Generic bivariate multi-point evaluation, interpolation and modular composition with precomputation

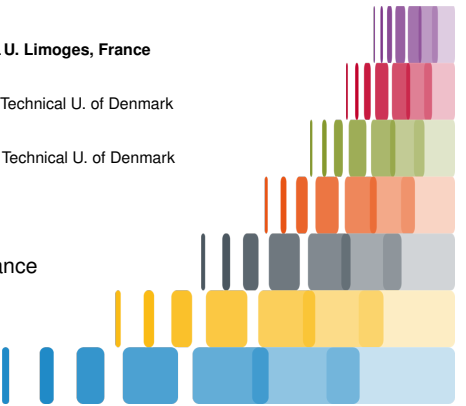
Vincent Neiger ..... **U. Limoges, France**

Johan Rosenkilde ..... Technical U. of Denmark

Grigory Solomatov ..... Technical U. of Denmark

Seminar PolSys - LIP6, Sorbonne Université, France

November 6, 2020



Outline

- Problem and result
- Approach: degree reshaping
- Application to composition, evaluation, interpolation

Outline

- Problem and result
- Approach: degree reshaping
- Application to composition, evaluation, interpolation

Computing with univariate polynomials

$p, f, a, b, \ldots \in \mathbb{K}[x]$, degree n, for some base field $\mathbb{K}$

quasi-linear algorithms are known for most basic operations

[v.z.Gathen - Gerhard, Modern Computer Algebra, 3rd ed. 2013]

- addition $f + g$, multiplication $f * g$
- division with remainder $f = ag + b$
- truncated inverse $f^{-1} \bmod x^n$
- extended GCD $af + bg = \gcd(f, g)$
- multipoint evaluation $f \mapsto f(x_1), \ldots, f(x_n)$
- interpolation $f(x_1), \ldots, f(x_n) \mapsto f$

Modular composition $b = p(a) \bmod f$

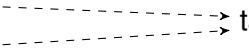
Notable exception: best known exponent $\frac{\omega+1}{2}$ [Brent - Kung 1978]

Work in progress: exponent $\approx \frac{\omega+2}{3}$, randomized [Neiger - Salvy - Schost - Villard]

Efficient? Fast?

Measuring efficiency/speed:

- low complexity bound
- low execution time
- low memory usage, low power consumption, ...



Complexity = algebraic cost (count number of operations in $\mathbb{K}$)

- standard model for algebraic computations
- accurate for finite fields $\mathbb{K} = \mathbb{F}_p$ (e.g. coding theory, crypto, ...)
- ignores coefficient growth (e.g. when working over $\mathbb{K} = \mathbb{Q}$)

Quasi-linear: $\tilde{O}(n)$ operations in $\mathbb{K}$

the holy grail of computer algebraists... or the start of the fight with logarithmic factors

Computing with bivariate polynomials

$p, q, r, \ldots \in \mathbb{K}[x, y]$, bi-degree (d_x, d_y) with $d_x d_y = n$.

quasi-linear algorithms are known for some basic operations,

generally with specific input/output requirements (sparsity, genericity, ...)

- addition $f + g$, multiplication $f * g$ (Kronecker substitution / evaluation-interpolation)
- division with remainder / reduction modulo an ideal
  - w.r.t. autoreduced family [v.d.Hoeven 2015] — sparse
  - modulo two polynomials [v.d.Hoeven - Larrieu 2019] — generic
- multiplication modulo a triangular set
  - using homotopy [Bostan - Chowdhury - v.d.Hoeven - Schost 2011] — sparse
  - reduction to modular composition [Poteaux - Schost 2013]
- multipoint evaluation & interpolation $p \longleftrightarrow p(x_1, y_1), \ldots, p(x_n, y_n)$
  - interpolation [Prony 1795] [Ben-or - Tiwari 1988] — sparse
  - eval./interp.: points&monomials on a grid [Pan 1994] [v.d.Hoeven - Schost 2013] — grid
  - evaluation: reduction to modular composition [Nüsken - Ziegler 2004]

Evaluation and composition

Bivariate multipoint evaluation

Input: $\{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathbb{K}^2$
$p(x, y)$ of degree $< (d_x, d_y)$ with $d_x d_y \in O(n)$
Output: $z_1 = p(x_1, y_1), \ldots, z_n = p(x_n, y_n)$

Bivariate modular composition

Input: $f(x), a(x)$ with $\deg(a) < \deg(f) = n$
$p(x, y)$ of degree $< (d_x, d_y)$ with $d_x d_y \in O(n)$
Output: $b(x) = p(x, a(x)) \bmod f(x)$

[Brent - Kung 1978] $\tilde{O}(n^{\frac{\omega+1}{2}})$ for $d_x = 1$ [Nüsken - Ziegler 2004] $\tilde{O}(d_x d_y^{\frac{\omega+1}{2}})$

