Topics in Algebraic Computation

Table of contents

1 Introduction

1.1 Overview

Course primarily concentrates on algorithmic approach to compute algebraic objects. Focus of the course will be on Algebraic objects that are built using Integers, Polynomials and Matrices. Algorithms for computing the following will be discussed as part of the course:

i. Addition, Multiplication, Inversion, GCD for Integers.

ii. Strassen's algorithm for Matrix Multiplication.

iii. Solving systems of Linear Equations.

iv. Computing Characteristic polynomial, Minimal Polynomail, Inverse Polynomial.

v. Conversion of Matrices into standard canonical forms like Hermite Canonical Form (HCF).

vi. Berlekamp's Algorithm for Polynomial factorization

Asymptotic computational complexity analysis and correctness proofs for these algorithms will also be covered. Beyond this, some more advanced topics will be discussed :

i. Lower bounds on complexity of Algebraic operations.

ii. Grobner Basis algorithm.

iii. Lattice reduction algorithm via L^3.

iv. Function Field Sieve, Number Field Sieve algorithm for factoring integers and finding discrete log.

1.2 References

Except for few topics course most content of the course is taken from Research publications. For standard topics, following text books are suggested:

i. Aho, Hopcroft, Ullman : Design and Analysis of Algorithms.

ii. Grobner Basis from Ideals, Varieties and Algorithms by Cox, Little, O'shea

iii. Lattice Reduction from Algorithmic aspects of Algebraic Number Theory by Cohen.

iv. Computational Algebra. by Abijit Das

1.3 Evaluation

Course has 3 evaluation components. Components and their weightage in the evaluation are as follows:

i. Mid Semester Examination - 30%

ii. Termpaper Assignment - 20%

iii. End Semester Examination - 50%

2 Hermite Canonical Form

We shall consider matrices with elements from a Field. Today's discussion will focus on key results and algorithms for conversion of matrices in to Hermite Normal Form (HNF), computing inverse of a square matrix, testing the consistency of linear equations and LUB decomposition of matrices. These forms of matrix help in computing matrix parameters like rank, basis of row and column, basis of null space.

We shall state the following result without proof. Subsequently we progress towards the construction.

Theorem 1. (Rank Factorization Theorem) *If A is an $m \times n$ matrix with rank r. Then there exist two full-rank matrices B and C with orders $m \times r$ and $r \times n$ such that $A = B \times C$.*

Notation 2. *Linear space spanned by columns of a matrix X is denoted by* $\mathrm{Col.Sp}(X)$ *and Linear space spanned by rows of a matrix X is denoted by* $\mathrm{Row.Sp}(X)$.

Notation 3. (Augumented Matrix) *If A, B are two matrices of orders $m \times k$ and $n \times k$, then the matrix $[A{:}\,B]$ of order $(m+n) \times k$ obtained by extending rows of A with corresponding rows of B.*

Note 4. System of linear equations : $Ax = b$ is consistent if $b \in \mathrm{Col.Sp}(A)$. Putting this in another way $\mathrm{Rank}(A) = (\mathrm{Rank}([A{:}\,b]))$.

Definition 5. (Null Space) *If A is an $n \times n$ matrix in $\mathbb{F}$ then space of vectors $v \in \mathbb{F}^n$ such that $Av{=}0$ is called Null Space.*

Note 6. For a given matrix A, $Ax{=}0$ has a unique solution $\Leftrightarrow$ A has a full rank. More generally $A_{m \times n}x{=}b$ has a unique solution $\Leftrightarrow m = n$ and A has a full rank. Otherwise there are more than one solutions.

Definition 7. (Generalized Inverse) *Let A be an $m \times n$ matrix. A matrix G is called generalized inverse of A if $Gb = b'$ is a solution of $Ax = b$, whenever $Ax = b$ has a solution.*

Note 8. If A is an invertible square matrix, then Generalized Inverse of A is A^{-1}.

2.1 Sweep-out Method

Following operations on any $m \times n$ matrix are called elementary row operations:

i. Interchange two rows. This corresponds to pre-multiplying the matrix by a Permutation Matrix.

ii. Multiply a rwo by a non-zero constant. This corresponds to pre-multiplying the matrix with matrix of the form $\begin{pmatrix} 1 & 0 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 1 & 0 & \ldots & \vdots \\ 0 & 0 & \ldots & \alpha & \ldots & \vdots \\ 0 & 0 & \ldots & \ldots & \ddots & . \end{pmatrix}$ which is a diagonal matrix and only of the diagonal elements different from 1.

iii. Addition of two rows. This corresponds to pre-multiplying the matrix with matrix of the form $I_{n \times n} + A_{ij}$ where $A_{xy} = (a_{ij})_{n \times n}$ and $a_{ij} = 1$ if $i = x, j = y$ and 0 otherwise.

Note 9. It is easy to observe that each of the operation above is invertible. It is easy to construct the corresponding inverse matrix for each of the operations. However, these operations dont commute in general.

Lemma 10. *If a matrix A in a field $\mathbb{F}$ can be reduced to a matrix A' by series of elementary row operations with corresponding pre-multiplication matrices as $P_1, P_2, \ldots, P_t$ then $A' = \left(\prod_{i=1}^{t} P_i\right)A$.*

Note 11. In all complexity calculations we consider addition and multiplication on the underlying field are $O(1)$ operations.

While matrix computation in general takes $O(n^3)$ steps, given the special nature of matrices corresponding to elementary row operations, multiplication by these matrices can be realized much more effeciently.

