INDIAN INSTITUTE OF TECHNOLOGY
GANDHINAGAR

GRÖBNER BASES OF RATIONAL NORMAL CURVES

AMOGH PARAB (14110089)

KSHITEEJ JITESH SETH (14110068)

SUPERVISOR: DR. INDRANATH SENGUPTA

August 27, 2016

Abstract

Let $n \geq 3$ be a natural number. Let $R = k[x_0, \ldots, x_n]$ be the polynomial ring in the indeterminates $x_0, x_1, \ldots, x_n$ over a field k. Let $A = \begin{bmatrix} x_0 & x_1 & \cdots & x_{n-1} \\ x_1 & x_2 & \cdots & x_n \end{bmatrix}$. Let $\mathcal{G}_n$ denote the set of all 2×2 minors of the matrix A, i.e., $\mathcal{G}_n = \{x_i x_{j+1} - x_{i+1} x_j \mid 0 \leq i < j \leq n\}$. Let I denote the ideal generated by $\mathcal{G}_n$ in $k[x_0, x_1, ..., x_n]$. Suppose that the monomial ordering in $R = k[x_0, x_1, ..., x_n]$ is given by $x_{i_0} > x_{i_1} > \ldots > x_{i_n}$, with the lexicographic ordering of monomials in R, where $(i_0, i_1, \ldots, i_n)$ denotes a permutation of the set $\{0, 1, \ldots, n\}$. We have classified all possible permutations $(i_0, i_1, \ldots, i_n)$ of $\{0, 1, \ldots, n\}$ such that $\mathcal{G}_n$ is a Gröbner basis of the ideal I.

Definitions

Definition 1. Set $S_k \subset \mathbb{N}$:
If monomial ordering in $R = k[x_0, x_1, \ldots, x_n]$ is given by $x_{i_0} > x_{i_1} > \ldots > x_{i_k} > \ldots > x_{i_n}$, then the set S_k is defined as $S_k = \{i_k, i_{k+1}, i_{k+2}, ..., i_n\}$; $k = 0, 1, \ldots, n$

Remark: S_0 is full set $\{0, 1, \ldots, n\}$ and S_n is singleton set $\{i_n\}$.

Example: For monomial ordering $x_2 > x_0 > x_1 > x_4 > x_3$ in $R = k[x_0, x_1, \ldots, x_4]$,
$S_0 = \{2, 0, 1, 4, 3\}$, $S_1 = \{0, 1, 4, 3\}$, $S_2 = \{1, 4, 3\}$, $S_3 = \{4, 3\}$, $S_4 = \{3\}$.

Definition 2. Property P_j for given monomial order:
If the monomial ordering in $R = k[x_0, x_1, \ldots, x_n]$ is given by $x_{i_0} > x_{i_1} > \ldots > x_{i_j} > \ldots > x_{i_n}$,where $0 \leq j \leq n$ then the given monomial order is said to satisfy property P_j if i_k is either $max(S_k)$ or $min(S_k)$ $\forall k \leq j$

Remark: If the given monomial order satisfies the property P_j, $0 < j \leq n$ then it satisfies property P_k $\forall k < j$.

Example: For monomial ordering $x_0 > x_5 > x_1 > x_3 > x_4 > x_2$ in $R = k[x_0, x_1, \ldots, x_5]$, it satisfy properties P_0, P_1 and P_2; but NOT P_3, P_4 and P_5.

Theorem:

Suppose that the monomial ordering in $k[x_0 > x_1 > \ldots > x_n]$ is given by $(n \geq 3)$. $x_{i_0} > x_{i_1} > \ldots > x_{i_n}$ with the lexicographic ordering. Let $\mathcal{G}_n$ denote the set of all 2×2 minors of the matrix A, i.e., $\mathcal{G}_n = \{x_i x_{j+1} - x_{i+1} x_j \mid 0 \leq i < j \leq n\}$. Let I denote the ideal generated by $\mathcal{G}_n$ in $k[x_0, x_1, ..., x_n]$. The set $\mathcal{G}_n$ is a Gröebner basis with respect to the said monomial order if and only if
given monomial order satisfies the property P_{n-3}.
That is i_k is either $\min(S_k)$ or $\max(S_k)$ for $0 \leq k \leq n-3$
.

And for $n = 2$ G_n forms a Gröbner basis.

Remark: There is relaxation on properties P_{n-2}, P_{n-1} and P_n. The monomial order may or may not satisfy the Properties P_{n-2}, P_{n-1} and P_n.

Proof for Only If part:

The set $\mathcal{G}_n$; $n > 2$; is a Gröebner basis with respect to the said monomial order only if
given monomial order satisfies the property P_{n-3}.

Theorem 1. *Suppose that the monomial ordering in $k[x_0, x_1, \ldots, x_n]$ is given by $(n \geq 3)$. $x_{i_0} > x_{i_1} > \ldots > x_{i_n}$ with the lexicographic ordering. Let $\mathcal{G}_n$ denote the set of all 2×2 minors of the matrix A, i.e., $\mathcal{G}_n = \{x_i x_{j+1} - x_{i+1} x_j \mid 0 \leq i < j \leq n\}$. Let I denote the ideal generated by $\mathcal{G}_n$ in $k[x_0, x_1, ..., x_n]$. The set $\mathcal{G}_n$ a Groebner basis with respect to the said monomial order only if i_0 is either 0 or n.*

Proof. By method of contradiction.

Case I: n=3

$$A = \begin{bmatrix} x_0 & x_1 & x_2 \\ x_1 & x_2 & x_3 \end{bmatrix}.$$

$\mathcal{G}_3 = x_0 x_2 - x_1^2, x_0 x_3 - x_2 x_1, x_1 x_3 - x_2^2$
Assume i_0 is neither 0 or 3.
$\Rightarrow i_0 = 1$ or $i_0 = 2$

Subcase I] $i_0 = 1$ i.e. x_1 largest
consider the S polynomial

$S(x_0x_3 - x_2x_1, x_1x_3 - x_2^2) = x_2^3 - x_0x_3^2$
$x_2^3 - x_0x_3^2$ does not tend to 0
as LT of each polynomial in $\mathcal{G}_3$ contains x_1 which is not present in $x_2^3 - x_0x_3^2$
thus $\mathcal{G}_3$ does not form a Groebner Basis for $i_0 = 1$.

