Lecture 4: Finding complex solutions to univariate polynomials

Algorithmic Number Theory (Fall 2014)
Rutgers University
Swastik Kopparty
Scribes: Amey Bhangale

1 Finding integer solutions to system of linear equations

Recall from the previous lecture that the problem is, given m linear equations over $\mathbb{Z}$ in n variables, we want to find integer solutions to this system. Suppose the linear equations are:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{mn}x_n &= b_m \end{aligned}$$

Let A be the $m \times n$ matrix where the $(i,j)^{th}$ entry of A is a_{ij} and b be the vector of length m, where the i^{th} entry is b_i. Let $x = [x_1, x_2, \dots, x_n]$. In the previous lecture, we saw how to efficiently transform the matrix A into its Hermite Normal form $H = [B \mid 0]$, where B is a lower triangular matrix, by doing a sequence of *integer* column operations. Note that, each *integer* column operation can be written as product of an original matrix with an upper triangular matrix U_i of the form:

$$U_i = \begin{pmatrix} \pm1 & 0 & \dots & 0 & \dots & \dots & 0 \\ 0 & \pm1 & \dots & 0 & \dots & \dots & 0 \\ \vdots & & \ddots & * & & & \vdots \\ \vdots & & & \pm1 & & & \vdots \\ \vdots & & & & \ddots & & \vdots \\ 0 & \dots & \dots & \dots & \dots & \pm1 & 0 \\ 0 & \dots & \dots & \dots & \dots & \dots & \pm1 \end{pmatrix}_{n\times n}$$

where there is at most one diagonal entry which is -1 and there is exactly one column with $*$ which represents an integer value. Note that when the jj^{th} entry of the matrix is -1 then the $*$ must be in the j^{th} column. Let $U_1, U_2, \dots, U_q$ be a sequence of matrices corresponding to the sequence of column operations performed on the matrix A. Thus, we have

$$A \cdot (U_1 \cdot U_2 \cdot U_3 \dots U_q) = H = [B \mid 0]$$

Let $U = U_1 \cdot U_2 \cdot U_3 \dots U_q$. Note that the $\det(U_i) = \pm1$. Hence, $\det(U) = \pm1$. Clearly, the entries in U are integers and are bounded since q is some polynomial in the input size and each column operation involves multiplying by a *small* integer. Let $Adj(U)$ be the adjugate of matrix U. Then,

$U^{-1} = \frac{1}{\det(U)} \cdot Adj(U)$. Thus, we conclude that all entries in the matrix U^{-1} are integral and bounded.

Our goal is to solve $Ax = b$ for $x \in \mathbb{Z}^n$. Let $y = U^{-1}x$ which is in $\mathbb{Z}^n$ if and only if $x \in \mathbb{Z}^n$. Consider the following equivalences:

$$\begin{tikzcd} Ax = b \arrow[r, phantom, "\Longleftrightarrow"] & AU(U^{-1}x) = b \arrow[r, phantom, "\Longleftrightarrow"] & (AU)y = b \arrow[d, Leftrightarrow] \\ & {[I \mid 0]y = B^{-1}b} \arrow[r, phantom, "\Longleftrightarrow"] & {[B \mid 0]y = b} \end{tikzcd}$$

Thus, it suffices to solve for y in $[I \mid 0]y = B^{-1}b$. If the vector $B^{-1}b \in \mathbb{Z}^n$, then the set of all solutions $y \in \mathbb{Z}^n$ is of the form $[B^{-1}b, *, *, \ldots, *]$ where the first m entries are $B^{-1}b$ and the remaining entries are any integers. Otherwise, it has no integral solution. x can be obtained from y by doing the operation $x = Uy$. Since, the number of bits required to represent each entries in the matrices U, U^{-1}, B, B^{-1} is bounded by polynomial in the input size, all these operations can be done in polynomial time.

The algorithm can be summarized as follows:

Input : Matrix $A \in \mathbb{Z}^{m\times n}$ and a vector $b \in \mathbb{Z}^m$.
Output : $x \in \mathbb{Z}^n$ such that $Ax = b$. If no such x exists then output 'no solution'.