Lemma 12. *Given an $n \times n$ matrix M in a field $\mathbb{F}$. There exists an $O(n)$ algorithm to compute the resulting matrix obtained by multiplying a matrix correponding to elementary row operations.*

Definition 13. (Hermite Canonical Form) *A Square matrix M is said to be HCF if:*

i. *M is upper triangular.*

ii. *Diagonal elements of M are either 0 or 1.*

iii. *If diagonal element is 1 then all other entries in that column are 0.*

iv. *If diagonal element is 0, then all other entries in that row are 0.*

Theorem 14. (Hermite Canonical Form) *Any Square matrix whose elements are from a field, can be converted to* ***Hermite Canonical Form*** *using elementary row operations in $O(n^3)$ time.*

Proof. We shall specify the steps to construct the required canonical form with inductive justification:

Let $A = (a_{ij})_{n \times n}$ be the given matrix with elements from a Field.

If $a_{11} \neq 0$, then sweep out the first column using a_{11} as pivot.

If $a_{11} = 0$, then if $\exists i > 1$ such that $a_{i1} \neq 0$, then interchange rows $i, 1$ and sweep out the first column using the new a_{11} as the pivot. Otherwise, whole of first column is zero.

Suppose, that after $k - 1$ steps, the $(k-1) \times (k-1)$ principal sub-matrix is in HCF.

If $a_{kk} \neq 0$ sweep out the k^{th} column using a_{kk}.

If $a_{kk} = 0$, and if $\exists l > k$ such that $a_{kl} > 0$, If $\exists i < k$, such that $a_{ik} \neq 0$, but $a_{ii} = 0$, then interchange i, j rows and sweep out the k^{th} column using the new a_{pk} as the pivot.

After these operations the principle submatrix of the resulting matrix of order k is in HCF.

These operations suggest the algorithm and use at most $O(n^3)$ field operations. □

Note 15. Above method clearly works for non square matrices as well.

Theorem 16. *Consider a system of equations $Ax = b$, where A is an $n \times n$ matrix in a field $\mathbb{F}$. If an augmented matrix formed as $[A\colon I_{n\times n}\colon b]$ can be redued to its Hermite Canonical Form as $[H_{n\times n}\colon G_{n\times n}\colon d]$. Then following statements hold:*

i. *The #1s on the diagonal of H is rank of A.*

ii. *G is a generalized inverse of A.*

iii. *System of equations $Ax = b$ is consistent $\Leftrightarrow d_i = 0$ whenever $h_{ii} = 0$.*

iv. *Any solution of the system of equations $Ax = b$ is of the form: $d + (1 - H)z, z \in \mathbb{F}^n$.*

v. *The columns of H form a basis for Null Space of A.*

vi. *Let $1 \leqslant i_1 \leqslant i_2 \ldots \leqslant n$ be such that $h_{i_j,i_j} = 1$ and all $h_{ii} = 0$ and $B = [A_{*i_1}\colon A_{*i_2}\colon \ldots\colon A_{*i_r}]$ and $C = \begin{pmatrix} H_{i_1*} \\ H_{i_2*} \\ \vdots \\ H_{i_r*} \end{pmatrix}$ then $A = B \times C$.*

Proof. -: FILL THE DETAILS :- ☐

Theorem 17. *Let A be an $n \times n$ non singular matrix. Then there is a permutation matrix P, s.t. all principal sub-matrices of PA are non-singular.*

Proof. Steps are similar to Theorem:14. In each step collect all the permutations required in order. Then P is the product of all those permutation matrices taken in order. ☐

Theorem 18. **(LU Decomposition)** *Let A be an $n \times n$ non singular matrix, such that all the principal sub matrices of A are non-singular, then we can write $A = LU$, where*

$$L = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 \\ l_{21} & 1 & 0 & \ldots & 0 \\ l_{31} & l_{32} & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \ldots & 1 \end{pmatrix}_{n\times n}$$

$$U = \begin{pmatrix} u_{11} & u_{12} & \ldots & u_{1n} \\ 0 & u_{22} & \ldots & u_{2n} \\ 0 & 0 & & \vdots \\ \vdots & \vdots & & \\ 0 & 0 & \ldots & u_{nn} \end{pmatrix}_{n\times n}$$

and both L, U can be computed in $O(n^3)$ time.

Proof. Since A is a such that all its principal sub matrices are non-singular we can solve for each of l_{ij} and u_{ij} by multiplying L and U and then comparing the result with A. ☐

---

3 LUP decomposition of Matrix and Applications

3.0.1 Efficient Methods for Multiplying Matrices and Polynomials

While general $n \times n$ matrix and n^{th} order Polynomial multiplication takes $O(n^3)$ time. Efficient methods for multiplication exist. Earliest successful attempt can be traced back to Karatsuba's algorithm for matrix multiplication published around 1950. This method is believed to be motivation for research towards more efficient matrix multiplication as well and first successful efficient algorithm for matrix multiplication has been published by Strassen in 1967.

Karatsuba's Algorithm for Multiplication of two polynomials. Let $P(x), Q(x)$ be two degree-$2n$, $n \in \mathbb{N}$, polynomials expressed as: $P(x) = x^n P_1(x) + P_0(x)$, $Q(x) = x^n Q_1(x) + Q_0(x)$. The product of both the polynomials can be computed if we have $P_0Q_0, P_1Q_0 + P_0Q_1, P_1Q_1$ computed, which normally requires computing four multiplications. However by using extra addition which is cost effective than multiplication we can reduce the number of required multiplications to three, as follows: We can compute $P_0Q_0, P_1Q_1, (P_1 + P_0)(Q_1 + Q_0) - P_0Q_0 - P_1Q_1$ which precisely give the 3 expressions required for computing product with only 3 multiplications. This suggests a recursive divide and conquor algorithm of order complexity: $O\big(n^{\log_2^3}\big)$ as opposed to the normal method which require $O(n^2)$ time.