Evaluation and composition

Bivariate multipoint evaluation

Input: $\{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathbb{K}^2$ with $x_1, \ldots, x_n$ distinct
$p(x, y)$ of degree $< (d_x, d_y)$ with $d_x d_y \in O(n)$
Output: $z_1 = p(x_1, y_1), \ldots, z_n = p(x_n, y_n)$

Ideal $\mathcal{I} = \langle f(x), y - a(x) \rangle$

Reduction: [Nüsken - Ziegler 2004]
· $f(x) = (x - x_1) \cdots (x - x_n)$
· $a(x)$ the interpolant $a(x_i) = y_i$

Bivariate modular composition

Input: $f(x), a(x)$ with $\deg(a) < \deg(f) = n$
$p(x, y)$ of degree $< (d_x, d_y)$ with $d_x d_y \in O(n)$
Output: $b(x) = p(x, a(x)) \bmod f(x)$

[Brent - Kung 1978] $\tilde{O}(n^{\frac{\omega+1}{2}})$ for $d_x = 1$ [Nüsken - Ziegler 2004] $\tilde{O}(d_x d_y^{\frac{\omega+1}{2}})$

Evaluation and composition

Bivariate multipoint evaluation

PreInput: $\{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathbb{K}^2$ with $x_1, \ldots, x_n$ distinct
Input: $p(x, y)$ of degree $< (d_x, d_y)$ with $d_x d_y \in O(n)$
Output: $z_1 = p(x_1, y_1), \ldots, z_n = p(x_n, y_n)$

Ideal $\mathcal{I} = \langle f(x), y - a(x) \rangle$

Reduction: [Nüsken - Ziegler 2004]
· $f(x) = (x - x_1) \cdots (x - x_n)$
· $a(x)$ the interpolant $a(x_i) = y_i$

Bivariate modular composition

PreInput: $f(x), a(x)$ with $\deg(a) < \deg(f) = n$
Input: $p(x, y)$ of degree $< (d_x, d_y)$ with $d_x d_y \in O(n)$
Output: $b(x) = p(x, a(x)) \bmod f(x)$

[Brent - Kung 1978] $\tilde{O}(n^{\frac{\omega+1}{2}})$ for $d_x = 1$ [Nüsken - Ziegler 2004] $\tilde{O}(d_x d_y^{\frac{\omega+1}{2}})$

Interpolation and inverse composition

Bivariate interpolation

Input: $\{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathbb{K}^2$
$z_1, \ldots, z_n \in \mathbb{K}$ and degree bound d_y
Output: $p(x, y)$ with $z_i = p(x_i, y_i)$ and degree $< (n/d_y, d_y)$

Bivariate inverse modular composition

Input: $f(x), a(x)$ with $\deg(a) < \deg(f) = n$
$b(x)$ of degree $< n$ and degree bound d_y
Output: $p(x, y)$ with $b(x) = p(x, a(x)) \bmod f(x)$ and deg $< (n/d_y, d_y)$

$\tilde{O}(d_y^{\omega-1} n)$ using [Storjohann 2003] + [Neiger 2016] $\rightsquigarrow \tilde{O}(n^{\frac{\omega+1}{2}})$ if $d_y \approx \sqrt{n}$

Interpolation and inverse composition

Bivariate interpolation

Input: $\{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathbb{K}^2$ with $x_1, \ldots, x_n$ distinct
$z_1, \ldots, z_n \in \mathbb{K}$ and degree bound d_y

Output: $p(x, y)$ with $z_i = p(x_i, y_i)$ and degree $< (n/d_y, d_y)$

Ideal $\mathcal{I} = \langle f(x), y - a(x) \rangle$

- $f(x) = (x - x_1) \cdots (x - x_n)$
- $a(x)$ the interpolant $a(x_i) = y_i$
- $b(x)$ the interpolant $b(x_i) = z_i$

Bivariate inverse modular composition

Input: $f(x), a(x)$ with $\deg(a) < \deg(f) = n$
$b(x)$ of degree $< n$ and degree bound d_y

Output: $p(x, y)$ with $b(x) = p(x, a(x)) \bmod f(x)$ and deg $< (n/d_y, d_y)$

$\tilde{O}(d_y^{\omega-1} n)$ using [Storjohann 2003] + [Neiger 2016] $\rightsquigarrow \tilde{O}(n^{\frac{\omega+1}{2}})$ if $d_y \approx \sqrt{n}$

Interpolation and inverse composition

Bivariate interpolation

PreInput: $\{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathbb{K}^2$ with $x_1, \ldots, x_n$ distinct
Input: $z_1, \ldots, z_n \in \mathbb{K}$ and degree bound d_y
Output: $p(x, y)$ with $z_i = p(x_i, y_i)$ and degree $< (n/d_y, d_y)$

