Near Optimal Subdivision Algorithms for Real Root Isolation

Vikram Sharma[1] and Prashant Batra[2]

[1]Institute of Mathematical Sciences, Chennai, India.

[2]Technische Universität Hamburg-Harburg, Hamburg, Germany.

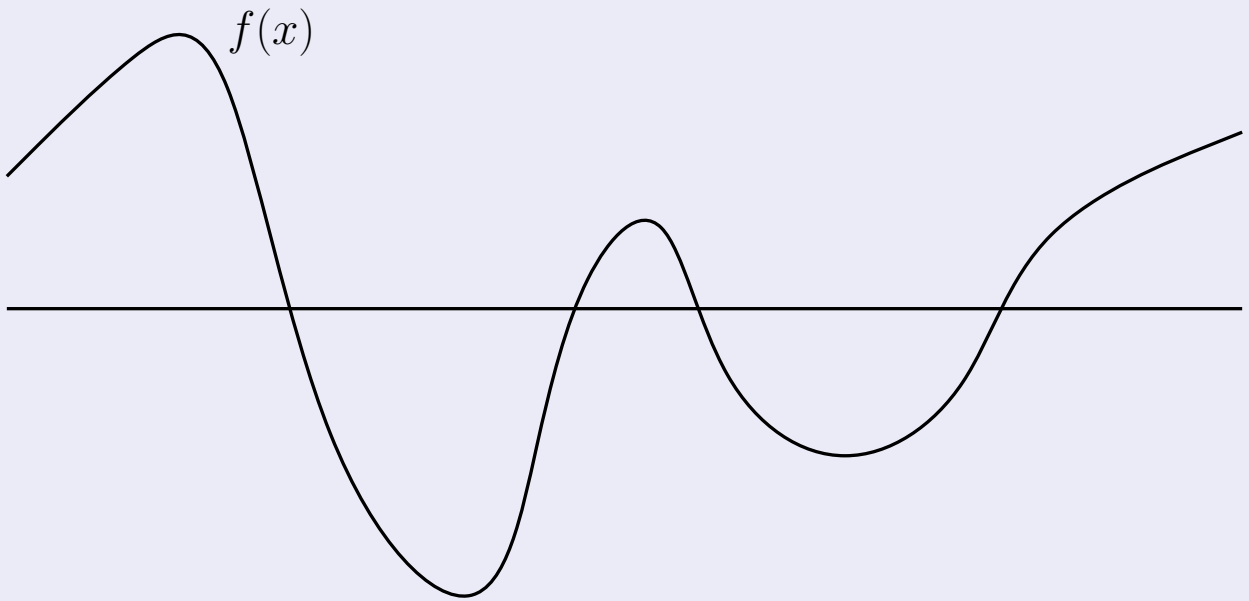
ISSAC, Bath, 2015

Real Root Isolation

The Problem

Input: $f(x) \in \mathbb{R}[x]$, degree n, and an interval I_0.

Output: Disjoint intervals with endpoints in $\mathbb{Q}$ containing roots of f.

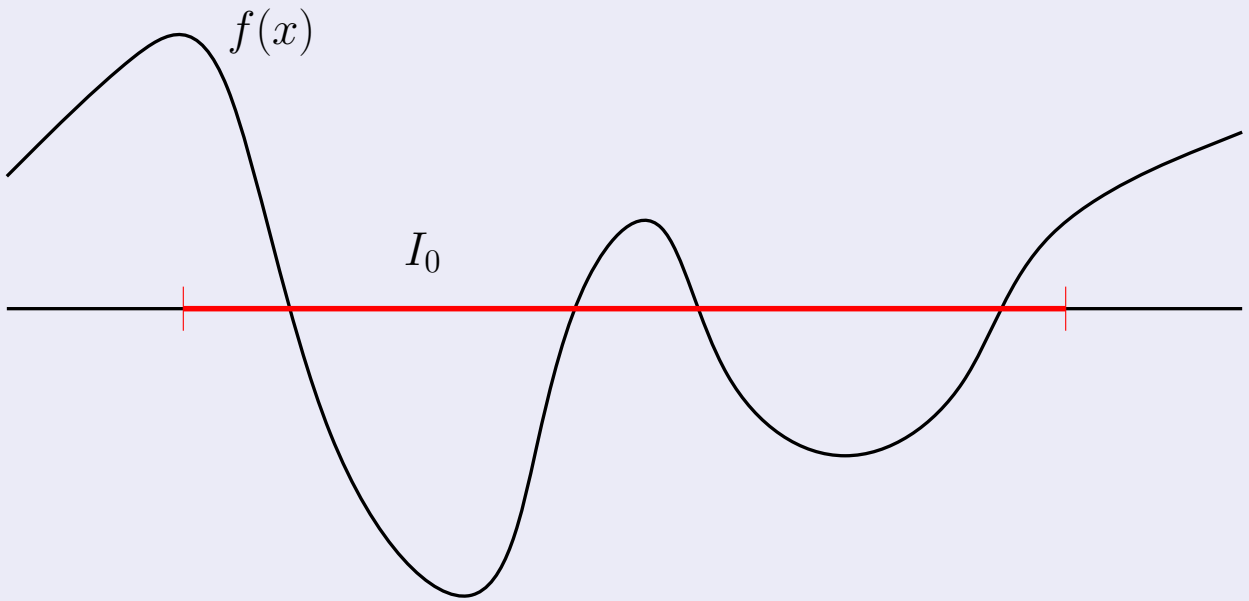


Real Root Isolation

The Problem

Input: $f(x) \in \mathbb{R}[x]$, degree n, and an interval I_0.

Output: Disjoint intervals with endpoints in $\mathbb{Q}$ containing roots of f.

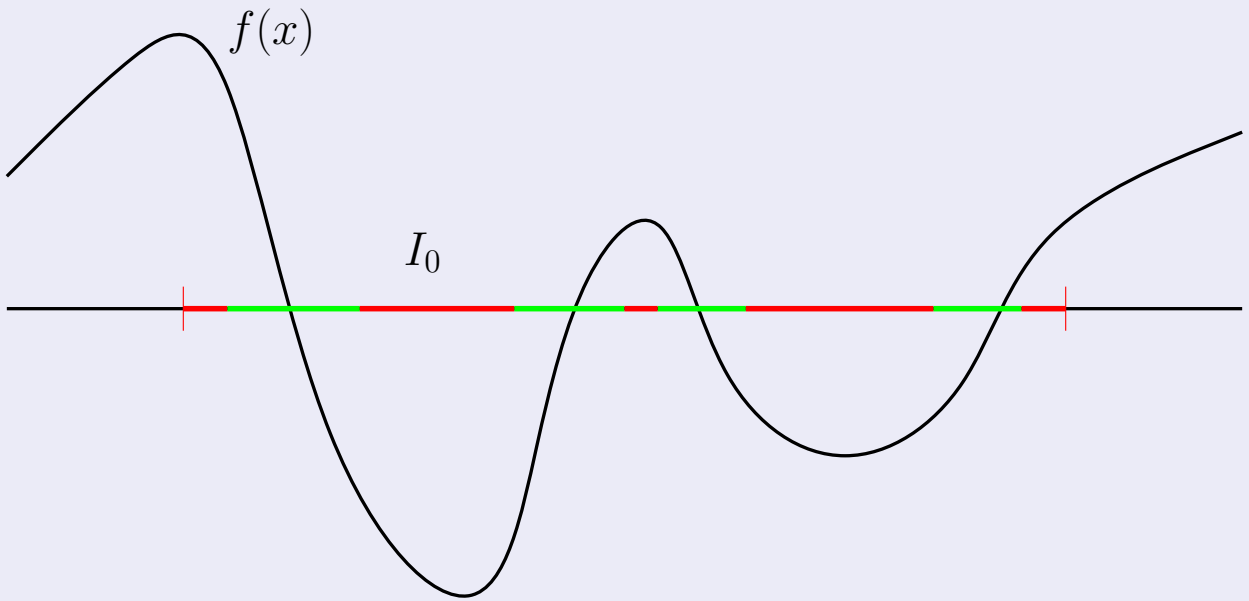


Real Root Isolation

The Problem

Input: $f(x) \in \mathbb{R}[x]$, degree n, and an interval I_0.

Output: Disjoint intervals with endpoints in $\mathbb{Q}$ containing roots of f.

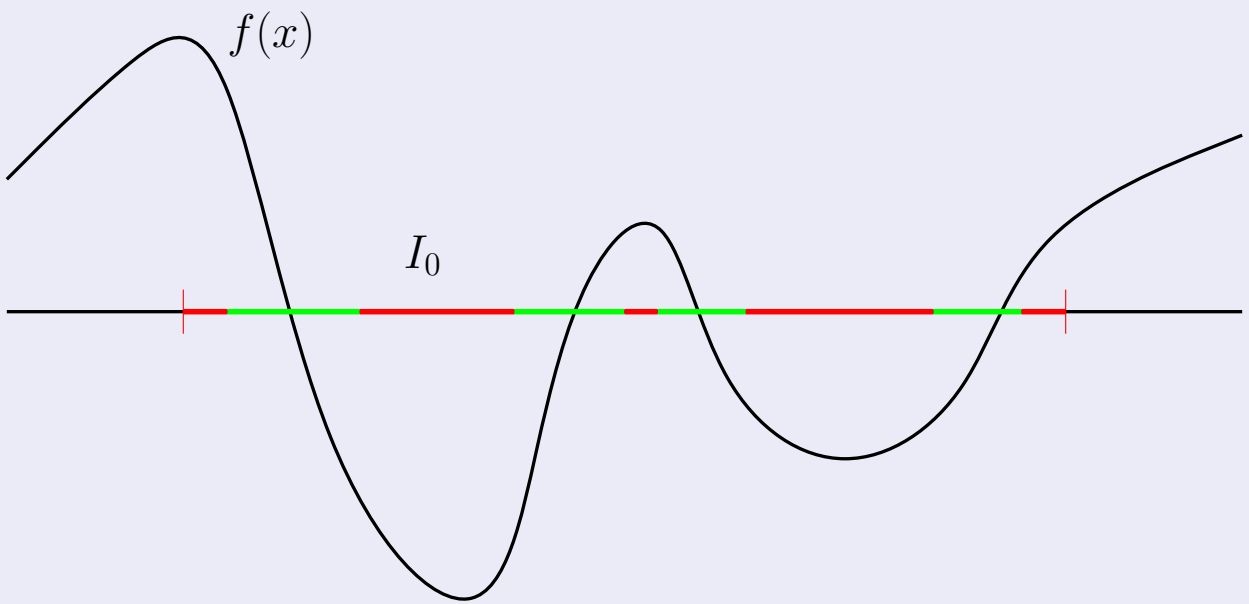


Real Root Isolation

The Problem

Input: $f(x) \in \mathbb{R}[x]$, degree n, and an interval I_0.

Output: Disjoint intervals with endpoints in $\mathbb{Q}$ containing roots of f.



Assumption

All the roots are of multiplicity one, i.e., f is square-free.

Subdivision Methods

Predicates

- Exclusion $C_0(I)$: if true then I has no roots.
- Inclusion $C_1(I)$: if true then I has exactly one root.

E.g., Sturm sequences, Descartes's rule of signs, Interval arithmetic,...

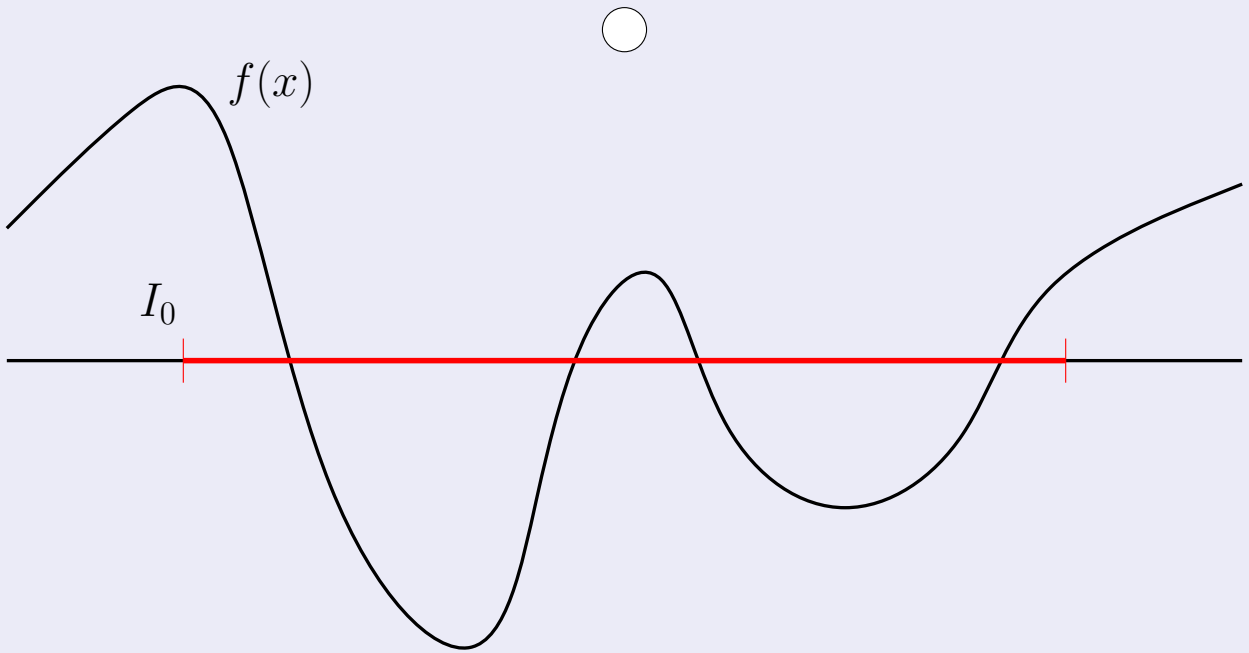
Subdivision Methods

Predicates

- Exclusion $C_0(I)$: if true then I has no roots.
- Inclusion $C_1(I)$: if true then I has exactly one root.

E.g., Sturm sequences, Descartes's rule of signs, Interval arithmetic,...

The Binary Tree T_{I_0}



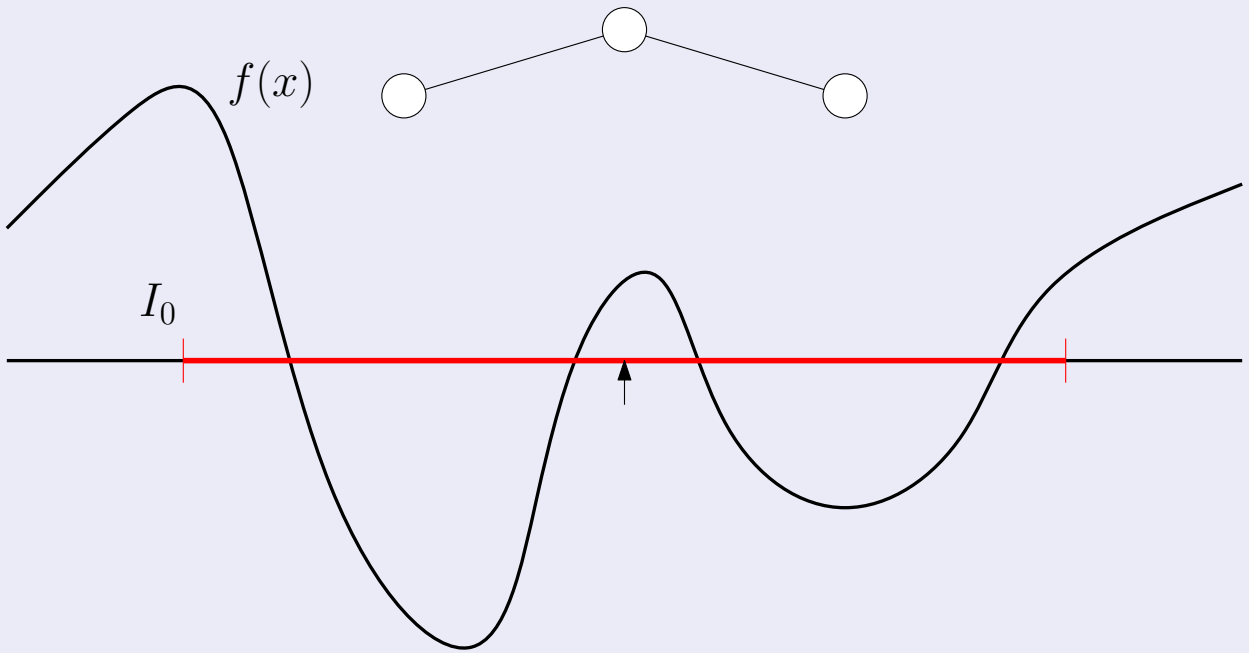
Subdivision Methods

Predicates

- Exclusion $C_0(I)$: if true then I has no roots.
- Inclusion $C_1(I)$: if true then I has exactly one root.

E.g., Sturm sequences, Descartes's rule of signs, Interval arithmetic,...

The Binary Tree T_{I_0}



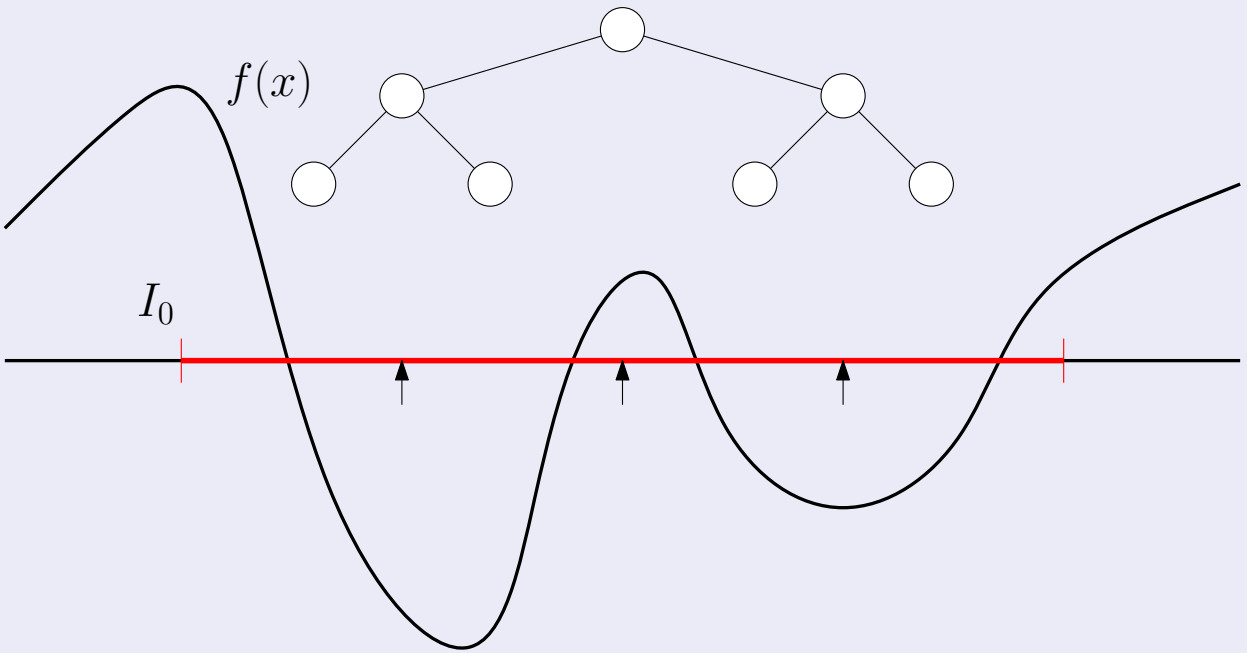
Subdivision Methods

Predicates

- Exclusion $C_0(I)$: if true then I has no roots.
- Inclusion $C_1(I)$: if true then I has exactly one root.

