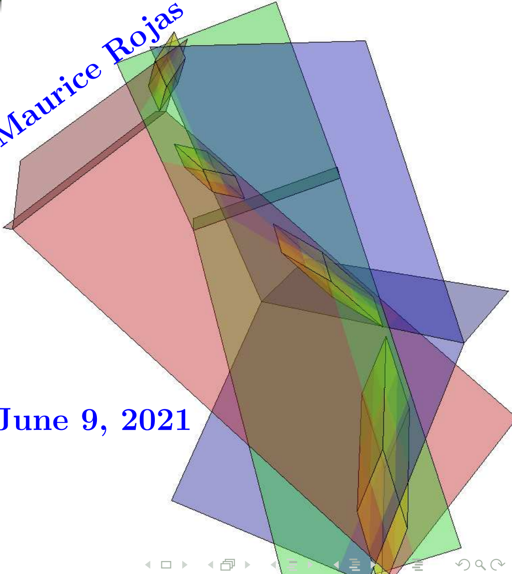
Counting Real Roots in Polynomial-Time for Systems Supported on Circuits

J. Maurice Rojas



June 9, 2021

1 Motivation & Background

1. Motivation & Background

2. Fewnomials and Number Theory

3. Faster Real Root Counting for Circuit Systems

We'd like to solve...

...systems like:

$$\begin{pmatrix}
2x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60}+x_1^{76}x_2^{240}x_4^{41}x_5+x_1^{74}x_2^{179}x_3^{25}x_5^{57}+x_1^{25}x_2^{203}x_3^{44}x_4+x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68}-37137x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25}-\frac{9}{2}x_3^{166}x_4^{68}x_5^{343}, \\
x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60}+2x_1^{76}x_2^{240}x_4^{41}x_5+x_1^{74}x_2^{179}x_3^{25}x_5^{57}+x_1^{25}x_2^{203}x_3^{44}x_4+x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68}-24849x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25}-\frac{21}{4}x_3^{166}x_4^{68}x_5^{343}, \\
x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60}+x_1^{76}x_2^{240}x_4^{41}x_5+2x_1^{74}x_2^{179}x_3^{25}x_5^{57}+x_1^{25}x_2^{203}x_3^{44}x_4+x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68}-21009x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25}-\frac{21}{4}x_3^{166}x_4^{68}x_5^{343}, \\
x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60}+x_1^{76}x_2^{240}x_4^{41}x_5+x_1^{74}x_2^{179}x_3^{25}x_5^{57}+2x_1^{25}x_2^{203}x_3^{44}x_4+x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68}-20769x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25}-\frac{21}{4}x_3^{166}x_4^{68}x_5^{343}, \\
x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60}+x_1^{76}x_2^{240}x_4^{41}x_5+x_1^{74}x_2^{179}x_3^{25}x_5^{57}+x_1^{25}x_2^{203}x_3^{44}x_4+2x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68}-20754x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25}-\frac{21}{4}x_3^{166}x_4^{68}x_5^{343}
\end{pmatrix}$$

quickly.

We'd like to solve...

...systems like:

$$\begin{pmatrix}
2x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 37137x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{9}{2}x_3^{166}x_4^{68}x_5^{343}, \\
x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + 2x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 24849x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343}, \\
x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + 2x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 21009x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343}, \\
x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + 2x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20769x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343}, \\
x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + 2x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20754x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343}
\end{pmatrix}$$

quickly.

But how?

We'd like to solve...

...systems like:

$$\left(2x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 37137x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{9}{2}x_3^{166}x_4^{68}x_5^{343},\right.$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + 2x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 24849x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + 2x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 21009x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + 2x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20769x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$\left.x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + 2x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20754x_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343}\right)$$

quickly.

But how?

More importantly, why?

Polynomials Equations and Complexity Theory

- Complexity theory is the friend of cryptography,

Polynomials Equations and Complexity Theory

- Complexity theory is the friend of cryptography, as well as the difficult cousin of learning theory

Polynomials Equations and Complexity Theory

- Complexity theory is the friend of cryptography, as well as the difficult cousin of learning theory and numerical computation.

Polynomials Equations and Complexity Theory

- Complexity theory is the friend of cryptography, as well as the difficult cousin of learning theory and numerical computation.
- Polynomials are really just circuits

Polynomials Equations and Complexity Theory

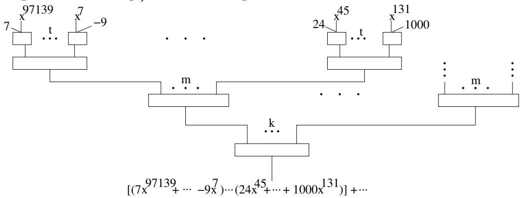
- Complexity theory is the friend of cryptography, as well as the difficult cousin of learning theory and numerical computation.
- Polynomials are really just circuits (in the sense of electrical engineering)

Polynomials Equations and Complexity Theory

- Complexity theory is the friend of cryptography, as well as the difficult cousin of learning theory and numerical computation.
- Polynomials are really just circuits (in the sense of electrical engineering) in disguise...

Polynomials Equations and Complexity Theory

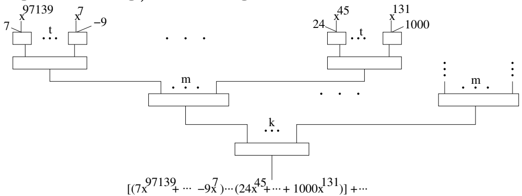
- Complexity theory is the friend of cryptography, as well as the difficult cousin of learning theory and numerical computation.
- Polynomials are really just circuits (in the sense of electrical engineering) in disguise...



Polynomials Equations and Complexity Theory

• Complexity theory is the friend of cryptography, as well as the difficult cousin of learning theory and numerical computation.

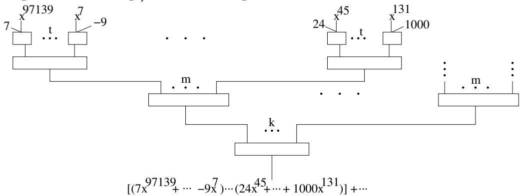
• Polynomials are really just circuits (in the sense of electrical engineering) in disguise...



Fundamental Idea: Theorems about existence of roots are close to the **P** vs. **NP** Problem.

Polynomials Equations and Complexity Theory

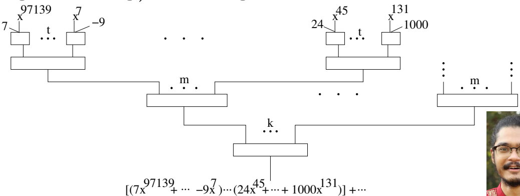
- Complexity theory is the friend of cryptography, as well as the difficult cousin of learning theory and numerical computation.
- Polynomials are really just circuits (in the sense of electrical engineering) in disguise...



