



Simula
UiB

A New Approach for Solving Determined Polynomial Systems

Irene di Muzio, Morten Øygarden and Enrico Piccione
(work in progress)

Ohrid, Macedonia

August 2026

Polynomial System Solving

Consider a set of m polynomials $f_1, \dots, f_m \in \mathbb{F}[x_1, \dots, x_n]$, and $(a_1, \dots, a_m) \in \mathbb{F}^m$. We want to solve the following system of equations:

$$\begin{aligned} f_1(x_1, \dots, x_n) &= a_1, \\ &\vdots \\ f_m(x_1, \dots, x_n) &= a_m. \end{aligned}$$



Polynomial System Solving

Consider a set of m polynomials $f_1, \dots, f_m \in \mathbb{F}[x_1, \dots, x_n]$, and $(a_1, \dots, a_m) \in \mathbb{F}^m$. We want to solve the following system of equations:

$$\begin{aligned} f_1(x_1, \dots, x_n) &= a_1, \\ &\vdots \\ f_m(x_1, \dots, x_n) &= a_m. \end{aligned}$$

Focus of this talk: we are interested in the *determined* case ($m = n$), with $\deg(f_i) = 2$ and $\mathbb{F}$ is a large finite field.

Exploring the *overdetermined* case $m = n + 1$ will also be interesting for us.



Example: Solving for $m = n + 1$



Example - I

Example of a polynomial equations system over $\mathbb{F}_5[x, y, z]$:

$$\begin{aligned}f_1(x, y, z) &= 3x^2 + 2xy + 1y + 4z, & f_2(x, y, z) &= 2xy + 3y^2 + z + 2 \\f_3(x, y, z) &= 2xz + 2yz + x + 1, & f_4(x, y, z) &= xy + 2xz + 3z^2 + y + 2.\end{aligned}$$



Example - I

Example of a polynomial equations system over $\mathbb{F}_5[x, y, z]$:

$$\begin{aligned}f_1(x, y, z) &= 3x^2 + 2xy + 1y + 4z, & f_2(x, y, z) &= 2xy + 3y^2 + z + 2 \\f_3(x, y, z) &= 2xz + 2yz + x + 1, & f_4(x, y, z) &= xy + 2xz + 3z^2 + y + 2.\end{aligned}$$

$$\begin{array}{c} f_1 \\ f_2 \\ f_3 \\ f_4 \end{array} \begin{pmatrix} x^2 & xy & xz & y^2 & yz & z^2 & x & y & z & 1 \\ \color{blue}{3} & \color{red}{2} & 0 & 0 & 0 & 0 & 0 & \color{brown}{1} & \color{red}{4} & 0 \\ 0 & 2 & 0 & 3 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 2 & 0 & 2 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 2 & 0 & 0 & 3 & 0 & 1 & 0 & 2 \end{pmatrix}$$



Example - II

$$\begin{array}{c}
 x^3 \quad x^2y \quad x^2z \quad xy^2 \quad xyz \quad xz^2 \quad y^3 \quad y^2z \quad yz^2 \quad z^3 \quad \dots \\
 \begin{array}{l}
 xf_1 \\
 yf_1 \\
 zf_1 \\
 xf_2 \\
 yf_2 \\
 zf_2 \\
 xf_3 \\
 yf_3 \\
 zf_3 \\
 xf_4 \\
 yf_4 \\
 zf_4
 \end{array}
 \left(\begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \right)
 \end{array}$$



Example III

There are 12 rows and 10 monomials (columns) of degree = 3. We find two "new" polynomials of degree 2:

$$h_1 = yf_1 + xf_2 = xz + y^2 + 4yz + x,$$

and

$$\begin{aligned} h_2 &= zf_1 + 4zf_2 + (x + 4y)f_3 \\ &= x^2 + 4xy + yz + 3z^2 + x + 4y + 4z. \end{aligned}$$

h_1 and h_2 can now be further multiplied by variables to create more polynomials of degree 3 that can be reduced.



Solution

By continuing this process we eventually recover the polynomials

$$g_1 = x - 4,$$

$$g_2 = y - 1,$$

$$g_3 = z - 2,$$

from which we immediately get the solution $(x, y, z) = (4, 1, 2)$.



