Computing minimal interpolation bases

Vincent Neiger[§,†,‡]

Claude-Pierre Jeannerod[§] Éric Schost[†] Gilles Villard[§]

[§]AriC, LIP, École Normale Supérieure de Lyon, France

[†]University of Waterloo, Ontario, Canada

[‡]Supported by the mobility grants *Explo'ra doc* from *Région Rhône-Alpes*, *Globalink Research Award - Inria* from *Mitacs & Inria*, and *Programme Avenir Lyon Saint-Étienne*.

Fields Institute, Toronto, Canada, October 2, 2015

Outline

1. List-decoding and interpolation problem
2. Fast algorithms using structured linear algebra
3. Fast algorithms using linear algebra over $\mathbb{K}[X]$

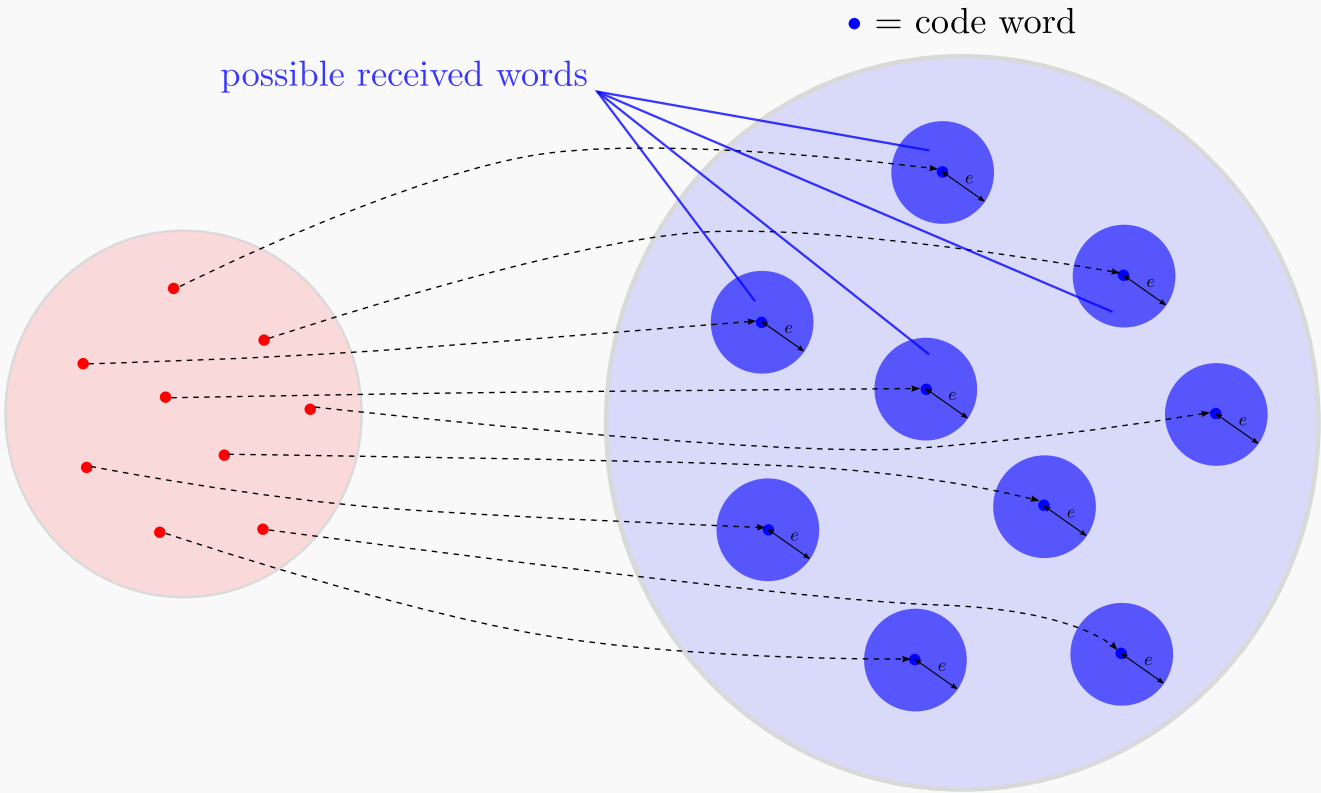
Outline

Reed-Solomon codes

Reliable delivery of data over an unreliable communication channel

$$f = f_0 + \cdots + f_k X^k \xrightarrow{\text{encoding}} (f(x_1), \ldots, f(x_n)) \xrightarrow{\text{noise}} y = (y_1, \ldots, y_n)$$

At most e errors during transmission: $\#\{i \mid f(x_i) \neq y_i\} \leqslant e$



Decoding

Sender

$$f = f_0 + \cdots + f_k X^k \xrightarrow{\text{encodes}} (f(x_1), \ldots, f(x_n))$$

Transmission

$$(f(x_1), \ldots, f(x_n)) \xrightarrow{\text{noisy}} (y_1, \ldots, y_n)$$

where $f(x_i) = y_i$ for $\geqslant n - e$ points

Receiver

$$(y_1, \ldots, y_n) \xrightarrow{\text{decodes}} f = f_0 + \cdots + f_k X^k$$

knowing the code: $x_1, \ldots, x_n$ and k, e

Unique decoding: constrained interpolation

Input: points $(x_1, y_1), \ldots, (x_n, y_n)$ and constraints k, e
Output: f of degree $\leqslant k$ with $f(x_i) = y_i$ for at least $n - e$ points

Define the error-locator polynomial

$$Q_1(X) = \prod_{i \mid \text{error}} (X - x_i)$$

Key equations

$$Q_1(x_i) f(x_i) = Q_1(x_i) y_i \quad \text{for all points}$$

Decoding algorithm [Berlekamp - Welch, 1983]

- find Q_0, Q_1 with

$$\begin{cases} \deg Q_0 \leqslant k + e, \quad \deg Q_1 \leqslant e, \\ Q_1 Y - Q_0 \text{ vanishes at all points } (x_i, y_i) \end{cases}$$

