Inverting Matrices Modulo 0-Dim Regular Chains

Paul Vrbik

University of Western Ontario

April 8, 2011
Consider a subset $F = \{f_1, \ldots, f_n\} \subset \mathbb{Q}[x_1, \ldots, x_n]$ which you would like to “solve”.

A 0-dimensional Regular Chain is another subset $T = \{T_1, \ldots, T_n\} \subset \mathbb{Q}[x_1, \ldots, x_n]$ such that

$\{x \mid f_1(x) = 0, \ldots, f_n(x) = 0\} = \{x \mid T_1(x) = 0, \ldots, T_n(x) = 0\}$

zeros of $F = $ zeros of $T$

1. The equations of $T$ admit trivial back substitution.
Example (Regular Chain)

Consider the system of equations $F = \{ f_1, \ldots, f_3 \} \subset \mathbb{Q}[x, y, z]$

\[
\begin{align*}
    x^2 + y^2 + z^2 - 1 &= 0 \\
    x^2 + z^2 - y &= 0 \\
    x - z &= 0.
\end{align*}
\]

A regular chain is given by $T = \{ T_1, \ldots, T_3 \} \subset \mathbb{Q}[x, y, z]$

\[
\begin{align*}
    x - z &= 0 & \in \mathbb{Q}[x, y, z] \\
    -y + 2z^2 &= 0 & \in \mathbb{Q}[y, z] \\
    z^4 + \frac{z^2}{2} - \frac{1}{4} &= 0 & \in \mathbb{Q}[z]
\end{align*}
\]
The example on the previous slide was a zero dimensional regular chain.

Zero dimensional regular chains are:

- derived from “squares systems” (number of equations and unknowns are equal) that have a *finite* number of zeros,
- of great interest in practice because we can apply modular methods to them,
- well suited for defining algebraic rings.
Polynomial Division

In high school we are taught how to divide polynomial $f, g \in \mathbb{Q}[x]$ to produce a quotient and remainder $q$ and $r$ so that $f = qg + r$.

To extend this to multivariates one just needs to specify a *monomial ordering* (i.e. define the leading term).

\[
q = x^3z + y^2 + z \\
\frac{x^2z + 1}{x^5z^2 + x^4y + x^2y^2z + x^3z + x^2z^2 + y^2}
\]

\[
q = x^3z + y^2 + z \\
x^5z^2 + x^3z \\
x^4y + x^2y^2z + x^2z^2 + y^2 \\
x^2y^2z + x^2z^2 + y^2 \\
x^2y^2z + y^2 \\
x^2z^2 \\
x^2z^2 + z \\
\rightarrow -z \\
0
\]

\[
r = x^4y - z \\
x^4y \\
\rightarrow -z
\]
Polynomial Division (of sets)

It is also possible to take a set of divisors

\[ G = \{g_1, \ldots, g_m\} \subset \mathbb{Q}[x_1, \ldots, x_n] \]

and do \( f \div G \) to produce \( \{q_1, \ldots, q_m\} \) and \( r \) (in \( \mathbb{Q}[x_1, \ldots, x_n] \)) so that

\[ f = q_1g_1 + \cdots + q_mg_m + r. \]

(Just do the regular division algorithm but choose any divisor at each step).

In general this operation is not well defined but when \( \{g_1, \ldots, g_m\} \) is a zero-dimensional regular chain the remainder is unique.

From now on assume \( f \mod \langle G \rangle \) returns the remainder when dividing \( f \) by \( G \).
The Quotient Ring of a Regular Chain

Suppose we have a zero dimensional regular chain $T \subset \mathbb{Q}[x_1, \ldots, x_n]$. We can now create an equivalence in $\mathbb{Q}[x_1, \ldots, x_n]$ that says $f = g$ when

$$f \mod \langle T \rangle \equiv g \mod \langle T \rangle.$$ 

This means that all we really care about are all possible remainders.