Fundamental Idea: Theorems about existence of roots are close to the **P** vs. **NP** Problem. Theorems about the deeper structure of polynomials are close to derandomization,

Polynomials Equations and Complexity Theory

- Complexity theory is the friend of cryptography, as well as the difficult cousin of learning theory and numerical computation.
- Polynomials are really just circuits (in the sense of electrical engineering) in disguise...



Fundamental Idea: Theorems about existence of roots are close to the **P** vs. **NP** Problem. Theorems about the deeper structure of polynomials are close to derandomization, i.e., the **P** vs. **BPP** Problem [Koiran, '11; Dutta, Saxena, Thierauf, '20].

Descartes' Rule, Biochemistry, and Learning Theory...




Recent work on chemical reaction networks

Descartes' Rule, Biochemistry, and Learning Theory...

Recent work on chemical reaction networks makes serious use of <u>*exact*</u> counting of real roots for sparse systems [Bihan, Dickenstein, Giaroli, Shiu, 2019–2020].

Descartes' Rule, Biochemistry, and Learning Theory...




Recent work on chemical reaction networks makes serious use of *exact* counting of real roots for sparse systems [Bihan, Dickenstein, Giaroli, Shiu, 2019–2020].

•[Ren, Zhang, 2020] applies tropical fewnomials to neural networks...

Descartes' Rule, Biochemistry, and Learning Theory...




Recent work on chemical reaction networks makes serious use of *exact* counting of real roots for sparse systems [Bihan, Dickenstein, Giaroli, Shiu, 2019–2020].

•[Ren, Zhang, 2020] applies tropical fewnomials to neural networks...

[Bihan, Dickenstein, Forsgård, 2020] *Multivariate* version of Descartes' Rule for $(n+2)$-nomial $n \times n$ systems...

Descartes' Rule, Biochemistry, and Learning Theory...




Recent work on chemical reaction networks makes serious use of *exact* counting of real roots for sparse systems [Bihan, Dickenstein, Giaroli, Shiu, 2019–2020].

• [Ren, Zhang, 2020] applies tropical fewnomials to neural networks...

[Bihan, Dickenstein, Forsgård, 2020] *Multivariate* version of Descartes' Rule for $(n+2)$-nomial $n \times n$ systems...

But what about *exact counting*?

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^{\varepsilon}$ *terms,*

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^\varepsilon$ *terms, decides if* f *has a real root,*

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^{\varepsilon}$ *terms, decides if* f *has a real root, using time polynomial in the bit-size of* f*.*

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^{\varepsilon}$ *terms, decides if* f *has a real root, using time polynomial in the bit-size of* f*. Then* $\mathbf{P} = \mathbf{NP}$*.*

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^{\varepsilon}$ *terms, decides if* f *has a real root, using time polynomial in the bit-size of* f*. Then* $\mathbf{P} = \mathbf{NP}$*.*

- We use the Turing (bit) model, and

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^{\varepsilon}$ *terms, decides if* f *has a real root, using time polynomial in the bit-size of* f*. Then* $\mathbf{P} = \mathbf{NP}$.

- We use the Turing (bit) model, and size(f):=*Total number of bits in coefficients*

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^{\varepsilon}$ *terms, decides if* f *has a real root, using time polynomial in the bit-size of* f*. Then* $\mathbf{P} = \mathbf{NP}$*.*

- We use the Turing (bit) model, and **size(f)**:=*Total number of bits in coefficients* and *the exponents of* f.

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^{\varepsilon}$ *terms, decides if* f *has a real root, using time polynomial in the bit-size of* f*. Then* $\mathbf{P} = \mathbf{NP}$*.*

- We use the Turing (bit) model, and **size(f)**:=*Total number of bits in coefficients* and *the exponents of* f.
- The case of $n+1$ terms is a cute exercise for your students:

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^{\varepsilon}$ *terms, decides if* f *has a real root, using time polynomial in the bit-size of* f. *Then* $\mathbf{P} = \mathbf{NP}$.

- We use the Turing (bit) model, and **size(f)**:=*Total number of bits in coefficients* and *the exponents of* f.
- The case of $n+1$ terms is a cute exercise for your students: *An* $O(n)$*-time algorithm*

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^{\varepsilon}$ *terms, decides if* f *has a real root, using time polynomial in the bit-size of* f*. Then* $\mathbf{P} = \mathbf{NP}$*.*

- We use the Turing (bit) model, and **size(f)**:=*Total number of bits in coefficients* and *the exponents of* f.
- The case of $n+1$ terms is a cute exercise for your students: *An* $O(n)$*-time algorithm (assuming affinely independent exponents)*

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^{\varepsilon}$ *terms, decides if* f *has a real root, using time polynomial in the bit-size of* f*. Then* $\mathbf{P} = \mathbf{NP}$.

- We use the Turing (bit) model, and **size(f)**:=*Total number of bits in coefficients* and *the exponents of* f.
- The case of $n + 1$ terms is a cute exercise for your students: *An* $O(n)$*-time algorithm (assuming affinely independent exponents) is possible!*

Detecting Real Roots is Already Hard for 1 Sparse (Multivariate) Polynomial!

Complexity Lower Bound.
[Bihan, Rojas, Stella, 2009] *Fix any* $\varepsilon > 0$ *and suppose there is an algorithm that, for any polynomial* $f \in \mathbb{Q}[x_1, \ldots, x_n]$ *with at most* $n + n^{\varepsilon}$ *terms, decides if* f *has a real root, using time polynomial in the bit-size of* f. *Then* $\mathbf{P} = \mathbf{NP}$.

- We use the Turing (bit) model, and **size(f)**:=*Total number of bits in coefficients* and *the exponents of* f.
- The case of $n + 1$ terms is a cute exercise for your students: *An* $O(n)$*-time algorithm (assuming affinely independent exponents) is possible!*
- But what about systems?

Counting Roots for $(n+1)$-nomial $n \times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}] \in \{-H, \ldots, H\}^{n\times(n+1)}$,

Counting Roots for $(n+1)$-nomial $n \times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}] \in \{-H, \ldots, H\}^{n\times(n+1)}$*, and affinely independent* $a_1, \ldots, a_{n+1} \in \{-d, \ldots, d\}^n$*,*

Counting Roots for $(n+1)$-nomial $n \times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}] \in \{-H, \ldots, H\}^{n\times(n+1)}$, *and affinely independent* $a_1, \ldots, a_{n+1} \in \{-d, \ldots, d\}^n$, **count the roots** *of*

Counting Roots for $(n+1)$-nomial $n \times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}] \in \{-H, \ldots, H\}^{n\times(n+1)}$, *and affinely independent* $a_1, \ldots, a_{n+1} \in \{-d, \ldots, d\}^n$, **count the roots** *of*

$$\boxed{\begin{array}{c} c_{1,1}x^{a_1} + \cdots + c_{1,n+1}x^{a_{n+1}} = 0 \\ \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+1}x^{a_{n+1}} = 0 \end{array}}$$