Strassen's algorithm for computing the product of a matrices is similar, however require computing more non-trivial expressions. To compute product of two matrices of order $2n \times 2n$, if we visualize the matrices as follows:

$$C_{2n\times 2n} = A_{2n\times 2n} B_{2n\times 2n}$$

$$\begin{pmatrix} C^0_{n\times n} & C^1_{n\times n} \\ C^2_{n\times n} & C^3_{n\times n} \end{pmatrix}_{2n\times 2n} = \begin{pmatrix} A^0_{n\times n} & A^1_{n\times n} \\ A^2_{n\times n} & A^3_{n\times n} \end{pmatrix}_{2n\times 2n} \begin{pmatrix} B^0_{n\times n} & B^1_{n\times n} \\ B^2_{n\times n} & B^3_{n\times n} \end{pmatrix}_{2n\times 2n}$$

In the above scheme, computing matrix C requires computing $C^i_{n\times n}$ and it requires computing products of 8 sub-matrices: A^0B^0, A^1B^2, A^0B^1, A^1B^3, A^2B^0, A^3B^2, A^2B^1, A^3B^3. However, we can reduce this to compting 7 products, by introducing extra additions which are cheaper than computing products, as follows - compute the following 7 expressions:

$$m_1 = (A^1_{n\times n} - A^3_{n\times n})(B^0_{n\times n} + B^3_{n\times n})$$
$$m_2 = (A^0_{n\times n} + A^3_{n\times n})(B^0_{n\times n} + B^3_{n\times n})$$
$$m_3 = (A^0_{n\times n} - A^2_{n\times n})(B^0_{n\times n} + B^1_{n\times n})$$
$$m_4 = (A^0_{n\times n} + A^1_{n\times n})(B^3_{n\times n})$$
$$m_5 = (A^0_{n\times n})(B^1_{n\times n} - B^3_{n\times n})$$
$$m_6 = (A^3_{n\times n})(B^2_{n\times n} - B^0_{n\times n})$$
$$m_7 = (A^2_{n\times n} + A^1_{n\times n})(B^0_{n\times n})$$

which in total require computing 7 products of order $n \times n$. Then matrix C can be computed as

$$C = \begin{pmatrix} m_1 + m_2 - m_4 + m_6 & m_4 + m_5 \\ m_6 + m_7 & m_2 - m_3 + m_5 - m_7 \end{pmatrix}_{2n\times 2n}$$

This procedure, clearly suggests a divide and conquer algorithm. If $T(n)$ denotes the time required for computing a matrix of order $n \times n$ then $T(n) = 7T\big(\frac{n}{2}\big) + 18\Big(\frac{n^2}{4}\Big)$. Since the above procedure requires computing 7 products and 18 sums of order $\frac{n}{2} \times \frac{n}{2}$. Then $T(n) = O\big(n^{\log_2^7}\big) = O(n^{\sim 2.81})$.

Note 19. In the above method clearly works only when the order of matrix is a power of 2. Given a matrix of arbitrary order, we embed the it in a larger square matrix whose order is a power of 2 to use the above method.

Note 20. Strassen's Algorithm being the first efficient method for multiplication of matrices, has created the interest in the question if multiplication can be performed better.

Notation 21. *Time complexity for matrix multiplication of $n \times n$ matrices is denoted as $M(n) = O(n^{\omega})$. If $\omega = 2 + \varepsilon$, ε is called the exponent of linear algebra.*

3.0.2 Matrix Multiplication $\Leftrightarrow$ Matrix Inversion

Theorem 22. *Matrix multiplication is no harder than Matrix Inversion.*

Proof. Suppose there exists and algorithm $\mathcal{A}$ to invert a Matrix. Then one can construct an algorithm $\mathcal{B}$ for multiplication as follows:

Let X, Y be two $n \times n$ matrices to be multiplied,

construct $D = \begin{pmatrix} I_{n\times n} & X & Y \\ O_{n\times n} & I_{n\times n} & Y \\ O_{n\times n} & O_{n\times n} & I_{n\times n} \end{pmatrix}_{3n\times 3n}$. Then the matrix D is invertible and inverse given as $D^{-1} = \begin{pmatrix} I_{n\times n} & -X & XY \\ O_{n\times n} & I_{n\times n} & -Y \\ O_{n\times n} & O_{n\times n} & I_{n\times n} \end{pmatrix}_{3n\times 3n}$. This can be verified by explicit computation.

So, we shall construct algorithm $\mathcal{B}$ as follows:

1. Accept two matrices $X_{n\times n}, Y_{n\times n}$.
2. Construct the matrix D as illustrated above.
3. Call Algorithm $\mathcal{A}$ with input as D to get D^{-1}.
4. Product of X, Y is the right principal submatrix of D^{-1} of order $n \times n$. □

Lemma 23. *If A is an $2n \times 2n$ matrix. written as $A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}_{2n\times 2n}$ each A_{ij} an $n \times n$ sub-matrix. If A is non-singular then A_{11} is non singular and the inverse of A is given as:*

$$A^{-1} = \begin{pmatrix} A_{11}^{-1} + A_{11}^{-1} A_{12} S^{-1} A_{21} A_{11}^{-1} & -A_{11}^{-1} A_{12} S^{-1} \\ -S^{-1} A_{21} A_{11}^{-1} & S^{-1} \end{pmatrix}$$

where $S = A_{22} - A_{21} A_{11}^{-1} A_{12}$.

Theorem 24. *Matrix Inversion is no harder than Matrix multiplication.*

Theorem 25. *If A is a matrix that is invertible, then A can be written as $A = LUP$ where P is a product of permutation matrices.*

Proof. From Theorem 17. If A is invertible then there exist permutation matrices $Q_1, Q_2, \ldots, Q_t$ such that $A\left(\prod_{i=1}^{t} Q_i\right) = B$ is such that all principal sub-matrices of B are invertible.