The (finite) set of all possible remainders on $\div T$ is called the quotient ring of $T$ and is denoted

$$\mathbb{Q}[x_1, \ldots, x_n]/\langle T \rangle = \{ f \mod \langle T \rangle \mid f \in \mathbb{Q}[x_1, \ldots, x_n] \}.$$
Working in Quotient Rings

A ring is a set with multiplication and 1, addition and 0. There may be some elements of the ring with multiplicative inverse, i.e.

\[ fg \mod \langle T \rangle = 1, \]

The extended euclidean algorithm (i.e. gcd algorithm) can calculate inverses by finding successive polynomial remainders.

**Example (Invertible Elements)**

Let \( m = x^3 - x + 2, \ a = x^2 \in \mathbb{Q}[x] \). The last row in the extended euclidean algorithm is

\[
\left( \frac{1}{4}x + \frac{1}{2} \right) (x^3 - x + 2) + \left( -\frac{1}{4}x^2 - \frac{1}{2}x + \frac{1}{4} \right) x^2 = 1,
\]

so \((-x^2 - 2x + 1)/4\) is the inverse of \( a \) modulo \( m \).
Linear Algebra In Quotient Rings

Our goal is to extend this inversion to Matrices.

Specifically, given a matrix $A \in \mathbb{Q}[x_1, \ldots, x_n]/\langle T \rangle^{m \times m}$ we want to find $B \in \mathbb{Q}[x_1, \ldots, x_n]/\langle T \rangle^{m \times m}$ so that

$$A \cdot B \equiv \begin{bmatrix} 1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & 1 \end{bmatrix} \mod \langle T \rangle.$$

(Remember that 1 and 0 are obtained after dividing by $T$).
Matrix Inversion Algorithms

**Naive**

Gauss-Jordan elimination does pivoting and requires many inversions/divisions.

This inversion is a bottleneck, mainly due to memory consumption.

**Leverrier-Faddeev**

Is a scheme for finding a matrix inverse that requires only a single division.
Leverrier-Faddeev Algorithm

Consider the characteristic polynomial of the $n \times n$ matrix $A$,

\[ p(\lambda) = \det (\lambda I - A) = \lambda^n - a_1\lambda^{n-1} - \cdots - a_{n-1}\lambda - a_n. \]

Evaluating $p(A)$, multiplying by $A^{-1}$ and re-arranging terms gives,

\[ 0 = A^n - a_1A^{n-1} - \cdots - a_{n-1}A - a_n \tag{1} \]

\[ A^{-1}a_n = A^{n-1} - a_1A^{n-2} - \cdots - a_{n-1} \tag{2} \]

\[ A^{-1} = \left( A^{n-1} - \sum_{i=1}^{n-1} a_iA^{n-i-1} \right) a_n^{-1}. \tag{3} \]

The $a_k$’s can be obtained in a successive manner by

\[ a_k = \frac{1}{k} \left( s_k - \sum_{i=1}^{k-1} s_{k-i}a_i \right). \tag{4} \]

where $s_k = \text{trace}(A^k)$ and $a_1 = s_1$. 
Optimizing Calculating the $s_k$’s

We can reduce the $n^4$ multiplications required to calculate $A^1, \ldots, A^{n-1}$ by doing something like repeated squaring. Let $d = \lceil \sqrt{n} \rceil$, if we instead store the sequence

$$M_0, M_1, M_2, \ldots, M_d = A^0, A^1, A^2, \ldots, A^d$$

and generate the sequence

$$N_0, N_1, N_2, \ldots, N_k = A^0, A^{d+1}, A^{2(d+1)}, \ldots, A^{2k(d+1)}$$

on the fly (using repeated multiplying by $A^{d+1}$, without storing). Then we can compute the required traces by

$$\text{tr}(M_i N_j) = \text{tr}(A^i A^{(d+1)j}) = \text{tr}(A^{i+(d+1)j})$$

taking $0 \leq i, j \leq d$. 

