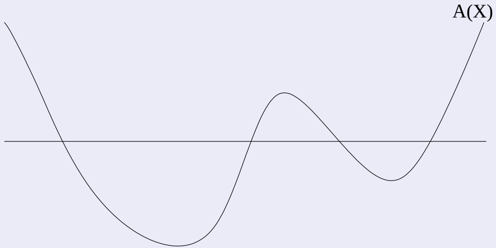
Complexity of Real Root Isolation Using Continued Fractions

Vikram Sharma

Project GALAAD
INRIA, Sophia-Antipolis

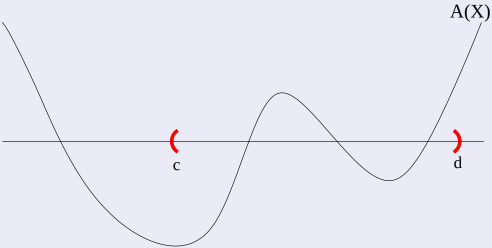
Real Root Isolation

Let $A(X)$ be a polynomial with coefficients in $\mathbb{R}$.



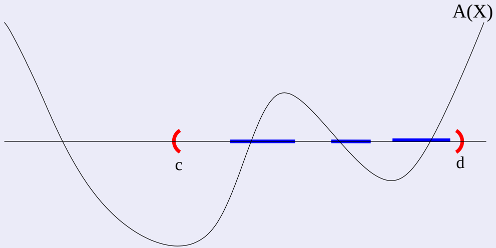
Real Root Isolation

Let $A(X)$ be a polynomial with coefficients in $\mathbb{R}$.



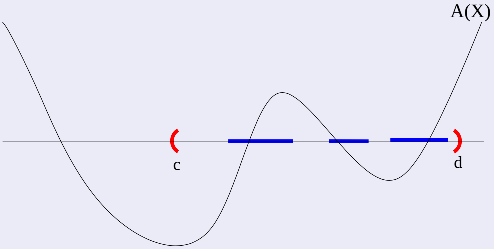
Real Root Isolation

Let $A(X)$ be a polynomial with coefficients in $\mathbb{R}$.



Real Root Isolation

Let $A(X)$ be a polynomial with coefficients in $\mathbb{R}$.

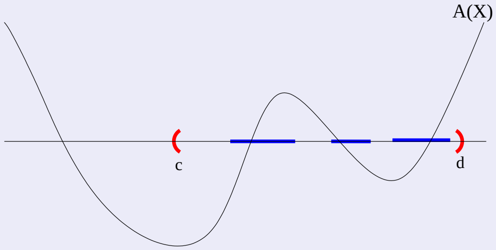


Fundamental Task

Computer Algebra, Computational Geometry, Quantifier Elimination etc.

Real Root Isolation

Let $A(X)$ be a polynomial with coefficients in $\mathbb{R}$.



Fundamental Task

Computer Algebra, Computational Geometry, Quantifier Elimination etc.

$A(X)$ is a *square-free* polynomial of degree n.

A General Subdivision Algorithm for Real Root Isolation

A General Subdivision Algorithm for Real Root Isolation

Estimate on number of real roots

- $E(A,(c,d))$ an *upper bound* on number of real roots of $A(X)$ in (c,d).
- If $E(A,(c,d)) = 1$ then there is exactly one real root of $A(X)$ in (c,d).

A General Subdivision Algorithm for Real Root Isolation

Estimate on number of real roots

- $E(A,(c,d))$ an *upper bound* on number of real roots of $A(X)$ in (c,d).
- If $E(A,(c,d)) = 1$ then there is exactly one real root of $A(X)$ in (c,d).

Input: Polynomial $A(X) \in \mathbb{R}[X]$ and $(c,d) \subseteq \mathbb{R}$.
Output: List of isolating intervals for real roots of $A(X)$ in (c,d).

A General Subdivision Algorithm for Real Root Isolation

Estimate on number of real roots

- $E(A,(c,d))$ an *upper bound* on number of real roots of $A(X)$ in (c,d).
- If $E(A,(c,d)) = 1$ then there is exactly one real root of $A(X)$ in (c,d).

Input: Polynomial $A(X) \in \mathbb{R}[X]$ and $(c,d) \subseteq \mathbb{R}$.
Output: List of isolating intervals for real roots of $A(X)$ in (c,d).

$RootIsol(A,(c,d))$

1. **If** $E(A,(c,d)) = 0$ **return**.
2. **If** $E(A,(c,d)) = 1$ **output** (c,d).
3. Partition (c,d) into two intervals I, J.
4. $RootIsol(A,I)$ and $RootIsol(A,J)$.

A General Subdivision Algorithm for Real Root Isolation

Estimate on number of real roots

- $E(A,(c,d))$ an *upper bound* on number of real roots of $A(X)$ in (c,d).
- If $E(A,(c,d)) = 1$ then there is exactly one real root of $A(X)$ in (c,d).

Input: Polynomial $A(X) \in \mathbb{R}[X]$ and $(c,d) \subseteq \mathbb{R}$.
Output: List of isolating intervals for real roots of $A(X)$ in (c,d).

$RootIsol(A,(c,d))$

1. **If** $E(A,(c,d)) = 0$ **return**.
2. **If** $E(A,(c,d)) = 1$ **output** (c,d).
3. Partition (c,d) into two intervals I, J.
4. $RootIsol(A,I)$ and $RootIsol(A,J)$.

How to implement $E(A,(c,d))$?

Two Varieties of Real Root Isolation Algorithm

Sturm Sequences

1. $E(A,(c,d))$ is computed from the Sturm sequence of $A(X), A'(X)$.
2. $E(A,(c,d)) =$ number of real roots of $A(X)$ in (c,d).

Two Varieties of Real Root Isolation Algorithm

Sturm Sequences

1. $E(A,(c,d))$ is computed from the Sturm sequence of $A(X), A'(X)$.
2. $E(A,(c,d))=$ number of real roots of $A(X)$ in (c,d).

The Descartes' Rule of Signs

1. $E(A,(c,d))=$ sign variation in the Bernstein coeffs. of $A(X)$ w.r.t. (c,d).
2. $E(A,(c,d)) \geq$ number of real roots of $A(X)$ in (c,d) by a +ve *even number*.

Two Varieties of Real Root Isolation Algorithm

Sturm Sequences

1. $E(A,(c,d))$ is computed from the Sturm sequence of $A(X), A'(X)$.
2. $E(A,(c,d)) =$ number of real roots of $A(X)$ in (c,d).

The Descartes' Rule of Signs

1. $E(A,(c,d)) =$ sign variation in the Bernstein coeffs. of $A(X)$ w.r.t. (c,d).
2. $E(A,(c,d)) \geq$ number of real roots of $A(X)$ in (c,d) by a +ve *even number*.

In practice, the second approach is more efficient than the first one.

Vincent's Theorem, 1836

Transform $A(X)$ as follows:

Vincent's Theorem, 1836

Transform $A(X)$ as follows:

$$A(X) \rightarrow X^n A\left(a_0 + \frac{1}{X}\right)$$

$a_0 \in \mathbb{N}_{\geq 0}$,

Vincent's Theorem, 1836

Transform $A(X)$ as follows:

$$A(X) \to X^n A\left(a_0 + \frac{1}{X}\right) \to (a_1 X + 1)^n A\left(a_0 + \frac{1}{a_1 + \frac{1}{X}}\right)$$

$a_0 \in \mathbb{N}_{\geq 0}$, $a_1 \in \mathbb{N}_{>0}$,

Vincent's Theorem, 1836

Transform $A(X)$ as follows:

$$A(X) \to X^n A\left(a_0 + \frac{1}{X}\right) \to (a_1 X + 1)^n A\left(a_0 + \frac{1}{a_1 + \frac{1}{X}}\right)$$

$$\to (a_1 a_2 X + a_1 + 1)^n A\left(a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{X}}}\right)$$

$a_0 \in \mathbb{N}_{\geq 0}$, $a_1 \in \mathbb{N}_{>0}$, $a_2 \in \mathbb{N}_{>0}$.

Vincent's Theorem, 1836

Transform $A(X)$ as follows:

$$A(X) \rightarrow X^n A\left(a_0 + \frac{1}{X}\right) \rightarrow (a_1 X + 1)^n A\left(a_0 + \frac{1}{a_1 + \frac{1}{X}}\right)$$
$$\rightarrow (a_1 a_2 X + a_1 + 1)^n A\left(a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{X}}}\right) \rightarrow \cdots$$

$a_0 \in \mathbb{N}_{\geq 0}$, $a_1 \in \mathbb{N}_{>0}$, $a_2 \in \mathbb{N}_{>0}$.

Vincent's Theorem, 1836

Transform $A(X)$ as follows:

$$A(X) \to X^n A\left(a_0 + \frac{1}{X}\right) \to (a_1X+1)^n A\left(a_0 + \frac{1}{a_1 + \frac{1}{X}}\right)$$

$$\to (a_1a_2X + a_1 + 1)^n A\left(a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{X}}}\right) \to \cdots$$

$a_0 \in \mathbb{N}_{\geq 0}$, $a_1 \in \mathbb{N}_{>0}$, $a_2 \in \mathbb{N}_{>0}$.

Resulting polynomial has at most one sign variation in its coefficients.

Vincent's Theorem, 1836

Transform $A(X)$ as follows:

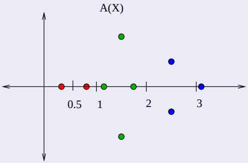
$$A(X) \to X^n A\left(a_0 + \frac{1}{X}\right) \to (a_1 X + 1)^n A\left(a_0 + \frac{1}{a_1 + \frac{1}{X}}\right)$$

$$\to (a_1 a_2 X + a_1 + 1)^n A\left(a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{X}}}\right) \to \cdots$$

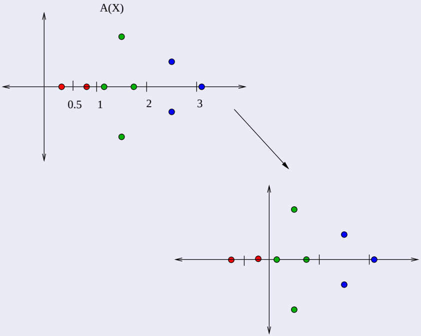
$a_0 \in \mathbb{N}_{\geq 0}$, $a_1 \in \mathbb{N}_{>0}$, $a_2 \in \mathbb{N}_{>0}$.

Resulting polynomial has at most one sign variation in its coefficients.

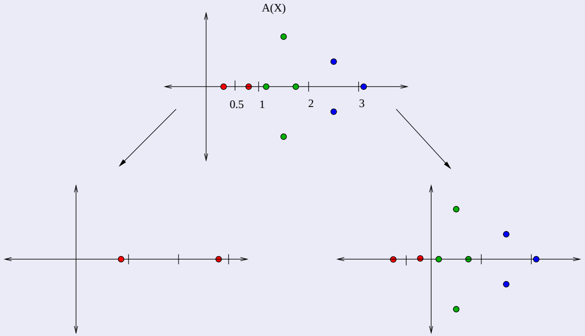
Let $\mathrm{Var}(A)$ be the number of sign variations in the coefficients of $A(X)$.



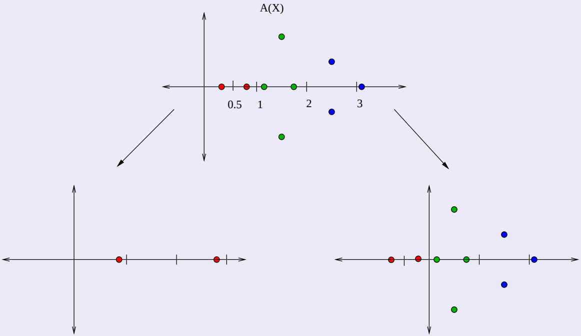
We want to isolate the positive roots of $A(X)$.



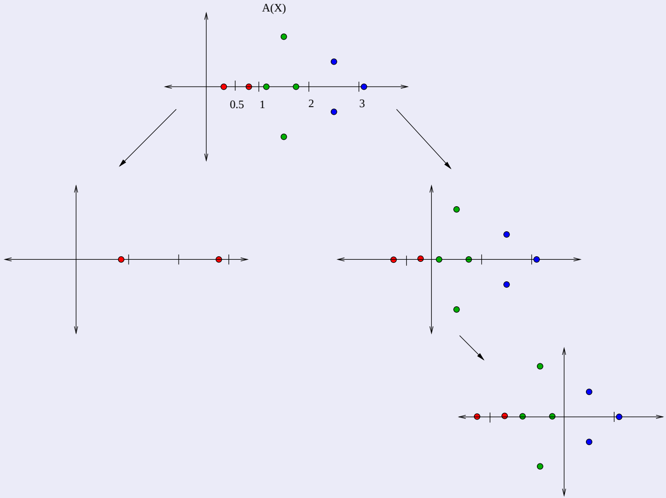
Construct $A_R(X) := A(X+1)$, $M_R(X) := X+1$. Check if $\texttt{Var}(A_R)$ is 0 or 1.



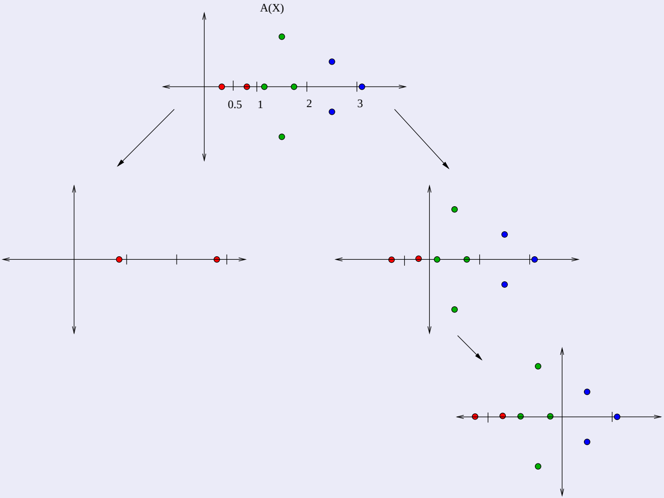
Construct $A_L(X) := (X+1)^n A\left(\frac{1}{X+1}\right)$, $M_L(X) := (X+1)^{-1}$.



