IRTG seminar series on
"Tools that ~~saved~~ took my life"

A script kiddie's guide to...

SINGULAR

...teaching you enough to break it,
but not enough to fix it.

Johannes Schmitt
TU Kaiserslautern
5th August 2021

What is SINGULAR?

Computer algebra system for polynomial computations

What is SINGULAR?

Computer algebra system for polynomial computations

Open source (GPL)

What is SINGULAR?

Computer algebra system for polynomial computations

Open source (GPL)

Developed at TU Kaiserslautern

What is SINGULAR?

Computer algebra system for polynomial computations

Open source (GPL)

Developed at TU Kaiserslautern

C-like user language

What is SINGULAR?

Computer algebra system for polynomial computations

Open source (GPL)

Developed at TU Kaiserslautern

C-like user language

Available at `singular.uni-kl.de`

What is SINGULAR?

Computer algebra system for polynomial computations

Open source (GPL)

Developed at TU Kaiserslautern

C-like user language

Available at `singular.uni-kl.de`

There's a book about it: G.-M. Greuel, G. Pfister: "A Singular Introduction to Commutative Algebra".

What is Singular?

Computer algebra system for polynomial computations

Open source (GPL)

Developed at TU Kaiserslautern

C-like user language

Available at `singular.uni-kl.de`

There's a book about it: G.-M. Greuel, G. Pfister: "A Singular Introduction to Commutative Algebra".

Julia-Package `Singular.jl` as part of Oscar

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

- $\mathbb{Q}$,

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

- $\mathbb{Q}$,
- $\mathbb{F}_q$ (there are weird restrictions on the size of q, though),

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

- $\mathbb{Q}$,
- $\mathbb{F}_q$ (there are weird restrictions on the size of q, though),
- transcendental or algebraic extensions of $\mathbb{Q}$ and $\mathbb{F}_p$,

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

- $\mathbb{Q}$,
- $\mathbb{F}_q$ (there are weird restrictions on the size of q, though),
- transcendental or algebraic extensions of $\mathbb{Q}$ and $\mathbb{F}_p$,
- $\mathbb{R}$ and $\mathbb{C}$ (user defined precision),

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

- $\mathbb{Q}$,
- $\mathbb{F}_q$ (there are weird restrictions on the size of q, though),
- transcendental or algebraic extensions of $\mathbb{Q}$ and $\mathbb{F}_p$,
- $\mathbb{R}$ and $\mathbb{C}$ (user defined precision),
- $\mathbb{Z}$ and $\mathbb{Z}/n\mathbb{Z}$ for $n \in \mathbb{Z}$.

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

- $\mathbb{Q}$,
- $\mathbb{F}_q$ (there are weird restrictions on the size of q, though),
- transcendental or algebraic extensions of $\mathbb{Q}$ and $\mathbb{F}_p$,
- $\mathbb{R}$ and $\mathbb{C}$ (user defined precision),
- $\mathbb{Z}$ and $\mathbb{Z}/n\mathbb{Z}$ for $n \in \mathbb{Z}$.

From now on called `ring`.

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

- $\mathbb{Q}$,
- $\mathbb{F}_q$ (there are weird restrictions on the size of q, though),
- transcendental or algebraic extensions of $\mathbb{Q}$ and $\mathbb{F}_p$,
- $\mathbb{R}$ and $\mathbb{C}$ (user defined precision),
- $\mathbb{Z}$ and $\mathbb{Z}/n\mathbb{Z}$ for $n \in \mathbb{Z}$.

From now on called `ring`.

Whatever you can do in/over a `ring`:

- polynomials,

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

- $\mathbb{Q}$,
- $\mathbb{F}_q$ (there are weird restrictions on the size of q, though),
- transcendental or algebraic extensions of $\mathbb{Q}$ and $\mathbb{F}_p$,
- $\mathbb{R}$ and $\mathbb{C}$ (user defined precision),
- $\mathbb{Z}$ and $\mathbb{Z}/n\mathbb{Z}$ for $n \in \mathbb{Z}$.

From now on called `ring`.

Whatever you can do in/over a `ring`:

- polynomials,
- ideals,

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

- $\mathbb{Q}$,
- $\mathbb{F}_q$ (there are weird restrictions on the size of q, though),
- transcendental or algebraic extensions of $\mathbb{Q}$ and $\mathbb{F}_p$,
- $\mathbb{R}$ and $\mathbb{C}$ (user defined precision),
- $\mathbb{Z}$ and $\mathbb{Z}/n\mathbb{Z}$ for $n \in \mathbb{Z}$.