E.g., Sturm sequences, Descartes's rule of signs, Interval arithmetic,...

The Binary Tree T_{I_0}



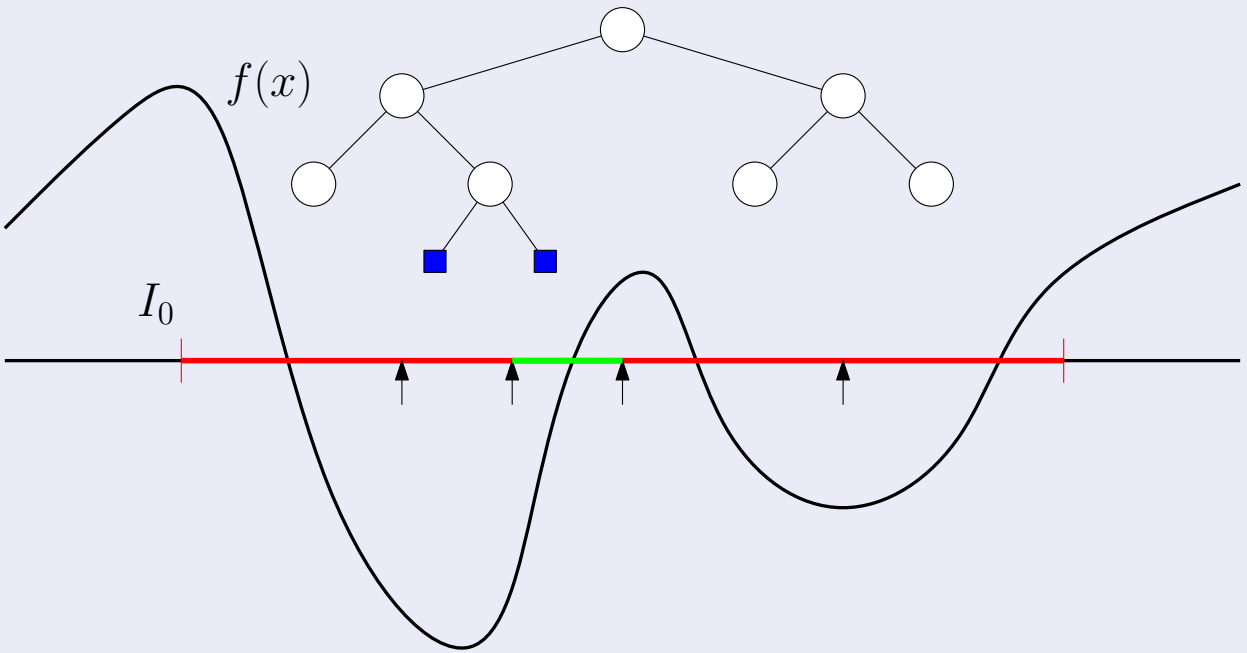
Subdivision Methods

Predicates

- Exclusion $C_0(I)$: if true then I has no roots.
- Inclusion $C_1(I)$: if true then I has exactly one root.

E.g., Sturm sequences, Descartes's rule of signs, Interval arithmetic,...

The Binary Tree T_{I_0}



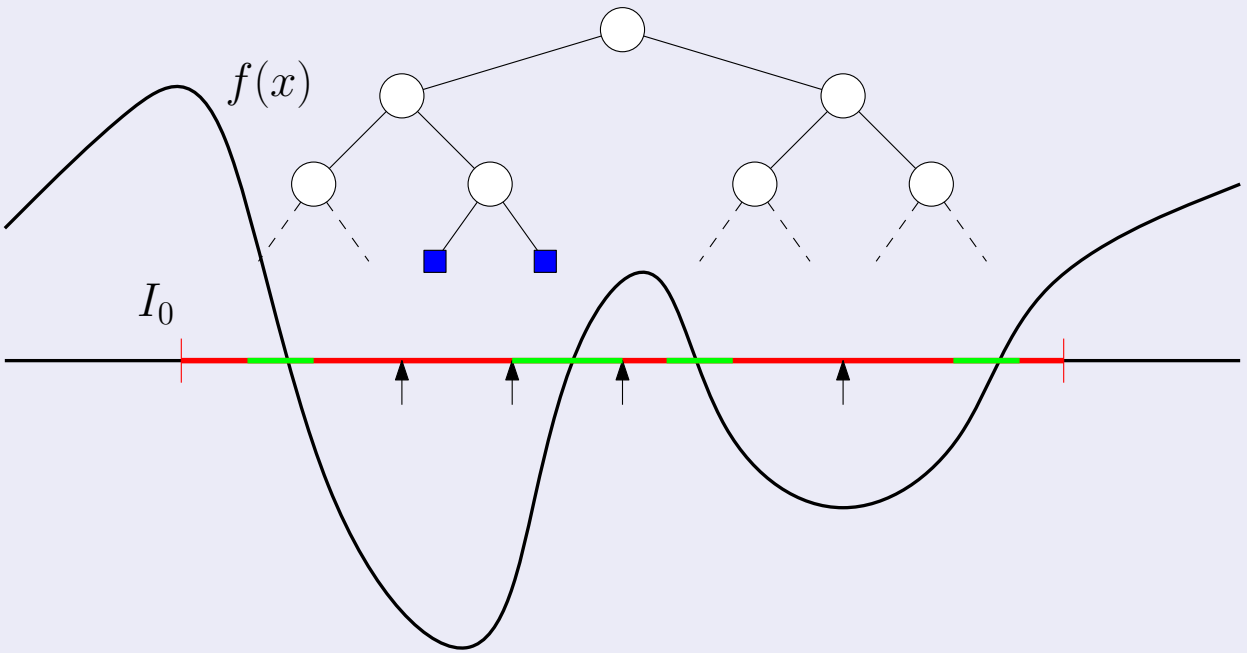
Subdivision Methods

Predicates

- Exclusion $C_0(I)$: if true then I has no roots.
- Inclusion $C_1(I)$: if true then I has exactly one root.

E.g., Sturm sequences, Descartes's rule of signs, Interval arithmetic,...

The Binary Tree T_{I_0}



Subdivision Methods – Complexity Analysis

Two Components

- The size $\#T_{I_0}$ of the tree T_{I_0}.
- The worst case *arithmetic* complexity at a node. Usually $\widetilde{O}(n)$.

Subdivision Methods – Complexity Analysis

Two Components

- The size $\#T_{I_0}$ of the tree T_{I_0}.
- The worst case *arithmetic* complexity at a node. Usually $\widetilde{O}(n)$.

Subdivision Methods – Complexity Analysis

Two Components

- The size $\#T_{I_0}$ of the tree T_{I_0}.
- The worst case *arithmetic* complexity at a node. Usually $\widetilde{O}(n)$.

Bounds on $\#T_{I_0}$

- $O(\log 1/\sigma)$, σ is the root separation bound for f.
- Sturm's method, Descartes's rule of signs, Interval arithmetic. Dav.'85, Eig.-S.-Yap'06, S.-Yap'12.

Subdivision Methods – Complexity Analysis

Two Components

- The size $\#T_{I_0}$ of the tree T_{I_0}.
- The worst case *arithmetic* complexity at a node. Usually $\widetilde{O}(n)$.

Bounds on $\#T_{I_0}$

- $O(\log 1/\sigma)$, σ is the root separation bound for f.
- Sturm's method, Descartes's rule of signs, Interval arithmetic. Dav.'85, Eig.-S.-Yap'06, S.-Yap'12.

Optimal for pure subdivision

Subdivision Methods – Complexity Analysis

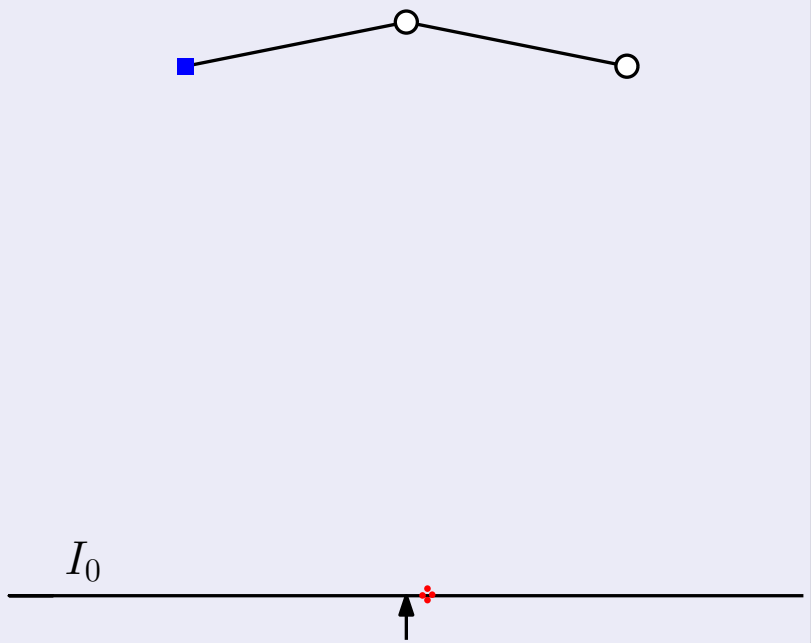
Two Components

- The size $\#T_{I_0}$ of the tree T_{I_0}.
- The worst case *arithmetic* complexity at a node. Usually $\widetilde{O}(n)$.

Bounds on $\#T_{I_0}$

- $O(\log 1/\sigma)$, σ is the root separation bound for f.
- Sturm's method, Descartes's rule of signs, Interval arithmetic. Dav.'85, Eig.-S.-Yap'06, S.-Yap'12.

Optimal for pure subdivision



Subdivision Methods – Complexity Analysis

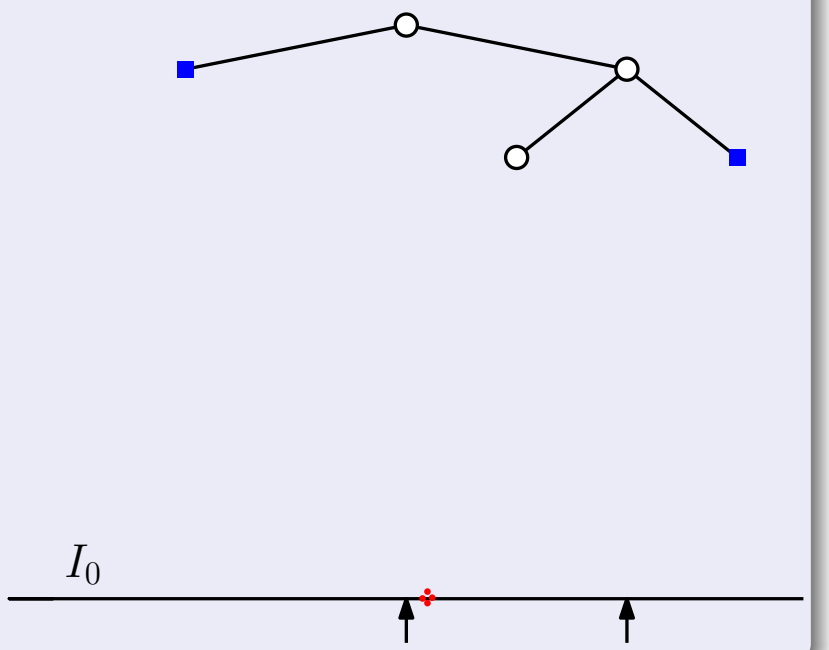
Two Components

- The size $\#T_{I_0}$ of the tree T_{I_0}.
- The worst case *arithmetic* complexity at a node. Usually $\widetilde{O}(n)$.

Bounds on $\#T_{I_0}$

- $O(\log 1/\sigma)$, σ is the root separation bound for f.
- Sturm's method, Descartes's rule of signs, Interval arithmetic. Dav.'85, Eig.-S.-Yap'06, S.-Yap'12.

Optimal for pure subdivision



Subdivision Methods – Complexity Analysis

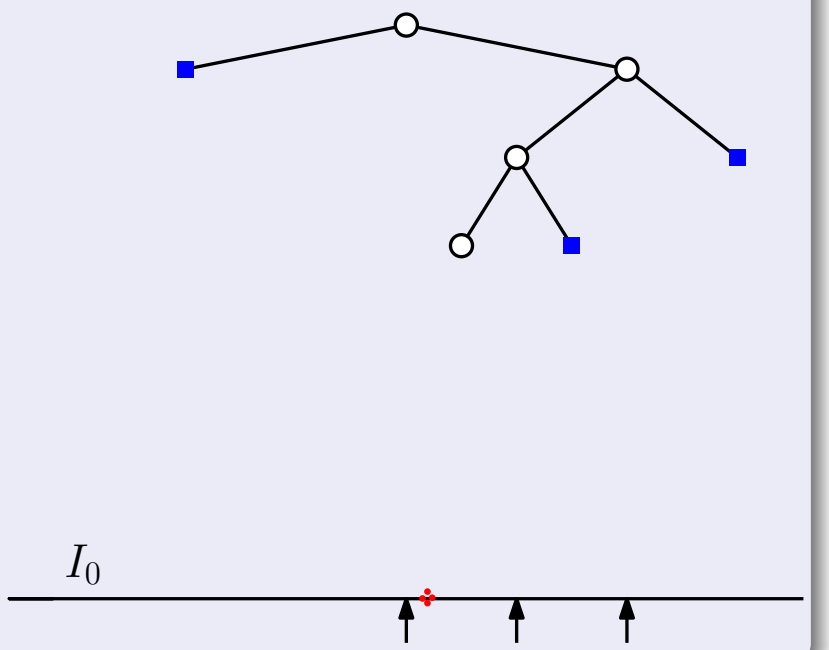
Two Components

- The size $\#T_{I_0}$ of the tree T_{I_0}.
- The worst case *arithmetic* complexity at a node. Usually $\widetilde{O}(n)$.

Bounds on $\#T_{I_0}$

- $O(\log 1/\sigma)$, σ is the root separation bound for f.
- Sturm's method, Descartes's rule of signs, Interval arithmetic. Dav.'85, Eig.-S.-Yap'06, S.-Yap'12.

Optimal for pure subdivision



Subdivision Methods – Complexity Analysis

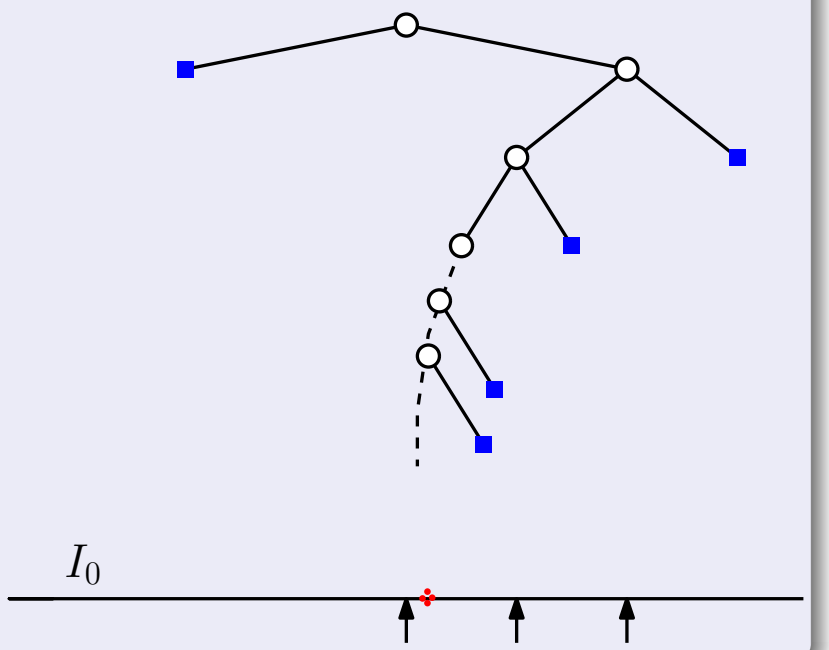
Two Components

- The size $\#T_{I_0}$ of the tree T_{I_0}.
- The worst case *arithmetic* complexity at a node. Usually $\widetilde{O}(n)$.

Bounds on $\#T_{I_0}$

- $O(\log 1/\sigma)$, σ is the root separation bound for f.
- Sturm's method, Descartes's rule of signs, Interval arithmetic. Dav.'85, Eig.-S.-Yap'06, S.-Yap'12.

Optimal for pure subdivision



Subdivision Methods – Complexity Analysis

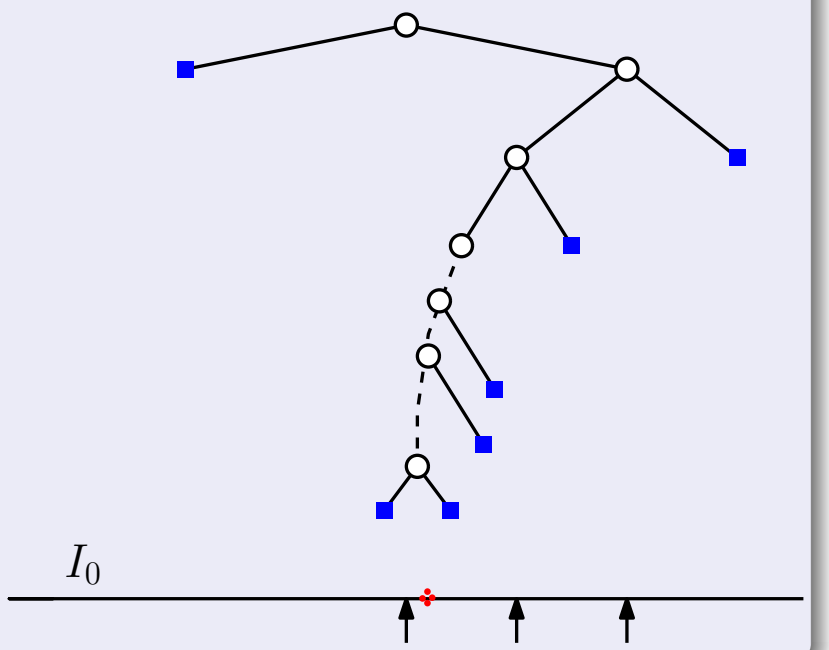
Two Components

- The size $\#T_{I_0}$ of the tree T_{I_0}.
- The worst case *arithmetic* complexity at a node. Usually $\widetilde{O}(n)$.

Bounds on $\#T_{I_0}$

- $O(\log 1/\sigma)$, σ is the root separation bound for f.
- Sturm's method, Descartes's rule of signs, Interval arithmetic. Dav.'85, Eig.-S.-Yap'06, S.-Yap'12.