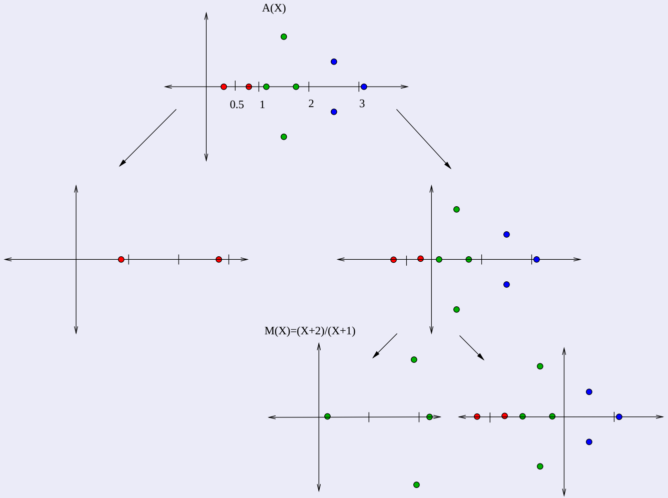
Check if $\mathtt{Var}(A_L)$ is 0 or 1.



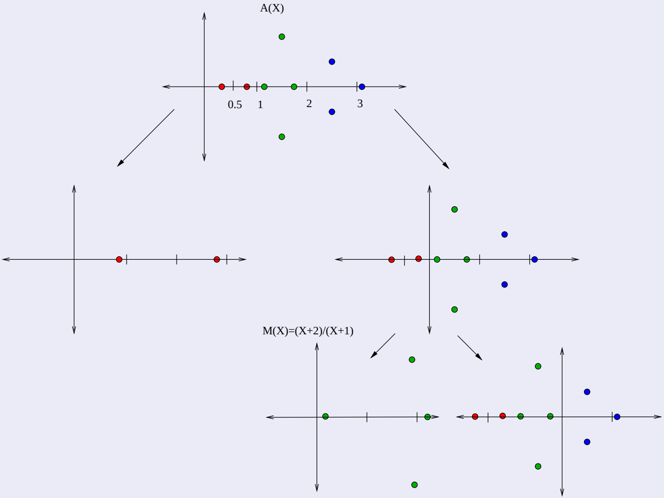
Construct $A_{RR}(X) := A_R(X+1) = A(X+2)$, $M_{RR}(X) := X+2$.



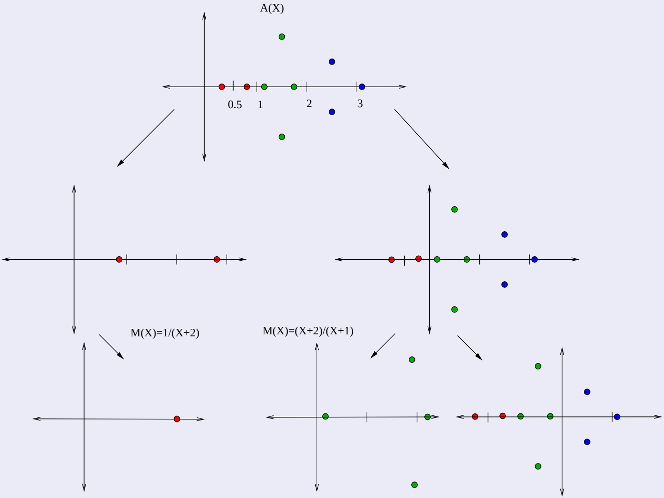
Check if $\mathtt{Var}(A_{RR})$ is 0 or 1.



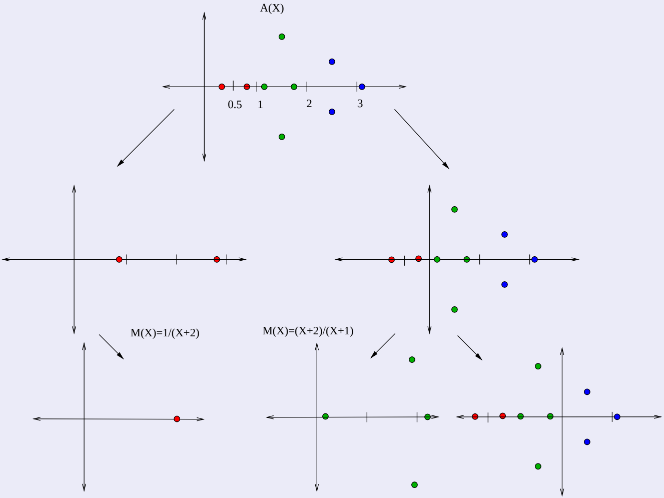
Construct $A_{RL}(X) := A_R(\frac{1}{1+X}) = A(1 + \frac{1}{1+X})$ and $M_{RL}(X) := 1 + \frac{1}{X+1}$.



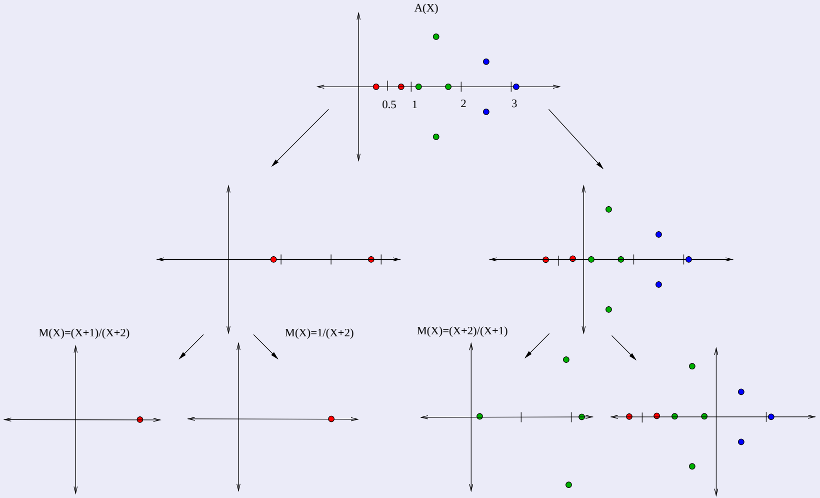
Check $\texttt{Var}(A_{RL})$.



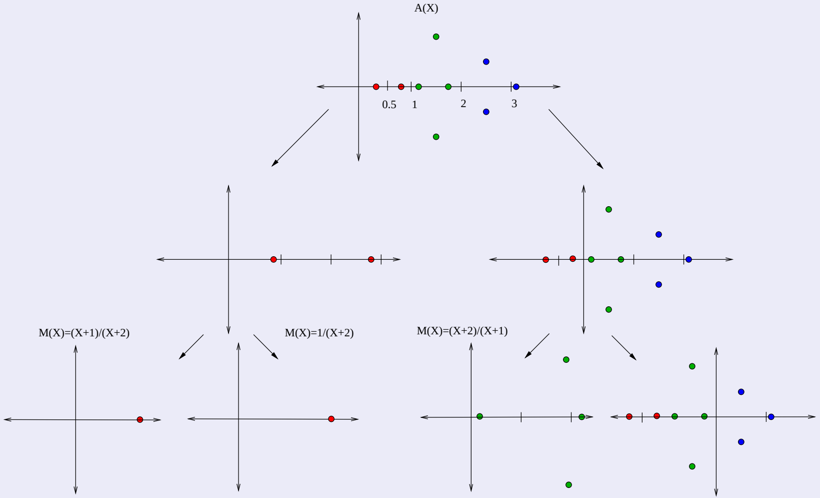
$$A_{LR}(X) = A_L(X+1) = (X+2)^n A\left(1 + \frac{1}{2+X}\right), M_{LR}(X) := (X+2)^{-1}$$



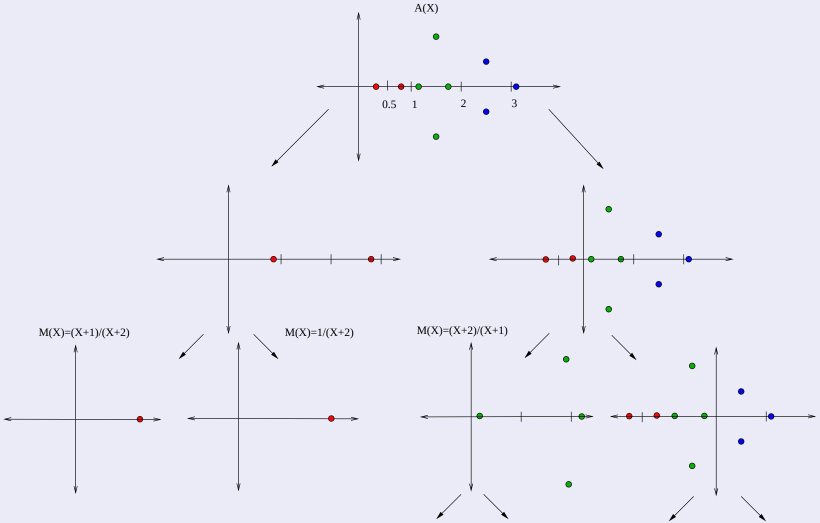
$\texttt{Var}(A_{LR}) = 1$, return $M_{LR}(0) = \frac{1}{2}$, $M_{LR}(\infty) = 0$



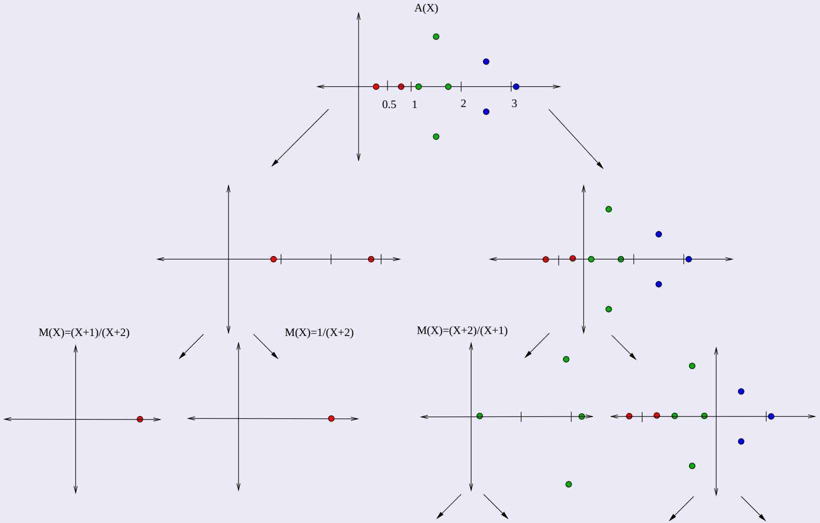
$$A_{RL}(X) := (X+1)^n A_L\left(\frac{1}{1+X}\right) = (X+2)^n A\left(\frac{1}{1+\frac{1}{1+X}}\right),\ M_{LR}(X) := \frac{X+1}{X+2}$$



$$\texttt{Var}(A_{RL}) = 1, \text{ return } M_{RL}(0) = \frac{1}{2}, M_{RL}(\infty) = 1$$

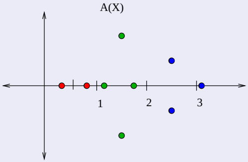


Continue recursively at each level



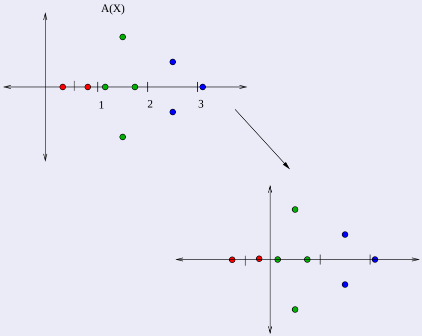
This was Uspensky's algorithm [Uspensky, 1948].

Vincent's Algorithm for Isolating Positive Roots



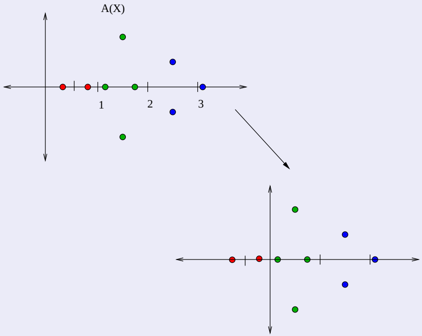
We want to isolate the positive roots of $A(X)$.

Vincent's Algorithm for Isolating Positive Roots



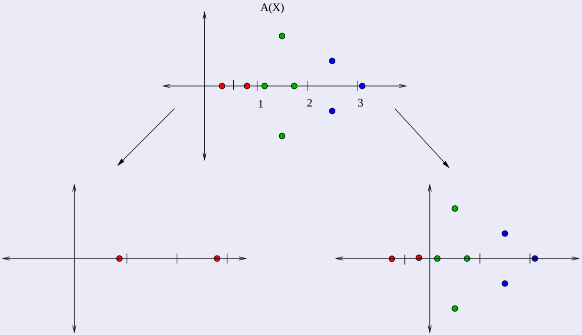
Construct $A_R(X) := A(X+1)$, $M_R(X) := X+1$ and check if $\mathrm{Var}(A_R)$ is 0 or 1.

Vincent's Algorithm for Isolating Positive Roots



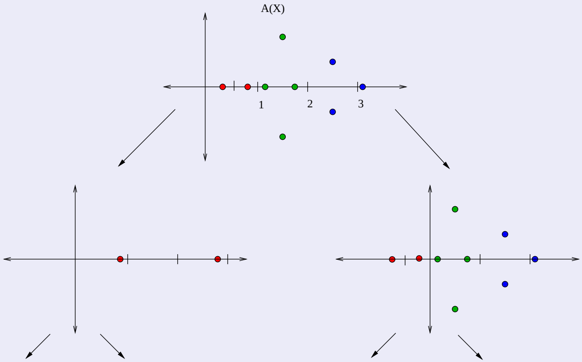
Is $\texttt{Var}(A_R) < \texttt{Var}(A)$?