Gröbner Basis

On a more conceptual level, the polynomials

$$g_1 = x - 4, \quad g_2 = y - 1, \quad g_3 = z - 2,$$

forms a Gröbner basis for our ideal.

Definition (Gröbner Basis)

A set of polynomials $G = \{g_1, \dots, g_r\}$ is a GB of I (w.r.t. a monomial order $\prec$) if

- i) $\langle G \rangle = I$; and
- ii) for all $f \in I$, there is some j such that $\text{LT}(g_j) \mid \text{Lt}(f)$.



Example: Solving for $m = n$



Example - I

Now consider the same system over $\mathbb{F}_5[x, y, z]$, but remove f_4 , i.e., $I = \langle f_1, f_2, f_3 \rangle$:

$$f_1(x, y, z) = 3x^2 + 2xy + y + 4z, \quad f_2(x, y, z) = 2xy + 3y^2 + z + 2$$

$$f_3(x, y, z) = 2xz + 2yz + x + 1,$$

$$\cancel{f_4(x, y, z) = xy + 2xz + 3z^2 + y + 2.}$$



Example - II

Computing¹ a Gröbner basis G of this ideal (in the grevlex order) yields:

$$g_1 = z^3 + 4z^2 + 3x + y + 2z + 4,$$

$$g_2 = x^2 + z^2 + x + y + z + 3,$$

$$g_3 = xy + z^2 + x + 4y + 3z + 3,$$

$$g_4 = y^2 + z^2 + x + 4y + 2,$$

$$g_5 = xz + 2z^2 + 2x + 3y + 3,$$

$$g_6 = yz + 3z^2 + x + 2y.$$

How are we suppose to find a solution from this?

¹Algorithms for this are reminiscent of, but more involved than the matrix approach we saw in the previous example.



Finding Solutions from G

In this case, G may not be “nice”, but it still has good computational properties, due to:

ii) for all $f \in I$, there is some j such that $\text{LT}(g_j) \mid \text{Lt}(f)$.



Finding Solutions from G

In this case, G may not be “nice”, but it still has good computational properties, due to:

ii) for all $f \in I$, there is some j such that $\text{LT}(g_j) \mid \text{Lt}(f)$.

We want to use this to compute in the quotient ring R/I , where $R = \mathbb{F}_5[x, y, z]$.

In particular, we do operations in R , and reduce with (the leading monomials of) G to get a unique representation in R/I .



Example: Standard Basis of R/I

$$g_1 = z^3 + 4z^2 + 3x + y + 2z + 4,$$

$$g_2 = x^2 + z^2 + x + y + z + 3,$$

$$g_3 = xy + z^2 + x + 4y + 3z + 3,$$

$$g_4 = y^2 + z^2 + x + 4y + 2,$$

$$g_5 = xz + 2z^2 + 2x + 3y + 3,$$

$$g_6 = yz + 3z^2 + x + 2y.$$

Since G is a Gröbner basis, we can reduce by the **leading monomials**. An element in R/I then has a unique representation in **monomials not divisible** by these **leading monomials**.



Multiplication

$\implies R/I$ can be seen as a 5-dimensional $\mathbb{F}_5$ -vector space with basis elements $z^2, x, y, z, 1$.



Multiplication

$\implies R/I$ can be seen as a 5-dimensional $\mathbb{F}_5$ -vector space with basis elements $z^2, x, y, z, 1$.

We can then write the multiplication map by x , i.e.,

$$\begin{aligned} R/I &\longrightarrow R/I \\ f &\longmapsto x \cdot f, \end{aligned}$$

as a matrix $M \in \mathbb{F}_q^{5 \times 5}$

(To find it: multiply each **generator** by x , and use $g_1, \dots, g_6$ to reduce until we are back to a polynomial/vector in $z^2, x, y, z, 1$.)



Finding a Solution for x

The multiplication matrix M has several useful relations with I (e.g., eigenvalues, FGLM-type algorithms). We will use the following:

Proposition

The characteristic polynomial of M ,

$$F(x) = \text{Det}(M - x \cdot Id)$$

is a univariate polynomial in I .



Finding a Solution for x

The multiplication matrix M has several useful relations with I (e.g., eigenvalues, FGLM-type algorithms). We will use the following:

Proposition

The characteristic polynomial of M ,

$$F(x) = \text{Det}(M - x \cdot Id)$$

is a univariate polynomial in I .