Spatial Optimization for Calculating the $s_k$'s

That is, we are calculating the traces in “blocks”, i.e. for $n = 8$ with $d = \lfloor \sqrt{8} \rfloor = 2$ we do

\[
\{ \text{tr}(A^0 A^0), \ldots, \text{tr}(A^0 A^2) \} \\
\{ \text{tr}(A^3 A^0), \ldots, \text{tr}(A^3 A^2) \} \\
\{ \text{tr}((A^3 A^3) A^0), \ldots, \text{tr}((A^3 A^3) A^2) \}
\]

$2n^3 \sqrt{n} \times$'s to get the $M$'s and $N$'s

$n^2 \times$'s $n$ times to get the traces (we assume some optimization has been done to calculate $\text{tr}(AB)$ by only calculating the diagonal of $AB$).

Therefore we only require $n^3 + 2n^3 \sqrt{n} = n^3(1 + 2\sqrt{n})$ multiplications to calculate the $s_k$’s.
Spatial Optimization for the Expansion.

The final step requires us to do

\[
A^{-1} = \left( A^{n-1} - \sum_{i=1}^{n-1} a_i A^{n-i-1} \right) a^{-1}_n,
\]

which makes it look like we are required to either recalculate or store \( A^0, \ldots, A^{n-1} \).

This better not be the case because it would render our last optimization useless!

Observe that any polynomial can be re-written in Horner (or nested) form,

\[
p(x) = a_0 x^n + \cdots + a_{n-1} x + a_n
\]

\[
= (((a_0 x + a_1) x + a_2) x + \cdots + a_{n-1}) x + a_n.
\]

Now think of the indeterminate \( x \) as a linear combination of the \( M \)'s.
Spatial Optimization for the Expansion.

Now to express this in the our modified Horner form let

\[ s \equiv n \mod (d + 1) \quad \text{and} \quad \sigma(k) = \sum_{i=s+kd+k}^{s+kd+k+d} a_i M_{(n-i-1)} \mod (d+1) \]

then

\[ p(A) = \left( \cdots \left( \left( \left( \sum_{i=0}^{s-1} a_i M_{s-1-i} \right) N_1 + \sigma(0) \right) N_1 + \sigma(1) \right) N_1 + \cdots \right) N_1 \]

\[ + \sigma \left( \frac{n - 1 - d - s}{d + 1} \right). \]

(Yes, I do indeed have an inductive proof for this. Yes, it’s ugly).
Spatial Optimization for the Expansion.

The complexity is given by the matrix multiplications needed to do $\sigma(k)$ and $\sum_{i=0}^{s-1} a_i M_{s-1-i}$.

So, $n^3$ many multiplications $\left(\frac{n+1+d-s}{d+1}\right)$-times.

To express this as a function in $n$ recall that $s = n \mod (d + 1) \leq d$ and $d \leq \sqrt{n}$.

$$n^3 \cdot \frac{n + 1 + d - s}{d + 1} < n^3 \left(\frac{n + 1 + \sqrt{n}}{1 + \sqrt{n}}\right) < n^3 \left(\frac{n}{\sqrt{n}} + 1\right) < O \left(n^3 \sqrt{n}\right).$$
Using Lev-Fad Recursively

We can do the single inversion required by the Lev-Fad algorithm using the Lev-Fad algorithm.

In order to do this we need to build a mapping between elements in \( \mathbb{Q}[x_1, \ldots, x_n]/\langle T \rangle \) and matrices in \((\mathbb{Q}[x_1, \ldots, x_n]/\langle T \rangle)^{m \times m}\).

\[
m_f : \mathbb{Q}[x_1, \ldots, x_n] \rightarrow \mathbb{Q}[x_1, \ldots, x_n]
\]
\[
\alpha \mapsto f \alpha
\]

That is \( m_f(g) = f \cdot g \).

