Gröbner bases -

An introduction

© Per Andersson
Luleå University of Technology, 1995-1997

Per Andersson
per@planethilmer.com
http://www.planethilmer.com/per/math

Overview

Gröbner bases

1. **What its all about**
   A two-minute introduction to Gröbner bases.

2. **Applications**
   - Solving equations
   - Ideal theory

3. **Theory**
   - Constructing objects - reduction and Buchbergers algorithm

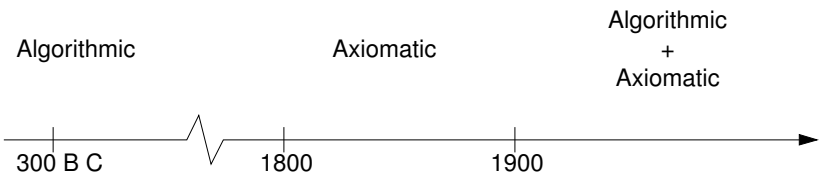
4. **How this assignement was carried out**

5. **Where to find Gröbner bases**

6. **Appendix:**
   - Software for Gröbner Bases
   - Functions for Gröbner Bases in Maple
   - WWW sites

Overview

Development of algebra



algorithmic constructions	properties of objects	computing with objects
Euclides alg.	Group Ring	Gröbner base
	short, elegant	

Computing with objects

Computing with even the simplest objects can be difficult and expensive.

Example: Computing with integers like 10^{500} is expensive (computation time, storage).

Solution: Approximate and use floating point numbers.

Problem: Approximation and original objects have different properties.

The integer 10^{500} *has* a successor (1000…001).
($10^{500} + 1 - 10^{500} = 1$)

The floating point number `10E500` does *not* have a successor!
(`10E500+1-10E500=0`[1])

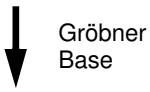
1. There are workarounds

Applications

1. **Example: Solving a system of equations**
2. **Methods offered by Gröbner bases**
   In what ways can Gröbner bases help when solving equations?
3. **Ideal theory**
   Gröbner bases offers a complete solution to several ideal theoretic problems like ideal membership.
4. **Computational aspects**
   Things to think about when using Gröbner bases.

Example

$$(1)\ \begin{cases} xy + 1 = 0 \\ x^2 + x = 0 \end{cases} \longrightarrow P = \{xy + 1, x^2 + x\}$$



$$(2)\ \begin{cases} 1 + x = 0 \\ y - 1 = 0 \end{cases} \longleftarrow G = \{1 + x, y - 1\}$$

The theory tells us that the systems (1) and (2) have exactly the same set of solutions.

Methods offered by GB

- **Solvability**
  Do a system of equations have a solution?
- **Finite/infinite number of solutions**
  Are there an infinite number of solutions (parametric solution) or not?
- **Exact number of solutions**
  Determine the exact number of solutions without solving the system.
- **Solving a system of equations**
  Two variants: Elegant and slow or hard and not-so-slow.
- **Gröbner bases compared with numerical methods**
  Are Gröbner bases the solution to every problem?

Theory

Term orderings

For univariate polynomials (in one variable) there is a natural ordering of terms, but for multivariate polynomials (several variables) no such natural ordering exists.

The two most common orderings are *lexiographic* and *total degree*.

Lexiographic ordering works just like when sorting words.

Total degree ordering is a two-step process: Sort terms by total degree and sort terms of equal degree lexiographically.

To understand the lexiographic ordering, think of a term as a word. E.g the term $x^2y^0z^3 = x^2z^3$ should be sorted as the word xxzzz.

By default the *ordering of variables* are the usual one (i.e xx comes before xy) but it is common to use a custom ordering of variables. E.g lexiographic ordering with $y > x$ will put xy before xx.