Counting Roots for $(n+1)$-nomial $n \times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}] \in \{-H, \ldots, H\}^{n\times(n+1)}$, *and affinely independent* $a_1, \ldots, a_{n+1} \in \{-d, \ldots, d\}^n$, **count the roots** *of*

$$\boxed{\begin{array}{c} c_{1,1}x^{a_1} + \cdots + c_{1,n+1}x^{a_{n+1}} = 0 \\ \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+1}x^{a_{n+1}} = 0 \end{array}}$$

in $\mathbb{R}^n_+$,

Counting Roots for $(n+1)$-nomial $n \times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}] \in \{-H, \ldots, H\}^{n\times(n+1)}$, *and affinely independent* $a_1, \ldots, a_{n+1} \in \{-d, \ldots, d\}^n$, **count the roots** *of*

$$\boxed{\begin{array}{c} c_{1,1}x^{a_1} + \cdots + c_{1,n+1}x^{a_{n+1}} = 0 \\ \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+1}x^{a_{n+1}} = 0 \end{array}}$$

in $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$,

Counting Roots for $(n+1)$-nomial $n \times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}] \in \{-H, \ldots, H\}^{n\times(n+1)}$, *and affinely independent* $a_1, \ldots, a_{n+1} \in \{-d, \ldots, d\}^n$, **count the roots** *of*

$$\boxed{\begin{array}{c} c_{1,1}x^{a_1} + \cdots + c_{1,n+1}x^{a_{n+1}} = 0 \\ \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+1}x^{a_{n+1}} = 0 \end{array}}$$

in $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *in time polynomial in* $n + \log(dH)$...

Counting Roots for $(n+1)$-nomial $n\times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}]\in\{-H,\ldots,H\}^{n\times(n+1)}$, *and affinely independent* $a_1,\ldots,a_{n+1}\in\{-d,\ldots,d\}^n$, **count the roots** *of*

$$\boxed{\begin{matrix} c_{1,1}x^{a_1}+\cdots+c_{1,n+1}x^{a_{n+1}}=0 \\ \vdots \\ c_{n,1}x^{a_1}+\cdots+c_{n,n+1}x^{a_{n+1}}=0 \end{matrix}}$$

in $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *in time polynomial in* $n+\log(dH)$...

- *Smith Factorization* helps,

Counting Roots for $(n+1)$-nomial $n \times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}] \in \{-H, \ldots, H\}^{n\times(n+1)}$, *and affinely independent* $a_1, \ldots, a_{n+1} \in \{-d, \ldots, d\}^n$, **count the roots** *of*

$$\boxed{\begin{array}{c} c_{1,1}x^{a_1} + \cdots + c_{1,n+1}x^{a_{n+1}} = 0 \\ \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+1}x^{a_{n+1}} = 0 \end{array}}$$

in $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *in time polynomial in* $n + \log(dH)$*...*

- *Smith Factorization* helps, and you'll ultimately need *(polynomial-time!) Linear Programming* for the case ∞!

Counting Roots for $(n+1)$-nomial $n \times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}] \in \{-H, \ldots, H\}^{n\times(n+1)}$, *and affinely independent* $a_1, \ldots, a_{n+1} \in \{-d, \ldots, d\}^n$, **count the roots** *of*

$$\boxed{\begin{array}{c} c_{1,1}x^{a_1} + \cdots + c_{1,n+1}x^{a_{n+1}} = 0 \\ \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+1}x^{a_{n+1}} = 0 \end{array}}$$

in $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *in time polynomial in* $n + \log(dH)$...

- *Smith Factorization* helps, and you'll ultimately need *(polynomial-time!) Linear Programming* for the case ∞!
- $(n+2)$-nomial $n \times n$ systems are the next step up,

Counting Roots for $(n+1)$-nomial $n \times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}] \in \{-H, \ldots, H\}^{n\times(n+1)}$, *and affinely independent* $a_1, \ldots, a_{n+1} \in \{-d, \ldots, d\}^n$, **count the roots** *of*

$$\boxed{\begin{array}{c} c_{1,1}x^{a_1} + \cdots + c_{1,n+1}x^{a_{n+1}} = 0 \\ \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+1}x^{a_{n+1}} = 0 \end{array}}$$

in $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *in time polynomial in* $n + \log(dH)$...

- *Smith Factorization* helps, and you'll ultimately need *(polynomial-time!) Linear Programming* for the case ∞!

- $(n+2)$-nomial $n \times n$ systems are the next step up, and the last tractable case

Counting Roots for $(n+1)$-nomial $n \times n$ Systems is Easy, but...

Bonus Exercise. *Given* $[c_{i,j}] \in \{-H, \ldots, H\}^{n\times(n+1)}$, *and affinely independent* $a_1, \ldots, a_{n+1} \in \{-d, \ldots, d\}^n$, **count the roots** *of*

$$\boxed{\begin{array}{c} c_{1,1}x^{a_1} + \cdots + c_{1,n+1}x^{a_{n+1}} = 0 \\ \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+1}x^{a_{n+1}} = 0 \end{array}}$$

in $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *in time polynomial in* $n + \log(dH)$*...*

- *Smith Factorization* helps, and you'll ultimately need *(polynomial-time!) Linear Programming* for the case ∞!

- $(n+2)$-nomial $n \times n$ systems are the next step up, and the last tractable case (for now)...

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n,

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n, *one can exactly count the roots in* $\mathbb{R}_+^n$,

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n, *one can exactly count the roots in* $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$,

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n, *one can exactly count the roots in* $\mathbb{R}_+^n$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n, *one can exactly count the roots in* $\mathbb{R}_+^n$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *of any* generic $(n+2)$*-nomial* $n \times n$ *polynomial system*

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n, *one can exactly count the roots in* $\mathbb{R}_+^n$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *of any* generic $(n+2)$*-nomial* $n \times n$ *polynomial system over* $\mathbb{Q}$

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n, *one can exactly count the roots in* $\mathbb{R}_+^n$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *of any* generic $(n+2)$*-nomial* $n \times n$ *polynomial system over* $\mathbb{Q}$ *in deterministic polynomial-time.*

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n, *one can exactly count the roots in* $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *of any* generic $(n+2)$*-nomial* $n \times n$ *polynomial system over* $\mathbb{Q}$ *in deterministic polynomial-time.*

Example: *In* <1 *second, we can see that the* 7*-nomial* 5×5 *system defined by*

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n, *one can exactly count the roots in* $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *of any* generic $(n+2)$*-nomial* $n \times n$ *polynomial system over* $\mathbb{Q}$ *in deterministic polynomial-time.*

Example: *In* <1 *second, we can see that the* 7*-nomial* 5×5 *system defined by*

$$\Big(2x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 37137cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{9}{2}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + 2x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 24849cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + 2x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 21009cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + 2x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20769cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + 2x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20754cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343}\Big).$$

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n, *one can exactly count the roots in* $\mathbb{R}_+^n$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *of any* generic $(n+2)$*-nomial* $n \times n$ *polynomial system over* $\mathbb{Q}$ *in deterministic polynomial-time.*

Example: *In* <1 *second, we can see that the* 7*-nomial* 5×5 *system defined by*

$$\Big(2x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 37137cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{9}{2}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + 2x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 24849cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + 2x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 21009cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + 2x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20769cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + 2x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20754cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343}\Big).$$

has exactly 2, 6, 6, 2, 2, *or* 0 *positive roots, respectively when* c *is* $\frac{1}{20731}, \frac{1}{20730}, \frac{1}{14392}, \frac{1}{14391}, \frac{1}{13059}, \frac{1}{13058}$.