Subcase II] $i_0 = 2$ i.e. x_2 is largest.

Consider the S polynomial
$S(x_0x_3 - x_2x_1, x_0x_2 - x_1^2) = x_1^3 - x_3x_0^2$
$x_2^3 - x_0x_3^2$ does not tend to 0
as LT of each polynomial in $\mathcal{G}_3$ contains x_2 which is not present in $x_0^2x_3 - x_1^3$
thus $\mathcal{G}_3$ does not form a Groebner Basis for $i_0 = 2$.

Thus $\mathcal{G}_3$ does not form a Groebner Basis if i_0 is neither 0 nor 3 which is a contradiction for n = 3

Case II: n = 4
Assume i_0 is neither 0 nor 4
$\Rightarrow$ $i_0 = 1$ or $i_0 = 2$ or $i_0 = 3$
$\mathcal{G}_3 = \{x_0x_2 - x_1^2, x_0x_3 - x_1x_2, x_0x_4 - x_1x_3, x_1x_3 - x_2^2, x_1x_4 - x_2x_3, x_2x_4 - x_2x_4 - x_3^2\}$
Subcase I] $i_0 = 1$ i.e. x_1 is largest.

Consider the S polynomial
$S(x_1x_3 - x_2^2, x_0x_4 - x_1x_3) = x_0x_4 - x_2^2$
which does not tend to 0 as except for $x_2x_4 - x_3^2$ all other polynomials LT contains x_1 which is not present in $x_0x_4 - x_2^2$ and $x_2x_4 - x_3^2$ does not divide the S polynomial. $\Rightarrow \mathcal{G}_3$ does not form a Groebner Basis for $i_0 = 1$

Subcase II] $i_0 = 2$ i.e. x_2 is largest.
Consider the S polynomial $S(x_0x_3 - x_1x_2, x_1x_4 - x_2x_3) = x_0x_3^2 - x_1^2x_4$ which does not tend to 0 as except for $x_0x_4 - x_1x_3$, all other polynomial's LT containx_2 which is not present in $x_0x_4 - x_1x_3$.
Thus $\mathcal{G}_3$ does not form a Gröebner Basis for $i_0 = 2$.

Subcase III] $i_0 = 3$ i.e. x_3 is largest.
Consider the S polynomial $S(x_3x_1 - x_4x_0, x_3x_1 - x_2^2) = x_2^2 - x_4x_0$ which do not tend to 0 as except for $x_0x_2 - x_1^2$, all other polynomial's LT contain x_3 which is not present in $x_2^2 - x_4x_0$.
Thus $\mathcal{G}_3$ does not form a Gröebner Basis for $i_0 = 3$.

Thus $\mathcal{G}_3$ does not form a Gröebner Basis if i_0 is neither 0 nor 4 which is a contradiction.

Lemma 1. *Let $R = k[x_0, x_1, \ldots, x_n]$ be polynomial ring ($n \geq 3$). For the lexicographic ordering in indeterminants with property $x_{i_0} > x_{i_1} > \ldots > x_{i_n}$. Let G_n be generator set defined by the set of all 2×2 minors of the matrix A. Let I be the ideal generated by the generator G_n. Let $P = x_{i+1}x_{i-3} - x_{i-1}^2 \in R$ where $3 \leq i \leq n-1$.*
If $LT(x_i x_{i-2} - x_{i+1}x_{i-3}) = x_i x_{i-2}$ and $LT(x_i x_{i-2} - x_{i-1}^2) = x_i x_{i-2}$, then P is not divisible by G_n.

Proof. Possible divisors of P from G_n are $x_{i+1}x_{i-3} - x_i x_{i-2}$, $x_{i+1}x_{i-3} - x_{i+2}x_{i-4}$ (if $4 \leq i \leq n-2$) if $LT(P) = x_{i+1}x_{i-3}$ and $x_{i-1}^2 - x_i x_{i-2}$ if $LT(P) = x_{i-1}^2$.

Now there are two possibilities

I. $LT(P) = -x_{i-1}^2$;
For this case possible divisor is $x_{i-1}^2 - x_i x_{i-2}$. But we have $LT(x_{i-1}^2 - x_i x_{i-2}) = x_i x_{i-2}$. Thus the leading term of P is not divisible by the leading term of the divisor, hence P is not divisible.

II. $LT(P) = x_{i+1}x_{i-3}$;
Now there are two possible divisors $x_{i+1}x_{i-3} - x_i x_{i-2}$ and $x_{i+1}x_{i-3} - x_{i+2}x_{i-4}$ (if $4 \leq i \leq n-2$), thus two subcases depending upon which polynomial would divide the polynomial P first.

$x_{i+1}x_{i-3} - x_i x_{i-2}$ can not divide P first because $LT(x_i x_{i-2} - x_{i+1}x_{i-3}) = x_i x_{i-2}$ and $LT(P) = x_{i+1}x_{i-3}$.

If $x_{i+1}x_{i-3} - x_{i+2}x_{i-4}$ divides P first ($4 \leq i \leq n-2$);
This implies that $LT(x_{i+1}x_{i-3} - x_{i+2}x_{i-4}) = x_{i+1}x_{i-3}$. After division we get $P = x_{i+1}x_{i-3} - x_{i-1}^2 = 1 \times (x_{i+1}x_{i-3} - x_{i+2}x_{i-4}) + (x_{i+2}x_{i-4} - x_{i-1}^2)$.
Thus the remainder R_1 after first step of division is $R_1 = x_{i+2}x_{i-4} - x_{i-1}^2$.
Now again there are three possible divisors of R_1;

1. $x_{i-1}^2 - x_i x_{i-2}$ if $LT(R_1) = -x_{i-1}^2$,
2. $x_{i+2}x_{i-4} - x_{i+1}x_{i-3}$ if $LT(R_1) = x_{i+2}x_{i-4}$.
3. $x_{i+2}x_{i-4} - x_{i+3}x_{i-5} (5 \leq i \leq n-3)$ if $LT(R_1) = x_{i+2}x_{i-4}$.

