Root finding over finite fields using Graeffe transforms

Bruno Grenet
LIRMM
U. Montpellier

Joris van der Hoeven & Grégoire Lecerf
CNRS – LIX
École polytechnique

RAIM — Banyuls — June 29., 2016

Root finding over finite fields

Given $f \in \mathbb{F}_q[X]$, compute its roots, that is $\{\alpha \in \mathbb{F}_q : f(\alpha) = 0\}$.

Input size: $(1 + d) \log q$ where $d = \deg(f)$

Root finding over finite fields

Given $f \in \mathbb{F}_q[X]$, compute its roots, that is $\{\alpha \in \mathbb{F}_q : f(\alpha) = 0\}$.

Input size: $(1 + d) \log q$ where $d = \deg(f)$

- **Assumption (A):** f is **monic**, **separable**, **splits** over $\mathbb{F}_q$, $f(0) \neq 0$:

$$f(X) = \prod_{i=1}^{d}(X - \alpha_i), \quad \alpha_i \in \mathbb{F}_q^*, \quad \alpha_i \neq \alpha_j$$

(easy reduction: $f \leftarrow \gcd(f, X^{q-1} - 1)$)

Root finding over finite fields

Given $f \in \mathbb{F}_q[X]$, compute its roots, that is $\{\alpha \in \mathbb{F}_q : f(\alpha) = 0\}$.

Input size: $(1+d)\log q$ where $d = \deg(f)$

- **Assumption (A):** f is **monic**, **separable**, **splits** over $\mathbb{F}_q$, $f(0) \neq 0$:

$$f(X) = \prod_{i=1}^{d}(X - \alpha_i), \quad \alpha_i \in \mathbb{F}_q^*, \quad \alpha_i \neq \alpha_j$$

(easy reduction: $f \leftarrow \gcd(f, X^{q-1} - 1)$)

- Motivated by sparse interpolation

[van der Hoeven & Lecerf, 2014]

- No deterministic polytime algorithm is known (even under ERH)

- No deterministic polytime algorithm is known (even under ERH)
- Randomized algorithm: $\tilde{O}(d \log^2 q)$ **in average** [Rabin (1980)]

- No deterministic polytime algorithm is known (even under ERH)
- Randomized algorithm: $\tilde{O}(d \log^2 q)$ **in average** [Rabin (1980)]
- Many factorization alg. $\rightsquigarrow$ no improvement for root finding
  [Cantor-Zassenhaus (1981), Kaltofen-Shoup (1998), Kedlaya-Umans (2011)]

- No deterministic polytime algorithm is known (even under ERH)
- Randomized algorithm: $\tilde{O}(d \log^2 q)$ in average [Rabin (1980)]
- Many factorization alg. $\rightsquigarrow$ no improvement for root finding [Cantor-Zassenhaus (1981), Kaltofen-Shoup (1998), Kedlaya-Umans (2011)]
- Better complexity bounds when $q-1$ is sufficiently *smooth* [Moenck (1977), von zur Gathen (1987), Mignotte-Schnorr (1988), Rónyai (1989), Shoup (1991, 1992), Źrałek (2010)]

- No deterministic polytime algorithm is known (even under ERH)
- Randomized algorithm: $\tilde{O}(d \log^2 q)$ in average [Rabin (1980)]
- Many factorization alg. $\rightsquigarrow$ no improvement for root finding [Cantor-Zassenhaus (1981), Kaltofen-Shoup (1998), Kedlaya-Umans (2011)]
- Better complexity bounds when $q-1$ is sufficiently *smooth* [Moenck (1977), von zur Gathen (1987), Mignotte-Schnorr (1988), Rónyai (1989), Shoup (1991, 1992), Źrałek (2010)]

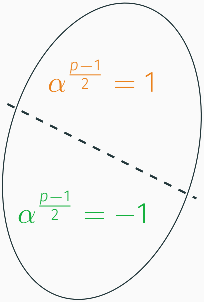
- FFT finite field: $p = M \cdot 2^m + 1$ with $M = O(\log p)$
  - Adapt old algorithms
  - New technique based on Graeffe transforms
  - Fast implementations

Adapt old algorithms

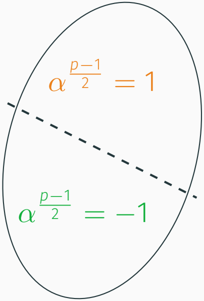
- $\prod_{\alpha \in \mathbb{F}_p^*} (X - \alpha) = X^{p-1} - 1$