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n, *one can exactly count the roots in* $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *of any* generic $(n+2)$*-nomial* $n \times n$ *polynomial system over* $\mathbb{Q}$ *in deterministic polynomial-time.*

Example: *In* <1 *second, we can see that the* 7*-nomial* 5×5 *system defined by*

$$\begin{aligned}
\Big(&2x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 37137cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{9}{2}x_3^{166}x_4^{68}x_5^{343},\\
&x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + 2x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 24849cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},\\
&x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + 2x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 21009cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},\\
&x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + 2x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20769cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},\\
&x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + 2x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20754cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343}\Big).
\end{aligned}$$

has exactly 2, 6, 6, 2, 2, *or* 0 *positive roots, respectively when* c *is* $\frac{1}{20731}, \frac{1}{20730}, \frac{1}{14392}, \frac{1}{14391}, \frac{1}{13059}, \frac{1}{13058}$. *(Bertini quickly dies.)* ◇

Main Theorem (coarsely)

[Rojas, 2020] *For any* fixed n, *one can exactly count the roots in* $\mathbb{R}_+^n$, $(\mathbb{R}^*)^n$, *and* $\mathbb{R}^n$ *of any* generic $(n+2)$*-nomial* $n \times n$ *polynomial system over* $\mathbb{Q}$ *in deterministic polynomial-time.*

Example: *In* <1 *second, we can see that the* 7*-nomial* 5×5 *system defined by*

$$\Big(2x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 3712\cdots x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{9}{2}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + 2x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + x_1^{20}\cdots cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + 2x_1^{74}x_2^{179}x_3^{25}x_5^{57} + x_1^{25}\cdots x_4^{12}x_5^{68} - 21009cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1^{74}\cdots x_4 + x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20769cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343},$$
$$x_1^{36}x_2^{194}x_3^{50}x_4^{82}x_5^{60} + x_1^{76}x_2^{240}x_4^{41}x_5 + x_1\cdots x_5^{57} + x_1^{25}x_2^{203}x_3^{44}x_4 + 2x_1^{20}x_2^{167}x_3^{64}x_4^{12}x_5^{68} - 20754cx_1^{58}x_2^{194}x_3^{24}x_4^{36}x_5^{25} - \frac{21}{4}x_3^{166}x_4^{68}x_5^{343}\Big).$$

Over 245 million ***complex*** **roots!**

has exactly 2, 6, 6, 2, 2, *or* 0 *positive roots, respectively when* c *is* $\frac{1}{20731}, \frac{1}{20730}, \frac{1}{14392}, \frac{1}{14391}, \frac{1}{13059}, \frac{1}{13058}$. *(Bertini quickly dies.)*

Fewnomials and the Quest for Tight Bounds: 2000s

[Bihan, Sottile, 2007] Let $c_{i,j} \in \mathbb{R}$ for all i, j,

Fewnomials and the Quest for Tight Bounds: 2000s

[Bihan, Sottile, 2007] Let $c_{i,j} \in \mathbb{R}$ for all i, j, let $a_1, \ldots, a_{n+k} \in \mathbb{R}^n$ be any points *not* all lying in an affine hyperplane,

Fewnomials and the Quest for Tight Bounds: 2000s

[Bihan, Sottile, 2007] Let $c_{i,j} \in \mathbb{R}$ for all i, j, let $a_1, \ldots, a_{n+k} \in \mathbb{R}^n$ be any points *not* all lying in an affine hyperplane, and write $x^{a_i} = x_1^{a_{1,i}} \cdots x_n^{a_{n,i}}$.

Fewnomials and the Quest for Tight Bounds: 2000s


[Bihan, Sottile, 2007] Let $c_{i,j} \in \mathbb{R}$ for all i, j, let $a_1, \ldots, a_{n+k} \in \mathbb{R}^n$ be any points *not* all lying in an affine hyperplane, and write $x^{a_i} = x_1^{a_{1,i}} \cdots x_n^{a_{n,i}}$.

Real Fewnomial Theorem$^+$. *Any real $(n+k)$-nomial $n \times n$ polynomial system of the form...*

$$F := (f_1, \ldots, f_n) = \begin{cases} c_{1,1}x^{a_1} + \cdots + c_{1,n+k}x^{a_{n+k}} \\ \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+k}x^{a_{n+k}} \end{cases}$$

Fewnomials and the Quest for Tight Bounds: 2000s


[Bihan, Sottile, 2007] Let $c_{i,j} \in \mathbb{R}$ for all i, j, let $a_1, \ldots, a_{n+k} \in \mathbb{R}^n$ be any points *not* all lying in an affine hyperplane, and write $x^{a_i} = x_1^{a_{1,i}} \cdots x_n^{a_{n,i}}$.

Real Fewnomial Theorem$^+$. *Any real $(n+k)$-nomial $n \times n$ polynomial system of the form...*

$$F := (f_1, \ldots, f_n) = \begin{cases} c_{1,1}x^{a_1} + \cdots + c_{1,n+k}x^{a_{n+k}} \\ \qquad\qquad \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+k}x^{a_{n+k}} \end{cases}$$

has no more than $\frac{e^2+3}{4} 2^{(k-1)(k-2)/2} n^{k-1}$ *non-degenerate roots in* $\mathbb{R}^n_+$.

Fewnomials and the Quest for Tight Bounds: 2000s


[Bihan, Sottile, 2007] Let $c_{i,j} \in \mathbb{R}$ for all i, j, let $a_1, \ldots, a_{n+k} \in \mathbb{R}^n$ be any points *not* all lying in an affine hyperplane, and write $x^{a_i} = x_1^{a_{1,i}} \cdots x_n^{a_{n,i}}$.

Real Fewnomial Theorem$^+$. *Any real $(n+k)$-nomial $n \times n$ polynomial system of the form...*

$$F := (f_1, \ldots, f_n) = \begin{cases} c_{1,1}x^{a_1} + \cdots + c_{1,n+k}x^{a_{n+k}} \\ \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+k}x^{a_{n+k}} \end{cases}$$

has no more than $\frac{e^2+3}{4} 2^{(k-1)(k-2)/2} n^{k-1}$ *non-degenerate roots in* $\mathbb{R}^n_+$.

[Bihan, Rojas, Sottile '07]: $\exists$ *systems with* $\left\lfloor \frac{n+k-1}{\min\{n,k-1\}} \right\rfloor^{\min\{n,k-1\}}$ *positive roots.*

Fewnomials and the Quest for Tight Bounds: 2000s

[Bihan, Sottile, 2007] Let $c_{i,j} \in \mathbb{R}$ for all i, j, let $a_1, \ldots, a_{n+k} \in \mathbb{R}^n$ be any points *not* all lying in an affine hyperplane, and write $x^{a_i} = x_1^{a_{1,i}} \cdots x_n^{a_{n,i}}$.