Vincent's Algorithm for Isolating Positive Roots



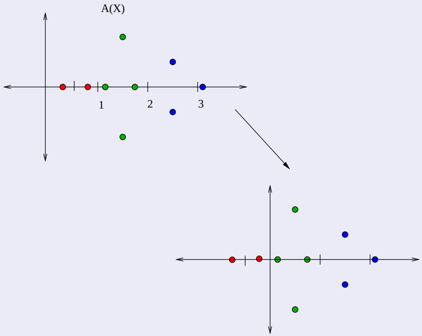
Construct $A_L(X) := (X+1)^n A\left(\frac{1}{X+1}\right)$, $M_L(X) := (X+1)^{-1}$.

Vincent's Algorithm for Isolating Positive Roots



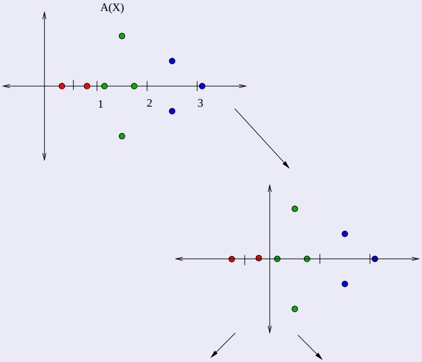
Continue recursively at each level

Vincent's Algorithm for Isolating Positive Roots



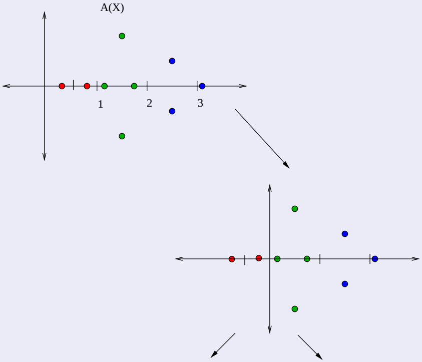
If $\texttt{Var}(A_R) = \texttt{Var}(A)$ then don't construct $A_L(X)$.

Vincent's Algorithm for Isolating Positive Roots



But proceed recursively from $A_R(X)$.

Vincent's Algorithm for Isolating Positive Roots



If $\mathtt{Var}(A_R) = \mathtt{Var}(A)$ then don't construct $A_L(X)$.

Budan-Fourier

#(roots in $(0,1)$) $\leq \mathrm{Var}(A(X)) - \mathrm{Var}(A(X+1)) = \mathrm{Var}(A) - \mathrm{Var}(A_R)$.

Drawback of Vincent's Algorithm

Exponential running time

- Consider the polynomial $A(X) = (X - 2^L)(X - 2^L - 1)$.

Drawback of Vincent's Algorithm

Exponential running time

- Consider the polynomial $A(X) = (X - 2^L)(X - 2^L - 1)$.
- At depth i on the right most path the polynomial is
$$A(X+i) = (X - (2^L - i))(X - (2^L + 1 - i)).$$

Drawback of Vincent's Algorithm

Exponential running time

- Consider the polynomial $A(X) = (X - 2^L)(X - 2^L - 1)$.
- At depth i on the right most path the polynomial is
$$A(X+i) = (X - (2^L - i))(X - (2^L + 1 - i)).$$
- To get the smallest positive root of $A(X)$ in the unit interval $i \geq 2^L$.

Drawback of Vincent's Algorithm

Exponential running time

- Consider the polynomial $A(X) = (X - 2^L)(X - 2^L - 1)$.
- At depth i on the right most path the polynomial is
$$A(X+i) = (X - (2^L - i))(X - (2^L + 1 - i)).$$
- To get the smallest positive root of $A(X)$ in the unit interval $i \geq 2^L$.

Two Solutions

- [Collins/Akritas,1976]: Bisect the interval at each recursion level.

Drawback of Vincent's Algorithm

Exponential running time

- Consider the polynomial $A(X) = (X - 2^L)(X - 2^L - 1)$.
- At depth i on the right most path the polynomial is
$$A(X+i) = (X - (2^L - i))(X - (2^L + 1 - i)).$$
- To get the smallest positive root of $A(X)$ in the unit interval $i \geq 2^L$.

Two Solutions

- [Collins/Akritas,1976]: Bisect the interval at each recursion level.
- [Akritas,1978]: Can we do better than shifts by unit length?
  Idea: Use a lower bound on the smallest positive root.

Drawback of Vincent's Algorithm

Exponential running time

- Consider the polynomial $A(X) = (X - 2^L)(X - 2^L - 1)$.
- At depth i on the right most path the polynomial is
$$A(X+i) = (X - (2^L - i))(X - (2^L + 1 - i)).$$
- To get the smallest positive root of $A(X)$ in the unit interval $i \geq 2^L$.

Two Solutions

- [Collins/Akritas,1976]: Bisect the interval at each recursion level.
- [Akritas,1978]: Can we do better than shifts by unit length?
Idea: Use a lower bound on the smallest positive root.

Advantages of Akritas' approach

- Faster in practice.
- Utilises distribution of roots.
- Computes the continued fraction approximation of the roots.

Akritas' Algorithm

Input: Polynomial $A(X)$ of degree n whose coefficients are real numbers.
Output: List of isolating intervals for the positive roots of $A(X)$.

Akritas' Algorithm

Input: Polynomial $A(X)$ of degree n whose coefficients are real numbers.
Output: List of isolating intervals for the positive roots of $A(X)$.

$RootIsol(A, M)$

- If $\texttt{Var}(A) = 0$ **return**.

Akritas' Algorithm

Input: Polynomial $A(X)$ of degree n whose coefficients are real numbers.
Output: List of isolating intervals for the positive roots of $A(X)$.

RootIsol(A, M)

- If $\mathtt{Var}(A) = 0$ **return**.
- If $\mathtt{Var}(A) = 1$ **output** the interval with end points $M(0)$, $M(\infty)$.

Akritas' Algorithm

Input: Polynomial $A(X)$ of degree n whose coefficients are real numbers.
Output: List of isolating intervals for the positive roots of $A(X)$.

$RootIsol(A, M)$

- If $\mathtt{Var}(A) = 0$ **return**.
- If $\mathtt{Var}(A) = 1$ **output** the interval with end points $M(0)$, $M(\infty)$.
- Compute a lower bound B on the positive roots of $A(X)$.

Akritas' Algorithm

Input: Polynomial $A(X)$ of degree n whose coefficients are real numbers.
Output: List of isolating intervals for the positive roots of $A(X)$.

$RootIsol(A, M)$

- If $\mathtt{Var}(A) = 0$ **return**.
- If $\mathtt{Var}(A) = 1$ **output** the interval with end points $M(0)$, $M(\infty)$.
- Compute a lower bound B on the positive roots of $A(X)$.
- If $B \geq 1$ then $A(X) := A(X+B)$, $M(X) := M(X+B)$.

Akritas' Algorithm

Input: Polynomial $A(X)$ of degree n whose coefficients are real numbers.
Output: List of isolating intervals for the positive roots of $A(X)$.

$RootIsol(A, M)$

- If $\texttt{Var}(A) = 0$ **return**.
- If $\texttt{Var}(A) = 1$ **output** the interval with end points $M(0)$, $M(\infty)$.
- Compute a lower bound B on the positive roots of $A(X)$.
- If $B \geq 1$ then $A(X) := A(X+B)$, $M(X) := M(X+B)$.
- Compute $A_R(X) := A(X+1)$ and $M_R(X) := M(X+1)$.

Akritas' Algorithm

Input: Polynomial $A(X)$ of degree n whose coefficients are real numbers.
Output: List of isolating intervals for the positive roots of $A(X)$.

$RootIsol(A, M)$

- If $\mathtt{Var}(A) = 0$ **return**.
- If $\mathtt{Var}(A) = 1$ **output** the interval with end points $M(0)$, $M(\infty)$.
- Compute a lower bound B on the positive roots of $A(X)$.
- If $B \geq 1$ then $A(X) := A(X+B)$, $M(X) := M(X+B)$.
- Compute $A_R(X) := A(X+1)$ and $M_R(X) := M(X+1)$.
- If $\mathrm{Var}(A_R) < \mathrm{Var}(A)$ then $A_L(X) := (X+1)^n A(\frac{1}{X+1})$, $M_L(X) := M(\frac{1}{X+1})$.

Akritas' Algorithm

Input: Polynomial $A(X)$ of degree n whose coefficients are real numbers.
Output: List of isolating intervals for the positive roots of $A(X)$.

$RootIsol(A, M)$

- If $\mathtt{Var}(A) = 0$ **return**.
- If $\mathtt{Var}(A) = 1$ **output** the interval with end points $M(0)$, $M(\infty)$.
- Compute a lower bound B on the positive roots of $A(X)$.
- If $B \geq 1$ then $A(X) := A(X+B)$, $M(X) := M(X+B)$.
- Compute $A_R(X) := A(X+1)$ and $M_R(X) := M(X+1)$.
- If $\mathrm{Var}(A_R) < \mathrm{Var}(A)$ then $A_L(X) := (X+1)^n A(\frac{1}{X+1})$, $M_L(X) := M(\frac{1}{X+1})$.
- $RootIsol(A_R, M_R)$ and $RootIsol(A_L, M_L)$.

Worst case bit-complexity of Akritas' algorithm?

Worst case bit-complexity of Akritas' algorithm?

Two steps for getting the worst-case bounds

1. Bound the worst-case size of the recursion tree:
   - number of inversion transformations, $X \rightarrow (X+1)^{-1}$ and
   - number of Taylor shifts.
2. Bound the worst-case complexity of a node in the recursion tree.

Worst case bit-complexity of Akritas' algorithm?

Two steps for getting the worst-case bounds

1. Bound the worst-case size of the recursion tree:
   - number of inversion transformations, $X \to (X+1)^{-1}$ and
   - number of Taylor shifts.
2. Bound the worst-case complexity of a node in the recursion tree.

Akritas' worst case bit-complexity

For $A(X) \in \mathbb{Z}[X]$, degree n, coefficients of bit-length L– $\widetilde{O}(n^4L^2)$:

- number of inversion transformations and Taylor shifts – $\widetilde{O}(n^2L)$.
- worst case bit-complexity of a node using fast integer arithmetic – $\widetilde{O}(n^2L)$.

Worst case bit-complexity of Akritas' algorithm?

Two steps for getting the worst-case bounds

1. Bound the worst-case size of the recursion tree:
   - number of inversion transformations, $X \rightarrow (X+1)^{-1}$ and
   - number of Taylor shifts.
2. Bound the worst-case complexity of a node in the recursion tree.

Akritas' worst case bit-complexity

For $A(X) \in \mathbb{Z}[X]$, degree n, coefficients of bit-length L– $\widetilde{O}(n^4L^2)$:

- number of inversion transformations and Taylor shifts – $\widetilde{O}(n^2L)$.
- worst case bit-complexity of a node using fast integer arithmetic – $\widetilde{O}(n^2L)$.

Drawbacks

- Assumes floor of the smallest positive root can be computed in $O(1)$.
- Assumes Taylor shifts don't increase the bit-size.

Worst case bit-complexity of Akritas' algorithm?

Two steps for getting the worst-case bounds

1. Bound the worst-case size of the recursion tree:
   - number of inversion transformations, $X \to (X+1)^{-1}$ and
   - number of Taylor shifts.
2. Bound the worst-case complexity of a node in the recursion tree.

Akritas' worst case bit-complexity

For $A(X) \in \mathbb{Z}[X]$, degree n, coefficients of bit-length L– $\widetilde{O}(n^4 L^2)$:

- number of inversion transformations and Taylor shifts – $\widetilde{O}(n^2 L)$.
- worst case bit-complexity of a node using fast integer arithmetic – $\widetilde{O}(n^2 L)$.

Our worst case bit-complexity

Worst case bit-complexity is $\widetilde{O}(n^7 L^2)$:

- number of inversion transformations $\widetilde{O}(nL)$; no. of Taylor shifts $\widetilde{O}(n^3 L)$.
- worst case bit-complexity of a node using fast integer arithmetic

Akritas' Algorithm

$RootIsol(A,M)$

- If $\texttt{Var}(A) = 0$ **return**.
- If $\texttt{Var}(A) = 1$ **output** the interval with end points $M(0)$, $M(\infty)$.
- Compute a lower bound B on the positive roots of $A(X)$.
- If $B \geq 1$ then $A(X) := A(X+B)$, $M(X) := M(X+B)$.
- Compute $A_R(X) := A(X+1)$ and $M_R(X) := M(X+1)$.
- If $\mathrm{Var}(A_R) < \mathrm{Var}(A)$ then $A_L(X) := (X+1)^n A(\frac{1}{X+1})$, $M_L(X) := M(\frac{1}{X+1})$.
- $RootIsol(A_R, M_R)$ and $RootIsol(A_L, M_L)$.

- What are the transformations M_R, M_L?
- What is the relation between A_R, A_L and the input polynomial?

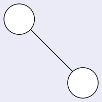
The transformation associated with a node in the tree

The transformation associated with a node in the tree

- Transformation associated with root is X.

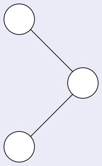
The transformation associated with a node in the tree

- Transformation associated with root is X.
- Do a Taylor shift $X \longrightarrow X + a$.



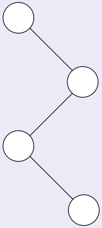
The transformation associated with a node in the tree

- Transformation associated with root is X.
- Do a Taylor shift $X \to X + a$.
- Then the transformation $X \to \frac{1}{1+X}$.