- $\prod_{\alpha \in \mathbb{F}_p^*} (X - \alpha) = X^{p-1} - 1 = (X^{\frac{p-1}{2}} - 1)(X^{\frac{p-1}{2}} + 1)$

- $\prod_{\alpha \in \mathbb{F}_p^*} (X - \alpha) = X^{p-1} - 1 = (X^{\frac{p-1}{2}} - 1)(X^{\frac{p-1}{2}} + 1)$

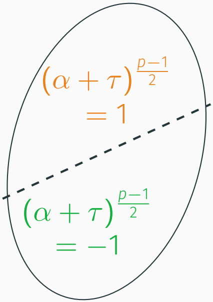


- $\prod_{\alpha \in \mathbb{F}_p^*} (X - \alpha) = X^{p-1} - 1 = (X^{\frac{p-1}{2}} - 1)(X^{\frac{p-1}{2}} + 1)$



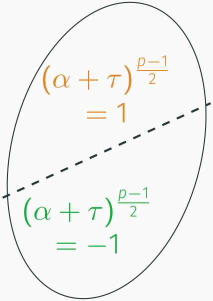
- With some luck, $\gcd(f, X^{\frac{p-1}{2}} - 1) \notin \{1, f\}$.

- $\prod_{\alpha \in \mathbb{F}_p^*} (X - \alpha) = X^{p-1} - 1 = (X^{\frac{p-1}{2}} - 1)(X^{\frac{p-1}{2}} + 1)$



- With some luck, $\gcd(f, X^{\frac{p-1}{2}} - 1) \notin \{1, f\}$.
- Push your luck: $\gcd(f, (X + \tau)^{\frac{p-1}{2}} - 1)$ for some **random** $\tau \in \mathbb{F}_p$

- $\prod_{\alpha \in \mathbb{F}_p^*} (X - \alpha) = X^{p-1} - 1 = (X^{\frac{p-1}{2}} - 1)(X^{\frac{p-1}{2}} + 1)$

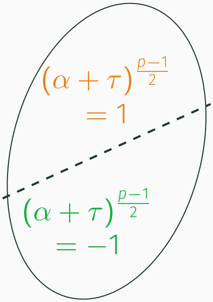


- With some luck, $\gcd(f, X^{\frac{p-1}{2}} - 1) \notin \{1, f\}$.
- Push your luck: $\gcd(f, (X + \tau)^{\frac{p-1}{2}} - 1)$ for some **random** $\tau \in \mathbb{F}_p$

$$\deg\left(\gcd(f, (X + \tau)^{\frac{p-1}{2}} - 1)\right) \simeq d/2$$

Rabin's algorithm

- $\prod_{\alpha \in \mathbb{F}_p^*} (X - \alpha) = X^{p-1} - 1 = (X^{\frac{p-1}{2}} - 1)(X^{\frac{p-1}{2}} + 1)$



- With some luck, $\gcd(f, X^{\frac{p-1}{2}} - 1) \notin \{1, f\}$.
- Push your luck: $\gcd(f, (X + \tau)^{\frac{p-1}{2}} - 1)$ for some **random** $\tau \in \mathbb{F}_p$

$$\deg\left(\gcd(f, (X + \tau)^{\frac{p-1}{2}} - 1)\right) \simeq d/2$$

Randomized algorithm

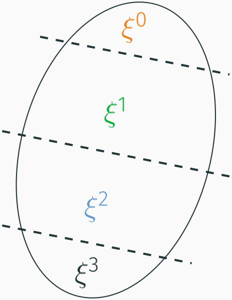
The roots of $f \in \mathbb{F}_p[X]$ can be computed in expected time $\tilde{O}(d \log^2 p)$.

Modified Rabin's algorithm (for FFT finite fields)

$$X^{p-1} - 1 = \prod_{i=0}^{2^\ell - 1} (X^{M2^{m-\ell}} - \xi^i), \text{ where } \xi \text{ is primitive of order } 2^\ell.$$

Modified Rabin's algorithm (for FFT finite fields)

$$X^{p-1} - 1 = \prod_{i=0}^{2^\ell - 1} (X^{M2^{m-\ell}} - \xi^i), \text{ where } \xi \text{ is primitive of order } 2^\ell.$$



$$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^0)$$

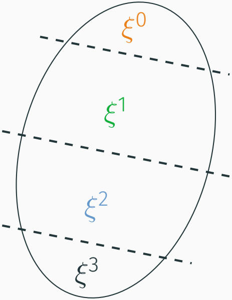
$$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^1)$$

$$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^2)$$

$$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^3)$$

Modified Rabin's algorithm (for FFT finite fields)

$$X^{p-1} - 1 = \prod_{i=0}^{2^\ell - 1} (X^{M2^{m-\ell}} - \xi^i),$$ where ξ is primitive of order 2^ℓ.