Theory

Term orderings

- **Lexiographic ordering** (called "`plex`" in Maple) $x > y > z$

  1. Write each term as a word.
  2. Define ordering of variables
  3. Sort as when sorting names

x^2z	xxz	$xxxyyy$	x^3y^3
x^2y^2z	$xxyyz$	$xxxz$	x^3z
x^3y^3	$xxxyyy$	$xxyyz$	x^2y^2z
x^3z	$xxxz$	xxz	x^2z

- **Total degree ordering** (called "`tdeg`" in Maple) $x > y > z$

  1. Sort by total degree
  2. Sort terms of equal degree lexiographically.

xz	xyz	xyz	xxy	x^2y
xyz	x^2z	xxz	xxz	x^2z
x^2z	x^2y	xxy	xyz	xyz
x^2y	xz	xz	xz	xz

Methods offered by GB

- **Solvability**

Let $P = \{p_1, \ldots, p_r\}$ be an equation system written as a set of polynomials and let G be the Gröbner base of P. The system is solvable if and only if $1 \notin G$ ($1 \in G$ implies that the equation $1 = 0$ has a solution which is a contradiction).

$P = \{xy^2, x-1, y-1\}$

$G = \{1\}$

Not solvable.

$P = \{xy^2, y-1\}$

$G = \{x-1, y-1\}$

Solvable.

Methods offered by GB

- **Finite/infinite number of solutions**

Let $P = \{p_1, \ldots, p_r\}$ be an equation system in variables $x_1, \ldots, x_n$ written as a set of polynomials and let G be the Gröbner base of P. Let

$$H = \{hterm(G)\}.$$

Then the equation system associated with P has finitely many solutions if and only if there for all x_i is a positive integer m such that $(x_i)^m \in H$.

$P = \{xy + 1, x^2 + x\}$

$G = \{1 + x, y - 1\}$

$H = \{x, y\}$

Finite number of solutions.

$P = \{xy, x^2 + x\}$

$G = \{x^2 + x, xy\}$

$H = \{x^2, xy\}$

Infinite number of solutions.

Methods offered by GB

- **Number of solutions**

Methods offered by GB

- **Solving a system of equations, lexical ordering** $x > y > z$

  Lexical ordering leads to a triangularization of the system:

$$\begin{cases} zy = 0 \\ x + y = 0 \\ zx = 0 \\ xy + z^2x = 0 \end{cases} \xrightarrow{\text{Gröbner Base}} \begin{cases} x + y = 0 \\ y^2 = 0 \\ zy = 0 \end{cases}$$

  1. Solve for z (the ”smallest” variable according to $x > y > z$).
     In this example the Gröbner base does not contain a polynomial that depends only on z, so z can take any value.
  2. Solve for y.
     We have two equations that depends on z and y. The second equation bounds y to 0.
  3. Solve for x.
     The first equation bounds x to 0.

Methods offered by GB

- **Solving a system of equations, total degree ordering**

  Total degree ordering does *not* lead to a triangularization of the system:

$$\begin{cases} xyz + 8z^2 = 0 \\ x^2y - 7z = 0 \\ 2x - 2y + 1 = 0 \end{cases} \xrightarrow{\text{Gröbner Base}} \begin{cases} 3z^2 + 8z^2y = 0 \\ 21z^2 + 512z^3 = 0 \\ 2x - 2y + 1 = 0 \\ -28z + 4y^3 - 4y^2 + y = 0 \\ 16z^2 + 2zy^2 - zy = 0 \end{cases}$$

  Use a special function to solve for each variable.

Methods offered by GB

Now we will solve the system in Maple using Gröbner bases:

1. Calculate the Gröbner base.

```
>G1:=grobner[gbasis]({z*x*y+8*z^2,x^2*y-7*z,2*x-
2*y+1},[x,y,z],tdeg)
    G1:=[-28z+4y^3-4y^2+y,2x-2y+1,
    16z^2+2y^2z-zy,21z^2+512z^3,3z^2+8yz^2]
```

2. Solve for one variable, e.g z.

```
>grobner[finduni](z,G1);
    21z^2+512z^3
>solve(");
    0,0,-21/512
```

3. Repeat steps 4-8 for each solution w.r.t z (only one is shown here).
4. Substitute 0 for z in G1 and use the result to calculate a new Gröbner base.

```
>z:=0;
    z:=0;
>G2:=grobner[gbasis](eval(G1),[x,y],tdeg);
    G2:=[2x-2y+1,4y^3-4y^2+y];
```