From now on called `ring`.

Whatever you can do in/over a `ring`:

- polynomials,
- ideals,
- modules,

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

- $\mathbb{Q}$,
- $\mathbb{F}_q$ (there are weird restrictions on the size of q, though),
- transcendental or algebraic extensions of $\mathbb{Q}$ and $\mathbb{F}_p$,
- $\mathbb{R}$ and $\mathbb{C}$ (user defined precision),
- $\mathbb{Z}$ and $\mathbb{Z}/n\mathbb{Z}$ for $n \in \mathbb{Z}$.

From now on called `ring`.

Whatever you can do in/over a `ring`:

- polynomials,
- ideals,
- modules,
- localizations,

What can it work with?

Multivariate polynomial rings $K[x_1, \ldots, x_n]$ where K can be

- $\mathbb{Q}$,
- $\mathbb{F}_q$ (there are weird restrictions on the size of q, though),
- transcendental or algebraic extensions of $\mathbb{Q}$ and $\mathbb{F}_p$,
- $\mathbb{R}$ and $\mathbb{C}$ (user defined precision),
- $\mathbb{Z}$ and $\mathbb{Z}/n\mathbb{Z}$ for $n \in \mathbb{Z}$.

From now on called `ring`.

Whatever you can do in/over a `ring`:

- polynomials,
- ideals,
- modules,
- localizations,
- quotient rings.

What can it do?

What can it do?

Gröbner bases

(A special generating system for an ideal in a `ring`.)

What can it do?

Gröbner bases

(A special generating system for an ideal in a `ring`.)

Wait, what? That's all?

What do you do with Gröbner bases?

- Ideal Membership, i.e. decide whether $f \in R$ is in I

What do you do with Gröbner bases?

- Ideal Membership, i.e. decide whether $f \in R$ is in I
- Solving non-linear polynomial equations

What do you do with Gröbner bases?

- Ideal Membership, i.e. decide whether $f \in R$ is in I
- Solving non-linear polynomial equations
- Ideal/Module operations: Radicals, Intersections, Quotients, Saturations

What do you do with Gröbner bases?

- Ideal Membership, i.e. decide whether $f \in R$ is in I
- Solving non-linear polynomial equations
- Ideal/Module operations: Radicals, Intersections, Quotients, Saturations
- Kernels of `ring` morphisms

What do you do with Gröbner bases?

- Ideal Membership, i.e. decide whether $f \in R$ is in I
- Solving non-linear polynomial equations
- Ideal/Module operations: Radicals, Intersections, Quotients, Saturations
- Kernels of `ring` morphisms

Those are basic parts for more complicated algorithmic tasks like, e.g.

- Noether Normalization,
- Primary decomposition,
- Free resolution of a module,
- Invariants,
- ...

Let's get started

```
doctor@tardis:~ $ Singular
                     SINGULAR                                 /
 A Computer Algebra System for Polynomial Computations       /   version 4.2.1
                                                            0<
 by: W. Decker, G.-M. Greuel, G. Pfister, H. Schoenemann     \   May 2021
FB Mathematik der Universitaet, D-67653 Kaiserslautern        \
>
```

Let's get started

Now we have to define a `ring`.

Let's get started

Now we have to define a `ring`. For $\mathbb{Q}[x, y, z]$ we use the following syntax:

```
> ring R = 0, (x, y, z), dp;
```

Let's get started

Now we have to define a `ring`. For $\mathbb{Q}[x, y, z]$ we use the following syntax:

```
> ring R = 0, (x, y, z), dp;
```

`ring`	variable type

Let's get started

Now we have to define a `ring`. For $\mathbb{Q}[x, y, z]$ we use the following syntax:

```
> ring R = 0, (x, y, z), dp;
```

`ring`	variable type
`R`	variable name

Let's get started

Now we have to define a `ring`. For $\mathbb{Q}[x, y, z]$ we use the following syntax:

```
> ring R = 0, (x, y, z), dp;
```

`ring`	variable type
`R`	variable name
`0`	characteristic
`(0, a)`	and optional list of parameters

Let's get started

Now we have to define a `ring`. For $\mathbb{Q}[x, y, z]$ we use the following syntax:

```
> ring R = 0, (x, y, z), dp;
```

