selected elements $\{x_1, \ldots, x_i\}$, choose an element $x_{i+1} \in E$ of maximum weight such that $\{x_1, \ldots, x_i, x_{i+1}\} \in \mathcal{I}$, and repeat. It turns out that the greedy algorithm will work if and only if $\mathcal{I}$ is the set of independent sets of a matroid:

Theorem 2 [Borůvka 1926]. *Let E be a finite set, and $\mathcal{I} \subseteq 2^E$. Then $\mathcal{I}$ is the set of independent sets of a matroid on E if and only if $\mathcal{I}$ is downward closed and for all weight functions $w : E \to \mathbb{R}$, the greedy algorithm successfully solves $(*)$.*

A solution to $(*)$ provided by the greedy algorithm can be obtained using the method `maxWeightBasis` (the weight function w is specified by its list of values on E):

```
i51 : w = {0, log(2), 4/3, 1, -4, 2, pi_RR}; maxWeightBasis(F7, w)
o52 = set {3, 5, 6}
```

Another application to optimization comes from the operation of *matroid union*: if M_1, M_2 are matroids with independent sets $\mathcal{I}_1, \mathcal{I}_2$, then the independent sets of the union are of the form $I_1 \cup I_2$, where $I_1 \in \mathcal{I}_1$, $I_2 \in \mathcal{I}_2$ (and thus coincides with the direct sum if the ground sets are disjoint).

```
i53 : matroid({a,b,c,d}, {{a},{b},{c}}) + matroid({a,b,c,d}, {{b},{c},{d}}) == U
o53 : true
i54 : F7 + F7 == uniformMatroid(6, 7)
o54 : true
```

Matroid union is an important operation in combinatorial optimization, and is closely related to transversal and matching problems: a matroid is *transversal* if and only if it is a union of rank 1 matroids, and *gammoids* (a class of matroids defined from vertex paths in directed graphs) are the minor-closure of the transversal matroids.

One can also find connections to group theory via the method `getIsos`, which computes all isomorphisms between two matroids. Many interesting groups can be realized as automorphism groups of small matroids:

```
i55 : aut = getIsos(F7, F7)
o55 : {{0, 1, 2, 3, 4, 5, 6}, {1, 0, 2, 3, 4, 6, 5}, {0, 2, 1, 3, 5, 4, 6}, {2, 0, 1, ...
i56 : #aut
o56 : 168
```

The above output is an explicit permutation representation of $\mathrm{Aut}(\mathbb{P}^2_{\mathbb{F}_2}) = \mathrm{PGL}(3, \mathbb{F}_2)$ as a subgroup of S_7. For a larger example, the automorphism group of the Steiner system $S(5, 6, 12)$ is the Mathieu group M_{12}, a sporadic simple group of order $95040 = 2^6 \cdot 3^3 \cdot 5 \cdot 11$. This in turn is also equal to the automorphism group of the realizable matroid associated to a particular 6×12 matrix over $\mathbb{F}_3$ ([Oxley 2011], p. 367), and a high-performance computing cluster took just under 2 hours to compute the entire permutation representation of this group inside S_{12}.

For an application to commutative algebra: matroids are closely related to the Cohen–Macaulay property, for symbolic powers of squarefree monomial ideals. Indeed, from [Terai and Trung 2012] we know that if I is a squarefree monomial ideal, then I is the circuit ideal of a matroid if and only if every symbolic power $I^{(n)}$ is Cohen–Macaulay, for $n \geq 1$ (in fact, this is equivalent to requiring just $I^{(3)}$ to be Cohen–Macaulay). As one can quickly check whether an ideal is the ideal of a matroid, this can give a quick proof that a particular symbolic power is Cohen–Macaulay:

```
i57 : M6 = matroid completeGraph 6; L = (irreducibleDecomposition ideal M6)/(P -> P^3);
i59 : try ( alarm 10; I3 = intersect L; ) -- doesn't finish in 10 seconds
i60 : time isWellDefined M6
     -- used 0.359306 seconds
o60 : true
```

Last but not least is algebraic geometry; in particular the emerging field of combinatorial Hodge theory. For a matroid M on ground set E with no loops, one can define a Chow ring associated to M: for a field k, set

$$R := k[x_F \mid F \text{ proper, nonempty flat}]/(I_1 + I_2),$$
$$I_1 := \left(\sum_{i_1 \in F} x_F - \sum_{i_2 \in F} x_F \;\middle|\; i_1, i_2 \in E \text{ distinct} \right),$$
$$I_2 := (x_F x_{F'} \mid F, F' \text{ incomparable}),$$

where F, F' run over all nonempty proper flats of M. Then R is a standard graded Artinian k-algebra of Castelnuovo–Mumford regularity $r := \operatorname{rank} M - 1$. A result of Adiprasito, Katz, and Huh [Adiprasito et al. 2018] states that R is a Poincaré duality algebra (in particular, is Gorenstein) and has the strong Lefschetz property: for general $l \in R_1$ and $j \leq r/2$, multiplication by l^{r-2j} is an isomorphism $R_j \xrightarrow{\sim} R_{r-j}$. We illustrate the Gorenstein property for the Vamos matroid (which is a smallest matroid not realizable over any field), and conclude by computing the dual socle generator or *volume polynomial* (which generates the Macaulay inverse system of R) for $M(K_4)$:

```
i61 : V = specificMatroid("vamos"); (rank V, #V.groundSet, #bases V, #flats V)
o62 = (4, 8, 65, 79)
i63 : I = idealChowRing V; apply(0..<rank V, i -> hilbertFunction(i, I))
o63 : Ideal of QQ[x   , x   , x   , x   , x   , x   , x   , x   , x      , x      , ...
                   {7}   {6}   {5}   {4}   {3}   {0}   {2}   {1}  {6, 7}   {5, 7}
o64 = (1, 70, 70, 1)
i65 : cogeneratorChowRing M4
            2       2       2       2       2       2
o65 = 2t     + 2t     + 2t     + 2t     + 2t     + 2t     - 2t   t        - 2t   t       + ...
        {5}      {4}      {3}      {2}      {1}      {0}     {5}  {0, 5}      {0} {0, 5}
```

ACKNOWLEDGEMENTS. This project was partially supported by grant DMS-1001867 from the NSF. The author is grateful to June Huh for explaining the connection to the Chow ring of a matroid. The author would also like to thank David Eisenbud and Daniel Grayson for advice with Macaulay2, Joe Kileel for enlightening discussions, Aaron Dall for software testing, Chris Eur for suggesting many valuable improvements, and the referee for helpful comments.

SUPPLEMENT. The online supplement contains version 0.9.7 of `Matroids.m2`.

REFERENCES.

[Adiprasito et al. 2018] K. Adiprasito, J. Huh, and E. Katz, "Hodge theory for combinatorial geometries", *Ann. of Math.* (2) **188**:2 (2018), 381–452. MR Zbl

[Borůvka 1926] O. Borůvka, "O jistém problému minimálním", *Práce moravské přirodovědecké společnosti* **3**:3 (1926), 37–58.

[Gelfand et al. 1987] I. M. Gelfand, R. M. Goresky, R. D. MacPherson, and V. V. Serganova, "Combinatorial geometries, convex polyhedra, and Schubert cells", *Adv. in Math.* **63**:3 (1987), 301–316. MR

[Oxley 2011] J. Oxley, *Matroid theory*, 2nd ed., Oxford Graduate Texts in Mathematics **21**, Oxford University Press, 2011. MR Zbl

[Terai and Trung 2012] N. Terai and N. V. Trung, "Cohen–Macaulayness of large powers of Stanley–Reisner ideals", *Adv. Math.* **229**:2 (2012), 711–730. MR Zbl

[Tutte 1958a] W. T. Tutte, "A homotopy theorem for matroids, I", *Trans. Amer. Math. Soc.* **88** (1958), 144–160. MR Zbl

[Tutte 1958b] W. T. Tutte, "A homotopy theorem for matroids, II", *Trans. Amer. Math. Soc.* **88** (1958), 161–174. MR Zbl

[Tutte 1959] W. T. Tutte, "Matroids and graphs", *Trans. Amer. Math. Soc.* **90** (1959), 527–552. MR Zbl

[Whitney 1935] H. Whitney, "On the abstract properties of linear dependence", *Amer. J. Math.* **57**:3 (1935), 509–533. MR Zbl

RECEIVED: 6 Sep 2015 REVISED: 26 Sep 2018 ACCEPTED: 27 Sep 2018

JUSTIN CHEN: Department of Mathematics, Georgia Institute of Technology, Atlanta, GA, United States

jchen646@gatech.edu

JSAG **9** (2019), 29–34
dx.doi.org/10.2140/jsag.2019.9.29

The Journal of Software for
Algebra and Geometry

Computing quasidegrees of A-graded modules

ROBERTO BARRERA

ABSTRACT: We describe the main functions of the Macaulay2 package `Quasidegrees.m2`. The purpose of this package is to compute the quasidegree set of a finitely generated $\mathbb{Z}^d$-graded module presented as the cokernel of a monomial matrix. We provide examples with motivation coming from A-hypergeometric systems.

1. INTRODUCTION. Throughout, $R = \Bbbk[x_1, \ldots, x_n]$ is a $\mathbb{Z}^d$-graded polynomial ring over a field $\Bbbk$ and $\mathfrak{m} = \langle x_1, \ldots, x_n \rangle$ denotes the homogeneous maximal ideal in R. Let $M = \bigoplus_{\beta \in \mathbb{Z}^d} M_\beta$ be a $\mathbb{Z}^d$-graded R-module. The *true degree set* of M is

$$\operatorname{tdeg}(M) = \{\beta \in \mathbb{Z}^d \mid M_\beta \neq 0\}.$$

The *quasidegree set* of M, denoted $\operatorname{qdeg}(M)$, is the Zariski closure in $\mathbb{C}^d$ of $\operatorname{tdeg}(M)$.

The purpose of the Macaulay2 package `Quasidegrees.m2` (provided as an online supplement) is to compute the quasidegree set of a finitely generated $\mathbb{Z}^d$-graded module presented as the cokernel of a monomial matrix. By a monomial matrix, we mean a matrix where each entry is either zero or a monomial in R. The initial motivation for `Quasidegrees.m2` was to compute the quasidegree sets of certain local cohomology modules supported at $\mathfrak{m}$ of $\mathbb{Z}^d$-graded R-modules, so there are some methods in the package specific to local cohomology. Recall that the *i-th local cohomology module* of M with support at the ideal $I \subset R$ is the i-th right derived functor of the left exact I-torsion functor

$$\Gamma_I(M) = \{m \in M \mid I^t m = 0 \text{ for some } t \in \mathbb{N}\}$$

on the category of R-modules.

By the vanishing theorems of local cohomology [Eisenbud 1995], the quasidegree sets of the local cohomology modules supported at $\mathfrak{m}$ of M can be seen as measuring how far the module is from being Cohen–Macaulay. From the A-hypergeometric systems point of view, the quasidegree set of the non-top local cohomology modules supported at $\mathfrak{m}$ of R/I_A, where I_A is the toric ideal associated

MSC2010: 13D45, 13P20, 33C70.
Keywords: quasidegree, hypergeometric system, local cohomology.
`Quasidegrees.m2` version 1.0

to A in R, determine the parameters β where the A-hypergeometric system $H_A(\beta)$ has rank higher than expected (see Section 3).

2. QUASIDEGREES. The main function of `Quasidegrees.m2` is `quasidegrees`, which computes the quasidegree set of a module that is presented by a monomial matrix.

We use the idea of standard pairs of monomial ideals to compute the quasidegree set of a $\mathbb{Z}^d$-graded R-module. Given a monomial $x^{\boldsymbol{u}}$ and a subset $Z \subset \{x_1, \ldots, x_n\}$, the pair $(x^{\boldsymbol{u}}, Z)$ indexes the monomials $x^{\boldsymbol{u}} \cdot x^{\boldsymbol{v}}$ where $\operatorname{supp}(x^{\boldsymbol{v}}) \subset Z$. A *standard pair* of a monomial ideal $I \subset R$ is a pair $(x^{\boldsymbol{u}}, Z)$ satisfying:

(1) $\operatorname{supp}(x^{\boldsymbol{u}}) \cap Z = \varnothing$.

(2) All of the monomials indexed by $(x^{\boldsymbol{u}}, Z)$ are outside of I.

(3) $(x^{\boldsymbol{u}}, Z)$ is maximal in the sense that $(x^{\boldsymbol{u}}, Z) \nsubseteq (x^{\boldsymbol{v}}, Y)$ for any other pair $(x^{\boldsymbol{v}}, Y)$ satisfying the first two conditions.

To compute the quasidegree set of M we first find a monomial presentation of M so that M is the cokernel of a monomial matrix ϕ. We then compute the standard pairs of the ideals generated by the rows of ϕ and to each standard pair we associate the degrees of the corresponding variables. Algorithm 1 below is implemented in `Quasidegrees.m2`. The input is an R-module presented by a monomial matrix

$$\phi : R^s \to R^t.$$

As in Macaulay2, we write the degree of the k-th factor of R^t next to the k-th row of the matrix ϕ.

In the Macaulay2 implementation of the algorithm, we represent the output as a list of pairs $(\boldsymbol{u}, Z)$ with $\boldsymbol{u} \in \mathbb{Q}^d$ and $Z \subset \mathbb{Q}^d$, where the pair $(\boldsymbol{u}, Z)$ represents the plane

$$\boldsymbol{u} + \sum_{\boldsymbol{v} \in Z} \mathbb{C} \cdot \boldsymbol{v}.$$

Input: R-module M presented by monomial matrix $\phi = \alpha_i [c_{j,k} \boldsymbol{x}^{\boldsymbol{u}_{j,k}}] : R^s \to R^t$
Output: $\operatorname{qdeg}(M)$

$Q = \varnothing$
for $1 \le k \le t$ **do**
  $SP = \{\text{standard pairs of } \langle c_{k,1}\boldsymbol{x}^{\boldsymbol{u}_{k,1}}, c_{k,2}\boldsymbol{x}^{\boldsymbol{u}_{k,2}}, \ldots, c_{k,s}\boldsymbol{x}^{\boldsymbol{u}_{k,s}} \rangle\}$
  $Q = Q \cup \{\deg(\boldsymbol{x}^{\boldsymbol{u}}) + \alpha_k + \sum_{x_i \in F} \mathbb{C} \cdot \deg(x_i) \mid (\boldsymbol{x}^{\boldsymbol{u}}, Z) \in SP\}$
end for
return Q

Algorithm 1. Compute $\operatorname{qdeg}(M)$.

The union of these planes over all such pairs in the output is the quasidegree set of M.

The following is an example of `Quasidegrees.m2` computing the quasidegree set of an R-module:

```
i1 : R=QQ[x,y,z]
o1 = R
o1 : PolynomialRing
i2 : I=ideal(x*y,y*z)
o2 = ideal (x*y, y*z)
o2 : Ideal of R
i3 : M=R^1/I
o3 = cokernel | xy yz |
                              1
o3 : R-module, quotient of R
i4 : Q = quasidegrees M
o4 = {{0, {| 1 |}}, {0, {| 1 |, | 1 |}}}
o4 : List
```

The above example displays a caveat of `quasidegrees` in that there may be some redundancies in the output. By a redundancy, we mean when one plane in the output is contained in another. The redundancy above is clear:

$$\mathrm{qdeg}(\Bbbk[x, y, z]/\langle xy, yz\rangle) = \mathbb{C} = \{z_1 + z_2 \in \mathbb{C} \mid z_1, z_2 \in \mathbb{C}\}.$$

The function `removeRedundancy` gets rid of redundancies in the list of planes:

```
i5 : removeRedundancy Q
o5 = {{0, {| 1 |, | 1 |}}}
o5 : List
```

3. QUASIDEGREES AND HYPERGEOMETRIC SYSTEMS. In this section, we discuss the motivation for `Quasidegrees.m2` and the methods therein which aid us in our studies. Let $A = [a_1\ a_2\ \cdots\ a_n]$ be an integer $(d\times n)$-matrix with $\mathbb{Z}A = \mathbb{Z}^d$ and such that the cone over its columns is pointed. There is a natural $\mathbb{Z}^d$-grading of R by the columns of A given by $\deg(x_j) = a_j$, the j-th column of A. A module that is homogeneous with respect to this grading is said to be *A-graded*. By the assumptions on A, R is positively graded by A, that is, the only polynomials of degree 0 are the constants. Given such a matrix A and a polynomial ring R in n variables, the method `toGradedRing` gives R an A-grading. For example, let

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & -2 \end{pmatrix}.$$

We make the A-graded polynomial ring $\mathbb{Q}[x_1, x_2, x_3, x_4, x_5]$:

```
i6 : A=matrix{{1,1,1,1,1},{0,0,1,1,0},{0,1,1,0,-2}}
o6 = | 1 1 1 1 1  |
     | 0 0 1 1 0  |
     | 0 1 1 0 -2 |
               3        5
o6 : Matrix ZZ  <--- ZZ
i7 : R=QQ[x_1..x_5]
o7 = R
o7 : PolynomialRing
i8 : R=toGradedRing(A,R)
o8 = R
o8 : PolynomialRing
i9 : describe R
o9 = QQ[x , x , x , x , x , Degrees => {{1}, {1}, {1}, {1}, {1 }},
         1   2   3   4   5              {0}  {0}  {1}  {1}  {0 }
                                        {0}  {1}  {1}  {0}  {-2}
     Heft=>{1, 2:0},MonomialOrder=>{MonomialSize=>32},DegreeRank=>3]
                                   {GRevLex=>{5:1}}
                                   {Position=>Up}
```

The *toric ideal associated to* A in R is the binomial ideal

$$I_A = \langle \boldsymbol{x}^{\boldsymbol{u}} - \boldsymbol{x}^{\boldsymbol{v}} : A\boldsymbol{u} = A\boldsymbol{v} \rangle.$$

The method `toricIdeal` computes the toric ideal associated to A in the ring R. We continue with the A and R from the above example and compute the toric ideal I_A associated to A in R:

```
i10 : I=toricIdeal(A,R)
                              2    2     2              3    2
o10 = ideal (x x  - x x , x x  - x x , x x  - x x x , x  - x x )
             1 3    2 4   1 4    3 5   1 4    2 3 5   1    2 5
o10 : Ideal of R
```

We now introduce A-hypergeometric systems. Given a matrix $A \in \mathbb{Z}^{d\times n}$ as above and a $\beta \in \mathbb{C}^d$, the *A-hypergeometric system with parameter $\beta \in \mathbb{C}^d$* [Saito et al. 2000], denoted $H_A(\beta)$, is the system of partial differential equations:

$$\frac{\partial^{|\boldsymbol{v}|}}{\partial \boldsymbol{x}^{\boldsymbol{v}}}\phi(\boldsymbol{x}) = \frac{\partial^{|\boldsymbol{u}|}}{\partial \boldsymbol{x}^{\boldsymbol{u}}}\phi(\boldsymbol{x}) \qquad \text{for all } \boldsymbol{u}, \boldsymbol{v},\ A\boldsymbol{u} = A\boldsymbol{v},$$

$$\sum_{j=1}^{n} a_{ij} x_j \frac{\partial}{\partial x_j}\phi(\boldsymbol{x}) = \beta_i \phi(\boldsymbol{x}), \qquad \text{for } i = 1, \ldots, d.$$

Such systems are sometimes called *GKZ-hypergeometric systems*. The function `gkz` in the Macaulay2 package `Dmodules` computes this system as an ideal in the Weyl algebra. The *rank* of $H_A(\beta)$ is

$$\operatorname{rank}(H_A(\beta)) = \dim_{\mathbb{C}} \left\{ \begin{array}{l} \text{germs of holomorphic solutions of } H_A(\beta) \\ \text{near a generic nonsingular point} \end{array} \right\}.$$

The function `holonomicRank` in `Dmodules` computes the rank of an A-hypergeometric system. In general, rank is not a constant function of β. Denote $\operatorname{vol}(A)$ to be $d!$ times the Euclidean volume of $\operatorname{conv}(A \cup \{0\})$, the convex hull of the columns of A and the origin in $\mathbb{R}^d$. The following theorem gives the parameters β for which $\operatorname{rank}(H_A(\beta))$ is higher than expected:

Theorem 3.1 [Matusevich et al. 2005]. *Let $H_A(\beta)$ be an A-hypergeometric system with parameter β. If $\beta \in \operatorname{qdeg}(\bigoplus_{i=0}^{d-1} H^i_{\mathfrak{m}}(R/I_A))$ then $\operatorname{rank}(H_A(\beta)) > \operatorname{vol}(A)$. Otherwise, $\operatorname{rank}(H_A(\beta)) = \operatorname{vol}(A)$.*

Since Theorem 3.1 was the initial motivation for `Quasidegrees.m2`, the package has a method `quasidegreesLocalCohomology` (abbreviated `qlc`) to compute the quasidegree set of the local cohomology modules $H^i_{\mathfrak{m}}(R/I_A)$. If the input is an integer i and the R-module R/I_A, then the method computes $\operatorname{qdeg}(H^i_{\mathfrak{m}}(R/I_A))$. If the input is only the module R/I_A, the method computes the quasidegree set in Theorem 3.1.