We discard possibilities 1 and 2 because $LT(x_i x_{i-2} - x_{i-1}^2) = x_i x_{i-2}$ and $LT(x_{i+1}x_{i-3} - x_{i+2}x_{i-4}) = x_{i+1}x_{i-3}$.

After dividing R_1 by (3) we get;
$R_1 = x_{i+2}x_{i-4} - x_{i-1}^2 = 1 \times (x_{i+2}x_{i-4} - x_{i+3}x_{i-5}) + (x_{i+3}x_{i-5} - x_{i-1}^2)$
$\therefore R_2 = x_{i+3}x_{i-5} - x_{i-1}^2$.

We claim that, At general step m, we get $R_m = x_{i+m+1}x_{i-m-3} - x_{i-1}^2$ $(m+3 \leq i \leq n-m-1)$ with $LT(x_{i+m+1}x_{i-m-3} - x_{i+m}x_{i-m-2}) = -x_{i+m}x_{i-m-2}$

Claim is true for the case $m = 1$ ($\because R_1 = x_{i+2}x_{i-4} - x_{i-1}^2$ and $LT(x_{i+2}x_{i-4} - x_{i+1}x_{i-3}) = -x_{i+1}x_{i-3}$).

Suppose the claim is true for the case $m = p$.
$\therefore R_p = x_{i+p+1}x_{i-p-3} - x_{i-1}^2$.
Now possible diviors are $x_{i-1}^2 - x_i x_{i-2}$ if $LT(R_p) = x_{i-1}^2$ and $x_{i+p+1}x_{i-p-3} - x_{i+p}x_{i-p-2}$, $x_{i+p+1}x_{i-p-3} - x_{i+p+2}x_{i-p-4}$ $(p+4 \leq i \leq n-p-2)$ if $LT(R_p) = x_{i+p+1}x_{i-p-3}$.

We discard the possibilities 1 and 2 because $LT(x_{i-1}^2 - x_i x_{i-2}) = -x_i x_{i-2}$ and $LT(x_{i+p+1}x_{i-p-3} - x_{i+p}x_{i-p-2}) = -x_{i+p}x_{i-p-2}$.

So, after dividing $R_p = x_{i+p+1}x_{i-p-3} - x_{i-1}^2$ by $x_{i+p+1}x_{i-p-3} - x_{i+p+2}x_{i-p-4}$ (assuming $LT(x_{i+p+1}x_{i-p-3} - x_{i+p+2}x_{i-p-4}) = x_{i+p+1}x_{i-p-3}$), we get remainder as $x_{i+p+2}x_{i-p-4} - x_{i-1}^2$, which can be rewritten as $R_{p+1} = x_{i+(p+1)+1}x_{i-(p+1)-3} - x_{i-1}^2$.
Here we assumed $LT(x_{i+p+2}x_{i-p-4} - x_{i+p+1}x_{i-p-3}) = LT(x_{i+(p+1)+1}x_{i-(p+1)-3} - x_{i+(p+1)}x_{i-(p+1)-2} = -x_{i+(p+1)}x_{i-(p+1)-2})$

Thus the claim is true for $m = p + 1$ Therefor by principle of mathematical induction we can say that the claim is true.

So after some repeatations the process will terminate at such m where either $i + m + 1 = n$ or $i - p - 3 = 0$. Then such R_m will look like $x_0 x_q - x_{i-1}^2$ or $x_q x_n - x_{i-1}^2$ which is not further divisible by any of the polynomial from G_n ($\because$ above claim).

This proves our lemma. $\square$

Case III: n $\geq$ 5
Assume i_0 is neither 0 nor n
Let i_0 = i i.e. x_i is largest. such that $0 < i < n$
$\Rightarrow$ either $i - 3 \geq 0$ or $i + 3 \leq n$
for if $i - 3 < 0$

$\Rightarrow i+3<6$
$\Rightarrow i+3 \leq 5 \leq n \ldots$ as i is an integer.

Subcase I] $i-3 \geq 0$
Consider the S polynomial
$S(x_i x_{i-2} - x_{i+1}x_{i-3}, x_i x_{i-2} - x_{i-1}^2) = x_{i-1}^2 - x_{i+1}x_{i-3}$
Only possible divisors are $x_{i-1}^2 - x_i x_{i-2}, x_{i+1}x_{i-3} - x_i x_{i-2}$ and $x_{i+1}x_{i-3} - x_{i+2}x_{i-4}$.
In this $x_{i-1}^2 - x_i x_{i-2}$ and $x_{i+1}x_{i-3} - x_i x_{i-2}$ will not divide the S-Polynomial as the leading term of S-Polynomial is not divisible by the leading terms of 2×2 minor.
Whereas $x_{i+1}x_{i-3} - x_{i+2}x_{i-4}$ gives nonzero remainder after division from lemma 1.
Thus S-Polynomial does not tend to 0 on division by $\mathcal{G}$. Thus $\mathcal{G}$ does not form a Gröebner Basis.

Subcase II] $i+3 \leq n$
Consider the S polynomial
$S(x_i x_{i-2} - x_{i+1}x_{i-3}, x_i x_{i-2} - x_{i-1}^2) = x_{i-1}^2 - x_{i+1}x_{i-3}$
By same reasons as above S-Polynomial does not tend to 0 on division by $\mathcal{G}$.
Thus $\mathcal{G}$ does not form a Gröoebner Basis.
Thus $\mathcal{G}$ does not form a gröoebner basis for $n \geq 5$.
Thus $\mathcal{G}$ does not form a gröebner basis for any $n \geq 3$,
if i_0 is neither 0 nor n.
Hence the contradiction.
$\Rightarrow i_0 = 0$ or $i_0 = n$.

□

Lemma 2. *If monomial ordering is $x_{i_0} > \ldots > x_{i_n}$, with property P_{j-1}; $1 \leq j \leq n$ and if $min(S_j) \leq m \leq max_(S_j)$; then $m \in S_j$*
That is all the integers in between $min(S_j)$ and $max(S_j)$ are contained in S_j, i.e. S_j is of the form {i, i+1, ... i+k}.

