Combinatorics of ideals of points: a Cerlienco-Mureddu-like approach for an iterative lex game.

Michela Ceria   Teo Mora

Polynomial Computer Algebra
St. Petersburg
16-21 April 2018

SOLVING

Given a 0-dim. ideal $I \in K[x_1, \ldots, x_n] =: \mathcal{P}$ **computes its roots** $\mathcal{R} \subset K^n$

SOLVING

Given a 0-dim. ideal $I \in K[x_1, \ldots, x_n] =: \mathcal{P}$ **computes its roots** $\mathcal{R} \subset K^n$

Trinks, Gianni-Kalkbrener, Auzinger-Stetter, Trianguar sets, RUR,...

SOLVING

Given a 0-dim. ideal $I \in K[x_1, \ldots, x_n] =: \mathcal{P}$ **computes its roots** $\mathcal{R} \subset K^n$

Trinks, Gianni-Kalkbrener, Auzinger-Stetter, Trianguar sets, RUR,...

v.s

BONDAGE

Given a finite set of points $\mathcal{R} \subset K^n$**, denoting** $I := \{f \in \mathcal{P} : f(P) = 0, P \in \mathcal{R}\}$ **compute the combinatorial structure of the algebra** $\mathcal{P}/I$ **(*escalier*)**

SOLVING

Given a 0-dim. ideal $I \in K[x_1, \ldots, x_n] =: \mathcal{P}$ **computes its roots** $\mathcal{R} \subset K^n$

Trinks, Gianni-Kalkbrener, Auzinger-Stetter, Trianguar sets, RUR,...

v.s

BONDAGE

Given a finite set of points $\mathcal{R} \subset K^n$**, denoting** $I := \{f \in \mathcal{P} : f(P) = 0, P \in \mathcal{R}\}$ **compute the combinatorial structure of the algebra** $\mathcal{P}/I$ **(*escalier*)**

Cerlienco-Mureddu, Möller, Lex Game, Cardinal, Auzinger-Stetter, Lundqvist

Once upon a time...

Cerlienco-Mureddu (1990)

Given a finite set of distinct points $\mathbf{X}$, compute the lexicographical Groebner escalier $\mathrm{N}(\mathbf{I}(\mathbf{X}))$ of the ideal of the points $I(\mathbf{X})$.

There is a $1-1$ correspondence between $\mathbf{X}$ and $\mathrm{N}(\mathbf{I}(\mathbf{X}))$.

The algorithm providing the correspondence is **iterative** on the points and **inductive** on the variables.
Complexity: n^2S^2 ($n =$number of variables, $S = |\mathbf{X}|$).

An improvement: the lex game

Felszeghy-Rath-Ronyai (2006)

By a clever use of tries (point trie - lex trie), they develop an algorithm that computes the lexicographical escalier in a more efficient way.

The algorithm **drops iterativity** for the sake of efficiency.
Complexity: $nS + S\min(S, nr)$
($r =$ maximal number of children of a node).

THE POINT TRIE

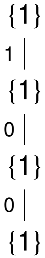
It is a trie representing the reciprocal relations among the coordinates of points.

same path from level 0 to level i = same $1, ..., i$ first coordinates

It is constructed **iteratively** on the points.

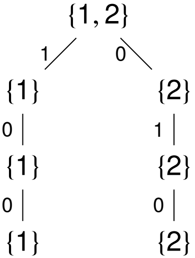
Example of point trie

$\mathbf{X} = \{(1,0,0),(0,1,0),(1,1,2),(1,0,3)\}$



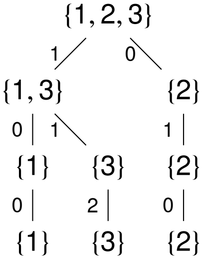
Example of point trie

$\mathbf{X} = \{(1,0,0),(0,1,0),(1,1,2),(1,0,3)\}$



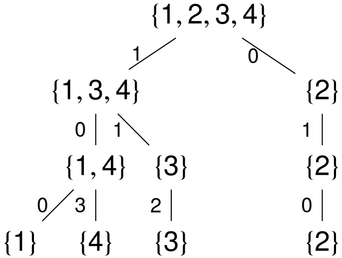
Example of point trie

$$\mathbf{X} = \{(1,0,0),(0,1,0),(1,1,2),(1,0,3)\}$$



Example of point trie

$$\mathbf{X} = \{(1,0,0),(0,1,0),(1,1,2),(1,0,3)\}$$



Another point of view: Moeller's algorithm

Moeller (1993)

Given an ordered finite set of distinct points $\mathbf{X} := \{P_1, ..., P_S\}$, find, for each ideal in Macaulay's chain $I_i := I(\{P_1, ..., P_i\})$ $1 \leq i \leq S$, the **escalier** $\mathrm{N}(\mathrm{I_i})$ and a **separator family** for the points (with some more steps you also get the Groebner bases).