We use graded local duality to compute the local cohomology modules of a finitely generated A-graded R-module supported at the maximal ideal $\mathfrak{m}$:

Theorem 3.2 (graded local duality [Bruns and Herzog 1993; Miller 2002]). *Given an A-graded R-module M, there is an A-graded vector space isomorphism*

$$\operatorname{Ext}^{n-i}_R(M, R)_\alpha \cong \operatorname{Hom}_{\Bbbk}(H^i_{\mathfrak{m}}(M)_{-\alpha-\varepsilon_A}, \Bbbk),$$

where $\mathfrak{m} = \langle x_1, \ldots, x_n \rangle$ and $\varepsilon_A = \sum_{j=1}^n a_j$.

The algorithm implemented for `quasidegreesLocalCohomology` is essentially Algorithm 1 applied to the Ext-modules of M with the additional twist of ε_A coming from local duality. For our purposes, we exploit the fact that the higher syzygies of R/I_A are generated by monomials in R^m (see [Miller and Sturmfels 2005], Chapter 9).

Continuing our running example, we use `quasidegreesLocalCohomology` to compute the quasidegree set of $\bigoplus_{i=0}^{d-1} H^i_{\mathfrak{m}}(R/I_A)$:

```
i11 : M=R^1/I
o11 = cokernel | x_1x_3-x_2x_4 x_1x_4^2-x_3^2x_5
x_1^2x_4-x_2x_3x_5 x_1^3-x_2^2x_5 |
                                   1
o11 : R-module, quotient of R
```

```
i12 : quasidegreesLocalCohomology M
o12 = {{| 0 |, {| 1  |}}}
        | 0 |   | 0  |
        | 1 |   | -2 |
o12 : List
```

Thus

$$\operatorname{qdeg}\left(\bigoplus_{i=0}^{d-1} H^i_{\mathfrak{m}}(R/I_A)\right) = \begin{bmatrix}0\\0\\1\end{bmatrix} + \mathbb{C}\cdot\begin{bmatrix}1\\0\\-2\end{bmatrix}. \tag{1}$$

As a check, we use the methods `gkz` and `holonomicRank` from the package `Dmodules` to compute $\operatorname{rank}(H_A(0))$ and $\operatorname{rank}(H_A(\beta))$ for two different β in (1) and demonstrate a rank jump:

```
i13 : holonomicRank gkz(A,{0,0,0}) -- vol A in this case
o13 = 4
i14 : holonomicRank gkz(A,{0,0,1})
o14 = 5
i15 : holonomicRank gkz(A,{3/2,0,-2})
o15 = 5
```

SUPPLEMENT. The online supplement contains version 1.0 of `Quasidegrees.m2`.

REFERENCES.

[Bruns and Herzog 1993] W. Bruns and J. Herzog, *Cohen–Macaulay rings*, Cambridge Studies in Advanced Mathematics **39**, Cambridge University Press, 1993. MR Zbl

[Eisenbud 1995] D. Eisenbud, *Commutative algebra: with a view toward algebraic geometry*, Graduate Texts in Mathematics **150**, Springer, 1995. MR Zbl

[Matusevich et al. 2005] L. F. Matusevich, E. Miller, and U. Walther, "Homological methods for hypergeometric families", *J. Amer. Math. Soc.* **18**:4 (2005), 919–941. MR Zbl

[Miller 2002] E. Miller, "Graded Greenlees–May duality and the Čech hull", pp. 233–253 in *Local cohomology and its applications* ((Guanajuato, 1999)), edited by G. Lyubeznik, Lecture Notes in Pure and Appl. Math. **226**, Dekker, New York, 2002. MR Zbl

[Miller and Sturmfels 2005] E. Miller and B. Sturmfels, *Combinatorial commutative algebra*, Graduate Texts in Mathematics **227**, Springer, 2005. MR Zbl

[Saito et al. 2000] M. Saito, B. Sturmfels, and N. Takayama, *Gröbner deformations of hypergeometric differential equations*, Algorithms and Computation in Mathematics **6**, Springer, 2000. MR Zbl

RECEIVED: 27 Jun 2015 REVISED: 13 Oct 2018 ACCEPTED: 26 Feb 2019

ROBERTO BARRERA:

rbarrera@txstate.edu

Department of Mathematics, Texas State University, San Marcos, TX, United States

JSAG **9** (2019), 35–41
dx.doi.org/10.2140/jsag.2019.9.35

The Journal of Software for
Algebra and Geometry

An algorithm for enumerating difference sets

DYLAN PEIFER

ABSTRACT: The `DifSets` package for GAP implements an algorithm for enumerating all difference sets in a group up to equivalence and provides access to a library of results. The algorithm functions by finding difference sums, which are potential images of difference sets in quotient groups of the original group, and searching their preimages. In this way, the search space can be dramatically decreased, and searches of groups of relatively large order (such as order 64 or order 96) can be completed.

1. INTRODUCTION. Let G be a finite group of order v and D a subset of G with k elements. Then D is a (v, k, λ)-*difference set* if each nonidentity element of G can be written as $d_i d_j^{-1}$ for $d_i, d_j \in D$ in exactly λ different ways. Difference sets were first studied in relation to finite geometries [Singer 1938] and have connections to symmetric designs, coding theory, and many other fields of mathematics [Moore and Pollatsek 2013; Davis and Jedwab 1996; Colbourn and Dinitz 1996; Beth et al. 1999].

Large libraries of difference sets are useful for developing conjectures and building examples. Gordon provides an extensive library of difference sets in abelian groups [Gordon], but has no results for nonabelian groups, which do show distinct behavior [Smith 1995]. A wide variety of techniques can be used to construct difference sets for these libraries (see, for example, [Dillon 1985] and [Davis and Jedwab 1997]), but fully enumerating all difference sets in a given group requires some amount of exhaustive search, which can quickly become computationally infeasible. Kibler [1978] performed the first major exhaustive enumeration of difference sets, and considered groups where difference sets could be found with $k < 20$. In recent years, AbuGhneim [2013; 2016] has performed almost complete enumerations for all groups of order 64, and several authors have found difference sets in groups of order 96 [Golemac et al. 2005; 2007; AbuGhneim and Smith 2007]. The `DifSets` package for the computer algebra system [GAP] efficiently and generally implements the techniques used by these and many other authors to

MSC2010: 05B10.
Keywords: difference sets, exhaustive search, GAP.
`DifSets` version 2.2.0

exhaustively enumerate all difference sets up to equivalence in a group. With the package loaded, a search of a given group can be performed with a single command.

```
gap> DifferenceSets(CyclicGroup(7));
[ [ 1, 2, 4 ] ]
```

The package has been used to give the first complete enumeration of all difference sets up to equivalence in groups of order 64 and 96, and in total provides a library of results for 1006 of the 1032 groups of order less than 100. Results are organized by their id in the `SmallGroups` library [SmallGrp] and can be easily loaded by GAP.

```
gap> LoadDifferenceSets(16,5);  # results for SmallGroup(16,5)
[ [ 1, 2, 3, 4, 8, 15 ], [ 1, 2, 3, 4, 11, 13 ] ]
```

The ease of use of these top-level functions is the primary interface difference between the `DifSets` package and a similar GAP package [RDS], which provides a variety of tools to search for difference sets. The functions involving coset signatures in `RDS` provide similar functionality to the `DifSets` package, but require substantial user interaction to perform efficient searches, and are not feasible for searching most groups of order 64 and 96. In addition, `RDS` provides no precomputed results, though it does provide significant additional functionality related to relative difference sets, partial difference sets, and projective planes.

2. DIFFERENCE SETS. For notational purposes it is useful to consider a subset $D \subseteq G$ as an element of the group ring $\mathbb{Z}[G]$. We will abuse notation to define the group ring elements

$$G = \sum_{g \in G} g, \quad D = \sum_{d \in D} d, \quad D^{(-1)} = \sum_{d \in D} d^{-1}, \quad gD = \sum_{d \in D} gd, \quad D^{\phi} = \sum_{d \in D} \phi(d),$$

where $g \in G$ and ϕ is a homomorphism from G. Then the statement that D is a (v, k, λ)-difference set is equivalent to the equation

$$DD^{(-1)} = (k - \lambda)1_G + \lambda G,$$

where D is an element of $\mathbb{Z}[G]$ with coefficients in $\{0, 1\}$. With this definition it is a quick exercise to prove the following (see page 298 of [Beth et al. 1999] and Theorem 4.2 and 4.11 of [Moore and Pollatsek 2013]).

Proposition 1. *Let G be a group of order v. Then*:

(1) *Any one element subset of G is a $(v, 1, 0)$-difference set.*

(2) *The complement of a (v, k, λ)-difference set in G is a $(v, v - k, \lambda + v - 2k)$-difference set in G.*

(3) *If D is a (v, k, λ)-difference set in G, $g \in G$, and $\phi \in \mathrm{Aut}(G)$, then gD^{ϕ} is also a (v, k, λ)-difference set in G.*

In addition, an immediate consequence of the definition is that $k(k-1) = \lambda(v-1)$ for any valid set of parameters of a difference set, so that for a given value of v there are typically only a few possible values of k and λ. More sophisticated results, such as the Bruck–Ryser–Chowla theorem, can reduce the number of possibilities even further.

As a result of Proposition 1, in enumerating difference sets we ignore the trivial one element difference sets, only take the smaller of each complementary pair of sets, and only consider sets distinct up to an equivalence given by part (3).

Definition 2. Let D_1 and D_2 be difference sets in G. Then D_1 and D_2 are *equivalent difference sets* if $D_1 = gD_2^{\phi}$ for some $g \in G$ and $\phi \in \mathrm{Aut}(G)$.

In the `DifSets` package, difference sets are stored as lists of integers. These integers represent indices in the list returned by the GAP function `Elements(G)`, which is a sorted[1] list of elements of the group `G`. For example, consider the group $C_7 = \langle x \mid x^7 = 1 \rangle$. In GAP we have

```
gap> C7 := CyclicGroup(7);;
gap> Elements(C7);
[ <identity> of ..., f1, f1^2, f1^3, f1^4, f1^5, f1^6 ]
```

where clearly `f1` is the generator corresponding to our x. Then the subset $D = \{x, x^2, x^4\}$ corresponds to the set consisting of the second, third, and fifth elements of `Elements(C7)`, which we can represent in indices as `[2, 3, 5]`. We can check that this is a difference set and also note that it is equivalent to the difference set $xD = \{x^2, x^3, x^5\}$, which is represented as `[3, 4, 6]`.

```
gap> IsDifferenceSet(C7, [2, 3, 5]);
true
gap> IsEquivalentDifferenceSet(C7, [2, 3, 5], [3, 4, 6]);
true
```

3. DIFFERENCE SUMS. A basic method for enumerating all difference sets in a group G is to enumerate all subsets of G and check if each is a difference set by definition. But since the number of subsets in a group is exponential in its order, we cannot feasibly enumerate and test all subsets for groups of even a modest size. The key to decreasing the search space is the following well-known lemma, which motivates our definition of a *difference sum*.[2]

[1] Element comparison (and thus the list `Elements(G)`) is instance-independent in GAP for permutation and pc groups, which includes, for example, all groups in the `SmallGroups` library.

[2] Concepts similar to difference sums are elsewhere referred to as difference lists, intersection numbers, or signatures. However, difference sums require both a group G and normal subgroup N, not just the group structure of the quotient G/N used in some other definitions. This precision is needed for specifying induced automorphisms in Definition 7 so that we can prove Lemma 8.

1	1	1	0	0	1	0	1	0	0	1	0	0	0	1	G/N_3
3			1			1			1			1			G/N_2
7															G/N_1

Figure 1. A difference set of size 7 in the group G of order 15 and the difference sums it induces in G/N_i where $G = N_1 \triangleright N_2 \triangleright N_3 = \{1\}$. Each row in the diagram is a group, with each block a coset.

Lemma 3. *Suppose D is a (v, k, λ)-difference set in G and θ is a homomorphism of G with $|\ker(\theta)| = w$. Let $S = D^\theta$ and $H = G^\theta$. Then*

$$SS^{(-1)} = (k - \lambda)1_H + \lambda w H.$$

Definition 4. Given a finite group G and normal subgroup N, a (v, k, λ)-*difference sum* is an element S of $\mathbb{Z}[G/N]$ such that $SS^{(-1)} = (k - \lambda)1_{G/N} + \lambda|N|G/N$ and the coefficients of S have values in $\{0, 1, \ldots, |N|\}$.

By construction, any difference set in G induces difference sums under the natural projection in quotients of G, as seen in Figure 1. Precisely, we have:

Lemma 5. *Suppose G is a finite group with normal subgroup N and natural projection $\pi : G \to G/N$. Then any (v, k, λ)-difference set D in G induces a (v, k, λ)-difference sum D^π in G/N.*

Lemma 6. *Suppose G is a finite group with normal subgroups N_1 and N_2 such that $N_2 \subseteq N_1$ and $\pi : G/N_2 \to G/N_1$ is the natural projection. Then any (v, k, λ)-difference sum S in G/N_2 induces a (v, k, λ)-difference sum S^π in G/N_1.*

Lemma 5 means that our search for difference sets only requires checking the subsets of G that induce difference sums in some quotient. In finding these difference sums, Lemma 6 additionally allows us to only test sums that induce difference sums in further quotients. In each case the search space is dramatically decreased. Since our search is for difference sets up to equivalence, we also define a complementary equivalence of difference sums such that equivalent difference sums are induced by equivalent collections of difference sets.

Definition 7. Let S_1 and S_2 be difference sums in G/N. Then S_1 and S_2 are *equivalent difference sums* if $S_1 = gS_2^\phi$ for some $g \in G/N$ and ϕ an automorphism of G/N induced by an automorphism of G.

Lemma 8. *Suppose S_1 and S_2 are equivalent difference sums in G/N. Then if D_1 is any difference set in G that induces S_1, there exists a difference set D_2 in G that induces S_2 such that D_1 and D_2 are equivalent.*

In the `DifSets` package, difference sums are stored as lists of integers representing the coefficients of the group ring elements, with position in the list given by the

position of the coset in the list returned by the GAP function `Elements(G/N)`. For example, `[3, 1, 1, 1, 1]` represents a difference sum in `SmallGroup(15, 1)` mod its normal subgroup of order 3 with coefficient 3 on the identity coset and coefficient 1 on all other cosets.

```
gap> G := SmallGroup(15, 1);; N := NormalSubgroups(G)[2];;
gap> IsDifferenceSum(G, N, [3, 1, 1, 1, 1]);
true
```

4. ALGORITHM. The basic structure of the algorithm is to start at the bottom of Figure 1 and travel upwards. Given a group G, first compute $v = |G|$ and then find all values of k that give solutions satisfying the Bruck–Ryser–Chowla theorem to the equation $k(k-1) = \lambda(v-1)$ mentioned in Section 2. For example,

```
gap> G := SmallGroup(15, 1);;
gap> PossibleDifferenceSetSizes(G);
[ 7 ]
```

Each value of k will be handled separately. The algorithm starts with the normal subgroup $N_1 = G$, where the only difference sum of size k in $G/N_1 = \{1\}$ is `[k]`.

```
gap> N1 := G;;
gap> difsums := [ [7] ];;
```

Given a normal subgroup N_2 of G such that $N_2 \subseteq N_1$, first enumerate all preimages in G/N_2 of current difference sums in G/N_1 and return those that are themselves difference sums. Then remove all but one representative of each equivalence class from this collection.

```
gap> N2 := NormalSubgroups(G)[2];;
gap> difsums := AllRefinedDifferenceSums(G, N1, N2, difsums);
[ [ 1, 1, 1, 1, 3 ], [ 1, 1, 1, 3, 1 ], [ 1, 1, 3, 1, 1 ],
  [ 1, 3, 1, 1, 1 ], [ 3, 1, 1, 1, 1 ] ]
gap> difsums := EquivalentFreeListOfDifferenceSums(G, N2, difsums);
[ [ 3, 1, 1, 1, 1 ] ]
```

In the general case, the above step is repeated along a chief series

$$G = N_1 \triangleright \cdots \triangleright N_r = \{1\}$$

of G with N_{r-1} a nontrivial normal subgroup of minimal possible size in G. At N_{r-1}, enumerate sets and remove equivalents to leave the final result.

```
gap> difsets := AllRefinedDifferenceSets(G, N2, difsums);
[ [ 1, 2, 4, 3, 8, 11, 12 ], [ 1, 2, 4, 3, 10, 13, 12 ],
  [ 1, 2, 4, 5, 6, 9, 14 ], [ 1, 2, 4, 5, 10, 13, 14 ],
  [ 1, 2, 4, 7, 6, 9, 15 ], [ 1, 2, 4, 7, 8, 11, 15 ] ]
gap> difsets := EquivalentFreeListOfDifferenceSets(G, difsets);
[ [ 1, 2, 4, 7, 8, 11, 15 ] ]
```

These steps are encapsulated in the function `DifferenceSets` mentioned in Section 1, with two modifications. First, since every difference set is equivalent to some difference set containing the identity, the algorithm does not enumerate some preimages that are guaranteed to be equivalent to others. Second, the final elimination of all but one representative of equivalence classes of difference sets uses the `SmallestImageSet` function [Linton 2004] from the GAP package [GRAPE]. Although roughly 20% slower than the function given above for most cases, `SmallestImageSet` gives a unique minimal result and handles groups with large automorphism groups much more efficiently.

5. RESULTS. The `DifSets` package successfully computed results for 1006 of the 1032 groups of order less than 100, including all groups of order 64 and 96. Full results with timings and comments can be found in the package and its documentation. Here we include a summary for order 64 and 96. All computations were performed with GAP 4.9.1 on a 4.00GHz i7-6700K using 8GB of RAM.