- return $f = Q_0 / Q_1$

Assumes $2e + k < n$

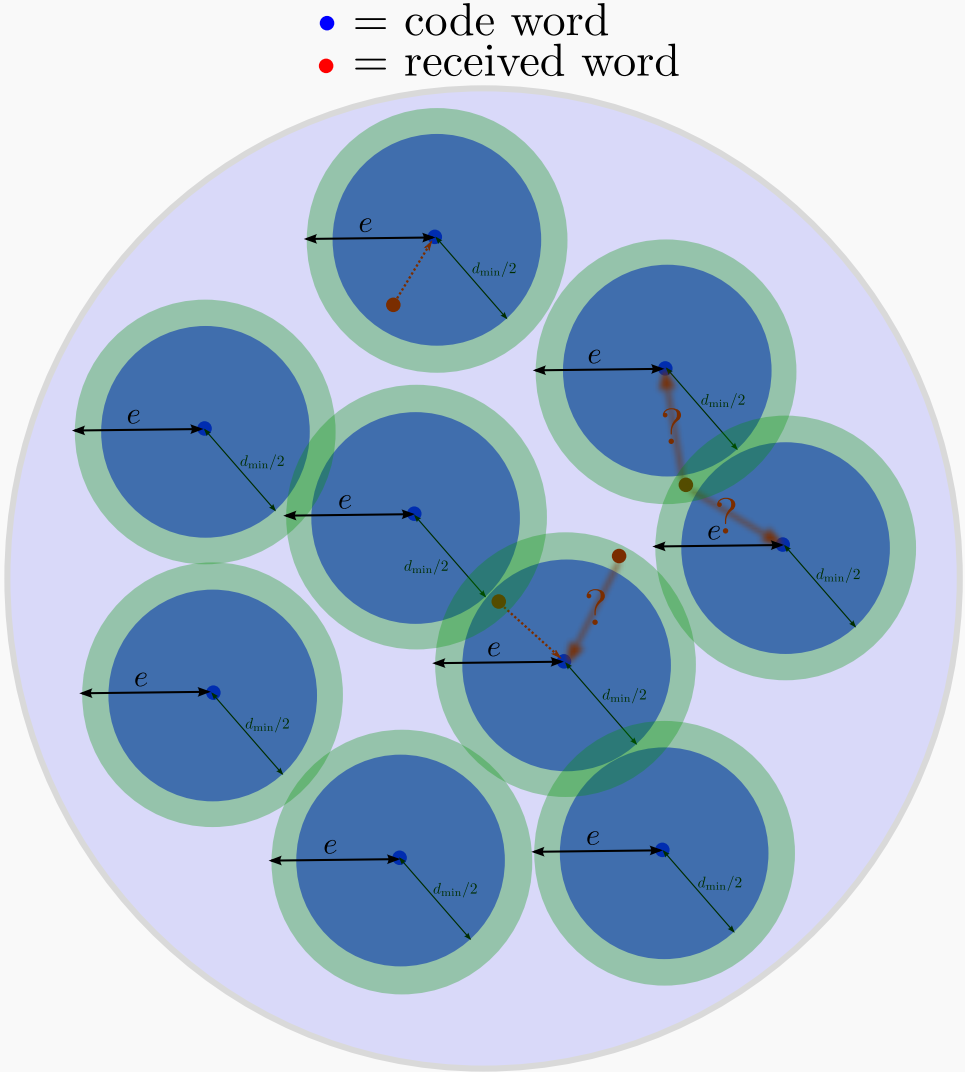
Non-unique decoding

How to "decode" when more errors?

transmission with $\leqslant e$ errors
where $e \geqslant d_{\min}/2$

possibly two (or more) code words
at the same distance...

the closest code word is not necessarily the one which was sent...



Non-unique decoding

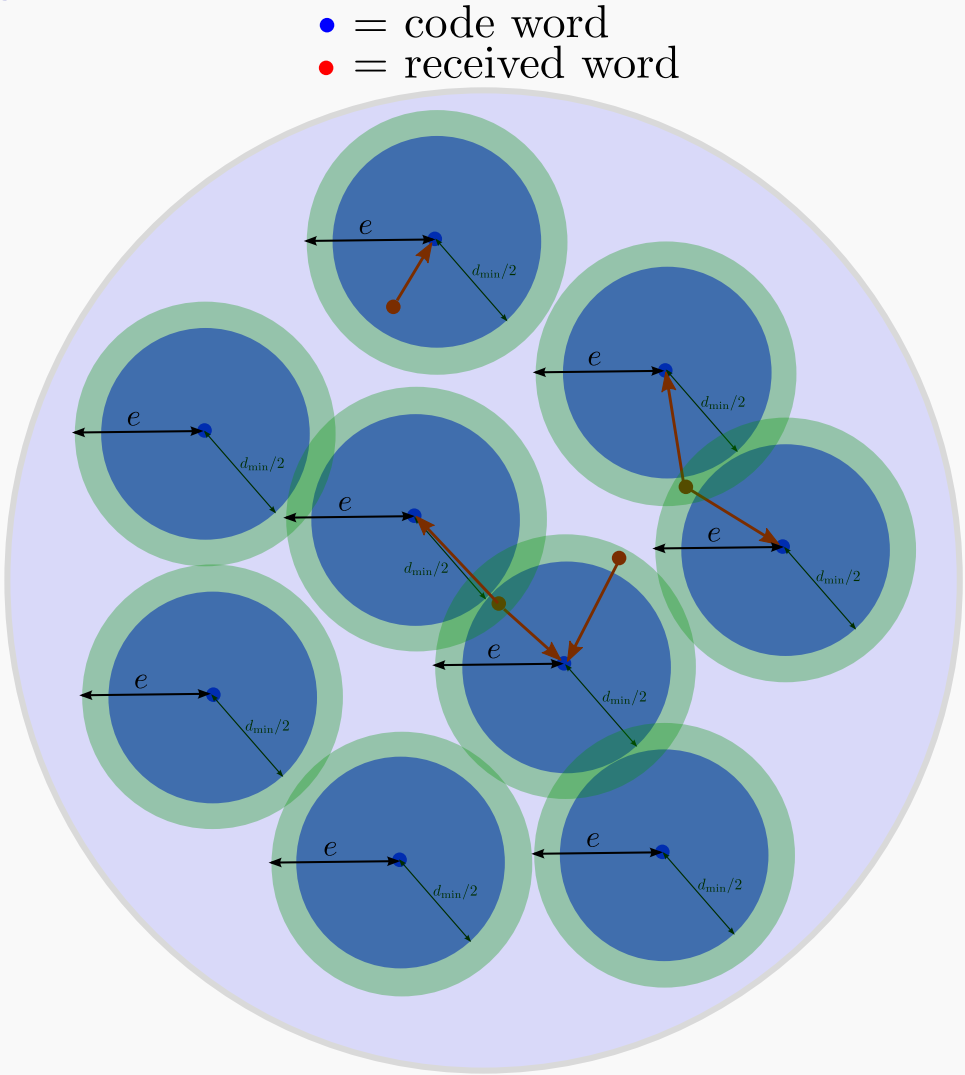
How to "decode" when more errors?

transmission with $\leqslant e$ errors
where $e \geqslant d_{\min}/2$

possibly two (or more) code words
at the same distance...

the closest code word is not necessarily the one which was sent...

$\rightsquigarrow$ Return a list of all code words
at distance $\leqslant e$
(called list-decoding)



List-decoding: Sudan algorithm

f solution: $\deg f \leqslant k$ and $f(x_i) = y_i$ for $\geqslant n - e$ points

[Sudan, 1997]

- Compute degree constraints ℓ and b
- Interpolation step
  compute $Q(X, Y) = Q_0 + Q_1 Y + \cdots + Q_\ell Y^\ell$ such that
  - $Q_0, \ldots, Q_\ell$ have small weighted degree: $\deg Q_j < b - jk$
  - $Q(x_i, y_i) = 0$ for all points
- Root-finding step
  the solutions f are among the Y-roots of $Q(X, Y)$

Here we focus on the Interpolation step.

Interpolation steps in related contexts

[Guruswami - Sudan, 1999]
List-decoding of Reed-Solomon codes,
further extends the error-correction bound

Compute $Q(X, Y) = Q_0 + Q_1 Y + \cdots + Q_\ell Y^\ell$ such that

- $Q_0, \ldots, Q_\ell$ have small weighted degree
- $Q(x_i, y_i) = 0$ with multiplicity μ for all points

Interpolation steps in related contexts

[Kötter - Vardy, 2003]
Soft-decision decoding of Reed-Solomon codes

$x_1, \ldots, x_n$ are not pairwise distinct
Compute $Q(X, Y) = Q_0 + Q_1 Y + \cdots + Q_\ell Y^\ell$ such that

- $Q_0, \ldots, Q_\ell$ have small weighted degree
- $Q(x_i, y_i) = 0$ with multiplicity μ_i for all points

Interpolation steps in related contexts

[Guruswami - Rudra, 2006]
List-decoding of folded Reed-Solomon codes:
extends the error-correction bound up to the information-theoretic limit

Compute $Q(X, Y_1, \ldots, Y_s) = \sum_{(j_1,\ldots,j_s)\in\Gamma} Q_{j_1,\ldots,j_s}\, Y_1^{j_1} \cdots Y_s^{j_s}$ such that

- the $Q_{(j_1,\ldots,j_s)}$ have small weighted degree
- $Q(x_i, y_{i1}, \ldots, y_{is}) = 0$ with multiplicity μ for all points

Outline

Linearization

Remember the constrained interpolation problem

$$\begin{cases} \deg Q_0 \leqslant e+k, \quad \deg Q_1 \leqslant e \\ Q_0(x_i) + Q_1(x_i)y_i = 0 \quad \text{for all points } (x_i, y_i) \end{cases}$$

Linearize it into a linear system

$$\text{polynomials} \xrightarrow{\text{linearization}} \text{linear algebra}$$

$$\text{degree bounds} \longrightarrow \text{matrix dimensions}$$

$$Q_0(X) \xrightarrow{\text{coefficients}} [Q_0^{(0)} \; Q_0^{(1)} \; \cdots \; Q_0^{(e+k)}]$$

$$Q_1(X) \xrightarrow{\text{coefficients}} [Q_1^{(0)} \; Q_1^{(1)} \; \cdots \; Q_1^{(e)}]$$

$$\text{equations} \longrightarrow \text{linear system}$$

Linearization: example ($n = 6, k = 1, e = 2$)

$$\begin{cases} \deg Q_0 \leqslant 3, \quad \deg Q_1 \leqslant 2 \\ Q_0(x_i) + Q_1(x_i) y_i = 0 \quad \text{for } i \in \{1, 2, 3, 4, 5, 6\} \end{cases}$$