Proof. Assume $m \notin S_j$
then $x_m > x_l \quad \forall\ l \in S_j \ldots$ if $x_m < x_l$ for some $l \in S_j$ then $m \in S_j$ by definition.
$\Rightarrow \quad m = i_p$ for some $p < j \ldots$ as $i_p \in S_j$ for $p \geq j$
$\Rightarrow \quad m = min(S_p)$ or $m = max(S_p)$
From definition of the set S_k we know that $S_j \subset S_p$ but $m \geq min(S_j) \in S_j \subset S_p$
thus m can't be $min(S_p)$
Similarly $m \leq max(S_j) \in S_j \subset S_p$

$\therefore m \neq max(S_p)$
Which is a contradiction.

$\square$

Theorem 2. *Suppose that the monomial ordering in* $k[x_0 > x_1 > \ldots > x_n]$ *is given by* $(n \geq 3)$. $x_{i_0} > x_{i_1} > \ldots > x_{i_n}$ *with the lexicographic ordering. Let* $\mathcal{G}_n$ *denote the set of all* 2×2 *minors of the matrix A, i.e.,* $\mathcal{G}_n = \{x_i x_{j+1} - x_{i+1} x_j \mid 0 \leq i < j \leq n\}$. *Let* I *denote the ideal generated by* $\mathcal{G}_n$ *in* $k[x_0, x_1, ..., x_n]$. *The set* $\mathcal{G}_n$ *a Gröebner basis with respect to the said monomial order only if*
i_k *is either min(*S_k*) or max(*S_k*)*
for $0 \leq k \leq n-3$
that is given monomial order satisfies the property P_{n-3}

Remark: Monomial ordering need not satisfy the property P_{n-2}, P_{n-1} or P_n

Proof.
Using the method of induction on subscript number of the property P_k
True for $k = 0$ from Theorem I
Consider true for $k = j-1; 1 \leq j \leq n-3$. i.e property P_{j-1} is satisfied and we have to show that property P_j is also satisfied by the monomial ordering.
Assume not true for $k = j \leq n-3$
$\therefore\ min(S_j) < i_j < max(S_j)$
Now, $j \leq n-3 \Rightarrow$ cardinality of S_j is at least 4

Case I: $|S_j| = 4$
From induction hypothesis, property P_{j-1} is satisfied and from lemma 2 we can say that S_j is of the form $\{i, i+1, i+2, i+3\}$.
Because $min(S_j) < i_j < max(S_j)$there are only two possibilities of S_j as follows.
$S_j = \{i_j - 1, i_j, i_j + 1, i_j + 2\}$ or $S_j = \{i_j - 2, i_{j-1}, i_j, i_j + 1\}$

Now, consider $S_j = \{i_j - 1, i_j, i_j + 1, i_j + 2\}$ then consider
$S(x_{i_j-1} x_{i_j+2} - x_{i_j} x_{i_j+1}, x_{i_j} x_{i_j+2} - x_{i_j+1}^2) = x_{i_j-1} x_{i_j+2}^2 - x_{i_j+1}^3$

if $LT(x_{i_j-1} x_{i_j+2}^2 - x_{i_j+1}^3) = x_{i_j+1}^3$
then $S \nrightarrow 0$ as only divisor to $x_{i_j+1}^3$ is $x_{i_j+1}^2 - x_{i_j+2} x_{i_j}$ who's leading term is $x_{i_j+2} x_{i_j}$

if $LT(x_{i_j-1}x_{i_j+2}^2 - x_{i_j+1}^3) = x_{i_j-1}x_{i_j+2}^2$
Then only possible divisors are $x_{i_j+2}^2 - x_{i_j+1}x_{i_j+3}, x_{i_j-1}x_{i_j+2} - x_{i_j}x_{i_j+1}$ and $x_{i_j-1}x_{i_j+2} - x_{i_j-2}x_{i_j+3}$(if exists)
In the case of $x_{i_j-1}x_{i_j+2} - x_{i_j}x_{i_j+1}$, $LT(x_{i_j-1}x_{i_j+2} - x_{i_j}x_{i_j+1}) = -x_{i_j}x_{i_{j+1}}$ which does not divide $x_{i_j-1}x_{i_j+2}$.
In the case of $x_{i_j-1}x_{i_j+2} - x_{i_j-2}x_{i_j+3}$;
$LT(x_{i_j-1}x_{i_j+2} - x_{i_j-2}x_{i_j+3} = x_{i_j-2}x_{i_j+3}$ as $x_{i_j+3} > x_{i_j-1}, x_{i_j+2}$
and in the case of $x_{i_j+2}^2 - x_{i_j+1}x_{i_j+3}$; $LT(x_{i_j+2}^2 - x_{i_j+1}x_{i_j+3}) = x_{i_j+1}x_{i_j+3}$

Similar arguments goes for $S_j = \{i_j - 2, i_j - 1, i_j, i_j + 1\}$
Thus for cardinality of $S_j = 4$, G_n does not form a Gröbner Basis. Hence contradiction.

case II: $|S_j| = 5$
From assumption and lemma 2 only possible cases are,
$S(j) = i_j - 1, i_j, i_j + 1, i_j + 2, i_j + 3$
$S(j) = i_j - 2, i_j - 1, i_j, i_j + 1, i_j + 2$
$S(j) = i_j - 3, i_j - 2, i_j - 1, i_j, i_j + 1$

Consider following examples in each cases respectively.
$S(f_{i_j-1,i_j+2}, f_{i_j,i_j+1}) = x_{i_j-1}x_{i_{j+3}} - x_{i_j+1}^2$
$S(f_{i_j-2,i_j}, f_{i_j-1,i_j+1}) = x_{i_j-2}x_{i_j+1}^2 - x_{i_j-1}^2$
$S(f_{i_j-1,i_j+2}, f_{i_j,i_j+1}) = x_{i_j-1}x_{i_j+3} - x_{i_j+1}^2$
are counter examples to each case respectively. Arguments for example 1 and 3 are similar as in case I.
For case 2
If LT is $x_{i_j-1}^2$, then LT of only possible divisor i.e. $LT(x_{i_j-1}^2 - x_{i_j}x_{i_j-2}) = x_{i_j}x_{i_j-2}$ for the reason $x_{i_j} > x_{i_j-2}$. Thus does not divide.
And is LT is $x_{i_j-2}x_{i_j+1}^2$; then from lemma 1 we can say that G_n does not divide.
Thus for cardinality of $S_j = 5$, G_n does not form a Gröbner Basis. Hence contradiction.