$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^0)$

$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^1)$

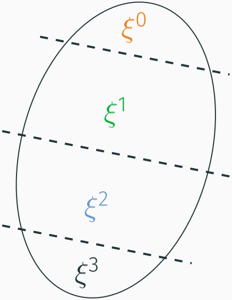
$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^2)$

$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^3)$

degrees $\simeq d/2^\ell$

Modified Rabin's algorithm (for FFT finite fields)

$$X^{p-1} - 1 = \prod_{i=0}^{2^\ell - 1} (X^{M2^{m-\ell}} - \xi^i), \text{ where } \xi \text{ is primitive of order } 2^\ell.$$



$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^0)$

$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^1)$

$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^2)$

$\gcd(f, (X+\tau)^{M2^{m-\ell}} - \xi^3)$

degrees $\simeq d/2^\ell$

Worthwhile in practice for **small** $\ell = 2, 3, \ldots$

New technique: Graeffe transform

Let $f(X) = \prod_i (X - \alpha_i) \in \mathbb{F}_p[X]$.

$$f(X)f(-X) = \prod_i (X - \alpha_i)(-X - \alpha_i) = (-1)^d \prod_i (X^2 - \alpha_i^2)$$

Let $f(X) = \prod_i (X - \alpha_i) \in \mathbb{F}_p[X]$.

$$f(X)f(-X) = \prod_i (X - \alpha_i)(-X - \alpha_i) = (-1)^d \prod_i (X^2 - \alpha_i^2)$$

Definition

$G_2(f)(X) = \prod_i (X - \alpha_i^2)$ is the **Graeffe transform** of f.

Let $f(X) = \prod_i (X - \alpha_i) \in \mathbb{F}_p[X]$.

$$f(X)f(-X) = \prod_i (X - \alpha_i)(-X - \alpha_i) = (-1)^d \prod_i (X^2 - \alpha_i^2)$$

Definition

$G_2(f)(X) = \prod_i (X - \alpha_i^2)$ is the **Graeffe transform** of f.

$G_\rho(f)(X) = \prod_i (X - \alpha_i^\rho)$ is the **Graeffe transform of order ρ** of f.

Let $f(X) = \prod_i (X - \alpha_i) \in \mathbb{F}_p[X]$.

$$f(X)f(-X) = \prod_i (X - \alpha_i)(-X - \alpha_i) = (-1)^d \prod_i (X^2 - \alpha_i^2)$$

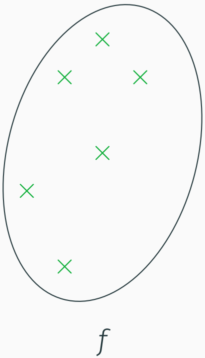
Definition

$G_2(f)(X) = \prod_i (X - \alpha_i^2)$ is the **Graeffe transform** of f.

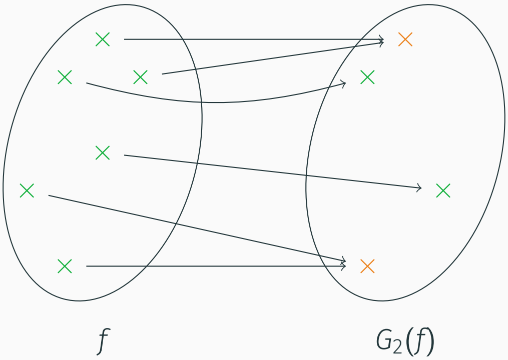
$G_\rho(f)(X) = \prod_i (X - \alpha_i^\rho)$ is the **Graeffe transform of order ρ** of f.