Derive a linear system:

$$Q_0(x_1) = Q_0^{(0)} 1 + Q_0^{(1)} x_1 + Q_0^{(2)} x_1^2 + Q_0^{(3)} x_1^3 = \begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 \end{bmatrix} \begin{bmatrix} Q_0^{(0)} \\ Q_0^{(1)} \\ Q_0^{(2)} \\ Q_0^{(3)} \end{bmatrix}$$

Linearization: example ($n = 6, k = 1, e = 2$)

$$\begin{cases} \deg Q_0 \leqslant 3, \quad \deg Q_1 \leqslant 2 \\ Q_0(x_i) + Q_1(x_i) y_i = 0 \quad \text{for } i \in \{1,2,3,4,5,6\} \end{cases}$$

Derive a linear system:

$$\left[\begin{array}{cccc|ccc} 1 & x_1 & x_1^2 & x_1^3 & y_1 & x_1 y_1 & x_1^2 y_1 \end{array}\right] \left[\begin{array}{c} Q_0^{(0)} \\ Q_0^{(1)} \\ Q_0^{(2)} \\ Q_0^{(3)} \\ \hline Q_1^{(0)} \\ Q_1^{(1)} \\ Q_1^{(2)} \end{array}\right] = 0$$

Linearization: example ($n = 6, k = 1, e = 2$)

$$\begin{cases} \deg Q_0 \leqslant 3, \quad \deg Q_1 \leqslant 2 \\ Q_0(x_i) + Q_1(x_i) y_i = 0 \quad \text{for } i \in \{1,2,3,4,5,6\} \end{cases}$$

Derive a linear system:

$$\left[\begin{array}{cccc|ccc} 1 & x_1 & x_1^2 & x_1^3 & y_1 & x_1 y_1 & x_1^2 y_1 \\ 1 & x_2 & x_2^2 & x_2^3 & y_2 & x_2 y_2 & x_2^2 y_2 \end{array}\right] \left[\begin{array}{c} Q_0^{(0)} \\ Q_0^{(1)} \\ Q_0^{(2)} \\ Q_0^{(3)} \\ \hline Q_1^{(0)} \\ Q_1^{(1)} \\ Q_1^{(2)} \end{array}\right] = 0$$

Linearization: example ($n = 6, k = 1, e = 2$)

$$\begin{cases} \deg Q_0 \leqslant 3, \quad \deg Q_1 \leqslant 2 \\ Q_0(x_i) + Q_1(x_i) y_i = 0 \quad \text{for } i \in \{1,2,3,4,5,6\} \end{cases}$$

Derive a linear system:

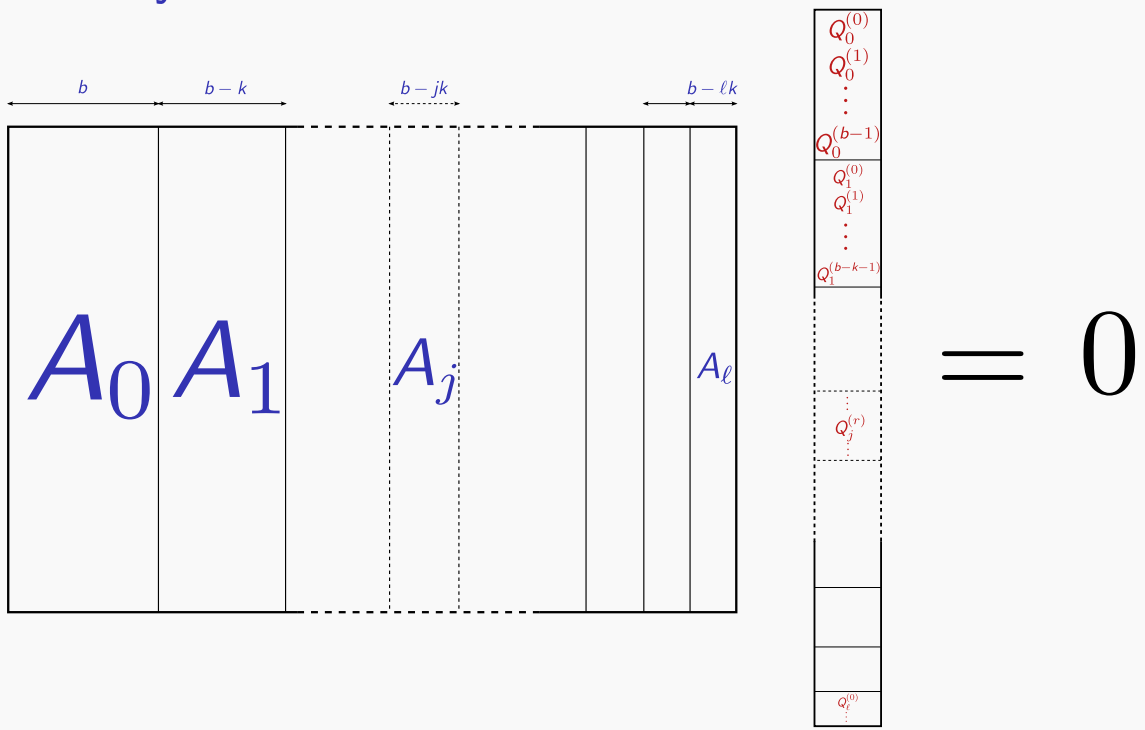
$$\left[\begin{array}{cccc|ccc} 1 & x_1 & x_1^2 & x_1^3 & y_1 & x_1 y_1 & x_1^2 y_1 \\ 1 & x_2 & x_2^2 & x_2^3 & y_2 & x_2 y_2 & x_2^2 y_2 \\ 1 & x_3 & x_3^2 & x_3^3 & y_3 & x_3 y_3 & x_3^2 y_3 \\ 1 & x_4 & x_4^2 & x_4^3 & y_4 & x_4 y_4 & x_4^2 y_4 \\ 1 & x_5 & x_5^2 & x_5^3 & y_5 & x_5 y_5 & x_5^2 y_5 \\ 1 & x_6 & x_6^2 & x_6^3 & y_6 & x_6 y_6 & x_6^2 y_6 \end{array}\right] \left[\begin{array}{c} Q_0^{(0)} \\ Q_0^{(1)} \\ Q_0^{(2)} \\ Q_0^{(3)} \\ \hline Q_1^{(0)} \\ Q_1^{(1)} \\ Q_1^{(2)} \end{array}\right] = 0$$

Structured system

The constrained interpolation problem

$$\begin{cases} \deg Q_0 < b, & \deg Q_1 < b-k, & \ldots, & \deg Q_\ell < b-\ell k \\ Q(x_i, y_i) = 0 & \text{for all points } (x_i, y_i) \end{cases}$$

becomes a linear system



where each block A_j is structured (e.g. Vandermonde)

Overview of results

[Olshevsky - Shokrollahi, 1999]
linearize the vanishing condition on each point
Vandermonde-like system, cost $\mathcal{O}(\ell n^2)$

[Roth - Ruckenstein, 2000] [Zeh - Gentner - Augot, 2011]
linearize the reversed extended key equation
Mosaic-Hankel system, cost $\mathcal{O}(\ell n^2)$
using an adapted version of [Feng - Tzeng, 1991]

[Chowdhury - Jeannerod - Neiger - Schost - Villard, 2015]
linearize the extended key equation
Toeplitz-like system, cost $\mathcal{O}^{\sim}(\ell^{\omega-1} n)$
using [Bostan - Jeannerod - Schost, 2007]

Structured system approach:

$$\tilde{\mathcal{O}}(\ell^{\omega-1} n), \quad \text{probabilistic algorithm}$$

assuming $x_1, \ldots, x_n$ pairwise distinct

Goal:

$$\tilde{\mathcal{O}}(\ell^{\omega-1} n), \quad \text{deterministic algorithm}$$

Structured matrices $\longrightarrow$ Polynomial matrices

Outline

Minimal interpolation bases

Constrained bivariate interpolation in Y-degree ℓ

$$\begin{cases} \deg Q_0 < b, \quad \deg Q_1 < b-k, \quad \ldots, \quad \deg Q_\ell < b-\ell k \\ Q(x_i, y_i) = 0 \quad \text{for all points } \{(x_i, y_i)\}_{1 \leqslant i \leqslant n} \end{cases}$$