Let $Q = \prod_{i=1}^{t} Q_i$. Clearly Q is invertible, since each of Q_i is invertible. By Theorem 18. $B = AQ$ can be decomposed as LU. So, $AQ = LU$. So, choose $P = Q^{-1}$. □

We state the following theorem without proof.

Theorem 26. *LUP decomposition of matrix A can be computed in $O(M(n))$ time.*

LUP decomposition of matrix can be used to compute determinant of the matrix efficiently.

3.0.3 Computing matrix determinant.

Lemma 27. *If we can decompose a matrix $A = LUP$, then* $\det(A) = (-1)^{\mathrm{Sgn}(P)} \det(U)$.

Proof. If $A = LUP$ then we have $\det(A) = \det(L)\det(U)\det(P)$.

Since P is a product of permuation matrices, det $(P) = (-1)^{\mathrm{Sgn}(P)}$, which can be computed in $O(n)$ time. Since L is a lower triangular matrices with all its diagonal elements 1, we have det $(L) = 1$. Hence, det $(A) = (-1)^{\mathrm{Sgn}(P)}\det(U)$. □

This suggests an algorithm to compute the determinant in $O(M(n))$ steps.

Theorem 28. *Determinant of an $n \times n$ matrix can be computed in $O(M(n))$ steps.*

Proof. Compute the determinant if matrix $A_{n \times n}$ as follows:

1. Compute LUP decomposition of matrix $A = LUP$
2. Compute det $(P) = (-1)^{\mathrm{Sgn}(P)}$.
3. Compute det $(U) = \prod_{i=1}^{n} u_{ii}$.
4. det $(A) = (-1)^{\mathrm{Sgn}(P)} \prod_{i=1}^{n} u_{ii}$.

Since, LUP decomposition requries $O(M(n))$ steps and all other steps require $O(n)$ steps, determinant can be computed in $O(M(n))$ steps. □

Lemma 23. Suggests a very efficient method for computing inverse of a non singular matrix. If $A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}_{2n \times 2n}$ is a non singular matrix, and is upper triangular then $A_{21} = 0_{n \times n}$. A_{11}, A_{22} are upper triangular and so is $A^{-1} = \begin{pmatrix} A_{11}^{-1} & -A_{11}^{-1} A_{12} A_{22}^{-1} \\ 0_{n \times n} & A_{22}^{-1} \end{pmatrix}_{2n \times 2n}$. This suggests a very simple divide and conquer algorithm for computing inverse of an upper triangular matrix:

```
InvertUTMatrix($A$):
1. $A_{11}^{-1} \leftarrow$ InvertUTMatrix($A_{11}$).
2. $A_{22}^{-1} \leftarrow$ InvertUTMatrix($A_{22}$).
3. $A^{-1} = \begin{pmatrix} A_{11}^{-1} & -A_{11}^{-1} A_{12} A_{22}^{-1} \\ 0 & A_{22}^{-1} \end{pmatrix}$.
```

If $T(n)$ is the number of steps required to compute the inverse of $n \times n$ upper triangular matrix. Then $T(n) = 2T\left(\frac{n}{2}\right) + 2\left(\frac{n}{2}\right)^3 + \left(\frac{n}{2}\right)^2$.

A MATRIX MULTIPLICATION INEQUALITY IS ASSUMED HERE! FILL THE DETAILS.

For a general matrix, we can use LUP decomposition and use the above method to invert the matrices U, L. P being a permutation matrix is invertible in $O(n)$ steps for an $n \times n$ matrix. So, Inversion of matrix can be done in $O(M(n))$ time.

Theorem 29. *Given a non singular matrix A. Inverse of A can be computed in $O(M(n))$ steps.*

So the methods above describe computing determinant and inverse of a matrix. Determinant of the matrix can be defined over a Ring but all the above operations assume a field. This restriction can be overcome in case of Integral Domains by embedding the elements of given matrix into the field of fractions. But the issue is growth of intermediate fractions, so even though total number of operations wont change, each operation may become very costly. So an alternative for integer matrices is to compute determinants in prime fields and then combine the results.

Given a matrix M whose entries are integers, suppose there is an apriori known bound on det $(M) < \mathbf{\Delta}$. Let p_1, p_2, ..., p_r be distinct primes such that $\mathbf{\Delta} < p_1 p_2 \ldots p_r$. Then compute $\mathbf{\Delta}_i =$ det $(M) (\mathrm{mod}\ p_i)$ and combine the results by using Chinese Reminder Theorem to obtain det $(M) (\mathrm{mod}\ p_1 p_2 \ldots p_r) = d$. But since $\mathbf{\Delta} < p_1 \ldots p_r$ we have $d =$ det (M). Such a bound on determinant can be obtained using hardamard bound on determinant given as $\mathbf{\Delta} \leqslant \prod_{1 \leqslant i \leqslant n} \left(\sum_{1 \leqslant j \leqslant n} |m_{ij}|^2\right)^{\frac{1}{2}}$.

---

4 Upper Hessenberg Form

Definition 30. (Characteristic Polynomial) *Characteristic Polynomial of a matrix A denoted as $Ch_A(x) := \det(xI - A)$.*

Lemma 31. *If A is an $n \times n$ matrix then $\deg(Ch_A(x)) = n$.*

Theorem 32. (Caley Hamilton) *Every square matrix satistifies its Characteristic polynomial.*

Definition 33. (Minimal Polynomial) *A Minimal polynomial of matrix A is least degree monic polynomial $m_A(x)$ such that $m_A(A) = O_{n\times n}$.*

Lemma 34. *If A is a matrix then $m_A(x)|Ch_A(x)$.*

Proof. □

Proposition 35. *If A is a matrix and $\rho(x)$ is an irreducible polynomial which divides $Ch_A(x)$ then $\rho(x)|m_A(x)$.*

Proof. Since $m_A(x)$ is a polynomial we have $(x-y)|m_A(x) - m_A(y)$.