Remarks:

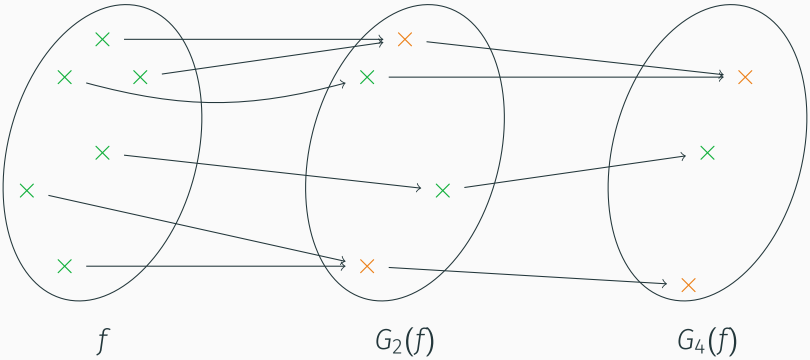
- $G_{\rho_1\rho_2} = G_{\rho_1} \circ G_{\rho_2}$, and in particular $G_{2^\ell} = G_2 \circ \cdots \circ G_2$
- $G_{p-1}(f)(X) = \prod_i (X - \alpha_i^{p-1}) = (X - 1)^d$


f

Effect of Graeffe transforms



Effect of Graeffe transforms



$$f \xrightarrow{G_2} g_1 \xrightarrow{G_2} g_2 \xrightarrow{G_2} \cdots \xrightarrow{G_2} g_m \xrightarrow{G_M} g_{m+1}$$

$$
\begin{array}{ccccccccccc}
f & \xrightarrow{G_2} & g_1 & \xrightarrow{G_2} & g_2 & \xrightarrow{G_2} & \cdots & \xrightarrow{G_2} & g_m & \xrightarrow{G_M} & g_{m+1} \\
 & & & & & & & & \downarrow & & \\
 & & & & & & & & Z_m & \longleftarrow & \{1\}
\end{array}
$$

- $Z_m \subseteq \{\zeta^{i2^m} : 0 \leq i \leq M-1\}$ (ζ: primitive element of $\mathbb{F}_p^*$)

Using Graeffe transforms

$$
\begin{array}{ccccccccccc}
f & \xrightarrow{G_2} & g_1 & \xrightarrow{G_2} & g_2 & \xrightarrow{G_2} & \cdots & \xrightarrow{G_2} & g_m & \xrightarrow{G_M} & g_{m+1} \\
 & & & & & & & & \downarrow & & \\
Z(f) & \longleftarrow & Z_1 & \longleftarrow & Z_2 & \longleftarrow & \cdots & \longleftarrow & Z_m & \longleftarrow & \{1\}
\end{array}
$$

- $Z_m \subseteq \{\zeta^{i2^m} : 0 \leq i \leq M-1\}$ (ζ: primitive element of $\mathbb{F}_p^*$)

Using Graeffe transforms

$$
\begin{array}{ccccccccccc}
f & \xrightarrow{G_2} & g_1 & \xrightarrow{G_2} & g_2 & \xrightarrow{G_2} & \cdots & \xrightarrow{G_2} & g_m & \xrightarrow{G_M} & g_{m+1} \\
 & & & & & & & & \downarrow & & \\
Z(f) & \longleftarrow & Z_1 & \longleftarrow & Z_2 & \longleftarrow & \cdots & \longleftarrow & Z_m & \longleftarrow & \{1\}
\end{array}
$$

- $Z_m \subseteq \{\zeta^{i2^m} : 0 \leq i \leq M-1\}$ (ζ: primitive element of $\mathbb{F}_p^*$)
- For $\beta \in Z_{k+1}$,
  - $\gcd(g_k, X^2 - \beta) = \begin{cases} X - \alpha_i & \text{(simple root)} \\ (X-\alpha_i)(X-\alpha_j) & \text{(multiple root)} \end{cases}$

Using Graeffe transforms

$$\begin{array}{ccccccccccc}
f & \xrightarrow{G_2} & g_1 & \xrightarrow{G_2} & g_2 & \xrightarrow{G_2} & \cdots & \xrightarrow{G_2} & g_m & \xrightarrow{G_M} & g_{m+1} \\
 & & & & & & & & \big\downarrow & & \\
Z(f) & \longleftarrow & Z_1 & \longleftarrow & Z_2 & \longleftarrow & \cdots & \longleftarrow & Z_m & \longleftarrow & \{1\}
\end{array}$$