- *interpolation basis* = basis of the module of interpolants

$$\{(Q_0, \ldots, Q_\ell) \in \mathbb{K}[X]^{\ell+1} \ / \ Q(x_i, y_i) = 0 \text{ for } 1 \leqslant i \leqslant n\}$$

$\rightsquigarrow$ matrix $\mathbf{P}$ in $\mathbb{K}[X]^{\ell+1 \times \ell+1}$ with rows $P_0, \ldots, P_\ell$ interpolants

- $\mathbf{P}$ *$\vec{w}$-minimal* interpolation basis
  if $(\vec{w}\text{-degree}(P_0), \ldots, \vec{w}\text{-degree}(P_\ell))$ is minimal

$\Longrightarrow$ an interpolant Q such that $\vec{w}\text{-degree}(Q) < b$ can be found in a $\vec{w}$-minimal interpolation basis (unless no such Q exists)

Interpolation basis: example

```
sage: bF = GF(997) ; pR.<X> = bF[]
sage: ell=4 ; n=20 ; k=2
sage: points = [ ( bF.random_element() , bF.random_element() ) for i in range(n) ]
sage:
sage: L = pR.lagrange_polynomial( points )
sage: M = prod( [ X-points[i][0] for i in range(n) ] )
sage: P = Matrix( pR, 4, 4, [ [M,0,0,0], [ -L,1,0,0], [0,-L,1,0], [0,0,-L,1] ] )
sage:
sage: print [ P( xi ) * vector( [ 1, yi, yi^2, yi^3 ] ) for (xi,yi) in points ]
[(0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0),
(0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0), (
0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0), (0, 0, 0, 0), (0
, 0, 0, 0), (0, 0, 0, 0)]
```

Using polynomial lattice reduction

[Alekhnovich, 2002] [Reinhard, 2003] [Lee - O'Sullivan 2006 & 2008]
[Beelen - Brander, 2010] [Bernstein, 2011] [Cohn - Heninger, 2012 & 2015]

- Compute a known basis of interpolants
- Lattice basis reduction $\rightsquigarrow$ $\vec{w}$-minimal interpolation basis
- Return row with smallest weighted-degree

Cost bound: $\tilde{\mathcal{O}}(\ell^\omega n)$
using deterministic lattice reduction
[Gupta - Sarkar - Storjohann - Valeriote, 2012]

Structured system approach:

$$\tilde{\mathcal{O}}(\ell^{\omega-1} n), \quad \text{probabilistic algorithm}$$

Lattice reduction approach:

$$\tilde{\mathcal{O}}(\ell^{\omega} n), \quad \text{deterministic algorithm}$$

Goal:

$$\tilde{\mathcal{O}}(\ell^{\omega-1} n), \quad \text{deterministic algorithm}$$

Using lattice reduction: example

```
sage: bF = GF(997) ; pR.<X> = bF[]
sage: ell=4 ; n=20 ; k=2
sage: points = [ ( bF.random_element() , bF.random_element() ) for i in range(n) ]
sage:
sage: L = pR.lagrange_polynomial( points )
sage: M = prod( [ X-points[i][0] for i in range(n) ] )
sage: P = Matrix( pR, 4, 4, [ [M,0,0,0], [ -L,1,0,0], [0,-L,1,0], [0,0,-L,1] ] )
sage: Pmin = weak_popov_form( P, [0,k,2*k,3*k] )
sage:
sage: print is_minimal( P, [0,k,2*k,3*k] ); print degree_matrix( P )
False
[20 -1 -1 -1]
[19  0 -1 -1]
[-1 19  0 -1]
[-1 -1 19  0]
sage: print is_minimal( Pmin, [0,k,2*k,3*k] ); degree_matrix( Pmin )
True
[8 5 3 1]
[8 6 4 2]
[8 6 3 1]
[8 6 4 0]
```

Iterative interpolation algorithm

In coding theory: Kötter iterative algorithm, cost: $\mathcal{O}(\ell n^2)$

Algorithm ([Kötter, 1996])

1. $\mathbf{P} = \text{Identity}$
2. weights $\vec{w} = (0, k, 2k, \ldots, \ell k)$
3. *For* i *from* 1 *to* n *do*
   - **a.** Evaluate:
     compute $P_0(x_i, y_i), \ldots, P_\ell(x_i, y_i)$
   - **b.** Choose pivot & update weights
     π with smallest w_π such that $P_\pi(x_i, y_i) \neq 0$
     $w_\pi = w_\pi + 1$
   - **c.** Eliminate:
     For $j \neq \pi$ *do* $P_j = P_j - \frac{P_j(x_i,y_i)}{P_\pi(x_i,y_i)} P_\pi$ /* $\forall j \neq \pi,\ P_j(x_i, y_i) = 0$ */
     $P_\pi = (X - x_i) P_\pi$ /* $P_\pi(x_i, y_i) = 0$ */

After i iterations $\mathbf{P}$ is a $\vec{w}$-minimal interpolation basis for points $1, \ldots, i$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 1$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$\begin{bmatrix} 0 & 2 & 4 & 6 \end{bmatrix}$$

Basis P

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 80 & 73 & 73 & 35 & 66 & 46 & 91 & 64 \\ 95 & 91 & 91 & 61 & 88 & 79 & 36 & 22 \\ 34 & 47 & 47 & 1 & 85 & 45 & 75 & 50 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 1$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$\begin{bmatrix} 0 & 2 & 4 & 6 \end{bmatrix}$$

Basis P

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 80 & 73 & 73 & 35 & 66 & 46 & 91 & 64 \\ 95 & 91 & 91 & 61 & 88 & 79 & 36 & 22 \\ 34 & 47 & 47 & 1 & 85 & 45 & 75 & 50 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 1$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$\begin{bmatrix} 0 & 2 & 4 & 6 \end{bmatrix}$$

Basis P

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 17 & 1 & 0 & 0 \\ 2 & 0 & 1 & 0 \\ 63 & 0 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 90 & 90 & 52 & 83 & 63 & 11 & 81 \\ 0 & 93 & 93 & 63 & 90 & 81 & 38 & 24 \\ 0 & 13 & 13 & 64 & 51 & 11 & 41 & 16 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 1$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$\begin{bmatrix} 1 & 2 & 4 & 6 \end{bmatrix}$$

Basis P

$$\begin{bmatrix} X + 73 & 0 & 0 & 0 \\ 17 & 1 & 0 & 0 \\ 2 & 0 & 1 & 0 \\ 63 & 0 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 7 & 88 & 8 & 59 & 3 & 93 & 35 \\ 0 & 90 & 90 & 52 & 83 & 63 & 11 & 81 \\ 0 & 93 & 93 & 63 & 90 & 81 & 38 & 24 \\ 0 & 13 & 13 & 64 & 51 & 11 & 41 & 16 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 2$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$\begin{bmatrix} 1 & 2 & 4 & 6 \end{bmatrix}$$

Basis P

$$\begin{bmatrix} X + 73 & 0 & 0 & 0 \\ 17 & 1 & 0 & 0 \\ 2 & 0 & 1 & 0 \\ 63 & 0 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 7 & 88 & 8 & 59 & 3 & 93 & 35 \\ 0 & 90 & 90 & 52 & 83 & 63 & 11 & 81 \\ 0 & 93 & 93 & 63 & 90 & 81 & 38 & 24 \\ 0 & 13 & 13 & 64 & 51 & 11 & 41 & 16 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 2$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$ $\quad [1 \quad 2 \quad 4 \quad 6]$

Basis P

$$\begin{bmatrix} X + 73 & 0 & 0 & 0 \\ X + 90 & 1 & 0 & 0 \\ 56X + 16 & 0 & 1 & 0 \\ 12X + 66 & 0 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 7 & 88 & 8 & 59 & 3 & 93 & 35 \\ 0 & 0 & 81 & 60 & 45 & 66 & 7 & 19 \\ 0 & 0 & 74 & 26 & 96 & 55 & 8 & 44 \\ 0 & 0 & 2 & 63 & 80 & 47 & 90 & 48 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 2$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$\begin{bmatrix} 2 & 2 & 4 & 6 \end{bmatrix}$$