Ideal $\mathcal{I} = \langle f(x), y - a(x) \rangle$

- $f(x) = (x - x_1) \cdots (x - x_n)$
- $a(x)$ the interpolant $a(x_i) = y_i$
- $b(x)$ the interpolant $b(x_i) = z_i$

Bivariate inverse modular composition

PreInput: $f(x), a(x)$ with $\deg(a) < \deg(f) = n$
Input: $b(x)$ of degree $< n$ and degree bound d_y
Output: $p(x, y)$ with $b(x) = p(x, a(x)) \bmod f(x)$ and deg $< (n/d_y, d_y)$

$\tilde{O}(d_y^{\omega-1} n)$ using [Storjohann 2003] + [Neiger 2016] $\rightsquigarrow \tilde{O}(n^{\frac{\omega+1}{2}})$ if $d_y \approx \sqrt{n}$

Contribution

Precomputation from f, a or from $(x_i, y_i)_i$

Reshaper polynomial $r_1(x, y), \ldots, r_k(x, y)$ with $k \approx \log_{3/2}(n)$
$\rightsquigarrow$ fast remainders modulo $\mathcal{I}$ with specific degree shapes

Cost of precomputing reshapers: naive exponent ω, prospective exponent $\frac{\omega+1}{2}$

Online stage (evaluation/composition/interpolation)

cost $\tilde{O}(n + D)$, quasi-linear $\tilde{O}(n)$ for generic preinput

D = total size of reshapers $= \sum_i \deg_x(r_i) \deg_y(r_i)$
Generically each reshaper has size $\approx n$, and $D \in O(n \log(n))$

Contribution

Precomputation from f, a or from $(x_i, y_i)_i$

Reshaper polynomial $r_1(x, y), \ldots, r_k(x, y)$ with $k \approx \log_{3/2}(n)$
$\rightsquigarrow$ fast remainders modulo $\mathcal{I}$ with specific degree shapes

Cost of precomputing reshapers: naive exponent ω, prospective exponent $\frac{\omega+1}{2}$

Online stage (evaluation/composition/interpolation)

cost $\tilde{O}(n + D)$, quasi-linear $\tilde{O}(n)$ for generic preinput

D = total size of reshapers $= \sum_i \deg_x(r_i) \deg_y(r_i)$
Generically each reshaper has size $\approx n$, and $D \in O(n \log(n))$

Van der Hoeven and Lecerf obtained a similar result [Technical Report, 2020]
· quasi-linear multivariate evaluation and interpolation
· with precomputation and genericity assumption
· different approach based on generalizing the subproduct trees
· does not seem easily amenable to performing modular composition

Genericity and precomputations

Genericity assumptions often used recently for bivariate polynomials

· resultant of bivariate polynomials [Villard 2018]
· Gröbner basis and reduction for bivariate ideals [v.d.Hoeven - Larrieu 2019]
· modular composition and related problems [Neiger - Salvy - Schost - Villard, in progress]

- used to predict and leverage degree shapes of bases of ideals
- hoped next step: genericity removed via randomization

Situations allowing precomputations arise naturally

- encoding of some algebraic geometry error correcting codes [Miura 1993]
  (similar to Reed-Solomon encoding, but with bivariate polynomials)
- subroutine in distinct-degree factorization
- endless source of speed-ups in implementations

Precomputations in implementations

Shoup's NTL C++ library:

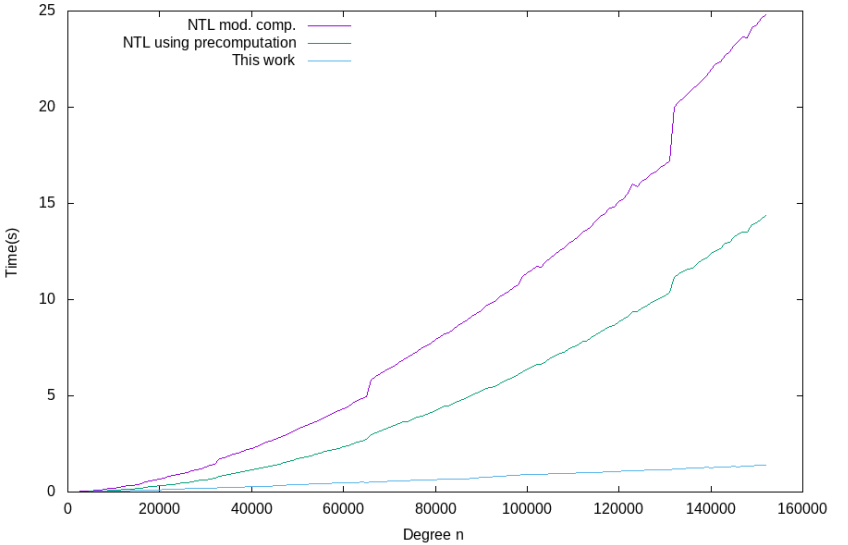
Excerpts from the documentation on univariate polynomials

```
/*************************************************************\
            Modular Arithmetic with Pre-Conditioning

If you need to do a lot of arithmetic modulo a fixed f, build
zz_pXModulus F for f.  This pre-computes information about f
that speeds up subsequent computations.

/*************************************************************\
                    More Pre-Conditioning

If you need to compute a * b % f for a fixed b, but for many
a's, it is much more efficient to first build a zz_pXMultiplier
B for b, and then use the MulMod routine below.

/*************************************************************\
    Faster Composition and Projection with Pre-Conditioning

If a single h is going to be used with many g's then you should
build a zz_pXArgument for h, and use the compose routine below.
```