Another point of view: Moeller's algorithm

Moeller (1993)

Given an ordered finite set of distinct points $\mathbf{X} := \{P_1, ..., P_S\}$, find, for each ideal in Macaulay's chain $I_i := I(\{P_1, ..., P_i\})$ $1 \leq i \leq S$, the **escalier** $\mathrm{N}(\mathrm{I}_\mathrm{i})$ and a **separator family** for the points (with some more steps you also get the Groebner bases).

→ subsuming FGLM and with the same complexity
→ iterative on points
→ the result (for Lex) is exactly that of Cerlienco-Mureddu algorithm.

Another point of view: Moeller's algorithm

Moeller (1993)

Given an ordered finite set of distinct points $\mathbf{X} := \{P_1, ..., P_S\}$, find, for each ideal in Macaulay's chain $I_i := I(\{P_1, ..., P_i\})$ $1 \leq i \leq S$, the **escalier** $\mathrm{N}(\mathrm{I}_\mathrm{i})$ and a **separator family** for the points (with some more steps you also get the Groebner bases).

→ subsuming FGLM and with the same complexity
→ iterative on points
→ the result (for Lex) is exactly that of Cerlienco-Mureddu algorithm.

My 2 cents...

If we evaluate each polynomial at each point, with the same complexity, we can get more information, such as Groebner representation, separator polynomials and Auzinger-Stetter matrices.

Can we construct a new algorithm, that is

iterative as Cerlienco-Mureddu

and has the

same complexity as the lex game?

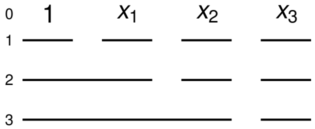
Bar Codes (Ceria)

Definition

A Bar Code B is a picture composed by segments, called *bars*, superimposed in horizontal rows, which satisfies

A. $\forall i, j,\ 1 \leq i \leq n-1,\ 1 \leq j \leq \mu(i),\ \exists! \bar{j} \in \{1, \ldots, \mu(i+1)\}$ s.t. $B_{\bar{j}}^{(i+1)}$ lies under $B_j^{(i)}$

B. $\forall i_1,\ i_2 \in \{1, \ldots, n\},\ \sum_{j_1=1}^{\mu(i_1)} l_1(B_{j_1}^{(i_1)}) = \sum_{j_2=1}^{\mu(i_2)} l_1(B_{j_2}^{(i_2)})$; we will then say that *all the rows have the same length*.

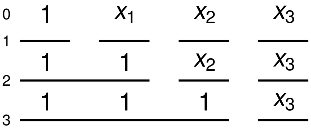


ASSOCIATING MONOMIALS TO BARS

For $t = x_1^{\gamma_1} \cdots x_n^{\gamma_n} \in \mathcal{T}$, $\forall i \in \{1, ..., n\}$, $\pi^i(t) := x_i^{\gamma_i} \cdots x_n^{\gamma_n}$;
$M = \{t_1, ..., t_m\} \subset \mathcal{T}$, $M^{[i]} := \pi^i(M)$, $\underline{M}$, $\underline{M}^{[i]}$ increasingly ordered w.r.t. Lex.

$$\mathcal{M} := \begin{pmatrix} \pi^1(t_1) & ... & \pi^1(t_m) \\ \pi^2(t_1) & ... & \pi^2(t_m) \\ \vdots & & \vdots \\ \pi^n(t_1) & ... & \pi^n(t_m) \end{pmatrix}$$

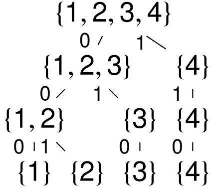
Bar Code: connecting with a bar the repeated terms

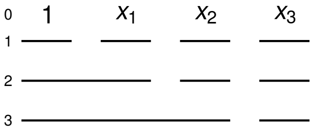


Bar Code and point trie

We can see the Bar Code as a **point trie** by taking as points the **exponents' lists** (→ **Macaulay's trick**) for the given terms.