Case III: $|S_j| \geq 6$
From assumption and lemma 1.1 we can say that
either $\{i_j - 1, i_j, i_j + 1, i_j + 2, i_j + 3\} \in S_j$
or $\{i_j - 3, i_j - 2, i_j - 1, i_j, i_j + 1\} \in S_j$

Consider the case where,
$\{i_j - 3, i_j - 2, i_j - 1, i_j, i_j + 1\} \in S_j$

Consider,
$S(f_{i_j,i_j-3}, f_{i_j-2,i_j-1}) = x_{i_j-1}^2 - x_{i_j+1}x_{i_j-3}$
if $LT(S) = x_{i-1}^2$ then only possible divisor is $x_{i_j-1}^2 - x_{i_j}x_{i_j-2}$ who's leading term is $x_{i_j}x_{i_j-2}$

if $LT(S) = x_{i_j+1}x_{i_j-3}$ then possible divisors are $x_{i_j+1}x_{i_j-3} - x_{i_j}x_{i_j-2}$ and $x_{i_j+1}x_{i_j-3} - x_{i_j+2}x_{i_j-4}$
The first one is not possible as leading term is $x_{i_j}x_{i_j-2}$. For second possibility; from lemma 1 we can say that remainder after division by G_n is non zero.

Similar arguments will go for other possibility of the set S_j.

Thus for cardinality of $S_j \geq 6$, G_n does not form a Gröbner Basis. Hence contradiction.

So, the assumption we made was wrong
$\therefore$ the set "G_n" forms a Gröbner basis only if monomial order satisfies the property P_{n-3}. i.e.i_j is either $max(S_j)$ or $min(S_j); \forall i_j \leq n-3$. $\square$

Proof for If part:

Definition 3. Mapping ϕ Suppose that the monomial ordering in $R_{n+1} = k[x_0, x_1, \ldots, x_{n+1}]$ is given by $x_{i_0} > x_{i_1} > \ldots > x_{i_{n+1}}$. Consider set $A_{n+1} = \{x_0, x_1, \ldots, x_{n+1}\}$ of all indeterminants in R_{n+1} and the set $A_n = \{x_0, x_1, \ldots, x_n\}$ of all indeterminants in R_n.
Let $x_{i_a} \in A_{n+1}$ then mapping ϕ is defined as,
$\phi : A_{n+1}/\{x_{i_a}\} \to A_n$
$\phi(x_i) = x_i \quad$ if $i < i_a$
$\phi(x_i) = x_{i-1} \quad$ if $i > i_a$
It is easy to show that this is a one-to-one and onto map.
Moreover, we extend the definition to all polynomials not containing i_a as,
$\phi(Ax^\alpha + Bx^\beta) = A\phi(x^\alpha) + B\phi(x^\beta) \quad$ where $\alpha_{i_a} = \beta_{i_a} = 0$
$\phi(Af + Bg) = A\phi(f) + B\phi(g) \quad f, g \in k[x_0, \ldots, x_n]$
$\phi(x_0^{\alpha_0}x_1^{\alpha_1}\ldots x_{n+1}^{\alpha_{n+1}}) = \phi(x_0)^{\alpha_0}\phi(x_1)^{\alpha_1}\ldots\phi(x_{n+1})^{\alpha_{n+1}} \quad$ where $\alpha_{i_a} = 0$
This is also a one to one and onto map.

Example: For monomial ordering $x_0 > x_5 > x_1 > x_3 > x_4 > x_2$ in $R = k[x_0, x_1, \ldots, x_5]$, mapping ϕ from $A_5/\{x_1\} = \{x_0, x_2, x_3, x_4, x_5\} \to A_4 = \{x_0, x_1, x_2, x_3, x_4\}$ is given by
$\phi = \{(x_0, x_0), (x_5, x_4), (x_3, x_2), (x_4, x_3), (x_2, x_1)\}$

Definition 4. Mapping ϕ on the order Suppose that the monomial ordering $>_{n+1}$ in $R_{n+1} = k[x_0, x_1, \ldots, x_{n+1}]$ is given by $x_{i_0} > x_{i_1} > \ldots > x_{i_{n+1}}$. Then the monomial ordering $\phi(>_{n+1})$ in $R_n = k[x_0, x_1, \ldots, x_n]$ is defined by $\phi(x_{i_0}) > \phi(x_{i_1}) > \ldots > \phi(x_{i_{n+1}})$ where nothing maps at the pace of i_a.

Example: For monomial ordering $>_5 = x_0 > x_5 > x_1 > x_3 > x_4 > x_2$ in $R = k[x_0, x_1, \ldots, x_5]$, mapping ϕ for invarient indeterminant x_1 maps $>_5$ to $>_4 = x_0 > x_4 > x_2 > x_3 > x_1$

Lemma 3. *Let $f = x_i x_{j+1} - x_{i+1} x_j$ be 2×2 minor of from G_{n+1} such that f does not contain x_{i_a}, then $\phi(f)$ is also a 2×2 minor from G_n.*

Proof. Consider $x_i x_{j+1} - x_j x_{i+1}$ which is a 2×2 minor and neither of i, i+1, j, j+1 is i_a. It is enough to show that if x_i maps to $x_{i'}$ then x_{i+1} maps to $x_{i'+1}$.