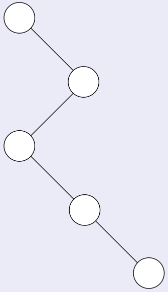
The transformation associated with a node in the tree

- Transformation associated with root is X.
- Do a Taylor shift $X \rightarrow X + a$.
- Then the transformation $X \rightarrow \frac{1}{1+X}$.
- Then a Taylor shift $X \rightarrow X + b_0$.



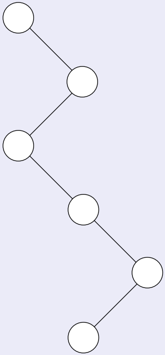
The transformation associated with a node in the tree

- Transformation associated with root is X.
- Do a Taylor shift $X \rightarrow X + a$.
- Then the transformation $X \rightarrow \frac{1}{1+X}$.
- Then a Taylor shift $X \rightarrow X + b_0$.
- Again a Taylor shift by $X \rightarrow X + b_1$.



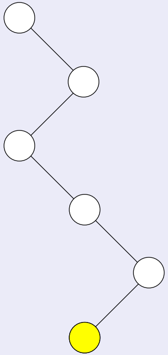
The transformation associated with a node in the tree

- Transformation associated with root is X.
- Do a Taylor shift $X \to X + a$.
- Then the transformation $X \to \frac{1}{1+X}$.
- Then a Taylor shift $X \to X + b_0$.
- Again a Taylor shift by $X \to X + b_1$.
- Followed by $X \to \frac{1}{1+X}$.



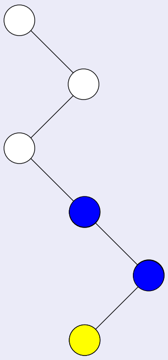
The transformation associated with a node in the tree

- Transformation associated with root is X.
- Do a Taylor shift $X \to X + a$.
- Then the transformation $X \to \frac{1}{1+X}$.
- Then a Taylor shift $X \to X + b_0$.
- Again a Taylor shift by $X \to X + b_1$.
- Followed by $X \to \frac{1}{1+X}$.
- Associated transformation is
$a + \frac{1}{1+b_0+b_1+\frac{1}{1+X}}$



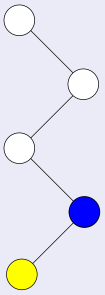
The transformation associated with a node in the tree

- Transformation associated with root is X.
- Do a Taylor shift $X \to X + a$.
- Then the transformation $X \to \frac{1}{1+X}$.
- Then a Taylor shift $X \to X + b_0$.
- Again a Taylor shift by $X \to X + b_1$.
- Followed by $X \to \frac{1}{1+X}$.
- Associated transformation is
  $a + \frac{1}{1+b_0+b_1+\frac{1}{1+X}}$
- Same as $a + \frac{1}{q+\frac{1}{1+X}}$, $q = 1 + b_0 + b_1$.



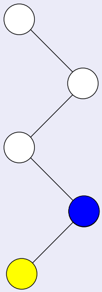
The transformation associated with a node in the tree

- Transformation associated with root is X.
- Do a Taylor shift $X \to X + a$.
- Then the transformation $X \to \frac{1}{1+X}$.
- Then a Taylor shift $X \to X + b_0$.
- Again a Taylor shift by $X \to X + b_1$.
- Followed by $X \to \frac{1}{1+X}$.
- Associated transformation is
  $a + \frac{1}{1+b_0+b_1+\frac{1}{1+X}}$
- Same as $a + \frac{1}{q+\frac{1}{1+X}}$, $q = 1 + b_0 + b_1$.
- Collapse consecutive Taylor shifts into one.



The transformation associated with a node in the tree

- Transformation associated with root is X.
- Do a Taylor shift $X \to X + a$.
- Then the transformation $X \to \frac{1}{1+X}$.
- Then a Taylor shift $X \to X + b_0$.
- Again a Taylor shift by $X \to X + b_1$.
- Followed by $X \to \frac{1}{1+X}$.
- Associated transformation is
  $a + \frac{1}{1+b_0+b_1+\frac{1}{1+X}}$
- Same as $a + \frac{1}{q+\frac{1}{1+X}}$, $q = 1 + b_0 + b_1$.
- Collapse consecutive Taylor shifts into one.



What is the transformation in general?

The transformation associated with a node in the tree

$$q_0 + \cfrac{1}{q_1 + \cfrac{1}{q_2 + \cfrac{1}{\ddots + \cfrac{1}{q_m + X}}}}$$

where

- m is the number of inversion transformations ($X \to \frac{1}{1+X}$).
- $q_0 \geq 0$ the total amount of Taylor shifts to the first inversion transformation.
- $q_i \geq 1$, for $i = 1, \ldots, m-1$, the total amount of Taylor shifts between i-th and $i+1$-th inversion transformation; if there are no Taylor shifts $q_i = 1$.

The transformation associated with a node in the tree

$$q_0 + \cfrac{1}{q_1 + \cfrac{1}{q_2 + \cfrac{1}{\ddots + \cfrac{1}{qm + X}}}}$$

where

- m is the number of inversion transformations ($X \rightarrow \frac{1}{1+X}$).
- $q_0 \geq 0$ the total amount of Taylor shifts to the first inversion transformation.
- $q_i \geq 1$, for $i = 1, \ldots, m-1$, the total amount of Taylor shifts between i-th and $i+1$-th inversion transformation; if there are no Taylor shifts $q_i = 1$.

Let ith quotient $\frac{P_i}{Q_i}$ be the finite continued fraction $q_0 + \cfrac{1}{q_1 + \cfrac{1}{\ddots + \frac{1}{q_i}}}$.

Then $P_i = q_i P_{i-1} + P_{i-2}$ and $Q_i = q_i Q_{i-1} + Q_{i-2}$.

The transformation associated with a node in the tree

$$q_0 + \cfrac{1}{q_1 + \cfrac{1}{q_2 + \cfrac{1}{\ddots + \cfrac{1}{q_m + X}}}} = \frac{P_m X + P_{m-1}}{Q_m X + Q_{m-1}}.$$

where

- m is the number of inversion transformations ($X \to \frac{1}{1+X}$).
- $q_0 \geq 0$ the total amount of Taylor shifts to the first inversion transformation.
- $q_i \geq 1$, for $i = 1, \ldots, m-1$, the total amount of Taylor shifts between i-th and $i+1$-th inversion transformation; if there are no Taylor shifts $q_i = 1$.

Let ith quotient $\frac{P_i}{Q_i}$ be the finite continued fraction $q_0 + \cfrac{1}{q_1 + \cfrac{1}{\ddots + \cfrac{1}{q_i}}}$.

Then $P_i = q_i P_{i-1} + P_{i-2}$ and $Q_i = q_i Q_{i-1} + Q_{i-2}$.

Two additional features associated with a node in the tree

The transformation associated with a node

Let m be the number of inversion transformations along the path and

$$M(X) := \frac{P_m X + P_{m-1}}{Q_m X + Q_{m-1}}.$$

Two additional features associated with a node in the tree

The transformation associated with a node

Let m be the number of inversion transformations along the path and

$$M(X) := \frac{P_m X + P_{m-1}}{Q_m X + Q_{m-1}}.$$

The two features

- Polynomial $A_m(X) := (Q_m X + Q_{m-1})^n A(M(X))$.

Two additional features associated with a node in the tree

The transformation associated with a node

Let m be the number of inversion transformations along the path and

$$M(X) := \frac{P_m X + P_{m-1}}{Q_m X + Q_{m-1}}.$$

The two features

- Polynomial $A_m(X) := (Q_m X + Q_{m-1})^n A(M(X))$.
- Interval I_m that has end-points $M(0) = \frac{P_{m-1}}{Q_{m-1}}$, $M(\infty) = \frac{P_m}{Q_m}$.

Note: Width of I_m is $\left|\frac{P_m}{Q_m} - \frac{P_{m-1}}{Q_{m-1}}\right| = (Q_m Q_{m-1})^{-1}$.

Two additional features associated with a node in the tree

The transformation associated with a node

Let m be the number of inversion transformations along the path and

$$M(X) := \frac{P_m X + P_{m-1}}{Q_m X + Q_{m-1}}.$$

The two features

- Polynomial $A_m(X) := (Q_m X + Q_{m-1})^n A(M(X))$.
- Interval I_m that has end-points $M(0) = \frac{P_{m-1}}{Q_{m-1}}$, $M(\infty) = \frac{P_m}{Q_m}$.

Note: Width of I_m is $\left|\frac{P_m}{Q_m} - \frac{P_{m-1}}{Q_{m-1}}\right| = (Q_m Q_{m-1})^{-1}$.

The positive roots of $A_m(X) \Leftrightarrow$ Roots of $A(X)$ in I_m.

$\mathtt{Var}(A_m) =$ #(number of roots of $A(X)$ in I_m) + even number.

Two additional features associated with a node in the tree

The transformation associated with a node

Let m be the number of inversion transformations along the path and

$$M(X) := \frac{P_m X + P_{m-1}}{Q_m X + Q_{m-1}}.$$

The two features

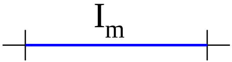
- Polynomial $A_m(X) := (Q_m X + Q_{m-1})^n A(M(X))$.
- Interval I_m that has end-points $M(0) = \frac{P_{m-1}}{Q_{m-1}}$, $M(\infty) = \frac{P_m}{Q_m}$.

Note: Width of I_m is $\left|\frac{P_m}{Q_m} - \frac{P_{m-1}}{Q_{m-1}}\right| = (Q_m Q_{m-1})^{-1}$.

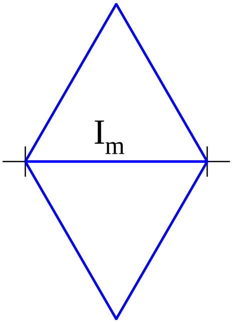
The positive roots of $A_m(X) \Leftrightarrow$ Roots of $A(X)$ in I_m.
$\mathtt{Var}(A_m) = $#(number of roots of $A(X)$ in I_m) + even number.

When does the algorithm terminate? When is $\mathrm{Var}(A_m) \leq 1$?

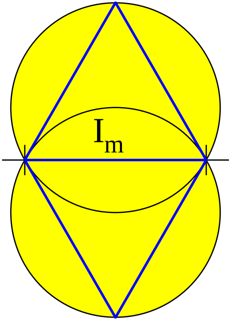
Termination Criterion: Two-Circle Theorem



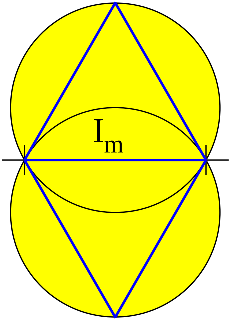
Termination Criterion: Two-Circle Theorem



Termination Criterion: Two-Circle Theorem



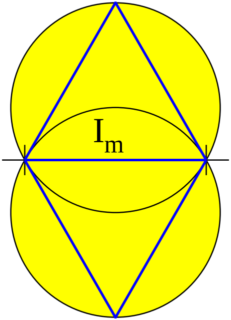
Termination Criterion: Two-Circle Theorem



Two-circle Theorem ([Ostrowski, 1950])

If the two-circles figure w.r.t. I_m *contains a single root of* $A(X)$ *then* $\mathrm{Var}(A_m) = 1$*;*
if no roots of $A(X)$ *then* $\mathrm{Var}(A_m) = 0$*.*

Termination Criterion: Two-Circle Theorem



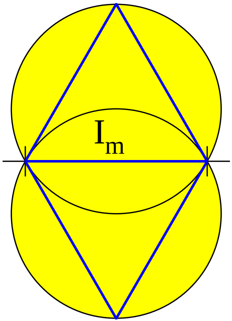
Two-circle Theorem ([Ostrowski, 1950])

If the two-circles figure w.r.t. I_m *contains a single root of* $A(X)$ *then* $\mathrm{Var}(A_m) = 1$*;*
if no roots of $A(X)$ *then* $\mathrm{Var}(A_m) = 0$*.*

Contrapositive

If $\mathrm{Var}(A_m) \geq 2$*, then the two-circles figure in* $\mathbb{C}$ *w.r.t. interval* I_m *contains two roots* α, β *of* $A(X)$*.*

Termination Criterion: Two-Circle Theorem



Two-circle Theorem ([Ostrowski, 1950])

If the two-circles figure w.r.t. I_m contains a single root of $A(X)$ then $\mathrm{Var}(A_m) = 1$; if no roots of $A(X)$ then $\mathrm{Var}(A_m) = 0$.

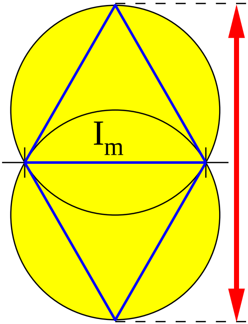
Contrapositive

If $\mathrm{Var}(A_m) \geq 2$, then the two-circles figure in $\mathbb{C}$ w.r.t. interval I_m contains two roots α, β of $A(X)$.

Corollary

We can choose a pair α, β of roots inside the two-circles such that

Termination Criterion: Two-Circle Theorem



Two-circle Theorem ([Ostrowski, 1950])

If the two-circles figure w.r.t. I_m *contains a single root of* $A(X)$ *then* $\mathrm{Var}(A_m) = 1$*; if no roots of* $A(X)$ *then* $\mathrm{Var}(A_m) = 0$*.*

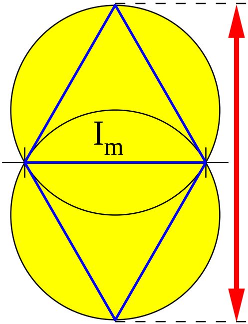
Contrapositive

If $\mathrm{Var}(A_m) \geq 2$*, then the two-circles figure in* $\mathbb{C}$ *w.r.t. interval* I_m *contains two roots* α, β *of* $A(X)$*.*

Corollary

We can choose a pair α, β *of roots inside the two-circles such that* $|\beta - \alpha| < \sqrt{3}|I_m|$

Termination Criterion: Two-Circle Theorem



Two-circle Theorem ([Ostrowski, 1950])

If the two-circles figure w.r.t. I_m contains a single root of $A(X)$ then $\mathrm{Var}(A_m) = 1$; if no roots of $A(X)$ then $\mathrm{Var}(A_m) = 0$.