1. Let $U_1, U_2, \ldots, U_q$ be the sequence of matrices corresponding to the set of column operations that transforms A to its Hermite Normal form $[B \mid 0]$.
2. Let $U = U_1 \cdot U_2 \cdots U_q$.
3. Let $c = [B^{-1}b, 0, 0, \ldots, 0]$ where c is a vector of length n with first m entries $B^{-1}b$.
4. If $c \in \mathbb{Z}^n$ then output Uc, otherwise output 'no solution'.

A 1: Algorithm to find integer solutions

Remark: Note that, we only used the lower triangular property of B and hence the other two conditions from the definition of HNF are not needed for solving this problem. Hence, the overall algorithm can be simplified i.e. instead of finding Hermite Normal form, it suffices to translate matrix A into a lower triangular matrix by just doing Gaussian elimination, preserving the integer span of columns.

2 Finding complex solutions to univariate polynomials

In this section, we will be interested in answering the following question, which is finding a root of an univariate polynomial up to small additive error.

Q 1. *Given a polynomial $P(X) \in \mathbb{C}[X]$ of degree n and a positive integer k, find $\beta \in \mathbb{C}$ such that* *there exists $\alpha \in \mathbb{C}$ such that $P(\alpha) = 0$ and*

$$|\alpha - \beta| \leq 2^{-k}$$

Remark 1: Note that, one way to find root of a polynomial is by Newton's method but the accuracy depends on the starting point which is a delicate issue.

Remark 2: Finding approximate root of a polynomial has an application to factoring a polynomial which we will see in later lectures. The high level idea is as follows : First, find an approximate root of the given polynomial. Since the polynomial is given as a set of coefficients, they are bounded. We can use this bounded coefficients condition to find a low degree polynomial that has a root close to the approximate root.

Back to the problem of finding approximate root of a polynomial :

Input size and efficiency : We are given a polynomial $P(X) = X^n + a_1X^{n-1} + a_2X^{n-2} + \ldots + a_n$ of degree n in the form of set of coefficients $\{a_0, a_1, \cdots, a_n\}$. Let m be a number such that each coefficient a_i can be represented by m bits. We are interested in solving $Q1$ in time polynomial in the input size and k i.e $\texttt{poly}(n \cdot m, k)$.

Definition 1 (Multiplicative error)**.** *An algorithm to estimate a given quantity q is correct up to multiplicative error of k, if the value its output always lies in a range $[\frac{q}{k}, kq]$.*

Consider the following set of questions.

Q 2. *Given a polynomial $P(X) \in \mathbb{C}[X]$, $z \in \mathbb{C}$ and a positive integer k, estimate a distance of z from a root of P which is closest to z, up to a multiplicative error of k.*

Q 3. *Given a polynomial $P(X) \in \mathbb{C}[X]$ and a positive integer k, estimate a distance of 0 from the root of P closest to 0, up to a multiplicative error of k.*

Q 4. *Given a polynomial $P(X) \in \mathbb{C}[X]$and k, find two real numbers r and s such that $\frac{r}{s} \leq k$, and both the cases are true:*

1. *All roots of P are at most r in absolute value*
2. *Some root of P is at least s in absolute value.*

As an intermediate step in finding a root of a polynomial, we will use an algorithm that solves $Q2$ for a certain value of k. We will show the following reductions,

$$Q2 \implies Q3 \implies Q4 \tag{1}$$

where $Qi \implies Qj$ means it suffices to solve Qj in polynomial time in order to solve Qi in polynomial time. At the end, we will show how to solve $Q4$. First let's see how solving $Q4$ suffices to solve $Q2$:

- $Q2 \implies Q3$: Given $P(X)$ and z as in $Q2$, consider the polynomial $Q(X) = P(X) - z$. By construction, the distance of z form a root of $P(X)$ which is closest to z is same as distance of origin from a root of $Q(X)$ which is closest to the origin.

- $Q3 \implies Q4$: Given a polynomial $P(X)$ as in $Q3$, consider the polynomial $Q(X) = P(\frac{1}{X}) \cdot X^n$. If α is a root of $P(X)$ then $\frac{1}{\alpha}$ is a root of a polynomial $Q(X)$. For a polynomial $Q(X)$, given k, if we can find real numbers r and s that satisfy two cases from $Q4$ (which is same as maximum absolute value of a root of $P(X)$ lies in the interval $[s, r]$), then we can also find two real numbers, namely $r' = \frac{1}{r}$ and $s' = \frac{1}{s}$, such that following is true:

  1. All roots of $P(X)$ is at least r' in absolute value.
  2. Some of $P(X)$ is at most s' in absolute value.

  Thus, if we can solve $Q4$ with certain accuracy (for a parameter k), then we can also find two reals, r', s' such that $\frac{s'}{r'} \leq k$, such that above two statements are true. It means we can find two real numbers, r' and s' such that the minimum root of $P(X)$ (in absolute value) lies in the interval $[r', s']$. Since $\frac{s'}{r'} \leq k$, r' is an estimate of distance of origin from the root closest to it up to multiplicative error of k.