Real Fewnomial Theorem$^+$. *Any real $(n+k)$-nomial $n \times n$ polynomial system of the form...*

$$F := (f_1, \ldots, f_n) = \begin{cases} c_{1,1}x^{a_1} + \cdots + c_{1,n+k}x^{a_{n+k}} \\ \vdots \\ c_{n,1}x^{a_1} + \cdots + c_{n,n+k}x^{a_{n+k}} \end{cases}$$

has no more than $\frac{e^2+3}{4} 2^{(k-1)(k-2)/2} n^{k-1}$ *non-degenerate roots in* $\mathbb{R}^n_+$.

[Bertrand, Bihan, Sottile '06]: Tight *bound for* $k = 2$ *of* $\boxed{n+1}$.

Fewnomials and the Quest for Tight Bounds: 2010s



Theorem. [Bürgisser, Ergür, Tonelli-Cueto, 2019] *For an* $(n+k)$*-nomial* $n \times n$ *system with independent standard real Gaussian coefficients*

Fewnomials and the Quest for Tight Bounds: 2010s



Theorem. [Bürgisser, Ergür, Tonelli-Cueto, 2019] *For an* $(n+k)$*-nomial* $n \times n$ *system with independent standard real Gaussian coefficients and fixed support not lying in an affine hyperplane,*

Fewnomials and the Quest for Tight Bounds: 2010s



Theorem. [Bürgisser, Ergür, Tonelli-Cueto, 2019] *For an $(n+k)$-nomial $n \times n$ system with independent standard real Gaussian coefficients and fixed support not lying in an affine hyperplane, the average number of roots in $(\mathbb{R}^*)^n$ is $\leq \frac{1}{2}\binom{n+k}{k}$.*

Bisection $\longrightarrow$ Diophantine Approximation

Lemma. (e.g., [Ye, 1995]) $O(\log D)$ *steps of* bisection *are enough to get you a succinct approximant to* $\sqrt[D]{c}$

Bisection $\longrightarrow$ Diophantine Approximation

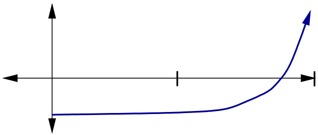
Lemma. (e.g., [Ye, 1995]) $O(\log D)$ *steps of* bisection *are enough to get you a succinct approximant to* $\sqrt[D]{c}$ *for any* $D \in \mathbb{N}$ *and* $c \in \mathbb{Q}_+$.

Bisection $\longrightarrow$ Diophantine Approximation

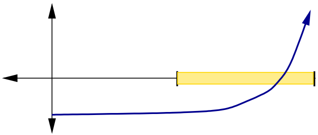
Lemma. (e.g., [Ye, 1995]) $O(\log D)$ *steps of* bisection *are enough to get you a succinct approximant to* $\sqrt[D]{c}$ *for any* $D \in \mathbb{N}$ *and* $c \in \mathbb{Q}_+$. *Counting bit operations, we get time* $O\left(\text{size}(x_1^D - c)^{2+\varepsilon}\right)$.

Bisection $\longrightarrow$ Diophantine Approximation

Lemma. (e.g., [Ye, 1995]) $O(\log D)$ *steps of* bisection *are enough to get you a succinct approximant to* $\sqrt[D]{c}$ *for any* $D \in \mathbb{N}$ *and* $c \in \mathbb{Q}_+$*. Counting bit operations, we get time* $O\big(\text{size}(x_1^D - c)^{2+\varepsilon}\big)$.

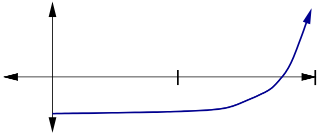


$$u^D \overset{?}{\substack{> \\ = \\ <}} cv^D$$

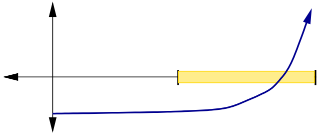


Bisection $\longrightarrow$ Diophantine Approximation

Lemma. (e.g., [Ye, 1995]) $O(\log D)$ *steps of* bisection *are enough to get you a succinct approximant to* $\sqrt[D]{c}$ *for any* $D \in \mathbb{N}$ *and* $c \in \mathbb{Q}_+$. *Counting bit operations, we get time* $O\big(\mathrm{size}(x_1^D - c)^{2+\varepsilon}\big)$.

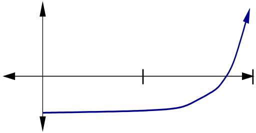


Trouble?

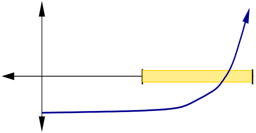


Bisection $\longrightarrow$ Diophantine Approximation

Lemma. (e.g., [Ye, 1995]) $O(\log D)$ *steps of* bisection *are enough to get you a succinct approximant to* $\sqrt[D]{c}$ *for any* $D \in \mathbb{N}$ *and* $c \in \mathbb{Q}_+$*. Counting bit operations, we get time* $O\left(\mathrm{size}(x_1^D - c)^{2+\varepsilon}\right)$.

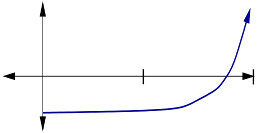


$$\boxed{D\log|u| \;\overset{?}{\substack{>\\=\\<}}\; \log|c| + D\log|v|}$$

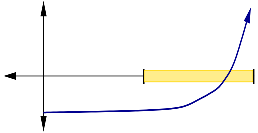


Bisection $\longrightarrow$ Diophantine Approximation

Lemma. (e.g., [Ye, 1995]) $O(\log D)$ *steps of* bisection *are enough to get you a succinct approximant to* $\sqrt[D]{c}$ *for any* $D \in \mathbb{N}$ *and* $c \in \mathbb{Q}_+$. *Counting bit operations, we get time* $O\left(\text{size}(x_1^D - c)^{2+\varepsilon}\right)$.



$$\boxed{D \log |u| \overset{?}{\underset{<}{\geq}} \log |c| + D \log |v|}$$



[Baker, 1966] If $a_i, b_i \in \mathbb{Z}$ with $A := \max_i \log |a_i|$, $B := \max_i \log |b_i|$, and $\Lambda := \sum_{i=1}^{n} b_i \log a_i$, then $\Lambda \neq 0 \Longrightarrow |\log |\Lambda|| = O(A)^n \log B$.

Real Univariate Trinomials?

• Deciding the sign of a trinomial $f \in \mathbb{Q}[x_1]$ at $z \in \mathbb{Q}$ in time $(\text{size}(f) + \text{height}(z))^{O(1)}$ is an open problem!

Real Univariate Trinomials?

- Deciding the sign of a trinomial $f \in \mathbb{Q}[x_1]$ at $z \in \mathbb{Q}$ in time $(\text{size}(f) + \text{height}(z))^{O(1)}$ is an open problem!