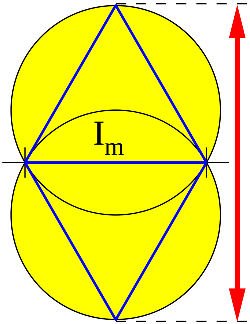
Contrapositive

If $\mathrm{Var}(A_m) \geq 2$, then the two-circles figure in $\mathbb{C}$ w.r.t. interval I_m contains two roots α, β of $A(X)$.

Corollary

We can choose a pair α, β of roots inside the two-circles such that $|\beta - \alpha| < \sqrt{3}|I_m|$, but $|I_m| = \left|\frac{P_m}{Q_m} - \frac{P_{m-1}}{Q_{m-1}}\right| = \frac{1}{Q_m Q_{m-1}}$. Thus

Termination Criterion: Two-Circle Theorem



Two-circle Theorem ([Ostrowski, 1950])

If the two-circles figure w.r.t. I_m contains a single root of $A(X)$ then $\mathrm{Var}(A_m) = 1$; if no roots of $A(X)$ then $\mathrm{Var}(A_m) = 0$.

Contrapositive

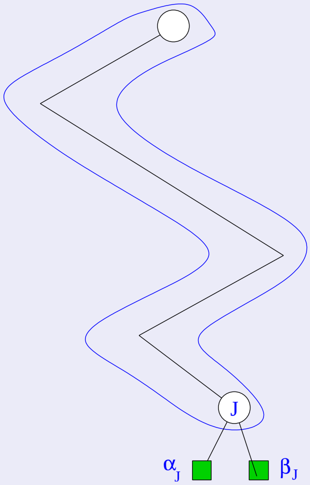
If $\mathrm{Var}(A_m) \geq 2$, then the two-circles figure in $\mathbb{C}$ w.r.t. interval I_m contains two roots α, β of $A(X)$.

Corollary

We can choose a pair α, β of roots inside the two-circles such that $|\beta - \alpha| < \sqrt{3}|I_m|$, but $|I_m| = \left|\frac{P_m}{Q_m} - \frac{P_{m-1}}{Q_{m-1}}\right| = \frac{1}{Q_m Q_{m-1}}$. Thus

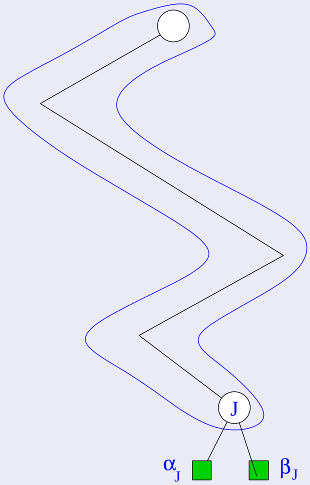
If $\mathrm{Var}(A_m) \geq 2$ then $\frac{1}{Q_m Q_{m-1}} > |\beta - \alpha|/\sqrt{3}$.

Number of Inversion Transformations along a path to a leaf



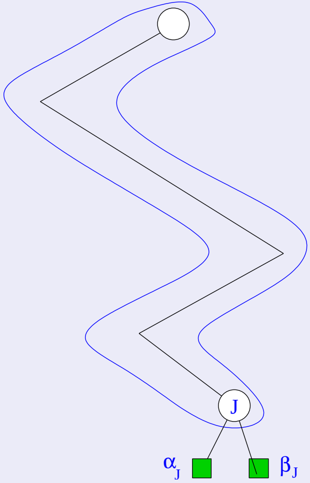
1. A path in the recursion tree of $RootIsol(A, X)$, from the root to a parent J of two leaves. Let m be the number of inversion transformations along the path.

Number of Inversion Transformations along a path to a leaf



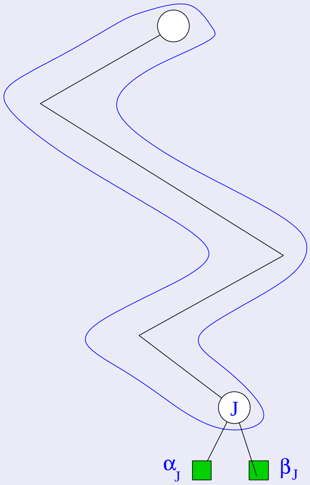
1. A path in the recursion tree of $RootIsol(A, X)$, from the root to a parent J of two leaves. Let m be the number of inversion transformations along the path.
2. Let I_m be the interval associated with J.

Number of Inversion Transformations along a path to a leaf



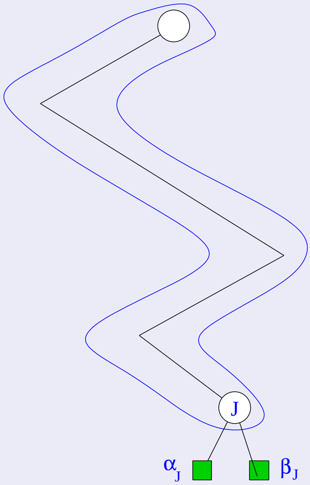
1. A path in the recursion tree of $RootIsol(A, X)$, from the root to a parent J of two leaves. Let m be the number of inversion transformations along the path.
2. Let I_m be the interval associated with J.
3. Since $\mathrm{Var}(A_m) \geq 2$, there is a pair of roots (α_J, β_J) of $A(X)$ such that $|I_m| = \frac{1}{Q_m Q_{m-1}} \geq |\beta_J - \alpha_J| / \sqrt{3}$.

Number of Inversion Transformations along a path to a leaf



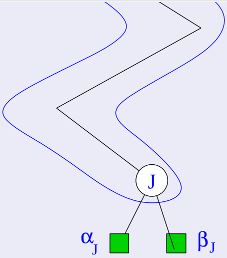
1. A path in the recursion tree of $RootIsol(A, X)$, from the root to a parent J of two leaves. Let m be the number of inversion transformations along the path.
2. Let I_m be the interval associated with J.
3. Since $\mathrm{Var}(A_m) \geq 2$, there is a pair of roots (α_J, β_J) of $A(X)$ such that $|I_m| = \frac{1}{Q_m Q_{m-1}} \geq |\beta_J - \alpha_J|/\sqrt{3}$.
4. But $Q_m = q_m Q_{m-1} + Q_{m-2} \geq Q_{m-1} + Q_{m-2} \geq F_m \geq \phi^{m-1}$.

Number of Inversion Transformations along a path to a leaf



1. A path in the recursion tree of $RootIsol(A,X)$, from the root to a parent J of two leaves. Let m be the number of inversion transformations along the path.
2. Let I_m be the interval associated with J.
3. Since $\mathrm{Var}(A_m) \geq 2$, there is a pair of roots (α_J, β_J) of $A(X)$ such that $|I_m| = \frac{1}{Q_m Q_{m-1}} \geq |\beta_J - \alpha_J|/\sqrt{3}$.
4. But $Q_m = q_m Q_{m-1} + Q_{m-2} \geq Q_{m-1} + Q_{m-2} \geq F_m \geq \phi^{m-1}$.
5. Thus $m \leq 2 - \log_\phi |\beta_J - \alpha_J|$.
6. This was shown by Uspensky and Ostrowski.

Number of Inversion Transformations along a path to a leaf



1. A path in the recursion tree of ... ent J ... th. ... J. ... oots (α_J, β_J) of $A(X)$ such that $|I_m| = \frac{1}{Q_m Q_{m-1}} \geq |\beta_J - \alpha_J|/\sqrt{3}$.
4. But $Q_m = q_m Q_{m-1} + Q_{m-2} \geq Q_{m-1} + Q_{m-2} \geq F_m \geq \phi^{m-1}$.
5. Thus $m \leq 2 - \log_\phi |\beta_J - \alpha_J|$.
6. This was shown by Uspensky and Ostrowski.

Proposition

The total number of inversion transformations in the tree are bounded by

$$\sum_J (2 - \log_\phi |\beta_J - \alpha_J|).$$

Akritas' Algorithm

$RootIsol(A,M)$

- If $\texttt{Var}(A) = 0$ **return**.
- If $\texttt{Var}(A) = 1$ **output** the interval with end points $M(0)$, $M(\infty)$.
- Compute a lower bound B on the positive roots of $A(X)$.
- If $B \geq 1$ then $A(X) := A(X+B)$, $M(X) := M(X+B)$.
- Compute $A_R(X) := A(X+1)$ and $M_R(X) := M(X+1)$.
- If $\mathrm{Var}(A_R) < \mathrm{Var}(A)$ then $A_L(X) := (X+1)^n A(\frac{1}{X+1})$, $M_L(X) := M(\frac{1}{X+1})$.
- $RootIsol(A_R, M_R)$ and $RootIsol(A_L, M_L)$.

Akritas' Algorithm

$RootIsol(A, M)$

- If $\mathtt{Var}(A) = 0$ **return**.
- If $\mathtt{Var}(A) = 1$ **output** the interval with end points $M(0)$, $M(\infty)$.
- Compute a lower bound B on the positive roots of $A(X)$.
- If $B \geq 1$ then $A(X) := A(X+B)$, $M(X) := M(X+B)$.
- Compute $A_R(X) := A(X+1)$ and $M_R(X) := M(X+1)$.
- If $\mathrm{Var}(A_R) < \mathrm{Var}(A)$ then $A_L(X) := (X+1)^n A(\frac{1}{X+1})$, $M_L(X) := M(\frac{1}{X+1})$.
- $RootIsol(A_R, M_R)$ and $RootIsol(A_L, M_L)$.

How do we compute a lower bound on positive roots of a polynomial?

Lower Bound on the smallest positive root

One Approach

- Roots of $X^n A(1/X)$ are inverse of the roots of $A(X)$.
- Compute an upper bound U on the largest positive root of $X^n A(1/X)$.
- $1/U$ is a lower bound on the smallest positive root of $A(X)$.

Lower Bound on the smallest positive root

One Approach

- Roots of $X^n A(1/X)$ are inverse of the roots of $A(X)$.
- Compute an upper bound U on the largest positive root of $X^n A(1/X)$.
- $1/U$ is a lower bound on the smallest positive root of $A(X)$.

Upper bound on positive roots, [Hong,98]

$B(X) = \sum_{i=0}^{n} b_i X^i$, $b_n > 0$. $U(B) := 2 \max_{b_i<0} \min_{b_j>0, j>i} \left\{ \left| \frac{b_i}{b_j} \right|^{1/(j-i)} \right\}$.

Lower Bound on the smallest positive root

One Approach

- Roots of $X^n A(1/X)$ are inverse of the roots of $A(X)$.
- Compute an upper bound U on the largest positive root of $X^n A(1/X)$.
- $1/U$ is a lower bound on the smallest positive root of $A(X)$.

Upper bound on positive roots, [Hong,98]

$B(X) = \sum_{i=0}^{n} b_i X^i$, $b_n > 0$. $U(B) := 2 \max_{b_i<0} \min_{b_j>0, j>i} \left\{ \left| \frac{b_i}{b_j} \right|^{1/(j-i)} \right\}$.

Tight lower bound

$$\text{Define } \mathrm{PLB}(A) := \frac{1}{U(X^n A(1/X))}.$$

Suppose $A(X)$ has only real roots in $\Re(z) > 0$ and α is the smallest positive root of $A(X)$. Then

$$\frac{\alpha}{2n} \leq \mathrm{PLB}(A) < \alpha.$$

Bound on the number of Taylor shifts along a path

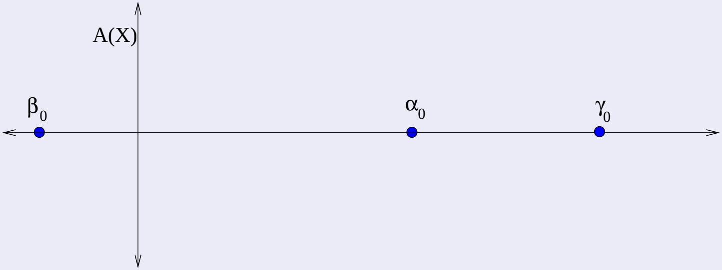
Assume the polynomial $A(X)$ has *only real roots*.

#(shifts needed to reach the floor of the smallest positive root α_0)?

Bound on the number of Taylor shifts along a path

Assume the polynomial $A(X)$ has *only real roots*.

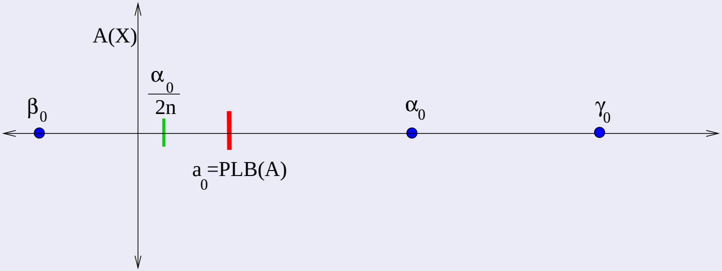
#(shifts needed to reach the floor of the smallest positive root α_0)?



Bound on the number of Taylor shifts along a path

Assume the polynomial $A(X)$ has *only real roots*.

#(shifts needed to reach the floor of the smallest positive root α_0)?