Rest of the section is to solve $Q4$. We first start with a following claim that give an approximate estimate of the absolute value of the largest root of a polynomial:

Claim 2. *Let $P(X) = X^n + a_1X^{n-1} + a_2X^{n-2} + \ldots + a_n$ and let $A = \max_i |a_i|^{\frac{1}{i}}$ then we have*

$$\frac{1}{n}A \leq \textit{absolute value largest root of } P(X) \leq 2A.$$

Proof. Let α be any root of P. We know that

$$\alpha^n = -\sum_{i=1}^{n} a_i \alpha^{n-i}$$

Therefore, for some $i \in [n]$,

$$|a_i \alpha^{n-i}| \geq \frac{|\alpha^n|}{2^i} \implies |a_i| \geq \frac{|\alpha^i|}{2^i} \implies |\alpha| \leq 2 \cdot |a_i|^{\frac{1}{i}}$$

Since the value of A is $\max_i |a_i|^{\frac{1}{i}}$, $|\alpha| \leq 2 \cdot A$. Hence, for every root α of $P(X)$, we have $|\alpha| \leq 2 \cdot A$

Let $\alpha_1, \alpha_2, \cdots, \alpha_n$ be the roots of $P(X)$. To prove the left inequality, suppose for contradiction, assume that all roots α_i of P satisfy $|\alpha_i| \leq u \cdot A$. Then, by comparing the coefficients from the following identity,

$$\begin{aligned} P(X) = \prod_{i \in [n]} (X - \alpha_i) =& X^n + \left(\sum_{1 \leq i \leq n} \alpha_i \right) X^{n-1} + \left(\sum_{1 \leq i_1 < i_2 \leq n} \alpha_{i_1} \cdot \alpha_{i_2} \right) X^{n-2} + \ldots \\ &+ \left(\sum_{1 \leq i_1 < i_2 < \ldots < i_k \leq n} \alpha_{i_1} \cdot \alpha_{i_2} \cdots \alpha_{i_k} \right) X^{n-k} + \ldots + \prod_{1 \leq i \leq n} \alpha_i \end{aligned}$$

we have,

$$\begin{aligned}
|a_1| &\leq n(u \cdot A) \\
|a_2| &\leq \binom{n}{2}(u \cdot A)^2 \\
&\vdots \\
|a_i| &\leq \binom{n}{i}(u \cdot A)^i \implies A \geq \left(\frac{a_i}{\binom{n}{i}u^i}\right)^{\frac{1}{i}}
\end{aligned}$$

If $u^i\binom{n}{i} > 1$ for all $i \in [n]$ then it contradicts the definition of A. Hence, if we can choose $u = \frac{1}{n}$, we have $u^i\binom{n}{i} \leq \frac{1}{n^i} \cdot n^i = 1$.

Thus there exists a root α of P such that $|\alpha| \geq \frac{1}{n} \cdot A$. □

This claim implies that the quantity A is a *good* estimate of the absolute value of the largest root of a polynomial P.

Corollary 3. *Given a polynomial $P(X) = X^n + a_1X^{n-1} + a_2X^{n-2} + \ldots + a_n$, we can estimate the absolute value of the largest root of $P(X)$ upto a multiplicative factor of $2n$. The estimate is given by the quantity $A = \max_i |a_i|^{\frac{1}{i}}$ which can be computed in time polynomial in the input size.*

Above corollary can be turned into an algorithm to solve $Q4$ with $k = 2n$ (r and s are the upper and lower bound from Claim 2). Thus using the reductions(1), given a point, we can estimate a distance of a root of $P(X)$ closest to it within a multiplicative error of $2n$, in time polynomial in the input size and n.