If \( \mathbb{Q}[x_1, \ldots, x_n]/\langle T \rangle \) is finite so it will have a finite monomial basis \( B \). We can thus represent \( m_f \) by its matrix with respect to this basis.
Example

Let $T = \langle y^2 - 1, x^2 - 1 \rangle$ monomial basis $B = \{1, y\}$. The multiplication matrix for the element $a = x - y$ is

$$m_a = \begin{bmatrix} x & -1 \\ -1 & x \end{bmatrix}.$$

To do $a \cdot 1 = 1 \cdot 1 + 0 \cdot y$ we calculate

$$\begin{bmatrix} x & -1 \\ -1 & x \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & y \end{bmatrix} = x - y$$

or to do $a \cdot y = 0 \cdot 1 + 1 \cdot y$ we calculate

$$\begin{bmatrix} x & -1 \\ -1 & x \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & y \end{bmatrix} = xy - y^2.$$
Using Lev-Fad Recursively

The multiplication matrix satisfies

\[ m_f \cdot m_g = m_{fg} \]

and thus we can find the inverse of \( f \) by inverting it’s corresponding multiplication matrix because

\[ m_f m_{f^{-1}} = m_{ff^{-1}} = m_1 = \mathbb{I} = (m_f) \cdot (m_f)^{-1} \Rightarrow m_{f^{-1}} = (m_f)^{-1}. \]
Space Complexity

Let $F(m, [d_1, \ldots, d_n])$ be the number of field elements required to invert an $m \times m$ matrix modulo a regular chain $T = \langle T_1, \ldots, T_n \rangle \subset \mathbb{Z}_p[x_1, \ldots, x_n]$ with $d_i = \text{degree}_{x_i}(T_i)$. Assuming completely dense input we have

$$F(m, [d_1, \ldots, d_n]) = m \cdot m \cdot d_1 \cdots d_n \quad \text{input}$$

$$+ m \cdot d_1 \cdots d_n \quad \text{traces}$$

$$+ F(d_n, [d_1, \ldots, d_{n-1}]) \quad \text{recursive call}$$

$$+ m \cdot m \cdot d_1 \cdots d_m \quad \text{expansion}$$

Letting $\sigma = \prod \text{degree}_{x_i}(T_i)$ and $\delta = \sum \text{degree}_{x_i}(T_i)$ we can bound the above recurrence by $O(2^m \delta + m^2 \delta + \delta \sigma)$ field elements.
Experimental Results

Random dense regular chain $T \subset \mathbb{Z}_p[x_1, \ldots, x_n]$ with degree$(T_i) = 6$, varying $n$ and $p = 962592769$. Our matrix is a random (invertible) $m \times m$ matrix with dense entries from $\mathbb{Z}_p[x_1, \ldots, x_n]/\langle T \rangle$.

<table>
<thead>
<tr>
<th>Vars</th>
<th>Matrix Size</th>
<th>Time</th>
<th>Trace</th>
<th>Inv</th>
<th>Expand</th>
<th>Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>11 $\times$ 11</td>
<td>157.34s</td>
<td>0.06%</td>
<td>2.74%</td>
<td>97.21%</td>
<td>0.10GB</td>
</tr>
<tr>
<td>4</td>
<td>7 $\times$ 7</td>
<td>408.15s</td>
<td>37.65%</td>
<td>10.56%</td>
<td>51.80%</td>
<td>0.11GB</td>
</tr>
<tr>
<td>5</td>
<td>1 $\times$ 1</td>
<td>800.43s</td>
<td>19.24%</td>
<td>60.91%</td>
<td>19.85%</td>
<td>0.41GB</td>
</tr>
</tbody>
</table>

**GCD Based**

<table>
<thead>
<tr>
<th>Vars</th>
<th>Matrix Size</th>
<th>Time</th>
<th>Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>11 $\times$ 11</td>
<td>1102.310s</td>
<td>0.18GB</td>
</tr>
<tr>
<td>4</td>
<td>7 $\times$ 7</td>
<td>–</td>
<td>4.0GB</td>
</tr>
<tr>
<td>5</td>
<td>1 $\times$ 1</td>
<td>*</td>
<td>&gt;4.0GB</td>
</tr>
</tbody>
</table>
Thanks

Dr Marc Moreno Maza and Dr Éric Shost.