Sparse Hensel Lifting Algorithms for Multivariate Polynomial Factorization

Tian Chen

Department of Mathematics,
Simon Fraser University,
Canada

Multivariate Polynomial Factorization

Problem $\mathcal{P}$: Given a polynomial $a \in \mathbb{Z}[x_1, \cdots, x_n]$, compute the irreducible factors of a with coefficients in $\mathbb{Z}$.

Note that we do not factor the integer content.
E.g., $6x^2 - 6y^2 = 6(x + y)(x - y)$.

- Polynomial factorization is one of the central problems in computer algebra.
- My work focuses on the **design**, **analysis** and **implementation** of algorithms to solve problem $\mathcal{P}$.

Multivariate polynomial factorization – a brief history

Hensel lifting

- Zassenhaus (1969): Hensel lifting for univariate polynomials in $\mathbb{Z}[x]$.
- Yun (1974), Wang (1975), (1978): **Multivariate Hensel lifting**. Recovers the factors one variable at a time. Solves the multivariate Diophantine equation (MDP) $\sigma_i g_{j-1} + \tau_i f_{j-1} = c_i$ for $\sigma_i, \tau_i \in \mathbb{Z}_p[x_1, ..., x_{j-1}]$ one variable at a time (can be exponential in n).

Sparse Hensel lifting

- Zippel (1981), Kaltofen (1985): Sparse Hensel lifting (SHL).
- Monagan and Tuncer (2016): MTSHL. Solves MDP by sparse interpolation.
- Monagan and Tuncer (2018): Use bivariate Hensel lifts to compute σ_i, τ_i.
- Chen and Monagan (2020): CMSHL. No expression swell, no multivariate polynomial arithmetic, highly parallelizable. Complexity analysis for both MTSHL and CMSHL.
  Dominating cost is evaluating the input polynomial –> consider black box

Multivariate polynomial factorization – a brief history

Black box factorization

- Kaltofen and Trager (1990): First computes the black boxes of the factors, then uses sparse polynomial interpolation to recover the sparse representation of the factors.
- Rubinfeld and Zippel (1994): For factoring $a \in \mathbb{Z}[x_1, \cdots, x_n]$.
- Diaz and Kaltofen (1998): FOXBOX. Implemented in C++.
- Chen and Monagan (2022), (2023): A modular algorithm. Output factors in the sparse representation directly. Needs fewer probes to the black box than Kaltofen and Trager and Rubinfeld and Zippel's algorithms.

Other work

- Huang and Gao (2023): Does not use Hensel lifting. Not efficient for polynomials with large number of terms.
- Lecerf (2007): Dense multivariate polynomial factorization

Contributions

1. T. Chen and M. Monagan. The complexity and parallel implementation of two sparse multivariate Hensel lifting algorithms for polynomial factorization. In Proceedings of CASC 2020, LNCS **12291**, 150–169. Springer (2020)
2. T. Chen and M. Monagan. Factoring multivariate polynomials represented by black boxes: A Maple + C Implementation. *Math. Comput. Sci.* **16**,18 (2022)
3. T. Chen and M. Monagan. A new black box factorization algorithm - the non-monic case. In Proceedings of ISSAC 2023, pp. 173–181. ACM (2023)

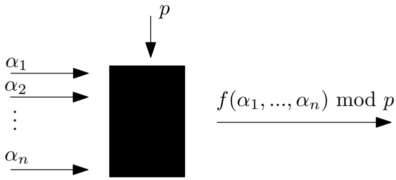
Table of content

The sparse and black box representation of a polynomial

The **sparse representation** [Von zur Gathen and Gerhard (2013)] of $f \in \mathbb{Z}[x_1, \cdots, x_n]$ consists of a list of coefficients $c_k \in \mathbb{Z}$, $c_k \neq 0$ and distinct exponent vectors $\vec{e_k} = (e_{k_1}, \cdots, e_{k_n}) \in \mathbb{N}^n$ such that

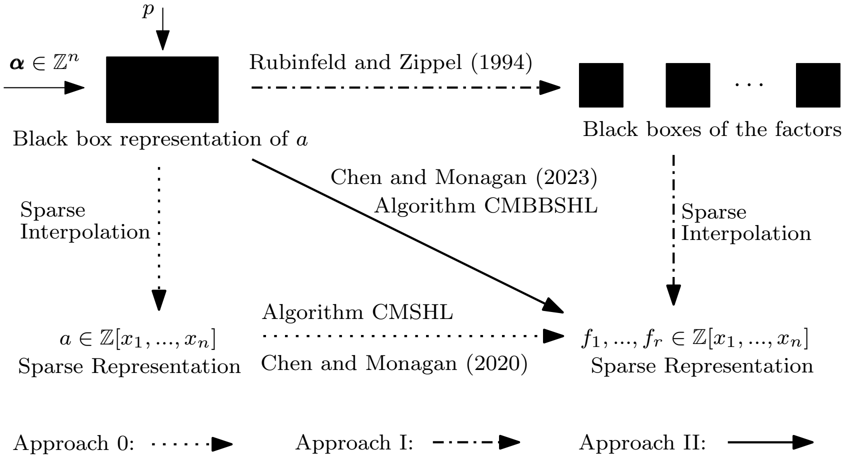
$$f = \sum_{k=1}^{\#f} c_k \cdot x_1^{e_{k_1}} \cdots x_n^{e_{k_n}}.$$

A **modular black box representation** of $f \in \mathbb{Z}[x_1, \cdots, x_n]$ is a computer program $\mathrm{B} : \mathbb{Z}^n \times \{p\} \to \mathbb{Z}_p$ that on input $\boldsymbol{\alpha} \in \mathbb{Z}^n$ and a prime p outputs $\mathrm{B}(\boldsymbol{\alpha}, p) = f(\boldsymbol{\alpha}) \bmod p$.



Factoring $a \in \mathbb{Z}[x_1, \cdots, x_n]$ represented by a black box

Given a polynomial $a \in \mathbb{Z}[x_1, \cdots, x_n]$ represented by a black box, we aim to compute its factors in the sparse representation.



Example: Computing the determinant of a Toeplitz matrix

Let

$$T_n = \begin{pmatrix} x_1 & x_2 & x_3 & \cdots & x_n \\ x_2 & x_1 & x_2 & & \\ x_3 & x_2 & x_1 & & \\ \vdots & & & \ddots & \vdots \\ x_n & & & \cdots & x_1 \end{pmatrix}.$$

For example, $\det(T_4) = (x_1^2 - x_1x_2 - x_1x_4 - x_2^2 + 2x_2x_3 + x_2x_4 - x_3^2)(x_1^2 + x_1x_2 + x_1x_4 - x_2^2 - 2x_2x_3 + x_2x_4 - x_3^2)$.

n	$\#\det(T_n)$	$\#f_i$
8	1628	167, 167
9	6090	294, 153
10	23797	931, 931
11	90296	1730, 849
12	350726	5579, 5579
13	1338076	10611, 4983
14	5165957	34937, 34937
15	19732508	66684, 30458
16	–	221854, 221854

Table: Number of terms of $\det(T_n)$ and its factors [Chen and Monagan (2022)].

Algorithm CMBBSHL (Approach II):

- Space efficient since $\boxed{\#f_i \ll \#\det(T_n)}$.
- Fewer probes to the black box than Rubinfeld and Zippel's algorithm.

Example: Computing the determinant of a Toeplitz matrix

Let $a = \det(T_n) \in \mathbb{Z}[x_1, \cdots, x_n]$. The modular black box representation of a can be coded in Maple as a procedure:

```
B := proc( alpha::Array, p::prime )
    local n := numlems(alpha), i,j,Tn;
    Tn := Matrix(n,n);
    for i to n do
        for j to n do
            Tn[i,j] := alpha[abs(i-j)+1];
        od;
    od;
    Det(Tn) mod p;
end:
```

Example (Computing the factors of $\det(T_4)$)

- Let $p = 101$ and choose $\boldsymbol{\alpha} = (3, 5, 4)$.