Optimal for pure subdivision



Subdivision Methods – Complexity Analysis

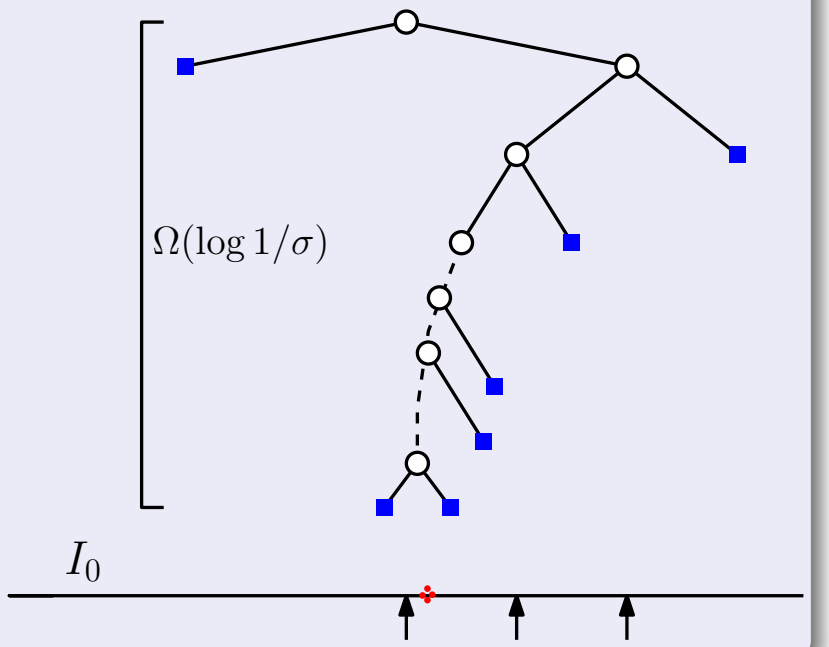
Two Components

- The size $\#T_{I_0}$ of the tree T_{I_0}.
- The worst case *arithmetic* complexity at a node. Usually $\widetilde{O}(n)$.

Bounds on $\#T_{I_0}$

- $O(\log 1/\sigma)$, σ is the root separation bound for f.
- Sturm's method, Descartes's rule of signs, Interval arithmetic. Dav.'85, Eig.-S.-Yap'06, S.-Yap'12.

Optimal for pure subdivision



Subdivision Methods – Complexity Analysis

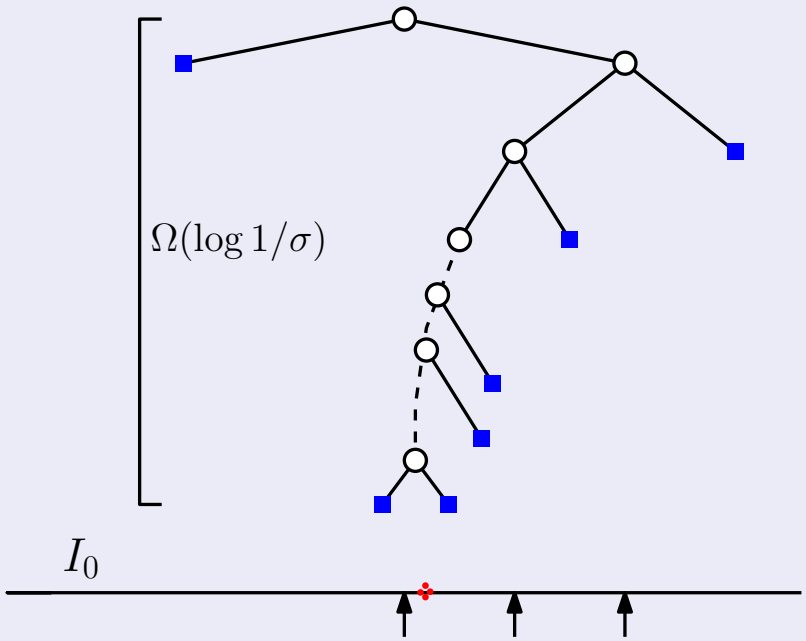
Two Components

- The size $\#T_{I_0}$ of the tree T_{I_0}.
- The worst case *arithmetic* complexity at a node. Usually $\widetilde{O}(n)$.

Bounds on $\#T_{I_0}$

- $O(\log 1/\sigma)$, σ is the root separation bound for f.
- Sturm's method, Descartes's rule of signs, Interval arithmetic. Dav.'85, Eig.-S.-Yap'06, S.-Yap'12.

Optimal for pure subdivision

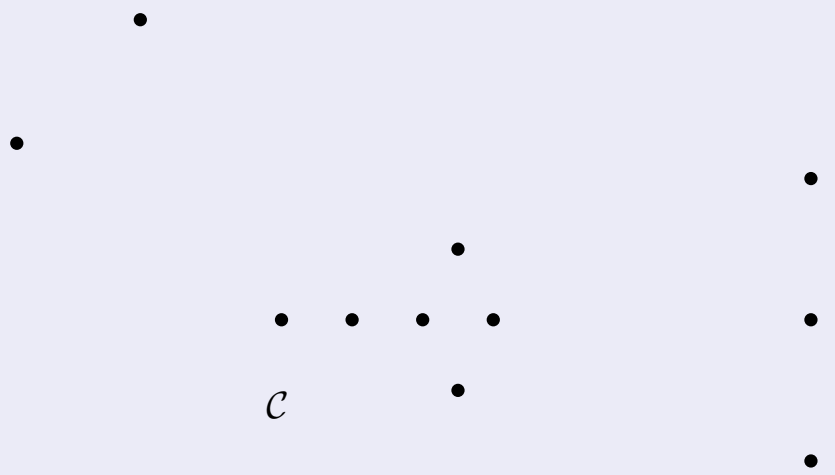


The Problem

Subdivision only gives linear convergence to clusters of roots.

The Remedy – Subdivision + Newton Iteration

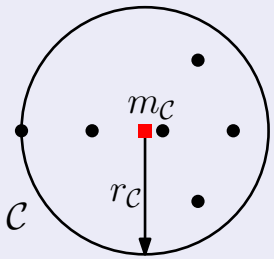
Cluster



- $\mathcal{C}$ a subset of roots.

The Remedy – Subdivision + Newton Iteration

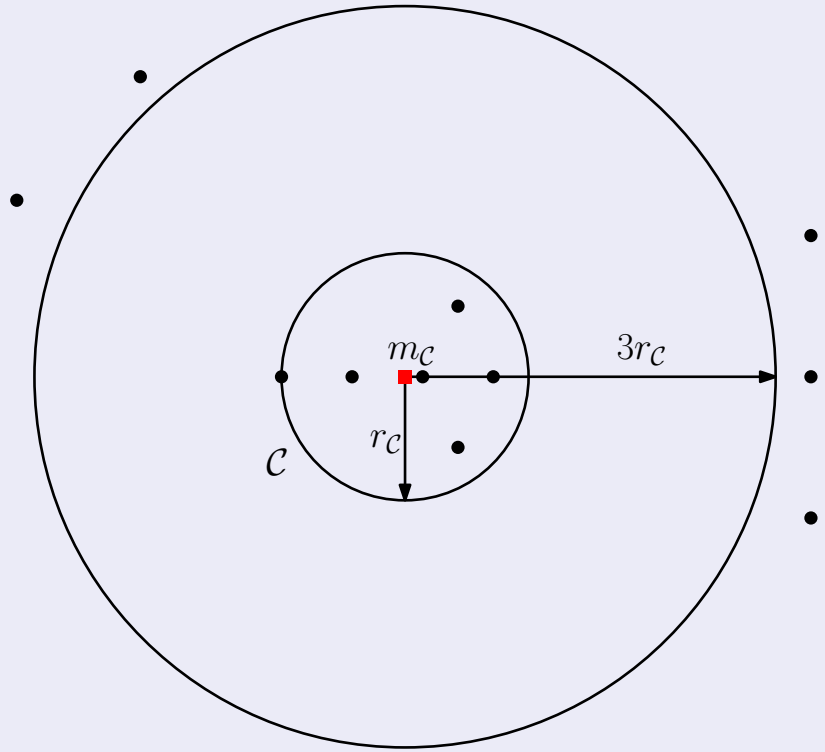
Cluster



- $\mathcal{C}$ a subset of roots.
- $m_{\mathcal{C}}$ is the centroid.
- $r_{\mathcal{C}}$ the cluster radius.

The Remedy – Subdivision + Newton Iteration

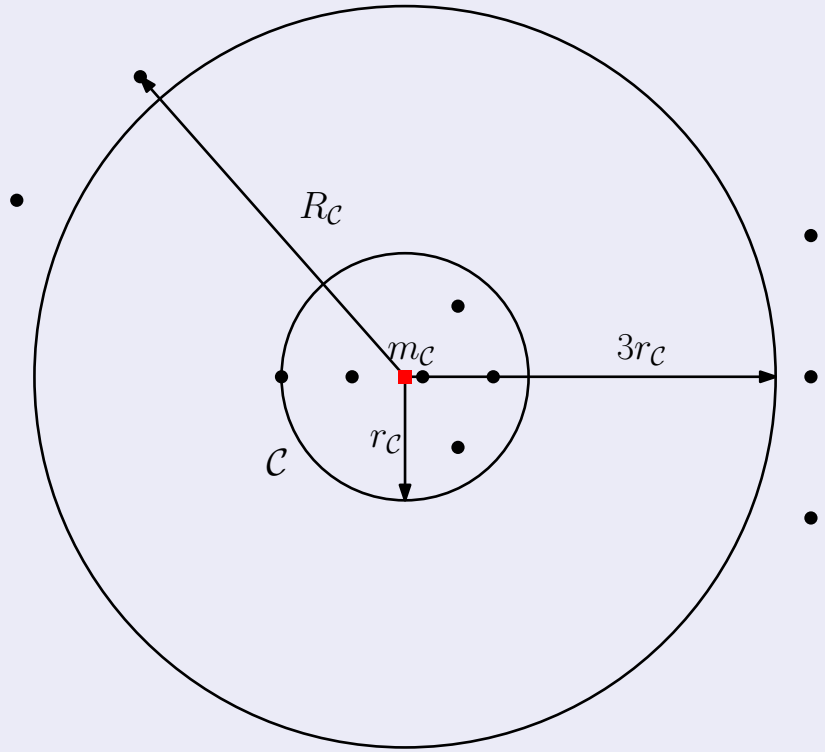
Cluster



- $\mathcal{C}$ a subset of roots.
- $m_{\mathcal{C}}$ is the centroid.
- $r_{\mathcal{C}}$ the cluster radius.

The Remedy – Subdivision + Newton Iteration

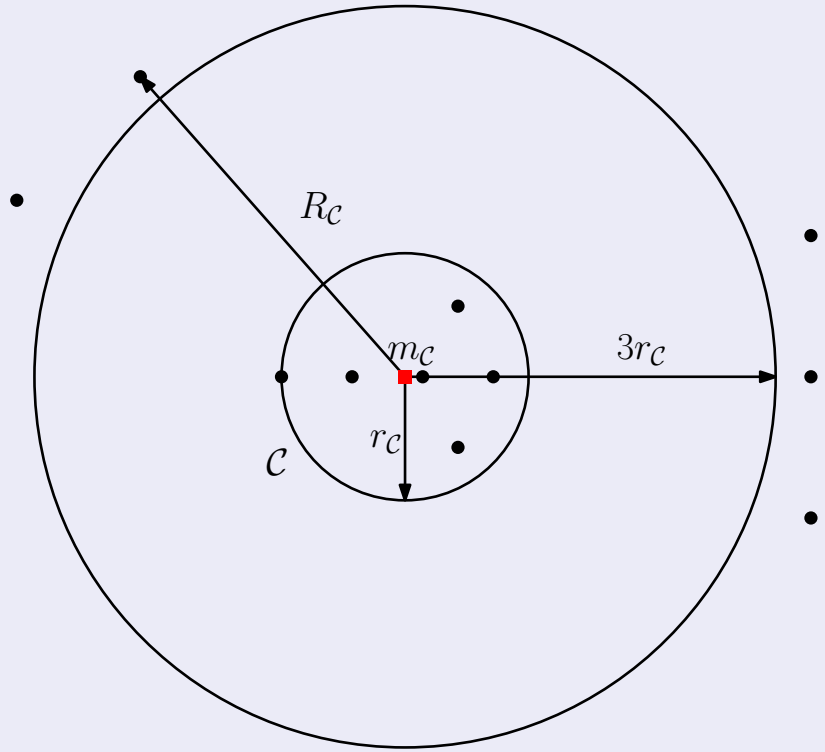
Cluster



- $\mathcal{C}$ a subset of roots.
- $m_{\mathcal{C}}$ is the centroid.
- $r_{\mathcal{C}}$ the cluster radius.
- $R_{\mathcal{C}}$ the isolation radius.

The Remedy – Subdivision + Newton Iteration

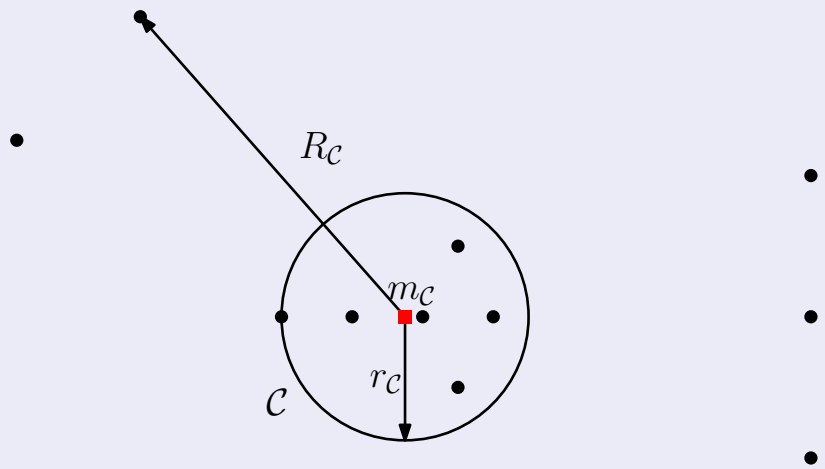
Cluster



- $\mathcal{C}$ a subset of roots.
- $m_{\mathcal{C}}$ is the centroid.
- $r_{\mathcal{C}}$ the cluster radius.
- $R_{\mathcal{C}}$ the isolation radius.
- $Z(f)$ and single roots are trivial clusters.

The Remedy – Subdivision + Newton Iteration

Cluster



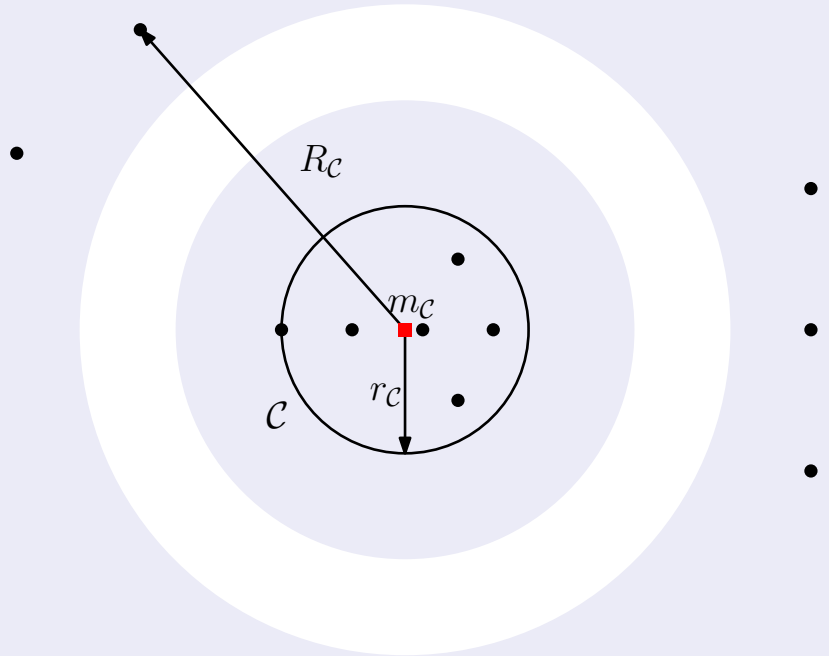
- $\mathcal{C}$ a subset of roots.
- $m_{\mathcal{C}}$ is the centroid.
- $r_{\mathcal{C}}$ the cluster radius.
- $R_{\mathcal{C}}$ the isolation radius.
- $Z(f)$ and single roots are trivial clusters.

Quadratic convergence of Newton iteration

If $R_{\mathcal{C}} \gtrsim n^2 r_{\mathcal{C}}$ then

The Remedy – Subdivision + Newton Iteration

Cluster



- $\mathcal{C}$ a subset of roots.
- $m_{\mathcal{C}}$ is the centroid.
- $r_{\mathcal{C}}$ the cluster radius.
- $R_{\mathcal{C}}$ the isolation radius.
- $Z(f)$ and single roots are trivial clusters.

Quadratic convergence of Newton iteration

If $R_{\mathcal{C}} \gtrsim n^2 r_{\mathcal{C}}$ then $\exists$ an annulus in which Newton iteration converges to $m_{\mathcal{C}}$.

Resulting Improvements

Relevant Results

- Pan'00 – Predicates based on distance to nearest root, Newton + Graeffe iteration for cluster detection, complex roots isolation.

Resulting Improvements

Relevant Results

- Pan'00 – Predicates based on distance to nearest root, Newton + Graeffe iteration for cluster detection, complex roots isolation.
- Sagraloff'12, Sagraloff-Mehlhorn'13 – Predicates based on Pellet's test or Descartes's rule of Signs, Quadratic Interval Refinement+Newton iteration.

Resulting Improvements

Relevant Results

- Pan'00 – Predicates based on distance to nearest root, Newton + Graeffe iteration for cluster detection, complex roots isolation.
- Sagraloff'12, Sagraloff-Mehlhorn'13 – Predicates based on Pellet's test or Descartes's rule of Signs, Quadratic Interval Refinement+Newton iteration.

Bounds on Treesize – n degree, σ root separation

$$O(n \log (n \log 1/\sigma)))$$

Resulting Improvements

Relevant Results

- Pan'00 – Predicates based on distance to nearest root, Newton + Graeffe iteration for cluster detection, complex roots isolation.
- Sagraloff'12, Sagraloff-Mehlhorn'13 – Predicates based on Pellet's test or Descartes's rule of Signs, Quadratic Interval Refinement+Newton iteration.

Bounds on Treesize – n degree, σ root separation

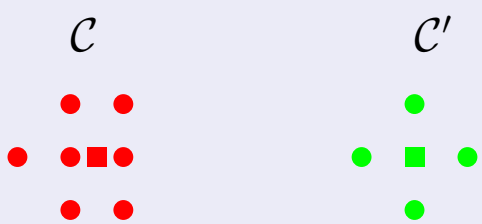
$$O(n \log (n \log 1/\sigma)))$$

Our result

- Choice of any inclusion-exclusion predicate, Newton iteration.
- $O(n \log n)$, for Descartes's rule of signs or Sturm sequences.
- Analysis is independent of root bounds, uses geometry of clusters.
- Use the framework of continuous amortization.