Case 1 $i < i_a$
$\Rightarrow i + 1 < i_a$
Thus i maps to i and i+1 maps to i+1

Case 2 $i > i_a$
$\Rightarrow i + 1 > i_a$
Thus i maps to i-1 and i+1 maps to i

Thus if x_i maps to $x_{i'}$ then x_{i+1} maps to $x_{i'} + 1$, which proves that polynomials are nothing but 2×2 minor of $2 \times n$ Matrix. □

Lemma 4. *Suppose that $<_1$ and $<_2$ denote the monomial orders of $k[x_0, x_1, \ldots, x_{n+1}]$ and $k[x_0, x_1, \ldots, x_n]$ respectively, such that $\phi(<_1) = <_2$. If $0 \neq f \in k[x_0, \ldots, x_{n+1}]$ and x_{i_a} does not occur in f then,*
$\phi(LT_{<_1}(f)) = LT_{<_2}(\phi(f))$

Proof. Let $<_1 = (x_{i_0} > x_{i_1} > \ldots > x_{i_{n+1}})$.
Let $x = x_{i_0} x_{i_1} \ldots x_{i_{n+1}}$.
Let $\alpha = (\alpha_0 \alpha_1 \ldots \alpha_{n+1})$ such that $\alpha_a = 0$. Let $LT_{<_1}(f) = x^\alpha$.

Let x^{α_1} be any arbitrary term in f other than x^α
Let i^{th} entry of $\alpha - \alpha_1$ be non zero.
As $x^\alpha = LT_{<_1}(f)$, $\alpha(i) - \alpha_1(i) > 0$.
We have $<_2 = (\phi(x_{i_0}) > \phi(x_{i_1}) > \ldots > \phi(x_{i_{n+1}}))$
and $\phi(x) = \phi(x_{i_0})\phi(x_{i_1})\ldots\phi(x_{i_{n+1}})$.
Then from definition 3 we have, $\phi(x^\alpha) = \phi(x)^\alpha$ which is a term of $\phi(f)$. Similarly $\phi(x^{\alpha_1}) = \phi(x)^{\alpha_1}$ is also a term of $\phi(f)$.
Now, as first non-negative entry of $\alpha - \alpha_1$ is positive and as α hence α_1 is arbitrary $\phi(x^\alpha)$ is leading term of $\phi(f)$ with respect to monomial order $<_2$.
Hence, $\phi(LT_{<_1}(f)) = LT_{<_2}(\phi(f))$ is proved.

□

Theorem 3. *Suppose that the monomial ordering in* $k[x_0, x_1, \ldots, x_n]$ *is given by* $(n \geq 3)$. $x_{i_0} > x_{i_1} > \ldots > x_{i_n}$ *with the lexicographic ordering. Let* $\mathcal{G}_n$ *denote the set of all* 2×2 *minors of the matrix* A*, i.e.,* $\mathcal{G}_n = \{x_i x_{j+1} - x_{i+1}x_j \mid 0 \leq i < j \leq n\}$. *Let* I *denote the ideal generated by* $\mathcal{G}_n$ *in* $k[x_0, x_1, ..., x_n]$. *The set* $\mathcal{G}_n$ *a Gröbner basis with respect to the said monomial order if given monomial order satisfy the property* P_{n-3}

Proof. The proof will follow the method of induction over the number of variables "N".
The statement is trivial in the case $N = 2$ but we will go one step ahead and show that using a computer program (appended below) that the statement is also true for $N = 2, \ldots, 7$.
Lets assume the statement is True for $N = n$, then we have to show that the statement is also True for $N = n + 1$.

Now, consider the S-Polynomial of 2×2 Minors $f = x_i x_{j+1} - x_j x_{i+1}$ and $g = x_l x_{m+1} - x_m x_{l+1}$ as $S(f, g)$
As $n > 7$, there exists a x_{i_a} such that x_{i_a} does not occur in any one of the 4 monomials appearing in f and g, for the reason that there can be as the most 8 distinct variables that may occur in 4 monomials.

Now consider x_{i_a} is not present in given pair of S-polynomial where monomial ordering is $<_1 = x_{i_0} > \ldots > x_{i_{a-1}} > x_{i_a} > x_{i_{a+1}} > \ldots > x_{i_{n+1}}$.
Here we can apply mapping ϕ as defined in the definition 3.
Lemma 2 tells that $\phi(f)$ and $\phi(g)$ are both 2×2 minors from set G_n.
As $\phi(f)$ and $\phi(g)$ are both 2×2 minors from set G_n, using Induction Hypothesis we can say that the S-Polynomial of $\phi(f)$ and $\phi(g)$ is divisible by G_n. More precisely

$S(\phi(f), \phi(g)) = \sum_i a_{i,j',k'} f_{j'} g_{k'} \quad a_{i,j',k'} \in k[x_0, \ldots, x_n].$

Division algorithm tells that
$multideg(a_{i'j'k'}) \leq multideg(S(\phi(f), \phi(g)))$

As ϕ being a one to one and onto map from $A_{n+1}/\{i_a\}$ to A_n. We can define ϕ^{-1}.
Applying ϕ^{-1} to above equation, we get
$\phi^{-1}(S(\phi(f), \phi(g))) = \phi^{-1}(\sum_i a_{i,j',k'} f_{j'} g_{k'})$

But $\phi^{-1}(S(\phi(f), \phi(g)))$ is nothing but $S(f, g)$
and $\phi^{-1}(\sum_i a_{i,j',k'} f_{j'} g_{k'})$ is nothing but $\sum_i a'_{i,j,k} f_j g_k$ where f_j and g_k are both 2×2 minors in R_{n+1}

Applying lemma 3 to above in-equation we will get
$multideg(a'_{i,j,k}) \leq multideg(S(f, g))$
This shows that $S(f, g)$ is divisible by G_{n+1}.
Hence G_{n+1} is also a Gröbner basis.
This completes the proof

$\square$

Result

Using Theorem 2 and Theorem 3 we can state the following.