$\implies$ The roots of $F(x)$ are the values x can take in a solution for the whole polynomial system. Finding the value for the remaining variables is now (comparatively) easy.



Some Remarks on Our Two Cases

The determined case ($m = n$) is conceptually, and computationally, more complex than the overdetermined case $m = n + 1$.



Some Remarks on Our Two Cases

The determined case ($m = n$) is conceptually, and computationally, more complex than the overdetermined case $m = n + 1$.

You usually don't want to solve the determined case directly. For small to medium sized fields to guess one (or a few) variables and then solve.



Some Remarks on Our Two Cases

The determined case ($m = n$) is conceptually, and computationally, more complex than the overdetermined case $m = n + 1$.

You usually don't want to solve the determined case directly. For small to medium sized fields to guess one (or a few) variables and then solve.

This becomes infeasible for very large fields. Then the best option is the one we just sketched.



Some Remarks on Our Two Cases

The determined case ($m = n$) is conceptually, and computationally, more complex than the overdetermined case $m = n + 1$.

You usually don't want to solve the determined case directly. For small to medium sized fields to guess one (or a few) variables and then solve.

This becomes infeasible for very large fields. Then the best option is the one we just sketched.

Question: Can we do something "in between" these two cases for large fields?



A New Method for Solving

$$m = n$$


Preliminary Thoughts

There are two considerations on the 2nd example that will be important for us.

1) The Gröbner basis computation is expensive, involving a high degree. We want to improve this.



Preliminary Thoughts

There are two considerations on the 2nd example that will be important for us.

- 1) The Gröbner basis computation is expensive, involving a high degree. We want to improve this.
- 2) Relying on a Gröbner basis to compute the multiplication map $R/I \rightarrow R/I$ may be too strict. What we really need is a finite set of elements to express R/I , and a consistent way of mapping between them.



Inspiration from $m = n + 1$

We incidentally saw a natural set of generators for R/I in the $m = n + 1$ case, namely all monomials of degree $< D_{\text{reg}}$.

In the first example we could have used this information to create a multiplication matrix.



Idea

Idea: take a variable, say, x_n and treat it as part of the coefficient ring. (For clarity, we rename $x_n \mapsto z$.) More formally:

$$\mathbb{F}_q[x_1, \dots, x_n] \cong (\mathbb{F}_q[z]) [x_1, \dots, x_{n-1}].$$



Idea

Idea: take a variable, say, x_n and treat it as part of the coefficient ring. (For clarity, we rename $x_n \mapsto z$.) More formally:

$$\mathbb{F}_q[x_1, \dots, x_n] \cong (\mathbb{F}_q[z]) [x_1, \dots, x_{n-1}].$$

The top homogeneous component will now look like the $m = n + 1$ case. We can stop our Macaulay matrix generation at D_{reg} , and use this to generate a multiplication matrix over R/I .



Idea

Idea: take a variable, say, x_n and treat it as part of the coefficient ring. (For clarity, we rename $x_n \mapsto z$.) More formally:

$$\mathbb{F}_q[x_1, \dots, x_n] \cong (\mathbb{F}_q[z]) [x_1, \dots, x_{n-1}].$$

The top homogeneous component will now look like the $m = n + 1$ case. We can stop our Macaulay matrix generation at D_{reg} , and use this to generate a multiplication matrix over R/I .

This achieves our goal of easing the Gröbner basis computation, but it comes at the cost of having z -variables in our multiplication matrix.



Macaulay Matrix in this Setting

$$\left(\begin{array}{c|c} \text{Degree} = D_{\text{reg}} & \text{Degree} < D_{\text{reg}} \\ \hline \textcolor{red}{A} & \textcolor{blue}{B} \end{array} \right)$$

$$\textcolor{red}{A} \in \mathbb{F}^{s \times s} \quad \textcolor{blue}{B} \in (\mathbb{F}[z])^{s \times t}$$



Reducing the Macaulay Matrix

$$\left(\begin{array}{c|c} \text{Degree} = D_{\text{reg}} & \text{Degree} < D_{\text{reg}} \\ \hline I & \textcolor{red}{A}^{-1} \textcolor{blue}{B} \end{array} \right)$$

$$\textcolor{red}{A} \in \mathbb{F}^{s \times s} \quad \textcolor{blue}{B} \in (\mathbb{F}[z])^{s \times t}$$