Practicality of the new technique

Comparison of timings: modular composition, excluding precomputation

⇝ **significant speed-up**, including for small degrees

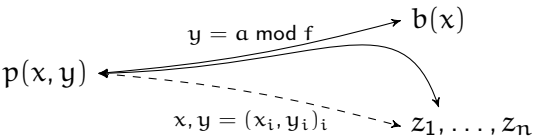


Outline

- Problem and result
- Approach: degree reshaping
- Application to composition, evaluation, interpolation

Reshaping degrees modulo the ideal

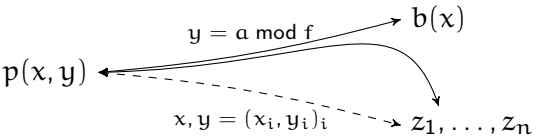
Our problems ask to compute remainders modulo $\mathcal{I} = \langle f(x), y - a(x) \rangle$



Example: $p(x, y)$ of degree $(0, n) \xleftrightarrow{\bmod \mathcal{I}} b(x)$ of degree $(n, 0)$
$\rightsquigarrow$ opposite degree shapes, difficult one-step transformation

Reshaping degrees modulo the ideal

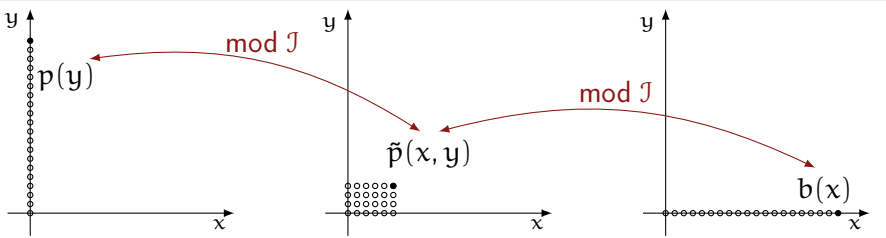
Our problems ask to compute remainders modulo $\mathcal{I} = \langle f(x), y - a(x) \rangle$



Example: $p(x, y)$ of degree $(0, n) \xleftrightarrow{\bmod \mathcal{I}} b(x)$ of degree $(n, 0)$

$\rightsquigarrow$ opposite degree shapes, difficult one-step transformation

we can use k intermediate transformations, for small k



Intermediate reshaping transformation

For input degree bound d_y, define reshaper $r(x, y)$:

- $y^{\frac{2}{3}d_y} - r(x, y)$ belongs to $\mathcal{I}$
- $r(x, y)$ has y-degree $< \frac{1}{3}d_y$, and minimal x-degree

$\rightsquigarrow$ $r(x, y)$ exists since $y - a(x) \in \mathcal{I}$
$\rightsquigarrow$ it has x-degree $< 3n/d_y$ generically

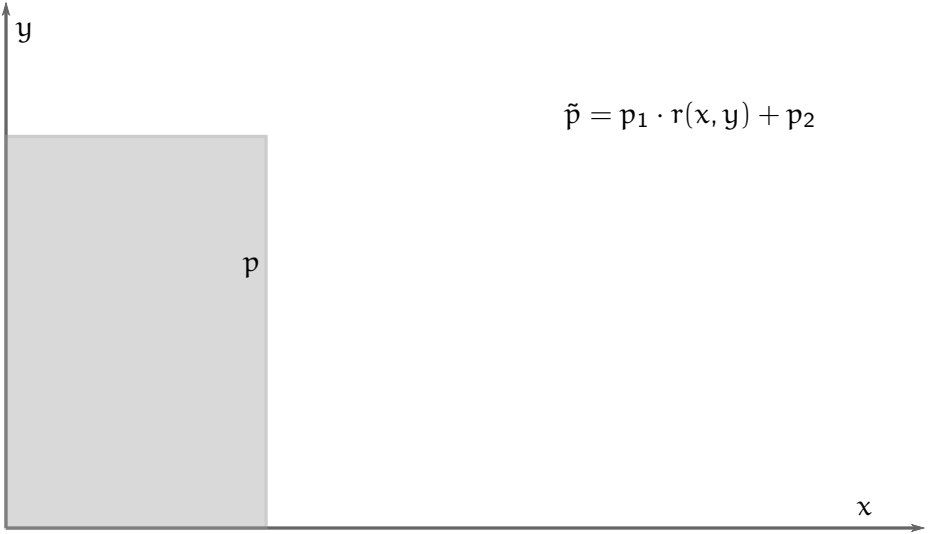
Remainder = bivariate multiplication, and degrees are controlled