So we can write $m_A(x) - m_A(y) = (x-y)k(x,y)$.

Substituting $x = \lambda I$ and $y = A$, we have $m_A(\lambda I) - m_A(A) = m_A(\lambda I)$ (by definition. of m_A)

$\Rightarrow m_A(\lambda I) = (\lambda I - A)k(\lambda I, A)$

$\Rightarrow m_A(\lambda)I = (\lambda I - A)k(\lambda I, A)$

$\Rightarrow \det(m_A(\lambda)I) = \det(\lambda I - A)\det(k(\lambda I, A))$

$\Rightarrow (m_A(\lambda))^n = Ch_A(\lambda)\det(k(\lambda I, A))$

So, if $\rho(\lambda)|Ch_A(\lambda)$ then $\rho(\lambda)|(m_A(\lambda))^n$. But since $\rho(x)$ is irreducible $\rho(x)|m_A(x)$. □

Corollary 36. *The distinct roots of characteristic polynomial and minimal polynomials of a matrix are the same they differ only in multiplicities.*

How to compute the minimal polynomial for a given matrix? It is clear to see that I, A, ..., A^n are linearly dependent, since characteristic polynomial is of degree n. So find smallest r such that I, A, ..., A^r are linearly dependent, r will be the deg of the minimal polynomial of A and coefficients of minimal polynomial can be obtained as coefficients of linear dependence. This can be done using Guassian Elimination which takes $O(n^4)$ time. This however is not the best, there exist algorithms to compute minimal polynomial in $O(n^3)$ time.

How to compute the characteristic polynomial? Let $F_n[x]$ be the vector space of all polynomial over F whose degree is atmost n. Then $F_n[x]$ has a dimension of $n+1$ over F with a basis $\{1, x, x^2, ..., x^n\}$. We shall give a more non trivial basis for $F_n[x]$.

Let c_i, $1 \leqslant i \leqslant n$ be n distinct elements of F. We shall define polynomials, $p_i(x) = \frac{(x-c_1)(x-c_2)...(x-c_{i-1})...(x-c_{i+1})...(x-c_n)}{(c_i-c_1)(c_i-c_2)...(c_i-c_n)}$ for $1 \leqslant i \leqslant n$. Then we can observe that $p_i(c_j) = 1 \Leftrightarrow i = j$ and 0 otherwise.

Lemma 37. *$\{p_1(x), ...p_n(x)\}$ forms a basis for $F_n[x]$ over F.*

Any $f(x) \in F_n[x]$ can be written as $f(x) = \sum_{1\leqslant i\leqslant n} f(c_i)p_i(x)$. This observation can be used to compute the characteristic polynomial of matrix M. We compute $Ch_A(c_i) = \det(c_iI - A)$ for all i. Then write $Ch_A(x) = \sum_{1\leqslant i\leqslant n} Ch_A(c_i)p_i(x)$. Since computing $n\times n$ determinant in F takes $O(M(n))$ time computing characteristic polynomial takes $O(nM(n))$ time.

Definition 38. *Let A, B be two $n \times n$ matrices. They are said to be similar if there exists an invertible matrix C such that $A = CBC^{-1}$.*

Note 39. Similarity is an equivalence relation.

Lemma 40. *If A, B are similar then they have same characteristic polynomial.*

Exercise 1. Find two non-similar matrices which have same characteristic polynomial.

Definition 41. (Upper Hessenberg Form) *A matrix of the following form is said to be in upper hessenberg form.*

$$\begin{pmatrix} h_{11} & h_{12} & h_{13} & \dots & h_{1n-1} & h_{1n} \\ k_2 & h_{22} & h_{23} & \dots & h_{2n-1} & h_{2n} \\ 0 & k_3 & h_{33} & \dots & h_{3n-1} & h_{3n} \\ 0 & 0 & k_4 & \dots & h_{4n-1} & h_{4n} \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & \dots & k_n & h_{nn} \end{pmatrix}_{n \times n}$$

Theorem 42. *Given a matrix A we can apply a series of similarity transforms on it to get it into Upper Hessenberg Form.*

5 Smith Canonical Form

Definition 43. (Non-derogatory Matrix) *A matrix M is said to be non-derogatory if minimal polynomial of matrix $P_M(x)$ is same as characterisic polynomial of M.*

Definition 44. (Elementary Jordan Matrix) *A matrix of the following form is called Elementary Jordan Matrix.*

$$J_n(c) = \begin{pmatrix} c & 0 & 0 & 0 & \dots & 0 \\ 1 & c & 0 & 0 & \dots & 0 \\ 0 & 1 & c & 0 & \dots & 0 \\ & & & & \vdots & \\ 0 & 0 & 0 & \dots & 1 & c \end{pmatrix}_{n \times n}$$

Note 45. Elementary Jordan Matrix is Non-derogatory matrix and that can be observed by computing the successive powers of the matrix that minimal polynomial cannot have degree less than n.

Definition 46. (Companion matrix of Polynomial) *Companion matrix of the polynomial $p(x) = x^n + a_{n-1}x^{n-1} + \dots + a_1 x + a_0$ denoted by $C(p)$ is the matrix of the following form:*

$$\begin{pmatrix} 0 & 0 & \dots & 0 & -a_0 \\ 1 & 0 & \dots & 0 & -a_1 \\ \vdots & & & & \\ 0 & 0 & \dots & 1 & -a_{n-1} \end{pmatrix}_{n \times n}$$

Note 47. It is easy to observe that characteristic polynomial of $C(p)$ is $p(x)$.