Suppose that the monomial ordering in $k[x_0 > x_1 > \ldots > x_n]$ is given by $(n \geq 3)$. $x_{i_0} > x_{i_1} > \ldots > x_{i_n}$ with the lexicographic ordering. Let $\mathcal{G}_n$ denote the set of all 2×2 minors of the matrix A, i.e., $\mathcal{G}_n = \{x_i x_{j+1} - x_{i+1} x_j \mid 0 \leq i < j \leq n\}$. Let I denote the ideal generated by $\mathcal{G}_n$ in $k[x_0, x_1, ..., x_n]$. The set $\mathcal{G}_n$ a Gröebner basis with respect to the said monomial order if and only if
i_k is either $\min(S_k)$ or $\max(S_k)$
for $0 \geq k \geq n - 3$
that is given monomial order satisfies the property P_{n-3}

Examples

Let $R = k[x_0, x_1, \ldots, x_5]$ be the polynomial ring over field k. Let $G = \{x_i x_{j+1} - x_{i+1} x_j \mid 0 \leq i < j \leq 5\}$. Let I denote the ideal generated by G. Then following are the some examples of lexicographic ordering where G forms a Grobner basis for ideal I.

1. For monomial ordering $x_0 > x_1 > x_2 > x_3 > x_4 > x_5$ in $R = k[x_0, x_1, \ldots, x_5]$; generator $\mathcal{G}_5$ **is** a Gröebner basis.
2. For monomial ordering $x_0 > x_5 > x_1 > x_3 > x_4 > x_2$ in $R = k[x_0, x_1, \ldots, x_5]$; generator $\mathcal{G}_5$ **is** a Gröebner basis.
3. For monomial ordering $x_5 > x_2 > x_1 > x_0 > x_4 > x_3$ in $R = k[x_0, x_1, \ldots, x_5]$; generator $\mathcal{G}_5$ **does not form** a Gröebner basis.
4. For monomial ordering $x_0 > x_5 > x_3 > x_1 > x_2 > x_4$ in $R = k[x_0, x_1, \ldots, x_5]$; generator $\mathcal{G}_5$ **does not form** a Gröebner basis.

Appendix:

A

Following is the pseudocode of our program.

```
########################## GB OF RNC #########################

def Minor_Gen(n):              #Gives Generator Set G
    minor = []
    if n==2:
        return [(x_0x_2 - x_1^2)]
    else:
        i = 1
        while i < 1:
            minor.append([x_(i-1)x_n - x_(n-1)x_i])
            i = i+1
    minor = minor + Minor_Gen(n-1)
```

```
    return minor


def S_Poly_Gen(n):             #Gives all S_Polynomials of Generator G
    minor = Minor_Gen(n)
    S_List = []
    if n==2:
        return S_Poly(minor[1], minor[2])
    else:
        i = 1
        while i < n:
            S_List.append(S_Poly(minor[i], minor[n]))
            i = i+1
    S_List = S_List + S_Poly_Gen(n-1)
    return S_List


def Check(n):                  #Checks if Generator is Groebner basis
    minor = Minor_Gen(n)       #Uses Buchberger's Criterion
    S_list = S_Poly_Gen(n)
    for poly in S_List:
        if minor divides poly:
            pass
        else:
            return False
    return True
```

B

Following is the Python code we used to check our result.

Use "isgrobner(n, order)", for a specific permutation order, where order is of the form $[i_0, i_1, \ldots, i_n]$ for the monomial order $x_0 > x_1 > \ldots > x_n$
Program consists of two parts. First, where we define a polynomial to Python and its operations like addition, multiplication, division. This part also gives LCM and S-polynomial of given polynomials.
Second part deals with only Rational Normal Curve. It gives the set G_n, i.e. set of all 2×2 minors of A. Calculates its all s-polynomials.
For incorporating given order to polynomials we just changed the positions of monomials with respect to given order. Eg. $x_0^2 x_1 x_3^7$ with monomial ordering $x_1 > x_3 > x_2 > x_0$, is same as $x_0 x_{1^7} x_3^2$ with monomial ordering $x_0 > x_1 >$