- $Z_m \subseteq \{\zeta^{i2^m} : 0 \leq i \leq M-1\}$ (ζ: primitive element of $\mathbb{F}_p^*$)
- For $\beta \in Z_{k+1}$,
  - $\gcd(g_k, X^2 - \beta) = \begin{cases} X - \alpha_i & \text{(simple root)} \\ (X-\alpha_i)(X-\alpha_j) & \text{(multiple root)} \end{cases}$
  - Multiple roots: If $\beta = \zeta^e$,

$$\alpha_i, \alpha_j \in \{\zeta^{e/2}, \zeta^{(e+2^m M)/2}\}$$

Theorem

Given $f \in \mathbb{F}_q[X]$ satisfying (A), the irreducible factorization of $(q-1)$ and a primitive element of $\mathbb{F}_q^*$, the roots of f can be computed in time

$$\tilde{O}(\sqrt{S_1(q-1)}d \log^2 q) + (d \log^2 q)^{1+o(1)}$$

where $S_1(q-1)$ is the largest factor of $q-1$.

Theorem

Given $f \in \mathbb{F}_q[X]$ satisfying (A), the irreducible factorization of $(q-1)$ and a primitive element of $\mathbb{F}_q^*$, the roots of f can be computed in time

$$\tilde{O}(\sqrt{S_1(q-1)}d\log^2 q) + (d\log^2 q)^{1+o(1)}$$

where $S_1(q-1)$ is the largest factor of $q-1$.

- Based on:
  - Modular composition [Kedlaya-Umans (2008)]
  - Fast discrete logarithms in $\mathbb{F}_q^*$ [Pohlig-Hellman (1978)]
  - Computation of roots *à la* Pollard-Strassen [Shoup (1991)]

Theorem

Given $f \in \mathbb{F}_q[X]$ satisfying (A), the irreducible factorization of $(q-1)$ and a primitive element of $\mathbb{F}_q^*$, the roots of f can be computed in time

$$\tilde{O}(\sqrt{S_1(q-1)}d \log^2 q) + (d \log^2 q)^{1+o(1)}$$

where $S_1(q-1)$ is the largest factor of $q-1$.

- Based on:
  - Modular composition [Kedlaya-Umans (2008)]
  - Fast discrete logarithms in $\mathbb{F}_q^*$ [Pohlig-Hellman (1978)]
  - Computation of roots *à la* Pollard-Strassen [Shoup (1991)]
- Refines Shoup's complexity bounds

Randomization

Definition

The **tangent Graeffe transform of order π** of $f \in \mathbb{F}_p[X]$ is

$$G_\pi(f + \varepsilon f') \in (\mathbb{F}_p[\varepsilon]/\langle \varepsilon^2 \rangle)[X].$$

Definition

The **tangent Graeffe transform of order π** of $f \in \mathbb{F}_p[X]$ is

$$G_\pi(f + \varepsilon f') \in (\mathbb{F}_p[\varepsilon]/\langle \varepsilon^2 \rangle)[X].$$

Remarks:

- $(f + \varepsilon f')(X) = f(X + \varepsilon)$
- $G_2(f + \varepsilon f') = G_2(f) + \varepsilon \overline{g}$ with $\overline{g}(X^2) = f(X)f'(-X) + f(-X)f'(X)$

Definition

The **tangent Graeffe transform of order π** of $f \in \mathbb{F}_p[X]$ is

$$G_\pi(f + \varepsilon f') \in (\mathbb{F}_p[\varepsilon]/\langle \varepsilon^2 \rangle)[X].$$

Remarks:

- $(f + \varepsilon f')(X) = f(X + \varepsilon)$
- $G_2(f + \varepsilon f') = G_2(f) + \varepsilon \overline{g}$ with $\overline{g}(X^2) = f(X)f'(-X) + f(-X)f'(X)$

Lemma

Let $g + \varepsilon \overline{g} = G_{2^\ell}(f + \varepsilon f')$. **A nonzero root β of g is simple iff $\overline{g}(\beta) \neq 0$.** The corresponding root of f is $\alpha = 2^\ell \beta g'(\beta)/\overline{g}(\beta)$.

Goal: Ensure many simple roots

- Replace f by $f_\tau(X) = f(X + \tau)$ for a random $\tau \in \mathbb{F}_p$.

Goal: Ensure many simple roots

- Replace f by $f_\tau(X) = f(X+\tau)$ for a random $\tau \in \mathbb{F}_p$.

Lemma

If $2^\ell \leq \frac{p-1}{d(d-1)}$, $G_{2^\ell}(f_\tau)$ has **no multiple root** with prob. $\geq 1/2$.

Goal: Ensure many simple roots

- Replace f by $f_\tau(X) = f(X+\tau)$ for a random $\tau \in \mathbb{F}_p$.