For $M = \{1, x_1, x_2, x_3\} \subset \mathbf{k}[x_1, x_2, x_3]$, we have $\mathfrak{M} = \{p_1 = (0,0,0), p_2 = (0,0,1), p_3 = (0,1,0), p_4 = (1,0,0)\}$, so we have





Several applications of Bar Code

Bar Codes are useful to study properties of monomial/polynomial ideals:

- counting **(strongly) stable ideals**;

Several applications of Bar Code

Bar Codes are useful to study properties of monomial/polynomial ideals:

- counting **(strongly) stable ideals**;
- computing **Pommaret bases** via interpolation;
- computing **Janet multiplicative variables**
- detect **completeness**;
- find variables' orderings which make a set of terms **Janet-complete**

Several applications of Bar Code

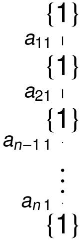
Bar Codes are useful to study properties of monomial/polynomial ideals:

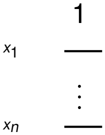
- counting **(strongly) stable ideals**;
- computing **Pommaret bases** via interpolation;
- computing **Janet multiplicative variables**
- detect **completeness**;
- find variables' orderings which make a set of terms **Janet-complete**
- Bar Code, point trie vs. **Janet tree**

Our algorithm

Base step

$|\mathbf{X}| = N = 1$: set $\mathrm{N}(1) = \{1\}$ and construct the point trie $T(P_1) = \mathfrak{T}(\mathbf{X})$ and the Bar Code $\mathrm{B}(1)$ displayed below. The output is stored in the matrix M.





$$M = \begin{bmatrix} & \mathbf{x_n} & \mathbf{x_{n-1}} & \dots & \mathbf{x_1} \\ & \downarrow & \downarrow & \dots & \downarrow \\ \mathbf{1} \rightarrow & 0 & 0 & \dots & 0 \end{bmatrix}$$

OUR ALGORITHM: $|\mathbf{X}| = N > 1$

- update the point trie: forking level $s = \sigma$**-value**; leftmost label of the rightmost sibling $l = \sigma$**-antecedent**;
- find the s–bar of t_l: $\mathrm{B}_j^{(s)}$

Information on t_N:

- it lies over $\mathrm{B}_1^{(n)}, \mathrm{B}_1^{(n-1)}, ..., \mathrm{B}_1^{(s+1)}$ so t_N lies over the first $n, ..., s+1$ bars, i.e. $a_{s+1}^{(N)} = ... = a_n^{(N)} = 0$, so $x_n, ..., x_{s+1} \nmid t_N$;
- it should lie over $\mathrm{B}_{j+1}^{(s)}$: $a_s^{(N)} = a_s^{(l)} + 1$.

Our algorithm: $|\mathbf{X}| = N > 1$

We test whether $B_{j+1}^{(s)}$ lies over $B_1^{(n)}, B_1^{(n-1)}, ..., B_1^{(s+1)}$; two possible cases

A. **NO**: we construct a new *s*-bar of lenght one over $B_1^{(n)}, B_1^{(n-1)}, ..., B_1^{(s+1)}$, on the right of $B_j^{(s)}$, we label it as $B_{j+1}^{(s)}$ and we construct a $1, ..., s-1$ bar of length 1 over $B_{j+1}^{(s)}$:
$t_N = x_s^{j+2}$; store the output in the N-th row of M.

B. **YES**: we must continue, repeating the procedure

Our algorithm: $|\mathbf{X}| = N > 1$

- restrict the point trie to the points whose corresponding terms lie over $\mathrm{B}_{j+1}^{(s)}$. The set containing these points is denoted by S and is obtained reading $\mathrm{B}_{j+1}^{(s)}$. More precisely, $S = \psi(\mathrm{B}_{j+1}^{(s)})$, where

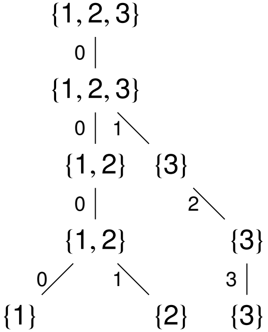
$$\psi : \mathrm{B} \to \mathcal{T}$$

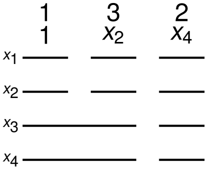
  is the function sending each 1-bar $\mathrm{B}_l^{(1)}$ in the term t_l over it and, inductively, for $1 < u \leq n$, $\psi(\mathrm{B}_h^{(u)}) = \bigcup_{B \text{ over } \mathrm{B}_h^{(u)}} \psi(B)$
- read P_N's path, from level $s - 1$ to level 1, looking for the first forking level w.r.t. S (σ-value/σ-antecedent as before).
- **repeat** the test

The procedure is repeated until we get to the 1-bars or if in the decision step we get case a.