Basic Idea

Cluster Tree

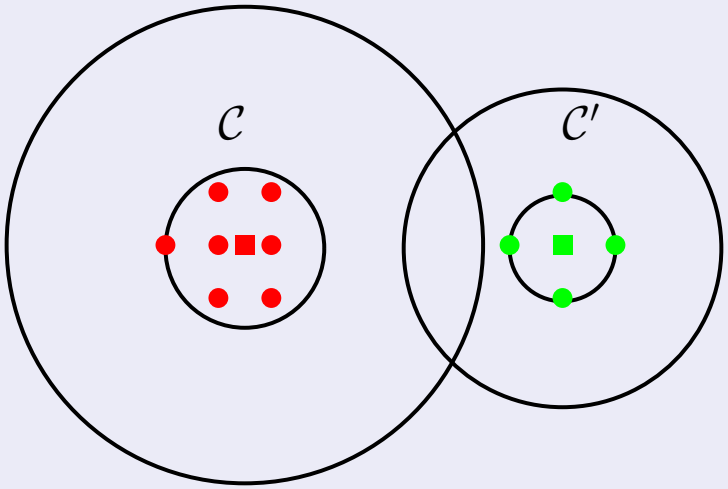


Claim

If $\mathcal{C} \cap \mathcal{C}' \neq \emptyset$ then either $\mathcal{C} \subseteq \mathcal{C}'$ or vice versa.

Basic Idea

Cluster Tree

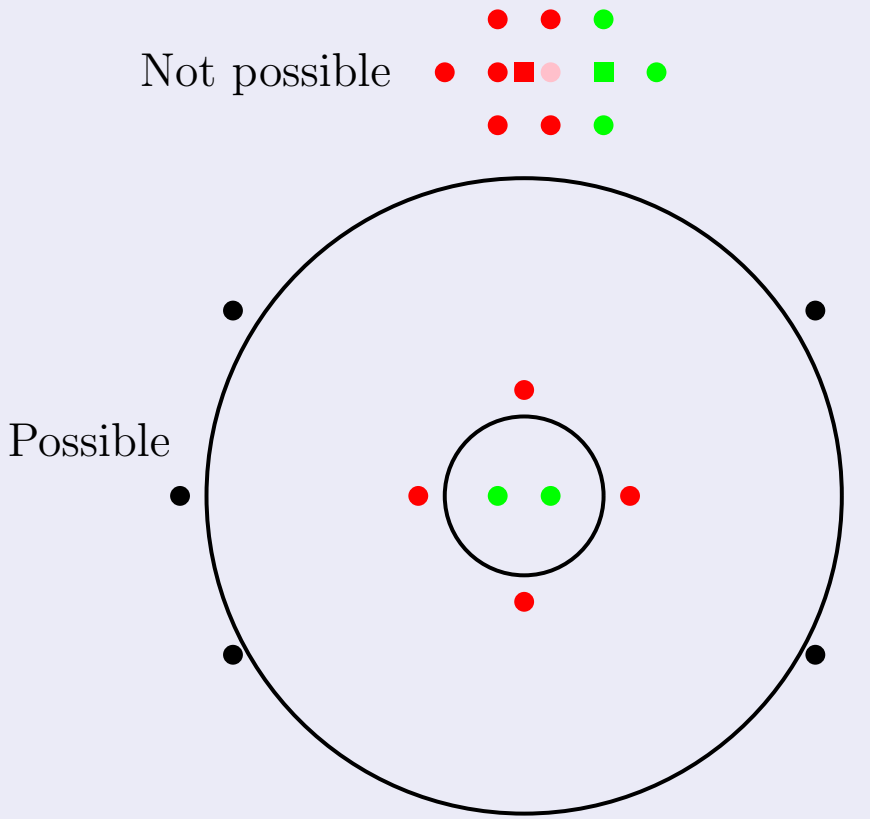


Claim

If $\mathcal{C} \cap \mathcal{C}' \neq \emptyset$ then either $\mathcal{C} \subseteq \mathcal{C}'$ or vice versa.

Basic Idea

Cluster Tree

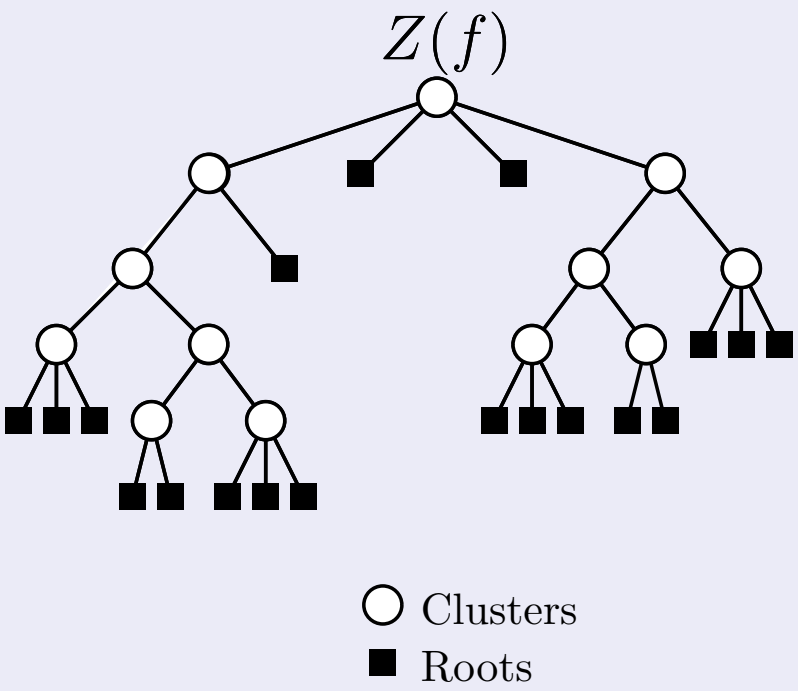


Claim

If $\mathcal{C} \cap \mathcal{C}' \neq \emptyset$ then either $\mathcal{C} \subseteq \mathcal{C}'$ or vice versa.

Basic Idea

Cluster Tree



Claim

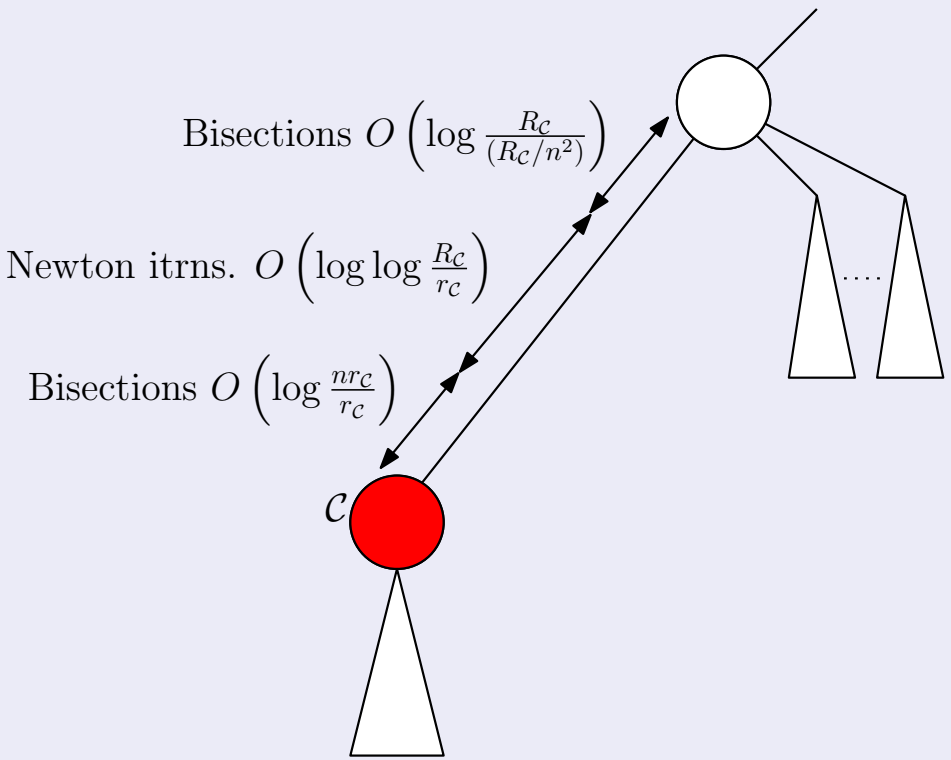
If $\mathcal{C} \cap \mathcal{C}' \neq \emptyset$ then either $\mathcal{C} \subseteq \mathcal{C}'$ or vice versa.

Lemma (Sagraloff-S.-Yap'13)

The set of clusters form a tree with root node as the set of all zeros $Z(f)$.

Basic Idea

Number of Bisections in Cluster Tree

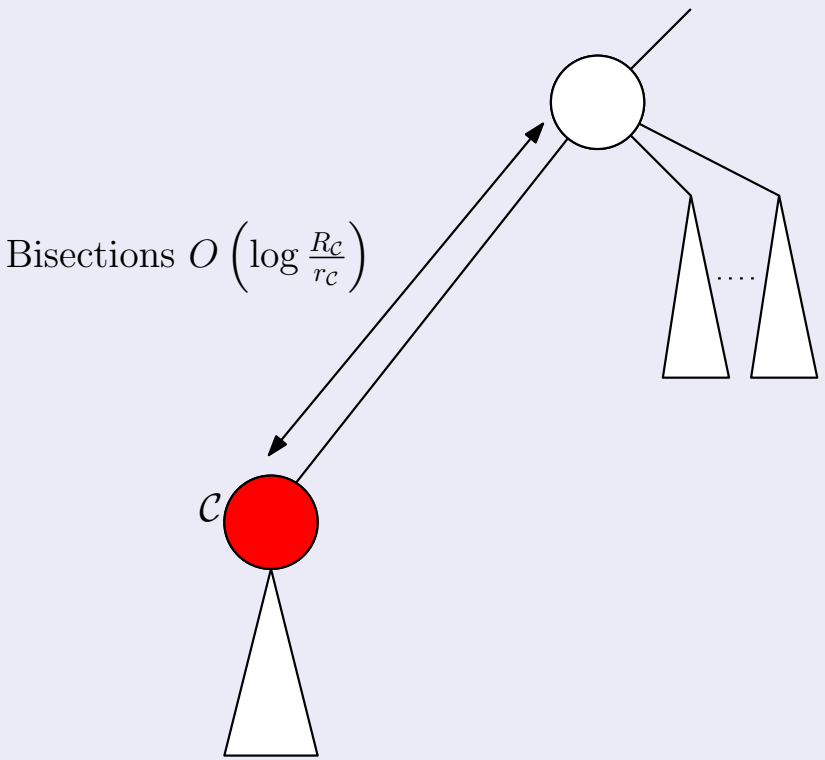


Strongly Separated Clusters

- $R_C \gtrsim n^3 r_C$.
- Bisection $O(\log R_C/r_C)$ steps.
- Detect and converge using $O(\log\log R_C/r_C)$ Newton iterations.

Basic Idea

Number of Bisections in Cluster Tree



Strongly Separated Clusters

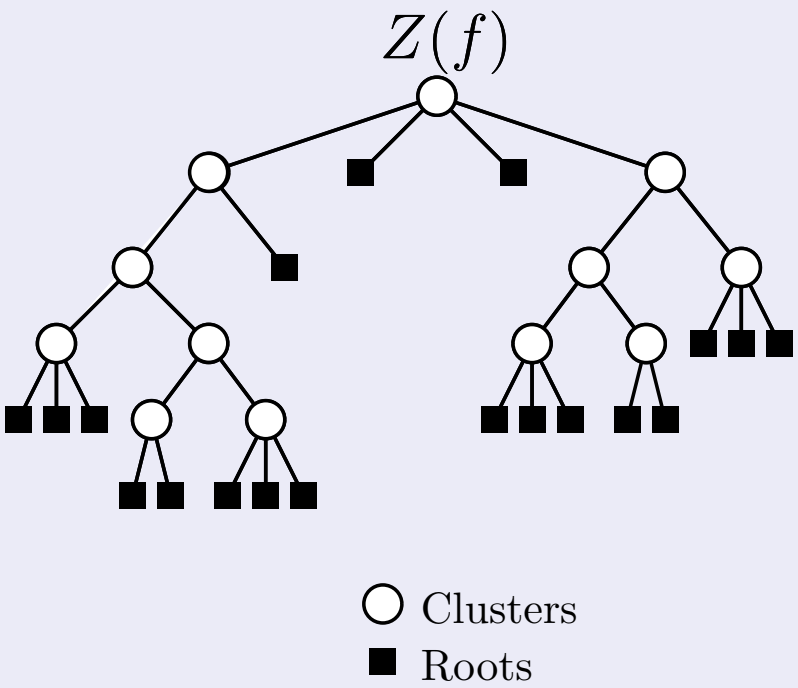
- $R_C \gtrsim n^3 r_C$.
- Bisection $O(\log R_C / r_C)$ steps.
- Detect and converge using $O(\log\log R_C / r_C)$ Newton iterations.

Ordinary Clusters

- $R_C \lesssim n^3 r_C$.
- Bisection steps $O(\log \frac{R_C}{r_C}) = O(\log n)$.

Basic Idea

Number of Bisections in Cluster Tree



Strongly Separated Clusters

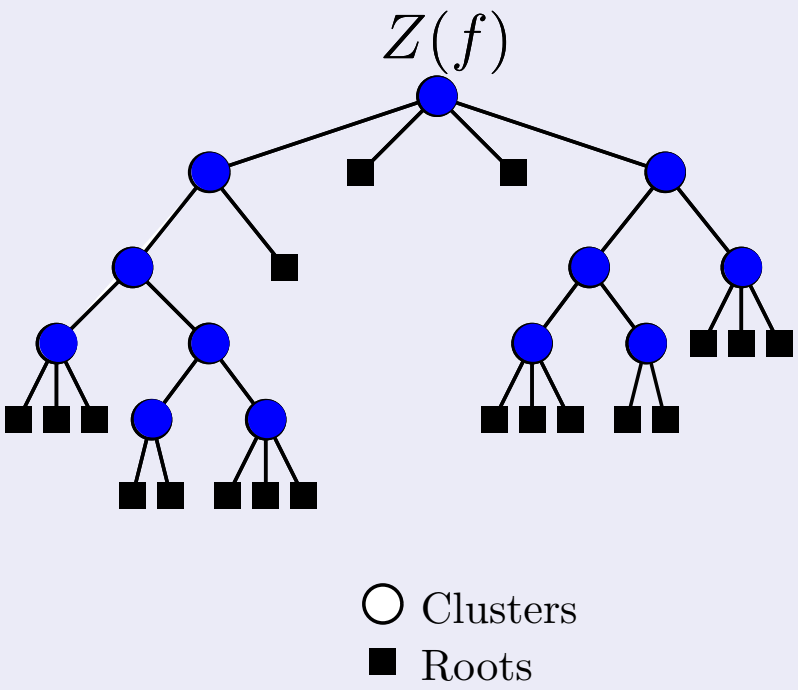
- $R_{\mathcal{C}} \gtrsim n^3 r_{\mathcal{C}}$.
- Bisection $O(\log R_{\mathcal{C}}/r_{\mathcal{C}})$ steps.
- Detect and converge using $O(\log\log R_{\mathcal{C}}/r_{\mathcal{C}})$ Newton iterations.

Ordinary Clusters

- $R_{\mathcal{C}} \lesssim n^3 r_{\mathcal{C}}$.
- Bisection steps $O(\log \frac{R_{\mathcal{C}}}{r_{\mathcal{C}}}) = O(\log n)$.

Basic Idea

Number of Bisections in Cluster Tree



Strongly Separated Clusters

- $R_{\mathcal{C}} \gtrsim n^3 r_{\mathcal{C}}$.
- Bisection $O(\log R_{\mathcal{C}}/r_{\mathcal{C}})$ steps.
- Detect and converge using $O(\log\log R_{\mathcal{C}}/r_{\mathcal{C}})$ Newton iterations.

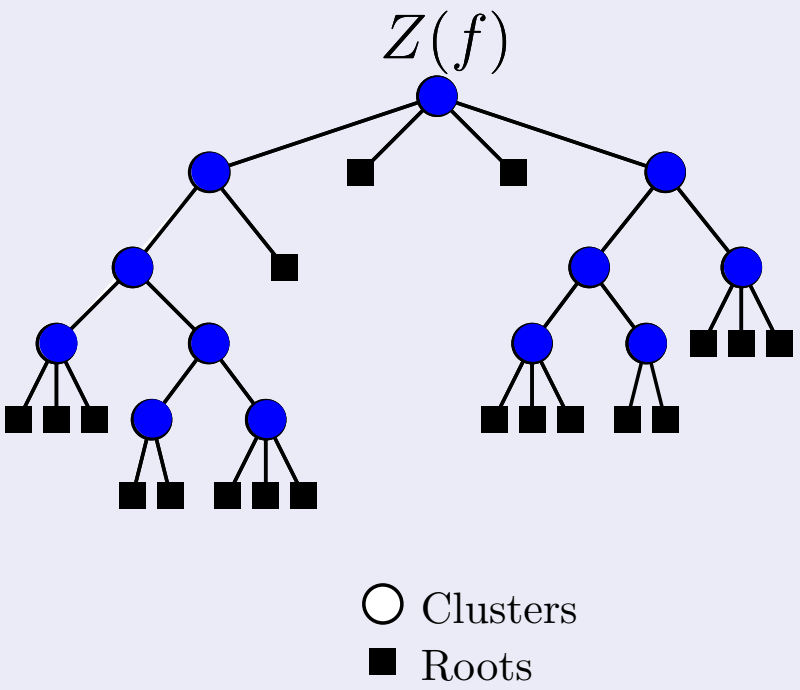
Ordinary Clusters

- $R_{\mathcal{C}} \lesssim n^3 r_{\mathcal{C}}$.
- Bisection steps $O(\log \frac{R_{\mathcal{C}}}{r_{\mathcal{C}}}) = O(\log n)$.

The number of bisection steps in T_{I_0} is $O(n \log n)$.

Basic Idea

Number of Bisections in Cluster Tree



Strongly Separated Clusters

- $R_{\mathcal{C}} \gtrsim n^3 r_{\mathcal{C}}$.
- Bisection $O(\log R_{\mathcal{C}}/r_{\mathcal{C}})$ steps.
- Detect and converge using $O(\log\log R_{\mathcal{C}}/r_{\mathcal{C}})$ Newton iterations.

Ordinary Clusters

- $R_{\mathcal{C}} \lesssim n^3 r_{\mathcal{C}}$.
- Bisection steps $O(\log \frac{R_{\mathcal{C}}}{r_{\mathcal{C}}}) = O(\log n)$.

The number of bisection steps in T_{I_0} is $O(n \log n)$.

Detecting and Approximating Clusters

Ostrowski'34, S.-Batra'15

Let $f(x+z) := \sum_j f_j(z)x^j$, i.e., $f_j(z)$ is the jth Taylor coefficient at z.