Order	Groups	Difference sets	Median time per group	Total time
64	267	330159	0.415 hours	295.811 hours
96	231	2627	3.133 hours	1568.746 hours

Timing comparisons with the `RDS` package mentioned in Section 1 are difficult since `RDS` provides a variety of tools rather than a single algorithm. Ordered coset signatures in `RDS` correspond to difference sums in `DifSets`, but, unlike difference sums, coset signatures cannot be refined through multiple stages, which makes the generation of good coset signatures in `RDS` infeasible for most order 64 and order 96 groups. However, if an ordered signature is available, building difference sets through partial difference sets in `RDS` can in some cases be much faster than searching the corresponding difference sum using `DifSets`. In particular, replacing the final step in Section 4 with a search using `RDS` can significantly improve times for some groups of order 96. Further work to combine the refining of difference sums used by `DifSets` and the generation of difference sets through partial difference sets used by `RDS` could lead to significantly better times than either package could manage alone.

ACKNOWLEDGEMENTS. The author thanks Ken Smith, Alexander Hulpke, and an anonymous reviewer for helpful comments that improved the `DifSets` package and this article.

SUPPLEMENT. The online supplement contains version 2.2.0 of `DifSets`.

REFERENCES.

[AbuGhneim 2013] O. AbuGhneim, "On (64, 28, 12) difference sets", *Ars Combin.* **111** (2013), 401–419. MR Zbl

[AbuGhneim 2016] O. AbuGhneim, "All (64, 28, 12) difference sets and related structures", *Ars Combin.* **125** (2016), 271–285. MR Zbl

[AbuGhneim and Smith 2007] O. AbuGhneim and K. Smith, "Nonabelian groups with (96, 20, 4) difference sets", *Electron. J. Combin.* **14**:1 (2007), art. id. R8, 17. MR Zbl

[Beth et al. 1999] T. Beth, D. Jungnickel, and H. Lenz, *Design theory, Vol. I*, 2nd ed., Encyclopedia of Mathematics and its Applications **69**, Cambridge University Press, 1999. MR Zbl

[Colbourn and Dinitz 1996] C. J. Colbourn and J. H. Dinitz (editors), *The CRC handbook of combinatorial designs*, CRC Press, Boca Raton, FL, 1996. MR Zbl

[Davis and Jedwab 1996] J. A. Davis and J. Jedwab, "A survey of Hadamard difference sets", pp. 145–156 in *Groups, difference sets, and the Monster* (Columbus, OH, 1993), edited by K. T. Arasu et al., Ohio State Univ. Math. Res. Inst. Publ. **4**, de Gruyter, Berlin, 1996. MR Zbl

[Davis and Jedwab 1997] J. A. Davis and J. Jedwab, "A unifying construction for difference sets", *J. Combin. Theory Ser. A* **80**:1 (1997), 13–78. MR Zbl

[Dillon 1985] J. F. Dillon, "Variations on a scheme of McFarland for noncyclic difference sets", *J. Combin. Theory Ser. A* **40**:1 (1985), 9–21. MR Zbl

[GAP] The GAP Group, "GAP – Groups, Algorithms, and Programming", available at https://www.gap-system.org.

[Golemac et al. 2005] A. Golemac, T. Vučičić, and J. Mandić, "One (96, 20, 4)-symmetric design and related nonabelian difference sets", *Des. Codes Cryptogr.* **37**:1 (2005), 5–13. MR

[Golemac et al. 2007] A. Golemac, J. Mandić, and T. Vučičić, "On the existence of difference sets in groups of order 96", *Discrete Math.* **307**:1 (2007), 54–68. MR Zbl

[Gordon] D. Gordon, "La Jolla difference set repository", available at https://www.dmgordon.org/diffset/.

[GRAPE] L. H. Soicher, "GRAPE – GRaph Algorithms using PErmutation groups", GAP package version 4.7, available at http://www.maths.qmul.ac.uk/∼leonard/grape/.

[Kibler 1978] R. E. Kibler, "A summary of noncyclic difference sets, $k < 20$", *J. Combinatorial Theory Ser. A* **25**:1 (1978), 62–67. MR Zbl

[Linton 2004] S. Linton, "Finding the smallest image of a set", pp. 229–234 in *ISSAC 2004*, edited by J. Gutierrez, ACM, New York, 2004. MR Zbl

[Moore and Pollatsek 2013] E. H. Moore and H. S. Pollatsek, *Difference sets: connecting algebra, combinatorics, and geometry*, Student Mathematical Library **67**, American Mathematical Society, Providence, RI, 2013. MR

[RDS] M. Roeder, "RDS – a package for searching relative difference sets", GAP package version 1.6, available at http://csserver.evansville.edu/∼mroeder.

[Singer 1938] J. Singer, "A theorem in finite projective geometry and some applications to number theory", *Trans. Amer. Math. Soc.* **43**:3 (1938), 377–385. MR Zbl

[SmallGrp] E. O. B. Eick, H. U. Besche, "SmallGrp – the GAP small groups library", GAP package, version 1.3, available at https://gap-packages.github.io/smallgrp/.

[Smith 1995] K. W. Smith, "Non-abelian Hadamard difference sets", *J. Combin. Theory Ser. A* **70**:1 (1995), 144–156. MR Zbl

RECEIVED: 5 Jul 2018 REVISED: 4 Sep 2018 ACCEPTED: 26 Feb 2019

DYLAN PEIFER:
djp282@cornell.edu
Department of Mathematics, Cornell University, Ithaca, NY, United States

JSAG **9** (2019), 43–54
dx.doi.org/10.2140/jsag.2019.9.43

The Journal of Software for
Algebra and Geometry

Hyperplane arrangements in CoCoA

ELISA PALEZZATO AND MICHELE TORIELLI

ABSTRACT: We introduce the package `arrangements` for the software CoCoA. This package provides a data structure and the necessary methods for working with hyperplane arrangements. In particular, the package implements methods to generate several known families of arrangements, to perform operations on them, and to calculate various invariants associated to them.

1. INTRODUCTION. An arrangement of hyperplanes is a finite collection of codimension one affine subspaces in a finite dimensional vector space. Associated to these spaces, there is a plethora of algebraic, combinatorial and topological invariants. Arrangements are easily defined but they lead to deep and beautiful results connecting various area of mathematics. We refer the reader to [Orlik and Terao 1992] for a comprehensive account of this subject.

One of the main goals in the study of hyperplane arrangements is to decide whether a given invariant is combinatorially determined, and, if so, to express it explicitly in terms of the intersection lattice of the arrangement.

We describe the new package `arrangements` for CoCoA [CoCoA; CoCoALib; Abbott and Bigatti 2018]). This package computes several combinatorial invariants (like the lattice of intersections and its flats, the Poincaré, the characteristic and the Tutte polynomials) and algebraic ones (like the Orlik–Terao and the Solomon–Terao ideals) of hyperplane arrangements. Moreover, several functions for the class of free hyperplane arrangements are implemented. In addition, this package also allows computations with multiarrangements. Finally, several known families of arrangements (like classic reflection arrangements, Shi arrangements, Catalan arrangements, Shi–Catalan arrangements, graphical arrangements and signed graphical ones) can be easily constructed: in CoCoA type `?ArrFamily` for the complete list. Some of the functions that compute combinatorial invariants rely on the CoCoA package `posets`, which we implemented for this purpose.

We introduce the package `arrangements` via several examples. Specifically, in Section 2 we first recall the definitions of various combinatorial invariants of

MSC2010: primary 32S22, 52C35; secondary 03G10.
Keywords: hyperplane arrangements, freeness, CoCoA.
`arrangements` version 1.0 for CoCoA-5.2.4

a given arrangement and then describe how to compute them. In Section 3, we describe how to work with free hyperplane arrangements, and in Section 4 how to define the Orlik–Terao and Solomon–Terao ideals. Finally, in Section 5 we describe the class of multiarrangements with particular emphasis on the free ones.

This package is part of the official release CoCoA-5.2.4, and has been used during the tutorials of the Hokkaido Summer Institute 2018 course "Hyperplane arrangements and computations with CoCoA" held at Hokkaido University from the 13th to the 17th of August 2018.

2. COMBINATORICS OF ARRANGEMENTS. Let V be a vector space of dimension l over a field K. Fix a system of coordinates $(x_1, \dots, x_l)$ of V^*. We denote by $S = S(V^*) = K[x_1, \dots, x_l]$ the symmetric algebra. A finite set of affine hyperplanes $\mathcal{A} = \{H_1, \dots, H_n\}$ in V is called a *hyperplane arrangement*.

For each hyperplane H_i we fix a polynomial $\alpha_i \in S$ such that $H_i = \alpha_i^{-1}(0)$, and let

$$Q(\mathcal{A}) = \prod_{i=1}^{n} \alpha_i.$$

An arrangement $\mathcal{A}$ is called *central* if each H_i contains the origin of V. In this case, the polynomial $\alpha_i \in S$ is linear homogeneous, and hence $Q(\mathcal{A})$ is a homogeneous polynomial of degree n.

The operation of *coning* allows one to transform any arrangement $\mathcal{A}$ of V with n hyperplanes into a central arrangement $c\mathcal{A}$ with $n+1$ hyperplanes in a vector space of dimension $l+1$; see [Orlik and Terao 1992].

Notice that in CoCoA to compute the cone of an arrangement $\mathcal{A}$, the homogenizing variable needs to be already present in the ring in which the equation of $\mathcal{A}$ is defined. For example, we can construct the cone of the *Shi arrangement of type A* as follows:

```
/**/ use S ::= QQ[x, y, z, w];
/**/ A := ArrShiA(S, 3); A;
[x-y, x-z, y-z, x-y-1, x-z-1, y-z-1]
/**/ ArrCone(A, w);
[x-y, x-z, y-z, x-y-w, x-z-w, y-z-w, w]
```

Let $L(\mathcal{A}) = \left\{\bigcap_{H\in\mathcal{B}} H \mid \mathcal{B} \subseteq \mathcal{A}\right\}$ be the *intersection poset* of $\mathcal{A}$. Define a partial order on $L(\mathcal{A})$ by $X \leq Y$ if and only if $Y \subseteq X$, for all $X, Y \in L(\mathcal{A})$. Note that this is the reverse inclusion. In addition, if $\mathcal{A}$ is central, $L(\mathcal{A})$ is a geometric lattice. The elements of $L(\mathcal{A})$ are called *flats* of $\mathcal{A}$. Define a rank function on $L(\mathcal{A})$ by $\mathrm{rk}(X) = \mathrm{codim}(X)$. The poset $L(\mathcal{A})$ plays a fundamental role in the study of hyperplane arrangements; in fact it determines the combinatorics of the arrangement.

We can compute the flats in the intersection lattice of the *reflection arrangement of type D* in the following way:

```
/**/ use S ::= QQ[x, y, z];
/**/ A := ArrTypeD(S, 3); A;
[x-y,  x+y,  x-z,  x+z,  y-z,  y+z]
/**/ ArrFlats(A);
[[ideal(0)],
 [ideal(x-y), ideal(x+y), ideal(x-z), ideal(x+z),
  ideal(y-z), ideal(y+z)],
 [ideal(x, y), ideal(x-z, y-z), ideal(x+z, y+z),
  ideal(x-z, y+z),  ideal(x+z, y-z), ideal(x, z),
  ideal(y, z)],
 [ideal(x, y, z)]]
```

In the rest of the section, we will introduce the Poincaré polynomial, the characteristic polynomial and the Tutte polynomial, and the restriction of an arrangement $\mathcal{A}$. Notice that, contrary to the operation of coning, in CoCoA these operations introduce new variables that do not need to be already present in the ring in which the equation of $\mathcal{A}$ is defined.

Let $\mu : L(\mathcal{A}) \to \mathbb{Z}$ be the *Möbius function* of $L(\mathcal{A})$ defined by

$$\mu(X) = \begin{cases} 1 & \text{for } X = V, \\ -\sum_{Y<X} \mu(Y) & \text{if } X > V. \end{cases}$$

The *Poincaré polynomial* of $\mathcal{A}$ is defined by

$$\pi(\mathcal{A}, t) = \sum_{X \in L(\mathcal{A})} \mu(X)(-t)^{\mathrm{rk}(X)},$$

and it satisfies the formula

$$\pi(c\mathcal{A}, t) = (t+1)\pi(\mathcal{A}, t).$$

We now verify the previous result for the *Shi arrangement of type A*.

```
/**/ use S ::= QQ[x, y, z, w];
/**/ A := ArrShiA(S, 3);
/**/ pi_A := ArrPoincarePoly(A); pi_A;
9*t^2 +6*t +1
/**/ cA := ArrCone(A, w);
/**/ pi_cA := ArrPoincarePoly(cA); pi_cA;
9*t^3+15*t^2+7*t+1
/**/ t := indet(RingOf(pi_A), 1);
/**/ pi_cA = (1+t)*pi_A;
```

```
true
```

For any flat $X \in L(\mathcal{A})$ define the *localization* of $\mathcal{A}$ to X as the subarrangement $\mathcal{A}_X$ of $\mathcal{A}$ by

$$\mathcal{A}_X = \{H \in \mathcal{A} \mid X \subseteq H\}.$$

Similarly, define the *restriction* of $\mathcal{A}$ to X as the arrangement $\mathcal{A}^X$ in X,

$$\mathcal{A}^X = \{X \cap H \mid H \in \mathcal{A}\backslash\mathcal{A}_X \text{ and } X \cap H \neq \varnothing\}.$$

The *characteristic polynomial* of $\mathcal{A}$ is

$$\chi(\mathcal{A}, t) = t^l \pi(\mathcal{A}, -t^{-1}) = \sum_{X \in L(\mathcal{A})} \mu(X) t^{\dim(X)}.$$

The characteristic polynomial is characterized by the recursive relation

$$\chi(\mathcal{A}, t) = \chi(\mathcal{A}\backslash H, t) - \chi(\mathcal{A}^H, t),$$

for any $H \in \mathcal{A}$. See [Orlik and Terao 1992, Corollary 2.57] for more details.

We verify the previous result for $\mathcal{A}^{[-1,2]}$ the *Shi–Catalan arrangement of type A*.

```
/**/ use S ::= QQ[x, y, z];
/**/ A := ArrShiCatalanA(S, 3, [-1, 2]); A;
[x-y, x-z, y-z, x-y-1, x-z-1, y-z-1, x-y+1, x-y+2, x-z+1,
 x-z+2, y-z+1, y-z+2]
/**/ ArrLocalization(A, [x-y, x-z]);
[x-y,  x-z,  y-z]
/**/ A_minusH := ArrDeletion(A, x-y-1); A_minusH;
[x-y, x-z, y-z, x-z-1, y-z-1, x-y+1, x-y+2, x-z+1, x-z+2,
 y-z+1, y-z+2]
/**/ A_restrH := ArrRestriction(A, x-y-1); A_restrH;
[y[1]-y[2]+1, y[1]-y[2], y[1]-y[2]-1, y[1]-y[2]+2,
 y[1]-y[2]+3]
/**/ ArrCharPoly(A) = ArrCharPoly(A_minusH) -
 ArrCharPoly(A_restrH);
true
```

For $i = 0, \ldots, l$ we define the *i-th Betti number* $b_i(\mathcal{A})$ to be the coefficients of $\chi(\mathcal{A}, t)$ as in the formula

$$\chi(\mathcal{A}, t) = \sum_{i=0}^{l} (-1)^i b_i(\mathcal{A}) t^{l-i}.$$

The following statement is the combination of three different results from [Crapo and Rota 1970], [Orlik and Solomon 1980] and [Zaslavsky 1975], and it describes

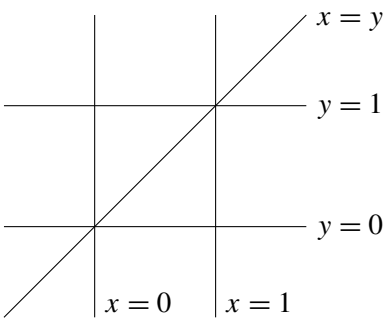

Figure 1. A line arrangement in $\mathbb{R}^2$.

the connection between the characteristic polynomial in combinatorics, and geometrical and topological aspects of arrangements.

Theorem 2.1. (1) *If $\mathcal{A}$ is an arrangement in $\mathbb{F}_q^l$ (vector space over a finite field $\mathbb{F}_q$), then $|\mathbb{F}_q^l \setminus \bigcup_{H\in\mathcal{A}} H| = \chi(\mathcal{A}, q)$.*

(2) *If $\mathcal{A}$ is an arrangement in $\mathbb{C}^l$, then the topological i-th Betti number of the complement of $\mathcal{A}$ is $b_i(\mathbb{C}^l \setminus \bigcup_{H\in\mathcal{A}} H) = b_i(\mathcal{A})$.*

(3) *If $\mathcal{A}$ is an arrangement in $\mathbb{R}^l$, then $|\chi(\mathcal{A}, -1)|$ is the number of chambers and $|\chi(\mathcal{A}, 1)|$ is the number of bounded chambers.*

Using the previous statements, we can compute the Betti numbers, the number of chambers and the number of bounded chambers of the arrangement in Figure 1.

```
/**/ use S ::= QQ[x, y];
/**/ A := [x, x-1, y, y-1, x-y];
/**/ ArrBettiNumbers(A);
[1,  5,  6]
/**/ NumChambers(A);
12
/**/ NumBChambers(A);
2
```

Associated to each hyperplane arrangement, we can naturally define a third polynomial. The *Tutte polynomial* of $\mathcal{A}$ is

$$T_{\mathcal{A}}(x, y) = \sum_{\substack{\mathcal{B}\subseteq\mathcal{A} \\ \mathcal{B} \text{ central}}} (x-1)^{\mathrm{rk}(\mathcal{A})-\mathrm{rk}(\mathcal{B})}(y-1)^{|\mathcal{B}|-\mathrm{rk}(\mathcal{B})}.$$

As shown in [Ardila 2007], it turns out that the Tutte and the characteristic polynomials are related by

$$\chi(\mathcal{A}, t) = (-1)^{\mathrm{rk}(\mathcal{A})} t^{l-\mathrm{rk}(\mathcal{A})} T_{\mathcal{A}}(1-t, 0).$$

We verify the previous result for the *reflection arrangement of type D*. Notice that here, since the Tutte and the characteristic polynomials live in different rings, we need to construct a ring homomorphism, with the command `PolyAlgebraHom`, to check the required equality.

```
/**/ use S ::= QQ[x, y, z];
/**/ A := ArrTypeD(S, 3);
/**/ Tutte_A := ArrTuttePoly(A); Tutte_A;
t[1]^3+t[2]^3+3*t[1]^2+4*t[1]*t[2]+3*t[2]^2+2*t[1]+2*t[2]
/**/ char_A := ArrCharPoly(A); char_A;
t^3-6*t^2+11*t-6
/**/ QQt1t2 := RingOf(Tutte_A); QQt := RingOf(char_A);
/**/ t := indet(QQt, 1);
/**/ phi := PolyAlgebraHom(QQt1t2, QQt, [1-t, 0]);
/**/ char_A = (-1)^3*t^(dim(S)-3)*phi(Tutte_A);
true
```

3. FREE HYPERPLANE ARRANGEMENTS. In the theory of hyperplane arrangements, the freeness of an arrangement is a very important algebraic property. In fact, freeness implies several interesting geometric and combinatorial properties of the arrangement itself. See, for example, [Terao 1980; Yoshinaga 2014; Abe 2016; Bigatti et al. 2019; Palezzato and Torielli 2018].

We use

$$\mathrm{Der}_V = \left\{ \sum_{i=1}^{l} f_i \partial_{x_i} \mid f_i \in S \right\}$$

to denote the S-module of *polynomial vector fields* on V (or S-derivations). Let $\delta = \sum_{i=1}^{l} f_i \partial_{x_i} \in \mathrm{Der}_V$. If $f_1, \ldots, f_l$ are homogeneous polynomials of degree d in S, then δ is said to be *homogeneous of polynomial degree* d, and we write $\mathrm{pdeg}(\delta) = d$.