Notation 48. *Henceforth we shall denote characteristic polynomial of a matrix M as $Ch_M(x)$.*

Definition 49. (Direct Sum) *The direct sum of two matrices square matrices $A_{n \times n}, B_{m \times m}$ is the matrix $\begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix}_{(m+n) \times (m+n)}$ and is denoted as $A \oplus B$.*

Lemma 50. $\det(A \oplus B) = \det(A)\det(B)$

We shall state the following result without proof.

Theorem 51. (Jordan Canonical Form) *Every matrix is similar to a direct sum of elementary jordan matrices, which is unique upto a permutation of elementary jordan blocks.*

Notation 52. *We shall denote F for a field and $F[x]$ for ring of polynomials over field F. We shall denote the set of $n \times n$ matrices in ring R as $M_{n\times n}(R)$.*

Definition 53. (Rational Canonical Form) *Let $M = \oplus_{k=1}^{s} C(d_k)$, where $d_1, d_2, \dots, d_s$ are non-constant monic polynomials in $F[x]$ such that $d_i | d_{i+1}$. Then M is said to be in Rational Canonical Form.*

Definition 54. (Invariant Factors) *If a matrix A is $\sim$ to $M = \oplus_{k=1}^{s} C(d_k)$ then the polynomials d_i are called invariant factors of A.*

Following results show why Rational Canonical Form is useful in computational prespective.

Lemma 55. *Minimal polynomaial of a matrix $M = \oplus_{j=1}^{s} C(d_j)$ is $\mathrm{LCM}(d_1, \dots d_s)$.*

Theorem 56. *If $d_1, d_2, \dots d_s$ are invariant factors of A, then*

i. $m_A(x) = d_s(x)$

ii. $Ch_A(x) = d_1 d_2 \dots d_s$

Proof. □

Definition 57. *A matrix P in $M_{n\times n}(F[x])$ is said to be unit if there exist a matrix $Q \in M_{n\times n}(F[x])$ such that $PQ = I$.*

Lemma 58. *If matrices P, Q are units then so is PQ.*

Lemma 59. *If matrix $P \in M_{n\times n}(F[x])$ is unit then $\det(P) \in F$.*

In the following discussion, following operations on matrices are treated as elementary row operations:

i. interchange two rows.

ii. multiply a row by an element of $F[x]$ and add it to another row.

Both the above operations correspond to multiplication of original matrix by another matrix in $M_{n\times n}(F[x])$.

Lemma 60. *Matrices corresponding to elementary row operations are units.*

Definition 61. (Unimodular Matrix) *Products of elementary row matrices and elementary column matrices are called **Unimodular**.*

We shall state the following theorem without proof.

Theorem 62. *A matrix is a unit iff it is unimodular.*

Definition 63. *Let $A, B \in M_{n\times n}(F[x])$. Then A is said to be equivalent to B, if there are units P, Q such that $PAQ = B$.*

Definition 64. *Let $A \in M_{n\times n}(F[x])$ then for $1 \leqslant k \leqslant n$ let $d_k(A)$ denote the gcd of all $k \times k$ minors of A. Then $d_k(A)$ is called the k^{th} determinant divisor of A.*

Lemma 65. *If two matrices in $M_{n\times n}(F[x])$ are similar then they have the same determinantal divisors.*

Note 66. The above notion can also be exnted to rectangular matrices.

Note 67. $\gcd(f_1,\ldots f_n)\neq 0 \Leftrightarrow$ atleast one of f_i is non-zero.

Definition 68. (Determinant Rank) *Determinant rank of a matrix A denoted by $\rho(A)$ is the largest integer r for which $d_r(A)\neq 0$.*

Theorem 69. *If A is a matrix in $M_{n\times n}(F[x])$ then for $1\leqslant k\leqslant\rho(A)$ $d_k(A)|d_{k+1}(A)$.*

Proof. □

Theorem 70. (Smith Canonical Form) *Every non-zero matrix $A\in M_{n\times n}(F[x])$ with $r=\rho(A)$ is equivalent to matrix of the following form*

$$\begin{pmatrix} f_1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & f_2 & 0 & \ldots & 0 & 0 \\ 0 & 0 & f_3 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \ldots & f_r & \ldots & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & 0 & \ldots & 0 \end{pmatrix}_{n\times n}$$

where $f_i\in F[x]$ and $f_i|f_{i+1}$ this is called Smith Canonical Form of the matrix.

Theorem 71. *Let $A\in M_{n\times n}(F[x])$ and $r=\rho(A)$ then the polynomials $f_1, f_2,\ldots f_r$ in the smith canonical form of A are called the invariant factors of A.*

Lemma 72. *If $d\in F[x]$ then the smith canonical form of $xI-c(d)$ is $\operatorname{diag}(1,1,\ldots 1,d)$.*

Proof. □

Theorem 73. *For every matrix $A\in M_{n\times n}(F[x])$ smith canonical form is unique.*

Proof. □

Following theorem gives connection between Rational Canonical Form and Smith Canonical Form.

Theorem 74. *Let $B\in M_{n\times n}(F)$. If the invariant factors of B are $d_1,\ldots d_s$ then smith canonical form of $xI-B$ is equal to $\operatorname{diag}(1,1,\ldots,1,d_1,\ldots d_s)$.*

Theorem 75. *Let $A,B\in M_{n\times n}(F)$. Then A is similar to B iff $xI-A$ is equivalent to $xI-B$ iff $xI-A$ and $xI-B$ have the same smith canonical form.*

So to test similarity of two matrices $A,B\in M_{n\times n}(F)$ we resort to following steps:

1. Obtain U,V such that $U(xI-A)V=\operatorname{diag}(p_1(x),\ldots,p_n(x))$

2. Obtain R, Q such that $R(xI - B)W = \mathrm{diag}(q_1(x), \ldots, q_n(x))$
3. If $p_i = q_i$ for $1 \leqslant i \leqslant n$ then $A \sim B$.
4. Suppose similarity holds. Then $P(xI - A)Q = xI - B$ where $P = R^{-1}U$ and $Q = VW^{-1}$ then T computed as right value of $Q(x)$ at B and then T^{-1} is left value of $P(x)$ at B is such that $TAT^{-1} = B$.