Basis P

$$\begin{bmatrix} X^2 + 42X + 65 & 0 & 0 & 0 \\ X + 90 & 1 & 0 & 0 \\ 56X + 16 & 0 & 1 & 0 \\ 12X + 66 & 0 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 0 & 47 & 8 & 61 & 85 & 44 & 10 \\ 0 & 0 & 81 & 60 & 45 & 66 & 7 & 19 \\ 0 & 0 & 74 & 26 & 96 & 55 & 8 & 44 \\ 0 & 0 & 2 & 63 & 80 & 47 & 90 & 48 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 3$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$\begin{bmatrix} 2 & 2 & 4 & 6 \end{bmatrix}$$

Basis P

$$\begin{bmatrix} X^2 + 42X + 65 & 0 & 0 & 0 \\ X + 90 & 1 & 0 & 0 \\ 56X + 16 & 0 & 1 & 0 \\ 12X + 66 & 0 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 0 & 47 & 8 & 61 & 85 & 44 & 10 \\ 0 & 0 & 81 & 60 & 45 & 66 & 7 & 19 \\ 0 & 0 & 74 & 26 & 96 & 55 & 8 & 44 \\ 0 & 0 & 2 & 63 & 80 & 47 & 90 & 48 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 3$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$ $\quad [3 \quad 2 \quad 4 \quad 6]$

Basis P

$$\begin{bmatrix} X^3 + 27X^2 + 17X + 92 & 0 & 0 & 0 \\ 54X^2 + 38X + 11 & 1 & 0 & 0 \\ 17X^2 + 91X + 54 & 0 & 1 & 0 \\ 66X^2 + 68X + 88 & 0 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 0 & 0 & 39 & 74 & 50 & 26 & 52 \\ 0 & 0 & 0 & 7 & 41 & 0 & 55 & 74 \\ 0 & 0 & 0 & 65 & 66 & 45 & 77 & 20 \\ 0 & 0 & 0 & 9 & 32 & 31 & 84 & 29 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 4$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$[3 \quad 2 \quad 4 \quad 6]$$

Basis P

$$\begin{bmatrix} X^3 + 27X^2 + 17X + 92 & 0 & 0 & 0 \\ 54X^2 + 38X + 11 & 1 & 0 & 0 \\ 17X^2 + 91X + 54 & 0 & 1 & 0 \\ 66X^2 + 68X + 88 & 0 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 0 & 0 & 39 & 74 & 50 & 26 & 52 \\ 0 & 0 & 0 & 7 & 41 & 0 & 55 & 74 \\ 0 & 0 & 0 & 65 & 66 & 45 & 77 & 20 \\ 0 & 0 & 0 & 9 & 32 & 31 & 84 & 29 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 4$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$ $\quad [3 \quad 3 \quad 4 \quad 6]$

Basis P

$$\begin{bmatrix} X^3 + 31X^2 + 27X + 3 & 36 & 0 & 0 \\ 54X^3 + 56X^2 + 56X + 36 & X + 65 & 0 & 0 \\ 56X^2 + 43X + 35 & 60 & 1 & 0 \\ 52X^2 + 33X + 60 & 68 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 95 & 50 & 66 & 0 \\ 0 & 0 & 0 & 0 & 54 & 0 & 19 & 58 \\ 0 & 0 & 0 & 0 & 4 & 45 & 79 & 95 \\ 0 & 0 & 0 & 0 & 7 & 31 & 41 & 17 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 5$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$\begin{bmatrix} 4 & 3 & 4 & 6 \end{bmatrix}$$

Basis P

$$\begin{bmatrix} X^4 + 45X^3 + 73X^2 + 90X + 42 & 36X + 19 & 0 & 0 \\ 81X^3 + 20X^2 + 9X + 20 & X + 67 & 0 & 0 \\ 2X^3 + 21X^2 + 41 & 35 & 1 & 0 \\ 52X^3 + 15X^2 + 79X + 22 & 0 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 13 & 13 & 0 \\ 0 & 0 & 0 & 0 & 0 & 89 & 55 & 58 \\ 0 & 0 & 0 & 0 & 0 & 48 & 17 & 95 \\ 0 & 0 & 0 & 0 & 0 & 12 & 78 & 17 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 6$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$\begin{bmatrix} 4 & 4 & 4 & 6 \end{bmatrix}$$

Basis P

$$\begin{bmatrix} X^4 + 19X^3 + 57X^2 + 44X + 26 & 74X + 43 & 0 & 0 \\ 81X^4 + 64X^3 + 51X^2 + 68X + 42 & X^2 + 40X + 34 & 0 & 0 \\ 3X^3 + 44X^2 + 54X + 64 & 6X + 49 & 1 & 0 \\ 28X^3 + 45X^2 + 44X + 52 & 50X + 52 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 66 & 70 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & 13 \\ 0 & 0 & 0 & 0 & 0 & 0 & 56 & 55 \\ 0 & 0 & 0 & 0 & 0 & 0 & 15 & 7 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8$ $\ell = 3$ $k = 2$, base field $\mathbb{F}_{97}$

Iteration: $i = 7$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$\begin{bmatrix} 5 & 4 & 4 & 6 \end{bmatrix}$$

Basis P

$$\begin{bmatrix} X^5 + 96X^4 + 65X^3 + 68X^2 + 19X + 62 & 74X^2 + 18X + 13 & 0 & 0 \\ 6X^4 + 94X^3 + 44X^2 + 66X + 32 & X^2 + 19X + 10 & 0 & 0 \\ 55X^4 + 78X^3 + 75X^2 + 49X + 39 & 2X + 86 & 1 & 0 \\ 13X^4 + 81X^3 + 10X^2 + 34X + 2 & 42X + 29 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 14 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 25 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 44 \end{bmatrix}$$

Kötter algorithm on an example

Parameters: $n = 8 \qquad \ell = 3 \qquad k = 2,$ base field $\mathbb{F}_{97}$

Iteration: $i = 8$

Points: $(24, 80), (31, 73), (15, 73), (32, 35), (83, 66), (27, 46), (20, 91), (59, 64)$

Weights $\vec{w}$

$$\begin{bmatrix} 5 & 5 & 4 & 6 \end{bmatrix}$$

Basis P

$$\begin{bmatrix} X^5 + 12X^4 + 10X^3 + 34X^2 + 65X + 2 & 60X^2 + 43X + 67 & 0 & 0 \\ 6X^5 + 31X^4 + 27X^3 + 89X^2 + 18X + 52 & X^3 + 57X^2 + 53X + 89 & 0 & 0 \\ 2X^4 + 56X^3 + 42X^2 + 48X + 15 & 72X^2 + 12X + 30 & 1 & 0 \\ 40X^4 + 19X^3 + 14X^2 + 40X + 49 & 53X^2 + 79X + 74 & 0 & 1 \end{bmatrix}$$

Evaluations

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Link with Hermite-Padé approximation

The Kötter algorithm is very similar to [Beckermann and Labahn, 1994] iterative algorithm for

Hermite-Padé approximation

Input: $n, \ell, \vec{w}$, a vector of polynomials $(F_0, \ldots, F_\ell)$ of degree $< n$
Output: a $\vec{w}$-minimal basis of approximants $(Q_0, \ldots, Q_\ell)$ such that

$$Q_0 F_0 + \cdots + Q_\ell F_\ell = 0 \bmod X^n$$

Fast deterministic algorithms have been developed

- general: $\mathcal{O}^{\sim}(\ell^\omega n)$
  [Beckermann and Labahn, 1994] [Giorgi - Jeannerod - Villard, 2003]
- assumption on $\vec{w}$ (satisfied here): $\mathcal{O}^{\sim}(\ell^{\omega-1} n)$
  [Storjohann, 2006] [Zhou - Labahn, 2012]

Unified problem

In [Beckermann and Labahn, 2000], "rational interpolation problem"
$\rightsquigarrow$ unifies both situations

Replaced in this general framework,
Beckermann-Labahn and Kötter iterative algorithms are two
specializations of the same unified algorithm

Fast Hermite-Padé $\Rightarrow$ unified fast algorithm?