- So bisection is obstructed!

Real Univariate Trinomials?

- Deciding the sign of a trinomial $f \in \mathbb{Q}[x_1]$ at $z \in \mathbb{Q}$ in time $(\text{size}(f) + \text{height}(z))^{O(1)}$ is an open problem!

- So bisection is obstructed!

- Despite nice progress by [Jindal, Sagraloff, 2017] on *coarse* approximants,

Real Univariate Trinomials?

- Deciding the sign of a trinomial $f \in \mathbb{Q}[x_1]$ at $z \in \mathbb{Q}$ in time $(\mathrm{size}(f) + \mathrm{height}(z))^{O(1)}$ is an open problem!

- So bisection is obstructed!

- Despite nice progress by [Jindal, Sagraloff, 2017] on *coarse* approximants, real root counting in time $(t \log(dH))^{O(1)}$ is still an open problem!

Exact Counting for $(n+2)$-nomial $n \times n$ Systems

Main Theorem. [Rojas, 2020] *For any $(n+2)$-nomial $n \times n$ system F over $\mathbb{Q}$,*

Exact Counting for $(n+2)$-nomial $n \times n$ Systems

Main Theorem. [Rojas, 2020] *For any $(n+2)$-nomial $n \times n$ system F over $\mathbb{Q}$, with exponents* not *lying in a common affine hyperplane,*

Exact Counting for $(n+2)$-nomial $n \times n$ Systems

Main Theorem. [Rojas, 2020] *For any* $(n+2)$*-nomial* $n \times n$ *system* F *over* $\mathbb{Q}$*, with exponents* not *lying in a common affine hyperplane, and coefficient matrix* $[c_{i,j}]$ *of rank** n*,*

Exact Counting for $(n+2)$-nomial $n \times n$ Systems

Main Theorem. [Rojas, 2020] *For any* $(n+2)$*-nomial* $n \times n$ *system* F *over* $\mathbb{Q}$*, with exponents* not *lying in a common affine hyperplane, and coefficient matrix* $[c_{i,j}]$ *of rank** n*, we can count exactly its number of roots in* $\mathbb{R}^n_+$*,*

Exact Counting for $(n+2)$-nomial $n \times n$ Systems

Main Theorem. [Rojas, 2020] *For any $(n+2)$-nomial $n \times n$ system F over $\mathbb{Q}$, with exponents* not *lying in a common affine hyperplane, and coefficient matrix $[c_{i,j}]$ of rank* n, we can count exactly its number of roots in $\mathbb{R}_+^n$, $(\mathbb{R}^*)^n$,*

Exact Counting for $(n+2)$-nomial $n \times n$ Systems

Main Theorem. [Rojas, 2020] *For any $(n+2)$-nomial $n \times n$ system F over $\mathbb{Q}$, with exponents* not *lying in a common affine hyperplane, and coefficient matrix $[c_{i,j}]$ of rank* n, we can count exactly its number of roots in $\mathbb{R}_+^n$, $(\mathbb{R}^*)^n$, and $\mathbb{R}^n$,*

Exact Counting for $(n+2)$-nomial $n \times n$ Systems

Main Theorem. [Rojas, 2020] *For any $(n+2)$-nomial $n \times n$ system F over $\mathbb{Q}$, with exponents* not *lying in a common affine hyperplane, and coefficient matrix $[c_{i,j}]$ of rank* n, we can count exactly its number of roots in $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$, and $\mathbb{R}^n$, in deterministic time*

$$O(n\log(nd) + n^2\log(nH))^{2n+4}.$$

Exact Counting for $(n+2)$-nomial $n \times n$ Systems

Main Theorem. [Rojas, 2020] *For any $(n+2)$-nomial $n \times n$ system F over $\mathbb{Q}$, with exponents* not *lying in a common affine hyperplane, and coefficient matrix $[c_{i,j}]$ of rank* n, we can count exactly its number of roots in $\mathbb{R}_+^n$, $(\mathbb{R}^*)^n$, and $\mathbb{R}^n$, in deterministic time*

$$O(n\log(nd) + n^2\log(nH)))^{2n+4}.$$

•Key new ingredients are a refined version of *Liouville's Theorem*,

Exact Counting for $(n+2)$-nomial $n \times n$ Systems

Main Theorem. [Rojas, 2020] *For any $(n+2)$-nomial $n \times n$ system F over $\mathbb{Q}$, with exponents* not *lying in a common affine hyperplane, and coefficient matrix $[c_{i,j}]$ of rank* n, we can count exactly its number of roots in $\mathbb{R}_+^n$, $(\mathbb{R}^*)^n$, and $\mathbb{R}^n$, in deterministic time*

$$O(n\log(nd)+n^2\log(nH))^{2n+4}.$$

•Key new ingredients are a refined version of *Liouville's Theorem*, and a theorem of [Baker, Wustholtz, 1993]

Exact Counting for $(n+2)$-nomial $n \times n$ Systems

Main Theorem. [Rojas, 2020] *For any $(n+2)$-nomial $n \times n$ system F over $\mathbb{Q}$, with exponents* not *lying in a common affine hyperplane, and coefficient matrix $[c_{i,j}]$ of rank* n, we can count exactly its number of roots in $\mathbb{R}_+^n$, $(\mathbb{R}^*)^n$, and $\mathbb{R}^n$, in deterministic time*

$$O(n\log(nd) + n^2\log(nH))^{2n+4}.$$

•Key new ingredients are a refined version of *Liouville's Theorem*, and a theorem of [Baker, Wustholtz, 1993] on linear combinations of logs in *algebraic* numbers...

Exact Counting for $(n+2)$-nomial $n \times n$ Systems

Main Theorem. [Rojas, 2020] *For any $(n+2)$-nomial $n \times n$ system F over $\mathbb{Q}$, with exponents* not *lying in a common affine hyperplane, and coefficient matrix $[c_{i,j}]$ of rank* n, we can count exactly its number of roots in $\mathbb{R}^n_+$, $(\mathbb{R}^*)^n$, and $\mathbb{R}^n$, in deterministic time*

$$O(n\log(nd) + n^2\log(nH))^{2n+4}.$$

•Key new ingredients are a refined version of *Liouville's Theorem*, and a theorem of [Baker, Wustholtz, 1993] on linear combinations of logs in *algebraic* numbers...

•Sufficiently refined versions of the *abc-Conjecture* can reduce the complexity to polynomial in n...