Example (Computing the factors of $\det(T_4)$)

- Let $p = 101$ and choose $\boldsymbol{\alpha} = (3, 5, 4)$.
- $a(x_1, \boldsymbol{\alpha}) = x_1^4 - 93x_1^2 + 420x_1 - 416 = (x_1^2 - 7x_1 + 8)(x_1^2 + 7x_1 - 52) \in \mathbb{Z}[x_1]$.

Example (Computing the factors of $\det(T_4)$)

- Let $p = 101$ and choose $\boldsymbol{\alpha} = (3, 5, 4)$.
- $a(x_1, \boldsymbol{\alpha}) = x_1^4 - 93x_1^2 + 420x_1 - 416 = (x_1^2 - 7x_1 + 8)(x_1^2 + 7x_1 - 52) \in \mathbb{Z}[x_1]$.
- The first Hensel lifting step recovers x_2 and we get

$$f_2 = x_1^2 - x_1x_2 - x_2^2 - 4x_1 + 14x_2 - 25,$$
$$g_2 = x_1^2 + x_1x_2 - x_2^2 + 4x_1 - 6x_2 - 25.$$

Example (Computing the factors of $\det(T_4)$)

- Let $p = 101$ and choose $\boldsymbol{\alpha} = (3, 5, 4)$.
- $a(x_1, \boldsymbol{\alpha}) = x_1^4 - 93x_1^2 + 420x_1 - 416 = (x_1^2 - 7x_1 + 8)(x_1^2 + 7x_1 - 52) \in \mathbb{Z}[x_1]$.
- The first Hensel lifting step recovers x_2 and we get

$$\begin{aligned} f_2 &= x_1^2 - x_1x_2 - x_2^2 - 4x_1 + 14x_2 - 25, \\ g_2 &= x_1^2 + x_1x_2 - x_2^2 + 4x_1 - 6x_2 - 25. \end{aligned}$$

- The second Hensel lifting step recovers x_3 and

$$\begin{aligned} f_3 &= x_1^2 - x_1x_2 - x_2^2 + 2x_2x_3 - x_3^2 - 4x_1 + 4x_2, \\ g_3 &= x_1^2 + x_1x_2 - x_2^2 - 2x_2x_3 - x_3^2 + 4x_1 + 4x_2. \end{aligned}$$

Example (Computing the factors of $\det(T_4)$)

- Let $p = 101$ and choose $\boldsymbol{\alpha} = (3, 5, 4)$.
- $a(x_1, \boldsymbol{\alpha}) = x_1^4 - 93x_1^2 + 420x_1 - 416 = (x_1^2 - 7x_1 + 8)(x_1^2 + 7x_1 - 52) \in \mathbb{Z}[x_1]$.
- The first Hensel lifting step recovers x_2 and we get

$$\begin{aligned} f_2 &= x_1^2 - x_1x_2 - x_2^2 - 4x_1 + 14x_2 - 25, \\ g_2 &= x_1^2 + x_1x_2 - x_2^2 + 4x_1 - 6x_2 - 25. \end{aligned}$$

- The second Hensel lifting step recovers x_3 and

$$\begin{aligned} f_3 &= x_1^2 - x_1x_2 - x_2^2 + 2x_2x_3 - x_3^2 - 4x_1 + 4x_2, \\ g_3 &= x_1^2 + x_1x_2 - x_2^2 - 2x_2x_3 - x_3^2 + 4x_1 + 4x_2. \end{aligned}$$

- At the final step, we recover x_4 and obtain the true factors

$$\begin{aligned} f &= x_1^2 - x_1x_2 - x_1x_4 - x_2^2 + 2x_2x_3 + x_2x_4 - x_3^2, \\ g &= x_1^2 + x_1x_2 + x_1x_4 - x_2^2 - 2x_2x_3 + x_2x_4 - x_3^2. \end{aligned}$$

Tools for sparse Hensel lifting

- The Schwartz-Zippel lemma
- Hilbertian point
- The weak SHL assumption
- Square-free factorization
- Bivariate Hensel lifting
- Solving Vandermonde systems of equations
- Rational number reconstruction

Tools for Hensel lifting

Prior to sparse Hensel lifting, an evaluation point $\boldsymbol{\alpha} \in \mathbb{Z}^{n-1}$ is chosen randomly from $[1, \tilde{N} - 1]^{n-1}$ where $\tilde{N} \in \mathbb{Z}^+$ and $\tilde{N} < p$.

For algorithm CMSHL and CMBBSHL to succeed, $\boldsymbol{\alpha}$ must satisfy

1. $\boldsymbol{\alpha}$ is Hilbertian;
2. $\boldsymbol{\alpha}$ satisfies the weak SHL assumption.

If $\tilde{N}$ is large (e.g. $\tilde{N} = 4001$), then ① and ② hold w.h.p.

Hilbertian point

Definition

Let $P \in \mathbb{Z}[x_1, \cdots, x_n]$ be an irreducible polynomial over $\mathbb{Z}$. We call a point $\boldsymbol{\alpha} = (\alpha_2, \cdots, \alpha_n) \in \mathbb{Z}^{n-1}$ **Hilbertian** if $P(x_1, \alpha_2, \cdots, \alpha_n)$ remains irreducible over $\mathbb{Z}$ and $\deg(P(x_1, \boldsymbol{\alpha})) = \deg(P, x_1)$ [Lee (2001)].

Example

Let $P = x^3 + y^3 + 1 \in \mathbb{Z}[x, y]$. The only non-Hilbertian points are $y = 0$ and $y = -1$.

The weak SHL assumption

Definition

Let $\alpha_j \in \mathbb{Z}_p$ be chosen at random. Let

$$f = \sum_{i=0}^{d_j} \sigma_i(x_1, \cdots, x_{j-1})(x_j - \alpha_j)^i \in \mathbb{Z}_p[x_1, \cdots, x_j]$$

be the Taylor polynomial of $f \in \mathbb{Z}_p[x_1, \cdots, x_n]$ about α_j of degree $d_j = \deg(f, x_j)$. Let $\mathrm{Supp}(\sigma_i)$ be the set of all monomials in σ_i. The assumption that $\mathrm{Supp}(\sigma_i) \subseteq \mathrm{Supp}(\sigma_0)$ for all $1 \le i \le d_j$ is called **the weak SHL assumption** [Monagan and Tuncer (2020)]

Lemma (Lemma 3.3.2)

$$\Pr[\mathrm{Supp}(\sigma_i) \not\subseteq \mathrm{Supp}(\sigma_0)] \le |\mathrm{Supp}(\sigma_i)| \frac{d_j}{p - d_j + i} \text{ for } 1 \le i \le d_j.$$

Algorithm CMSHL

$$
\begin{tikzcd}
f_j(x_1, x_j) = \sum_{i=0}^{df_j} \sigma_i(x_1)(x_j - \alpha_j)^i \arrow[r] \arrow[d, dashed, "\text{Sparse Interpolation}"] & \sum_{i=0}^{df_j} \bar{\sigma}_i(x_1) x_j^i \arrow[d, "\text{Sparse Interpolation}"] \\
f_j(x_1, \cdots, x_j) = \sum_{i=0}^{df_j} \sigma_i(x_1, \cdots, x_{j-1})(x_j - \alpha_j)^i \arrow[r, dashed, "\text{Expansion}"'] & \sum_{i=0}^{df_j} \bar{\sigma}_i(x_1, \cdots, x_{j-1}) x_j^i
\end{tikzcd}
$$

- Algorithm in Monagan and Tuncer (2018): dashed arrows.
  Expression swell occurs at the expansion step.
- Algorithm CMSHL: lined arrows.
  No multivariate polynomial arithmetic.
  No more expression swell.
  More parallelizable.

Let $f_{j-1} = x_1^{df} + \sum_{i=0}^{df-1} \sigma_i(x_2, ..., x_{j-1}) x_1^i$ with $\sigma_i = \sum_{k=1}^{s_i} c_{ik} M_{ik}$
and $g_{j-1} = x_1^{dg} + \sum_{i=0}^{dg-1} \tau_i(x_2, ..., x_{j-1}) x_1^i$ with $\tau_i = \sum_{k=1}^{t_i} d_{ik} N_{ik}$,
where M_{ik}, N_{ik} are the monomials in σ_i, τ_i respectively.