- For $k \in [n]$, $\rho_k(z) := \max_{j<k} \left|\frac{f_j(z)}{f_k(z)}\right|^{\frac{1}{(k-j)}}$, $\rho_{k+1}(z) := \min_{j>k} \left|\frac{f_k(z)}{f_j(z)}\right|^{\frac{1}{(j-k)}}$.

Detecting and Approximating Clusters

Ostrowski'34, S.-Batra'15

Let $f(x+z) := \sum_j f_j(z)x^j$, i.e., $f_j(z)$ is the jth Taylor coefficient at z.

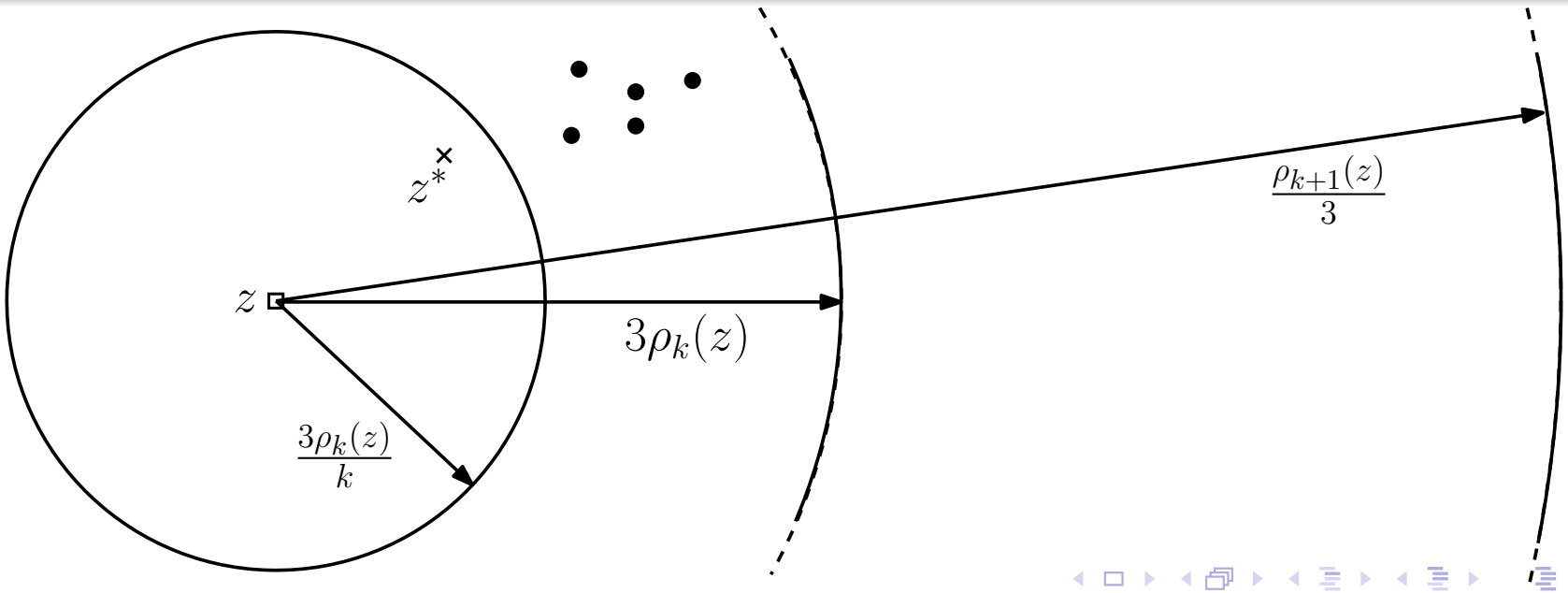
- For $k \in [n]$, $\rho_k(z) := \max_{j<k} \left|\frac{f_j(z)}{f_k(z)}\right|^{\frac{1}{(k-j)}}$, $\rho_{k+1}(z) := \min_{j>k} \left|\frac{f_k(z)}{f_j(z)}\right|^{\frac{1}{(j-k)}}$.
- If $\rho_{k+1}(z)/3 > 3 \cdot 3\rho_k(z)$ then k roots in $D(z, 3\rho_k(z)) \subseteq D(z, \frac{\rho_{k+1}(z)}{3})$.

Detecting and Approximating Clusters

Ostrowski'34, S.-Batra'15

Let $f(x+z) := \sum_j f_j(z)x^j$, i.e., $f_j(z)$ is the jth Taylor coefficient at z.

- For $k \in [n]$, $\rho_k(z) := \max_{j<k} \left|\frac{f_j(z)}{f_k(z)}\right|^{\frac{1}{(k-j)}}$, $\rho_{k+1}(z) := \min_{j>k} \left|\frac{f_k(z)}{f_j(z)}\right|^{\frac{1}{(j-k)}}$.
- If $\rho_{k+1}(z)/3 > 3 \cdot 3\rho_k(z)$ then k roots in $D(z, 3\rho_k(z)) \subseteq D(z, \frac{\rho_{k+1}(z)}{3})$.

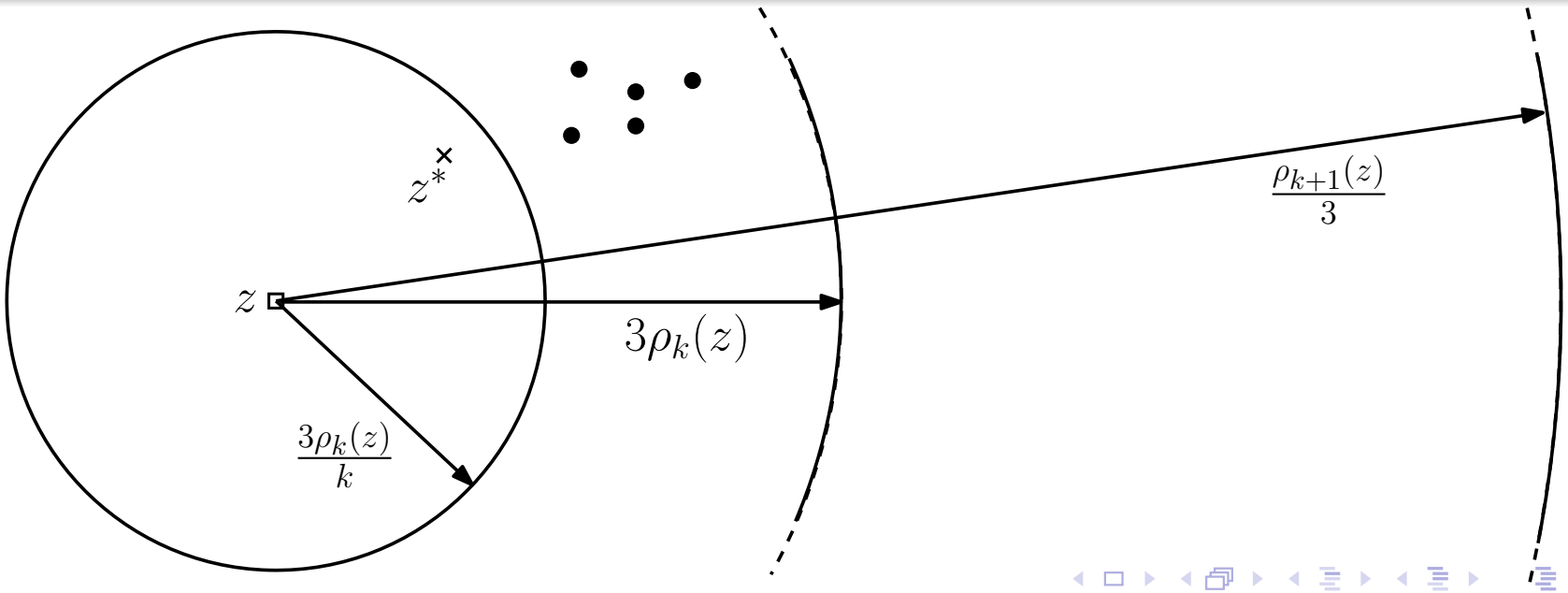


Detecting and Approximating Clusters

Ostrowski'34, S.-Batra'15

Let $f(x+z) := \sum_j f_j(z)x^j$, i.e., $f_j(z)$ is the jth Taylor coefficient at z.

- For $k \in [n]$, $\rho_k(z) := \max_{j<k} \left|\frac{f_j(z)}{f_k(z)}\right|^{\frac{1}{(k-j)}}$, $\rho_{k+1}(z) := \min_{j>k} \left|\frac{f_k(z)}{f_j(z)}\right|^{\frac{1}{(j-k)}}$.
- If $\rho_{k+1}(z)/3 > 3 \cdot 3\rho_k(z)$ then k roots in $D(z, 3\rho_k(z)) \subseteq D(z, \frac{\rho_{k+1}(z)}{3})$.
- Newton iteration starting from z converges to a unique root z^* of $f^{(k-1)}$.

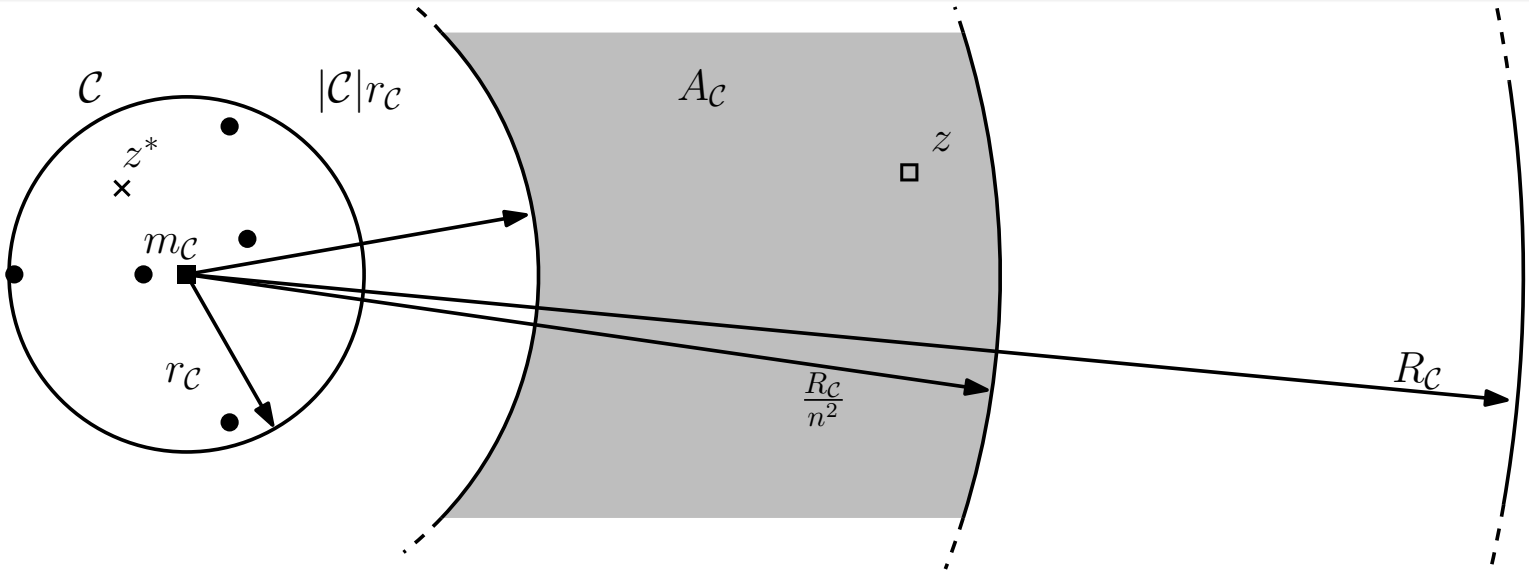


Detecting and Approximating Clusters

Ostrowski'34, S.-Batra'15

Let $f(x+z) := \sum_j f_j(z)x^j$, i.e., $f_j(z)$ is the jth Taylor coefficient at z.

- For $k \in [n]$, $\rho_k(z) := \max_{j<k} \left|\frac{f_j(z)}{f_k(z)}\right|^{\frac{1}{(k-j)}}$, $\rho_{k+1}(z) := \min_{j>k} \left|\frac{f_k(z)}{f_j(z)}\right|^{\frac{1}{(j-k)}}$.
- If $\rho_{k+1}(z)/3 > 3 \cdot 3\rho_k(z)$ then k roots in $D(z, 3\rho_k(z)) \subseteq D(z, \frac{\rho_{k+1}(z)}{3})$.
- Newton iteration starting from z converges to a unique root z^* of $f^{(k-1)}$.
- For a ssc $\mathcal{C}$ all points in an annulus $A_{\mathcal{C}}$ satisfy $\rho_{k+1}(z)/3 > 3 \cdot 3\rho_k(z)$.



The algorithm NewtonIsol

NewtonIsol(f, I_0)

1. Initialize $Q \leftarrow \{I_0\}$. While Q is not empty
   Remove an interval $I = (a, b)$ from Q. $m \leftarrow (a+b)/2$.
   If $C_0(I) \vee C_1(I)$ then add I to $\mathcal{P}$.

   else
   Push (a, m) and (m, b) into Q.
3. Output $\mathcal{P}$.

The algorithm NewtonIsol

NewtonIsol(f, I_0)

1. Initialize $Q \leftarrow \{I_0\}$. While Q is not empty
   Remove an interval $I = (a, b)$ from Q. $m \leftarrow (a+b)/2$.
   If $C_0(I) \vee C_1(I)$ then add I to $\mathcal{P}$.
   else if there is a k s.t. $\rho_{k+1}(m) \gtrsim \rho_k(m)$ and

   else
      Push (a, m) and (m, b) into Q.
3. Output $\mathcal{P}$.

The algorithm NewtonIsol

NewtonIsol(f, I_0)

1. Initialize $Q \leftarrow \{I_0\}$. While Q is not empty
   Remove an interval $I = (a, b)$ from Q. $m \leftarrow (a+b)/2$.
   If $C_0(I) \vee C_1(I)$ then add I to $\mathcal{P}$.
   else if there is a k s.t. $\rho_{k+1}(m) \gtrsim \rho_k(m)$ and
   for the smallest such k, $I \subseteq D(m, \rho_{k+1}(m)/3)$ then

   else
   Push (a, m) and (m, b) into Q.
3. Output $\mathcal{P}$.

The algorithm NewtonIsol

NewtonIsol(f, I_0)

1. Initialize $Q \leftarrow \{I_0\}$. While Q is not empty
   Remove an interval $I = (a, b)$ from Q. $m \leftarrow (a+b)/2$.
   If $C_0(I) \vee C_1(I)$ then add I to $\mathcal{P}$.
   else if there is a k s.t. $\rho_{k+1}(m) \gtrsim \rho_k(m)$ and
      for the smallest such k, $I \subseteq D(m, \rho_{k+1}(m)/3)$ then
      $z_0 := m$, $g := f^{(k-1)}$, $i := 0$.

   else
      Push (a, m) and (m, b) into Q.
3. Output $\mathcal{P}$.

The algorithm NewtonIsol

NewtonIsol(f, I_0)

1. Initialize $Q \leftarrow \{I_0\}$. While Q is not empty
   Remove an interval $I = (a, b)$ from Q. $m \leftarrow (a+b)/2$.
   If $C_0(I) \vee C_1(I)$ then add I to $\mathcal{P}$.
   else if there is a k s.t. $\rho_{k+1}(m) \gtrsim \rho_k(m)$ and
      for the smallest such k, $I \subseteq D(m, \rho_{k+1}(m)/3)$ then
      $z_0 := m$, $g := f^{(k-1)}$, $i := 0$.
      While $\rho_k(z_i) \leq 2^{1-2^i} \rho_k(z_0)$

   else
      Push (a, m) and (m, b) into Q.
3. Output $\mathcal{P}$.

The algorithm NewtonIsol

NewtonIsol(f, I_0)

1. Initialize $Q \leftarrow \{I_0\}$. While Q is not empty
   Remove an interval $I = (a, b)$ from Q. $m \leftarrow (a+b)/2$.
   If $C_0(I) \vee C_1(I)$ then add I to $\mathcal{P}$.
   else if there is a k s.t. $\rho_{k+1}(m) \gtrsim \rho_k(m)$ and
   for the smallest such k, $I \subseteq D(m, \rho_{k+1}(m)/3)$ then
   $z_0 := m$, $g := f^{(k-1)}$, $i := 0$.
   While $\rho_k(z_i) \leq 2^{1-2^i} \rho_k(z_0)$
   $z_{i+1} := z_i - g(z_i)/g'(z_i)$; $i := i+1$.

   else
   Push (a, m) and (m, b) into Q.
3. Output $\mathcal{P}$.

The algorithm NewtonIsol

NewtonIsol(f, I_0)

1. Initialize $Q \leftarrow \{I_0\}$. While Q is not empty
   Remove an interval $I = (a, b)$ from Q. $m \leftarrow (a+b)/2$.
   If $C_0(I) \vee C_1(I)$ then add I to $\mathcal{P}$.
   else if there is a k s.t. $\rho_{k+1}(m) \gtrsim \rho_k(m)$ and
   for the smallest such k, $I \subseteq D(m, \rho_{k+1}(m)/3)$ then
   $z_0 := m$, $g := f^{(k-1)}$, $i := 0$.
   While $\rho_k(z_i) \leq 2^{1-2^i} \rho_k(z_0)$
   $z_{i+1} := z_i - g(z_i)/g'(z_i)$; $i := i+1$.
   $J := [z_{i-1} \pm 3\rho_k(z_{i-1})]$.

   else
   Push (a, m) and (m, b) into Q.
3. Output $\mathcal{P}$.

The algorithm NewtonIsol

NewtonIsol(f, I_0)

1. Initialize $Q \leftarrow \{I_0\}$. While Q is not empty
   Remove an interval $I = (a, b)$ from Q. $m \leftarrow (a+b)/2$.
   If $C_0(I) \vee C_1(I)$ then add I to $\mathcal{P}$.
   else if there is a k s.t. $\rho_{k+1}(m) \gtrsim \rho_k(m)$ and
      for the smallest such k, $I \subseteq D(m, \rho_{k+1}(m)/3)$ then
      $z_0 := m$, $g := f^{(k-1)}$, $i := 0$.
      While $\rho_k(z_i) \leq 2^{1-2^i} \rho_k(z_0)$
         $z_{i+1} := z_i - g(z_i)/g'(z_i)$; $i := i+1$.
      $J := [z_{i-1} \pm 3\rho_k(z_{i-1})]$.
      If $w(J) < w(I)/2$ and $J \cap I_0 \neq \emptyset$ then

   else
      Push (a, m) and (m, b) into Q.
3. Output $\mathcal{P}$.