Lemma

If $2^\ell \leq \frac{p-1}{d(d-1)}$, $G_{2^\ell}(f_\tau)$ has **no multiple root** with prob. $\geq 1/2$.

$$f(X+\tau+\varepsilon) \xrightarrow{G_2} \cdots \xrightarrow{G_2} g_\ell + \varepsilon \overline{g}_\ell \xrightarrow{G_2} \cdots \xrightarrow{G_2} g_m + \varepsilon \overline{g}_m$$

Goal: Ensure many simple roots

- Replace f by $f_\tau(X) = f(X+\tau)$ for a random $\tau \in \mathbb{F}_p$.

Lemma

If $2^\ell \leq \frac{p-1}{d(d-1)}$, $G_{2^\ell}(f_\tau)$ has **no multiple root** with prob. $\geq 1/2$.

$$f(X+\tau+\varepsilon) \xrightarrow{G_2} \cdots \xrightarrow{G_2} g_\ell + \varepsilon\overline{g}_\ell \xrightarrow{G_2} \cdots \xrightarrow{G_2} g_m + \varepsilon\overline{g}_m$$

$$g_m + \varepsilon\overline{g}_m \longmapsto Z_m \subset \{\xi^e : 0 \leq e < M\}$$

Randomized algorithm

Goal: Ensure many simple roots

- Replace f by $f_\tau(X) = f(X+\tau)$ for a random $\tau \in \mathbb{F}_p$.

Lemma

If $2^\ell \le \frac{p-1}{d(d-1)}$, $G_{2^\ell}(f_\tau)$ has **no multiple root** with prob. $\ge 1/2$.

$$f(X+\tau+\varepsilon) \xrightarrow{G_2} \cdots \xrightarrow{G_2} g_\ell + \varepsilon \overline{g}_\ell \xrightarrow{G_2} \cdots \xrightarrow{G_2} g_m + \varepsilon \overline{g}_m$$

$$g_m + \varepsilon \overline{g}_m \longmapsto Z_m$$

$$Z_\ell \longleftarrow \cdots \longleftarrow Z_m \subseteq \{\xi^e : 0 \le e < M\}$$

Goal: Ensure many simple roots

- Replace f by $f_\tau(X) = f(X+\tau)$ for a random $\tau \in \mathbb{F}_p$.

Lemma

If $2^\ell \leq \frac{p-1}{d(d-1)}$, $G_{2^\ell}(f_\tau)$ has **no multiple root** with prob. $\geq 1/2$.

$$f(X+\tau+\varepsilon) \xrightarrow{G_2} \cdots \xrightarrow{G_2} g_\ell + \varepsilon \overline{g}_\ell \xrightarrow{G_2} \cdots \xrightarrow{G_2} g_m + \varepsilon \overline{g}_m$$

$$g_m + \varepsilon \overline{g}_m \longmapsto Z_m$$

$$Z_0 \xleftarrow[\text{simple roots}]{} Z_\ell \longleftarrow \cdots \longleftarrow Z_m$$

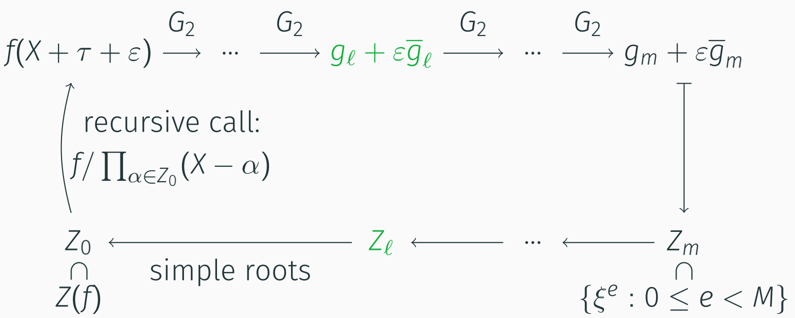
$$Z_0 \subset Z(f) \qquad Z_m \subset \{\xi^e : 0 \leq e < M\}$$

Goal: Ensure many simple roots

- Replace f by $f_\tau(X) = f(X+\tau)$ for a random $\tau \in \mathbb{F}_p$.

Lemma

If $2^\ell \leq \frac{p-1}{d(d-1)}$, $G_{2^\ell}(f_\tau)$ has **no multiple root** with prob. $\geq 1/2$.