Pick $\boldsymbol{\beta} = (\beta_2, \cdots, \beta_{j-1}) \in (\mathbb{Z}_p \backslash \{0\})^{j-2}$ at random.

Evaluate monomials at $\boldsymbol{\beta}$: .... $\mathcal{O}((j-2)(\#f + \#g + d_{\max}))$
$\mathcal{S} = \{S_i = \{m_{ik} = M_{ik}(\boldsymbol{\beta}), 1 \le k \le s_i\}, 0 \le i \le df - 1\}$ and
$\mathcal{T} = \{T_i = \{n_{ik} = N_{ik}(\boldsymbol{\beta}), 1 \le k \le t_i\}, 0 \le i \le dg - 1\}$.

if any $|S_i| \ne s_i$ **or** any $|T_i| \ne t_i$ **then return** FAIL **end if**

Let s be the maximum of the s_i and t_i.

for k from 1 to s **in parallel do**

    Let $Y_k = (x_2 = \beta_2^k, \cdots, x_{j-1} = \beta_{j-1}^k)$.

    $A_k, F_k, G_k \leftarrow a_j(x_1, Y_k, x_j), f_{j-1}(x_1, Y_k), g_{j-1}(x_1, Y_k)$. .... $\mathcal{O}(s(\#f + \#g + \#a))$

    if $\gcd(F_k, G_k) \ne 1$ **then return** FAIL **end if** // unlucky evaluation

    $f_k, g_k \leftarrow$ BivariateHenselLift$(A_k, F_k, G_k, \alpha_j, p)$. .... $\mathcal{O}(s(d_1^2 d_j + d_1 d_j^2))$

end for

Let $f_k = x_1^{df} + \sum_{l=1}^{\mu} \alpha_{kl} \tilde{M}_l(x_1, x_j)$ for $1 \le k \le s$, where $\mu \le d_1 d_j$.

for l from 1 to μ **in parallel do**

    $i \leftarrow \deg(\tilde{M}_l, x_1)$.

    Solve the $s_i \times s_i$ linear system for c_{lk}: $\{\sum_{k=1}^{s_i} m_{ik}^t c_{lk} = \alpha_{nl}$ for $1 \le t \le s_i\}$ ... $\mathcal{O}(s d_j \#f)$

end for

Construct $f_j \leftarrow x_1^{df} + \sum_{l=1}^{\mu} \left(\sum_{k=1}^{s_i} c_{lk} M_{ik}(x_2, ..., x_{j-1})\right) \tilde{M}_l(x_1, x_j)$.

Similarly, construct g_j. .... $\mathcal{O}(s d_j \#g)$

if $a_j = f_j g_j$ **then return** (f_j, g_j) **else return** FAIL **end if**

Complexity of CMSHL

Theorem (Theorem 4.4.12)

Let $a \in \mathbb{Z}[x_1, \cdots, x_n]$ be monic in x_1. Let f and g be the irreducible factors of a. Let $T_{fg} = \#f + \#g$. Let p be a large prime s.t. p does not divide any term of f and g. Let $\boldsymbol{\alpha} = (\alpha_2, \cdots, \alpha_n) \in \mathbb{Z}^{n-1}$ be a randomly chosen evaluation point from $[1, \tilde{N} - 1]^{n-1}$. Assume $\boldsymbol{\alpha}$ is Hilbertian. Let $d = \deg(a)$, $d_i = \deg(a, x_i)$ for $1 \le i \le n$, and $d_{\max} = \max_{i=2}^{n}(d_i)$. Let $f_j = \sum_{i=0}^{df_j} \sigma_i(x_2, \cdots, x_j)x_1^i$ and $g_j = \sum_{i=0}^{dg_j} \tau_i(x_2, \cdots, x_j)x_1^i$. Let $s_j = \max(\max_i \#\sigma_i, \max_i \#\tau_i)$. Define $s_{\max} = \max_{j=3}^{n}(s_j)$. The failure probability of CMSHL is less than

$$\frac{(n-2)(2d^2(s_{\max}^2 + T_{fg}) + ds_{\max}T_{fg})}{2(p - d + 1)}. \tag{1}$$

Let $f_1 = f(x_1, \boldsymbol{\alpha})$ and $g_1 = g(x_1, \boldsymbol{\alpha})$ be the image polynomials with $\gcd(f_1, g_1) = 1$. If CMSHL succeeds, the number of arithmetic operations in $\mathbb{Z}_p$ for lifting f_1 and g_1 to f_n and g_n in $n-1$ steps (via Algorithm 9) in the worst case is

$$\mathcal{O}(\underbrace{d_1^2 d_2 + d_1 d_2^2}_{\text{first BHL}} + (n-2)\underbrace{(d_{\max}s_{\max}(T_{fg} + d_1^2 + d_1 d_{\max}) + s_{\max}\#a)}_{\text{CMSHL } x_3, x_4, \cdots, x_n}). \tag{2}$$

Algorithm CMBBSHL ($a \in \mathbb{Z}[x_1, \cdots, x_n]$ is monic in x_1 and square-free) [Algorithm 11]

Input: A prime p, $\alpha_j \in \mathbb{Z}$, the modular black box $\mathrm{B} : \mathbb{Z}^n \times \{p\} \to \mathbb{Z}_p$ s.t. $\mathrm{B}(\boldsymbol{\beta}, p) = a(\boldsymbol{\beta}) \bmod p$, $d_i = \deg(a, x_i)$ $(1 \le i \le n)$ (pre-computed), $f_{\rho,j-1} \in \mathbb{Z}_p[x_1, \cdots, x_{j-1}]$ $(1 \le \rho \le r)$ s.t. $a_j(x_j = \alpha_j) = \prod_{\rho=1}^{r} f_{\rho,j-1}$ with $j > 2$.

Output: $f_{\rho,j} \in \mathbb{Z}_p[x_1, \cdots, x_j]$ $(1 \le \rho \le r)$ s.t. (i) $a_j = \prod_{\rho=1}^{r} f_{\rho,j}$, (ii) $f_{\rho,j}(x_j = \alpha_j) = f_{\rho,j-1}$ for $1 \le \rho \le r$; Otherwise, **return** FAIL.

Let $f_{\rho,j-1} = x_1^{df_\rho} + \sum_{i=0}^{df_\rho - 1} \sigma_{\rho,i}(x_2, ..., x_{j-1}) x_1^i$ where $\sigma_{\rho,i} = \sum_{k=1}^{s_{\rho,i}} c_{\rho,ik} M_{\rho,ik}$, $M_{\rho,ik}$ are the monomials in $\sigma_{\rho,i}$ for $1 \le \rho \le r$. $df_\rho = \deg(f_{\rho,j-1}, x_1)$.
Pick $\boldsymbol{\beta} = (\beta_2, \cdots, \beta_{j-1}) \in (\mathbb{Z}_p \backslash \{0\})^{j-2}$ at random.
Evaluate: $\{\mathcal{S}_\rho = \{S_{\rho,i} = \{m_{\rho,ik} = M_{\rho,ik}(\boldsymbol{\beta}), 1 \le k \le s_{\rho,i}\}, 0 \le i \le df_\rho - 1\}, 1 \le \rho \le r\}$.
if any $|S_{\rho,i}| \ne s_{\rho,i}$ **then return** FAIL **end if**
Let s be the maximum of $s_{\rho,i}$.
for k from 1 to s **do**
  Let $Y_k = (x_2 = \beta_2^k, \cdots, x_{j-1} = \beta_{j-1}^k)$.
  $A_k \leftarrow a_j(x_1, Y_k, x_j) \in \mathbb{Z}_p[x_1, x_j]$. . . . . $\mathcal{O}(sd_1 d_j \mathrm{C}(\text{probe B}) + s(d_1^2 d_j + d_1 d_j^2))$
  if $\deg(A_k, x_j) \ne d_j$ **then return** FAIL **end if**
  $F_{\rho,k} \leftarrow f_{\rho,j-1}(x_1, Y_k) \in \mathbb{Z}_p[x_1]$ for $1 \le \rho \le r$. . . . . $\mathcal{O}\left(s\left(\sum_{\rho=1}^{r} \# f_{\rho,j-1}\right)\right)$
  if $\gcd(F_{\rho,k}, F_{\phi,k}) \ne 1$ for any $\rho \ne \phi$ $(1 \le \rho, \phi \le r)$ **then return** FAIL **end if**
  $f_{\rho,k} \leftarrow \mathrm{BivariateHenselLift}(A_k(x_1, x_j), F_{\rho,k}(x_1), \alpha_j, p)$. . . . . $\mathcal{O}(s(d_1 d_j^2 + d_1^2 d_j))$
end for
Let $f_{\rho,k} = x_1^{df_\rho} + \sum_{l=1}^{t_\rho} \alpha_{\rho,kl} \tilde{M}_{\rho,l}(x_1, x_j)$ for $1 \le k \le s$ where $t_\rho \le d_1 d_j$ for $1 \le \rho \le r$.