- $\tilde{\mathcal{O}}(\ell^{\omega} n)$: adapting Beckermann-Labahn / Giorgi-Jeannerod-Villard divide-and-conquer algorithms requires some work
- $\tilde{\mathcal{O}}(\ell^{\omega-1} n)$ under assumptions: unclear to me how to generalize the input linearization in [Storjohann, 2006] [Zhou - Labahn, 2012]

Structured system approach:

$$\tilde{\mathcal{O}}(\ell^{\omega-1} n), \quad \text{probabilistic algorithm}$$

Lattice reduction approach:

$$\tilde{\mathcal{O}}(\ell^{\omega} n), \quad \text{deterministic algorithm}$$

For the similar problem of Hermite-Padé approximation:

$$\tilde{\mathcal{O}}(\ell^{\omega-1} n), \quad \text{deterministic algorithm}$$

Structured system approach:

$$\tilde{\mathcal{O}}(\ell^{\omega-1} n), \quad \text{probabilistic algorithm}$$

Lattice reduction approach:

$$\tilde{\mathcal{O}}(\ell^{\omega} n), \quad \text{deterministic algorithm}$$

For the similar problem of Hermite-Padé approximation:

$$\tilde{\mathcal{O}}(\ell^{\omega-1} n), \quad \text{deterministic algorithm}$$

Now,

$$\tilde{\mathcal{O}}(\ell^{\omega-1} n), \quad \text{deterministic algorithm}$$

for the unified problem

Weights and size of the basis

Here: we show the main ingredients, focusing on

divide-and-conquer version of Kötter algorithm in $\tilde{\mathcal{O}}(\ell^{\omega-1} n)$

Previous work: [J. Nielsen, 2014] in $\tilde{\mathcal{O}}(\ell^{\omega} n)$

Weights and size of the basis

Here: we show the main ingredients, focusing on

divide-and-conquer version of Kötter algorithm in $\tilde{\mathcal{O}}(\ell^{\omega-1} n)$

Previous work: [J. Nielsen, 2014] in $\tilde{\mathcal{O}}(\ell^{\omega} n)$

In general, the size of a $\vec{w}$-minimal basis may be $\Theta(\ell^2 n)$
$\rightsquigarrow$ seems to compromise our target cost $\tilde{\mathcal{O}}(\ell^{\omega-1} n)$

Assumption: weights $\vec{w}$ satisfies $w_0 + \cdots + w_\ell \in \mathcal{O}(n)$

Consequence: a $\vec{w}$-minimal basis has sum of row degrees $\mathcal{O}(n)$
$\rightsquigarrow$ in particular, size $\mathcal{O}(\ell n)$

Ingredient 0: polynomial matrix multiplication

Divide-and-conquer algorithm:

- $\mathbf{P}^{(1)}$ for first $n/2$ points and weights $\vec{w}$
- $\mathbf{P}^{(2)}$ for last $n/2$ points and updated weights $\vec{w}'$
- return $\mathbf{P} = \mathbf{P}^{(2)}\mathbf{P}^{(1)}$

Then $\mathbf{P}$ is a $\vec{w}$-minimal interpolation basis

Ingredient 0: polynomial matrix multiplication

Divide-and-conquer algorithm:

- $\mathbf{P}^{(1)}$ for first $n/2$ points and weights $\vec{w}$
- $\mathbf{P}^{(2)}$ for last $n/2$ points and updated weights $\vec{w}'$
- return $\mathbf{P} = \mathbf{P}^{(2)}\mathbf{P}^{(1)}$

Then $\mathbf{P}$ is a $\vec{w}$-minimal interpolation basis

Using assumption on the weights,

$$\mathbf{P} = \mathbf{P}^{(2)}\mathbf{P}^{(1)} \text{ is computed in } \tilde{\mathcal{O}}(\ell^{\omega-1} n)$$

using fast multiplication with unbalanced row degrees
[Zhou - Labahn - Storjohann, 2012]

Ingredient 1: maintaining an evaluation matrix (1/2)

Divide-and-conquer algorithm:

- $\mathbf{P}^{(1)}$ for first $n/2$ points and $\vec{w}$
- $\mathbf{P}^{(2)}$ for last $n/2$ points and $\vec{w}'$, with $\mathbf{P}^{(1)}$-dependent evaluation
- return $P^{(2)}P^{(1)}$

Store an evaluation matrix $\mathbf{E}$
= evaluations of the global basis at points currently being processed

Ingredient 1: maintaining an evaluation matrix (1/2)

Divide-and-conquer algorithm:

- $\mathbf{P}^{(1)}$ for first $n/2$ points and $\vec{w}$
- $\mathbf{P}^{(2)}$ for last $n/2$ points and $\vec{w}'$, with $\mathbf{P}^{(1)}$-dependent evaluation
- return $P^{(2)}P^{(1)}$

Store an evaluation matrix $\mathbf{E}$
= evaluations of the global basis at points currently being processed

Example: after first recursive call,

$$\mathbf{E} = \begin{bmatrix} 0 & \cdots & 0 & P_0^{(1)}(x_{n/2+1}, y_{n/2+1}) & \cdots & P_0^{(1)}(x_n, y_n) \\ 0 & \cdots & 0 & P_1^{(1)}(x_{n/2+1}, y_{n/2+1}) & \cdots & P_1^{(1)}(x_n, y_n) \\ \vdots & \cdots & \vdots & \vdots & \cdots & \vdots \\ 0 & \cdots & 0 & P_\ell^{(1)}(x_{n/2+1}, y_{n/2+1}) & \cdots & P_\ell^{(1)}(x_n, y_n) \end{bmatrix} \in \mathbb{K}^{\ell+1 \times n}$$

Ingredient 1: maintaining an evaluation matrix (2/2)

Updating the evaluation matrix:

Given some evaluation matrix at n points $\mathbf{E} = [E_1 | \cdots | E_n] \in \mathbb{K}^{\ell+1 \times n}$

Compute $\mathbf{P} \star \mathbf{E} = [\mathbf{P}(x_1)E_1 \mid \cdots \mid \mathbf{P}(x_n)E_n]$

Assumption on weights $\Rightarrow$ $\mathbf{P}$ has sum of row degrees in $\mathcal{O}(n)$

then evaluation matrix update in $\tilde{\mathcal{O}}(\ell^{\omega-1} n)$

Compromise between the number of distinct x_i and the degrees involved

- if few distinct x_i:
  - evaluate $\mathbf{P}$ at those points,
  - perform scalar matrices multiplications (e.g. $\mathbf{P}(x_1)E$)
- if many distinct x_i:
  - interpolate with some $\mathbf{F}(X)$ the values in $\mathbf{E}$ at distinct x_i
  - perform a polynomial matrix multiplication of the form $\mathbf{P}(X)\mathbf{F}(X)$
  - evaluate the result at distinct x_i

Ingredient 2: linearizing at small orders (1/3)

Base case $\ell + 1 = n$, goal $\tilde{\mathcal{O}}(\ell^\omega)$

Input: points $x_1, \ldots, x_{\ell+1}$, evaluation matrix $\mathbf{E} \in \mathbb{K}^{\ell+1 \times \ell+1}$, here $\vec{w} = \vec{0}$

Output: $\vec{0}$-minimal interpolation basis $\mathbf{P}$

sum of row degrees $\leqslant \ell + 1 \Rightarrow$ average degree in the matrix $\leqslant 1$

Complete linearization over $\mathbb{K}$

$$\mathcal{K} = \begin{bmatrix} \mathbf{E} \\ \mathbf{ED} \\ \mathbf{ED^2} \\ \mathbf{ED^3} \\ \vdots \end{bmatrix} \qquad \text{where} \quad \mathbf{D} = \mathrm{Diag}(x_1, \ldots, x_{\ell+1})$$

minimal interpolation basis $\longleftrightarrow$ minimal linear relations between rows of $\mathcal{K}$