Example

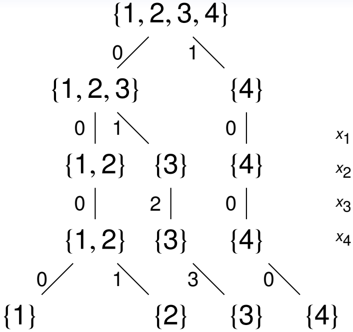
$$\mathbf{X} = \{(0,0,0,0),(0,0,0,1),(0,1,2,3),(1,0,0,0),(1,0,0,1)\}$$

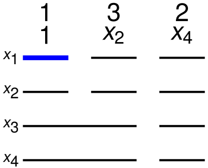




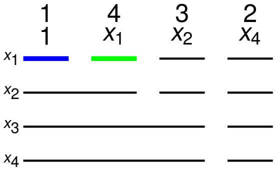
$$M = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

For $P_4 = (1,0,0,0)$, $s = 1$, $l = 1$; B the blue bar

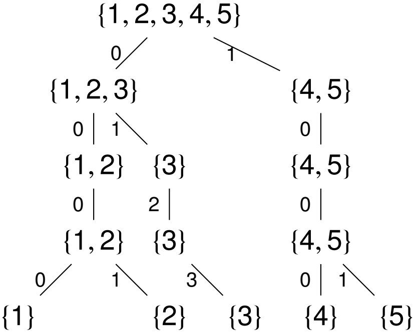




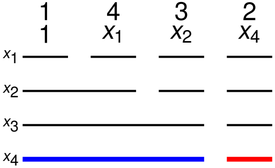
There is no 1-bar on the right of B, lying over $B_1^{(4)}, B_1^{(3)}, B_1^{(2)}$:



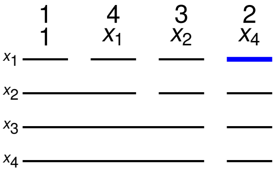
$P_5 = (1, 0, 0, 1)$; $s = 4$ $l = 4$:



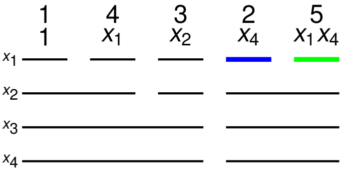
$B = \mathrm{B}_1^{(4)}$; $B' = \mathrm{B}_2^{(4)}$, $S = \{P_2\}$.



The fork with P_2 happens at $s = 1$ and the σ-antecedent is P_l, for $l = 2$, so $B = \mathrm{B}_4^{(1)}$.



Since B' still does not exist, we create it



$\mathrm{N} = \{1, \mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_4, \mathrm{x}_1\mathrm{x}_4\}$

Separator polynomials

A **family of separators** for a finite set $\mathbf{X} = \{P_1, ..., P_N\}$ of distinct points is a set $Q = \{Q_1, ..., Q_N\}$ s.t.
$Q_i(P_i) = 1$ and $Q_i(P_j) = 0$, for each $1 \leq i, j \leq N$, $i \neq j$.

$\mathbf{X} = \{P_1, ..., P_N\}$, with $P_i := (a_{1,i}, ..., a_{n,i})$, $i = 1, ..., N$, we denote by $C = (c_{i,j})$ the **witness matrix** i.e. the (symmetric) matrix s.t., for $i, j = 1, ..., N$, $c_{i,j} = 0$ if $i = j$ and if $i \neq j$,
$c_{i,j} = \min\{h : 1 \leq h \leq n \text{ s.t. } a_{h,i} \neq a_{h,j}\}$.

Building blocks:

$$p_{i,j}^{[c_{i,j}]} = \frac{x_{c_{i,j}} - a_{c_{i,j},j}}{a_{c_{i,j},i} - a_{c_{i,j},j}}$$

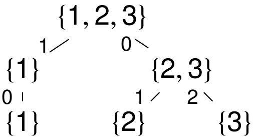
$|\mathbf{X}| = 1$: $Q_1 = 1$. $Q_1, ..., Q_{N-1}$ associated to $\{P_1, ..., P_{N-1}\}$: P_N?
We see now how to get the new separators $Q'_1, ..., Q'_N$ for **X**.

- Set $Q'_N = 1$.
- $\forall j = 1, ..., n$, we take the node $v_{j,u}$ of N
- for each sibling $v_{j,u'}$ of $v_{j,u}$, we pick an element $\bar{i}$ of its label and set $Q'_N = Q'_N p^{[j]}_{N,\bar{i}}$.
- if $v_{j,u}$ is labelled only by N, then, for each sibling $v_{j,u'}$, for each element i in its label we set $Q'_i = Q_i p^{[j]}_{i,N}$.