For input polynomial $p(x, y)$ of degree $< (d_x, d_y)$,

- write $p = p_1 y^{\frac{2}{3}d_y} + p_2$
- then $p \xrightarrow{\bmod \mathcal{I}} \tilde{p} = p_1 \cdot r(x, y) + p_2$ (multiplication, quasi-linear)
- $\tilde{p}$ has y-degree $< \frac{2}{3}d_y$, and x-degree $\leqslant d_x + \deg_x(r)$

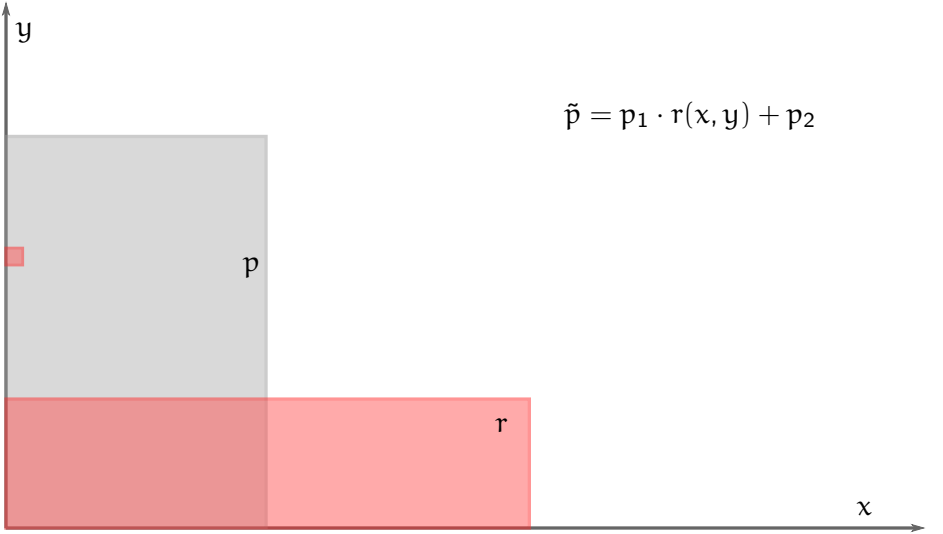
$\rightsquigarrow$ $\tilde{p}$ has x-degree $< d_x + 3n/d_y$ generically