2.1 The Algorithm

Given a polynomial $P(X)$, the following algorithm (Algorithm A 2) finds a point such that there is a root of a polynomial at at a distance at most $\frac{1}{2^k}$ from that point.

2.2 Analysis

In this section we analyze algorithm A 2.

Lemma 4. *In iteration i, we are estimating the distance of a root of $P(X)$ from a point which is at a distance at most $\frac{B_0}{2^i \cdot 8n^2}$ from it.*

Proof. It is clear from the algorithm that the size of B_i is $\frac{B_0}{2^i}$. We will show that in every iteration, there is a root of $P(X)$ inside the region R_i. This suffices to prove the claim since, at iteration i, we are dividing the region R_i into $8n^2 \times 8n^2$ grid, there is a square which contains a root and the distance of the center point of the square from that root is at most $\frac{B_i}{8n^2} = \frac{B_0}{2^i \cdot 8n^2}$.

We will prove the existence of a root inside the region R_i using induction on i. The case for $i = 0$ is true by the bound on B_0 and Claim 2. Suppose there is a root inside region R_i. We want to

```
Input : A polynomial $P(X) = X^n + a_1 X^{n-1} + a_2 X^{n-2} + \ldots + a_n \in \mathbb{C}[X]$ of degree $n$ and error parameter $k$
Output : $\beta \in \mathbb{C}$ such that $|\beta - \alpha| \leq 2^{-k}$, where $\alpha$ is the root of $P(X)$.

1. Let $B_0 = 2 \cdot \max_i \left( |a_i|^{\frac{1}{i}} \right), m = \log(B_0 \cdot 2^k)$.
2. Define the region $R_0 = \{z : |z| \leq B_0\}$.
3. Repeat for $i = 0$ to $m$:
    (a) Divide the region $R_i$ into $8n^2 \times 8n^2$ grid.
    (b) For each center point of a square in a grid, estimate the distance of a root of $P(X)$ closest to it (with a multiplicative error of $2n$).
    (c) Amongst all the center point, let $c_i$ be a point where the estimate is the minimum.
    (d) Set $B_{i+1} = \frac{B_i}{2}$. Set $R_{i+1} = \{z : |c_i - z| \leq B_{i+1}\}$.
4. Output the point $c_m$ from the last iteration.
```

A 2: Algorithm to find an approximate root of a polynomial

show that there exists a root inside region R_{i+1}. Consider the i^{th} iteration and a $8n^2 \times 8n^2$ grid on R_i. We know that there is a square which contains a root. Let c be the center point of the square. The actual distance of the root from c is at most $\frac{B_i}{8n^2}$. Hence the estimate of this distance is at most $\frac{B_i}{4n}$. Since c_i is a center point with minimum value of estimate (the estimate is at most $\frac{B_i}{4n}$), its distance from root is at most $\frac{B_i}{2}$. The region R_{i+1} is all points at a distance at most $\frac{B_i}{2}$ from c_i. Hence it contains a root. □

Lemma 5 **(Correctness).** *c_m is at a distance of at most $\frac{1}{2^k}$ from a root of $P(X)$.*

Proof. From the above claim and the guarantee on the estimation algorithm, in the last iteration, the minimum distance that we get is at most $\frac{2n \cdot B_0}{2^m \cdot 8n^2} = \frac{B_0}{2^m \cdot 4n}$. Suppose this claim is not true - c_m is at a distance greater than $\frac{1}{2^k}$ form a root closest to it. The estimate algorithm will estimate the distance to be at least $\frac{1}{2n \cdot 2^k}$ (recall the definition of multiplicative error). Plugging the value of m implies, $\frac{1}{2n \cdot 2^k} > \frac{B_0}{2^m \cdot 4n}$ which is a contradiction to the fact that the estimate from c_m is the minimum in the last iteration. □

Having shown the correctness of the algorithm, lets argue about the running time:

Running time: The parameter m is bounded by $\texttt{poly}(n \cdot m, k)$. At each iteration, we are subdividing the region into $O(n^4)$ squares. Estimating the distance from each center point in step $3(b)$ takes time $\texttt{poly}(n, m)$. Hence, each iteration takes time *poly*(n, m). Therefore, the overall running time is bounded by $\texttt{poly}(n \cdot m, k)$ as required. In particular, we get a polynomial time algorithm for $k = \texttt{poly}(n)$.