---

6 Fast Fourier Transform

This part is mostly covered from "The Design and Analysis of Computer Algorithms" by Aho, Hopcroft and Ullman.

6.1 Bit-Complexity of Fast Fourier Transform

6.2 Schonhage-Strassen algorithm

7 Computational complexity of Fundamental Integer operations.

This part is mostly covered from "The Design and Analysis of Computer Algorithms" by Aho, Hopcroft and Ullman.

8 Greatest Common Divisor

Definition 76. (GCD) *If R is a euclidean ring and $a, b \in R$ then g is called the* $\mathrm{GCD}(a, b)$ *if $g|a$, $g|b$ and if for any $h \in R, h|a, h|b$ then $h|g$.*

8.1 Euclidean GCD Algorithm

8.2 Half-GCD Algorithm

9 Polynomial Factoring

9.1 Berlekamp's Method

Fix a prime finite field $\mathbb{F}_p$. Let $f(x) \in \mathbb{F}_p[x]$ and $\deg(f) = n$. Without loss of generality we shall consider f is a square-free polynomial, for otherwise we can make it square free using the following lemma:

Definition 77. (Derivative) *If $f(x) \in R[x]$ where R is a ring, and $f(x) = \sum a_i x^i$ then we shall define derivative of f as $f'(x) = \sum i a_i x^{i-1}$.*

Notation 78. *Going further we assume $p \in \mathbb{N}$ is a prime number and $\mathbb{F}_p$ is the finite field of order p.*

Lemma 79. *$f \in \mathbb{F}_p[x]$ is square-free iff* $\gcd(f, f') = 1$.

Lemma 80. *If $f \in \mathbb{F}_p[x]$ is not square-free then $\frac{f(x)}{g(x)}$ is square-free, where $g(x) = \gcd(f, f')$.*

Note 81. There is a procedure to check if given polynomial is irreducible, Ref. Mc.William-sloane Ch.4.

To factor the polynomial $f(x)$, the basic intuition is that we take the gcd of $f(x)$ with "some" polynomial, such that gcd turns out to be non-trivial.

Over $\mathbb{F}_p$ we know that $x^p \equiv x \pmod p, \forall x \in \mathbb{F}_p$. Also let $g(x) = (x^p - x)' = px^{p-1} - 1 = -1$ in $\mathbb{F}_p[x]$. So $f(x) = x^p - x$ is square-free. Hence $f(x) = (x^p - x) = \prod_{i \in \mathbb{F}_p} (x - i)$. So for any $v(x) \in \mathbb{F}_p[x]$ we have $(v(x))^p - v(x) = \prod_{i \in \mathbb{F}_p} (v(x) - i)$. And this is a square free splitting as $\gcd((v(x) - i), (v(x) - j)) = 1$ for $i \neq j$. So for a given square-free $f(x)$, if we can find $v(x)$ such that $(v(x))^p \equiv (v(x)) \bmod f(x)$ then any irreducible factor of $f(x)$ will divide exactly one factor $v(x) - k$. So we shall try to find such $v(x)$. We shall start by assuming $v(x)$ with required property and derive the necessary condition.

Let $f(x)$ be a square-free polynomial in $\mathbb{F}_p[x]$ and let $v(x)$ be a polynomial such that $(v(x))^p - v(x) \equiv 0 \pmod{f(x)}$. Let $v(x) = \sum_{0 \leqslant i \leqslant t} (a_i x^i)$ then in $\mathbb{F}_p[x]$ we have $(v(x))^p = \sum_{0 \leqslant i \leqslant t} (a_i x^{ip})$. Then $(v(x))^p \pmod{f(x)} = \sum_{0 \leqslant i \leqslant t} a_i (x^{pi} \bmod f(x))$. Each of Let $x^{pi} \bmod f(x)$ is of degree less than $n = \deg(f(x))$, so we shall each of $(x^{pi} \bmod f(x))$ as $\left(\sum_{0 \leqslant j < n} b_{i,j} x^j\right)$. Then we can easily see following holds:

Lemma 82. *Choose the degree of $v(x)$ to be $n - 1$. Then taking the notations above and defining $B = (b_{i,j})_{n \times n}$. $(v(x))^p - v(x) \equiv 0 \pmod{f(x)}$ iff $\tilde{v}\, B = \tilde{v}$, where $\tilde{v} = (a_i)_{0 \leqslant i < n}$ is an n vector of coefficients of $v(x)$.*

So, $\tilde{v}$ is in the null space of $B - I$. Since $B - I$ is an $n \times n$ matrix we can compute the basis of null space using sweep-out method in Sec.1 in $O(n^3)$ steps. Let the basis of $B - I$ be $\bigcup_{i<r} \{v^{[i]}(x)\}$. Where $r = \#$of irreducible factors of $f(x)$.

9.2 Cantor-Zassenhaus randomized algorithm

Let v be a random linear combination of the basis vectors of the null space of $B - I$. Compute $\gcd\left((v(x))^{\frac{p-1}{2}} - 1, f(x)\right) = \gcd\left((v(x))^{\frac{p-1}{2}} - 1 (\bmod f(x)), f(x)\right)$ (here exponentiation can be done in $O(\log p)$ steps using square-multiply algorithm). We can argue that with probability atleast $\frac{4}{9}$ this gives a factor of $f(x)$.