`ring`	variable type
`R`	variable name
`0`	characteristic
`(0, a)`	and optional list of parameters
`(x, y, z)`	the variables (of the polynomial ring)
`(x(1..3), y)`	

Let's get started

Now we have to define a `ring`. For $\mathbb{Q}[x, y, z]$ we use the following syntax:

```
> ring R = 0, (x, y, z), dp;
```

`ring`	variable type
`R`	variable name
`0`	characteristic
`(0, a)`	and optional list of parameters
`(x, y, z)`	the variables (of the polynomial ring)
`(x(1..3), y)`	
`dp`	monomial ordering
`(dp, c)`	optional additional ordering for module elements

Let's get started

Now we have to define a `ring`. For $\mathbb{Q}[x, y, z]$ we use the following syntax:

```
> ring R = 0, (x, y, z), dp;
```

`ring`	variable type
`R`	variable name
`0`	characteristic
`(0, a)`	and optional list of parameters
`(x, y, z)`	the variables (of the polynomial ring)
`(x(1..3), y)`	
`dp`	monomial ordering
`(dp, c)`	optional additional ordering for module elements
`;`	yes, SINGULAR needs semicolons

Now that we have a `ring`...

```
> ring R = 0, (x, y, z), dp;
```

Now that we have a `ring`...

```
> ring R = 0, (x, y, z), dp;
```

We can define polynomials

```
> poly f = x^2 + y;
> poly g = 1/2*x;
```

Now that we have a `ring`...

```
> ring R = 0, (x, y, z), dp;
```

We can define polynomials

```
> poly f = x^2 + y;
> poly g = 1/2*x;
```

and ideals

```
> ideal I = f, g;
```

Now that we have a `ring`...

```
> ring R = 0, (x, y, z), dp;
```

We can define polynomials

```
> poly f = x^2 + y;
> poly g = 1/2*x;
```

and ideals

```
> ideal I = f, g;
```

and modules

```
> module M = [x^2, y], [y^2 - x, 0];
```

Now that we have a `ring`...

And do "basic" operations with them:

```
> f + g;
x2+1/2x+y
> f*g;
1/2x3+1/2xy
> f^2;
x4+2x2y+y2

> I^2;
_[1]=x4+2x2y+y2
_[2]=1/2x3+1/2xy
_[3]=1/4x2
> I + ideal(x + y);
_[1]=x2+y
_[2]=1/2x
_[3]=x+y
```

And of course we can do Gröbner bases!

```
> ring R = 0, (x, y), dp;
> ideal I = x^2 + y, x*y + x;
> groebner(I);
_[1]=y2+y
_[2]=xy+x
_[3]=x2+y
```

And of course we can do Gröbner bases!

```
> ring R = 0, (x, y), dp;
> ideal I = x^2 + y, x*y + x;
> groebner(I);
_[1]=y2+y
_[2]=xy+x
_[3]=x2+y

> ring R = 0, (x, y, z), (c, dp);
> module M = [x, y, 1], [xy, z, z^2];
> groebner(M);
_[1]=[0,y2-z,-z2+y]
_[2]=[x,y,1]
```

Leaving the shell

You can load a file "`file`" and execute the contained code:

- Directly on the command line

```
doctor@tardis:~ $ Singular file
```

Leaving the shell

You can load a file "`file`" and execute the contained code:

- Directly on the command line

  ```
  doctor@tardis:~ $ Singular file
  ```

- or inside SINGULAR

  ```
  > < "file";
  ```

Leaving the shell

You can load a file "`file`" and execute the contained code:

- Directly on the command line

```
doctor@tardis:~ $ Singular file
```

- or inside SINGULAR

```
> < "file";
```

For more complex projects: Write a library!

Leaving the shell

You can load a file "`file`" and execute the contained code:

- Directly on the command line

```
doctor@tardis:~ $ Singular file
```

- or inside SINGULAR

```
> < "file";
```

For more complex projects: Write a library!

- Filename should end in `.lib`, say, `file.lib`.

Leaving the shell

You can load a file "`file`" and execute the contained code:

- Directly on the command line

```
doctor@tardis:~ $ Singular file
```

- or inside SINGULAR

```
> < "file";
```

For more complex projects: Write a library!