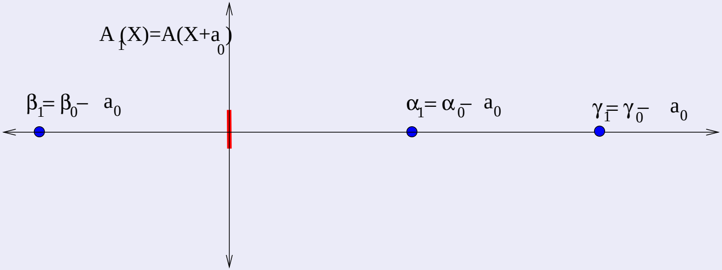


Since $\mathrm{PLB}(A)$ is a lower bound on α_0 we have $\frac{|\alpha_0|}{2n} \leq \mathrm{PLB}(A)$.

Bound on the number of Taylor shifts along a path

Assume the polynomial $A(X)$ has *only real roots*.

#(shifts needed to reach the floor of the smallest positive root α_0)?

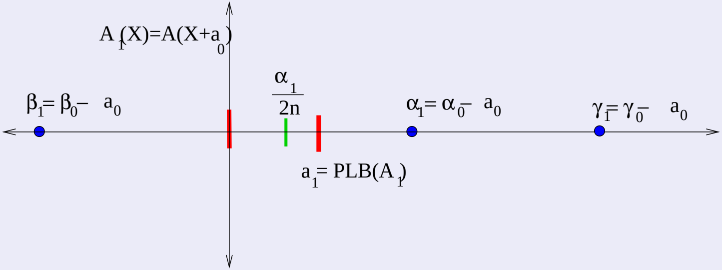


Shift $A(X)$ by $\mathrm{PLB}(A)$ to obtain $A_1(X)$.

Bound on the number of Taylor shifts along a path

Assume the polynomial $A(X)$ has *only real roots*.

#(shifts needed to reach the floor of the smallest positive root α_0)?

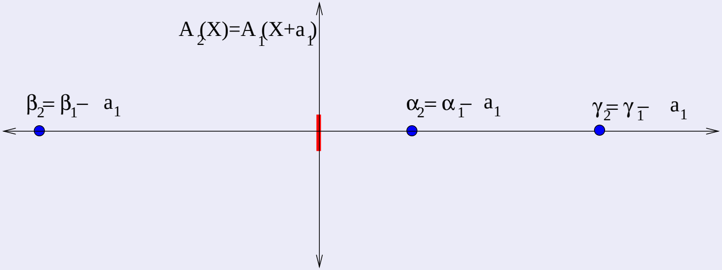


Suppose $\alpha_1 > 1$. Then $\frac{\alpha_1}{2n} \leq \mathrm{PLB}(A_1)$.

Bound on the number of Taylor shifts along a path

Assume the polynomial $A(X)$ has *only real roots*.

#(shifts needed to reach the floor of the smallest positive root α_0)?

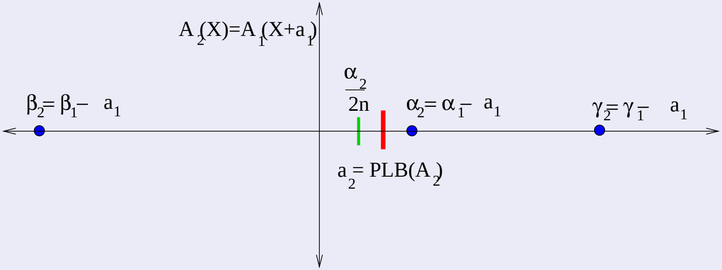


Shift $A_1(X)$ by a_1 to get $A_2(X)$.

Bound on the number of Taylor shifts along a path

Assume the polynomial $A(X)$ has *only real roots*.

#(shifts needed to reach the floor of the smallest positive root α_0)?

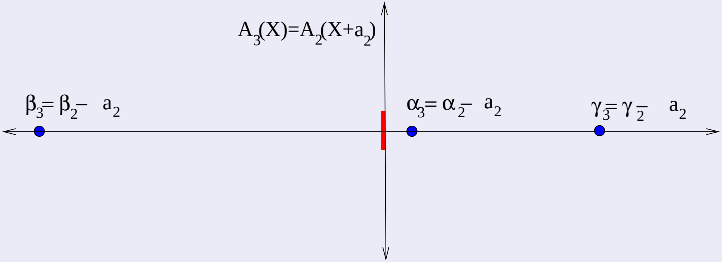


Again suppose $\alpha_2 > 1$. Then $\frac{\alpha_2}{2n} \leq \mathrm{PLB}(A_2)$.

Bound on the number of Taylor shifts along a path

Assume the polynomial $A(X)$ has *only real roots*.

#(shifts needed to reach the floor of the smallest positive root α_0)?

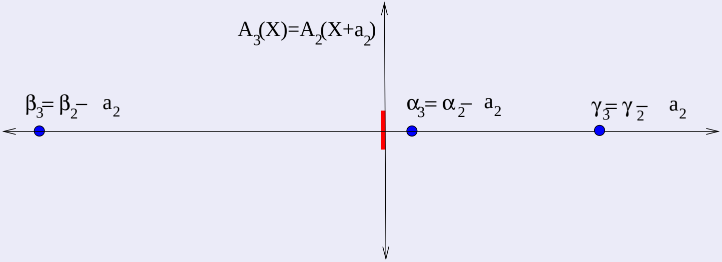


This continues until $\alpha_i < 1$, i.e., we compute the floor of α_0.

Bound on the number of Taylor shifts along a path

Assume the polynomial $A(X)$ has *only real roots*.

#(shifts needed to reach the floor of the smallest positive root α_0)?



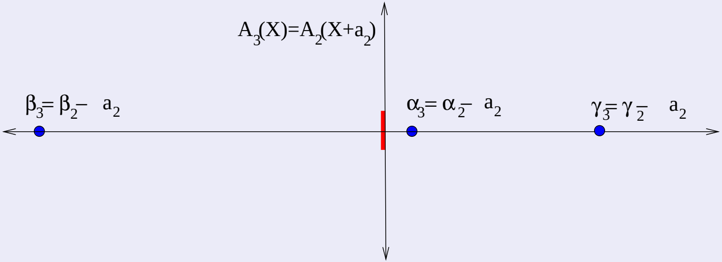
This continues until $\alpha_i < 1$, i.e., we compute the floor of α_0.

- $\alpha_i = \alpha_{i-1} - \mathrm{PLB}(A_{i-1}) \leq \alpha_{i-1}(1 - \frac{1}{2n})$.

Bound on the number of Taylor shifts along a path

Assume the polynomial $A(X)$ has *only real roots*.

#(shifts needed to reach the floor of the smallest positive root α_0)?



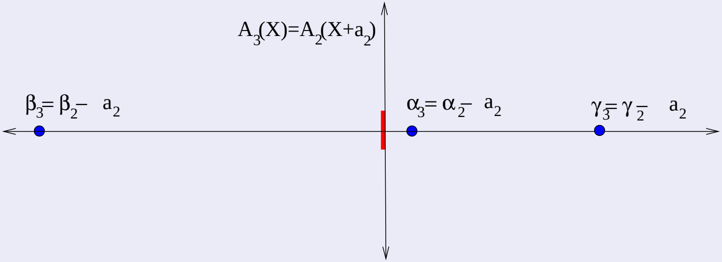
This continues until $\alpha_i < 1$, i.e., we compute the floor of α_0.

- $\alpha_i = \alpha_{i-1} - \mathrm{PLB}(A_{i-1}) \le \alpha_{i-1}(1 - \frac{1}{2n})$.
- Thus $\alpha_i \le \alpha_0(1 - \frac{1}{2n})^i$.

Bound on the number of Taylor shifts along a path

Assume the polynomial $A(X)$ has *only real roots*.

#(shifts needed to reach the floor of the smallest positive root α_0)?

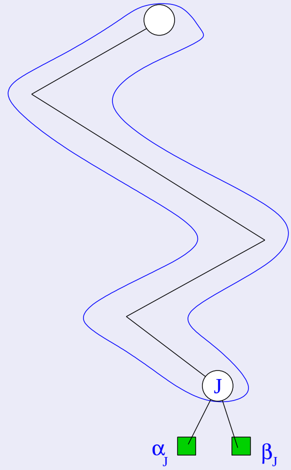


This continues until $\alpha_i < 1$, i.e., we compute the floor of α_0.

- $\alpha_i = \alpha_{i-1} - \mathrm{PLB}(A_{i-1}) \leq \alpha_{i-1}(1 - \frac{1}{2n})$.
- Thus $\alpha_i \leq \alpha_0(1 - \frac{1}{2n})^i$.
- Need at most $2n \log \alpha_0$ Taylor shifts to compute floor of α_0.

Number of Taylor shifts along a path in the tree

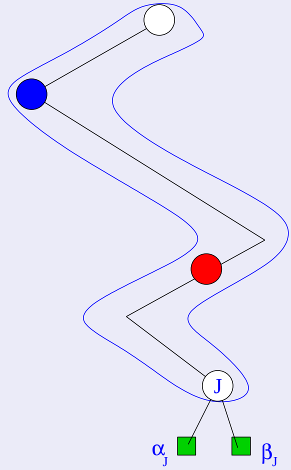
Assume the polynomial $A(X)$ has *only real roots*.



1. Consider a path in the recursion tree of $RootIsol(A, M(X))$, $M(X) = X$, from the root to a parent J of two leaves.
2. Let α_J, β_J be the roots associated with the leaves.
3. m be the number of inversion transformations along the path.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.



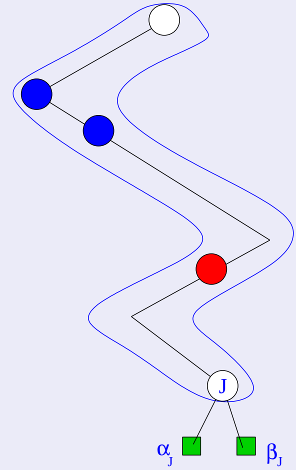
Consider the i-th and $i+1$-th inversion transformation.

$A_i(X)$ be the polynomial associated with the blue node. $a_1, \dots, a_\ell$ be its positive real roots.

#(Taylor shift) from i-th to $i+1$-th transformation is bounded by $2n(\log a_1 + \cdots + \log a_\ell) \leq 2n^2 \log a_\ell \leq 2n^2 \log q_i$.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.



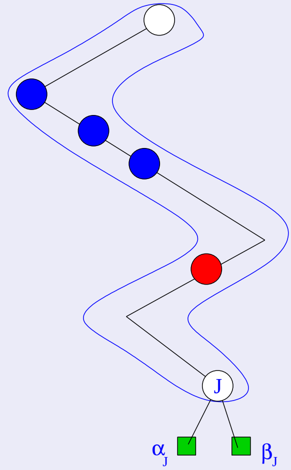
Consider the i-th and $i+1$-th inversion transformation.

$A_i(X)$ be the polynomial associated with the blue node. $a_1, \ldots, a_\ell$ be its positive real roots.

#(Taylor shift) from i-th to $i+1$-th transformation is bounded by $2n(\log a_1 + \cdots + \log a_\ell) \leq 2n^2 \log a_\ell \leq 2n^2 \log q_i$.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.



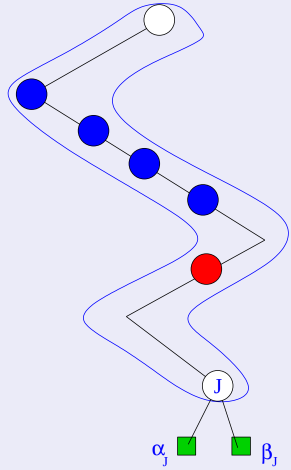
Consider the i-th and $i+1$-th inversion transformation.

$A_i(X)$ be the polynomial associated with the blue node. $a_1, \ldots, a_\ell$ be its positive real roots.

#(Taylor shift) from i-th to $i+1$-th transformation is bounded by $2n(\log a_1 + \cdots + \log a_\ell) \leq 2n^2 \log a_\ell \leq 2n^2 \log q_i$.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.



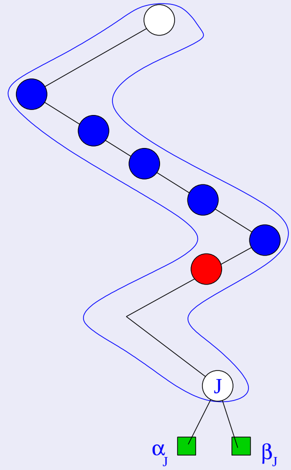
Consider the i-th and $i+1$-th inversion transformation.

$A_i(X)$ be the polynomial associated with the blue node. $a_1, \ldots, a_\ell$ be its positive real roots.

#(Taylor shift) from i-th to $i+1$-th transformation is bounded by $2n(\log a_1 + \cdots + \log a_\ell) \leq 2n^2 \log a_\ell \leq 2n^2 \log q_i$.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.



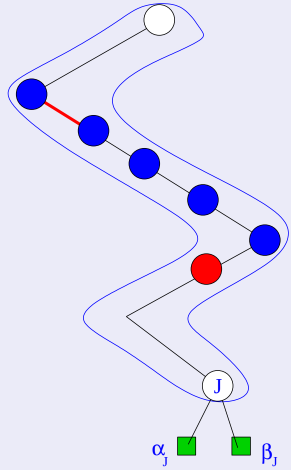
Consider the i-th and $i+1$-th inversion transformation.

$A_i(X)$ be the polynomial associated with the blue node. $a_1, \ldots, a_\ell$ be its positive real roots.

#(Taylor shift) from i-th to $i+1$-th transformation is bounded by $2n(\log a_1 + \cdots + \log a_\ell) \leq 2n^2 \log a_\ell \leq 2n^2 \log q_i$.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.



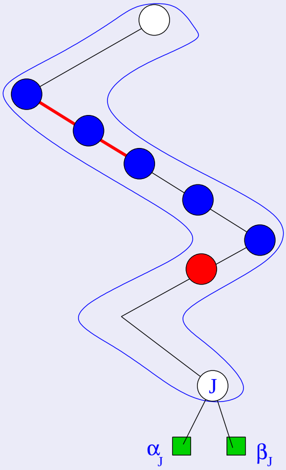
Consider the i-th and $i+1$-th inversion transformation.