Multiplication Matrix over $(\mathbb{F}[z])[x_1, \dots, x_{n-1}]/I$

$$\left(\begin{array}{c|c} \text{Degree} = D_{\text{reg}} & \text{Degree} < D_{\text{reg}} \\ \hline I & \textcolor{red}{A}^{-1} \textcolor{blue}{B} \end{array} \right)$$

$\implies$ We can take as a “basis” for $(\mathbb{F}[z])[x_1, \dots, x_{n-1}]/I$ all monomials of degree $< D_{\text{reg}}$, and use $\textcolor{red}{A}^{-1} \textcolor{blue}{B}$ to construct a multiplication matrix.



Multiplication Matrix over $(\mathbb{F}[z])[x_1, \dots, x_{n-1}]/I$

One problem is that B , and hence our multiplication matrix, will contain elements from $\mathbb{F}[z]$.



Multiplication Matrix over $(\mathbb{F}[z])[x_1, \dots, x_{n-1}]/I$

One problem is that B , and hence our multiplication matrix, will contain elements from $\mathbb{F}[z]$.

Proposition

Let $M \in \mathbb{F}[z]$ be a multiplication matrix (w.r.t. x) over R/I . Then the characteristic polynomial of M ,

$$\text{Det}(M(z) - x \times Id) \in \mathbb{F}[x, z],$$

is a bivariate polynomial in I .



Multiplication Matrix over $(\mathbb{F}[z])[x_1, \dots, x_{n-1}]/I$

One problem is that B , and hence our multiplication matrix, will contain elements from $\mathbb{F}[z]$.

Proposition

Let $M \in \mathbb{F}[z]$ be a multiplication matrix (w.r.t. x) over R/I . Then the characteristic polynomial of M ,

$$\text{Det}(M(z) - x \times Id) \in \mathbb{F}[x, z],$$

is a bivariate polynomial in I .

Presumption: Since this characteristic polynomial is in I and only contains two variables, it should be useful in finding solutions.



Computing the Characteristic Polynomial

$\text{Det}(M(z) - x \times Id)$ can be defined, e.g., via the Leibnitz formula, but this is way too inefficient in practice.

Question: How do we compute it efficiently?



Computing the Characteristic Polynomial

$\text{Det}(M(z) - x \times Id)$ can be defined, e.g., via the Leibnitz formula, but this is way too inefficient in practice.

Question: How do we compute it efficiently?

Fast algorithms running in $\mathcal{O}(\ell^\omega)$ require some kind of inversion of matrix elements, which will not be applicable in our case.

Instead, we suggest an interpolation-type approach.



Characteristic Polynomial via Interpolation

$$\text{Det}(M(z) - x \times Id) = f_0(z) + f_1(z) \cdot x + \dots + f_\ell(z) \cdot x^\ell$$



Characteristic Polynomial via Interpolation

$$\text{Det}(M(z) - x \times Id) = f_0(z) + f_1(z) \cdot x + \dots + f_\ell(z) \cdot x^\ell$$

- 1) Pick a constant $a \in \mathbb{F}$.
- 2) Evaluate M at a , $M(a) \in \mathbb{F}^{\ell \times \ell}$.
- 3) Compute the characteristic polynomial of $M(a)$ (using, e.g., Keller-Gehrig), which gives

$$f_0(a) + f_1(a) \cdot x + \dots + f_\ell(a) \cdot x^\ell$$



Characteristic Polynomial via Interpolation

$$\text{Det}(M(z) - x \times Id) = f_0(z) + f_1(z) \cdot x + \dots + f_\ell(z) \cdot x^\ell$$

- 1) Pick a constant $a \in \mathbb{F}$.
- 2) Evaluate M at a , $M(a) \in \mathbb{F}^{\ell \times \ell}$.
- 3) Compute the characteristic polynomial of $M(a)$ (using, e.g., Keller-Gehrig), which gives

$$f_0(a) + f_1(a) \cdot x + \dots + f_\ell(a) \cdot x^\ell$$

With enough $f_i(a_1), f_i(a_2), \dots, f_i(a_t)$ we can now find the $f_i(z)$ by interpolation.