Fast computation of those relations, in $\mathcal{O}(\ell^\omega \log \ell)$

Ingredient 2: linearizing at small orders (2/3)

$$\mathcal{K} = \left[\begin{array}{cccc}
E_{11} & E_{12} & E_{13} & E_{14} \\
E_{21} & E_{22} & E_{23} & E_{24} \\
E_{31} & E_{32} & E_{33} & E_{34} \\
E_{41} & E_{42} & E_{43} & E_{44} \\
\hline
x_1 E_{11} & x_2 E_{12} & x_3 E_{13} & x_4 E_{14} \\
x_1 E_{21} & x_2 E_{22} & x_3 E_{23} & x_4 E_{24} \\
x_1 E_{31} & x_2 E_{32} & x_3 E_{33} & x_4 E_{34} \\
x_1 E_{41} & x_2 E_{42} & x_3 E_{43} & x_4 E_{44} \\
\hline
x_1^2 E_{11} & x_2^2 E_{12} & x_3^2 E_{13} & x_4^2 E_{14} \\
x_1^2 E_{21} & x_2^2 E_{22} & x_3^2 E_{23} & x_4^2 E_{24} \\
x_1^2 E_{31} & x_2^2 E_{32} & x_3^2 E_{33} & x_4^2 E_{34} \\
x_1^2 E_{41} & x_2^2 E_{42} & x_3^2 E_{43} & x_4^2 E_{44} \\
\hline
x_1^3 E_{11} & x_2^3 E_{12} & x_3^3 E_{13} & x_4^3 E_{14} \\
x_1^3 E_{21} & x_2^3 E_{22} & x_3^3 E_{23} & x_4^3 E_{24} \\
x_1^3 E_{31} & x_2^3 E_{32} & x_3^3 E_{33} & x_4^3 E_{34} \\
x_1^3 E_{41} & x_2^3 E_{42} & x_3^3 E_{43} & x_4^3 E_{44} \\
\hline
x_1^4 E_{11} & x_2^4 E_{12} & x_3^4 E_{13} & x_4^4 E_{14} \\
x_1^4 E_{21} & x_2^4 E_{22} & x_3^4 E_{23} & x_4^4 E_{24} \\
x_1^4 E_{31} & x_2^4 E_{32} & x_3^4 E_{33} & x_4^4 E_{34} \\
x_1^4 E_{41} & x_2^4 E_{42} & x_3^4 E_{43} & x_4^4 E_{44}
\end{array}\right]$$

Linearize in degree $\leqslant \ell + 1$

left nullspace $\longleftrightarrow$ interpolants

- Find the first $\operatorname{rank}(\mathcal{K})$ linearly independent rows
- Compute well-chosen linear relations between rows

$\rightsquigarrow$ minimal interpolation basis

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices

$$\left[\quad\right]$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 7 & 0 & 25 & 35 \\ 1 & 32 & 5 & 67 \\ 25 & 18 & 81 & 59 \\ \hline 59 & 75 & 61 & 73 \\ 12 & 0 & 45 & 95 \\ 71 & 75 & 9 & 71 \\ 29 & 24 & 10 & 77 \\ \hline 18 & 3 & 71 & 18 \\ 76 & 0 & 81 & 50 \\ 94 & 3 & 55 & 68 \\ 22 & 32 & 18 & 15 \\ \hline 17 & 4 & 89 & 35 \\ 61 & 0 & 10 & 11 \\ 78 & 4 & 2 & 46 \\ 10 & 75 & 13 & 13 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices

$$\begin{bmatrix} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 7 & 0 & 25 & 35 \\ 1 & 32 & 5 & 67 \\ 25 & 18 & 81 & 59 \\ \hline 59 & 75 & 61 & 73 \\ 12 & 0 & 45 & 95 \\ 71 & 75 & 9 & 71 \\ 29 & 24 & 10 & 77 \\ \hline 18 & 3 & 71 & 18 \\ 76 & 0 & 81 & 50 \\ 94 & 3 & 55 & 68 \\ 22 & 32 & 18 & 15 \\ \hline 17 & 4 & 89 & 35 \\ 61 & 0 & 10 & 11 \\ 78 & 4 & 2 & 46 \\ 10 & 75 & 13 & 13 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices

$$\begin{bmatrix} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 59 & 75 & 61 & 73 \\ 12 & 0 & 45 & 95 \\ 71 & 75 & 9 & 71 \\ 29 & 24 & 10 & 77 \\ \hline 18 & 3 & 71 & 18 \\ 76 & 0 & 81 & 50 \\ 94 & 3 & 55 & 68 \\ 22 & 32 & 18 & 15 \\ \hline 17 & 4 & 89 & 35 \\ 61 & 0 & 10 & 11 \\ 78 & 4 & 2 & 46 \\ 10 & 75 & 13 & 13 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices 00 01 02

$$\begin{bmatrix} 96 & 96 & 1 \\ & & \\ & & \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 59 & 75 & 61 & 73 \\ 12 & 0 & 45 & 95 \\ 71 & 75 & 9 & 71 \\ 29 & 24 & 10 & 77 \\ \hline 18 & 3 & 71 & 18 \\ 76 & 0 & 81 & 50 \\ 94 & 3 & 55 & 68 \\ 22 & 32 & 18 & 15 \\ \hline 17 & 4 & 89 & 35 \\ 61 & 0 & 10 & 11 \\ 78 & 4 & 2 & 46 \\ 10 & 75 & 13 & 13 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices 00 01 02

$$\begin{bmatrix} 96 & 96 & 1 \\ & & \\ & & \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 59 & 75 & 61 & 73 \\ 12 & 0 & 45 & 95 \\ 0 & 0 & 0 & 0 \\ 29 & 24 & 10 & 77 \\ \hline 18 & 3 & 71 & 18 \\ 76 & 0 & 81 & 50 \\ 0 & 0 & 0 & 0 \\ 22 & 32 & 18 & 15 \\ \hline 17 & 4 & 89 & 35 \\ 61 & 0 & 10 & 11 \\ 0 & 0 & 0 & 0 \\ 10 & 75 & 13 & 13 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices 00 01 02

$$\begin{bmatrix} 96 & 96 & 1 \\ & & \\ & & \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 59 & 75 & 61 & 73 \\ 12 & 0 & 45 & 95 \\ 0 & 0 & 0 & 0 \\ 29 & 24 & 10 & 77 \\ \hline 18 & 3 & 71 & 18 \\ 76 & 0 & 81 & 50 \\ 0 & 0 & 0 & 0 \\ 22 & 32 & 18 & 15 \\ \hline 17 & 4 & 89 & 35 \\ 61 & 0 & 10 & 11 \\ 0 & 0 & 0 & 0 \\ 10 & 75 & 13 & 13 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices 00 01 02

$$\begin{bmatrix} 96 & 96 & 1 \\ & & \\ & & \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 0 & 0 & 0 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 18 & 3 & 71 & 18 \\ 76 & 0 & 81 & 50 \\ 0 & 0 & 0 & 0 \\ 22 & 32 & 18 & 15 \\ \hline 17 & 4 & 89 & 35 \\ 61 & 0 & 10 & 11 \\ 0 & 0 & 0 & 0 \\ 10 & 75 & 13 & 13 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices 00 01 02 03 10 11 13

$$\begin{bmatrix} 96 & 96 & 1 & & & & \\ 44 & 33 & 0 & 62 & 53 & 1 & \\ 31 & 35 & 0 & 68 & 93 & 0 & 1 \\ & & & & & & \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 0 & 0 & 0 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 18 & 3 & 71 & 18 \\ 76 & 0 & 81 & 50 \\ 0 & 0 & 0 & 0 \\ 22 & 32 & 18 & 15 \\ \hline 17 & 4 & 89 & 35 \\ 61 & 0 & 10 & 11 \\ 0 & 0 & 0 & 0 \\ 10 & 75 & 13 & 13 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices 00 01 02 03 10 11 13

$$\begin{bmatrix} 96 & 96 & 1 & & & & \\ 44 & 33 & 0 & 62 & 53 & 1 & \\ 31 & 35 & 0 & 68 & 93 & 0 & 1 \\ & & & & & & \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 0 & 0 & 0 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 18 & 3 & 71 & 18 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 17 & 4 & 89 & 35 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices 00 01 02 03 10 11 13

$$\begin{bmatrix} 96 & 96 & 1 & & & & \\ 44 & 33 & 0 & 62 & 53 & 1 & \\ 31 & 35 & 0 & 68 & 93 & 0 & 1 \\ & & & & & & \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 0 & 0 & 0 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 18 & 3 & 71 & 18 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 17 & 4 & 89 & 35 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices 00 01 02 03 10 11 13

$$\begin{bmatrix} 96 & 96 & 1 & & & & \\ 44 & 33 & 0 & 62 & 53 & 1 & \\ 31 & 35 & 0 & 68 & 93 & 0 & 1 \\ & & & & & & \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 0 & 0 & 0 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices 00 01 02 03 10 11 13 20

$$\begin{bmatrix} 96 & 96 & 1 & & & & & \\ 44 & 33 & 0 & 62 & 53 & 1 & & \\ 31 & 35 & 0 & 68 & 93 & 0 & 1 & \\ 19 & 44 & 0 & 15 & 18 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 0 & 0 & 0 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices 00 01 02 03 10 11 13 20

$$\begin{bmatrix} 96 & 96 & 1 & & & & & \\ 44 & 33 & 0 & 62 & 53 & 1 & & \\ 31 & 35 & 0 & 68 & 93 & 0 & 1 & \\ 19 & 44 & 0 & 15 & 18 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 0 & 0 & 0 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