for ρ from 1 to r **do**
  for l from 1 to t_ρ **do**
    $i \leftarrow \deg(\tilde{M}_{\rho,l}, x_1)$.
    Solve the linear system for $c_{\rho,lk}$: $\left\{\sum_{k=1}^{s_{\rho,i}} m_{\rho,ik}^t c_{\rho,lk} = \alpha_{\rho,tl} \text{ for } 1 \le t \le s_{\rho,i}\right\}$.
  end for .... $\mathcal{O}\left(sd_j\left(\sum_{\rho=1}^{r} \#f_{\rho,j-1}\right)\right)$
  Construct $f_{\rho,j} \leftarrow x_1^{df_\rho} + \sum_{l=1}^{t_\rho}\left(\sum_{k=1}^{s_{\rho,i}} c_{\rho,lk} M_{\rho,ik}(x_2,\ldots,x_{j-1})\right)\tilde{M}_{\rho,l}(x_1,x_j)$.
end for
Pick $\boldsymbol{\beta} = (\beta_2, \cdots, \beta_j) \in \mathbb{Z}_p^{j-1}$ at random.
if $\mathrm{B}(\boldsymbol{\beta}, \alpha_{j+1}, \cdots, \alpha_n) = \prod_{\rho=1}^{r} f_{\rho,j}(\boldsymbol{\beta})$ **then return** $f_{\rho,j}$ $(1 \le \rho \le r)$ **else return** FAIL
end if

Theorem (Theorem 5.3.1)

Let B *be the modular black box representation of* $a \in \mathbb{Z}[x_1, \cdots, x_n]$ *where* a *is square-free and monic in* x_1. *Let* $d_j = \deg(a, x_j)$ *for* $1 \le j \le n$ *and* $d_{\max} = \max_{j=2}^{n} d_j$. *Let* $df_\rho = \deg(f_{\rho,j}, x_1)$ *and let* $f_{\rho,j} = \sum_{i=0}^{df_\rho - 1} \sigma_{\rho,i}(x_2, \cdots, x_j) x_1^i$. *Let* $s_j = \max_\rho(\max_i \#\sigma_{\rho,i})$. *Define* $s_{\max} = \max_{j=3}^{n} s_j$. *The total number of probes to the black box* B *for CMBBSHL is*

$$\sum_{j=2}^{n} s_j(d_1+1)(d_j+1) \in \mathcal{O}(n d_1 d_{\max} s_{\max}).$$

The number of probes to the black box

	Approach I	Kaltofen & Trager	Rubinfeld & Zippel
Zippel's S.I.	# probes # univariate fac.	$\mathcal{O}(n\delta_{\max} d^2 \# f_{\max})$ $\mathcal{O}(n\delta_{\max} \# f_{\max})$	$\mathcal{O}(rn^2 \delta_{\max}^2 d_1 T_{\max})$ $\mathcal{O}(rn^2 \delta_{\max}^2 T_{\max})$
Ben-Or/ Tiwari	# probes # univariate fac.	$\mathcal{O}(d^2 \# f_{\max})$ $\mathcal{O}(\# f_{\max})$	$\mathcal{O}(rn\delta_{\max} d_1 T_{\max})$ $\mathcal{O}(rn\delta_{\max} T_{\max})$

Approach II	CMBBSHL
# probes # univariate fac.	$\mathcal{O}(nd_1 d_{\max} s_{\max})$ 1

Algorithm CMBBSHL requires the least number of probes since $T_{\max} \geq s_{\max}$ and $r\delta_{\max} \geq d_{\max}$.

Benchmark: Computing the factors of det(T_n)

n	10	11	12	13	14	15	16
CMBBSHL	6.299	14.679	43.927	106.838	403.089	1020.001	4876.827
# probes	109,139	267,465	894,358	2,180,399	6,981,462	17,175,949	53,416,615
Maple det	0.306	1.754	8.429	49.080	315.842	$>$ 72gb	N/A
Maple fac	1.91	3.48	23.11	57.75	509.82	7334.50	N/A
Maple tot	2.22	5.23	31.54	106.83	825.66	-	-
Magma det	1.89	5.10	36.12	327.79	2108.42	$>$ 72gb	N/A
Magma fac	1.21	7.58	158.97	583.39	13,640.79	$>$ 72gb	N/A
Magma tot	3.10	12.68	195.09	911.18	15,749.21	-	-

Table: CPU timings in seconds for computing det(T_n) using the fast Vandermonde solver. N/A: Not attempted.

Algorithm CMBBSSHL (non-monic, non-square-free, and non-primitive)

Technicalities for our algorithm design:

1. Non-monic: Use **non-monic** bivariate Hensel lifts (BHL), modified from Monagan and Paluck (2022). Cubic cost: $O(d_1^2 d_j + d_1 d_j^2)$.
2. Non-square-free: Compute the **square-free part** of the bivariate images $a(x_1, \beta^k, x_j)$ with bivariate dense interpolation, gcd computation and division. Cost of gcd: $O(d_1^2 d_j + d_1 d_j^2)$ [Brown (1971)].
3. Non-primitive: Compute the content recursively after computing the factors of the primitive part of a.

Square-free part

Lemma (Lemma 3.4.2)

Let $a \in \mathbb{Z}[x_2, \cdots, x_n][x_1]$ *be non-primitive and let* $h = \mathrm{cont}(a, x_1)$. *Let the irreducible factorization of* a *be*

$$a = h \prod_{i=1}^{r} f_i^{e_i},$$

where $\deg(f_i, x_1) > 0$, $\gcd(f_i, f_j) = 1$ *for* $i \neq j$ *and* f_i *is irreducible in* $\mathbb{Z}[x_1, \cdots, x_n]$.
Let $g = \gcd(a, \partial a / \partial x_1)$. *Then,*
(i) $g = \pm h \prod_{i=1}^{r} f_i^{e_i - 1}$,
(ii) $a/g = \pm \prod_{i=1}^{r} f_i = \pm \mathrm{sqf}(\mathrm{pp}(a))$,
(iii) $\mathrm{cont}(a/g) = \pm 1$.

Definition (Definition 3.4.3)

Let $a \in \mathbb{Z}[x_2, \cdots, x_n][x_1]$ (not necessarily primitive). We define the square-free part of a, denoted as $\mathrm{sqf}(a)$ to be the square-free part of the primitive part of a, i.e. $\mathrm{sqf}(a) = \mathrm{sqf}(\mathrm{pp}(a)) = \prod_{i=1}^{r} f_i$.

How does our algorithm work?