For any central arrangement $\mathcal{A}$ we define the *module of vector fields logarithmic tangent* to $\mathcal{A}$ (logarithmic vector fields) by

$$D(\mathcal{A}) = \{\delta \in \mathrm{Der}_V \mid \delta(\alpha_i) \in \langle \alpha_i \rangle S, \text{ for all } i\}.$$

The module $D(\mathcal{A})$ is a graded S-module and we have

$$D(\mathcal{A}) = \{\delta \in \mathrm{Der}_V \mid \delta(Q(\mathcal{A})) \in \langle Q(\mathcal{A}) \rangle S\}.$$

Definition 3.1. We say a central arrangement $\mathcal{A}$ is *free with exponents* $(e_1, \ldots, e_l) \in \mathbb{N}^l$ if and only if $D(\mathcal{A})$ is a free S-module and there exists a basis $\delta_1, \ldots, \delta_l \in D(\mathcal{A})$ such that $\mathrm{pdeg}(\delta_i) = e_i$, or equivalently $D(\mathcal{A}) \cong \bigoplus_{i=1}^{l} S(-e_i)$.

Let $\delta_1, \ldots, \delta_l \in D(\mathcal{A})$. Then $\det(\delta_i(x_j))$ is divisible by $Q(\mathcal{A})$. One of the most famous characterizations of freeness is due to Saito [1980] and it uses the determinant of the coefficient matrix of $\delta_1, \ldots, \delta_l$.

Theorem 3.2 (Saito's criterion). *Let $\delta_1, \ldots, \delta_l \in D(\mathcal{A})$. Then the following facts are equivalent*:

(1) *$D(\mathcal{A})$ is free with basis $\delta_1, \ldots, \delta_l$, i.e., $D(\mathcal{A}) = S \cdot \delta_1 \oplus \cdots \oplus S \cdot \delta_l$.*

(2) *$\det(\delta_i(x_j)) = cQ(\mathcal{A})$, where $c \in K \setminus \{0\}$.*

(3) *$\delta_1, \ldots, \delta_l$ are linearly independent over S and $\sum_{i=1}^{l} \operatorname{pdeg}(\delta_i) = n$.*

Given a simple graph G, we can define the *graphical arrangement* $\mathcal{A}(G)$; see [Orlik and Terao 1992]. Stanley [2007], showed that $\mathcal{A}(G)$ is free if and only if G is a chordal graph. See also [Suyama and Tsujie 2019] and [Suyama et al. 2019] for more general results.

We verify this result for a given graphical arrangement.

```
/**/ use S ::= QQ[x, y, z, w];
/**/ G := [[1, 2], [1, 3], [1, 4], [2, 4], [3, 4]];
/**/ A := ArrGraphical(S, G);
/**/ ArrDerModule(A);
matrix( /*RingWithID(18935, "QQ[x, y, z, w]")*/
 [[1, 0, 0, 0],
  [1, x-y, 0, 0],
  [1, x-z, x*z-z^2-x*w+z*w, x*y-y*z-x*w+z*w],
  [1, x-w, 0, x*y-x*w-y*w+w^2]])
/**/ IsArrFree(A);
true
/**/ ArrExponents(A);
[0,  1,  2,  2]
/**/ B := ArrDeletion(A, x-w);
/**/ IsArrFree(B);
false
```

4. ALGEBRAS. Orlik and Terao [1994] introduced a commutative analogue of the Orlik–Solomon algebra in order to answer a question of Aomoto related to cohomology groups of a certain "twisted" de Rham chain complex. The crucial difference between the Orlik–Solomon algebra and Orlik–Terao algebra is not just the difference between the exterior algebra and symmetric algebra, but rather the fact that the Orlik–Terao algebra actually captures the "coefficients" of the dependencies among the hyperplanes.

Let $\mathcal{A} = \{H_1, \dots, H_n\}$ be a central arrangement in V and $\Lambda \subseteq \{1, \dots, n\}$. If $\operatorname{codim}\left(\bigcap_{i\in\Lambda} H_i\right) < |\Lambda|$, then we say that Λ is *dependent*. If Λ is dependent, then there exist $c_i \in K$ such that

$$\sum_{i\in\Lambda} c_i \alpha_i = 0.$$

Definition 4.1. Let $R = K[y_1, \dots, y_n]$. For each dependent set $\Lambda = \{i_1, \dots, i_k\}$, let $r_\Lambda = \sum_{j=1}^{k} c_{i_j} y_{i_j} \in R$. Define now

$$f_\Lambda = \partial(r_\Lambda) = \sum_{j=1}^{k} c_{i_j} (y_{i_1} \cdots \hat{y}_{i_j} \cdots y_{i_k}),$$

where $\hat{y}_{i_j}$ means that the variable y_{i_j} is omitted, and let I be the ideal of R generated by the f_Λ. This ideal is called the *Orlik–Terao ideal* of $\mathcal{A}$. The *Orlik–Terao algebra* $\mathrm{OT}(\mathcal{A})$ is the quotient R/I. The *Artinian Orlik–Terao algebra* $\mathrm{AOT}(\mathcal{A})$ is the quotient of $\mathrm{OT}(\mathcal{A})$ by the square of the variables.

These algebras and their Betti diagrams give us a lot of information on the given arrangement, for example about its *formality*; see for example [Schenck and Tohăneanu 2009].

We can construct the Orlik–Terao ideal, its Artinian version and the Betti diagram of the Orlik–Terao algebra of the *Braid arrangement* as follows:

```
/**/ use S ::= QQ[x, y, z];
/**/ A := ArrBraid(S, 3);
/**/ OT_A := OrlikTeraoIdeal(A); OT_A;
ideal(y[1]*y[2]-y[1]*y[3]+y[2]*y[3])
/**/ PrintBettiDiagram(RingOf(OT_A)/OT_A);
         0     1
---------------
  0:     1     -
  1:     -     1
---------------
Tot:     1     1
/**/ ArtinianOrlikTeraoIdeal(A);
ideal(y[1]*y[2]-y[1]*y[3]+y[2]*y[3],  y[1]^2,  y[2]^2,
      y[3]^2)
```

In [Abe et al. 2018], the authors introduced a new algebra associated to a central hyperplane arrangement. This algebra can be considered as a generalization of the coinvariant algebras in the setting of hyperplane arrangements and it contains the cohomology rings of regular nilpotent Hessenberg varieties.

Definition 4.2. Let $\mathcal{A}$ be a central arrangement in V and $f \in S$ a homogeneous polynomial. Then the ideal

$$\mathfrak{a}(\mathcal{A}, f) = \{\delta(f) \mid \delta \in D(\mathcal{A})\}$$

is called the *Solomon–Terao ideal* of $\mathcal{A}$ with respect to f. The *Solomon–Terao algebra* of $\mathcal{A}$ with respect to f is the quotient $\mathrm{ST}(\mathcal{A}, f) = S/\mathfrak{a}(\mathcal{A}, f)$.

We can construct the Solomon–Terao ideal of the *reflection arrangement of type D* with respect to f, the sum of the square of the variables, as follows:

```
/**/ use S ::= QQ[x, y, z];
/**/ A := ArrTypeD(S, 3);
/**/ f := x^2+y^2+z^2;
/**/ SolomonTeraoIdeal(A, f);
ideal(2*x^2+2*y^2+2*z^2, 6*x*y*z,
      2*x^2*y^2-2*y^4+2*x^2*z^2-2*z^4)
```

5. MULTIARRANGEMENTS OF HYPERPLANES. A *multiarrangement* is a pair $(\mathcal{A}, m)$ of an arrangement $\mathcal{A}$ with a map $m : \mathcal{A} \to \mathbb{Z}_{\geq 0}$, called the *multiplicity*. An arrangement $\mathcal{A}$ can be identified with a multiarrangement with constant multiplicity $m \equiv 1$, and it is sometimes called a *simple arrangement*. Define $Q(\mathcal{A}, m) = \prod_{i=1}^{n} \alpha_i^{m(H_i)}$ and $|m| = \sum_{i=1}^{n} m(H_i)$. With this notation, the main object is the *module of vector fields logarithmic tangent* to $\mathcal{A}$ with multiplicity m (logarithmic vector field) defined by

$$D(\mathcal{A}, m) = \{\delta \in \mathrm{Der}_V \mid \delta(\alpha_i) \in \langle \alpha_i \rangle^{m(H_i)} S, \text{ for all } i\}.$$

The module $D(\mathcal{A}, m)$ is a graded S-module. In general, in contrast to the case of simple arrangements, $D(\mathcal{A}, m)$ does not coincide with

$$\{\delta \in \mathrm{Der}_V \mid \delta(Q(\mathcal{A})) \in \langle Q(\mathcal{A}, m) \rangle S\}.$$

Definition 5.1. Let $\mathcal{A}$ be a central arrangement. The multiarrangement $(\mathcal{A}, m)$ is said to be *free with exponents* $(e_1, \ldots, e_l)$ if and only if $D(\mathcal{A}, m)$ is a free S-module and there exists a basis $\delta_1, \ldots, \delta_l \in D(\mathcal{A}, m)$ such that $\mathrm{pdeg}(\delta_i) = e_i$, or equivalently $D(\mathcal{A}, m) \cong \bigoplus_{i=1}^{l} S(-e_i)$.

As for simple arrangements, if $\delta_1, \ldots, \delta_l \in D(\mathcal{A}, m)$, then $\det(\delta_i(x_j))$ is divisible by $Q(\mathcal{A}, m)$. Moreover, we can generalize Theorem 3.2; see [Ziegler 1989].

Theorem 5.2 (generalized Saito's criterion). *Let* $\delta_1, \ldots, \delta_l \in D(\mathcal{A}, m)$. *Then the following are equivalent*:

(1) $D(\mathcal{A}, m)$ *is free with basis* $\delta_1, \ldots, \delta_l$, *i.e.*, $D(\mathcal{A}, m) = S \cdot \delta_1 \oplus \cdots \oplus S \cdot \delta_l$.

(2) $\det(\delta_i(x_j)) = cQ(\mathcal{A}, m)$, *where* $c \in K \setminus \{0\}$.

(3) $\delta_1, \ldots, \delta_l$ *are linearly independent over* S *and* $\sum_{i=1}^{l} \mathrm{pdeg}(\delta_i) = |m|$.

Given a simple arrangement $\mathcal{A}$ and H one of its hyperplanes, we can naturally define *Ziegler's multirestriction* (see [Ziegler 1989]) as the multiarrangement $(\mathcal{A}^H, m^H)$, where the function $m^H : A^H \to \mathbb{Z}_{>0}$ is defined by

$$X \in \mathcal{A}^H \mapsto \#\{H' \in \mathcal{A} \mid H' \supset X\} - 1.$$

Theorem 5.3 [Ziegler 1989]. *Let* $\mathcal{A}$ *be a central arrangement. If* $\mathcal{A}$ *is free with exponents* $(1, e_2, \ldots, e_l)$, *then* $(\mathcal{A}^H, m^H)$ *is free with exponents* $(e_2, \ldots, e_l)$, *for any* $H \in \mathcal{A}$.

In general, the converse is false. However, we have the following:

Theorem 5.4 [Yoshinaga 2004]. *Assume* $l \geq 4$. *Let* $\mathcal{A}$ *be a central arrangement and* $H \in \mathcal{A}$. *Then* $\mathcal{A}$ *is free with exponents* $(1, e_2, \ldots, e_l)$ *if and only if the following conditions are satisfied*:

(1) $\mathcal{A}$ *is locally free along* H, *i.e.*, $\mathcal{A}_X$ *is free for any* $X \in L(\mathcal{A})$ *with* $X \subset H$ *and* $X \neq \varnothing$.

(2) *Ziegler's multirestriction* $(\mathcal{A}^H, m^H)$ *is a free multiarrangement with exponents* $(e_2, \ldots, e_l)$.

We can construct Ziegler's multirestriction of a given arrangement and check its freeness as follows:

```
/**/ use S ::= QQ[x, y, z];
/**/ A := [x, y, z, x-y, x-y-z, x-y+2*z];
/**/ A_1 := MultiArrRestrictionZiegler(A, z); A_1;
[[y[1], 1], [y[2], 1], [y[1]-y[2], 3]]
/**/ IsMultiArrFree(A_1);
true
/**/ MultiArrDerModule(A_1);
matrix( /*RingWithID(18, "QQ[y[1], y[2]]")*/
 [[y[1]*y[2], y[1]^3],
  [y[1]*y[2], 3*y[1]^2*y[2]-3*y[1]*y[2]^2+y[2]^3]])
/**/ MultiArrExponents(A_1);
[2, 3]
/**/ ArrExponents(A);
[1, 2, 3]
```

SUPPLEMENT. The online supplement contains version 1.0 of `arrangements` for CoCoA-5.2.4.

REFERENCES.

[Abbott and Bigatti 2018] J. Abbott and A. M. Bigatti, "Gröbner bases for everyone with CoCoA-5 and CoCoALib", pp. 1–24 in *The 50th anniversary of Gröbner bases*, edited by T. Hibi, Adv. Stud. Pure Math. **77**, Math. Soc. Japan, Tokyo, 2018. MR Zbl

[Abe 2016] T. Abe, "Divisionally free arrangements of hyperplanes", *Invent. Math.* **204**:1 (2016), 317–346. MR Zbl

[Abe et al. 2018] T. Abe, T. Maeno, S. Murai, and Y. Numata, "Solomon–Terao algebra of hyperplane arrangements", 2018. arXiv

[Ardila 2007] F. Ardila, "Computing the Tutte polynomial of a hyperplane arrangement", *Pacific J. Math.* **230**:1 (2007), 1–26. MR Zbl

[Bigatti et al. 2019] A. M. Bigatti, E. Palezzato, and M. Torielli, "New characterizations of freeness for hyperplane arrangements", *Journal of Algebraic Combinatorics* (online publication March 2019).

[CoCoA] J. Abbott and A. M. Bigatti, "CoCoA: a system for doing Computations in Commutative Algebra", available at http://cocoa.dima.unige.it.

[CoCoALib] J. Abbott and A. M. Bigatti, "CoCoALib: a C++ library for doing Computations in Commutative Algebra", available at http://cocoa.dima.unige.it/cocoalib.

[Crapo and Rota 1970] H. H. Crapo and G.-C. Rota, *On the foundations of combinatorial theory: Combinatorial geometries*, The M.I.T. Press, Cambridge, MA, 1970. MR Zbl

[Orlik and Solomon 1980] P. Orlik and L. Solomon, "Combinatorics and topology of complements of hyperplanes", *Invent. Math.* **56**:2 (1980), 167–189. MR Zbl

[Orlik and Terao 1992] P. Orlik and H. Terao, *Arrangements of hyperplanes*, Grundlehren der Mathematischen Wissenschaften **300**, Springer, 1992. MR Zbl

[Orlik and Terao 1994] P. Orlik and H. Terao, "Commutative algebras for arrangements", *Nagoya Math. J.* **134** (1994), 65–73. MR Zbl

[Palezzato and Torielli 2018] E. Palezzato and M. Torielli, "Free hyperplane arrangements over arbitrary fields", 2018. arXiv

[Saito 1980] K. Saito, "Theory of logarithmic differential forms and logarithmic vector fields", *J. Fac. Sci. Univ. Tokyo Sect. IA Math.* **27**:2 (1980), 265–291. MR Zbl

[Schenck and Tohăneanu 2009] H. Schenck and c. O. Tohăneanu, "The Orlik–Terao algebra and 2-formality", *Math. Res. Lett.* **16**:1 (2009), 171–182. MR Zbl

[Stanley 2007] R. P. Stanley, "An introduction to hyperplane arrangements", pp. 389–496 in *Geometric combinatorics*, edited by V. R. Ezra Miller and B. Sturmfels, IAS/Park City Math. Ser. **13**, Amer. Math. Soc., Providence, RI, 2007. MR Zbl

[Suyama and Tsujie 2019] D. Suyama and S. Tsujie, "Vertex-Weighted Graphs and Freeness of ψ-Graphical Arrangements", *Discrete Comput. Geom.* **61**:1 (2019), 185–197. MR

[Suyama et al. 2019] D. Suyama, M. Torielli, and S. Tsujie, "Signed graphs and the freeness of the Weyl subarrangements of type B_ℓ", *Discrete Math.* **342**:1 (2019), 233–249. MR Zbl

[Terao 1980] H. Terao, "Arrangements of hyperplanes and their freeness, I", *J. Fac. Sci. Univ. Tokyo Sect. IA Math.* **27**:2 (1980), 293–312. MR Zbl

[Yoshinaga 2004] M. Yoshinaga, "Characterization of a free arrangement and conjecture of Edelman and Reiner", *Invent. Math.* **157**:2 (2004), 449–454. MR Zbl

[Yoshinaga 2014] M. Yoshinaga, "Freeness of hyperplane arrangements and related topics", *Ann. Fac. Sci. Toulouse Math.* (6) **23**:2 (2014), 483–512. MR Zbl

[Zaslavsky 1975] T. Zaslavsky, *Facing up to arrangements: face-count formulas for partitions of space by hyperplanes*, vol. 1, Mem. Amer. Math. Soc. **154**, Amer. Math. Soc., Providence, RI, 1975. MR Zbl

[Ziegler 1989] G. M. Ziegler, "Multiarrangements of hyperplanes and their freeness", pp. 345–359 in *Singularities* (Iowa City, IA, 1986)), edited by R. Randell, Contemp. Math. **90**, Amer. Math. Soc., Providence, RI, 1989. MR Zbl

RECEIVED: 29 Aug 2018 REVISED: 7 Feb 2019 ACCEPTED: 26 Feb 2019

ELISA PALEZZATO:

palezzato@math.sci.hokudai.ac.jp
Department of Mathematics, Hokkaido University, Sapporo, Japan

MICHELE TORIELLI:

torielli@math.sci.hokudai.ac.jp
Department of Mathematics, Hokkaido University, Sapporo, Japan

JSAG **9** (2019), 55–63
dx.doi.org/10.2140/jsag.2019.9.55

The Journal of Software for
Algebra and Geometry

Numerical implicitization

JUSTIN CHEN AND JOE KILEEL

ABSTRACT: We present the `NumericalImplicitization.m2` package for Macaulay2, which allows for user-friendly computation of the invariants of the image of a polynomial map, such as dimension, degree, and Hilbert function values. This package relies on methods of numerical algebraic geometry, including homotopy continuation and monodromy.

INTRODUCTION. Many varieties of interest in algebraic geometry and its applications are usefully described as images of polynomial maps, via a parametrization. Implicitization is the process of converting a parametric description of a variety into an intrinsic — or implicit — description. Classically, implicitization refers to the procedure of computing the defining equations of a parametrized variety, and in theory this is accomplished by finding the kernel of a ring homomorphism, via Gröbner bases. In practice however, symbolic Gröbner basis computations are often time consuming, even for medium scale problems, and do not scale well with respect to the size of the input.

Despite this, one would often like to know basic information about a parametrized variety, even when symbolic methods are prohibitively expensive (in terms of computation time). Examples of such information are discrete invariants such as the dimension, the degree, or Hilbert function values. Other examples include Boolean tests, for example whether or not a particular point lies on a parametrized variety. The goal of this [Macaulay2] package is to provide such information; in other words, to *numerically implicitize* a parametrized variety by using methods of numerical algebraic geometry. `NumericalImplicitization.m2` builds on top of existing numerical algebraic geometry software: NAG4M2 [Leykin 2011], Bertini [Bates et al.] and PHCpack [Verschelde 1999]. Each of these can be used for path tracking and point sampling; by default the native software `M2engine` in NAG4M2 is used. The latest version of the code and documentation can be found at https://github.com/Joe-Kileel/Numerical-Implicitization.