Ingredient 2: linearizing at small orders (3/3)

Field $\mathbb{F}_{97}$, $\ell + 1 = n = 4$, points $71, 66, 60, 72$

Indices 00 01 02 03 10 11 13 20

$$\begin{bmatrix} 96 & 96 & 1 & & & & & \\ 44 & 33 & 0 & 62 & 53 & 1 & & \\ 31 & 35 & 0 & 68 & 93 & 0 & 1 & \\ 19 & 44 & 0 & 15 & 18 & 0 & 0 & 1 \end{bmatrix}$$

$$\downarrow$$

$$\begin{bmatrix} 96 & 96 & 1 & 0 \\ 53X + 44 & X + 33 & 0 & 62 \\ 93X + 31 & 35 & 0 & X + 68 \\ X^2 + 18X + 19 & 44 & 0 & 15 \end{bmatrix}$$

$$\mathcal{K} = \left[\begin{array}{cccc} 91 & 32 & 77 & 32 \\ 0 & 5 & 34 & 40 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 55 & 81 \\ \hline 0 & 0 & 0 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

Ingredient 3: controlling the weights (1/2)

Divide-and-conquer algorithm:

- $\mathbf{P}^{(1)}$ for first $n/2$ points, evaluations $\mathbf{E}$, and weights $\vec{w}$
- $\mathbf{E}' =$ updated evaluations $\mathbf{P}^{(1)} \star \mathbf{E}$
- $\mathbf{P}^{(2)}$ for last $n/2$ points, evaluations $\mathbf{E}'$, and updated weights $\vec{w}'$
- return $\mathbf{P} = \mathbf{P}^{(2)}\mathbf{P}^{(1)}$

When $w_0 + \cdots + w_\ell \in \mathcal{O}(n)$,
complexity bound equation: $\mathcal{C}(\ell, n) = 2\mathcal{C}(\ell, n/2) + \tilde{\mathcal{O}}(\ell^{\omega-1} n)$??

Ingredient 3: controlling the weights (1/2)

Divide-and-conquer algorithm:

- $\mathbf{P}^{(1)}$ for first $n/2$ points, evaluations $\mathbf{E}$, and weights $\vec{w}$
- $\mathbf{E}'$ = updated evaluations $\mathbf{P}^{(1)} \star \mathbf{E}$
- $\mathbf{P}^{(2)}$ for last $n/2$ points, evaluations $\mathbf{E}'$, and updated weights $\vec{w}'$
- return $\mathbf{P} = \mathbf{P}^{(2)}\mathbf{P}^{(1)}$

When $w_0 + \cdots + w_\ell \in \mathcal{O}(n)$,
complexity bound equation: $\mathcal{C}(\ell, n) = 2\mathcal{C}(\ell, n/2) + \mathcal{O}^{\sim}(\ell^{\omega-1}n)$??

Problem: in the recursive call, n becomes $n/2$ but $\vec{w}$ is unchanged

$\Rightarrow$ Preserve the assumption on $\vec{w}$ throughout recursive calls

- Compute a minimal interpolation basis for $\vec{w} = \vec{0}$
- Recover a $\vec{w}$-minimal basis by change of weights

Ingredient 3: controlling the weights (2/2)

Change of weights

Input: $\vec{0}$-minimal basis $\mathbf{P}$, weights $\vec{w}$
Output: $\vec{w}$-minimal basis $\mathbf{R}$
Cost: $\mathcal{O}^{\sim}(\ell^{\omega-1}(|\vec{d}| + |\vec{w}|))$, where $\vec{d}$ = row degrees of $\mathbf{P}$

- $\mathbf{R}$ and $\mathbf{P}$ bases: $\mathbf{R} = \mathbf{U}\mathbf{P}$ for some unimodular $\mathbf{U}$
- $\mathbf{P}$ is $\vec{0}$-minimal $\Rightarrow$ $\mathbf{U}$ has small degree (predictable degree property):

$$\vec{d}\text{-weighted row degrees of } \mathbf{U} \quad \leqslant \quad \text{row degrees of } \mathbf{R}$$

Algorithm: Compute $\mathbf{U}, \mathbf{R}$ via a $\vec{d}, \vec{w}$-minimal nullspace basis
[Zhou - Labahn - Storjohann, 2012]

$$\begin{array}{c|c} \vec{d} & \vec{w} \\ \hline \mathbf{U} & \mathbf{R} \end{array} \begin{bmatrix} \mathbf{P} \\ \hline -\mathbf{Id} \end{bmatrix} = \mathbf{0}$$

Divide-and-conquer algorithm

Algorithm (Fast bivariate interpolation, $\vec{w} = \vec{0}$)

- If $\ell = n$:
  - compute $\mathbf{P}$ by full linearization
- Else
  - $\mathbf{P}^{(1)}$ for first $n/2$ points, evaluations $\mathbf{E}$, and weights $\vec{0}$
  - $\mathbf{E}'$ = updated evaluations $\mathbf{P}^{(1)} \star \mathbf{E}$
  - $\mathbf{P}^{(2)}$ for last $n/2$ points, evaluations $\mathbf{E}'$, and weights $\vec{0}$
  - $\mathbf{R}^{(2)}$ = Change weights of $\mathbf{P}^{(2)}$ to $\vec{w}$ = row-degrees$(\mathbf{P}^{(1)})$
  - return $\mathbf{P} = \mathbf{R}^{(2)}\mathbf{P}^{(1)}$

Base case $\ell = n$: $\mathbf{P}$ computed in $\mathcal{O}^{\sim}(\ell^{\omega})$

Because $\vec{w}$ satisfies $w_0 + \cdots + w_\ell \leqslant n$:

$$\mathbf{E}' = \mathbf{P}^{(1)} \star \mathbf{E}, \qquad \mathbf{P} = \mathbf{P}^{(2)}\mathbf{P}^{(1)}, \qquad \mathbf{R}^{(2)} \quad \text{computed in} \quad \mathcal{O}^{\sim}(\ell^{\omega-1} n)$$

Conclusion

Bivariate interpolation algorithm:

- Cost bound $\mathcal{O}^{\sim}(\ell^{\omega-1} n)$, deterministic
- When no assumption on the weights: cost $\mathcal{O}^{\sim}(\ell^{\omega-1} n + \ell^{\omega-1} |\vec{w}|)$
- Repetitions of the x_i are allowed
- Returns a minimal basis of interpolants

Conclusion

Bivariate interpolation algorithm:

- Cost bound $\mathcal{O}^{\sim}(\ell^{\omega-1} n)$, deterministic
- When no assumption on the weights: cost $\mathcal{O}^{\sim}(\ell^{\omega-1} n + \ell^{\omega-1} |\vec{w}|)$
- Repetitions of the x_i are allowed
- Returns a minimal basis of interpolants

General algorithm: fast solution to a unified problem, including

- Prescribed multiplicity on each point (x_i, y_i)
  (e.g. for soft-decoding of Reed-Solomon codes)
- Several variables $Q(X, Y_1, \ldots, Y_s)$
  (e.g. for folded Reed-Solomon codes,
  or Private Information Retrieval)
- simultaneous Hermite-Padé approximants; M-Padé approximants