The algorithm NewtonIsol

NewtonIsol(f, I_0)

1. Initialize $Q \leftarrow \{I_0\}$. While Q is not empty
   Remove an interval $I = (a, b)$ from Q. $m \leftarrow (a+b)/2$.
   If $C_0(I) \vee C_1(I)$ then add I to $\mathcal{P}$.
   else if there is a k s.t. $\rho_{k+1}(m) \gtrsim \rho_k(m)$ and
      for the smallest such k, $I \subseteq D(m, \rho_{k+1}(m)/3)$ then
      $z_0 := m$, $g := f^{(k-1)}$, $i := 0$.
      While $\rho_k(z_i) \leq 2^{1-2^i} \rho_k(z_0)$
         $z_{i+1} := z_i - g(z_i)/g'(z_i)$; $i := i+1$.
      $J := [z_{i-1} \pm 3\rho_k(z_{i-1})]$.
      If $w(J) < w(I)/2$ and $J \cap I_0 \neq \emptyset$ then
      Add J to Q. Remove all intervals in Q that intersect $D(m, \rho_{k+1}(m)/3)$
   else
      Push (a, m) and (m, b) into Q.
3. Output $\mathcal{P}$.

The algorithm NewtonIsol

NewtonIsol(f, I_0)

1. Initialize $Q \leftarrow \{I_0\}$. While Q is not empty
   Remove an interval $I = (a, b)$ from Q. $m \leftarrow (a+b)/2$.
   If $C_0(I) \vee C_1(I)$ then add I to $\mathcal{P}$.
   else if there is a k s.t. $\rho_{k+1}(m) \gtrsim \rho_k(m)$ and
      for the smallest such k, $I \subseteq D(m, \rho_{k+1}(m)/3)$ then
      $z_0 := m$, $g := f^{(k-1)}$, $i := 0$.
      While $\rho_k(z_i) \leq 2^{1-2^i} \rho_k(z_0)$
         $z_{i+1} := z_i - g(z_i)/g'(z_i)$; $i := i+1$.
      $J := [z_{i-1} \pm 3\rho_k(z_{i-1})]$.
      If $w(J) < w(I)/2$ and $J \cap I_0 \neq \emptyset$ then
      Add J to Q. Remove all intervals in Q that intersect $D(m, \rho_{k+1}(m)/3)$
   else
      Push (a, m) and (m, b) into Q.
3. Output $\mathcal{P}$.

Bounding the size of the tree T_{I_0}

T_{I_0} is a binary tree. Let $L(T_{I_0})$ be its set of leaves. Bound $|L(T_{I_0})|$.

Intervals I in $L(T_{I_0})$ – Two Types

1. Where $C_0(I)$ or $C_1(I)$ holds.
2. I was discarded by the Newton-step.

Observation: If J is the parent of I then both $C_0(J)$ and $C_1(J)$ failed.

Bounding the size of the tree T_{I_0}

T_{I_0} is a binary tree. Let $L(T_{I_0})$ be its set of leaves. Bound $|L(T_{I_0})|$.

Intervals I in $L(T_{I_0})$ – Two Types

1. Where $C_0(I)$ or $C_1(I)$ holds.
2. I was discarded by the Newton-step.

Observation: If J is the parent of I then both $C_0(J)$ and $C_1(J)$ failed.

Continuous Amortization (Burr-Krahmer-Yap'09, Burr'13)

Bounding the size of the tree T_{I_0}

T_{I_0} is a binary tree. Let $L(T_{I_0})$ be its set of leaves. Bound $|L(T_{I_0})|$.

Intervals I in $L(T_{I_0})$ – Two Types

1. Where $C_0(I)$ or $C_1(I)$ holds.
2. I was discarded by the Newton-step.

Observation: If J is the parent of I then both $C_0(J)$ and $C_1(J)$ failed.

Continuous Amortization (Burr-Krahmer-Yap'09, Burr'13)

1. Stopping Function $G : \mathbb{R} \to \mathbb{R}_{\geq 0}$ wrt the predicates C_0, C_1.
   Given J, if $\exists x \in J$ such that $w(J)G(x) \leq 1$ then $C_0(J)$ or $C_1(J)$ holds.

Bounding the size of the tree T_{I_0}

T_{I_0} is a binary tree. Let $L(T_{I_0})$ be its set of leaves. Bound $|L(T_{I_0})|$.

Intervals I in $L(T_{I_0})$ – Two Types

1. Where $C_0(I)$ or $C_1(I)$ holds.
2. I was discarded by the Newton-step.

Observation: If J is the parent of I then both $C_0(J)$ and $C_1(J)$ failed.

Continuous Amortization (Burr-Krahmer-Yap'09, Burr'13)

1. Stopping Function $G : \mathbb{R} \to \mathbb{R}_{\geq 0}$ wrt the predicates C_0, C_1.
   Given J, if $\exists x \in J$ such that $w(J)G(x) \leq 1$ then $C_0(J)$ or $C_1(J)$ holds.
2. If $C_0(J)$ and $C_1(J)$ failed then $\forall\ I \subseteq J$, $2w(I) \geq w(J)$, $2\int_I G(x)dx \geq 1$.

Bounding the size of the tree T_{I_0}

T_{I_0} is a binary tree. Let $L(T_{I_0})$ be its set of leaves. Bound $|L(T_{I_0})|$.

Intervals I in $L(T_{I_0})$ – Two Types

1. Where $C_0(I)$ or $C_1(I)$ holds.
2. I was discarded by the Newton-step.

Observation: If J is the parent of I then both $C_0(J)$ and $C_1(J)$ failed.

Continuous Amortization (Burr-Krahmer-Yap'09, Burr'13)

1. Stopping Function $G : \mathbb{R} \to \mathbb{R}_{\geq 0}$ wrt the predicates C_0, C_1.
   Given J, if $\exists x \in J$ such that $w(J)G(x) \leq 1$ then $C_0(J)$ or $C_1(J)$ holds.
2. If $C_0(J)$ and $C_1(J)$ failed then $\forall\ I \subseteq J$, $2w(I) \geq w(J)$, $2\int_I G(x)dx \geq 1$.
   Proof: $2\int_I G(x)dx \geq 2w(I)\min_{x\in I} G(x) \geq w(J)\min_{x\in I} G(x) \geq 1$.

Bounding the size of the tree T_{I_0}

T_{I_0} is a binary tree. Let $L(T_{I_0})$ be its set of leaves. Bound $|L(T_{I_0})|$.

Intervals I in $L(T_{I_0})$ – Two Types

1. Where $C_0(I)$ or $C_1(I)$ holds.
2. I was discarded by the Newton-step.

Observation: If J is the parent of I then both $C_0(J)$ and $C_1(J)$ failed.

Continuous Amortization (Burr-Krahmer-Yap'09, Burr'13)

1. Stopping Function $G : \mathbb{R} \to \mathbb{R}_{\geq 0}$ wrt the predicates C_0, C_1.
   Given J, if $\exists x \in J$ such that $w(J)G(x) \leq 1$ then $C_0(J)$ or $C_1(J)$ holds.
2. If $C_0(J)$ and $C_1(J)$ failed then $\forall\ I \subseteq J$, $2w(I) \geq w(J)$, $2\int_I G(x)dx \geq 1$.
   Proof: $2\int_I G(x)dx \geq 2w(I)\min_{x\in I} G(x) \geq w(J)\min_{x\in I} G(x) \geq 1$.

Lemma (S.-Batra'15)

$$|L(T_{I_0})| \leq O(n) + 2\int_{I_0 \setminus \cup_C A_C} G(x)dx, \text{ where } C \text{ are ssc.}$$

Bounding treesize for Descartes's rule of signs

Theorem (S.-Batra'15)

$$|L(T_{I_0})| \leq O(n) + 2\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(n \log n), \text{ where } C \text{ are ssc.}$$

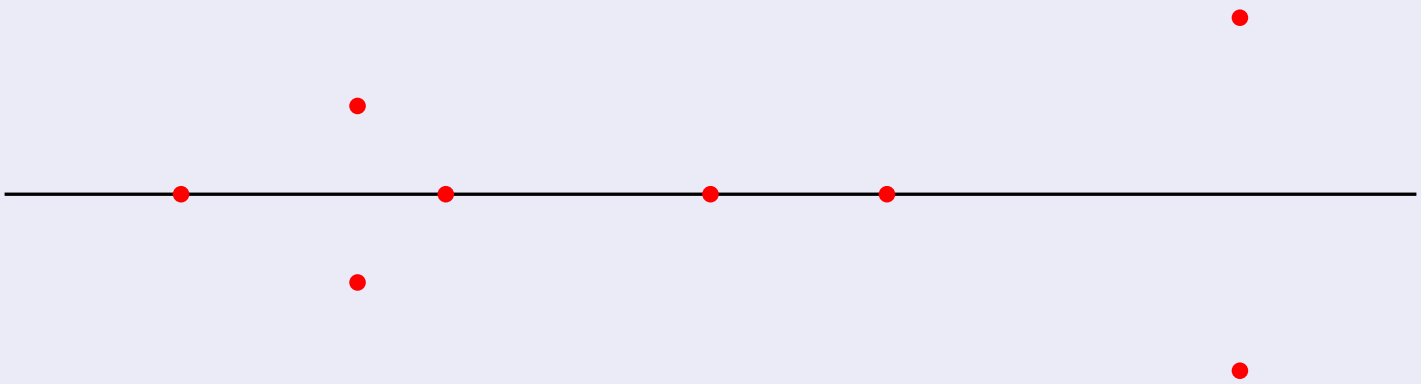
Bounding treesize for Descartes's rule of signs

Theorem (S.-Batra'15)

$$|L(T_{I_0})| \leq O(n) + 2\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(n \log n), \textit{ where } C \textit{ are ssc.}$$

Stopping function for Descartes's rule of signs

1. $d(x, Z(f))$ distance to nearest root.
2. $d_2(x, Z(f))$ distance to second nearest root.



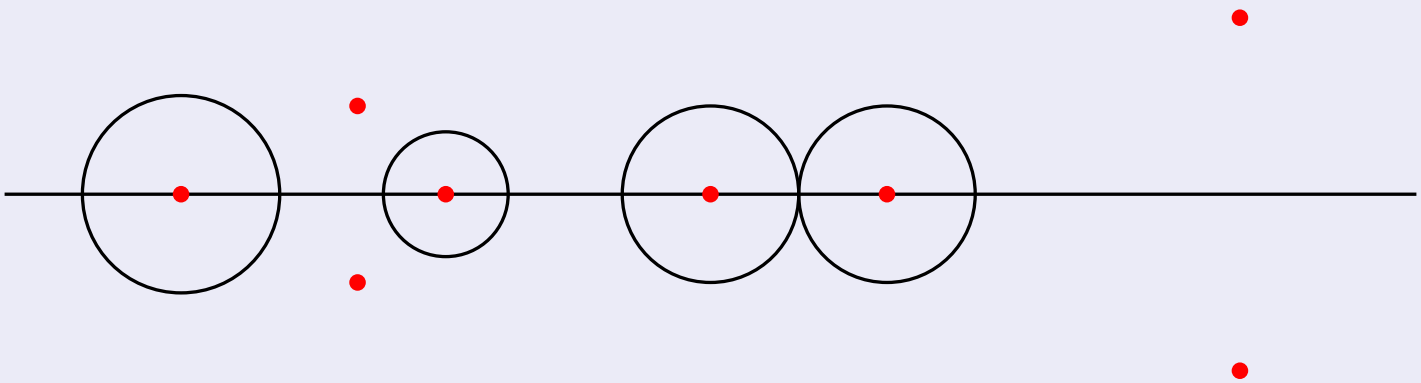
Bounding treesize for Descartes's rule of signs

Theorem (S.-Batra'15)

$$|L(T_{I_0})| \leq O(n) + 2\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(n \log n), \textit{ where } C \textit{ are ssc.}$$

Stopping function for Descartes's rule of signs

1. $d(x, Z(f))$ distance to nearest root.
2. $d_2(x, Z(f))$ distance to second nearest root.



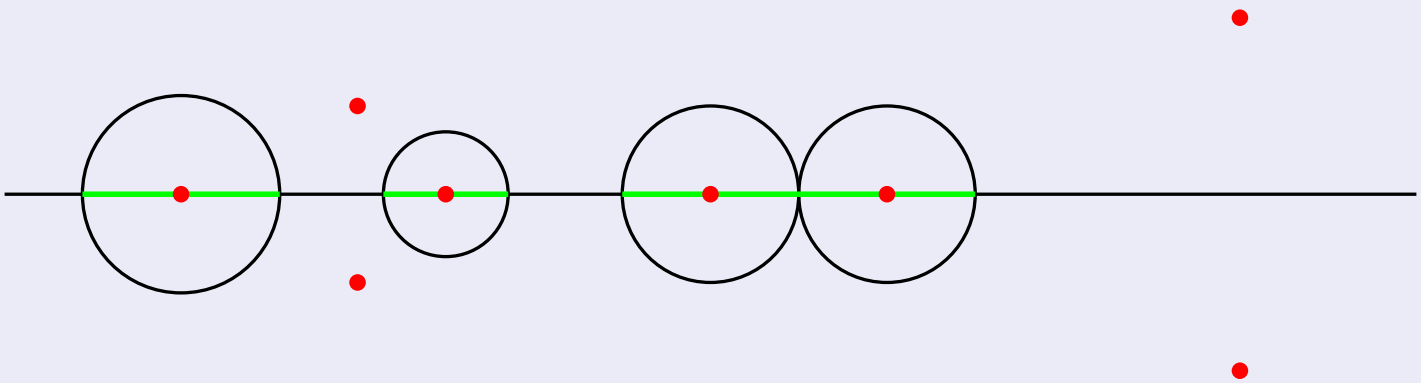
Bounding treesize for Descartes's rule of signs

Theorem (S.-Batra'15)

$$|L(T_{I_0})| \leq O(n) + 2\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(n \log n), \textit{ where } C \textit{ are ssc.}$$

Stopping function for Descartes's rule of signs

1. $d(x, Z(f))$ distance to nearest root.
2. $d_2(x, Z(f))$ distance to second nearest root.
3. $G(x) = 1/d_2(x, Z(f))$ near real roots.



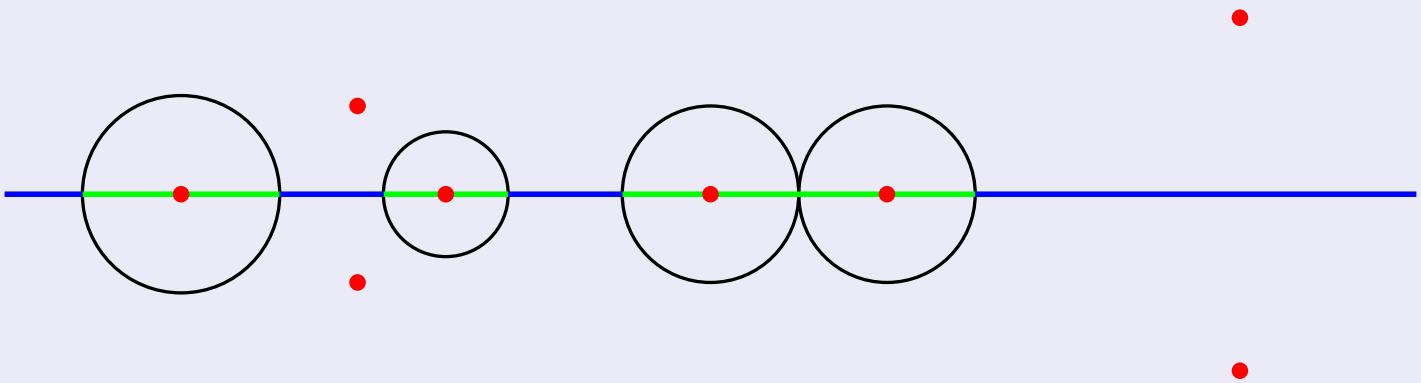
Bounding treesize for Descartes's rule of signs

Theorem (S.-Batra'15)

$$|L(T_{I_0})| \leq O(n) + 2\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(n \log n), \textit{ where } C \textit{ are ssc.}$$

Stopping function for Descartes's rule of signs

1. $d(x, Z(f))$ distance to nearest root.
2. $d_2(x, Z(f))$ distance to second nearest root.
3. $G(x) = 1/d_2(x, Z(f))$ near real roots.
4. $G(x) = 1/d(x, Z(f))$ everywhere else.



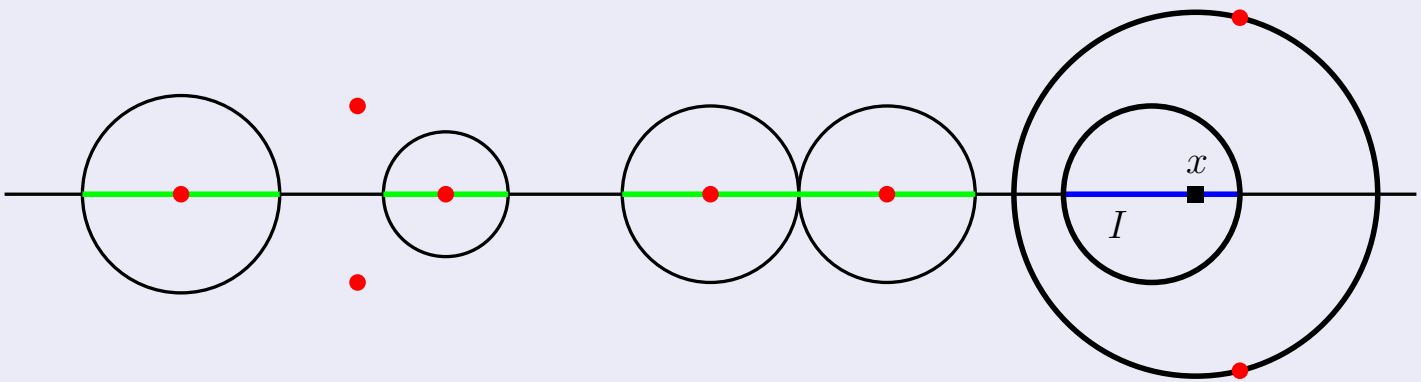
Bounding treesize for Descartes's rule of signs

Theorem (S.-Batra'15)

$$|L(T_{I_0})| \leq O(n) + 2\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(n \log n), \text{ where } C \text{ are ssc.}$$

Stopping function for Descartes's rule of signs

1. $d(x, Z(f))$ distance to nearest root.
2. $d_2(x, Z(f))$ distance to second nearest root.
3. $G(x) = 1/d_2(x, Z(f))$ near real roots.
4. $G(x) = 1/d(x, Z(f))$ everywhere else.
5. If $\exists x \in I$ s.t. $w(I)G(x) \leq 1$, i.e., $w(I) \leq d(x, Z(f))$ then $C_0(I)$ holds.