Methods offered by GB

5. Solve for x.

```
>grobner[finduni](x,G2);
    x^2+2x^3;
>solve(”);
    0,0,-1/2
```

6. Repeat the steps 7-8 for each solution w.r.t x (only one is shown here).
7. Substitute 0 for x in G2 and use the result to calculate a new Gröbner base.

```
>x:=0;
    x:=0;
>G3:=grobner[gbasis](eval(G2),[y],tdeg);
    G3:=[2y-1];
```

8. Solve for y.

```
>grobner[finduni](y,G3);
    -2y+1
>solve(”);
    1/2
```

We now have one solution:

```
    (x,y,z)=(0,1/2,0)
```


Methods offered by GB

- **Gröbner bases compared with numerical methods**

Gröbner bases	**Numerical methods**
Slow	Fast
Exact solutions	Approximate solutions
Parametric solutions	

Ideal theory

With Gröbner bases, a number of problems in ideal theory are solved:

- **Ideal membership**
  Is an element a member of a given ideal?
- **Calculating with residue classes (modulo an ideal)**
- **Subideal**
  Given two ideals I and J, are $I \subseteq J$?
- **Proper subset**
  Given two ideals I and J, are $I \subset J$?
- **Intersection of ideals**
- **Intersection of residue classes**

Computational aspects

- Total degree ordering is as fast as/faster than the ”best” lexiographic ordering.
- Often high complexity
- Lexiographic ordering is very sensitive to different permutations of variables (”unstable”)

Theory

1. **Construction of objects and calculus**

2. **Integers**
   $Z \to Z/\mathrm{aZ}$

3. **Univariate polynomials**
   $K[X] \to K[X]/\langle p \rangle$

4. **Multivariate polynomials**
   $K[X_1, \ldots, X_n] \to K[X_1, \ldots, X_n]/\langle p_1, \ldots, p_m \rangle$

Construction of objects and calculus

Procedure:

1. Decide for a set of objects to start from.

2. Construct an equivalence relation.

3. Set of new objects <=> set of equivalence classes.

4. For each equivalence class, decide for one object that will represent the class (called the *class representative*).

5. Construct a function $Reduce(.)$ that ”reduces” an object to its class representative.

- We now have a *new set of objects* which we can manipulate in more or less the same way as the original objects (e.g add and multiply).

- The new objects have *properties inherited* from the original objects. They also have some *new properties* that come from the equivalence relation.

Theory

Integers

Objective: Split integers in two parts - quotient and remainder after division by 3.

Procedure:

1. Z
2. $a \sim b \Leftrightarrow a \bmod 3 = b \bmod 3$
3. New objects: $\{\ldots, -3,0,3,\ldots\}, \{\ldots,-2,1,4,\ldots\},\{\ldots,-1,2,5,\ldots\}$.
4. Representatives: 0, 1, 2
5. $Reduce(a) = a \bmod 3$

Example:

(in Z) $2 + 8 = 10$

(mod 3) $Reduce(2 + 8) = Reduce(10) = 1$

$Reduce(Reduce(2) + Reduce(8)) = Reduce(2 + 2) = Reduce(4) = 1$

The new objects can be treated the same way as integers, but by using the *Reduce* function we mask away the quotient part and only considers the remainder part.

Theory

Univariate polynomials

Objective: Split polynomials in two parts - quotient and remainder after division by $q = x^2$.