- Filename should end in `.lib`, say, `file.lib`.
- Can be loaded in SINGULAR via

```
> LIB "file.lib";
```

Leaving the shell

You can load a file "`file`" and execute the contained code:

- Directly on the command line

```
doctor@tardis:~ $ Singular file
```

- or inside SINGULAR

```
> < "file";
```

For more complex projects: Write a library!

- Filename should end in `.lib`, say, `file.lib`.
- Can be loaded in SINGULAR via

```
> LIB "file.lib";
```

- Then all procedures (aka functions) in `file.lib` can be used.

Leaving the shell

You can load a file "`file`" and execute the contained code:

- Directly on the command line

```
doctor@tardis:~ $ Singular file
```

- or inside SINGULAR

```
> < "file";
```

For more complex projects: Write a library!

- Filename should end in `.lib`, say, `file.lib`.
- Can be loaded in SINGULAR via

```
> LIB "file.lib";
```

- Then all procedures (aka functions) in `file.lib` can be used.
- Should start with a `info`-string.

Leaving the shell

Syntax for procedures:

```
proc the_answer(int i, ideal I, poly f)
{
   return(42);
}
```

Leaving the shell

Syntax for procedures:

```
proc the_answer(int i, ideal I, poly f)
{
   return(42);
}
```

And then call it:

```
> LIB "answer.lib";
// ** loaded answer.lib
> def j = the_answer(1, ideal(x), x);
> j;
42
> typeof(j);
int
```

What's that dp-stuff about?

For Gröbner bases we need to order monomials!

What's that dp-stuff about?

For Gröbner bases we need to order monomials!

For univariate rings it's obvious: $1 < x < x^2 < x^3 < \ldots$.

What's that dp-stuff about?

For Gröbner bases we need to order monomials!

For univariate rings it's obvious: $1 < x < x^2 < x^3 < \ldots$.

But what about more variables? Is $x < y$ or $x > y$? Is $xy < x$?

What's that dp-stuff about?

For Gröbner bases we need to order monomials!

For univariate rings it's obvious: $1 < x < x^2 < x^3 < \ldots$.

But what about more variables? Is $x < y$ or $x > y$? Is $xy < x$?

A monomial ordering is a total ordering on the monomials of a `ring`, which is well-behaved w.r.t. multiplications, i.e.

$$x^\alpha > x^\beta \Longrightarrow x^\gamma x^\alpha > x^\gamma x^\beta \ .$$

What's that dp-stuff about?

For Gröbner bases we need to order monomials!

For univariate rings it's obvious: $1 < x < x^2 < x^3 < \ldots$.

But what about more variables? Is $x < y$ or $x > y$? Is $xy < x$?

A monomial ordering is a total ordering on the monomials of a `ring`, which is well-behaved w.r.t. multiplications, i.e.

$$x^\alpha > x^\beta \Longrightarrow x^\gamma x^\alpha > x^\gamma x^\beta \ .$$

For almost all computations in multivariate polynomial rings, we need to specify the monomial ordering.

What's that dp-stuff about?

Different options for different problems

- standard orderings like
  - lexicographic `lp`,
  - reverse-lexicographic `rp`,
  - degree-reverse-lexicographic `dp`,
  - ...

What's that dp-stuff about?

Different options for different problems

- standard orderings like
  - lexicographic `lp`,
  - reverse-lexicographic `rp`,
  - degree-reverse-lexicographic `dp`,
  - ...
- weighted orderings `wp(w1,...,wn)`

What's that dp-stuff about?

Different options for different problems

- standard orderings like
  - lexicographic `lp`,
  - reverse-lexicographic `rp`,
  - degree-reverse-lexicographic `dp`,
  - ...
- weighted orderings `wp(w1,...,wn)`
- matrix orderings

What's that dp-stuff about?

Different options for different problems

- standard orderings like
  - lexicographic `lp`,
  - reverse-lexicographic `rp`,
  - degree-reverse-lexicographic `dp`,
  - ...
- weighted orderings `wp(w1,...,wn)`
- matrix orderings
- local orderings like `ls`, `ds`, `ws`...

What's that dp-stuff about?

Different options for different problems

- standard orderings like
  - lexicographic `lp`,
  - reverse-lexicographic `rp`,
  - degree-reverse-lexicographic `dp`,
  - ...
- weighted orderings `wp(w1,...,wn)`
- matrix orderings
- local orderings like `ls`, `ds`, `ws`...
- mixtures of those:

```
> ring R = 0, (x(1..3), y(1..2)), (dp(3), lp(2));
```