MSC2010: primary 14Q99; secondary 14-04, 65H10, 65H20.

Keywords: numerical algebraic geometry, implicitization, homotopy continuation, monodromy, interpolation.

`NumericalImplicitization.m2` version 2.1.0

NOTATION. The following notation will be used throughout this article:

- $X \subseteq \mathbb{A}^n$ is a *source variety*, defined by an ideal $I = \langle g_1, \ldots, g_r \rangle$ in the polynomial ring $\mathbb{C}[x_1, \ldots, x_n]$.
- $F : \mathbb{A}^n \to \mathbb{A}^m$ is a regular map sending $x \mapsto (f_1(x), \ldots, f_m(x))$, where $f_i \in \mathbb{C}[x_1, \ldots, x_n]$.
- Y is the Zariski closure of the image $\overline{F(X)} = \overline{F(V(I))} \subseteq \mathbb{A}^m$, the *target variety* under consideration.
- $\widetilde{Y} \subseteq \mathbb{P}^m$ is the projective closure of Y, with respect to the standard embedding $\mathbb{A}^m \subseteq \mathbb{P}^m$.

Currently, `NumericalImplicitization` is implemented for integral varieties X. Equivalently, the ideal I is prime. Since numerical methods are used, we always work with a floating-point representation for complex numbers. Moreover, $\widetilde{Y}$ is internally represented by its affine cone. This is because it is easier to work with affine, as opposed to projective, coordinates; at the same time, this suffices to find the invariants of $\widetilde{Y}$.

SAMPLING. All the methods in this package rely on the ability to sample general points on X. To this end, the method `numericalSourceSample` is provided to allow the user to sample general points on X. This method works by computing a witness set for X, via a numerical irreducible decomposition of I — once this is known, points on X can be quickly sampled.

One way to view the difference in computation time between symbolic and numerical methods is that the upfront cost of computing a Gröbner basis is replaced with the upfront cost of computing a numerical irreducible decomposition, which is used to sample general points. However, if $X = \mathbb{A}^n$, then sampling is done by generating random tuples, so in this unrestricted (or rational) parametrization case, the upfront cost of numerical methods becomes negligible. Another situation where the cost of computing a numerical irreducible decomposition can be avoided is if the user can provide a single point on X: in this case, `numericalSourceSample` can use the given point to quickly generate new general points on X via path tracking.

DIMENSION. The most basic invariant of an algebraic variety is its dimension. To compute the dimension of the image of a variety numerically, we use the following theorem:

Theorem 1 [Hartshorne 1977, III.10.4–10.5]. *Let $F : X \to Y$ be a dominant morphism of irreducible varieties over $\mathbb{C}$. Then there is a Zariski open subset $U \subseteq X$ such that for all $x \in U$, the induced map on tangent spaces $dF_x : T_x X \to T_{F(x)} Y$ is surjective.*

In the setting above, since the singular locus $\operatorname{Sing} Y$ is a proper closed subset of Y, for general $y = F(x) \in Y$,

$$\dim Y = \dim T_y Y = \dim dF_x(T_x X) = \dim T_x X - \dim \ker dF_x.$$

Now $T_x X$ is the kernel of the Jacobian matrix of I evaluated at x, given by

$$\operatorname{Jac}(I)(x) = ((\partial g_i/\partial x_j)(x))_{1\le i\le r,\, 1\le j\le n},$$

and $\ker dF_x$ is the kernel of the Jacobian of F evaluated at x, intersected with $T_x X$. Explicitly, $\ker dF_x$ is the kernel of the $(r+m)\times n$ matrix:

$$\begin{bmatrix} \operatorname{Jac}(I)(x) \\ \operatorname{Jac}(F)(x) \end{bmatrix} = \begin{bmatrix} \frac{\partial g_1}{\partial x_1}(x) & \cdots & \frac{\partial g_1}{\partial x_n}(x) \\ \vdots & \ddots & \vdots \\ \frac{\partial g_r}{\partial x_1}(x) & \cdots & \frac{\partial g_r}{\partial x_n}(x) \\ \frac{\partial f_1}{\partial x_1}(x) & \cdots & \frac{\partial f_1}{\partial x_n}(x) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1}(x) & \cdots & \frac{\partial f_m}{\partial x_n}(x) \end{bmatrix}.$$

We compute these kernel dimensions numerically to obtain $\dim Y$.

Example 2. Let $Y \subseteq \operatorname{Sym}^4(\mathbb{C}^5) \cong \mathbb{A}^{70}$ be the variety of $5\times 5\times 5\times 5$ symmetric tensors of border rank ≤ 14. Equivalently, Y is the affine cone over $\sigma_{14}(\nu_4(\mathbb{P}^4))$, the 14th secant variety of the fourth Veronese embedding of $\mathbb{P}^4$. Naively, one expects $\dim(Y) = 14\cdot 4 + 13 + 1 = 70$. In fact, $\dim(Y) = 69$ as verified by the following code:

```
Macaulay2, version 1.13
i1 : needsPackage "NumericalImplicitization"
i2 : R=CC[s_(1,1)..s_(14,5)];
i3 : F=sum(1..14,i->basis(4,R,Variables=>toList(s_(i,1)..s_(i,5))));
i4 : elapsedTime numericalImageDim(F,ideal 0_R)
     -- 0.0767826 seconds elapsed
o4 = 69
```

This example is the largest exceptional case from the celebrated work [Alexander and Hirschowitz 1995].

HILBERT FUNCTION. We now turn to the problem of determining the Hilbert function of $\widetilde{Y}$. If $\widetilde{Y} \subseteq \mathbb{P}^m$ is a projective variety given by a homogeneous ideal $J \subseteq \mathbb{C}[y_0, \ldots, y_m]$, then the Hilbert function of $\widetilde{Y}$ at an argument $d \in \mathbb{N}$ is by definition the vector space dimension of the d-th graded part of J, namely $H_{\widetilde{Y}}(d) := \dim J_d$. This counts the maximum number of linearly independent degree d hypersurfaces in $\mathbb{P}^m$ containing $\widetilde{Y}$.

To compute the Hilbert function of $\widetilde{Y}$ numerically, we use *multivariate polynomial interpolation*. For a fixed argument $d \in \mathbb{N}$, let $\{p_1, \ldots, p_N\}$ be a set of N general points on $\widetilde{Y}$. For $1 \leq i \leq N$, consider an $i \times \binom{m+d}{d}$ interpolation matrix $A^{(i)}$ with rows indexed by points $\{p_1, \ldots, p_i\}$ and columns indexed by degree d monomials in $\mathbb{C}[y_0, \ldots, y_m]$, whose entries are the values of the monomials at the points. A vector in the kernel of $A^{(i)}$ corresponds to a choice of coefficients for a homogeneous degree d polynomial that vanishes on $p_1, \ldots, p_i$. If i is large, then one expects such a form to vanish on the entire variety $\widetilde{Y}$. The following theorem makes this precise:

Theorem 3. *Let* $\{p_1, \ldots, p_{s+1}\}$ *be a set of general points on* $\widetilde{Y}$, *and let* $A^{(i)}$ *be the interpolation matrix above. If* $\dim\ker A^{(s)} = \dim\ker A^{(s+1)}$, *then* $\dim\ker A^{(s)} = \dim J_d$.

Proof. Identifying $v \in \ker A^{(i)}$ with the form in $\mathbb{C}[y_0, \ldots, y_m]$ of degree d having v as its coefficients, it suffices to show that $\ker A^{(s)} = J_d$. If $h \in J_d$, then h vanishes on all of $\widetilde{Y}$, in particular on $\{p_1, \ldots, p_s\}$, so $h \in \ker A^{(s)}$. For the converse $\ker A^{(s)} \subseteq J_d$, we consider the universal interpolation matrices over $\mathbb{C}[y_{0,1},\ y_{1,1},\ \ldots,\ y_{m,i}]$:

$$\mathcal{A}^{(i)} := \begin{bmatrix} y_{0,1}^d & y_{0,1}^{d-1}y_{1,1} & \cdots & y_{m,1}^d \\ y_{0,2}^d & y_{0,2}^{d-1}y_{1,2} & \cdots & y_{m,2}^d \\ \vdots & \vdots & \ddots & \vdots \\ y_{0,i}^d & y_{0,i}^{d-1}y_{1,i} & \cdots & y_{m,i}^d \end{bmatrix}.$$

Set $r_i := \min\{r \in \mathbb{Z}_{\geq 0} \mid \text{all } (r+1)\text{-minors of } \mathcal{A}^{(i)} \text{ lie in the ideal of } \widetilde{Y}^{\times i} \subseteq (\mathbb{P}^m)^{\times i}\}$. Then any specialization of $\mathcal{A}^{(i)}$ to i points in $\widetilde{Y}$ is a matrix over $\mathbb{C}$ of rank $\leq r_i$; moreover if the points are general, then the specialization has rank exactly r_i, since $\widetilde{Y}$ is irreducible. In particular $\operatorname{rank}(A^s) = r_s$ and $\operatorname{rank}(A^{s+1}) = r_{s+1}$, so $\dim\ker A^{(s)} = \dim\ker A^{(s+1)}$ implies that $r_s = r_{s+1}$. It follows that specializing $\mathcal{A}^{(s+1)}$ to $p_1, p_2, \ldots, p_s, q$ for any $q \in \widetilde{Y}$ gives a rank r_s matrix. Hence, every degree d form in $\ker A^{(s)}$ evaluates to 0 at every $q \in \widetilde{Y}$. Since $\widetilde{Y}$ is reduced, we deduce that $\ker A^{(s)} \subseteq J_d$. □

It follows from Theorem 3 that the integers $\dim\ker A^{(1)}, \dim\ker A^{(2)}, \ldots$ decrease by exactly 1, until the first instance where they fail to decrease, at which point they stabilize: $\dim\ker A^{(i)} = \dim\ker A^{(s)}$ for $i \geq s$. This stable value is the value of the Hilbert function, $\dim\ker A^{(s)} = H_{\widetilde{Y}}(d)$. In particular, it suffices to compute $\dim\ker A^{(N)}$ for $N = \binom{m+d}{d}$, so one may assume the interpolation matrix is square. Although this may seem wasteful (as stabilization may have occurred with fewer rows), this is indeed what `numericalHilbertFunction` does, due to the algorithm used to compute kernel dimension numerically. To be precise, kernel

dimension is found via a singular value decomposition (SVD) — namely, if a gap (the ratio of consecutive singular values) exceeds the option `SVDGap` (with default value 10^5), then this is taken as an indication that all singular values past this gap are numerically zero. On example problems, it was observed that taking only one more additional row than was needed often did not reveal a satisfactory gap in singular values. In addition, numerical stability is improved via preconditioning on the interpolation matrices — namely, each row is normalized to have Euclidean norm 1 before computing the SVD. Furthermore, for increased computational efficiency, the option `UseSLP` allows for the usage of straight-line programs in creating interpolation matrices.

Example 4. Let X be a random canonical curve of genus 4 in $\mathbb{P}^3$, so X is the complete intersection of a random quadric and cubic. Let $F : \mathbb{P}^3 \dashrightarrow \mathbb{P}^2$ be a projection by three random cubics. Then $\widetilde{Y}$ is a plane curve of degree

$$3^{\dim(\widetilde{Y})} \cdot \deg(X) = 3 \cdot 2 \cdot 3 = 18,$$

so the ideal of $\widetilde{Y}$ contains a single form of degree 18. We verify this as follows:

```
i5 : R = CC[w_0..w_3]; I = ideal(random(2,R), random(3,R));
     F = toList(1..3)/(i -> random(3,R));
i8 : elapsedTime T = numericalHilbertFunction(F,I,18,Verbose=>false)
     -- 6.01226 seconds elapsed
o8 : a numerical interpolation table, indicating
     the space of degree 18 forms in the ideal of the image has
     dimension 1
```

The output is a `NumericalInterpolationTable`, which is a `HashTable` storing the results of the interpolation computation described above. From this, one can obtain a floating-point approximation to a basis of J_d. This is done via the command `extractImageEquations`:

```
i9 : extractImageEquations T
o9 : | -.0000712719y_0^18+(.000317507-.000100639i)y_0^17y_1- ... |
```

The option `AttemptZZ=>true` calls the Lenstra–Lenstra–Lovász algorithm to compute short equations over $\mathbb{Z}$.

DEGREE. After dimension, degree is the most basic invariant of a projective variety $\widetilde{Y} \subseteq \mathbb{P}^m$. Set $k := \dim(\widetilde{Y})$. For a general linear space $L \in \mathrm{Gr}(\mathbb{P}^{m-k}, \mathbb{P}^m)$ of complementary dimension to $\widetilde{Y}$, the intersection $L \cap \widetilde{Y}$ is a finite set of reduced points. The degree of $\widetilde{Y}$ is by definition the cardinality of $L \cap \widetilde{Y}$, which is independent of the general linear space L. Thus one way to find $\deg(\widetilde{Y})$ is to fix a random L_0 and compute the set of points $L_0 \cap \widetilde{Y}$.

`NumericalImplicitization` takes this approach, but the method used to find $L_0 \cap \widetilde{Y}$ is not the most obvious. First and foremost, we do not know the equations of $\widetilde{Y}$, so all solving must be done in X. Secondly, we do *not* compute $F^{-1}(L_0) \cap X$ from the equations of X and the equations of L_0 pulled back under F, because fibers of F may be positive-dimensional and of high degree. Instead, *monodromy* is employed to find $L_0 \cap \widetilde{Y}$.

To state the technique, we consider the map:

$$\{(L, y) \in \mathrm{Gr}(\mathbb{P}^{m-k}, \mathbb{P}^m) \times \widetilde{Y} \mid y \in L\} \subseteq \mathrm{Gr}(\mathbb{P}^{m-k}, \mathbb{P}^m) \times \widetilde{Y} \xrightarrow{\rho_1} \mathrm{Gr}(\mathbb{P}^{m-k}, \mathbb{P}^m),$$

where ρ_1 is projection onto the first factor. There is a nonempty Zariski open subset $U \subseteq \mathrm{Gr}(\mathbb{P}^{m-k}, \mathbb{P}^m)$ such that the restriction $\rho_1^{-1}(U) \to U$ is a $\deg(\widetilde{Y})$-to-1 covering map, namely U equals the complement of the Hurwitz divisor from [Sturmfels 2017]. For a fixed generic basepoint $L_0 \in U$, the fundamental group $\pi_1(U, L_0)$ acts on the fiber $\rho_1^{-1}(L_0) = L_0 \cap \widetilde{Y}$. This action is known as monodromy. It is a key fact that irreducibility of $\widetilde{Y}$ implies the group homomorphism

$$\pi_1(U, L_0) \to \mathrm{Sym}(L_0 \cap \widetilde{Y}) \cong \mathrm{Sym}_{\deg(\widetilde{Y})}$$

is surjective (see [Sommese and Wampler 2005, Theorem A.12.2]).

We compute the degree of $\widetilde{Y}$ by constructing a *pseudo-witness set* for $\widetilde{Y}$, which is a numerical representation of a parametrized variety (see [Hauenstein and Sommese 2010]). First, we sample a general point $x \in X$, and translate a general linear slice L_0 so that $F(x) \in L_0 \cap \widetilde{Y}$. Then L_0 is moved around in a random loop of the form described in [Sommese and Wampler 2005, Lemma 7.1.3]. This loop pulls back to a homotopy in X, where we use the equations of X to track x. The endpoint of the track is a point $x' \in X$ such that $F(x') \in L_0 \cap \widetilde{Y}$. If $F(x)$ and $F(x')$ are numerically distinct, then the loop has *learned* a new point in $L_0 \cap \widetilde{Y}$; otherwise x' is discarded. We then repeat this process of tracking points in X over each known point in $L_0 \cap \widetilde{Y}$, via new loops. In practice, if several consecutive loops do not learn new points in $L_0 \cap \widetilde{Y}$, then we suspect that all of $L_0 \cap \widetilde{Y}$ has been calculated. To verify this, we pass to the *trace test* (see [Sommese et al. 2002, Corollary 2.2]), which provides a characterization for when a subset of $L_0 \cap \widetilde{Y}$ equals $L_0 \cap \widetilde{Y}$. If the trace test is failed, then L_0 is replaced by a new random L_0' and preimages in X of known points of $L_0 \cap \widetilde{Y}$ are tracked to those preimages of points of $L_0' \cap \widetilde{Y}$. Afterwards, monodromy for $L_0' \cap \widetilde{Y}$ begins anew. If the trace test is failed `MaxAttempts` (by default 5) times, then the method exits with only a lower bound on $\deg(\widetilde{Y})$. To speed up computation, the option `MaxThreads` allows for loop tracking to be parallelized.

Example 5. Let $\widetilde{Y} = \sigma_2(\mathbb{P}^1 \times \mathbb{P}^1 \times \mathbb{P}^1 \times \mathbb{P}^1 \times \mathbb{P}^1) \subseteq \mathbb{P}^{31}$. We find that $\deg(\widetilde{Y}) = 3256$, using the commands below:

```
i10 : R = CC[a_1..a_5, b_1..b_5, t_0, t_1];
i11 : F1 = terms product(apply(toList(1..5), i -> 1 + a_i));
i12 : F2 = terms product(apply(toList(1..5), i -> 1 + b_i));
i13 : F = apply(toList(0..<2^5), i -> t_0*F1#i + t_1*F2#i);
i14 : elapsedTime pseudoWitnessSet(F, ideal 0_R, Repeats=>2,
      MaxThreads=>2)
Sampling point in source ...
Tracking monodromy loops ...
Points found: 2
Points found: 4
...
Points found: 3256
Running trace test ...
      -- 336.737 seconds elapsed
o14 = a pseudo-witness set, indicating
      the degree of the image is 3256
```

From [Raicu 2012, Theorem 4.1], it is known that the prime ideal J of $\widetilde{Y}$ is generated by the 3×3 minors of all flattenings of $2^{\times 5}$ tensors, so we can confirm that $\deg(J) = 3256$. However, the naive attempt to compute the degree of $\widetilde{Y}$ symbolically by taking the kernel of a ring map — from a polynomial ring in 32 variables — has no hope of finishing in any reasonable amount of time.

MEMBERSHIP. Classically, given a variety $Y \subseteq \mathbb{A}^m$ and a point $y \in \mathbb{A}^m$, we determine whether or not $y \in Y$ by finding set-theoretic equations of Y (which generate the ideal of Y up to radical), and then testing if y satisfies these equations. If a `PseudoWitnessSet` for Y is available, then point membership in Y can instead be verified by *parameter homotopy*. More precisely, `isOnImage` determines if y lies in the constructible set $F(X) \subseteq Y$, as follows. We fix a general affine linear subspace $L_y \subseteq \mathbb{A}^m$ of complementary dimension $m - \dim Y$ passing through y. Then $y \in F(X)$ if and only if $y \in L_y \cap F(X)$, so it suffices to compute the set $L_y \cap F(X)$. Now, a `PseudoWitnessSet` for Y provides a general section $L \cap F(X)$, and preimages in X. We move L to L_y as in [Sommese and Wampler 2005, Theorem 7.1.6]. This pulls back to a homotopy in X, where we use the equations of X to track the preimages. Applying F to the endpoints of the track gives all isolated points in $L_y \cap F(X)$ by [Sommese and Wampler 2005, Theorem 7.1.6]. Since L_y was general, the proof of [Eisenbud 1995, Corollary 10.5] shows $L_y \cap F(X)$ is zero-dimensional, so this procedure computes the entire set $L_y \cap F(X)$.