Bounding treesize for Descartes's rule of signs

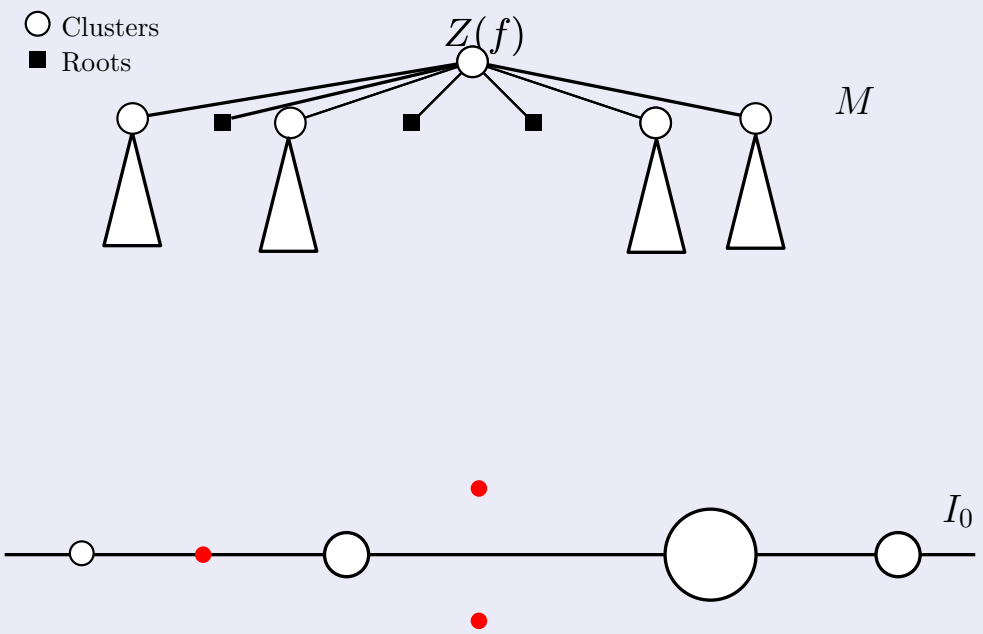
Claim

$\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(|T| \log n)$, where C are ssc, T is cluster tree?

Bounding treesize for Descartes's rule of signs

Claim

$\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(|T| \log n)$, where C are ssc, T is cluster tree?

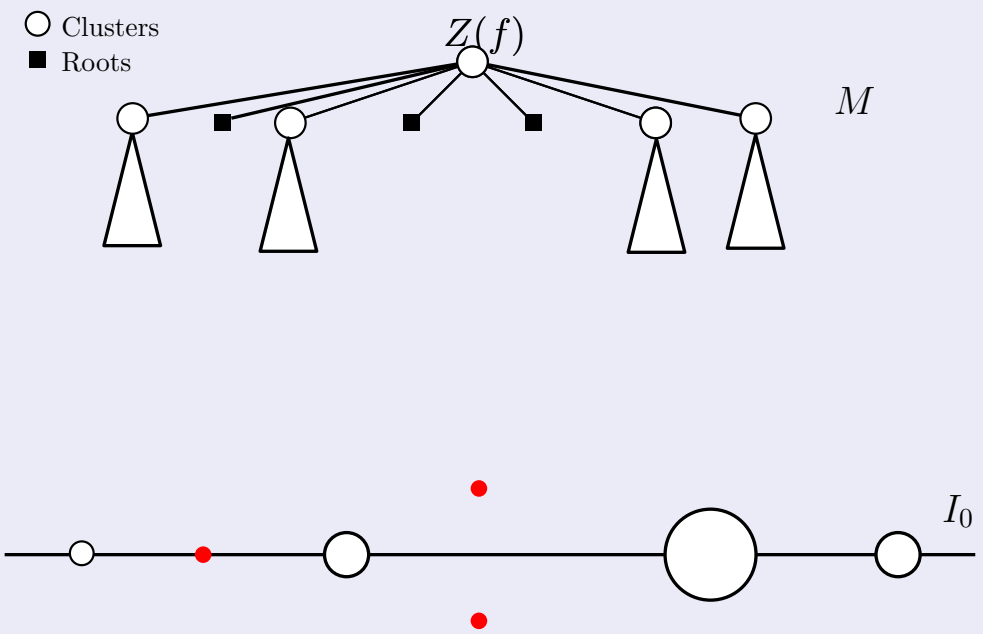


– Cluster C, $I_C := [m_C \pm 2r_C]$.

Bounding treesize for Descartes's rule of signs

Claim

$\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(|T| \log n)$, where C are ssc, T is cluster tree?

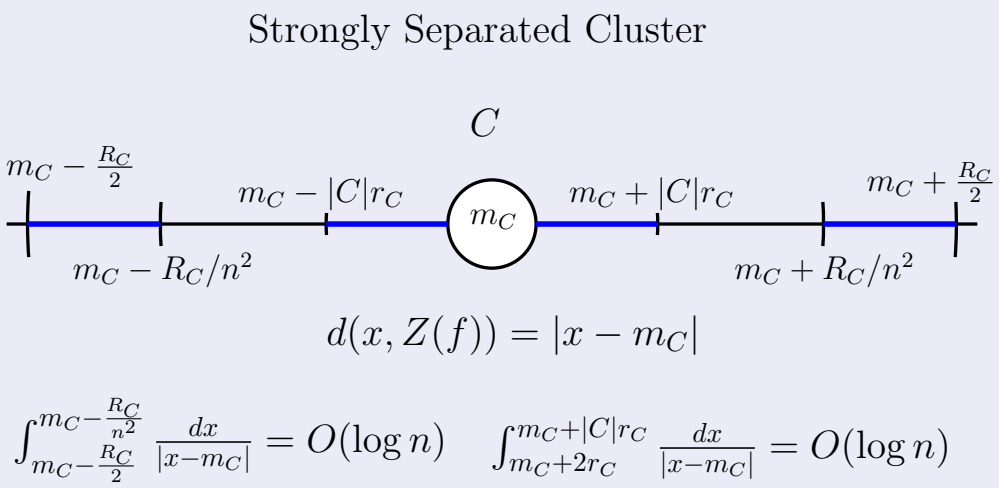


– Cluster C, $I_C := [m_C \pm 2r_C]$.
– $\int_{I_C \setminus \cup_{C'} A_{C'}} G(x)dx = O(|T_C| \log n)$.

Bounding treesize for Descartes's rule of signs

Claim

$\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(|T| \log n)$, where C are ssc, T is cluster tree?

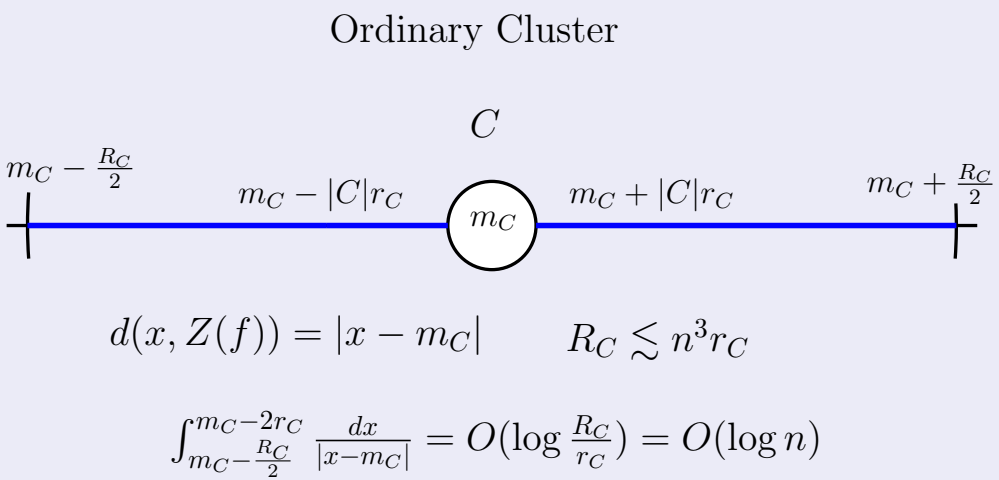


– Cluster C, $I_C := [m_C \pm 2r_C]$.
– $\int_{I_C \setminus \cup_{C'} A_{C'}} G(x)dx = O(|T_C| \log n)$.
– Near C, integral is $O(\log n)$.

Bounding treesize for Descartes's rule of signs

Claim

$\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(|T| \log n)$, where C are ssc, T is cluster tree?

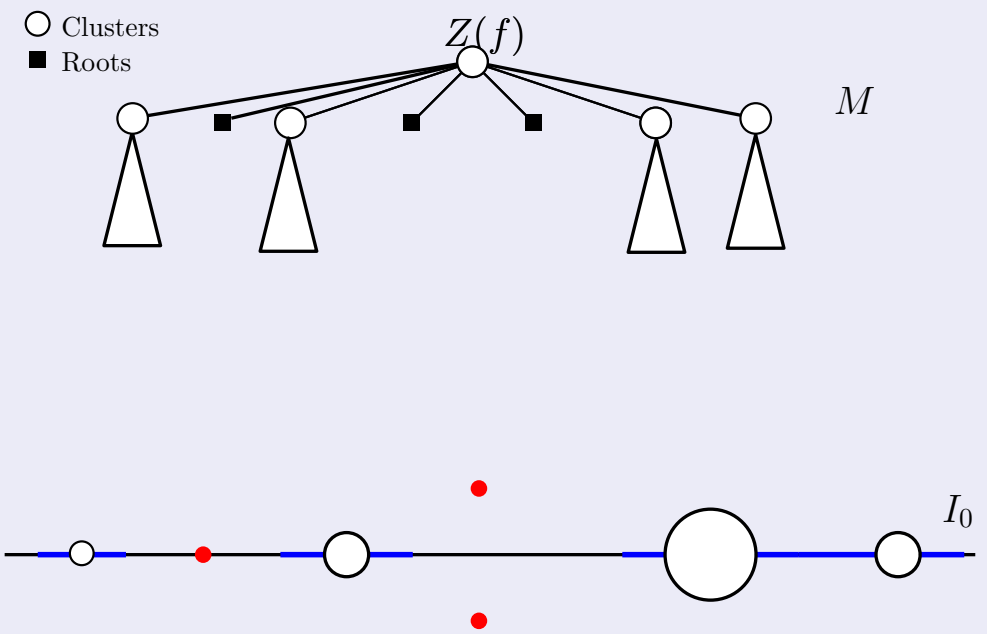


- Cluster C, $I_C := [m_C \pm 2r_C]$.
- $\int_{I_C \setminus \cup_{C'} A_{C'}} G(x)dx = O(|T_C| \log n)$.
- Near C, integral is $O(\log n)$.

Bounding treesize for Descartes's rule of signs

Claim

$\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(|T| \log n)$, where C are ssc, T is cluster tree?

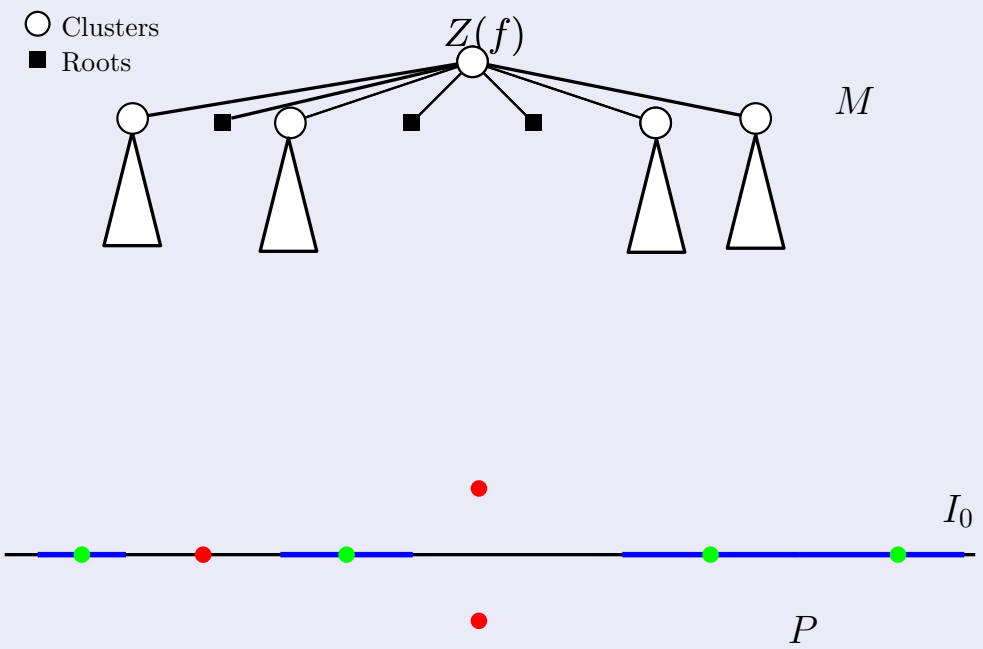


– Cluster C, $I_C := [m_C \pm 2r_C]$.
– $\int_{I_C \setminus \cup_{C'} A_{C'}} G(x)dx = O(|T_C| \log n)$.
– Near C, integral is $O(\log n)$.
– $|M| + \sum_{C \in M} |T_C| = O(|T|)$.

Bounding treesize for Descartes's rule of signs

Claim

$\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(|T| \log n)$, where C are ssc, T is cluster tree?

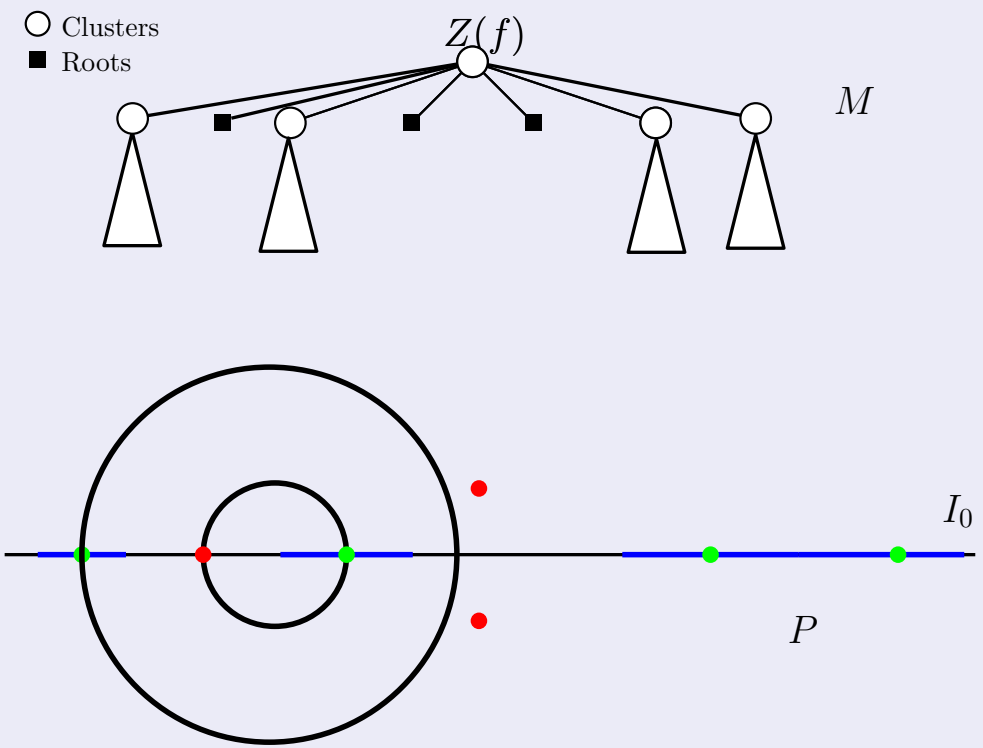


– Cluster C, $I_C := [m_C \pm 2r_C]$.
– $\int_{I_C \setminus \cup_{C'} A_{C'}} G(x)dx = O(|T_C| \log n)$.
– Near C, integral is $O(\log n)$.
– $|M| + \sum_{C \in M} |T_C| = O(|T|)$.
– Shrink clusters to points to get P.

Bounding treesize for Descartes's rule of signs

Claim

$\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(|T| \log n)$, where C are ssc, T is cluster tree?

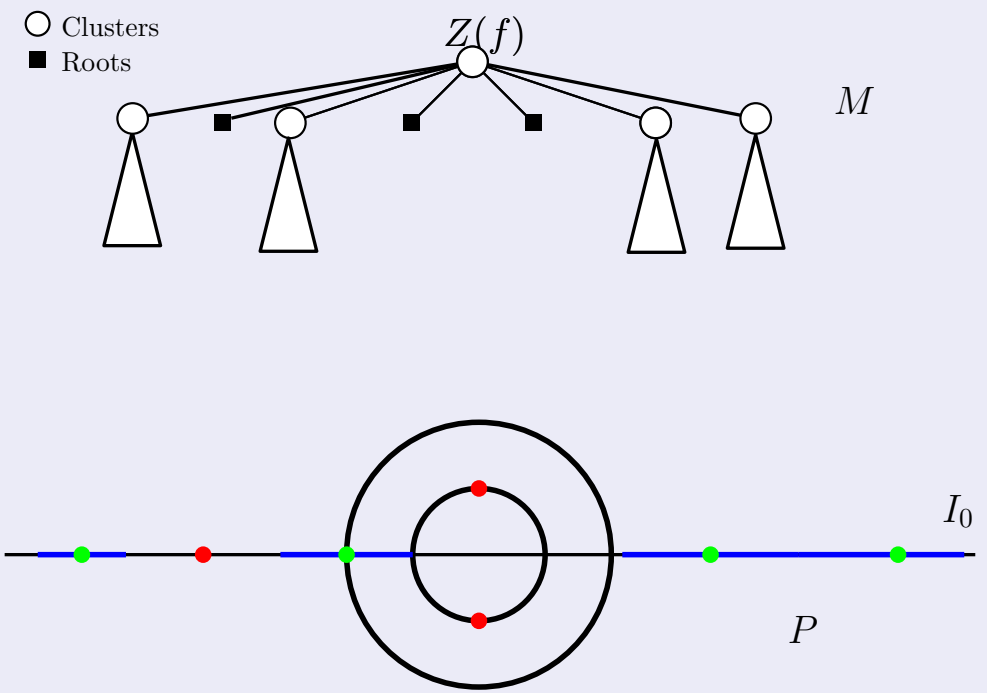


- Cluster C, $I_C := [m_C \pm 2r_C]$.
- $\int_{I_C \setminus \cup_{C'} A_{C'}} G(x)dx = O(|T_C| \log n)$.
- Near C, integral is $O(\log n)$.
- $|M| + \sum_{C \in M} |T_C| = O(|T|)$.
- Shrink clusters to points to get P.
- Resulting pointset is dense.

Bounding treesize for Descartes's rule of signs

Claim

$\int_{I_0 \setminus \cup_C A_C} G(x)dx = O(|T| \log n)$, where C are ssc, T is cluster tree?