Key Idea #1: Gaussian Elimination

By Gauss-Jordan Elimination applied to the linear combinations of monomials, our original 5×5 example can be reduced to

$$\begin{aligned}
x_1^{36}x_2^{194}x_3^{-116}x_4^{14}x_5^{-283} &= 16384cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318} + \frac{1}{4} \\
x_1^{76}x_2^{240}x_3^{-166}x_4^{-27}x_5^{-342} &= 4096cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318} + 1 \\
x_1^{74}x_2^{179}x_3^{-141}x_4^{-68}x_5^{-286} &= 256cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318} + 1 \\
x_1^{25}x_2^{203}x_3^{-122}x_4^{-67}x_5^{-343} &= 16cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318} + 1 \\
x_1^{20}x_2^{167}x_3^{-102}x_4^{-56}x_5^{-275} &= cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318} + 1
\end{aligned}$$

Key Idea #1: Gaussian Elimination

By Gauss-Jordan Elimination applied to the linear combinations of monomials, our original 5×5 example can be reduced to

$$\begin{aligned}
x_1^{36}x_2^{194}x_3^{-116}x_4^{14}x_5^{-283} &= 16384cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318}+\frac{1}{4}\\
x_1^{76}x_2^{240}x_3^{-166}x_4^{-27}x_5^{-342} &= 4096cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318}+1\\
x_1^{74}x_2^{179}x_3^{-141}x_4^{-68}x_5^{-286} &= 256cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318}+1\\
x_1^{25}x_2^{203}x_3^{-122}x_4^{-67}x_5^{-343} &= 16cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318}+1\\
x_1^{20}x_2^{167}x_3^{-102}x_4^{-56}x_5^{-275} &= cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318}+1
\end{aligned}$$

Next: Observe that the set of exponent vectors is a *circuit in the sense of combinatorics*:

Key Idea #1: Gaussian Elimination

By Gauss-Jordan Elimination applied to the linear combinations of monomials, our original 5×5 example can be reduced to

$$\begin{aligned}
x_1^{36}x_2^{194}x_3^{-116}x_4^{14}x_5^{-283} &= 16384cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318} + \frac{1}{4} \\
x_1^{76}x_2^{240}x_3^{-166}x_4^{-27}x_5^{-342} &= 4096cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318} + 1 \\
x_1^{74}x_2^{179}x_3^{-141}x_4^{-68}x_5^{-286} &= 256cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318} + 1 \\
x_1^{25}x_2^{203}x_3^{-122}x_4^{-67}x_5^{-343} &= 16cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318} + 1 \\
x_1^{20}x_2^{167}x_3^{-102}x_4^{-56}x_5^{-275} &= cx_1^{58}x_2^{194}x_3^{-142}x_4^{-32}x_5^{-318} + 1
\end{aligned}$$

Next: Observe that the set of exponent vectors is a *circuit in the sense of combinatorics*: It is the union of a point and the vertex set of a simplex...

Key Idea #2: Gale Dual Form

The exponent vectors $a_1, \ldots, a_7$ in our example have a *unique* affine relation:

Key Idea #2: Gale Dual Form

The exponent vectors $a_1, \ldots, a_7$ in our example have a *unique* affine relation:

$$-2a_1 + 2a_2 - 2a_3 + 2a_4 - 2a_5 + a_6 + a_7 = \mathbf{O}.$$

Key Idea #2: Gale Dual Form

The exponent vectors $a_1, \ldots, a_7$ in our example have a *unique* affine relation:

$$-2a_1 + 2a_2 - 2a_3 + 2a_4 - 2a_5 + a_6 + a_7 = \mathbf{O}.$$

So then, $\zeta \in \mathbb{R}^5_+$ a root $\Longrightarrow u := \zeta_1^{58}\zeta_2^{194}\zeta_4^{-32}\zeta_5^{-318}$ must be a root

Key Idea #2: Gale Dual Form

The exponent vectors $a_1, \ldots, a_7$ in our example have a *unique* affine relation:

$$-2a_1 + 2a_2 - 2a_3 + 2a_4 - 2a_5 + a_6 + a_7 = \mathbf{O}.$$

So then, $\zeta \in \mathbb{R}^5_+$ a root $\Longrightarrow u := \zeta_1^{58}\zeta_2^{194}\zeta_4^{-32}\zeta_5^{-318}$ must be a root of the *Gale Dual rational function*
$(16384cu+1)^{-2}(4096cu+1)^2(256cu+1)^{-2}(16cu+1)^2(cu+1)^{-2}u-1.$

Equivalently, u must be a real root of the *linear combination of logarithms*

Key Idea #2: Gale Dual Form

The exponent vectors $a_1, \ldots, a_7$ in our example have a *unique* affine relation:

$$-2a_1 + 2a_2 - 2a_3 + 2a_4 - 2a_5 + a_6 + a_7 = \mathbf{O}.$$

So then, $\zeta \in \mathbb{R}^5_+$ a root $\Longrightarrow u := \zeta_1^{58}\zeta_2^{194}\zeta_4^{-32}\zeta_5^{-318}$ must be a root of the *Gale Dual rational function*

$$(16384cu+1)^{-2}(4096cu+1)^2(256cu+1)^{-2}(16cu+1)^2(cu+1)^{-2}u - 1.$$

Equivalently, u must be a real root of the *linear combination of logarithms*

$$-2\log|16384cu+1| + 2\log|4096cu+1| - 2\log|256cu+1| + 2\log|16cu+1| - 2\log|cu+1| + \log|u|,$$

Key Idea #2: Gale Dual Form

The exponent vectors $a_1, \ldots, a_7$ in our example have a *unique* affine relation:

$$-2a_1 + 2a_2 - 2a_3 + 2a_4 - 2a_5 + a_6 + a_7 = \mathbf{O}.$$

So then, $\zeta \in \mathbb{R}^5_+$ a root $\Longrightarrow u := \zeta_1^{58}\zeta_2^{194}\zeta_4^{-32}\zeta_5^{-318}$ must be a root of the *Gale Dual rational function*

$$(16384cu+1)^{-2}(4096cu+1)^2(256cu+1)^{-2}(16cu+1)^2(cu+1)^{-2}u - 1.$$

Equivalently, u must be a real root of the *linear combination of logarithms*

$$-2\log|16384cu+1| + 2\log|4096cu+1| - 2\log|256cu+1| + 2\log|16cu+1| - 2\log|cu+1| + \log|u|,$$

provided some additional sign conditions are met...

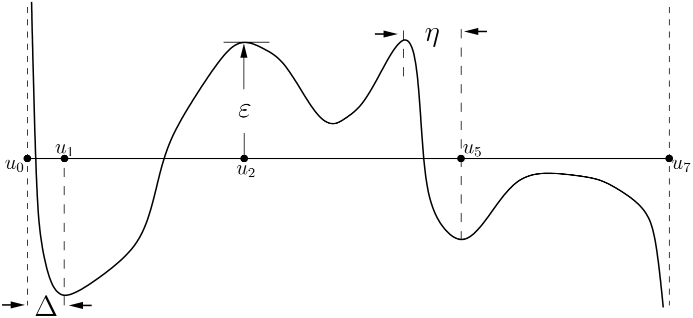
Note: The exponents are usually *much* larger!

Key Idea #3: Critical Points and Diophantine Approximation

Examine the graph of the linear combination of logarithms...