Intermediate reshaping transformation



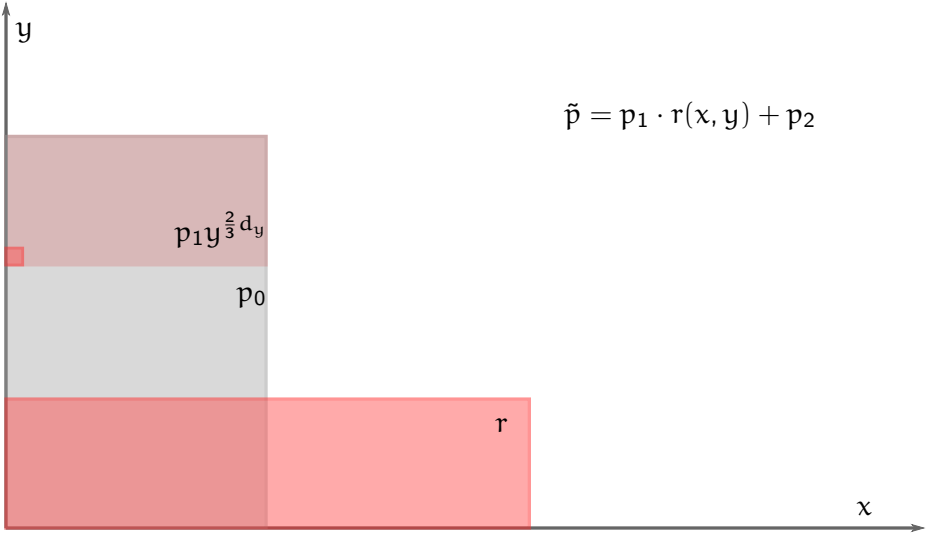
drawing courtesy of Grigory

Intermediate reshaping transformation



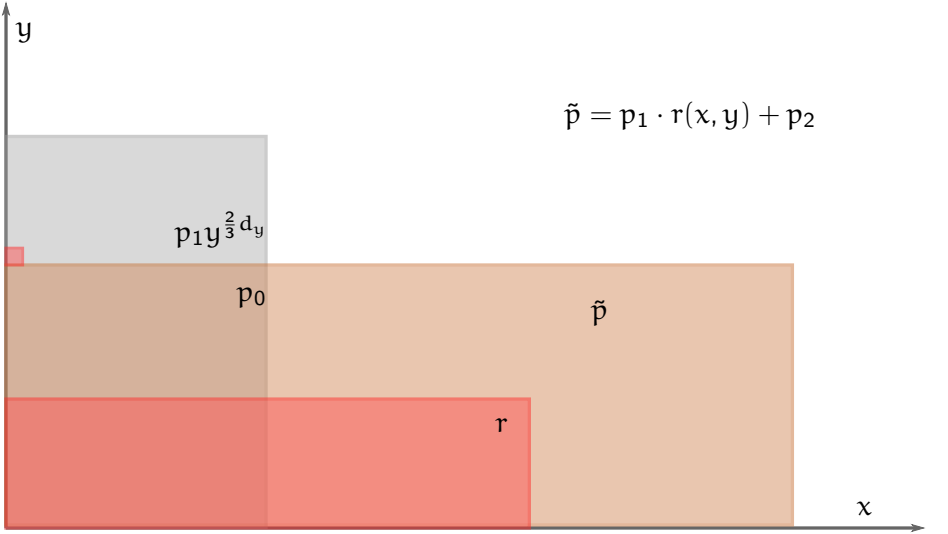
drawing courtesy of Grigory

Intermediate reshaping transformation



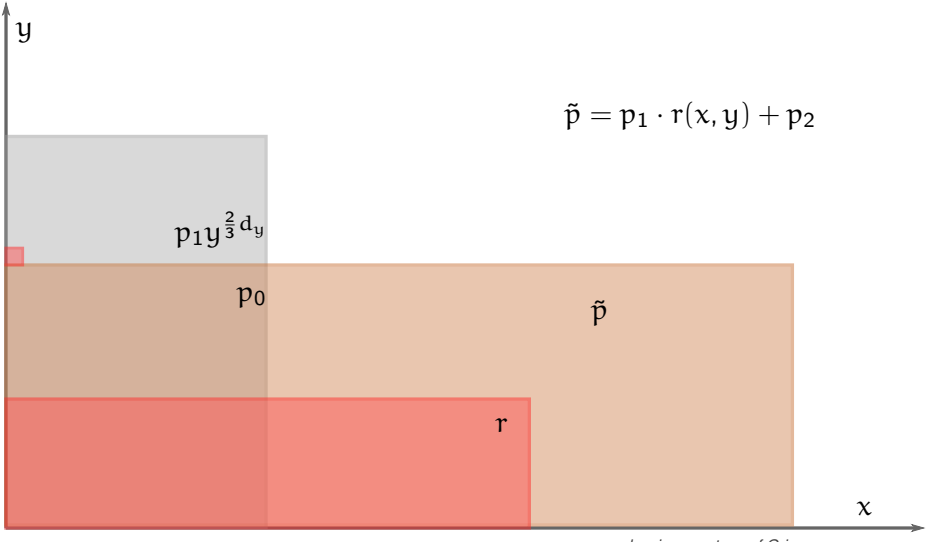
drawing courtesy of Grigory

Intermediate reshaping transformation



drawing courtesy of Grigory

Intermediate reshaping transformation



Note: $\text{size}(p) = d_x d_y \xrightarrow{\text{reshaping}} \text{size}(\tilde{p}) = (d_x + 3n/d_y)\frac{2}{3} d_y = \frac{2}{3} d_x d_y + 2n$

Iterating the reshaping: if initially $d_x d_y = n$, $\text{size}(p)$ grows to $6n$ and stabilizes

Iterated reshaping

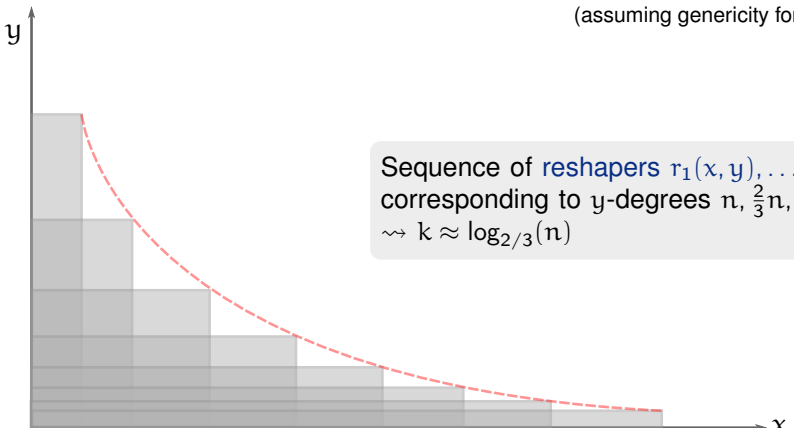
(animation courtesy of Grigory)