We can analyze the performance by seeing in what all ways can gcd computation above can turn out to be trivial?

i. If $p_i(x) \Big| (v(x))^{\frac{p-1}{2}} - 1$ for all i. In this case gcd is $(v(x))^{\frac{p-1}{2}} - 1$

ii. If $p_i(x) \nmid (v(x))^{\frac{p-1}{2}} - 1$ for all i. In this case gcd is 1.

Consider the case i. Given a $p_i(x)$ and $v(x)$ there is a unique s_i such that $p_i(x) | v(x) - s_i$. Then $(v(x))^{\frac{p-1}{2}} \equiv 1 (\bmod p(x))$ and $(v(x)) \equiv s_i (\bmod p_i(x))$. Both these conditions imply $s_i^{\frac{p-1}{2}} \equiv 1 (\bmod p)$ which happens iff s_i is a Quadratic residue in $\mathbb{F}_p$.

Assumption: If $v(x)$ is chosen randomly from the null space of $B - I$ and s_i is unifomly chosen from $\mathbb{F}_p$ Since there are $\frac{p-1}{2}$ quadratic residues in $\mathbb{F}_p$, the probability that s_i is a Quadratic residue is $\frac{p-1}{2p}$. So probability for trivial case i. is $\left(\frac{p-1}{2p}\right)^r$.

Similarly, $\gcd\left((v(x))^{\frac{p-1}{2}} - 1, f(x)\right) = 1$ iff s_i are all Quadratic Non Residues and this happens with probabilty $\left(\frac{p+1}{2p}\right)^r$. So, combining above cases probability that $\gcd\left((v(x))^{\frac{p-1}{2}} - 1, f(x)\right)$ is non-trivial is atlesat $1 - \left(\frac{p-1}{2p}\right)^r - \left(\frac{p+1}{2p}\right)^r > \left(\frac{4}{9}\right)$.

9.3 Factorization over $\mathbb{Z}[x]$

Let $u(x)\in\mathbb{Z}[x]$ and $u(x)=u_0+u_1x+\ldots+u_nx^n$, without loss of generality assume gcd $(u_0,u_1,\ldots,u_n)=1$ and $u(x)$ is square-free.

Example 83. $f(x)=x^8+x^6-3x^4-3x^3+8x^2+2x-5$ can be factored in $\mathbb{Z}_{13}$ as well as $\mathbb{Z}_2$ but is irreducible in $\mathbb{Z}$.

Note 84. There exist polynomials which have consistent degree factorization in $\mathbb{Z}_p$ for every prime p but not in $\mathbb{Z}$.

Note 85. It is easy to be convinced that almost all polynomials over integers are irreducible.

Lemma 86. *Suppose p is a big enough prime, such that coefficient in any true factorization of $u(x)=v(x)w(x)$ be in the range $\left(\frac{-p}{2},\frac{p}{2}\right)$ then factoring in $\mathbb{Z}_p$ we obtain true factorization over $\mathbb{Z}$.*

However there are two problems: In general it is difficult to get a good bound on p, p may be too large. So, solution is to use a method called **"Hensel Lifting"**

9.3.1 Hensel Lifting

Lemma 87. *Let $u(x),v(x),a(x)$ and $b(x)$ be such that:*

i. $u(x)=v(x)w(x)$ *in* $\mathbb{Z}_q$.

ii. $a(x)v(x)+b(x)w(x)\equiv 1$ *in* $\mathbb{Z}_p$.

iii. $c.l(v)\equiv 1(\operatorname{mod} r)$ $(l(v)\text{ is the leading coefficient of }v(x))$

iv. $\deg(u)=\deg(v)+\deg(w)$

v. $r=\gcd(p,q)$

then there exist polynomials $V(x),W(x)$ such that :

i. $u(x)=V(x)W(x)$ *in* $\mathbb{Z}_{qr}$.

ii. $V(x)=v(x)$ *in* $\mathbb{Z}_q$, $W(x)=w(x)$ *in* $\mathbb{Z}_q$

iii. $l(v)=l(V)$, $\deg(v)=\deg(V)$, $\deg(w)=\deg(W)$.

Proof. □

Lemma 88. (Hensel) *Let $u(x)$, $v_e(x)$, $w_e(x)$, $a(x)$, $b(x)$ be such that $u(x)=v_e(x)w_e(x)\bmod p^2$ and $a(x)v_e(x)+b(x)w_e(x)\equiv 1(\operatorname{mod} p)$ where p is a prime and $v_e(x)$ is monic $\deg(a)\leqslant\deg(w_e)$ and $\deg(b)<\deg(v_e)$ and $\deg(u)=\deg(v_e)+\deg(w_e)$. Then, there are polynomials $v_{e+1}(x)$ and $w_{e+1}(x)$ satisfying the same conditions e increased by 1. Further $v_{e+1}(x)$ and $w_{e+1}(x)$ are unique modulo p^{e+1}.*

Using this lemma once we have factorization in $\mathbb{Z}_p$ we can lift the factorization to $\mathbb{Z}_{p^2},\mathbb{Z}_{p^3},\ldots$ There is an improvement suggested to this by Zessenhaus using which we can lift the factorization much faster.

Lemma 89. (Zessenhaus) *Suppose $u(x)\equiv v(x)w(x)$ in $\mathbb{Z}_q$ and $a(x)v(x)+b(x)w(x)=1$ in $\mathbb{Z}_p$ where $p=\gcd(p,q)=r$. Then there are polynomials $V(x)$, $W(x)$, $A(x)$, $B(x)$ such that $u(x)=V(x)W(x)$ in $\mathbb{Z}_{qr}$ and $A(x)V(x)+B(x)W(x)\equiv 1(\operatorname{mod} pr)$.*

Proof. □