Prior Hensel lifting steps:

1. Choose a large prime p, e.g. $p = 2^{62} - 57$ and a positive integer $\tilde{N} < p$.
2. Choose $\boldsymbol{\alpha} = (\alpha_2, \cdots, \alpha_n) \in \mathbb{Z}^{n-1}$ from $[1, \tilde{N} - 1]^{n-1}$ randomly. W.h.p.
   - $\boldsymbol{\alpha}$ is Hilbertian
   - $\boldsymbol{\alpha}$ satisfies the weak SHL assumption
3. Compute $a(x_1, \boldsymbol{\alpha}) \in \mathbb{Z}[x_1]$ from the black box **B** with Chinese remaindering.
4. Factor $a(x_1, \boldsymbol{\alpha})$ over $\mathbb{Z}$ as follows.
   Let the irreducible factorization of a over $\mathbb{Z}$ be

$$a = h f_1^{e_1} f_2^{e_2} \cdots f_r^{e_r} \in \mathbb{Z}[x_1, \cdots, x_n]$$

where $\deg(f_\rho, x_1) > 0$ $(1 \leq \rho \leq r)$, f_ρ is irreducible over $\mathbb{Z}$ and h is the content of a in x_1. Then, with high probability (w.h.p.),

$$a(x_1, \boldsymbol{\alpha}) = \hat{h} \hat{f}_1^{\,e_1} \hat{f}_2^{\,e_2} \cdots \hat{f}_r^{\,e_r} \in \mathbb{Z}[x_1],$$

where $\boxed{\hat{f}_\rho(x_1, \boldsymbol{\alpha}) = (1/\lambda_\rho) f_\rho(x_1, \boldsymbol{\alpha})}$ and $\lambda_\rho = \mathrm{icont}(f_\rho(x_1, \boldsymbol{\alpha})) \in \mathbb{Z}$ and $\hat{f}_\rho$ is irreducible in $\mathbb{Z}[x_1]$ $(1 \leq \rho \leq r)$.

CMBBSHL: Hensel lifting x_j (non-monic, non-square-free)

[Algorithm 13]

Input: The modular black box $\mathrm{B} : \mathbb{Z}^n \times \{p\} \to \mathbb{Z}_p$ s.t. $\mathrm{B}(\boldsymbol{\beta}, p) = a(\boldsymbol{\beta}) \bmod p$, $(\hat{f}_{\rho,j-1}, 1 \le \rho \le r) \in \mathbb{Z}_p[x_1, \cdots, x_{j-1}]^r$, $\boldsymbol{\alpha} \in \mathbb{Z}^{n-1}$, a prime p, $d_i = \deg(a, x_i)$ for $1 \le i \le n$ (pre-computed), $X = [x_1, \cdots, x_n]$, $j \in \mathbb{Z}$ s.t. $\mathrm{sqf}(a_j(x_j = \alpha_j)) = \prod_{\rho=1}^{r} \lambda_\rho \prod_{\rho=1}^{r} \hat{f}_{\rho,j-1}$.

Output: $(\hat{f}_{\rho,j}, 1 \le \rho \le r) \in \mathbb{Z}_p[x_1, \cdots, x_j]^r$ s.t. (i) $\mathrm{sqf}(a_j) = \prod_{\rho=1}^{r} \lambda_\rho \prod_{\rho=1}^{r} \hat{f}_{\rho,j}$, (ii) $\hat{f}_{\rho,j}(x_j = \alpha_j) = \hat{f}_{\rho,j-1}$ for all $1 \le \rho \le r$; Otherwise, **return** FAIL.

Let $\hat{f}_{\rho,j-1} = \sum_{i=0}^{df_\rho} \sigma_{\rho,i}(x_2, \ldots, x_{j-1}) x_1^i$ $(1 \le \rho \le r)$ where $\sigma_{\rho,i} = \sum_{k=1}^{s_{\rho,i}} c_{\rho,ik} M_{\rho,ik}$.
Pick $\boldsymbol{\beta} = (\beta_2, \cdots, \beta_{j-1}) \in (\mathbb{Z}_p \backslash \{0\})^{j-2}$ at random.
Evaluate (for $1 \le \rho \le r$): $\mathcal{S}_\rho = \{\mathcal{S}_{\rho,i} = \{m_{\rho,ik} = M_{\rho,ik}(\boldsymbol{\beta}), 1 \le k \le s_{\rho,i}\}, 0 \le i \le df_\rho\}$.
if any $|\mathcal{S}_{\rho,i}| \ne s_{\rho,i}$ **then return** FAIL **end if** // monomial evals must be distinct
Let s be the maximum of $s_{\rho,i}$. // Compute s images of the factors in $\mathbb{Z}_p[x_1, x_j]$:
for k from 1 to s **do**
Let $Y_k = (x_2 = \beta_2^k, \cdots, x_{j-1} = \beta_{j-1}^k)$.
$A_k \leftarrow a_j(x_1, Y_k, x_j) \in \mathbb{Z}_p[x_1, x_j]$. .... $\mathcal{O}(sd_1 d_j \mathrm{C}(\text{probe B})) + \mathcal{O}(s(d_1^2 d_j + d_1 d_j^2))$
if $\deg(A_k, x_1) \ne d_1$ **or** $\deg(A_k, x_j) \ne d_j$ **then return** FAIL **end if**
$g_k \leftarrow \gcd(A_k, \frac{\partial A_k}{\partial x_1}) \bmod p \in \mathbb{Z}_p[x_1, x_j]$. .... $\mathcal{O}(s(d_1^2 d_j + d_1 d_j^2))$
if $\deg(g_k, x_1) \ne d_1 - \sum_{\rho=1}^{r} df_\rho$ **then return** FAIL **end if**
$A_{sf} \leftarrow \mathrm{quo}\ (A_k, g_k) \bmod p$. // $A_{sf} = \mathrm{sqf}(A_k) \bmod p$, up to a constant in $\mathbb{Z}_p$.
$A_{sfm} \leftarrow A_{sf} / (\mathrm{LC}(\mathrm{LC}(A_{sf}, x_1), x_j)) \bmod p$. // make $\mathrm{LC}(A_{sf}, x_1)$ monic in x_j.
$F_{\rho,k} \leftarrow \hat{f}_{\rho,j-1}(x_1, Y_k) \in \mathbb{Z}_p[x_1]$ for $1 \le \rho \le r$. .... $\mathcal{O}(s(\sum_{\rho=1}^{r} \#\hat{f}_{\rho,j-1}))$
if any $\deg(F_{\rho,k}) < df_\rho$ (for $1 \le \rho \le r$) **then return** FAIL **end if**

if $\gcd(F_{\rho,k}, F_{\phi,k}) \neq 1$ for any $1 \leq \rho < \phi \leq r$ **then return** FAIL **end if**

$\hat{f}_{\rho,k} \leftarrow \text{BivariateHenselLift}(A_{sfm}(x_1, x_j), F_{\rho,k}(x_1), \alpha_j, p)$. .... $\mathcal{O}(s(\tilde{d}_1 \tilde{d}_j^2 + \tilde{d}_1^2 \tilde{d}_j))$

end for

Let $\hat{f}_{\rho,k} = \sum_{l=1}^{t_\rho} \alpha_{\rho,kl} \tilde{M}_{\rho,l}(x_1, x_j) \in \mathbb{Z}_p[x_1, x_j]$ for $1 \leq k \leq s$, for $1 \leq \rho \leq r$ $(t_\rho = \#\hat{f}_{\rho,k})$.

for ρ from 1 to r **do**

  for l from 1 to t_ρ **do**

    $i \leftarrow \deg(\tilde{M}_{\rho,l}, x_1)$.

    Solve the linear system for $c_{\rho,lk}$: $\left\{ \sum_{k=1}^{s_{\rho,i}} m_{\rho,ik}^t c_{\rho,lk} = \alpha_{\rho,tl} \text{ for } 1 \leq t \leq s_{\rho,i} \right\}$.

  end for .... $\mathcal{O}(s\tilde{d}_j(\sum_{\rho=1}^{r} \#\hat{f}_{\rho,j-1}))$

  Construct $\hat{f}_{\rho,j} \leftarrow \sum_{l=1}^{t_\rho} \left(\sum_{k=1}^{s_{\rho,i}} c_{\rho,lk} M_{\rho,ik}(x_2, \ldots, x_{j-1}) \right) \tilde{M}_{\rho,l}(x_1, x_j)$.

end for

Pick $\boldsymbol{\beta} = (\beta_2, \cdots, \beta_j) \in \mathbb{Z}_p^{j-1}$ at random until $\deg(\hat{f}_{\rho,j}(x_1, \boldsymbol{\beta})) = df_\rho$ for all $1 \leq \rho \leq r$.