Iterated reshaping

Degree bounds, starting from $p(x, y)$ of degree $(0, n)$

transformation		1	2	3	$\cdots$	i	$\cdots$
x-degree	0	3	$3 + \frac{9}{2}$	$3 + \frac{9}{2} + \frac{27}{4}$	$\cdots$	$6(\frac{3}{2})^i - 6$	$\cdots$
y-degree	n	$\frac{2}{3}n$	$\frac{4}{9}n$	$\frac{8}{27}n$	$\cdots$	$(\frac{2}{3})^i n$	$\cdots$

(assuming genericity for the x-degree)



drawing courtesy of Grigory

Sequence of reshapers $r_1(x, y), \ldots, r_k(x, y)$ corresponding to y-degrees n, $\frac{2}{3}n$, $\frac{4}{9}n$, $\ldots$
$\rightsquigarrow k \approx \log_{2/3}(n)$

Algorithm and cost

Quasi-linear $\tilde{O}(n)$ for generic preinput and input $d_x d_y \in O(n)$

General: cost quasi-linear $\tilde{O}(d_x d_y + \sum_i \deg_x(r_i) \deg_y(r_i))$

Algorithm 1 RESHAPE$(p, r_1, \ldots, r_k)$

Input: bivariate polynomial $p \in \mathbb{K}[x, y]$
reshapers $r_1, \ldots, r_k$ w.r.t. $\mathcal{I} = \langle f(x), y - a(x) \rangle$
Output: univariate polynomial $\tilde{p} \in (p + \mathcal{I}) \cap \mathbb{K}[x]$ of degree $< n$

1: $\tilde{p} \leftarrow p$
2: **for** $i = 1, \ldots, k$ **do** ▷ *reshape in degree* d_i *with* $y^{\frac{2}{3}d_i} - r_i \in \mathcal{I}$
3: Write $\tilde{p} = p_1 y^{\frac{2}{3}d_i} + p_0$ where $\deg_y(p_0) < \frac{2}{3}d_i$
4: $\tilde{p} \leftarrow p_1 r_i + p_0$ ▷ *equivalent to* $\tilde{p} \leftarrow \tilde{p} - p_1 r_i \in \mathcal{I}$
5: **return** $\tilde{p} \bmod f$

Can be generalized to any ideal $\mathcal{I}$, including not zero-dimensional
. . . as long as reshapers are available!

Outline

- Problem and result
- Approach: degree reshaping
- Application to composition, evaluation, interpolation

Bivariate modular composition

$$p(x, y) \xrightarrow{\bmod \langle f(x), y - a(x) \rangle} b(x) = p(x, a(x)) \bmod f(x)$$

The reshaping strategy applies directly
$\rightsquigarrow$ quasi-linear modular composition after precomputation on a, f

Genericity $=$ reshapers have small x-degree

- "conjectured" for any f and generic a
- observed experimentally (random a over large finite field $\mathbb{K}$)
- proved for squarefree f and generic a

  $\rightsquigarrow$ brings us to multipoint evaluation, up to a field extension

Bivariate multipoint evaluation

$$p(x, y) \xrightarrow{\bmod \langle f(x), y - a(x) \rangle} b(x) \xrightarrow{x = x_1, \ldots, x_n} p(x_1, y_1), \ldots, p(x_n, y_n)$$

The reshaping strategy applies, assuming pairwise distinct x_i's
$\rightsquigarrow$ quasi-linear bivariate evaluation after precomputation on $(x_i, y_i)_i$

- distinct coordinates $x_1, \ldots, x_n$
- generic coordinates $y_1, \ldots, y_n$

proved → small x-degree reshapers

Bivariate multipoint evaluation

$$p(x,y) \xrightarrow{\bmod \langle f(x), y-a(x)\rangle} b(x) \xrightarrow{x=x_1,\ldots,x_n} p(x_1,y_1),\ldots,p(x_n,y_n)$$

The reshaping strategy applies, assuming pairwise distinct x_i's
$\rightsquigarrow$ quasi-linear bivariate evaluation after precomputation on $(x_i, y_i)_i$

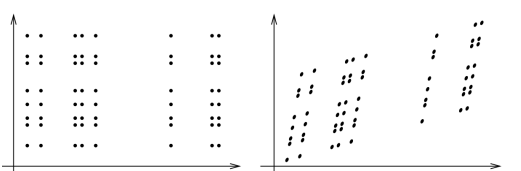
- distinct coordinates $x_1, \ldots, x_n$
- generic coordinates $y_1, \ldots, y_n$

proved → small x-degree reshapers

If the x_i's are not distinct, apply random point shearing [Nüsken - Ziegler 2004]