Bivariate Polynomial

We now have a bivariate polynomial $C(z, x) \in I$, but we need at least one more to say something meaningful for solutions of z and x .

Solution: We can repeat the same procedure by reversing the roles of z and x , to get a second polynomial² $C'(z, x) \in I$.

²This works as expected when n is prime, some slight modifications are needed when n is even.



Bivariate Polynomial

We now have a bivariate polynomial $C(z, x) \in I$, but we need at least one more to say something meaningful for solutions of z and x .

Solution: We can repeat the same procedure by reversing the roles of z and x , to get a second polynomial² $C'(z, x) \in I$.

Valid solutions for (x, z) can be recovered from $C(z, x), C'(z, x)$ using existing resultant methods (which we will not cover in this talk). The rest of the solutions is now negligible to find given (x, z) .

²This works as expected when n is prime, some slight modifications are needed when n is even.



Quick Recap

- 1) Macaulay matrix reduction on top part of the system in $(\mathbb{F}[z])[x_1, \dots, x_{n-1}]$.



Quick Recap

- 1) Macaulay matrix reduction on top part of the system in $(\mathbb{F}[z])[x_1, \dots, x_{n-1}]$.
- 2 a) Evaluate $M(a_1), \dots, M(a_t)$ and compute their characteristic polynomials.



Quick Recap

- 1) Macaulay matrix reduction on top part of the system in $(\mathbb{F}[z])[x_1, \dots, x_{n-1}]$.
- 2 a) Evaluate $M(a_1), \dots, M(a_t)$ and compute their characteristic polynomials.
- 2 b) Use the information from 2 a) to compute $f_1(z), \dots, f_\ell(z)$ via interpolation. This yields $C(x, z)$.



Quick Recap

- 1) Macaulay matrix reduction on top part of the system in $(\mathbb{F}[z])[x_1, \dots, x_{n-1}]$.
- 2 a) Evaluate $M(a_1), \dots, M(a_t)$ and compute their characteristic polynomials.
- 2 b) Use the information from 2 a) to compute $f_1(z), \dots, f_\ell(z)$ via interpolation. This yields $C(x, z)$.
 - Perform steps 1) - 2) twice to find $C(x, z), C'(x, z)$.



Quick Recap

- 1) Macaulay matrix reduction on top part of the system in $(\mathbb{F}[z])[x_1, \dots, x_{n-1}]$.
- 2 a) Evaluate $M(a_1), \dots, M(a_t)$ and compute their characteristic polynomials.
- 2 b) Use the information from 2 a) to compute $f_1(z), \dots, f_\ell(z)$ via interpolation. This yields $C(x, z)$.
 - Perform steps 1) - 2) twice to find $C(x, z), C'(x, z)$.
- 3) Compute potential solutions for $C(x, z), C'(x, z)$ via resultant computation.



Quick Recap

- 1) Macaulay matrix reduction on top part of the system in $(\mathbb{F}[z])[x_1, \dots, x_{n-1}]$.
- 2 a) Evaluate $M(a_1), \dots, M(a_t)$ and compute their characteristic polynomials.
- 2 b) Use the information from 2 a) to compute $f_1(z), \dots, f_\ell(z)$ via interpolation. This yields $C(x, z)$.
 - Perform steps 1) - 2) twice to find $C(x, z), C'(x, z)$.
- 3) Compute potential solutions for $C(x, z), C'(x, z)$ via resultant computation.
- 4) Use your favorite method to recover the remaining variables from the candidate solutions for (x, z) .



An Improved Variant

(Glossing over for time)

This works, but the complexity will not that good. The problem is the large multiplication matrices we get by considering all elements of degree $< D_{\text{reg}}$.



An Improved Variant

(Glossing over for time)

This works, but the complexity will not that good. The problem is the large multiplication matrices we get by considering all elements of degree $< D_{\text{reg}}$.

A better method is to collect lower-degree basis elements from step 1), which can be reduced our multiplication matrix. This is fine as long as we avoid z -variables in the leading terms, and is more in line with the normal Gröbner basis approach.



An Improved Variant

(Glossing over for time)

This works, but the complexity will not that good. The problem is the large multiplication matrices we get by considering all elements of degree $< D_{\text{reg}}$.