Procedure:

1. $R[X]$
2. $p_1 = a_1 q + r_1 \qquad p_2 = a_2 q + r_2$

   $p_1 \sim p_2 \Leftrightarrow r_1 = r_2$
3. New objects:

   $\{\mathbf{3x}, x^2 + 3x, x^3 + x^2 + 3x, \ldots\}$,

   $\{\mathbf{x + 6}, x^2 + x + 6, 7x^2 + x + 6, \ldots\}$,

   $\{\mathbf{18x}, 2x^2 + 18x, 3x^3 - x^2 + 18x, \ldots\}$,

   …
4. Representatives: **3x**, **x + 6**, **18x**
5. $Reduce(p) = p \bmod q$

Theory

Univariate polynomials

Example:

(in $R[X]$) $(x^2 + 3x) \cdot (x + 6) = x^3 + 9x^2 + 18x$

(polmod x^2) $Reduce((x^2 + 3x) \cdot (x + 6)) = Reduce(x^3 + 9x^2 + 18x) = 18x$

$$Reduce(Reduce(x^2 + 3x) \cdot Reduce(x + 6)) = Reduce((3x) \cdot (x + 6))$$
$$= Reduce(3x^2 + 18x) = 18x$$

The new objects can be treated the same way as ordinary polynomials, but by using the *Reduce* function we mask away the quotient part and only considers the remainder part.

Theory

Multivariate polynomials

Objective: Split polynomials in two parts - quotient and remainder after "division" by $q_1 = x^2y + y^2$ and $q_2 = xy^2 + x^2$. "Division" in this case means to write a polynomial as a linear combination of q_1, q_2 and, in most cases, a remainder:

$$p = a_1q_1 + a_2q_2 + r$$

The new object will in this case be r.

Procedure:

1. $R[X, Y]$
2. $p_1 = a_{11}q_1 + a_{12}q_2 + r_1 \qquad p_2 = a_{21}q_1 + a_{22}q_2 + r_2$

$p_1 \sim p_2 \Leftrightarrow r_1 = r_2$

But how do we write a polynomial as a linear combination of the base polynomials?

When we had only *one univariate base polynomial* we could use polynomial division.

We need to generalize polynomial division to cover *several* and *multivariate base polynomials*!

Theory

Reduction

To reduce

To *reduce* one polynomial p with another polynomial q in this case means to subtract a multiple of q from p:

$$r = p - aq$$

Example:

$$p = 2x^3y^2 + 2x^2y + 7y \qquad q = x^2y + x$$

We can reduce p with $2xy \cdot q = 2x^3y^2 + 2x^2y$ and have

$$r = p - 2x^2y \cdot q = 7y$$

This reduction of p to r is written as

$$p \to r$$

and $7y$ is our ”new object”.

Theory

Reduction

Set of reducers

Given a polynomial p and a set of polynomials (”reducers”) Q, we want to know which, if any, polynomials in Q that can reduce p.

This is called the ”reducer set” and is calculated as

$$R(p, Q) = \{q \in Q - \{0\} \text{ such that } hterm(q) | hterm(p)\}$$

Example:

$$p = xy \qquad Q = \{x, y, xy, z, x^2y\}$$

Then

$$R(p, Q) = \{x, y, xy\}$$

Theory

Reduction

Reduction algorithm

procedure `Reduce(p,Q)`

```
q:=0;
while p!=0 do
   while R(p,Q)!={} do
      q:=selectpoly(R(p,Q));
      a:=hmon(p)/hmon(q);
      p:=p-a*q;
   end;
   r:=r+hmon(p);
   p:=p-hmon(p);
end;
return(r);
```


Theory

Reduction

Example

Let

$$q_1 = xy + x \qquad q_2 = xy + y$$

and

$$p = xy + x + y$$

In the first iteration we have

$$R(p, Q) = \{q_1, q_2\}$$

If we choose q_1 we get $a = 1$ and $p = y$. We can not reduce any further so our new object is y.
If we choose q_2 we get $a = 1$ and $p = x$. We can not reduce any further so our new object is x.

If we begin with $p = q_1 - q_2 = x - y$ we can not reduce anything at all. But since p is a linear combination of q_1 and q_2 it *should* reduce to *zero*!

Theory

Multivariate polynomials

The last example demonstrated several weaknesses with our reduction algorithm:

- The result is not unique. Depending on how we implement *selectpoly* we can end up with different decompositions.
  Both decompositions are correct, but it would be nice if there was only one possible result (one posible ”new object”).
- The reduction algorithm cannot reduce every polynomial that is reducible (or *should* be reducible).