$A_i(X)$ be the polynomial associated with the blue node. $a_1, \ldots, a_\ell$ be its positive real roots.

#(Taylor shift) from i-th to $i+1$-th transformation is bounded by
$2n(\log a_1 + \cdots + \log a_\ell) \leq 2n^2 \log a_\ell \leq 2n^2 \log q_i$.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.



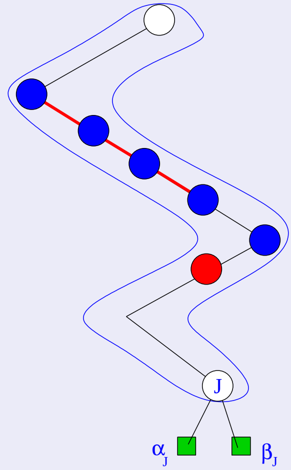
Consider the i-th and $i+1$-th inversion transformation.

$A_i(X)$ be the polynomial associated with the blue node. $a_1, \ldots, a_\ell$ be its positive real roots.

#(Taylor shift) from i-th to $i+1$-th transformation is bounded by $2n(\log a_1 + \cdots + \log a_\ell) \leq 2n^2 \log a_\ell \leq 2n^2 \log q_i$.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.



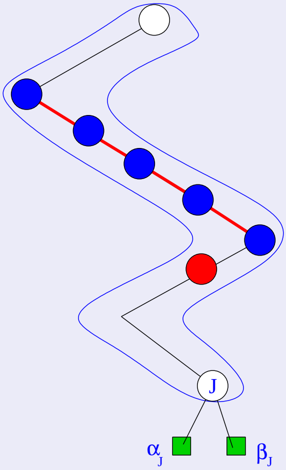
Consider the i-th and $i+1$-th inversion transformation.

$A_i(X)$ be the polynomial associated with the blue node. $a_1, \ldots, a_\ell$ be its positive real roots.

#(Taylor shift) from i-th to $i+1$-th transformation is bounded by $2n(\log a_1 + \cdots + \log a_\ell) \leq 2n^2 \log a_\ell \leq 2n^2 \log q_i$.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.



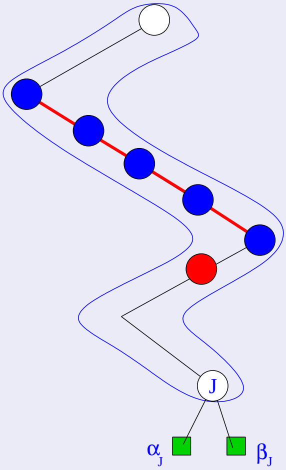
Consider the i-th and $i+1$-th inversion transformation.

$A_i(X)$ be the polynomial associated with the blue node. $a_1, \ldots, a_\ell$ be its positive real roots.

#(Taylor shift) from i-th to $i+1$-th transformation is bounded by
$2n(\log a_1 + \cdots + \log a_\ell) \leq 2n^2 \log a_\ell \leq 2n^2 \log q_i$.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.



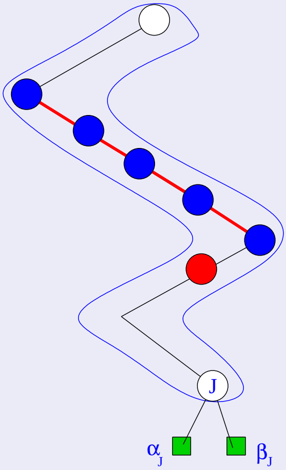
$A_i(X)$ be the polynomial associated with the blue node. $a_1, \ldots, a_\ell$ be its positive real roots.

#(Taylor shift) from i-th to $i+1$-th transformation is bounded by $2n(\log a_1 + \cdots + \log a_\ell) \leq 2n^2 \log a_\ell \leq 2n^2 \log q_i$.

Total number of Taylor shifts on the path to J is $n^2 O(\sum_{i=1}^{m} \log q_i)$.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.

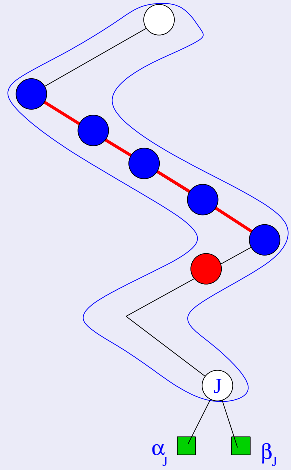


Total number of Taylor shifts on the path to J is $n^2 O(\sum_{i=1}^{m} \log q_i)$.

We can show $\sum_{i=1}^{m} \log q_i = O(\log |\alpha_J - \beta_J|^{-1})$.

Number of Taylor shifts along a path in the tree

Assume the polynomial $A(X)$ has *only real roots*.



Total number of Taylor shifts on the path to J is $n^2 O(\sum_{i=1}^{m} \log q_i)$.

We can show $\sum_{i=1}^{m} \log q_i = O(\log |\alpha_J - \beta_J|^{-1})$.

Total number of Taylor shifts on the path to J is $n^2 O(\log |\alpha_J - \beta_J|^{-1})$.

Number of Taylor shifts along a path in the tree

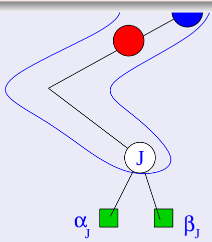
Assume the polynomial $A(X)$ has *only real roots*.

Proposition

The total number of Taylor shifts in the tree is bounded by

$$n^2 O(\sum_J \log|\alpha_J - \beta_J|^{-1}).$$

Lower bound on $\prod_J |\alpha_J - \beta_J|$*?*



$\sum_{i=1}^{m} \log q_i = O(\log|\alpha_J - \beta_J|^{-1})$.

Total number of Taylor shifts on the path to J is $n^2 O(\log|\alpha_J - \beta_J|^{-1})$.

The Davenport–Mahler bound

Theorem (Davenport–Mahler [Dav., 1985] [Du/Sharma/Yap, 2005])

Consider a polynomial $A(X) \in \mathbb{C}[X]$ of degree n.
Let $G = (V, E)$ be a DAG whose vertices are the roots of $A(X)$. If

(i) $(\alpha, \beta) \in E \implies |\alpha| \leq |\beta|$, *and*

(ii) *in-degree of all vertices is at most one.*

then

$$\prod_{(\alpha,\beta)\in E} |\beta - \alpha| \geq \frac{\sqrt{|\mathrm{discr}(A)|}}{\mathrm{M}(A)^{n-1}} \cdot 2^{-O(n \log n)},$$

where, if ϑ_i are roots of $A(X)$,

$$\mathrm{discr}(A) := a_n^{2n-2} \prod_{i>j} (\vartheta_i - \vartheta_j)^2 \quad \textit{and} \quad \mathrm{M}(A) := |a_n| \prod_i \max\{1, |\vartheta_i|\}.$$

The Davenport–Mahler bound

Theorem (Davenport–Mahler [Dav., 1985] [Du/Sharma/Yap, 2005])

Consider a polynomial $A(X) \in \mathbb{C}[X]$ *of degree n.*
Let $G = (V, E)$ *be a DAG whose vertices are the roots of* $A(X)$*. If*

(i) $(\alpha, \beta) \in E \implies |\alpha| \leq |\beta|$, *and*

(ii) *in-degree of all vertices is at most one.*

then

$$\prod_{(\alpha,\beta)\in E} |\beta - \alpha| \geq \frac{\sqrt{|\mathrm{discr}(A)|}}{\mathrm{M}(A)^{n-1}} \cdot 2^{-O(n \log n)},$$

where, if ϑ_i *are roots of* $A(X)$*,*

$$\mathrm{discr}(A) := a_n^{2n-2} \prod_{i>j} (\vartheta_i - \vartheta_j)^2 \quad \textit{and} \quad \mathrm{M}(A) := |a_n| \prod_i \max\{1, |\vartheta_i|\}.$$

The Davenport–Mahler bound

Theorem (Davenport–Mahler [Dav., 1985] [Du/Sharma/Yap, 2005])

Consider a polynomial $A(X) \in \mathbb{C}[X]$ *of degree n.*
Let $G = (V, E)$ *be a DAG whose vertices are the roots of* $A(X)$*. If*

(i) $(\alpha, \beta) \in E \implies |\alpha| \leq |\beta|$, *and*

(ii) *in-degree of all vertices is at most one.*

then

$$\prod_{(\alpha,\beta)\in E} |\beta - \alpha| \geq \frac{\sqrt{|\mathrm{discr}(A)|}}{\mathrm{M}(A)^{n-1}} \cdot 2^{-O(n \log n)},$$

where, if ϑ_i *are roots of* $A(X)$*,*

$$\mathrm{discr}(A) := a_n^{2n-2} \prod_{i>j} (\vartheta_i - \vartheta_j)^2 \quad \textit{and} \quad \mathrm{M}(A) := |a_n| \prod_i \max\{1, |\vartheta_i|\}.$$

Corollary

If $A(X) \in \mathbb{Z}[X]$ is square-free, has degree n, and coefficient bit-length L then
$\prod_{(\alpha,\beta)\in E} |\beta - \alpha| = 2^{-O(nL)}$.

Bound on the Size of the Recursion Tree

Assume $A(X)$ has degree n, coefficient bit-length L and only real roots.

Bound on the Size of the Recursion Tree

Assume $A(X)$ has degree n, coefficient bit-length L and only real roots.

Number of inversion transformations

$$\sum_J (2 - \log|\beta_J - \alpha_J|) = \widetilde{O}(nL).$$

Bound on the Size of the Recursion Tree

Assume $A(X)$ has degree n, coefficient bit-length L and only real roots.

Number of inversion transformations

$$\sum_J (2 - \log|\beta_J - \alpha_J|) = \widetilde{O}(nL).$$

Number of Taylor shifts

$$n^2 \sum_J \log|\beta_J - \alpha_J|^{-1} = \widetilde{O}(n^3 L)$$

Bound on the Size of the Recursion Tree

Assume $A(X)$ has degree n, coefficient bit-length L and only real roots.

Number of inversion transformations

$$\sum_J (2 - \log|\beta_J - \alpha_J|) = \widetilde{O}(nL).$$

Number of Taylor shifts

$$n^2 \sum_J \log|\beta_J - \alpha_J|^{-1} = \widetilde{O}(n^3 L)$$

Theorem

Let $A(X)$ be a square-free polynomial of degree n, and integer coefficients of bit-length L. The size of the recursion tree of Akritas' algorithm run on $A(X)$ is bounded by

- $\widetilde{O}(n^3 L)$, if $A(X)$ has *only real roots*

Bound on the Size of the Recursion Tree

Assume $A(X)$ has degree n, coefficient bit-length L and only real roots.

Number of inversion transformations

$$\sum_J (2 - \log|\beta_J - \alpha_J|) = \widetilde{O}(nL).$$

Number of Taylor shifts

$$n^2 \sum_J \log|\beta_J - \alpha_J|^{-1} = \widetilde{O}(n^3 L)$$

Theorem

Let $A(X)$ be a square-free polynomial of degree n, and integer coefficients of bit-length L. The size of the recursion tree of Akritas' algorithm run on $A(X)$ is bounded by

- $\widetilde{O}(n^3 L)$, if $A(X)$ has *only real roots*
- The result holds in general!

Worst case bit-complexity of a node in the tree

- Let m be the number of inversion transformations along a path.

Worst case bit-complexity of a node in the tree

- Let m be the number of inversion transformations along a path.
- Let the total amount of Taylor shift from i-th to $i+1$-th inversion transformation be q_i, $i = 0, \ldots, m$.

Worst case bit-complexity of a node in the tree

- Let m be the number of inversion transformations along a path.
- Let the total amount of Taylor shift from i-th to $i+1$-th inversion transformation be q_i, $i = 0, \ldots, m$.
- Cost of computing $A(X + q_i)$ using classical Taylor shifts?
  1. Total number of operations (Taylor shifts & Hong's bound) $O(n^2)$.

Worst case bit-complexity of a node in the tree

- Let m be the number of inversion transformations along a path.
- Let the total amount of Taylor shift from i-th to $i+1$-th inversion transformation be q_i, $i = 0, \ldots, m$.
- Cost of computing $A(X+q_i)$ using classical Taylor shifts?
  1. Total number of operations (Taylor shifts & Hong's bound) $O(n^2)$.
  2. Bit-size of the quantities added $L + n\log q_0 + \cdots + n\log q_i$.

Worst case bit-complexity of a node in the tree

- Let m be the number of inversion transformations along a path.
- Let the total amount of Taylor shift from i-th to $i+1$-th inversion transformation be q_i, $i = 0, \ldots, m$.
- Cost of computing $A(X + q_i)$ using classical Taylor shifts?
  1. Total number of operations (Taylor shifts & Hong's bound) $O(n^2)$.
  2. Bit-size of the quantities added $L + n \log q_0 + \cdots + n \log q_i$.
  3. Cost is $O(n^2 M(L + n \log q_0 + \cdots + n \log q_i))$; $M(p)$ is the cost of multiplying two p-bit integers.

Worst case bit-complexity of a node in the tree

- Let m be the number of inversion transformations along a path.
- Let the total amount of Taylor shift from i-th to $i+1$-th inversion transformation be q_i, $i = 0, \ldots, m$.
- Cost of computing $A(X + q_i)$ using classical Taylor shifts?
  1. Total number of operations (Taylor shifts & Hong's bound) $O(n^2)$.
  2. Bit-size of the quantities added $L + n\log q_0 + \cdots + n\log q_i$.
  3. Cost is $O(n^2 M(L + n\log q_0 + \cdots + n\log q_i))$; $M(p)$ is the cost of multiplying two p-bit integers.
- We can show $\sum_{i=0}^{m} \log q_i = \widetilde{O}(nL)$.