Example 6. Let $Y \subseteq \mathbb{A}^{18}$ be defined by the resultant of three quadratic equations in three unknowns. In other words, Y consists of all coefficients

$$(c_1, \ldots, c_6, d_1, \ldots, d_6, e_1, \ldots, e_6) \in \mathbb{A}^{18}$$

such that the system

$$0 = c_1x^2 + c_2xy + c_3xz + c_4y^2 + c_5yz + c_6z^2$$
$$0 = d_1x^2 + d_2xy + d_3xz + d_4y^2 + d_5yz + d_6z^2$$
$$0 = e_1x^2 + e_2xy + e_3xz + e_4y^2 + e_5yz + e_6z^2$$

admits a solution $(x : y : z) \in \mathbb{P}^2$. Here Y is a hypersurface, and a matrix formula for its defining equation was derived in [Eisenbud et al. 2003], using exterior algebra methods. We rapidly determine point membership in Y numerically as follows:

```
i15 : R = CC[c_1..c_6, d_1..d_6, e_1..e_6, x, y, z];
i16 : I = ideal(c_1*x^2+c_2*x*y+c_3*x*z+c_4*y^2+c_5*y*z+c_6*z^2,
                d_1*x^2+d_2*x*y+d_3*x*z+d_4*y^2+d_5*y*z+d_6*z^2,
                e_1*x^2+e_2*x*y+e_3*x*z+e_4*y^2+e_5*y*z+e_6*z^2);
i17 : F = toList(c_1..c_6 | d_1..d_6 | e_1..e_6);
i18 : W = pseudoWitnessSet(F, I, Verbose=>false); -- Y has degree 12
i19 : p1 = first numericalImageSample(F, I);
      p2 = point random(CC^1, CC^#F);
i21 : elapsedTime (isOnImage(W, p1), isOnImage(W, p2))
      -- used 0.186637 seconds
o21 = (true, false)
```

ACKNOWLEDGEMENTS. We are grateful to Anton Leykin for his encouragement, and to Luke Oeding for testing `NumericalImplicitization.m2`. We thank David Eisenbud and Bernd Sturmfels for helpful discussions, and the anonymous referee for insightful comments. This work has been partially supported by NSF DMS-1001867. Additionally, Kileel has been partially supported by the Simons Collaboration on Algorithms and Geometry.

SUPPLEMENT. `NumericalImplicitization.m2` version 2.1.0 is contained in the online supplement.

REFERENCES.

[Alexander and Hirschowitz 1995] J. Alexander and A. Hirschowitz, "Polynomial interpolation in several variables", *J. Algebraic Geom.* **4**:2 (1995), 201–222. MR

[Bates et al.] D. Bates, J. Hauenstein, A. Sommese, and C. Wampler, "Bertini: software for numerical algebraic geometry", available at http://www.nd.edu/∼sommese/bertini.

[Eisenbud 1995] D. Eisenbud, *Commutative algebra: with a view toward algebraic geometry*, Graduate Texts in Mathematics **150**, Springer, 1995. MR Zbl

[Eisenbud et al. 2003] D. Eisenbud, F.-O. Schreyer, and J. Weyman, "Resultants and Chow forms via exterior syzygies", *J. Amer. Math. Soc.* **16**:3 (2003), 537–579. MR

[Hartshorne 1977] R. Hartshorne, *Algebraic geometry*, Graduate Texts in Mathematics **52**, Springer, 1977. MR Zbl

[Hauenstein and Sommese 2010] J. D. Hauenstein and A. J. Sommese, "Witness sets of projections", *Appl. Math. Comput.* **217**:7 (2010), 3349–3354. MR Zbl

[Leykin 2011] A. Leykin, "Numerical algebraic geometry", *J. Softw. Algebra Geom.* **3** (2011), 5–10. MR Zbl

[Macaulay2] D. R. Grayson and M. E. Stillman, "Macaulay2, a software system for research in algebraic geometry", available at https://faculty.math.illinois.edu/Macaulay2/.

[Raicu 2012] C. Raicu, "Secant varieties of Segre–Veronese varieties", *Algebra Number Theory* **6**:8 (2012), 1817–1868. MR Zbl

[Sommese and Wampler 2005] A. J. Sommese and C. W. Wampler, II, *The numerical solution of systems of polynomials: arising in engineering and science*, World Scientific, Hackensack, NJ, 2005. MR

[Sommese et al. 2002] A. J. Sommese, J. Verschelde, and C. W. Wampler, "Symmetric functions applied to decomposing solution sets of polynomial systems", *SIAM J. Numer. Anal.* **40**:6 (2002), 2026–2046. MR Zbl

[Sturmfels 2017] B. Sturmfels, "The Hurwitz form of a projective variety", *J. Symbolic Comput.* **79**:1 (2017), 186–196. MR Zbl

[Verschelde 1999] J. Verschelde, "Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation", *ACM Trans. Math. Softw.* **25**:2 (June 1999), 251–276. Zbl

RECEIVED: 11 Oct 2016 REVISED: 17 Feb 2019 ACCEPTED: 11 Apr 2019

JUSTIN CHEN:

jchen646@math.gatech.edu
School of Mathematics, Georgia Institute of Technology, Atlanta, GA, United States

JOE KILEEL:

jkileel@math.princeton.edu
Program in Applied and Computational Mathematics, Princeton University, Princeton, NJ, United States

JSAG **9** (2019), 65–70
dx.doi.org/10.2140/jsag.2019.9.65

The Journal of Software for
Algebra and Geometry

Random Monomial Ideals: a Macaulay2 package

SONJA PETROVIĆ, DESPINA STASI AND DANE WILBURNE

ABSTRACT: The Macaulay2 package `RandomMonomialIdeals.m2` provides users with a set of tools that allow for the systematic generation and study of random monomial ideals. It also introduces new objects, `Sample` and `Model`, to allow for streamlined handling of random objects and their statistics in Macaulay2.

ERDŐS–RÉNYI RANDOM MONOMIAL IDEALS. Given their central role in commutative algebra and their inherent combinatorial structure (see, e.g., [Miller and Sturmfels 2005; Stanley 1996]), monomial ideals are a natural class of object to study probabilistically. This study was initiated in [De Loera et al. 2019], where random monomial ideals were produced from random sets of monomial generators in a manner inspired by the Erdős–Rényi model of random graphs. Working within the framework of such a model allows one to ask well-posed questions about the distributions of various algebraic invariants of interest.

The [Macaulay2] package `RandomMonomialIdeals.m2` implements several basic probabilistic models for monomial ideals based on the work in [De Loera et al. 2019]; it also computes summary statistics of (algebraic properties of) samples of monomial ideals, and sets up the framework to define new probabilistic models and generate samples from them using two new Macaulay2 types, `Sample` and `Model`.

The fundamental method in the package is `randomMonomialSets`, which randomly generates sets of monomials in a fixed number of variables up to a given degree from the Erdős–Rényi-type distribution $\mathcal{B}(n, D, p)$ defined in [De Loera et al. 2019], as well as various other related distributions.

In the following examples, we set the number of variables to `n=3` and sample size to `N=5`.

Each command in Table 1 returns a list of five sets of randomly generated monomials. In the case of $\mathcal{B}(n, D, M)$, if M is larger than the total number of monomials in n variables of degree at most D, all such monomials are returned.

MSC2010: primary 13F20; secondary 05E40.

Keywords: random monomial ideals, random commutative algebra, combinatorial commutative algebra, Macaulay2 package.

`RandomMonomialIdeals.m2` version 1.0

Model	Example of M2 command
$\mathcal{B}(n, D, p)$: select each monomial of degree $\leq D$ independently with probability $p \in [0, 1]$	`D=3; p=0.2; randomMonomialSets(n,D,p,N)`
$\mathcal{B}(n, D, \boldsymbol{p})$: select each monomial of degree $1 \leq d \leq D$ independently with probability $p_d \in [0, 1]$, where $\boldsymbol{p} = (p_1, \ldots, p_D)$ is a list of probabilities	`D=3; p = {0.5,0.0,0.1}; randomMonomialSets(n,D,p,N)`
$\mathcal{B}(n, D, M)$: select a set of M monomials of degree $\leq D$ uniformly at random all among such sets	`D=3; M=2; randomMonomialSets(n,D,M,N)`
$\mathcal{B}(n, D, \boldsymbol{M})$: randomly select M_d monomials of each degree $1 \leq d \leq D$	`D=4; M={1,0,3,0}; randomMonomialSets(n,D,M,N)`

Table 1. Examples of models of random monomial ideals implemented in the `RandomMonomialIdeals.m2` Macaulay2 package (left) and the corresponding M2 commands to produce random instances for each model (right).

One can also force the monomial sets to be minimal generating sets as follows.

```
i1 : n=3; D=4; N=4; p={0.5,0.0,1.0,0.0};
     netList pack(4,randomMonomialSets(n,D,p,N, Strategy=>"Minimal"))
      +------------+-----------------------+-----------------------+------------+
      |          3 |      3   2       2   3 |      3   2       2   3 |            |
o1 =  |{x , x , x }|{x , x , x x , x x , x }|{x , x , x x , x x , x }|{x , x , x }|
      |  1   2   3 |  3   1   1 2   1 2   2 |  1   2   2 3   2 3   3 |  1   2   3 |
      +------------+-----------------------+-----------------------+------------+
```

In the above sample of four sets of monomials, there are no monomials of degrees 2 or 4. Each of the variables was selected with probability 0.5. All monomials of degree 3 were selected, but of course some are not included as they are not minimal generators of the corresponding monomial ideal.

The distributions that we have described above and, in fact, any probability distribution on sets of monomials, naturally induce distributions on monomial ideals. The distribution on ideals induced by $\mathcal{B}(n, D, p)$ is denoted by $\mathcal{I}(n, D, p)$. Samples of monomial ideals for each of the models in Table 1 can be generated as follows:

```
i2 : n=2;D=5;p=0.2;N=3; randomMonomialIdeals(n,D,p,N)
                                          3
o2 = {monomialIdeal(x ), monomialIdeal (x , x x ), monomialIdeal(x x )}
                     2                    1   1 2                  1 2
o2 : List
```

SUMMARY STATISTICS. Given a list of monomial ideals, the package contains an array of methods for computing and summarizing various algebraic invariants of the ideals in the list: Krull dimension, degree, projective dimension, Castelnuovo–Mumford regularity, Betti tables and Betti shapes, and the proportion of ideals which are Borel-fixed or Cohen–Macaulay.

For instance, `dimStats` is a method which computes the Krull dimension of $k[x_1, \ldots, x_n]/I$ for each monomial ideal I in the list and returns the mean and standard deviation of the sample. When the optional input `ShowTally` is set to `true`, `dimStats` also returns a histogram of the Krull dimensions of the ideals in the list.

```
i3 : B = randomMonomialIdeals(3,10,0.01,1000);
i4 : dimStats(B, ShowTally=>true)
o4 = (1.92, .46, Tally{0 => 1  })
                       1 => 146
                       2 => 785
                       3 => 68
o4 : Sequence
```

In this sample of $N = 1000$ monomial ideals from the distribution $\mathcal{J}(3, 10, 0.01)$, the proportion of ideals with Krull dimension 0, 1, 2, 3 was 0.001, 0.146, 0.785, 0.068, respectively. By [De Loera et al. 2019, Theorem 3.2], the probability that a monomial ideal from this distribution has Krull dimension t for $t = 0, 1, 2, 3$ is 0.0009, 0.1458, 0.7963, 0.570, respectively. By the same theorem, the expected Krull dimension in this case is 1.9094 whereas the observed sample mean in the example is 1.92.

Each method that computes sample statistics of a particular invariant or property of a list monomial ideals can also be applied more generally to any list of algebraic objects for which that invariant or property is defined, whether or not the objects were generated using the ER-model.

NEW TYPES FOR CREATING AND STORING PROBABILISTIC MODELS AND SAMPLES IN MACAULAY2. The package comes equipped with the predefined Erdős–Rényi-type model. For example, let us consider the graded ER-type model with $\boldsymbol{p} = (0.1, 0, 0.2)$ in four variables and degree bound $D = 3$. We store this in an object of class `Model`:

```
i5 : myModel = ER(ZZ/101[a..d],3,{0.1,0.0,0.2})
o5 = Model{Generate => {*Function[RandomMonomialIdeals.m2:168:22-168:46]*}}
           Name => Erdos-Renyi
                        ZZ
           Parameters => (---[a, b, c, d], 3, {.1, 0, .2})
                        101
o5 : Model
```

It is now easy to obtain a sample of 1000 monomial ideals from this model. Note that model parameters are also stored with the sample object for easy access.

```
i6 : time mySample = sample(myModel,1000);
     -- used 2.32541 seconds
i7 : mySample.ModelName
o7 = Erdos-Renyi
i8 : mySample.Parameters
        ZZ
o8 = (---[a, b, c, d], 3, {.1, 0, .2})
       101
o8 : Sequence
i9 : mySample.SampleSize
o9 = 1000
```

The raw data (i.e., actual sets of monomials without the parameter values etc.) from the sample can be loaded as follows:

```
i10 :  time myIdeals = getData mySample;
     -- used 0.00005 seconds
i11 : myIdeals_0
        2      2     2
o11 = {b d, b*d , c*d }
o11 : List
```

To obtain statistics on any algebraic property of interest for the given sample, one simply runs the `statistics` command on the sample. Let us look at the distribution of Krull dimensions for this sample of 1000 monomial ideals:

```
i12 : time statistics(mySample, dim@@ideal)
     -- used 0.325259 seconds
                          2053
o12 = HashTable{Mean => ----, StdDev=> .654363, Histogram=> Tally{0 => 5  }}
                          1000                                    1 => 166
                                                                  2 => 608
                                                                  3 => 213
                                                                  4 => 8
o12 : HashTable
```

In addition, the command `writeSample(Sample,String)` can be used to write a sample to disk (the string is the filename), a feature that will be useful when large samples are generated and their statistics take a lot of computational time (e.g., Gröbner bases or free resolutions). This command creates a folder in which the model and data are stored. The sample can then be read via calls to the `sample(String)` method; for more details, the user is referred to the package documentation.

The new types, `Model` and `Sample`, along with the `statistics` function, allow one to define a new way to sample random algebraic objects, store the data as a

proper statistical sample, and study their algebraic properties under the probabilistic regime. Here is a simple example of a model that generates M polynomials in n variables of degree D randomly using Macaulay2's built-in `random` function:

```
i13: f=(D,n,M)->(R=QQ[x_1..x_n];apply(M,i->random(D,R)))
o13=f
o13: FunctionClosure
i14: myModel = model({2,3,4},f,"rand(D,n,M):
        M random polynomials in n variables of degree D")
o14=Model{Generate=>{*Function[RandomMonomialIdeals.m2:107:22-107:37]*}    }
          Name=>rand(D,n,M): M random polynomials in n variables of degree D
          Parameters=>{2, 3, 4}
o14: Model
i15: mySample = sample(myModel,10);
```

The last line produces a sample of size 10 from `myModel`. One can use the method `statistics` to generate an ideal from each of the ten sets in the sample, compute the Gröbner basis, and report its size:

```
i16 : statistics(mySample, numcols@@gens@@gb@@ideal)
o16 = HashTable{Histogram => Tally{6 => 10}}
                Mean => 6
                StdDev => 0
o16 : HashTable
```

Any function that can be run through `tally` can also serve as input for the `statistics` method; for more extensive examples, the user is directed to the package documentation.

ACKNOWLEDGEMENTS. The following undergraduate students made valuable contributions to the development of the package during summer of 2017: Genevieve Hummel, Parker Joncus, Daniel Kosmas, Richard Osborn, Monica Yun, and Tanner Zielinski. The project is funded by the NSF grant NSF-DMS-1522662, the Illinois Institute of Technology College of Science student summer research stipend, and the McMorris student summer research stipend from the Department of Applied Mathematics at Illinois Institute of Technology.

SUPPLEMENT. Version 1.0 of `RandomMonomialIdeals.m2` is contained in the online supplement.

REFERENCES.

[De Loera et al. 2019] J. A. De Loera, S. Petrović, L. Silverstein, D. Stasi, and D. Wilburne, "Random monomial ideals", *J. Algebra* **519** (2019), 440–473. MR Zbl

[Macaulay2] D. R. Grayson and M. E. Stillman, "Macaulay2, a software system for research in algebraic geometry", available at https://faculty.math.illinois.edu/Macaulay2/.

[Miller and Sturmfels 2005] E. Miller and B. Sturmfels, *Combinatorial commutative algebra*, Graduate Texts in Mathematics **227**, Springer, 2005. MR Zbl

[Stanley 1996] R. P. Stanley, *Combinatorics and commutative algebra*, 2nd ed., Progress in Mathematics **41**, Birkhäuser, Boston, 1996. MR Zbl

RECEIVED: 27 Nov 2017 REVISED: 27 Feb 2019 ACCEPTED: 11 Apr 2019

SONJA PETROVIĆ:

sonja.petrovic@iit.edu
Department of Applied Mathematics, Illinois Institute of Technology, Chicago, IL, United States

DESPINA STASI:

stasdes@iit.edu
Department of Applied Mathematics, Illinois Institute of Technology, Chicago, IL, United States

DANE WILBURNE:

drw@yorku.ca
Department of Mathematics and Statistics, York University, Toronto ON, Canada

JSAG **9** (2019), 71–80
dx.doi.org/10.2140/jsag.2019.9.71

The Journal of Software for
Algebra and Geometry

Calculations involving symbolic powers

BEN DRABKIN, ELOÍSA GRIFO,
ALEXANDRA SECELEANU AND BRANDEN STONE

ABSTRACT: Symbolic powers is a classical commutative algebra topic that relates to primary decomposition, consisting, in some circumstances, of the functions that vanish up to a certain order on a given variety. However, these are notoriously difficult to compute, and there are seemingly simple questions related to symbolic powers that remain open even over polynomial rings. In this paper, we describe a Macaulay2 software package that allows for computations of symbolic powers of ideals and which can be used to study the equality and containment problems, among others.

1. INTRODUCTION. Given an ideal I in a Noetherian domain R, the *n-th symbolic power of* I is the ideal defined by

$$I^{(n)} = \bigcap_{P \in \mathrm{Ass}(I)} (I^n R_P \cap R). \tag{1-1}$$

When I has no embedded primes, the minimal primes of I^n coincide with the associated primes of I, and $I^{(n)}$ as above corresponds to the intersection of the primary components corresponding to minimal primes of I^n. In particular, under these circumstances the definition is unchanged if instead we have P ranging over the set of minimal associated primes $\mathrm{Min}(I)$. However, if we consider any ideal I, with no assumptions on its associated primes, there are two possible notions of symbolic powers: the one above and the one given by

$$I^{(n)} = \bigcap_{P \in \mathrm{Min}(I)} (I^n R_P \cap R). \tag{1-2}$$

The `SymbolicPowers.m2` package allows the user to compute the symbolic powers of any ideal over a polynomial ring, using the definition of symbolic powers given in (1-1) as the standard, but allowing the user to take the definition in (1-2) instead via the option `UseMinimalPrimes`. This option can be used in any method included in the package.

MSC2010: primary 13P99; secondary 13A15, 13C99.
Keywords: symbolic powers, Macaulay2.
`SymbolicPowers.m2` version 2.0

Symbolic powers are a classical topic that relates to many subjects within commutative algebra and algebraic geometry, and is an active area of current research. If P is a prime ideal in a regular ring, the classical Zariski–Nagata theorem [Zariski 1949; Nagata 1962] says that the symbolic powers of P consist of the functions that vanish up to order n in the corresponding variety. For a polynomial ring over a perfect field, these coincide with differential powers. For a survey on symbolic powers, see [Dao et al. 2018].