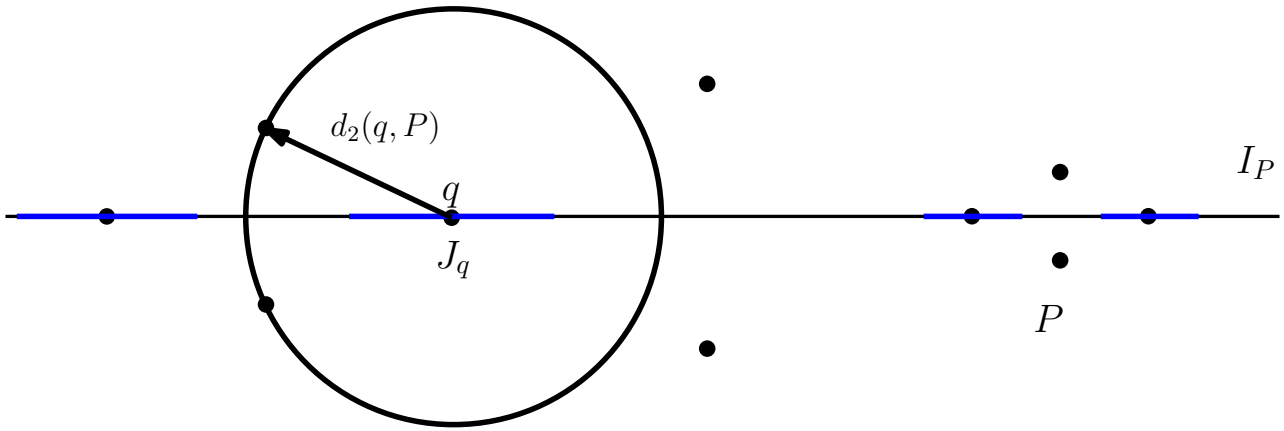


- Cluster C, $I_C := [m_C \pm 2r_C]$.
- $\int_{I_C \setminus \cup_{C'} A_{C'}} G(x)dx = O(|T_C| \log n)$.
- Near C, integral is $O(\log n)$.
- $|M| + \sum_{C \in M} |T_C| = O(|T|)$.
- Shrink clusters to points to get P.
- Resulting pointset is dense.

The Main Lemma

Bound for dense pointset P

1. For a $q \in P \cap \mathbb{R}$, $J_q := [q \pm d_2(q, P)/2]$.
2. $I_P := [m_P \pm r_P]$
3. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{1}{d(x,P)} = O(|P| \log |P|)$.



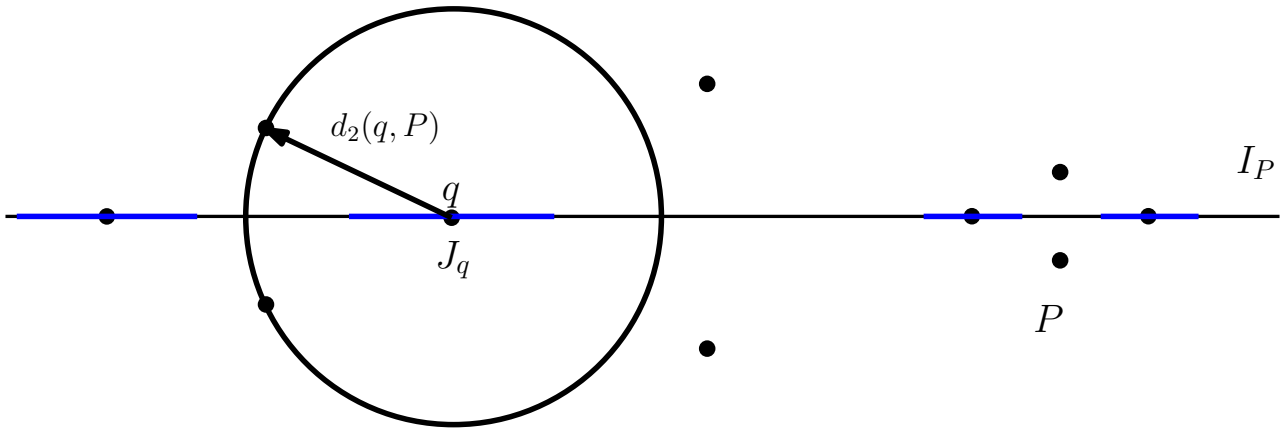
The Main Lemma

Bound for dense pointset P

1. For a $q \in P \cap \mathbb{R}$, $J_q := [q \pm d_2(q, P)/2]$.
2. $I_P := [m_P \pm r_P]$
3. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{1}{d(x,P)} = O(|P| \log |P|)$.

1. V_q be the real Voronoi region of q.
2. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{dx}{d(x,P)}$

.



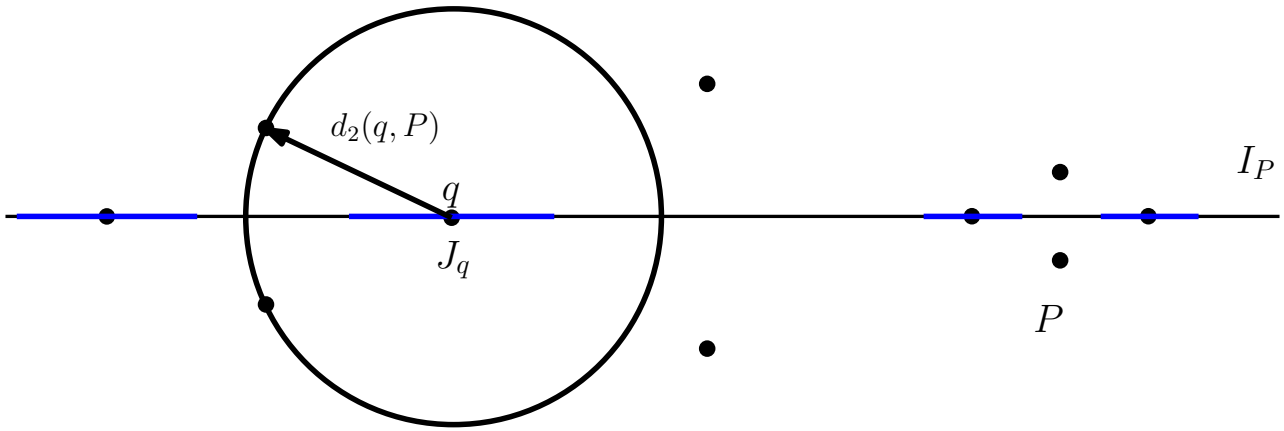
The Main Lemma

Bound for dense pointset P

1. For a $q \in P \cap \mathbb{R}$, $J_q := [q \pm d_2(q,P)/2]$.
2. $I_P := [m_P \pm r_P]$
3. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{1}{d(x,P)} = O(|P| \log |P|)$.

1. V_q be the real Voronoi region of q.
2. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{dx}{d(x,P)} = \sum_{q \in P} \int_{V_q} \frac{dx}{|x-q|}$

.

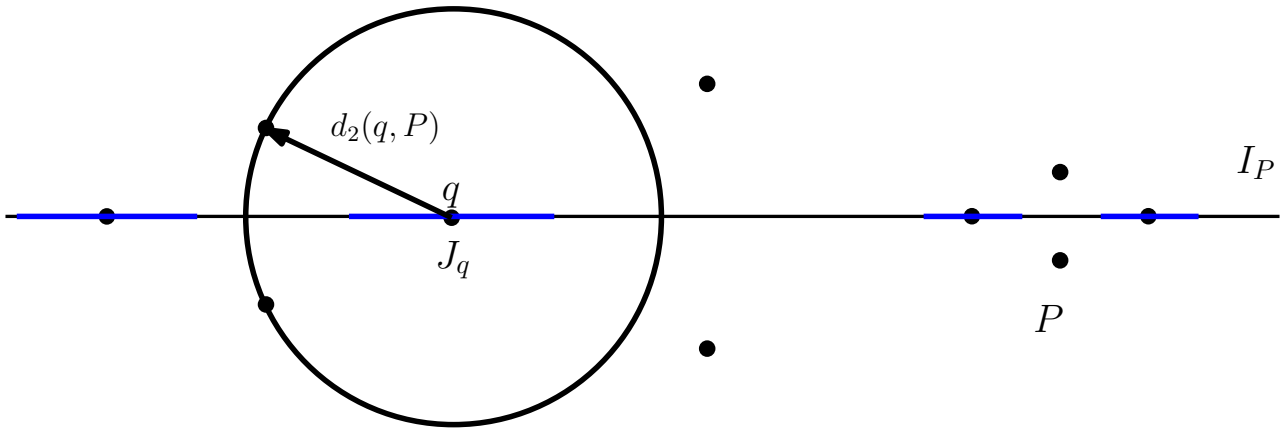


The Main Lemma

Bound for dense pointset P

1. For a $q \in P \cap \mathbb{R}$, $J_q := [q \pm d_2(q,P)/2]$.
2. $I_P := [m_P \pm r_P]$
3. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{1}{d(x,P)} = O(|P| \log |P|)$.

1. V_q be the real Voronoi region of q.
2. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{dx}{d(x,P)} = \sum_{q \in P} \int_{V_q} \frac{dx}{|x-q|} = \sum_{q \in P} O(\log \frac{w(I_P)}{d_2(q,P)})$.

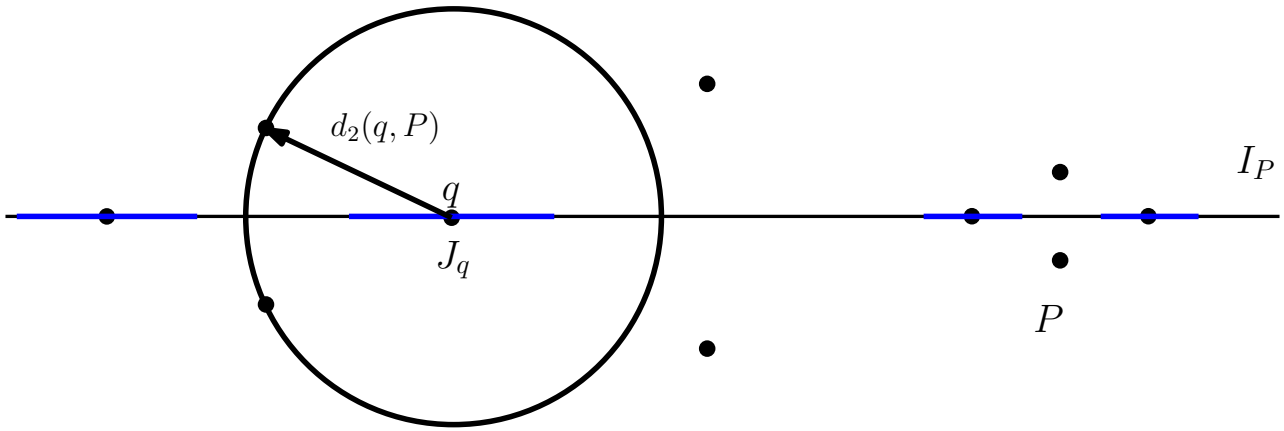


The Main Lemma

Bound for dense pointset P

1. For a $q \in P \cap \mathbb{R}$, $J_q := [q \pm d_2(q,P)/2]$.
2. $I_P := [m_P \pm r_P]$
3. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{1}{d(x,P)} = O(|P| \log |P|)$.

1. V_q be the real Voronoi region of q.
2. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{dx}{d(x,P)} = \sum_{q \in P} \int_{V_q} \frac{dx}{|x-q|} = \sum_{q \in P} O(\log \frac{w(I_P)}{d_2(q,P)})$.
3. for all $q \in P$, $w(I_P) \leq 3^{|P|} d_2(q,P)$.

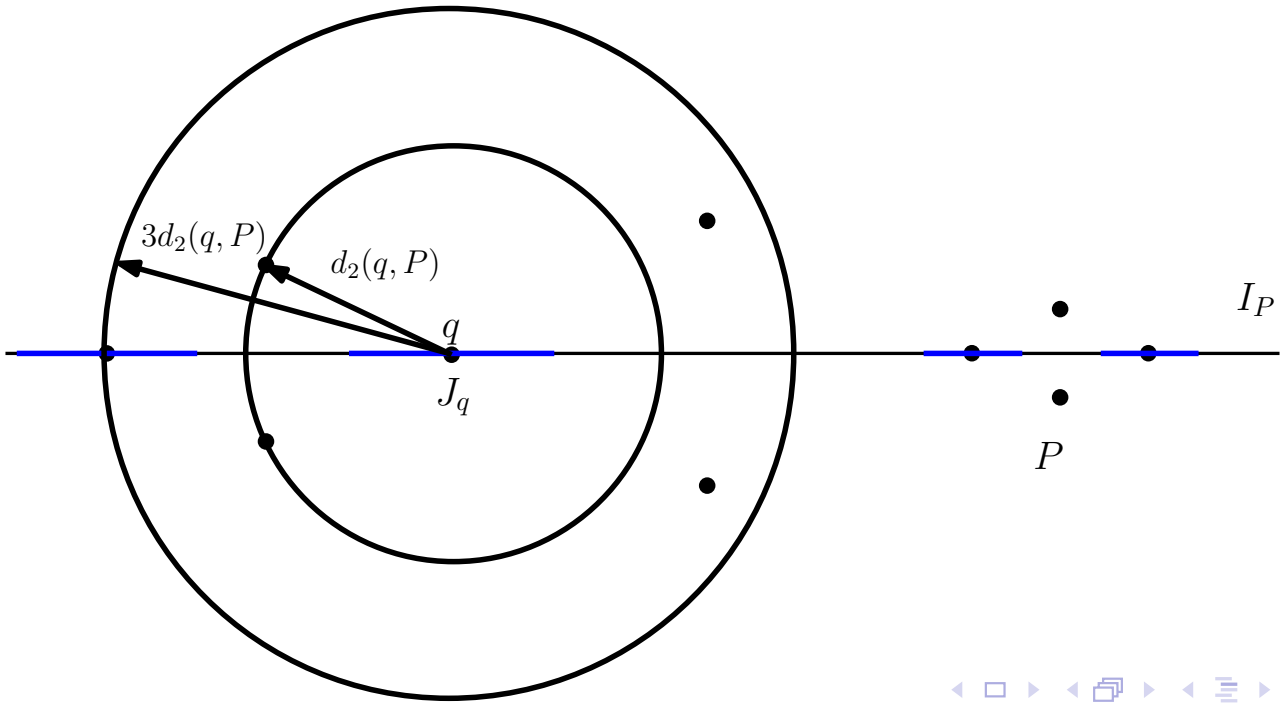


The Main Lemma

Bound for dense pointset P

1. For a $q \in P \cap \mathbb{R}$, $J_q := [q \pm d_2(q,P)/2]$.
2. $I_P := [m_P \pm r_P]$
3. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{1}{d(x,P)} = O(|P| \log |P|)$.

1. V_q be the real Voronoi region of q.
2. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{dx}{d(x,P)} = \sum_{q \in P} \int_{V_q} \frac{dx}{|x-q|} = \sum_{q \in P} O(\log \frac{w(I_P)}{d_2(q,P)})$.
3. for all $q \in P$, $w(I_P) \leq 3^{|P|} d_2(q,P)$.

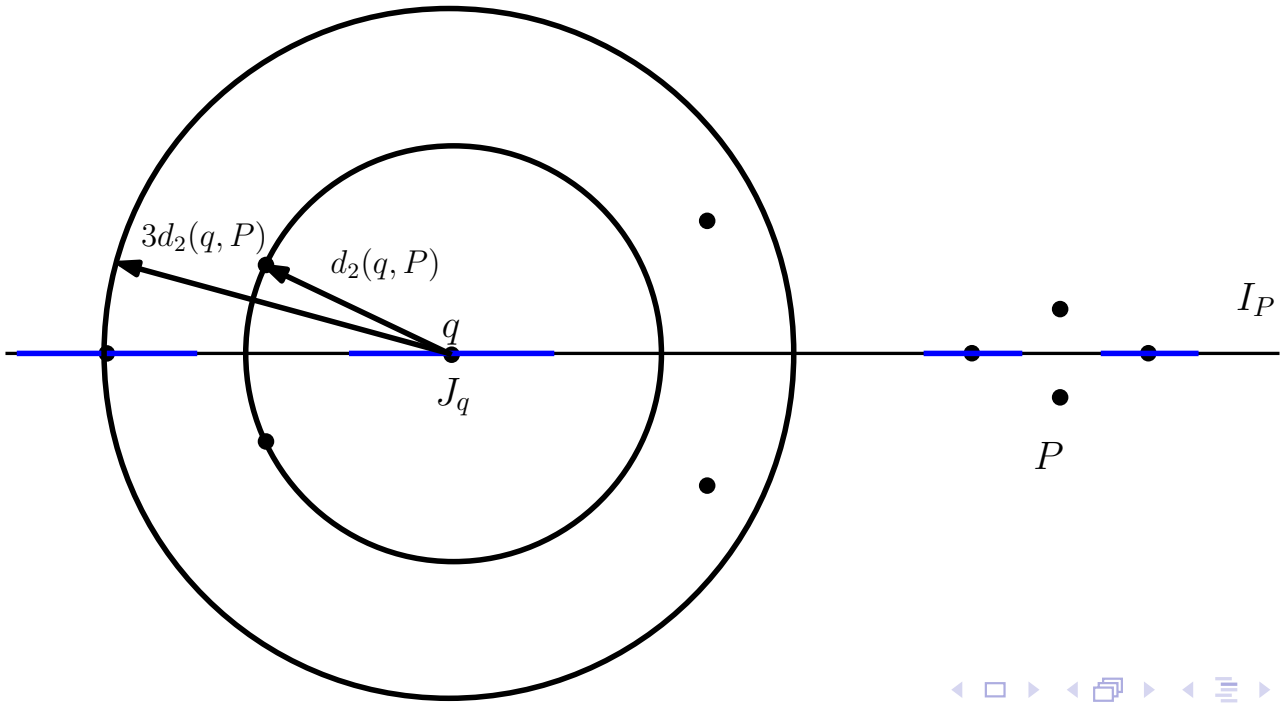


The Main Lemma

Bound for dense pointset P

1. For a $q \in P \cap \mathbb{R}$, $J_q := [q \pm d_2(q,P)/2]$.
2. $I_P := [m_P \pm r_P]$
3. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{1}{d(x,P)} = O(|P| \log |P|)$.

1. V_q be the real Voronoi region of q.
2. $\int_{I_P \setminus \cup_{q \in P \cap \mathbb{R}} J_q} \frac{dx}{d(x,P)} = \sum_{q \in P} \int_{V_q} \frac{dx}{|x-q|} = \sum_{q \in P} O(\log \frac{w(I_P)}{d_2(q,P)}) = O(|P|^2)$.
3. for all $q \in P$, $w(I_P) \leq 3^{|P|} d_2(q,P)$.



Conclusion and Further directions

Summary

1. Provide a subroutine that speedsup any subdivision algorithm for real root isolation.
2. Independent of the inclusion-exclusion predicates.
3. Number of bisections for Descartes's rule of signs and sturms method is $O(n \log n)$.
4. Highlights the geometry of root clusters.

Conclusion and Further directions

Summary

1. Provide a subroutine that speedsup any subdivision algorithm for real root isolation.
2. Independent of the inclusion-exclusion predicates.
3. Number of bisections for Descartes's rule of signs and sturms method is $O(n \log n)$.
4. Highlights the geometry of root clusters.

Further Directions

1. Root isolation in the complex plane.
2. Bit complexity of the algorithm where coefficients are in $\mathbb{R}$.

Conclusion and Further directions

Summary

1. Provide a subroutine that speedsup any subdivision algorithm for real root isolation.
2. Independent of the inclusion-exclusion predicates.
3. Number of bisections for Descartes's rule of signs and sturms method is $O(n \log n)$.
4. Highlights the geometry of root clusters.

Further Directions

1. Root isolation in the complex plane.
2. Bit complexity of the algorithm where coefficients are in $\mathbb{R}$.

Thank You!