A better method is to collect lower-degree basis elements from step 1), which can be reduced our multiplication matrix. This is fine as long as we avoid z -variables in the leading terms, and is more in line with the normal Gröbner basis approach.

This comes at a cost of including an extra reduction step (between 1 and 2) to reduce the multiplication matrix. $M(z)$ will now also be of higher degree. Nevertheless, it yields better total complexity.



(Some) Complexity Estimates



Quick Background

Complexity estimation is largely a matter of linear algebra costs for different matrix sizes. For a square matrix of size A , this is $\mathcal{O}(A^\omega)$, with $2 \leq \omega \leq 3$. (We typically use $\omega \approx 2.8$ or $\omega \approx 2.4$.)



Quick Background

Complexity estimation is largely a matter of linear algebra costs for different matrix sizes. For a square matrix of size A , this is $\mathcal{O}(A^\omega)$, with $2 \leq \omega \leq 3$. (We typically use $\omega \approx 2.8$ or $\omega \approx 2.4$.)

The various matrix sizes we've talked about can be determined via the Hilbert series of I , $H_I(t)$.



Quick Background

Complexity estimation is largely a matter of linear algebra costs for different matrix sizes. For a square matrix of size A , this is $\mathcal{O}(A^\omega)$, with $2 \leq \omega \leq 3$. (We typically use $\omega \approx 2.8$ or $\omega \approx 2.4$.)

The various matrix sizes we've talked about can be determined via the Hilbert series of I , $H_I(t)$.

Let $(R/I)_i$ be the elements of R/I in homogeneous degree i . Then we have

$$H_I(t) = \sum_{i=0}^{\infty} \dim (R/I)_i \cdot t^i$$



Hilbert Series and Random Polynomial Systems

We know the Hilbert series for generic (i.e., semi-regular) polynomial systems. This is also assumed to hold for randomly generated polynomial systems with high probability.



Hilbert Series and Random Polynomial Systems

We know the Hilbert series for generic (i.e., semi-regular) polynomial systems. This is also assumed to hold for randomly generated polynomial systems with high probability.

By combining known results we are able to show a very loose lower bound on this probability which is meaningful for some cases with $\mathbb{F}$ extremely large (and depending on n).

We take semi-regularity as an assumption (even in cases without the proven bound).



Complexity of the Approach

In the interest of time, we only note that the bottleneck for the improved variant is dominated by the work to generate the interpolation points (step 2a), i.e., computing the different characteristic polynomials for $M(a)$.



Complexity of the Approach

In the interest of time, we only note that the bottleneck for the improved variant is dominated by the work to generate the interpolation points (step 2a), i.e., computing the different characteristic polynomials for $M(a)$.

Let I' denote the top homogeneous component of our polynomial system where we have set $x_n = 0$. Then this step is upper bounded by

$$\mathcal{O} \left(D_{\text{reg}, I'} \cdot (H_{I'}(1))^{\omega+1} \right) .$$



Some Rough Comparisons

We use $\mathcal{O}\left(\binom{n+D_{\text{reg}}}{n}^\omega\right)$ for the previous method (complexity of F_5).
Here $D_{\text{reg}} = n + 1$.

n	F_5 $\omega = 2.81$	This talk $\omega = 2.81$	F_5 $\omega = 2.37$	This talk $\omega = 2.37$
20	107	70	90	62
30	162	108	137	96
50	274	183	231	163
80	441	297	372	263
100	553	373	467	330

Table: $\log_2$ complexities for some determined polynomial systems.



Concluding Remarks

- Work in progress, things can change.
- I don't know the exact crossover for field size, but for small fields I still expect the usual hybrid-type solvers (say, crossbred or FXL with Wiedemann) to dominate.
- I've glossed over a lot of details here, but I'm happy to discuss things further throughout this week.



Potential Further Directions/Open Problems (?)

- Better complexity comparisons.
- Will it be meaningful to expand these techniques using more than one variable? (I expect diminishing returns quickly, but perhaps there is a space to explore?)
- Can we do something clever for smaller fields?
- Interpolating the characteristic polynomial is somewhat naive. Can we develop better tools for linear algebra over a polynomial ring with few indeterminates?