Various invariants have been defined to compare symbolic and ordinary powers of ideals: the resurgence [Bocci and Harbourne 2010], the Waldschmidt constant [Bocci and Harbourne 2010], and the symbolic defect [Galetto et al. 2019], among others. Using the `SymbolicPowers.m2` package, these can be in some cases explicitly computed and in others approximated.

2. BASIC USAGE. The main method in the `SymbolicPowers.m2` package is `symbolicPower`, which takes as inputs an ideal I and an integer n and returns $I^{(n)}$. Computations are done using the standard definition of symbolic powers; if the option `UseMinimalPrimes` is set true, then the definition of symbolic powers used in the computations will be the nonstandard one, as described in the introduction. When `UseMinimalPrimes` is set `true`, the algorithm takes a primary decomposition of I^n and intersects the components corresponding to minimal primes. Throughout the rest of the paper, we will assume that the `UseMinimalPrimes` option is set to `false`, which is the default setting.

Various algorithms are used for the computation of symbolic powers. This package follows the order given below to decide the optimal algorithm applicable for computing `symbolicPower(I,n)`:

(1) If I is a squarefree monomial ideal, the routine intersects the n-th powers of the associated primes of I.

(2) If I is a monomial ideal, but not squarefree, the routine takes a primary decomposition of I and intersects the n-th powers of the intersections of the primary components associated to primes contained in each maximal element of $\mathrm{Ass}(I)$ (see [Cooper et al. 2017, Lemma 3.1]).

(3) If I is a saturated homogeneous ideal whose height is one less than the dimension of its ambient ring, the routine returns the saturation of I^n with respect to the maximal ideal.

(4) If I is height unmixed (meaning that all the associated primes of I have the same height) the routine computes the top dimensional components of I^n using an algorithm of Eisenbud, Huneke and Vasconcelos [Eisenbud et al. 1992] (see Section 3).

(5) If all else fails, the routine compares the radicals of a primary decomposition of I^n with the associated primes of I, and intersects the components corresponding to minimal primes.

Whenever primary decomposition is computed, the package uses the existing Macaulay2 routine for computing primary decompositions, which by default employs the Shimoyama–Yokoyama algorithm [1996] except when the given ideal is monomial. However, note that finding primary decompositions is generally a fairly slow process, and certainly slower than the first four strategies listed above. Explicit experiments demonstrating that the first, third and fourth strategies outperform the last, even when factoring in the time needed to check their applicability, are given in Examples 2.1, 2.2 and 2.4. For this reason, we avoid computing the primary decomposition of I^n whenever possible.

There is one notable exception to this philosophy: in the case when the primary components of an ideal are complete intersections, the extra time spent computing a primary decomposition can be worth it (cf. Example 2.5). If the option `CIPrimes` is set to `true`, then `symbolicPower(I,n)` outputs the intersection of the n-th powers of the primary components of the input ideal I, if each of these components is a complete intersection and they all have the same height. Using the `CIPrimes` option computes the symbolic power much more quickly than the other five strategies in cases when there are sufficiently many associated primes.

We compare below the running times of the various algorithms that we use for computing symbolic powers in several examples. In the following, we denote the first algorithm listed above by `mon'l`, the third by `sat`, the fourth by `unmixed`, and the last by `pdec`.

Example 2.1. Set $R = k[x, y, z]$, where k is a field of characteristic not equal to 2, and

$$I = (x(y^3 - z^3), y(z^3 - x^3), z(x^3 - y^3))$$

is an ideal which has become known in the literature as a Fermat ideal. The table below compares the running times in seconds for the algorithms `pdec` and `sat` as well as the total running time for `symbolicPower(I,5)`. Note that in this example the `symbolicPower` method checks the hypotheses needed for applying the saturation algorithms and then runs this routine:

	`pdec`	`sat`	`symbolicPower`
running times for $I^{(5)}$	4	0.036	0.040

Example 2.2. Set $R = k[x_1, x_2, x_3, x_4, x_5]$ and let I be the ideal generated by all the squarefree monomials of degree 2 in R. The running times in seconds for the algorithms `pdec` and `mon'l` are compared to the running time for `symbolicPower(I,5)`

in the following table:

	pdec	mon'l	symbolicPower
running times for $I^{(5)}$	1.35	0.004	0.004

Example 2.3. Set $R = k[x, y, z]$ and let $I = (xy, xz, yz)$. In this example we compare the `mon'l` and `sat` strategies, since both are applicable. The running times in seconds for the algorithms `pdec`, `sat` and `mon'l` are compared to the running time for `symbolicPower`, which also checks the applicability of the `mon'l` strategy.

	mon'l	sat	pdec	symbolicPower
running times for $I^{(5)}$	0.001	0.006	0.021	0.002
running times for $I^{(10)}$	0.001	0.369	0.558	0.002

Example 2.4. Set $R = k[x_1, \ldots, x_{12}]$ and let I be the ideal generated by the 2×2 minors of a generic 3×4 matrix with entries the variables of R. The running times in seconds for the algorithms `unmixed` and `pdec` are compared to the running time for `symbolicPower(I,5)` in the following table:

	unmixed	pdec	symbolicPower
running times for $I^{(5)}$	3.970	44.538	4.231

This example shows that even including the overhead of checking that the ideal above is height unmixed, the routine `symbolicPower`, which in this case uses the `unmixed` strategy based on the method of Eisenbud, Huneke and Vasconcelos, outperforms the `pdec` algorithms.

Example 2.5. Let I be the ideal of ten general points in $\mathbb{P}^2$. We illustrate the computation times for the fifth symbolic powers of I with the option `CIPrimes` turned on in comparison to the default strategy for this case, which is to use the saturation algorithm.

	CIPrimes	sat	symbolicPower
running times for $I^{(5)}$	0.447	3.483	3.495

3. APPLICATIONS.

Methods based on a result of Eisenbud, Huneke, and Vasconcelos. We can identify the heights of all the associated primes of an ideal in a regular ring using the following result:

Theorem 3.1 [Eisenbud et al. 1992]. *Given an ideal I in a regular domain R of height h, then for each $e \geqslant h$, I has an associated prime of height e if and only if the height of* $\operatorname{Ext}^e(R/I, R)$ *is e. In particular, the intersection of the top dimensional components of I is given by* $\operatorname{Ann} \operatorname{Ext}^h_R(R/I, R)$.

The already existing method `topComponents`, also based on this result, returns the intersection of the primary components of minimal height of an ideal. In particular, if I has pure height h, then `topComponents`(I^n) returns $I^{(n)}$. This is one of the strategies used by the method `symbolicPower`.

Further, the `SymbolicPowers.m2` package also includes the method `bigHeight`, which computes the largest height of an associated prime of I, and the method `assPrimesHeight`, which returns a list of all the heights of the associated primes of I. Both of these are based on Theorem 3.1.

The method `minimalPart` returns the intersection of the minimal components of a given ideal, which is in general different from `topComponents`. Instead of explicitly finding the associated primes of I and taking their heights, Theorem 3.1 is used.

Equality. Symbolic powers do not, in general, coincide with the ordinary powers, even in the case of prime ideals. In fact, the question of characterizing the ideals I for which $I^{(n)} = I^n$ for all n is essentially open. One can determine whether the n-th symbolic and ordinary powers of a given ideal coincide using `isSymbolicEqualOrdinary`, often without computing the actual symbolic power of I. For this, the package makes use of `bigHeight`. To determine whether $I^{(n)} = I^n$ for a specific value of n, `isSymbolicEqualOrdinary` first compares the big heights of I^n and I: if the big heights differ, then I^n must have embedded components, and `isSymbolicEqualOrdinary` returns `false`; if the big heights are both equal to the height of I, then I^n cannot have embedded components, and `isSymbolicEqualOrdinary` returns `true`. This is faster than computing the set of associated primes of I^n. Using `symbolicDefect`, one can quantify the difference between I^m and $I^{(m)}$ by computing the symbolic defect of I in the power m, defined by Galetto, Geramita, Shin, and Van Tuyl in [Galetto et al. 2019] to be the minimal number of generators of $I^{(m)}/I^m$.

The packing problem. Besides allowing the user to determine when $I^{(n)} = I^n$ holds without the need to explicitly compute $I^{(n)}$, the `SymbolicPowers.m2` package also includes other methods that can be applied to this question. In particular, the package includes methods related to the packing problem, which was originally formulated in the context of max-flow min-cut properties by Conforti and Cornuéjols [1990]. Work of Gitler, Villarreal and others shows that this problem can be rewritten as a conjectural characterization of the squarefree monomial ideals having $I^{(n)} = I^n$ for all n as those ideals that satisfy the packing property. The

method `isPacked` determines if a given squarefree monomial ideal has this property. In particular, should the packing problem have an affirmative answer, this method could be used as a test for whether the equality $I^{(n)} = I^n$ holds for all n. For a quick survey on the packing problem, see [Dao et al. 2018].

The containment problem. The containment problem for ordinary and symbolic powers of ideals consists of answering the following question: given an ideal I, for which values of a and b does the containment $I^{(a)} \subseteq I^b$ hold? Over a regular ring, a well known theorem of Ein, Lazersfeld and Smith [2001], Hochster and Huneke [2002], and Ma and Schwede [2018] gives a partial answer to that question: when I is a radical ideal, $I^{(hn)} \subseteq I^n$ holds for all n, where h denotes the big height of the ideal I. However, this is not necessarily best possible; see [Szemberg and Szpond 2017] for a survey. Using `containmentProblem`, the user can determine the smallest value of a, given b, for which $I^{(a)} \subseteq I^b$. Conversely, using the option `InSymbolic`, the user can determine the largest value of b, given a, for which $I^{(a)} \subseteq I^b$.

Example 3.2 (containment problem).

```
i1 : loadPackage "SymbolicPowers";
i2 : R=QQ[x,y,z];
i3 : I=ideal(x*(y^3-z^3),y*(z^3-x^3),z*(x^3-y^3));
o3 : ideal of R
i4 : containmentProblem(I,2)
o4 : 4
i6 : containmentProblem(I,5, InSymbolic=>true)
o6 : 3
```

The computation `containmentProblem(I,2)=4` illustrated above should be interpreted as stating that $I^{(4)} \subseteq I^2$ and $I^{(3)} \nsubseteq I^2$, while we can interpret the computation `containmentProblem(I,5,InSymbolic=>true)=3` as stating that $I^{(5)} \subseteq I^3$ and $I^{(5)} \nsubseteq I^4$.

Other applications. Some of the other methods in the package include specialized functionality for computations in positive characteristic and for computations specific to ideals defining monomial curves.

The method `symbolicPowerPrimePosChar` gives another algorithm for computing symbolic powers which is specific to working in prime characteristic p. This method can be faster than the other algorithms for computing symbolic powers $I^{(n)}$ for values of n very close to being a power of p, but not for general values of n.

For the special case of monomial curves $k[t^{a_1}, \ldots, t^{a_k}]$, both of the methods `symbolicPowerMonomialCurve` and `containmentProblemMonomialCurve` essentially run `symbolicPower` and `containmentProblem`.

4. ASYMPTOTIC INVARIANTS. In an effort to make progress on the containment problem, various asymptotic interpolation invariants have been proposed by Bocci and Harbourne [2010]. One such invariant is the Waldschmidt constant for a homogeneous ideal I. This is an asymptotic measure of the initial degree of the symbolic powers of I. The *initial degree* of a homogeneous ideal I is $\alpha(I) = \min\{d \mid I_d \neq 0\}$, i.e., the smallest degree of a nonzero element in I. The *Waldschmidt constant* of I is defined to be

$$\widehat{\alpha}(I) = \lim_{m\to\infty} \frac{\alpha(I^{(m)})}{m}.$$

Due to the asymptotic nature of the Waldschmidt constant, there is no a priori algorithm to determine this invariant for arbitrary ideals, although the initial degrees of individual symbolic powers can be computed using `minDegreeSymbPower`. An important exception is the case when the ideal I is a monomial ideal. In this context, the Waldschmidt constant can be computed as the smallest among the sums of the coordinates of all points in a convex body termed the *symbolic polyhedron* of I [Cooper et al. 2017; Bocci et al. 2016]. Our package computes Waldschmidt constants of monomial ideals by finding their symbolic polyhedron. The `symbolicPolyhedron` routine makes heavy use of the `Polyhedra.m2` package by René Birkner, which in turn relies on the `FourierMotzkin.m2` package by Greg Smith. This allows to determine the Waldschmidt constants of monomial ideals exactly as in the following example.

Example 4.1 (Waldschmidt constant of monomial ideals).

```
i1 : loadPackage "SymbolicPowers";
i2 : R=QQ[x,y,z];
i3 : I=ideal(x*y,x*z,y*z);
i4 : symbolicPolyhedron(I)
o4 = {ambient dimension => 3           }
      dimension of lineality space => 0
      dimension of polyhedron => 3
      number of facets => 6
      number of rays => 3
      number of vertices => 4
o4 : Polyhedron
i5 : waldschmidt I
Ideal is monomial, the Waldschmidt constant is computed exactly
      3
o5 =  -
      2
o5 : QQ
```

In the case of arbitrary ideals, the Waldschmidt constant is approximated by

taking the minimum of the values $\alpha(I^{(m)})/m$, where m ranges from 1 to a specified optional input `SampleSize`.

Example 4.2 (Waldschmidt constant of arbitrary ideals).

```
i1 : loadPackage "SymbolicPowers";
i2 : R=QQ[x,y,z];
i3 : I=ideal(x*(y^3-z^3),y*(z^3-x^3),z*(x^3-y^3));
o3 : Ideal of R
i4 : waldschmidt I
Ideal is not monomial, the  Waldschmidt constant is approximated
using first 5 powers.
o4 = 3
o4 : QQ
```

Note that the true value for the Waldschmidt constant of the above ideal is indeed 3 as proven in [Dumnicki et al. 2015]. In general, for an ideal that is not monomial, the function `waldschmidt` will return an upper bound on the true value of the Waldschmidt constant.

Another asymptotic invariant termed *resurgence* [Bocci and Harbourne 2010] is defined as

$$\rho(I) = \sup\left\{\frac{m}{r} \;\middle|\; I^{(m)} \not\subseteq I^r\right\}.$$

There are no algorithms known to date that compute resurgence exactly; therefore, our package computes a lower bound for the resurgence by taking the maximum of the values $\frac{m}{r}$, where r ranges from 1 to the optional input `SampleSize`.

Continuing with the ideal in the previous example, we compute a lower bound on its resurgence using the default `SampleSize`, which is 5, and also a custom `SampleSize`. As expected, the lower bound increases as the `SampleSize` is increased, i.e., a larger `SampleSize` produces a better lower bound.

Example 4.3 (lower bound on resurgence).

```
i1 : loadPackage "SymbolicPowers";
i2 : R=QQ[x,y,z];
i3 : I=ideal(x*y,x*z,y*z);
i5 : lowerBoundResurgence(I)
     6
o5 = -
     5
o5 : QQ
i6 : lowerBoundResurgence(I,SampleSize=>10)
     5
o6 = -
     4
o6 : QQ
```

ACKNOWLEDGMENTS. We would like to thank the organizers of the July 2017 Macaulay2 Workshop at the University of California, Berkeley, where a large portion of this work was done. The code for computing the symbolic polyhedron and Waldschmidt constant of a monomial ideal was developed by Seceleanu in collaboration with Andrew Conner and Xuehua (Diana) Zhong. We thank them for their contribution to these routines.

We also thank the anonymous referee for many helpful suggestions and comments.

This research was partially supported by NSF grant DMS #1502282; we thank Luis Núñez Betancourt and Craig Huneke for that support, and for organizing a conference at the University of Virginia where part of this work took place. Seceleanu was supported by NSF grant DMS#1601024. Drabkin and Seceleanu were supported by EPSCoR grant OIA–1557417.

SUPPLEMENT. Version 2.0 of `SymbolicPowers.m2`. is contained in the online supplement.

REFERENCES.

[Bocci and Harbourne 2010] C. Bocci and B. Harbourne, "Comparing powers and symbolic powers of ideals", *J. Algebraic Geom.* **19**:3 (2010), 399–417. MR Zbl

[Bocci et al. 2016] C. Bocci, S. Cooper, E. Guardo, B. Harbourne, M. Janssen, U. Nagel, A. Seceleanu, A. Van Tuyl, and T. Vu, "The Waldschmidt constant for squarefree monomial ideals", *J. Algebraic Combin.* **44**:4 (2016), 875–904. MR Zbl

[Conforti and Cornuéjols 1990] M. Conforti and G. Cornuéjols, "A decomposition problem for balanced matrices", working paper 1990-10, Carnegie Mellon Univ., Tepper School of Business, 1990.

[Cooper et al. 2017] S. M. Cooper, R. J. D. Embree, H. T. Hà, and A. H. Hoefel, "Symbolic powers of monomial ideals", *Proc. Edinb. Math. Soc.* (2) **60**:1 (2017), 39–55. MR Zbl

[Dao et al. 2018] H. Dao, A. De Stefani, E. Grifo, C. Huneke, and L. Núñez Betancourt, "Symbolic powers of ideals", pp. 387–432 in *Singularities and foliations: geometry, topology and applications* (Salvador, Brazil, 2015), edited by R. N. Araújo dos Santos et al., Springer Proc. Math. Stat. **222**, Springer, 2018. MR Zbl

[Dumnicki et al. 2015] M. Dumnicki, B. Harbourne, U. Nagel, A. Seceleanu, T. Szemberg, and H. Tutaj-Gasińska, "Resurgences for ideals of special point configurations in $\mathbb{P}^N$ coming from hyperplane arrangements", *J. Algebra* **443** (2015), 383–394. MR Zbl

[Ein et al. 2001] L. Ein, R. Lazarsfeld, and K. E. Smith, "Uniform bounds and symbolic powers on smooth varieties", *Invent. Math.* **144**:2 (2001), 241–252. MR Zbl

[Eisenbud et al. 1992] D. Eisenbud, C. Huneke, and W. Vasconcelos, "Direct methods for primary decomposition", *Invent. Math.* **110**:1 (1992), 207–235. MR Zbl

[Galetto et al. 2019] F. Galetto, A. V. Geramita, Y.-S. Shin, and A. Van Tuyl, "The symbolic defect of an ideal", *J. Pure Appl. Algebra* **223**:6 (2019), 2709–2731. MR Zbl

[Hochster and Huneke 2002] M. Hochster and C. Huneke, "Comparison of symbolic and ordinary powers of ideals", *Invent. Math.* **147**:2 (2002), 349–369. MR Zbl

[Ma and Schwede 2018] L. Ma and K. Schwede, "Perfectoid multiplier/test ideals in regular rings and bounds on symbolic powers", *Invent. Math.* **214**:2 (2018), 913–955. MR Zbl

[Nagata 1962] M. Nagata, *Local rings*, Interscience Tracts in Pure Appl. Math **13**, Interscience, New York, 1962. MR Zbl

[Shimoyama and Yokoyama 1996] T. Shimoyama and K. Yokoyama, "Localization and primary decomposition of polynomial ideals", *J. Symbolic Comput.* **22**:3 (1996), 247–277. MR Zbl

[Szemberg and Szpond 2017] T. Szemberg and J. Szpond, "On the containment problem", *Rend. Circ. Mat. Palermo* (2) **66**:2 (2017), 233–245. MR Zbl

[Zariski 1949] O. Zariski, "A fundamental lemma from the theory of holomorphic functions on an algebraic variety", *Ann. Mat. Pura Appl.* (4) **29** (1949), 187–198. MR Zbl

RECEIVED: 5 Dec 2017 REVISED: 2 Feb 2019 ACCEPTED: 20 May 2019

BEN DRABKIN:

benjamin.drabkin@huskers.unl.edu
Department of Mathematics, University of Nebraska, Lincoln, NE, United States

ELOÍSA GRIFO:

grifo@umich.edu
Department of Mathematics, University of Michigan, Ann Arbor, MI, United States

ALEXANDRA SECELEANU:

aseceleanu@unl.edu
Department of Mathematics, University of Nebraska, Lincoln, NE, United States

BRANDEN STONE:

bstone@hamilton.edu
Mathematics Department, Hamilton College, Clinton, NY, United States

JSAG **9** (2019), 81–87
dx.doi.org/10.2140/jsag.2019.9.81

The Journal of Software for
Algebra and Geometry

The gfanlib interface in Singular and its applications

ANDERS JENSEN, YUE REN AND HANS SCHÖNEMANN

ABSTRACT: We briefly report on SINGULAR's low-level interface to GFANLIB: its usage on interpreter level, its implementation using SINGULAR's `blackbox` framework and two of its applications.