Worst case bit-complexity of a node in the tree

- Let m be the number of inversion transformations along a path.
- Let the total amount of Taylor shift from i-th to $i+1$-th inversion transformation be q_i, $i = 0, \dots, m$.
- Cost of computing $A(X + q_i)$ using classical Taylor shifts?
  1. Total number of operations (Taylor shifts & Hong's bound) $O(n^2)$.
  2. Bit-size of the quantities added $L + n \log q_0 + \cdots + n \log q_i$.
  3. Cost is $O(n^2 M(L + n \log q_0 + \cdots + n \log q_i))$; $M(p)$ is the cost of multiplying two p-bit integers.
- We can show $\sum_{i=0}^{m} \log q_i = \widetilde{O}(nL)$.
- Thus worst-case complexity of a node *assuming fast integer arithmetic is* $\widetilde{O}(n^4 L)$.

Worst case bit-complexity of a node in the tree

- Let m be the number of inversion transformations along a path.
- Let the total amount of Taylor shift from i-th to $i+1$-th inversion transformation be q_i, $i = 0, \ldots, m$.
- Cost of computing $A(X+q_i)$ using classical Taylor shifts?
  1. Total number of operations (Taylor shifts & Hong's bound) $O(n^2)$.
  2. Bit-size of the quantities added $L + n\log q_0 + \cdots + n\log q_i$.
  3. Cost is $O(n^2 M(L + n\log q_0 + \cdots + n\log q_i))$; $M(p)$ is the cost of multiplying two p-bit integers.
- We can show $\sum_{i=0}^{m} \log q_i = \widetilde{O}(nL)$.
- Thus worst-case complexity of a node *assuming fast integer arithmetic is* $\widetilde{O}(n^4 L)$.

Cannot use asymptotically fast Taylor shifts [vzGathen/Gerhard, 1997].

Worst case bit-complexity of a node in the tree

- Let m be the number of inversion transformations along a path.
- Let the total amount of Taylor shift from i-th to $i+1$-th inversion transformation be q_i, $i = 0, \ldots, m$.
- Cost of computing $A(X+q_i)$ using classical Taylor shifts?
  1. Total number of operations (Taylor shifts & Hong's bound) $O(n^2)$.
  2. Bit-size of the quantities added $L + n\log q_0 + \cdots + n\log q_i$.
  3. Cost is $O(n^2 M(L + n\log q_0 + \cdots + n\log q_i))$; $M(p)$ is the cost of multiplying two p-bit integers.
- We can show $\sum_{i=0}^{m} \log q_i = \widetilde{O}(nL)$.
- Thus worst-case complexity of a node *assuming fast integer arithmetic is* $\widetilde{O}(n^4 L)$.

Cannot use asymptotically fast Taylor shifts [vzGathen/Gerhard, 1997].

Combined with our worst-case bound $\widetilde{O}(n^3 L)$ on tree-size.

Main Result

Theorem

For a square-free integer polynomial of degree n, and coefficients of bit-length L, the worst-case running time of Akritas' algorithm is bounded by $\widetilde{O}(n^7L^2)$.

Expected Complexity, [Tsigaridas/Emiris, 2006]

The size of the tree

- Assumes floor of the smallest positive root can be computed in $O(1)$.
- Number of Taylor shifts $\sim$ Number of inversion transformations.
- So the size of the tree is $\widetilde{O}(nL)$.

Expected Complexity, [Tsigaridas/Emiris, 2006]

The size of the tree

- Assumes floor of the smallest positive root can be computed in $O(1)$.
- Number of Taylor shifts $\sim$ Number of inversion transformations.
- So the size of the tree is $\widetilde{O}(nL)$.

Expected complexity of a node

- From Khinchin's result we know $E[\sum_{j=0}^{i} \log q_j] = i + 1 = \widetilde{O}(nL)$.
- Expected cost at a node is $O(n^2 M(L + n \sum_{j=0}^{i} b_i)) = \widetilde{O}(n^4 L)$.

Expected Complexity, [Tsigaridas/Emiris, 2006]

The size of the tree

- Assumes floor of the smallest positive root can be computed in $O(1)$.
- Number of Taylor shifts $\sim$ Number of inversion transformations.
- So the size of the tree is $\widetilde{O}(nL)$.

Expected complexity of a node

- From Khinchin's result we know $E[\sum_{j=0}^{i} \log q_j] = i + 1 = \widetilde{O}(nL)$.
- Expected cost at a node is $O(n^2 M(L + n\sum_{j=0}^{i} b_i)) = \widetilde{O}(n^4 L)$.

Theorem

Expected running time of Akritas' algorithm:

- $\widetilde{O}(n^5 L^2)$ using classical Taylor shifts with fast integer arithmetic,
- $\widetilde{O}(n^4 L^2)$ using asymptotically fast Taylor shifts.

Comparison with the Descartes method

Input: $A(X) \in \mathbb{R}[X]$ of degree n, and (c, d).
Output: Isolating intervals for roots of $A(X)$ in (c, d).

Comparison with the Descartes method

Input: $A(X) \in \mathbb{R}[X]$ of degree n, and (c,d).
Output: Isolating intervals for roots of $A(X)$ in (c,d).

Estimate on the number of roots, $E(A,(c,d))$

Let $A(X) = \sum_{i=0}^{n} a_i \binom{n}{i} (X-c)^i (d-X)^{n-i} (d-c)^{-n}$.

$$E(A,(c,d)) := \#(\text{sign variations in } (a_n, a_{n-1}, \ldots, a_0)).$$

- If $E(A,(c,d)) = 0$ then $A(X)$ has no roots in (c,d).
- If $E(A,(c,d)) = 1$ then $A(X)$ has one root in (c,d).

Comparison with the Descartes method

Input: $A(X) \in \mathbb{R}[X]$ of degree n, and (c,d).
Output: Isolating intervals for roots of $A(X)$ in (c,d).

Estimate on the number of roots, $E(A,(c,d))$

Let $A(X) = \sum_{i=0}^{n} a_i \binom{n}{i} (X-c)^i (d-X)^{n-i} (d-c)^{-n}$.

$E(A,(c,d))$:= #(sign variations in $(a_n, a_{n-1}, \ldots, a_0)$).

- If $E(A,(c,d)) = 0$ then $A(X)$ has no roots in (c,d).
- If $E(A,(c,d)) = 1$ then $A(X)$ has one root in (c,d).

$Descartes(A,(c,d))$

- If $E(A,(c,d)) = 0$ **return**.
- If $E(A,(c,d)) = 1$ **output** (c,d).

Comparison with the Descartes method

Input: $A(X) \in \mathbb{R}[X]$ of degree n, and (c,d).
Output: Isolating intervals for roots of $A(X)$ in (c,d).

Estimate on the number of roots, $E(A,(c,d))$

Let $A(X) = \sum_{i=0}^{n} a_i \binom{n}{i} (X-c)^i (d-X)^{n-i} (d-c)^{-n}$.
$E(A,(c,d)) :=$ #(sign variations in $(a_n, a_{n-1}, \ldots, a_0)$).

- If $E(A,(c,d)) = 0$ then $A(X)$ has no roots in (c,d).
- If $E(A,(c,d)) = 1$ then $A(X)$ has one root in (c,d).

Descartes$(A,(c,d))$

- If $E(A,(c,d)) = 0$ **return**.
- If $E(A,(c,d)) = 1$ **output** (c,d).
- If $m := (c+d)/2$ is a root of $A(X)$ **output** $[m,m]$.

Comparison with the Descartes method

Input: $A(X) \in \mathbb{R}[X]$ of degree n, and (c,d).
Output: Isolating intervals for roots of $A(X)$ in (c,d).

Estimate on the number of roots, $E(A,(c,d))$

Let $A(X) = \sum_{i=0}^{n} a_i \binom{n}{i} (X-c)^i (d-X)^{n-i} (d-c)^{-n}$.
$E(A,(c,d)) :=$ #(sign variations in $(a_n, a_{n-1}, \dots, a_0)$).

- If $E(A,(c,d)) = 0$ then $A(X)$ has no roots in (c,d).
- If $E(A,(c,d)) = 1$ then $A(X)$ has one root in (c,d).

$Descartes(A,(c,d))$

- If $E(A,(c,d)) = 0$ **return**.
- If $E(A,(c,d)) = 1$ **output** (c,d).
- If $m := (c+d)/2$ is a root of $A(X)$ **output** $[m,m]$.
- Call $Descartes(A,(c,m))$ and $Descartes(A,(m,d))$.

Comparison with the Descartes method

Input: $A(X) \in \mathbb{R}[X]$ of degree n, and (c,d).
Output: Isolating intervals for roots of $A(X)$ in (c,d).

Estimate on the number of roots, $E(A,(c,d))$

Let $A(X) = \sum_{i=0}^{n} a_i \binom{n}{i} (X-c)^i (d-X)^{n-i} (d-c)^{-n}$.

$E(A,(c,d)) :=$ #(sign variations in $(a_n, a_{n-1}, \ldots, a_0)$).

- If $E(A,(c,d)) = 0$ then $A(X)$ has no roots in (c,d).
- If $E(A,(c,d)) = 1$ then $A(X)$ has one root in (c,d).

$Descartes(A,(c,d))$

- If $E(A,(c,d)) = 0$ **return**.
- If $E(A,(c,d)) = 1$ **output** (c,d).
- If $m := (c+d)/2$ is a root of $A(X)$ **output** $[m,m]$.
- Call $Descartes(A,(c,m))$ and $Descartes(A,(m,d))$.

Comparison with the Descartes method

	Descartes	Akritas
Complexity	$\widetilde{O}(n^5L^2)$	$\widetilde{O}(n^7L^2)$
Size of the tree	$\widetilde{O}(nL)$	$\widetilde{O}(n^3L)$

Comparison with the Descartes method

	Descartes	Akritas
Complexity	$\widetilde{O}(n^5L^2)$	$\widetilde{O}(n^7L^2)$
Size of the tree	$\widetilde{O}(nL)$	$\widetilde{O}(n^3L)$

Reasons

- Width of the interval doesn't necessarily go down by half at each recursion step.
- Lower bound is off by a factor of n (▸).

Comparison with the Descartes method

	Descartes	Akritas
Complexity	$\widetilde{O}(n^5L^2)$	$\widetilde{O}(n^7L^2)$
Size of the tree	$\widetilde{O}(nL)$	$\widetilde{O}(n^3L)$

Reasons

- Width of the interval doesn't necessarily go down by half at each recursion step.
- Lower bound is off by a factor of n ( ).

But...

- Degree 100 Mignotte's polynomials ($X^n - (aX-1)^2$):
  [Emiris/Tsigaridas, '06]: Descartes 7.83sec. and Akritas 0.02sec.
- Available in `Mathematica`.
- [Collins/Akritas, 1976]: $O(n^6L^2)$; [Johnson, 1998]: $O(n^4L^2)$.

Possible ways to improve the complexity

1. Derive tight bounds on largest positive root of a polynomial in $O(n)$ operations. Bounds by [Kioustelidis, 1986; Ştefănescu, 2005] are known to be not tight. A recent bound by Akritas et al. might help.

Possible ways to improve the complexity

1. Derive tight bounds on largest positive root of a polynomial in $O(n)$ operations. Bounds by [Kioustelidis, 1986; Ştefănescu, 2005] are known to be not tight. A recent bound by Akritas et al. might help.
2. Instead of using Horner's method for computing $A(X+b)$, scale by b and shift by one.

Possible ways to improve the complexity

1. Derive tight bounds on largest positive root of a polynomial in $O(n)$ operations. Bounds by [Kioustelidis, 1986; Ştefănescu, 2005] are known to be not tight. A recent bound by Akritas et al. might help.
2. Instead of using Horner's method for computing $A(X+b)$, scale by b and shift by one.

Open question

For Mignotte's polynomial $X^n - 2(aX-1)^2$, $a \in \mathbb{N}$, the size of the recursion tree is $O(\log a)$ using Zassenhaus' bound
(the Descartes method has recursion tree size $\Omega(n \log a)$).

Summary

Main Result

For a square-free polynomial $A(X)$, degree n, and coefficient bit-length L:

1. Worst case bit-complexity of Akritas' algorithm is $\widetilde{O}(n^7 L^2)$.

Summary

Main Result

For a square-free polynomial $A(X)$, degree n, and coefficient bit-length L:

1. Worst case bit-complexity of Akritas' algorithm is $\widetilde{O}(n^7 L^2)$.
2. Worst case number of Taylor shifts – $\widetilde{O}(n^3 L)$.

Summary

Main Result

For a square-free polynomial $A(X)$, degree n, and coefficient bit-length L:

1. Worst case bit-complexity of Akritas' algorithm is $\widetilde{O}(n^7L^2)$.
2. Worst case number of Taylor shifts – $\widetilde{O}(n^3L)$.
3. Using a bound by Hong to compute the floor of the smallest positive root (instead of assuming that it can be done in constant time).

Summary

Main Result

For a square-free polynomial $A(X)$, degree n, and coefficient bit-length L:

1. Worst case bit-complexity of Akritas' algorithm is $\widetilde{O}(n^7L^2)$.
2. Worst case number of Taylor shifts – $\widetilde{O}(n^3L)$.
3. Using a bound by Hong to compute the floor of the smallest positive root (instead of assuming that it can be done in constant time).
4. Using Davenport's bound to amortize the bit-size of continued fractions (instead of using Khinchin's result).

Summary

Main Result

For a square-free polynomial $A(X)$, degree n, and coefficient bit-length L:

1. Worst case bit-complexity of Akritas' algorithm is $\widetilde{O}(n^7L^2)$.
2. Worst case number of Taylor shifts – $\widetilde{O}(n^3L)$.
3. Using a bound by Hong to compute the floor of the smallest positive root (instead of assuming that it can be done in constant time).
4. Using Davenport's bound to amortize the bit-size of continued fractions (instead of using Khinchin's result).

Paper is available from `http://www.cs.nyu.edu/sharma/pap/`.

Merci!