Is the reduction algorithm useless?

Lets try to modify the base. We *extend* the base with the linear combination $q_1 - q_2$ and see what happpens.

Theory

Reduction

Example (continued from p t.4.2.4)

Let

$$q_1 = xy + x \qquad q_2 = xy + y \qquad q_3 = x - y$$

and

$$p = xy + x + y$$

In the first iteration we have

$$R(p, Q) = \{q_1, q_2, q_3\}$$

If we choose q_1 we get $a = 1$ and $p = y$. We can not reduce any further so our new object is y.
If we choose q_2 we get $a = 1$ and $p = x$. We *can* reduce further and our new object is $x - (x - y) = y$ which is the *same* (new) *object!*

If we begin with $p = q_1 - q_2 = x - y$ we can reduce to zero in one step.

Theory

Multivariate polynomials

As we have seen, there are some serious problems with the reduction algorithm. We solved the problems by modifying the base.

As it turns out, the problems are not related to the algorithm, but rather to the structure of the ”base” (the ”base polynomials” $q_1, q_2, \dots$).

With the help of another algorithm, **Buchbergers algorithm**, a base consisting of a number of polynimials can be transformed into a **Gröbner base**. Buchbergers algorithm will extend a given base with certain new elements, much in the same way as we did in the last example. Such a modified base will give us a *different decomposition* but the *same result* (same new object, same remainder).

It can be proved that the reduction algorithm will be gaurenteed to always work on any Gröbner base, and since there for every polynomial base exists one unique Gröbner base that is always computable (in finite time), we *can* use the reduction algorithm to decompose a polynomial!

Theory

Buchbergers algorithm

```
procedure Gbasis(P)

G:=P;
k:=length(G);
B:={[i,j]:1<=i<j<=k};
while B!={} do
    [i,j]:=selectpair(B,G);
    B:=B-{[i,j]};
    h:=Reduce(Spoly(G[i],G[j]),G);
    if h!=0 then
        G:=G union {h};
        k:=k+1;
        B:=B union {[i,j]:1<=i<k};
    end;
end;
return(G);
```

Theory

Multivariate polynomials

3. New objects (continued from page t.4.1)

   $\{\mathbf{3x}, 3x + x^2y + y^2, 3x + x^3y + xy^2, \ldots\}$,

   $\{\mathbf{y + 6}, y + 6 + xy^2 + x^2, 6 + y + xy^3 + x^2y, \ldots\}$,

   $\{\mathbf{3xy + 18x}, 3xy + 18x + x^2y + y^2, \ldots\}$,

   …

4. Representatives: **3x**, **y + 6**, **3xy + 18x**

5. $Reduce(p) = Reduce(p, Gbasis(\{q_1, q_2\}))$

Theory

Multivariate polynomials

Using Gröbner Bases we can…

…construct objects with customized properties.

…enumerate/construct a basis for the new objects.

…perform operations on the new objects by lifting objects to the original domain, performing the operation and reduce the result back to the new domain.

The Work

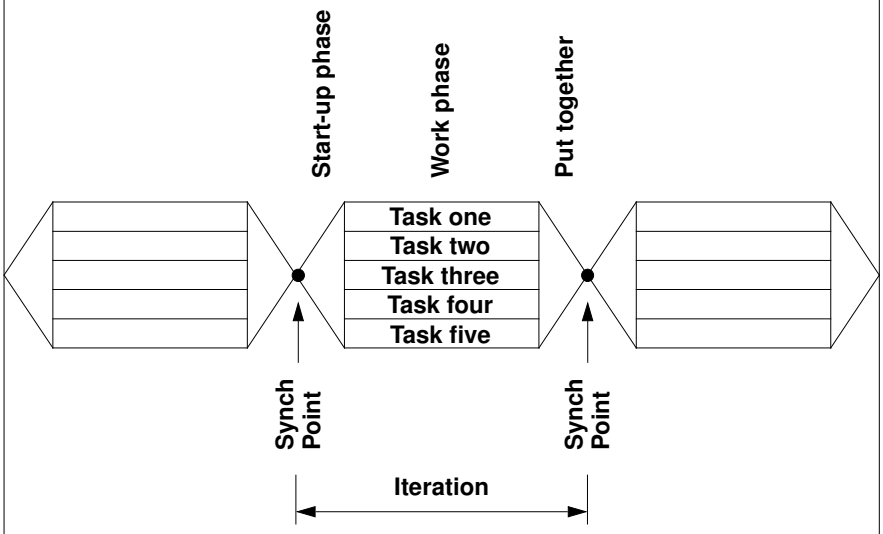
1. **Goals with the project**
2. **How to execute the project**
3. **How to organize the material**