1. INTRODUCTION. SINGULAR [Decker et al. 2016] is a comprehensive computer algebra system for polynomial computations with particular emphasis on applications in algebraic geometry, commutative algebra and singularity theory. GFANLIB is a library derived from GFAN [Jensen 2011], a software package for computing tropical varieties and Gröbner fans by the first author. The GFANLIB interface allows SINGULAR users to work with polyhedral cones, polyhedral fans and operations thereon.

The goal of this paper is threefold: In Section 2, we provide an overview of the functionality in GFANLIB and how it can be accessed in SINGULAR. In Section 3, we describe the `blackbox`-framework in SINGULAR and illustrate how it can be used to introduce new types and functions to SINGULAR using the GFANLIB interface as an example. In Section 4, we showcase two applications of GFANLIB, demonstrating its use in high performance computations.

2. GFANLIB AND ITS INTERFACE IN SINGULAR. The class `ZCone` of GFANLIB encapsulates a rational polyhedral cone built on CDDLIB [Fukuda 2016] with arbitrary precision integral normal vectors. A `ZCone` object can be in one of four states (with an increasing level of knowledge), and it automatically advances its state if necessary:

(0) Defining equations and inequalities are known.

(1) A basis for the orthogonal complement is known. Its cardinality equals the codimension of the cone.

(2) A minimal set of inequalities has been determined. Each inequality cuts out a facet of the cone.

(3) The equations and inequalities from (1) and (2) have been written in a unique canonical form. This allows fast cone comparisons.

The class is accessible from SINGULAR as the type `cone`; see Figure 1. Cone A is defined as the non-negative span of the row vectors of the matrix M, $A = (\mathbb{R}_{\geq 0})^4 \cdot M$, while cone B is defined by considering the rows as inequalities, $B = (\mathbb{R}_{\geq 0})^2 \cdot M^t$. In both cases the cones are stored with an outer (H-) description, which means the first case requires a double description conversion. As a side effect, the inequalities that are obtained through the conversion are indeed facet normals and they are therefore marked as such.

MSC2010: primary 68W30; secondary 14L24, 14T05.
Keywords: Singular, gfanlib, interface.

```
> LIB "gfan.lib";
> intmat M[4][2]=(1,0),
  (1,1),(1,2),(1,3);
> cone A=coneViaPoints(M);
> A;
AMBIENT_DIM
2
FACETS
0, 1,
3,-1
> cone B=coneViaInequalities(M);
> B;
AMBIENT_DIM
2
INEQUALITIES
1,0,
1,1,
1,2,
1,3
```

Figure 1. A sample SINGULAR session using the type `cone`.

A `ZFan` class for representing polyhedral fans is provided. Cones can be inserted to such fan and when that happens, all faces are implicitly regarded as being inserted too. The `ZFan` class allows for easy conversion to ray incidence lists as in Figure 5.

Finally, the interface offers the possibility of computing mixed volume of Newton polytopes using [Jensen 2016], see Figure 2.

```
> LIB "gfan.lib";
> ring r=0,(x,y),dp;
> mixedVolume(list(x2y+1,xy2+1));
3
```

Figure 2. An example of computing mixed volume in SINGULAR.

3. THE BLACKBOX FRAMEWORK OF SINGULAR. Adding new data types and associated functions to SINGULAR used to require changes in many places due to the static nature of its type system. This process is now streamlined with the introduction of the type `blackbox`. An object of type `blackbox` is a `struct` of function pointers as explained in the following subsection. This is largely equivalent to defining new types as subclasses of an abstract class having the methods listed below being virtual, thereby taking advantage of inheritance in C++. With our solution on the other hand, we have full control and can add types at runtime as we like.

***Adding new data types to* SINGULAR.** Data types governed by `blackbox` are represented by a pair: a `void*` pointer to a call table, and an `int` serving as a unique type identifier. Registering a new type begins with filling a `struct` with function pointers to the following functions, of which the latter five are optional and come with preset defaults (see also Figure 3, lines 23–27):

(1) `blackbox_Init`: creates a default object
(2) `blackbox_destroy`: destroys an object (see Figure 3, lines 3–6)
(3) `blackbox_Copy`: copies an object (see Figure 3, lines 8–15)
(4) `blackbox_String`: converts an object to string
(5) `blackbox_Print`: prints an object (default: print conversion to string)
(6) `blackbox_Assign`: assign other types to object (default: raises error)
(7) `blackbox_OpX`: operations with object as first operand (default: raises error)
(8) `blackbox_serialize`: serialization for parallel computing (default: raises error)

(9) `blackbox_deserialize`: deserialization (default: raises error)

Next, that `struct` plus the name of the new type has to be passed to `setBlackboxStuff`, which then returns the unique type identifier (see Figure 3, line 31).

```
1  int coneID; // type identifier
2
3  void* bbcone_Init(blackbox* b)
4  {
5    return new gfan::ZCone();
6  }
7
8  char* bbcone_String(blackbox* b,
9                      void* d)
10 {
11   gfan::ZCone* zc =
12     (gfan::ZCone*) d;
13   string s = zc->toString();
14   return omStrDup(s.c_str());
15 }
```

```
18 void mod_init(SModulFunctions* p)
19 {
20   blackbox *b =
21   (blackbox*) omAlloc0(sizeof(*b));
22
23   b->blackbox_Init=bbcone_Init;
24   b->blackbox_Copy=bbcone_Copy;
25   b->blackbox_Assign=bbcone_Assign;
26   b->blackbox_destroy=bbcone_Destroy;
27   b->blackbox_String=bbcone_String;
28
29   [...]
30
31   coneID=setBlackboxStuff(b,"cone");
32 }
```

Figure 3. `blackbox` wrapper for type `gfan::ZCone`.

***Adding new functions to* SINGULAR.** Functions governed by `blackbox` require two `leftv` as input and a `BOOLEAN` as ouput (see Figure 4). The first `leftv` represents the return value of the function when called from the interpreter, the second `leftv` represents the input. The `BOOLEAN` that is returned signals the interpreter whether an error occured during the function call. Objects of type `leftv` are generic objects which can represent any object of any type that is available on the interpreter-level of SINGULAR. The unique type identifier of the input can be seen by calling `leftv->Typ()`, while the identifier of the output is to be stored in `leftv->rtyp`. The data of the input can be obtained by calling `leftv->Data()`, and the data of the output is to be stored in `leftv->data`. Moreover, `leftv->next` can be used to access the latter arguments of the input. It is set to `NULL` if the function has been called with one argument. To make the function callable from the interpreter, `iiAddCproc` needs to be called with:

(1) a `string` with the library containing documentation (empty string possible)

(2) a `string` with the desired function name on the interpreter level

(3) a `BOOLEAN` which marks the routine as static (`TRUE`) or global (`FALSE`)

(4) the function name on the kernel level

4. APPLICATIONS OF THE GFANLIB INTERFACE.

In this section, we briefly present two applications of SINGULAR's low-level interface to GFANLIB based on the previously described `blackbox` framework.

```
BOOLEAN facets(leftv res, leftv args)
{
  if ((args != NULL) && (args->Typ() == coneID) && (args->next == NULL))
  {
    gfan::ZCone* zc = (gfan::ZCone*) args->Data();
    gfan::ZMatrix zm = zc->getFacets();
    res->rtyp = BIGINTMAT_CMD;
    res->data = (void*) zMatrixToBigintmat(zm); // conversion function
    return FALSE;
  }
  WerrorS("facets: unexpected parameters");
  return TRUE;
}
void bbcone_setup(SModulFunctions* p)
{
  [...]
  p->iiAddCproc("gfan.lib","facets",FALSE,facets);
  [...]
}
```

Figure 4. `blackbox` wrapper for function `gfan::ZCone::getFacets()`.

Tropical varieties. Tropical geometry studies algebraic varieties based on what is commonly described as their combinatorial shadow. These so-called tropical varieties are the supports of finite polyhedral complexes.

Definition 1. Let K be a field with non-trivial valuation $\nu : K^* \to \mathbb{R}$ and residue field $\mathfrak{K}$. Fix a splitting $\nu(K^*) \to K^*$, writing t^a when referring to the image of $a \in \Gamma_\nu$. Given a polynomial $f = \sum_{\alpha \in \mathbb{N}^n} c_\alpha \cdot x^\alpha \in K[x]$ and an ideal $I \trianglelefteq K[x]$, we define for any weight vector $w \in \mathbb{R}^n$:

$$\mathrm{in}_w(f) = \sum_{w\cdot\alpha - \nu(c_\alpha)\ \text{max.}} \overline{t^{-\nu(c_\alpha)} c_\alpha} \cdot x^\alpha \in \mathfrak{K}[x] \text{ and } \mathrm{in}_w(I) = \langle \mathrm{in}_w(f) \mid f \in I \rangle \trianglelefteq \mathfrak{K}[x].$$

The tropical variety of I is then given by:

$$\mathcal{T}(I) = \overline{\{w \in \mathbb{R}^n \mid \mathrm{in}_w(I) \text{ monomial free}\}}.$$

Computing tropical varieties is an algorithmically challenging task, requiring sophisticated techniques from both computer algebra and convex geometry. The first techniques were developed for the field of complex Puiseux series $\mathbb{C}\{\{t\}\}$ [Bogart et al. 2007], in which the authors exploited that its uniformizing parameter t can be regarded as a variable of the polynomial ring, allowing them to rely on classical Gröbner bases techniques. For more general fields with valuation, a new theory of Gröbner bases taking the valuation on the field into account was introduced [Chan and Maclagan 2019].

In contrast, the implementation in SINGULAR is based on recent results [Markwig and Ren 2019], which describe tropical varieties over valued fields K with tropical varieties over any dense subring of the ring of integers $\mathcal{O}_K \subseteq K$. The practical advantage of this approach is that it is compatible with existing standard bases techniques in SINGULAR.

Example 2. Consider the ideal $I := \langle x_1 - 2x_2 + 3x_3, 3x_2 - 4x_3 + 5x_4 \rangle \trianglelefteq \mathbb{Q}[x_1, \ldots, x_4]$ and the 2-adic valuation on $\mathbb{Q}$. Figure 5 shows input and output for computing the tropical variety in SINGULAR.

```
> LIB "tropical.lib";
> ring r = 0,x(1..4),dp; number p = 2;
> ideal I = x(1)+2*x(2)-3*x(3), 3*x(2)-4*x(3)+5*x(4);
> tropicalVariety(I,p);

RAYS                                MAXIMAL_CONES
-2 -1 1 -1 1\# 0                     {0 1}    \# Dimension 3
-1 1 -1 1 -1\# 1                     {0 2}
0 -3 1 1 1\# 2                       {0 4}
0 1 -3 1 1\# 3                       {1 3}
0 1 1 -3 1\# 4                       {1 5}
0 1 1 1 -3\# 5
```

Figure 5. SINGULAR output for Example 2.

The output is a polyhedral fan in $\mathbb{R}^5$ whose intersection with the affine hyperplane $\{e_1 = 1\}$ yields a polyhedral complex covering the tropical variety. Hence the rays #2–#5 represent vertices at infinity so that the last four maximal 2-dimensional cones are in fact unbounded edges; see Figure 6.

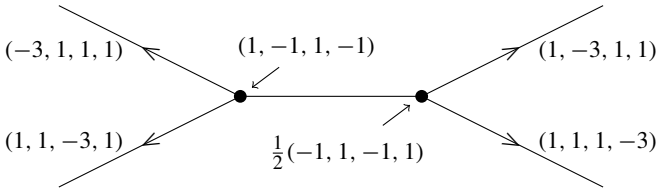

Figure 6. $\mathcal{T}(I)$ for Example 2.

GIT-fans. GIT-fans describe the variation of possible GIT-quotients [Dolgachev and Hu 1998]. A case of particular importance is the action of an algebraic torus on an affine variety, for which explicit algorithms exist [Berchtold and Hausen 2006; Keicher 2012; Boehm et al. 2016]. All algorithms have been implemented in the SINGULAR library `gitfan.lib` and are publicly available as part of the official SINGULAR distribution.

Example 3. [Boehm et al. 2016, Example 5.2]
The Cox ring of Grass(2, 5) is isomorphic to $\mathbb{C}[T_1, \ldots, T_{10}]/\mathfrak{a}$, where the ideal $\mathfrak{a}$ is generated by the following Plücker relations and the $(\mathbb{C}^*)^5$ action on Grass(2, 5) is induced by the following grading matrix Q with respect to which $\mathfrak{a}$ is homogeneous:

$$\begin{aligned} &T_5T_{10} - T_6T_9 + T_7T_8,\\ &T_1T_9 - T_2T_7 + T_4T_5,\\ &T_1T_8 - T_2T_6 + T_3T_5,\\ &T_1T_{10} - T_3T_7 + T_4T_6,\\ &T_2T_{10} - T_3T_9 + T_4T_8 \end{aligned} \qquad Q := \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The natural S_5-symmetry on Grass(2, 5) acts, up to sign, as the following permutations on the variables:

$$S_5 = \langle (2,3)(5,6)(9,10), (1,5,9,10,3)(2,7,8,4,6) \rangle \leq S_{10}.$$

The GIT-fan consists of 76 cones, which decompose into 6 orbits under the S_5 action. Figure 7 shows the computation of the GIT-fan without and with symmetry. The GIT-fan is generated by 76 maximal cones, which decompose into 6 orbits under the group action.

```
> LIB "gitfan.lib";
> ring R = 0,T(1..10),dp;
> ideal J =
.   T(5)*T(10)-T(6)*T(9)+T(7)*T(8),
.   T(1)*T(9)-T(2)*T(7)+T(4)*T(5),
.   T(1)*T(8)-T(2)*T(6)+T(3)*T(5),
.   T(1)*T(10)-T(3)*T(7)+T(4)*T(6),
.   T(2)*T(10)-T(3)*T(9)+T(4)*T(8);
> intmat Q[5][10] =
.   1, 1, 1, 1, 0, 0, 0, 0, 0, 0,
.   1, 0, 0, 0, 1, 1, 1, 0, 0, 0,
.   0, 1, 1, 0, 0, 0, -1, 1, 0, 0,
.   0, 1, 0, 1, 0, -1, 0, 0, 1, 0,
.   0, 0, 1, 1, -1, 0, 0, 0, 0, 1;
> list S5 = G25Action();
```

```
> S5;
[1]:
   |  1  2  3  4  5  6  7  8  9 10|
[...]
[120]:
   |  1  2  3  4  5  6  7  8  9 10|
   | 10  9  7  4  8  6  3  5  2  1|
> fan SigmaIgnoringSymmetry =
.   GITfan(J,Q);
> fVector(SigmaIgnoringSymmetry);
1,20,110,240,225,76
> fan SigmaModuloSymmetry =
.   GITfan(J,Q,S5);
> fVector(SigmaModuloSymmetry);
1,16,56,67,33,6
```

Figure 7. SINGULAR input and output for Example 3.

Figure 8 shows the adjacency graph of the maximal cones and their S_5 orbits.

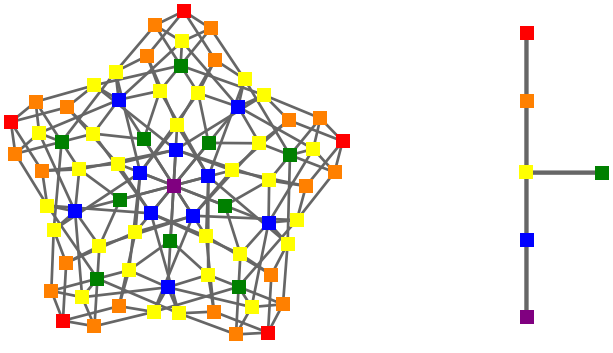

Figure 8. Adjacency graph of the maximal cones in the GIT-fan of Grass(2, 5) and of their S_5-orbits.

SUPPLEMENT. The online supplement contains source code for the GFANLIB interface to SINGULAR.

REFERENCES.

[Berchtold and Hausen 2006] F. Berchtold and J. Hausen, "GIT equivalence beyond the ample cone", *Michigan Math. J.* **54**:3 (2006), 483–515. MR

[Boehm et al. 2016] J. Boehm, S. Keicher, and Y. Ren, "Computing GIT-fans with symmetry and the Mori chamber decomposition of $\bar{M}_{0,6}$", 2016. arXiv

[Bogart et al. 2007] T. Bogart, A. N. Jensen, D. Speyer, B. Sturmfels, and R. R. Thomas, "Computing tropical varieties", *J. Symbolic Comput.* **42**:1-2 (2007), 54–73. MR

[Chan and Maclagan 2019] A. J. Chan and D. Maclagan, "Gröbner bases over fields with valuations", *Math. Comp.* **88**:315 (2019), 467–483. MR

[Decker et al. 2016] W. Decker, G.-M. Greuel, G. Pfister, and H. Schönemann, "SINGULAR 4-1-0: a computer algebra system for polynomial computations", 2016, available at http://www.singular.uni-kl.de.

[Dolgachev and Hu 1998] I. V. Dolgachev and Y. Hu, "Variation of geometric invariant theory quotients", *Inst. Hautes Études Sci. Publ. Math.* 87 (1998), 5–56. With an appendix by Nicolas Ressayre. MR

[Fukuda 2016] K. Fukuda, "cddlib reference manual, cddlib Version 094h", 2016, available at https://www.inf.ethz.ch/personal/fukudak/cdd_home.

[Jensen 2011] A. N. Jensen, "Gfan 0.5, a software system for Gröbner fans and tropical varieties", 2011, available at http://home.math.au.dk/jensen/software/gfan/gfan.html.

[Jensen 2016] A. N. Jensen, "An implementation of exact mixed volume computation", pp. 198–205 in *Mathematical software—ICMS 2016*, Lecture Notes in Comput. Sci. **9725**, Springer, [Cham], 2016. MR

[Keicher 2012] S. Keicher, "Computing the GIT-fan", *Internat. J. Algebra Comput.* **22**:7 (2012), 1250064, 11. MR

[Markwig and Ren 2019] T. Markwig and Y. Ren, "Computing tropical varieties over fields with valuation", *Found. Comput. Math.* (online publication August 2019).

RECEIVED: 11 Jul 2018 ACCEPTED: 19 Nov 2018

ANDERS JENSEN:

jensen@imf.au.dk
Aarhus Universitet, Aarhus, Denmark

YUE REN:

yue.ren@mis.mpg.de
Max Planck Institute for Mathematics in the Sciences, Leipzig, Germany

HANS SCHÖNEMANN:

hannes@mathematik.uni-kl.de
Technische Universität Kaiserslautern, Kaiserslautern, Germany

JOURNAL OF SOFTWARE FOR ALGEBRA AND GEOMETRY vol 9, no 1, 2019

1948-7916(2019)9:1;1-D