$x_2 > x_3$.
We used Buchberger's criterion to determine if given generator with given monomial order is a Grobner basis or not.

```
################GrobnerBasisOfRationalNormalCurve #####################
from numpy import *
from copy import deepcopy

"""Following is the program to check if a given generator
is a Grobener basis or not for given monomial order."""


def array_to_object(arrayin):
    obj=[]
    for i in list(arrayin):
        obj.append(tuple(i))
    return obj


class Poly(object):                                  #constructing multivariable
    """takes list of tuples
        each list represent a monomial with first entry as coeficient
        5xy-2x^2 = Poly([(5,1,1),(-2,2,0)])"""
    def __init__(self, poly):
        self.vari=deepcopy(len(poly[0])-1)
        zeropoly=[0]
        zeroterm=(0,)*(self.vari+1)
        zeropoly[0]=zeroterm
        self.zero=zeropoly
        dtype=[("0", float)]
        for i in range(self.vari):
            x=("%d"%(i+1), int)
            dtype.append(x)
        if len(poly)==0:
            self.poly=[]
            return None
        #self.poly=poly
        d={}
        for i in poly:
            if i[1:] not in d.keys():
                if float(i[0]) != 0.0:
```

```
                d[i[1:]]=i
        else:
            if d[i[1:]][0]+i[0]==0.0:
                del d[i[1:]]
            else:
                dummylist=list(d[i[1:]])
                dummylist[0]=dummylist[0]+i[0]
                d[i[1:]]=tuple(dummylist)
    unsortpoly=d.values()
    unsortarray=array(unsortpoly, dtype=dtype)
    if len(unsortpoly)==0:
        self.poly=self.zero
        return None
    order=[]
    for i in range(self.vari):
        x="%d"%(i+1)
        order.append(x)
    revsort=sort(unsortarray, order=order)
    reqlist=list(revsort)
    reqlist.reverse()
    self.poly=array_to_object(reqlist)

def __eq__(self, other):
    return self.poly == other.poly

def __ne__(self, other):
    return self.poly != other.poly

def LT(self):                     #note o/p is tuple
    return self.poly[0]

def leadingterm(self):                          #gives leading term
    leadingterm=[0]
    leadingterm[0]=self.LT()
    return Poly(leadingterm)

def multideg(self):
    return self.LT()[1:]

def isdivisible(self, other):
"""checks if the leading term of the polynomial is divisible by
```

```
the leading term of the other polynomial."""
    for i in range(len(self.multideg())):
        if self.multideg()[i]<other.multideg()[i]:
            return False
    return True

def __add__(self, other):    #adds
    added = self.poly+other.poly
    return Poly(added)

def __sub__(self, other):    #subtracts
    neglist=[]
    other1=other.poly
    for i in other1:
        dummytuple=(-1*i[0],)+i[1:]
        neglist.append(dummytuple)
    return self.__add__(Poly(neglist))

def __mul__(self, other):     #multiplies
    if type(other)==int or type(other)==float:
        mullist=[]
        self1=self.poly
        for i in self1:
            dummytuple=(float(other)*i[0],)+i[1:]
            mullist.append(dummytuple)
        return Poly(mullist)
    mullist=[]
    self1=self.poly
    other1=other.poly
    for i in self1:
        for j in other1:
            mulnum=i[0]*j[0]
            muldeg=tuple(array(i[1:])+array(j[1:]))
            multerm=(mulnum,)+muldeg
            mullist.append(multerm)
    return Poly(mullist)

def monodiv(self, other):
"""divides the polynomial by the leading term of the other"""
    if type(other)==type(Poly([(0,0,0)])):
        if len(other.poly)==1 and len(self.poly)==1:
```

```
            T = self.isdivisible(other)
            if T:
                a, b = self.poly[0], other.poly[0]
                anscoef=float(a[0])/float(b[0])
                ansdeg=array(a[1:])-array(b[1:])
                return Poly([(anscoef,)+tuple(ansdeg)])

def __div__(self, other):
"""divides: gives quotient and remainder"""
    if type(other)==int or type(other)==float:
        return self.__mul__(1.0/other)
    s=len(other)   #no. of polynomials
    quotient=[Poly(self.zero)]*s
    remainder=Poly(self.zero)
    dummyself=Poly(self.poly)
    while dummyself.poly != self.zero:
        i=0
        divisionoccurred = False
        while i<s and divisionoccurred == False:
            T = dummyself.isdivisible(other[i])
            if T:
                quotient[i]=quotient[i]+
                (dummyself.leadingterm().
                monodiv(other[i].leadingterm()))
                dummyself=dummyself-(dummyself.leadingterm().
                monodiv(other[i].leadingterm()))*other[i]
                divisionoccurred=True
            else:
                i=i+1
        if divisionoccurred==False:
            remainder=remainder+dummyself.leadingterm()
            dummyself=dummyself-dummyself.leadingterm()
    return [quotient, remainder]

def LCM(self, other):  #gives LCM
    lead_self=self.LT()
    lead_other=other.LT()
    LCM_term=[1]
    for i in range(len(lead_self)-1):
        LCM_term.append(max(lead_self[i+1], lead_other[i+1]))
    LCM=[tuple(LCM_term)]
```

```
        return Poly(LCM)

    def s_poly(self, other):   #gives S_Polynomial
        LCM = self.LCM(other)
        first = LCM.monodiv(self.leadingterm())
        second = LCM.monodiv(other.leadingterm())
        return first*self-second*other

    def iszero(self):   #checks if the polynomial is zero polynomial
        return self.poly==self.zero




"""GB of RNC starts"""
#n is no. of variables-1; varables are x0, x1,...,x_n

def encode_order(term, order=0):     #term is only tuple
"""order is induced in the polynomial:
  eg: x^3*y^4*z with y>z>x    is same as x^4*y*z^3       with x>y>z  """
    if order==0:
        return term
    term_list=list(term)
    for i in range(len(order)):
        term_list[i+1]=term[order[i]+1]
    return tuple(term_list)

def minor_helper(n, m, order=0): #gives 2minor object
""" helping function for generating all 2 X 2 minors of the matrix
     A = | x_0    x_1    ...   x_(n-1) |
             | x_1    x_2   ....   x_n        |

             """



    if n<=1:
        return "Not Possible"
    if n == 2:
        term_1 = [0]*(m+2)
        term_2 = [0]*(m+2)
        term_1[0], term_1[1], term_1[3] = 1, 1, 1
```

```
        term_2[0], term_2[2] = -1, 2
        poly = [encode_order(tuple(term_1), order),
         encode_order(tuple(term_2), order)]
        return [Poly(poly)]
    minorr=minor_helper(n-1, m, order)
    for i in range(n-2):
        term_1 = [0]*(m+2)
        term_2 = [0]*(m+2)
        term_1[0], term_1[i+1], term_1[n+1] = 1, 1, 1
        term_2[0], term_2[i+2], term_2[n] = -1, 1, 1
        poly = [encode_order(tuple(term_1), order),
        encode_order(tuple(term_2), order)]
        minorr.append(Poly(poly))
    final_poly_1 = [0]*(m+2)
    final_poly_2 = [0]*(m+2)
    final_poly_1[0], final_poly_1[n-1], final_poly_1[n+1] = 1, 1, 1
    final_poly_2[0], final_poly_2[n] = -1, 2
    final_poly = [encode_order(tuple(final_poly_1), order),
    encode_order(tuple(final_poly_2), order)]
    minorr.append(Poly(final_poly))
    return minorr

def minor(n, order=0):
""" Generates all 2 X 2 minors of the matrix
     A = | x_0    x_1    ...   x_(n-1) |
             | x_1    x_2   ....   x_n        |

             """

    return minor_helper(n, n, order)

def all_s_poly(n, order=0):
""" Generates all possible S-Polynomials of 2 X 2 minors generated from matrix
     A = | x_0    x_1    ...   x_(n-1) |
             | x_1    x_2   ....   x_n        |

             """

    s_poly_list=[]
    minorr=minor(n, order)
    i=0
```

```
    while i<n*(n-1)/2-1:
        j=i+1
        while j<n*(n-1)/2:
            s_poly_list.append(minorr[i].s_poly(minorr[j]))
            j=j+1
        i=i+1
    return s_poly_list

def isgrobner(n, order=0):
"""checks if each of the S-polynomial is divisible by G
    return all(map(lambda x: (x/minor(n, order))[1].iszero(),
    all_s_poly(n, order)))

############################################################################
```