$A_{\boldsymbol{\beta}} \leftarrow a_j(x_1, \boldsymbol{\beta}) \bmod p$ via probes to **B** and Lagrange interpolation.

if $\hat{f}_{\rho,j}(x_1, \boldsymbol{\beta}) \mid A_{\boldsymbol{\beta}}$ for all $1 \leq \rho \leq r$ **then return** $(\hat{f}_{\rho,j}, 1 \leq \rho \leq r)$ **else return** FAIL **end if**

Example

Example

Consider $a = f_1 f_2 \in \mathbb{Z}[x_1, \cdots, x_4]$ where

$$f_1 = (2x_2^2x_3^3 + 4)x_1^8 + (4x_2^2x_3^3 + 22x_2^2x_4^3 + 1452x_2^2x_4)x_1 + x_2^2x_3x_4 - 4x_3,$$
$$f_2 = (3x_2 + 39x_4 + 3x_3)x_1^8 + (5x_2x_3^2x_4 + 33x_2x_3x_4^2)x_1^2 - 363x_4^2 + 44.$$

In this case, $h = 1$ (a has no content in x_1, neither integer content) and $\mathrm{sqf}(a) = a$. Let $\boldsymbol{\alpha} = (2, 3, 9)$ and $p = 2^{31} - 1$,

$$\begin{aligned} a(x_1, \boldsymbol{\alpha}) &= 80520x_1^{16} + 3706560x_1^{10} + \cdots - 3430775304x_1 - 2818464 \\ &= \underbrace{4}_{\lambda_1} \underbrace{(55x_1^8 + 29214x_1 + 24)}_{\hat{f}_1} \underbrace{(366x_1^8 + 16848x_1^2 - 29359)}_{\hat{f}_2} \\ &= f_1(x_1, \boldsymbol{\alpha}) f_2(x_1, \boldsymbol{\alpha}). \end{aligned}$$

Example ctd..

After the first Hensel lifting step (a bivariate Hensel lift only),

$$\hat{f}_{1,2} = (1073741837x_2^2 + 1)x_1^8 + 1073749127x_2^2x_1 + 1610612742x_2^2 + 2147483644,$$
$$\hat{f}_{2,2} = (3x_2 + 360)x_1^8 + 8424x_2x_1^2 + 2147454288.$$

The second Hensel lifting step recovers x_3, and we get

$$\begin{aligned}\hat{f}_{1,3} &= (1073741824x_2^2x_3^3 + 1)x_1^8 + (x_2^2x_3^3 + 1073749100x_2^2)x_1 + 536870914x_2^2x_3 \\ &\quad + 2147483646x_3, \\ \hat{f}_{2,3} &= (3x_2 + 3x_3 + 351)x_1^8 + (45x_2x_3^2 + 2673x_2x_3)x_1^2 + 2147454288.\end{aligned}$$

The last Hensel lifting step outputs $\hat{f}_{\rho,4}$ ($\rho = 1, 2$) s.t.

$\boxed{a_4 = \mathrm{sqf}(a_4) = (\lambda_1\lambda_2)\hat{f}_{1,4}\hat{f}_{2,4} \bmod p}$ with

$$\begin{aligned}\hat{f}_{1,4} &= (1073741824x_2^2x_3^3 + 1)x_1^8 + (x_2^2x_3^3 + 1073741829x_2^2x_4^3 + 363x_2^2x_4)x_1 \\ &\quad + 536870912x_2^2x_3x_4 + 2147483646x_3 \\ \hat{f}_{2,4} &= (3x_2 + 39x_4 + 3x_3)x_1^8 + (5x_2x_3^2x_4 + 33x_2x_3x_4^2)x_1^2 + 2147483284x_4^2 + 44.\end{aligned}$$

Example ctd..

Rational number reconstruction in Maple:

```
> ff[1] := iratrecon(fhat[1,4], p);
```

$$\mathrm{ff}_1 := \frac{1}{2}x_2^2x_1^8x_3^3 + x_1^8 + x_1x_2^2x_3^3 + \frac{11}{2}x_2^2x_1x_4^3 + 363x_2^2x_1x_4 + \frac{1}{4}x_2^2x_3x_4 - x_3$$

λ_1 is the least common multiple of the denominators of coefficients of ff_1.
Multiply ff_1 by $\lambda_1 = 4$, we get the true factor $f_1 \in \mathbb{Z}[x_1, \cdots, x_n]$:

```
> f[1] := numer(ff[1]);
```

$$\mathrm{f}_1 := 2x_2^2x_1^8x_3^3 + 4x_1^8 + 4x_1x_2^2x_3^3 + 22x_2^2x_1x_4^3 + 1452x_2^2x_1x_4 + x_2^2x_3x_4 - 4x_3$$

A Hybrid Maple + C Implementation

Table: CPU timings (in seconds) for computing the factors of $\det(B_n)$.

n	5	6	7	8	9
$N = 2n$	10	12	14	16	18
$\#f_1, \#f_2$	12,7	32,32	56,30	167,167	153,294
$\#f_3, \#f_4$	12,7	32,32	56,30	167,167	253,294
$\#\det(B_n)$	701	5162	79740	1716810	7490224
CMBBSHL tot	0.323	0.999	3.320	17.542	34.150
probes tot	1944	6156	18936	84240	143775
Maple det	0.455	7.880	382.80	$>$ 64 gigs	N/A
Maple fac	0.109	0.326	1.270	42.15	139.80
Maple tot	0.564	8.206	384.07	-	-
Magma det	1.680	6.290	594.60	$>$ 3h	N/A
Magma fac	0.120	0.480	33.140	N/A	N/A
Magma tot	1.800	6.770	627.74	N/A	N/A

N/A: Not attempted.

Benchmark 2: Vandermonde matrices

Table: CPU timings (in seconds) for computing the factors of $\det(V_n)$.

$n = N$	7	8	9	10	11	12	13
$r = \binom{n}{2}$	21	28	36	45	55	66	78
$\#\det(V_n)$	5040	40320	362880	3628800	39916800	O/M*	N/A
CMBBSHL tot	0.336	0.649	1.137	1.990	3.290	5.190	8.175
probes tot	1328	2256	3597	5467	7975	11263	15479
$\mathrm{pp}(a)$ fac only	0.097	0.130	0.175	0.262	0.331	0.401	0.513
Maple det	0.061	0.100	0.446	5.700	45.07	$>$ 64 gigs	N/A
Maple minor	0.009	0.036	0.297	5.391	35.518	$>$ 64 gigs	N/A
Maple fac	0.012	0.068	0.882	17.96	523.80	N/A	N/A
Maple tot	0.021	0.104	1.179	23.351	559.318	N/A	N/A

N/A: Not attempted. O/M*: Out of memory at expanding the factors in Maple.

Benchmark 3: Dixon matrices

Table: Timings (in seconds) for computing the determinant of Dixon matrices.

	heron3d	heron4d	robotarms (b_1)	robotarms (t_1)	heron5d
n	7	11	8	8	16
$N \times N$	13×13	63×63	16×16	16×16	399×399
$d_j = \deg(a, x_j)$ ($1 \le j \le n$)	19,12,12, 8,8,8,4	89,26,26, 12,12,12 8,8,8,8,8	16,64,128,44 36,16,16,16	16,64,32,56, 32,128,16,16	2159,328,328,144, 144,144,64,64, 64,64,32,32, 32,32,32,16
r	6	4	8	7	8
$\#f_i$ ($1 \le i \le r$)	3,23,3, 3,1,3	22,1, 6,131	1,1,2,39, 2,1,4,7	2,1,30,6, 2,7,7	823,130,22,3 3,3,3,1
e_i ($1 \le i \le r$)	1,2,1, 1,7,1	2,37, 7,4	8,24,48,8 24,4,4,4	48,16,8,8, 16,4,4	8,8,20,46 46,46,1831
$\#\det(A)$	525	37666243	O/M*	O/M*	N/A
$\max \lambda_\rho$	1	1	1	2	1
CMBBSHL tot	0.685	43.809	169.851	350.809	165208.747
probes tot	5701	201183	99652	131250	36008392
$\mathrm{pp}(a)$ fac	0.683	43.804	18.972	43.178	165106.278
probes $\mathrm{pp}(a)$	5699	201181	11448	16626	-
Maple det	0.614	O/M	N/A	N/A	N/A
Maple minor	0.006	0.383	O/M	O/M	N/A
Maple fac	0.084	O/M	N/A	N/A	N/A
Maple tot	0.620	-	-	-	-

N/A: Not attempted. O/M: Out of memory.