Theorem

Given $f \in \mathbb{F}_p[X]$ satisfying (A) and a primitive element of $\mathbb{F}_p^*$, the randomized algorithm runs in **expected time** $\tilde{O}(d \log^2 p)$, for $p = M \cdot 2^m + 1$ with $M = O(\log p)$.

Theorem

Given $f \in \mathbb{F}_p[X]$ satisfying (A) and a primitive element of $\mathbb{F}_p^*$, the randomized algorithm runs in **expected time** $\tilde{O}(d \log^2 p)$, for $p = M \cdot 2^m + 1$ with $M = O(\log p)$.

- Same asymptotic as Rabin's algorithm
- Better efficiency in practice
- Primitive elements easy to compute in practice

Heuristic

If $2^\ell \simeq p/d$, $G_{2^\ell}(f(X+\tau))$ has **$\Omega(d)$ simple roots** with probability $\geq 1/2$, for a random $\tau \in \mathbb{F}_p$.

Justification: holds for a random f rather than $f(X+\tau)$.

Heuristic

If $2^\ell \simeq p/d$, $G_{2^\ell}(f(X+\tau))$ has $\Omega(d)$ **simple roots** with probability $\geq 1/2$, for a random $\tau \in \mathbb{F}_p$.

Justification: holds for a random f rather than $f(X+\tau)$.

$$f(X+\tau+\varepsilon) \xrightarrow{\quad G_{2^\ell} \quad} g_\ell + \varepsilon \overline{g}_\ell$$

Heuristic algorithm

Heuristic

If $2^\ell \simeq p/d$, $G_{2^\ell}(f(X+\tau))$ has $\Omega(d)$ **simple roots** with probability $\geq 1/2$, for a random $\tau \in \mathbb{F}_p$.

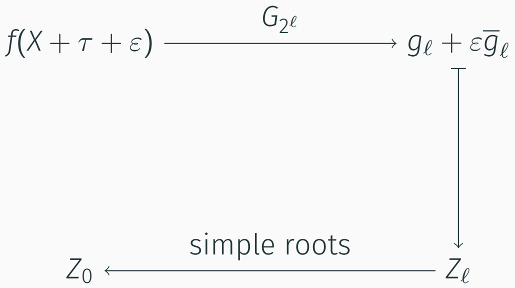
Justification: holds for a random f rather than $f(X+\tau)$.

$$\begin{array}{ccc} f(X+\tau+\varepsilon) & \xrightarrow{\quad G_{2^\ell} \quad} & g_\ell + \varepsilon \overline{g}_\ell \\ & & \downarrow \\ & & Z_\ell \end{array}$$

Heuristic

If $2^\ell \simeq p/d$, $G_{2^\ell}(f(X+\tau))$ has $\Omega(d)$ **simple roots** with probability $\geq 1/2$, for a random $\tau \in \mathbb{F}_p$.

Justification: holds for a random f rather than $f(X+\tau)$.

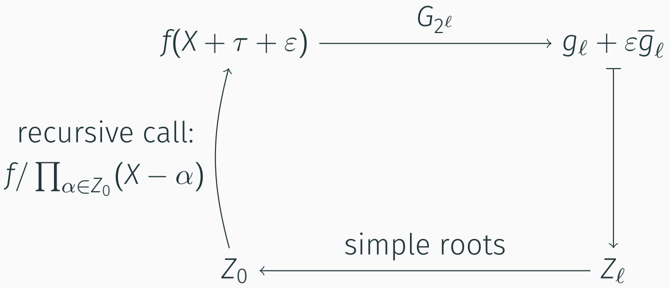


Heuristic algorithm

Heuristic

If $2^\ell \simeq p/d$, $G_{2^\ell}(f(X+\tau))$ has $\Omega(d)$ **simple roots** with probability $\geq 1/2$, for a random $\tau \in \mathbb{F}_p$.

Justification: holds for a random f rather than $f(X+\tau)$.