Evaluate $p(x + \theta y, y)$
at points $(x_i - \theta y_i, y_i)_i$

- distinct new x-coordinates
- new degrees $d_x + d_y, d_y$



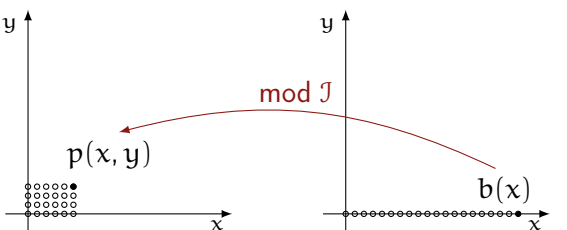
Bivariate interpolation

$$z_1, \ldots, z_n \xrightarrow[\text{interpolation}]{\text{univariate}} a(x), b(x) \xrightarrow{\bmod \langle f(x), y - a(x) \rangle} p(x, y)$$

Reshaping for $b(x) \longrightarrow p(x, y)$: precomputation = "transposed" reshapers

$\rightsquigarrow$ assuming distinct $y_1, \ldots, y_n$ and genericity in $x_1, \ldots, x_n$

quasi-linear bivariate interpolation after precomputation on $(x_i, y_i)_i$



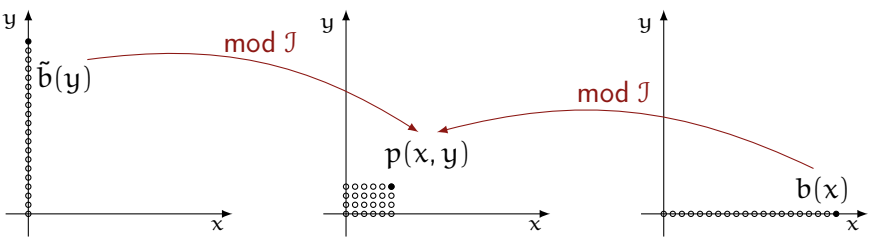
Bivariate interpolation

$$z_1, \ldots, z_n \xrightarrow[\text{interpolation}]{\text{univariate}} a(x), b(x) \xrightarrow{\bmod \langle f(x), y - a(x) \rangle} p(x, y)$$

Reshaping for $b(x) \longrightarrow p(x, y)$: precomputation = "transposed" reshapers

$\rightsquigarrow$ assuming distinct $y_1, \ldots, y_n$ and genericity in $x_1, \ldots, x_n$

quasi-linear bivariate interpolation after precomputation on $(x_i, y_i)_i$



Bivariate interpolation

$$z_1, \dots, z_n \xrightarrow[\text{interpolation}]{\text{univariate}} a(x), b(x) \xrightarrow{\bmod \langle f(x), y - a(x) \rangle} p(x, y)$$

Reshaping for $b(x) \longrightarrow p(x, y)$: precomputation = "transposed" reshapers

$\rightsquigarrow$ assuming distinct $y_1, \dots, y_n$ and genericity in $x_1, \dots, x_n$

quasi-linear bivariate interpolation after precomputation on $(x_i, y_i)_i$

Other approach, with distinct $x_1, \dots, x_n$ and genericity in $y_1, \dots, y_n$

- Shear points ensuring distinct $\tilde{y}_1, \dots, \tilde{y}_n$
- Interpolate $\tilde{b}(y)$ such that $\tilde{b}(\tilde{y}_i) = z_i$
- Reshape $\tilde{b}(y) \longrightarrow \tilde{p}(x, y)$
- Shift $\tilde{p}$ to make it agree with non-sheared points

Bivariate interpolation

$$z_1, \dots, z_n \xrightarrow[\text{interpolation}]{\text{univariate}} a(x), b(x) \xrightarrow{\bmod \langle f(x), y - a(x) \rangle} p(x, y)$$

Reshaping for $b(x) \longrightarrow p(x, y)$: precomputation = "transposed" reshapers

$\rightsquigarrow$ assuming distinct $y_1, \dots, y_n$ and genericity in $x_1, \dots, x_n$

quasi-linear bivariate interpolation after precomputation on $(x_i, y_i)_i$

Other approach, with distinct $x_1, \dots, x_n$ and genericity in $y_1, \dots, y_n$

- Shear points ensuring distinct $\tilde{y}_1, \dots, \tilde{y}_n$
- Interpolate $\tilde{b}(y)$ such that $\tilde{b}(\tilde{y}_i) = z_i$
- Reshape $\tilde{b}(y) \longrightarrow \tilde{p}(x, y)$ with degree about $(\sqrt{n}, \sqrt{n})$
- Shift $\tilde{p}$ to make it agree with non-sheared points
- Carry on reshaping until desired y-degree

Summary and perspectives

Summary

- under genericity assumptions, and allowing precomputations
- quasi-linear algorithms for basic operations on bivariate polynomials
- main technique: reshaping degrees w.r.t. an ideal
- experiments suggest good potential in practice even in low degree

Perspectives

- Precomputation in time $\tilde{O}(n^{\frac{\omega+1}{2}})$
- Efficient implementation and study of practical impact
- Extension to several variables
- Removing genericity assumptions via randomization