O/M*: Out of memory when expanding the factors in Maple.

Failure Probability

Proposition (Proposition 6.4.3)

Let p be a 63-bit prime, i.e. $p \in (2^{62}, 2^{63})$. Let $\mathbb{P}_{63} = \{$all 63-bit primes$\}$. Let $f_\rho = \sum_{i=1}^{\#f_\rho} c_{\rho,i} \cdot x_1^{e_{i_1}} \cdots x_n^{e_{i_n}}$ for $1 \le \rho \le r$, where $c_{\rho,i} \neq 0$, $c_{\rho,i} \in \mathbb{Z}$, and $(e_{i_1}, \cdots, e_{i_n}) \in \mathbb{N}^n$. Let $\chi_\rho = \{i \in \mathbb{Z} \mid |c_{\rho,i}| \ge p\}$ and let $\#f_{\rho,p} = |\chi_\rho|$ for $1 \le \rho \le r$. Let $h_\rho = \|f_\rho\|_\infty$ for $1 \le \rho \le r$. Let $h_{\max} = \max_{\rho=1}^{r} h_\rho$. Then,

$$\Pr[p \mid \textit{at least one } c_{\rho,i} \textit{ in any } f_\rho] \le \frac{1}{|\mathbb{P}_{63}|} \left\lfloor \frac{\log_2(h_{\max})}{62} \right\rfloor \sum_{\rho=1}^{r} \#f_{\rho,p}. \tag{3}$$

Complexity

Theorem (Theorem 6.4.4)

Let p be a large prime and $\tilde{N} < p$, $\tilde{N} \in \mathbb{Z}^+$. Let $a \in \mathbb{Z}[x_1, \cdots, x_n]$ and $\boldsymbol{\alpha} = (\alpha_2, \cdots, \alpha_n) \in \mathbb{Z}_p^{n-1}$ be a randomly chosen evaluation point from $[1, \tilde{N}]^{n-1}$. Suppose $\boldsymbol{\alpha}$ is Hilbertian. Then, if algorithm CMBBSHL returns an answer that is not FAIL, the total number of arithmetic operations in $\mathbb{Z}_p$ in the worst case for lifting $\hat{f}_{\rho,1}$ to $\hat{f}_{\rho,n}$ using Algorithm 13 $n-1$ times is

$$O\left((n-2)s_{\max}d_{\max}\left(\sum_{\rho=1}^{r}\#\hat{f}_{\rho,j-1} + d_1^2 + d_1 d_{\max} + d_1 C(\text{probe } \boldsymbol{B})\right)\right). \tag{4}$$

where $d_1 = \deg(a, x_1)$, $d_{\max} = \max_{j=2}^{n}(\deg(a, x_j))$, and C(probe $\boldsymbol{B}$) is the number of arithmetic operations in $\mathbb{Z}_p$ for one probe to the black box $\boldsymbol{B}$. The total number of probes to the black box is $O(nd_1 d_{\max} s_{\max})$.

Conclusion

- We designed and implemented a new black box algorithm for factoring polynomials $a \in \mathbb{Z}[x_1, \cdots, x_n]$.
- We did a worst-case complexity analysis with bounds on failure probabilities of algorithm CMBBSHL.
- We tested our algorithm on a variety of artificial and real factorization problems.

Thank you for attending!

Thanks to my supervisor, Prof. Monagan, and the exam committee.

References

M. Ben-Or and P. Tiwari. A deterministic algorithm for sparse multivariate polynomial interpolation. In Proceedings of STOC '88, pp. 301–309. ACM (1988)

Chen, T., Monagan, M.: The complexity and parallel implementation of two sparse multivariate Hensel lifting algorithms for polynomial factorization. In Proceedings of CASC 2020, LNCS **12291**: 150–169. Springer (2020)

Chen, T., Monagan, M.: Factoring multivariate polynomials represented by black boxes: A Maple + C Implementation. *Math. Comput. Sci.* **16**,18 (2022)

Chen, T., Monagan, M.: A new black box factorization algorithm - the non-monic case. In Proceedings of ISSAC 2023, pp. 173–181. ACM (2023)

A. Diaz and E. Kaltofen. FOXBOX: a system for manipulating symbolic objects in black box representation. Proceedings of ISSAC 1998. ACM (1998)

Von zur Gathen, J., Gerhard, J.: *Modern Computer Algebra*. Cambridge University Press (2013)

Q-L. Huang and X-S Gao. New sparse multivariate polynomial factorization algorithms over integers. In Proceedings of ISSAC 2023. ACM (2023)

E. Kaltofen. Sparse Hensel lifting. In Proceedings of EUROCAL '85, LNCS **204**, 4–17. Springer (1985)

E. Kaltofen and B. M. Trager. Computing with polynomials given by black boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. *J. Symb. Cmpt.* **9**(3), 301–320. Elsevier (1990)

- G. Lecerf. Improved dense multivariate polynomial factorization algorithms. *J. Symbolic Computation* **42**:477–494 (2007)
- Lee, W.S.: Early termination strategies in sparse interpolation algorithms. Ph.D. Thesis (2001)
- M. Monagan and B. Tuncer. Using Sparse Interpolation in Hensel Lifting. In Proceedings of CASC 2016, LNCS **9890**, 381–400. Springer (2016)
- M. Monagan and B. Tuncer. Sparse multivariate Hensel lifting: a high-performance design and implementation. In Proceedings of ICMS 2018, LNCS **10931**, 359–368. Springer (2018)
- Monagan, M., Tuncer, B.: Sparse multivariate Hensel lifting: a high-performance design and implementation. In Proceedings of ICMS 2018, LNCS **10931**, 359–368. Springer (2018)
- Monagan, M., Tuncer, B.: The complexity of sparse Hensel lifting and sparse polynomial factorization. *J. Symb. Cmpt.* **99**, 189–230. Elsevier (2020)
- R. Rubinfeld and R. E. Zippel. A new modular interpolation algorithm for factoring multivariate polynomials. In Proceedings of Algorithmic Number Theory, First International Symposium, ANTS-I (1994)
- Schwartz, J.: Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, **27**(4), 701–717 (1980)
- Wang, P.S., Rothschild, L.P.: Factoring multivariate polynomials over the integers. *Math. Comp.* **29**, 935–950 (1975)
- Wang, P.S.: An improved multivariate polynomial factoring algorithm. *Math. Comp.* **32**, 1215–1231 (1978)
- Yun, D.Y.Y.: The Hensel Lemma in algebraic manipulation. Ph.D. Thesis (1974)

H. Zassenhaus. On Hensel factorization I. *J. Number Theory*, **1**(3), 291–311 (1969)

Zippel, R.E.: Probabilistic algorithms for sparse polynomials. LNCS **72**, 216–226 (1979)

Zippel, R.E.: Newton's iteration and the sparse Hensel algorithm. In Proceedings of the ACM Symposium on Symbolic Algebraic Computation, pp. 68–72 (1981)

Zippel, R.E.: Interpolating polynomials from their values. *J. Symb. Cmpt.* **9**(3), 375–403 (1990)

Future work

- Solve parametric linear systems of equations.
  Let A be a matrix of size $m \times m$ with $A_{ij} \in \mathbb{Z}[y_1, \cdots, y_n]$.
  Let b be a vector of size $m \times 1$ with $b_i \in \mathbb{Z}[y_1, \cdots, y_n]$.
  Let $A^{(i)}$ be formed from A by replacing the i^{th} column of A with b.
  We can solve $Ax = b$ by computing $x_i = \det(A^{(i)})/\det(A)$ for $1 \leq i \leq m$ in factored form using a black box factorization algorithm.
- Parallelization?