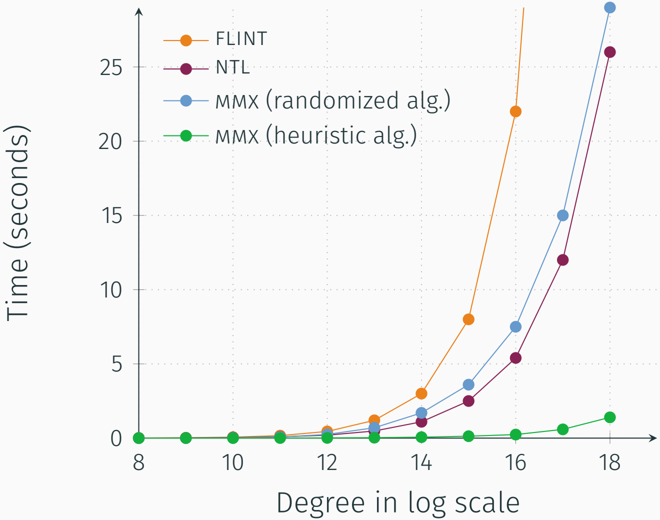


Theorem

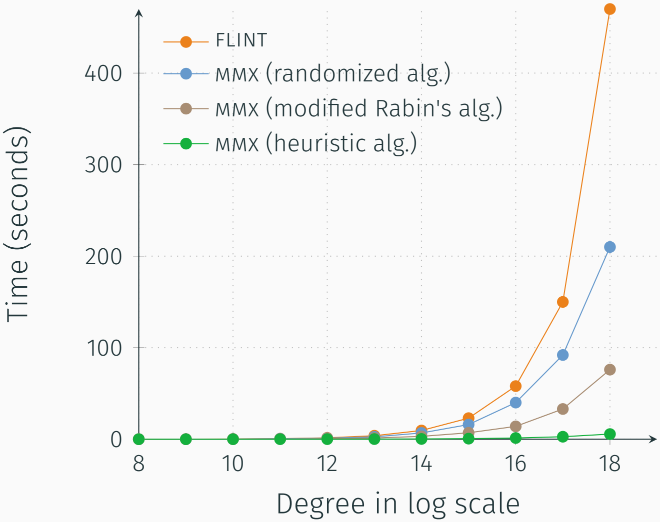
Suppose that f is chosen at random in $\mathbb{F}_p[X]$ or that the heuristic holds. Given a primitive element of $\mathbb{F}_p^*$, the heuristic algorithm runs in **expected time $\tilde{O}(d \log^2 p)$**, for $p = M \cdot 2^m + 1$ with $M = O(\log p)$.

Running times

$p = 7 \cdot 2^{26} + 1$


FLINT
NTL
MMX (randomized alg.)
MMX (heuristic alg.)
Time (seconds)
0
5
10
15
20
25
8
10
12
14
16
18
Degree in log scale

$$p = 5 \cdot 2^{55} + 1$$



FLINT
MMX (randomized alg.)
MMX (modified Rabin's alg.)
MMX (heuristic alg.)
Time (seconds)
400
300
200
100
0
8
10
12
14
16
18
Degree in log scale

- Revisit classical algorithms for FFT finite fields

- Revisit classical algorithms for FFT finite fields
- New approach using Graeffe transforms
    - ✓ deterministic complexity bounds
    - ✓ probabilistic complexity bounds
    - ✓ running times

Conclusion

- Revisit classical algorithms for FFT finite fields
- New approach using Graeffe transforms
  - ✓ deterministic complexity bounds
  - ✓ probabilistic complexity bounds
  - ✓ running times
- Source code in C++ within MATHEMAGIX

Conclusion

- Revisit classical algorithms for FFT finite fields
- New approach using Graeffe transforms
  - ✓ deterministic complexity bounds
  - ✓ probabilistic complexity bounds
  - ✓ running times
- Source code in C++ within MATHEMAGIX
- Not the bottleneck anymore for sparse interpolation

Conclusion

- Revisit classical algorithms for FFT finite fields
- New approach using Graeffe transforms
  - ✓ deterministic complexity bounds
  - ✓ probabilistic complexity bounds
  - ✓ running times
- Source code in C++ within MATHEMAGIX
- Not the bottleneck anymore for sparse interpolation
- Open questions:
  - ? Deterministic alg.: use of tangent Graeffe transforms
  - ? Heuristic alg.: Graeffe transform of order 2^ℓ
  - ? Prove the heuristic

- Revisit classical algorithms for FFT finite fields
- New approach using Graeffe transforms
  - ✓ deterministic complexity bounds
  - ✓ probabilistic complexity bounds
  - ✓ running times
- Source code in C++ within MATHEMAGIX
- Not the bottleneck anymore for sparse interpolation
- Open questions:
  - ? Deterministic alg.: use of tangent Graeffe transforms
  - ? Heuristic alg.: Graeffe transform of order 2^ℓ
  - ? Prove the heuristic

Merci de votre attention !