Goals

- Texts about Gröbner Bases can roughly be divided into two categories:
  (1) Introductions requiring almost no mathematical background.
  (2) Complete texts on a graduate level.
  The goal for this project was to write a textbook about Gröbner bases. It should be accessible to *anyone* with a M.Sc degree (mechanical engineering, industrial and management engineering, ...), and at the same time be mathematically correct.
- It should also teach the reader some basic ”mathematical thinking”. For example, in the beginning there will be a lot of explanations of what is going on.
- It should be easy to read and understand without loosing mathematical precision.

The work

Execution of the project



a.2.1

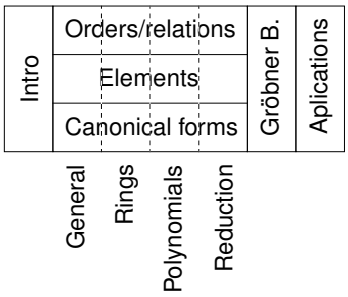
The work

Execution of the project

- Underestimated the time required for ”put together”.
- Good utilization of time when working in parallell.
- During the ”work phase” I was often unsure about how far the total work had proceeded.

a.2.3

How to organize the material



Can be used in two ways:

1. Chapter order: General -> Rings -> Polynomials -> Reduction
2. Subject order: Orders/relations -> Elements -> Canonical forms

Appendix

- **Software for calculating with Gröbner bases.**
- **Gröbner bases in Maple.**
- **Sites on the WWW dedicated to Gröbner bases.**

Appendix

Software

- **Maple** (commercial)
  Package ”grobner”. See help browser under ”Mathematics/Algebra/Polynomials/Grobner Bases.
- **MuPAD** (free)
  Multi Processing Algebra Tool
  Public domain. Comparable to Maple.
  `http://math-www.uni-paderborn.de/MuPAD/`
- **SACLIB** (free)
  Symbolic Algebraic Computation LIBrary. A C-library for performing symbolic computation.
  `http://www.can.nl/SystemsOverview/Special/Algebra/SACLIB.html`
- **Groebner** (free)
  A C-library for computing with Gröbner bases.
  `http://www.can.nl/SystemsOverview/Special/Algebra/GROEBNER/productinfo.html`
- **Macaulay** (free)
  Algebraic geometry and computer algebra
  `http://www.math.uiuc.edu/Macaulay2/`

Appendix

Gröbner Bases in Maple

Function in this tutorial	Function in Maple
$hterm(.)$	`grobner[leadmon]`
$Reduce(.)$	`grobner[normalf]`
$Gbasis(.)$	`grobner[gbasis]`
$Spoly(.)$	`grobner[spoly]`

Appendix

WWW sites

- **RISC**
  Research Institute for Symbolic Computation.
  Directed by Prof. Bruno Buchberger, the inventor of Gröbner bases.
  `http://www.risc.uni-linz.ac.at/`

- **CAIN**
  Computer Algebra Information Network
  Information service dedicated to computer algebra.
  `http://math-www.uni-paderborn.de/CAIN/`

- My home page where my work on Gröbner Bases can be found (including a textbook in swedish).
  `http://www.ludd.luth.se/~per/GB/`