Factoring the content recursively

Let $\mathcal{C}_n = a \in \mathbb{Z}[x_1, \cdots, x_n]$. We have $\mathcal{C}_n = \mathrm{cont}(\mathcal{C}_n) \cdot \mathrm{pp}(\mathcal{C}_n)$.
$\boxed{\mathrm{pp}(\mathcal{C}_n) = \prod_{\rho=1}^{r} f_\rho^{e_\rho}}$. f_ρ's are irreducible over $\mathbb{Z}$ and primitive in x_1.
Let $\mathcal{C}_{n-1} = \mathrm{cont}(\mathcal{C}_n) \in \mathbb{Z}[x_2, \cdots, x_n]$.

- After factoring $\mathrm{pp}(\mathcal{C}_n)$, we create a black box $\mathsf{F}_n : \mathbb{Z}^n \times \{p\} \to \mathbb{Z}_p$ s.t. $\mathsf{F}_n(\boldsymbol{\alpha}, p) = \mathrm{pp}(\mathcal{C}_n)(\boldsymbol{\alpha}) \bmod p$ by computing

$$\mathrm{pp}(\mathcal{C}_n)(\boldsymbol{\alpha}) \bmod p = f_1(\boldsymbol{\alpha})^{e_1} \cdots f_r(\boldsymbol{\alpha})^{e_r} \bmod p.$$

- Next, we create another black box $\mathsf{C}_{n-1} : \mathbb{Z}^{n-1} \times \{p\} \to \mathbb{Z}_p$ for the content $\mathcal{C}_{n-1} = \mathrm{cont}(\mathcal{C}_n)$ s.t.

$$\mathsf{C}_{n-1}(\boldsymbol{\beta}, p) = \frac{\mathsf{C}_n([\gamma, \boldsymbol{\beta}], p)}{\mathsf{F}_n([\gamma, \boldsymbol{\beta}], p)} \bmod p$$

  for a fixed $\gamma \in \mathbb{Z}$. If $\mathsf{F}_n([\gamma, \boldsymbol{\beta}], p) = 0$ then FAIL is returned.
- Then the content is factored recursively. $\mathcal{C}_0 \in \mathbb{Z}$.

Computing the content recursively

```
MakeCont := proc( C::procedure, F::procedure, gamma::integer,
p::prime )
    proc( alpha::Array, p::prime )
        local na := numelems(alpha), alphaNew, g;
        alphaNew := Array(1..na+1);
        alphaNew[1] := gamma;
        for i to na do alphaNew[i+1] := alpha[i]; od;
        g := F( alphaNew, p );
        if g = 0 then return FAIL; fi;
        C( alphaNew, p )/g mod p;
    end;
end;
gamma0 := rand(p)();
BBC := MakeCont( Cn, Fn, gamma0, p );
```

CMBBSHL: non-monic, non-square-free

[Algorithm 12]

Input: The modular black box $\mathrm{B} : \mathbb{Z}^n \times \{p\} \to \mathbb{Z}_p$ s.t. $\mathrm{B}(\boldsymbol{\beta}, p) = a(\boldsymbol{\beta})$ mod p, $(\hat{f}_{\rho,1}, 1 \le \rho \le r) \in \mathbb{Z}_p[x_1]^r$, $\boldsymbol{\alpha} \in \mathbb{Z}^{n-1}$, a prime p, $d_i = \deg(a, x_i)$ for $1 \le i \le n$ (pre-computed), $X = [x_1, \cdots, x_n]$, $n \in \mathbb{Z}$ (the recursive variable) s.t. conditions (i)-(iii) of the input are satisfied.

Output: $(\hat{f}_{\rho,n}, 1 \le \rho \le r) \in \mathbb{Z}_p[x_1, \cdots, x_n]^r$ s.t. conditions (i)-(iii) of the output are satisfied. Otherwise, **return** FAIL.

if $n = 2$ **then**
  $A_k \leftarrow a_2(x_1, x_2) \in \mathbb{Z}_p[x_1, x_2]$. .... $\mathcal{O}(d_1 d_2 \mathrm{C}(\text{probe B})) + \mathcal{O}(d_1^2 d_2 + d_1 d_2^2)$
  if $\deg(A_k, x_1) \ne d_1$ **or** $\deg(A_k, x_2) \ne d_2$ **then return** FAIL **end if**
  $g_k \leftarrow \gcd(A_k, \frac{\partial A_k}{\partial x_1})$ mod $p \in \mathbb{Z}_p[x_1, x_2]$. .... $\mathcal{O}(d_1^2 d_2 + d_1 d_2^2)$
  if $\deg(g_k, x_1) \ne d_1 - \sum_{\rho=1}^{r} \deg(\hat{f}_{\rho,1}, x_1)$ **then return** FAIL **end if**
  $A_{sf} \leftarrow \mathrm{quo}(A_k, g_k)$ mod p. // $A_{sf} = \mathrm{sqf}(A_k)$ mod p, up to a constant in $\mathbb{Z}_p$.
  $A_{sfm} \leftarrow A_{sf} / (\mathrm{LC}(\mathrm{LC}(A_{sf}, x_1), x_2))$ mod p. // make $\mathrm{LC}(A_{sf}, x_1)$ monic in x_2.
  return BivariateHenselLift$(A_{sfm}, (\hat{f}_{\rho,1}, 1 \le \rho \le r), \alpha_2, p)$ .... $\mathcal{O}(d_1^2 d_2 + d_1 d_2^2)$
end if
$(\hat{f}_{\rho,n-1}, 1 \le \rho \le r) \leftarrow$ CMBBSHL$(\mathrm{B}, (\hat{f}_{\rho,1}, 1 \le \rho \le r), \boldsymbol{\alpha}, p, d_i, X, n-1)$.
return CMBBSHLstepj$(\mathrm{B}, (\hat{f}_{\rho,n-1}, 1 \le \rho \le r), \boldsymbol{\alpha}, p, d_i, X, n)$

Large Vandermonde matrices

Table: CPU timings (in seconds) for computing the factors of $\det(V_n)$ for larger n.

$n = N$	15	20	25	30	35	40
$r = \binom{n}{2}$	105	190	300	435	595	780
CMBBSHL tot	18.625	109.996	440.17	1376.793	3560.706	9057.977
probes tot	27311	85622	207912	429752	793809	1350786
$\mathrm{pp}(a)$ fac	0.791	2.246	5.891	13.968	29.597	57.745
H.L. x_n	0.055	0.117	0.256	0.467	0.800	1.487
probes x_n	465	820	1275	1830	2485	3240
s (H.L. x_n)	1	1	1	1	1	1
BB tot	0.024	0.070	0.156	0.353	0.650	1.224
BB eval	0.015	0.039	0.090	0.208	0.368	0.709
BB det	0.009	0.031	0.066	0.145	0.282	0.515
Interp2var	0.001	0.002	0.004	0.009	0.015	0.027
Eval $\hat{f}_{\rho_j-1}$	0.002	0.002	0.003	0.004	0.004	0.005
BHL	0.004	0.005	0.008	0.009	0.010	0.011
VSolve	0.004	0.003	0.006	0.009	0.007	0.012

Breakdown of timings for Dixon matrices

Table: Breakdown of timings (in seconds) for computing the determinant of Dixon Matrices.

	heron3d	heron4d	robotarms (b_1)	robotarms (t_1)	heron5d
n	7	11	8	8	16
$N \times N$	13×13	63×63	16×16	16×16	399×399
H.L. x_n tot	0.146	10.431	1.451	2.958	14347.878
probes x_n	930	41112	901	1173	-
s	13	85	3	3	571
BB tot	0.039	9.303	1.398	2.910	13771.116
BB eval	0.022	4.764	1.369	2.888	7254.237
BB det	0.008	3.720	0.029	0.022	6414.483
Interp2var	0.000	0.061	0.003	0.003	17.574
Eval $\hat{f}_{\rho,j-1}$	0.029	0.029	0.000	0.000	0.576
BHL	0.029	0.160	0.000	0.000	450.033
VSolve	0.001	0.002	0.000	0.002	0.031