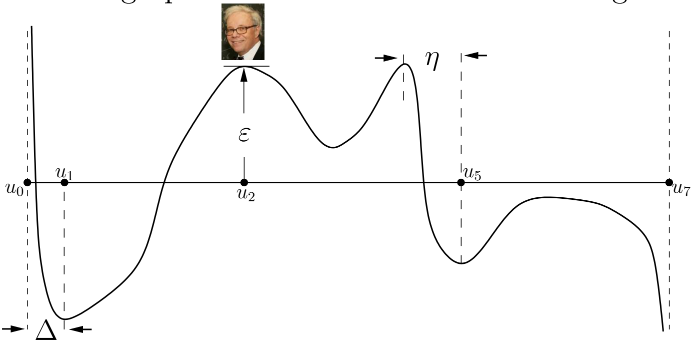
Key Idea #3: Critical Points and Diophantine Approximation

Examine the graph of the linear combination of logarithms...



Key Idea #3: Critical Points and Diophantine Approximation

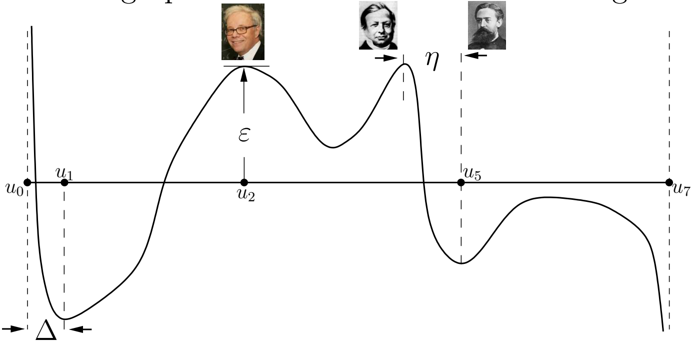
Examine the graph of the linear combination of logarithms...



Baker-*Wustholtz* gives height of peaks...

Key Idea #3: Critical Points and Diophantine Approximation

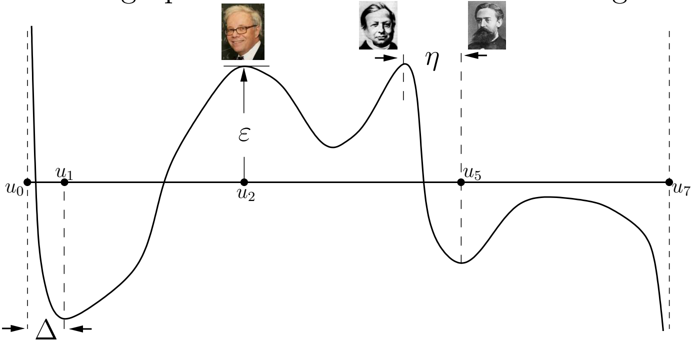
Examine the graph of the linear combination of logarithms...



Baker-*Wustholtz* gives height of peaks... Bounds of Liouville and Markov control root spacing...

Key Idea #3: Critical Points and Diophantine Approximation

Examine the graph of the linear combination of logarithms...

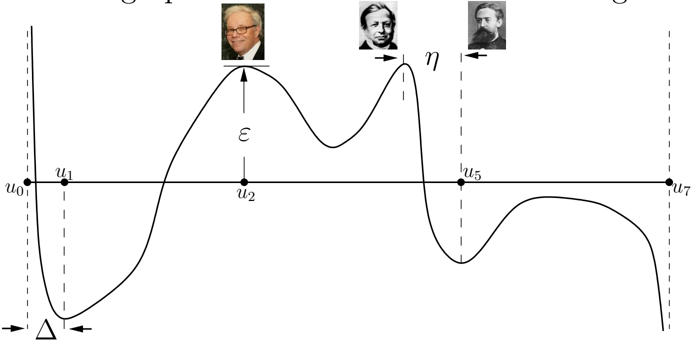


Baker-*Wustholtz* gives height of peaks... Bounds of Liouville and Markov control root spacing... Then use

Rolle's Theorem,

Key Idea #3: Critical Points and Diophantine Approximation

Examine the graph of the linear combination of logarithms...

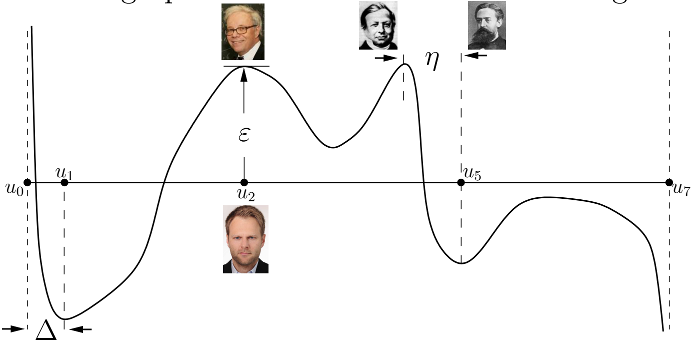


Baker-*Wustholtz* gives height of peaks... Bounds of Liouville and Markov control root spacing... Then use

Rolle's Theorem, AGM Iteration for accurate logs,

Key Idea #3: Critical Points and Diophantine Approximation

Examine the graph of the linear combination of logarithms...



Baker-*Wustholtz* gives height of peaks... Bounds of Liouville and Markov control root spacing... Then use Rolle's Theorem, AGM Iteration for accurate logs, and Sturm-Habicht sequences to isolate critical points!...

Upshot: Randomize!

Even though there *are* $(n+1)$-nomial $n \times n$ systems,

Upshot: Randomize!

Even though there *are* $(n+1)$-nomial $n \times n$ systems, and *univariate tetranomials* with exponentially close real roots,

Upshot: Randomize!

Even though there *are* $(n+1)$-nomial $n \times n$ systems, and *univariate tetranomials* with exponentially close real roots, these appear to be rare in practice [Mignotte, 1995; Rojas, Zhu, 2021].

Upshot: Randomize!

Even though there *are* $(n+1)$-nomial $n \times n$ systems, and *univariate tetranomials* with exponentially close real roots, these appear to be rare in practice [Mignotte, 1995; Rojas, Zhu, 2021]. 

While we can now counting real roots in time $(\log(dH))^{O(n)}$,

Upshot: Randomize!

Even though there *are* $(n+1)$-nomial $n \times n$ systems, and *univariate tetranomials* with exponentially close real roots, these appear to be rare in practice [Mignotte, 1995; Rojas, Zhu, 2021]. 

While we can now counting real roots in time $(\log(dH))^{O(n)}$, there is growing evidence that we can attain complexity $(n \log(DH))^{O(1)}$

Upshot: Randomize!

Even though there *are* $(n+1)$-nomial $n \times n$ systems, and *univariate tetranomials* with exponentially close real roots, these appear to be rare in practice [Mignotte, 1995; Rojas, Zhu, 2021]. 

While we can now counting real roots in time $(\log(dH))^{O(n)}$, there is growing evidence that we can attain complexity $(n \log(DH))^{O(1)}$ *on average* [Deng, Ergür, Paouris, Rojas, 2021]   ...

♡ Thank you for your attention!

See `www.math.tamu.edu/~rojas` for preprints and further info...