Once concluded this procedure, if a separator Q_h, $1 \leq h \leq N$ has **not** been involved in the above steps, we set $Q'_h = Q_h$, getting a family of separators $\{Q'_1, ..., Q'_N\}$ for $\mathbf{X} = \{P_1, ..., P_N\}$.
Complexity of a single iterative round: $\mathcal{O}(\min(N, nr))$.

Example

$\mathbf{X} = \{P_1 = (1,0), P_2 = (0,1), P_3 = (0,2)\}$



In the first step, we set $Q_1'' = 1$; then, adding P_2 to the trie we set $Q_2' = p_{2,1}^{[1]} = -(x-1)$ and we modify also Q_1'', setting $Q_1' = Q_1'' p_{1,2}^{[1]} = x$, since, when P_3 is still not in the trie, the node $v_{1,2}$, has $V_{1,2} = \{2\}$. So, w.r.t. $\{P_1, P_2\}$, we have $Q_1' = x$, $Q_2' = -(x-1)$. Finally, we add P_3. This way, $Q_3 = p_{3,1}^{[1]} p_{3,2}^{[2]} = -(x-1)(y-1)$ and since $V_{2,3} = \{3\}$, $Q_2 = Q_2' p_{2,3}^{[2]} = (x-1)(y-2)$. Finally, we have

$$Q_1 = x; \; Q_2 = (x-1)(y-2); \; Q_3 = -(x-1)(y-1).$$

Comparisons?

$$Q_1 = x;\ Q_2 = (x-1)(y-2);\ Q_3 = -(x-1)(y-1).$$

From Lex game

$$Q_1 = \frac{1}{2}x(y-1)(y-2);\ Q_2 = y(x-1)(y-2);\ Q_3 = -\frac{1}{2}(x-1)y(y-1),$$

Lundqvist

$$Q_1 = x^2;\ Q_2 = (x-1)(y-2);\ Q_3 = -(x-1)(y-1).$$

Moeller

$$Q_1 = x;\ Q_2 = 2-2x-y;\ Q_3 = x+y-1.$$

Auzinger-Stetter

$I \triangleleft \mathbf{k}[x_1, ..., x_n]$ zerodimensional ideal; $A := \mathbf{k}[x_1, ..., x_n]/I$. $\forall f \in A$, $\Phi_f : A \to A$ multiplication by f in A and, fixed a basis $B = \{[b_1], \ldots, [b_m]\}$ for A, $A_f = (a_{ij})$ so that

$$[b_i f] = \sum_j a_{ij}[b_j], \forall i.$$

We call **Auzinger-Stetter matrices** associated to I, the matrices A_{x_i}, $i = 1, ..., n$, defined w.r.t. the basis given by the lex escalier of **I**.

Lundqvist

$\mathbf{X} = \{P_1, ..., P_N\}$, $I := I(\mathbf{X}) \triangleleft \mathbf{k}[x_1, ..., x_n]$; $\mathrm{N} = \{\mathrm{t}_1, ..., \mathrm{t}_\mathrm{N}\} \subset \mathbf{k}[\mathrm{x}_1, ..., \mathrm{x}_\mathrm{n}]$ s.t. $[\mathrm{N}] = \{[\mathrm{t}_1], ..., [\mathrm{t}_\mathrm{N}]\}$ is a basis for $A := \mathbf{k}[x_1, ..., x_n]/I$. Then, for each $f \in \mathbf{k}[x_1, ..., x_n]$ we have

$$\mathbf{Nf}(f, \mathrm{N}) = (\mathrm{t}_1, ..., \mathrm{t}_\mathrm{N})(\mathrm{N}(\mathbf{X})^{-1})^\mathrm{t}(\mathrm{f}(\mathrm{P}_1), ..., \mathrm{f}(\mathrm{P}_\mathrm{N}))^\mathrm{t},$$

where $\mathrm{Nf}(f, \mathrm{N})$ is the normal form of f w.r.t. N.

Notation

- $A_{x_h} := \left(a_{li}^{(h)}\right)_{li}$, $1 \leq h \leq n$, $1 \leq l, i \leq N$, the Auzinger-Stetter matrices w.r.t. $\mathrm{N}(\mathbf{I})$;
- $B := \mathrm{N}(\mathbf{I})(\mathbf{X}) := (\mathrm{b}_{\mathrm{lj}})_{\mathrm{lj}}$, $1 \leq l, j \leq N$, $b_{lj} := t_l(P_j)$;
- $C := (c_{ji})_{ji}$, $1 \leq j, i \leq N$, the inverse matrix of B, i.e. $C := B^{-1}$
- $D^{(h)} := \left(d_{lj}^{(h)}\right)_{lj}$, $1 \leq h \leq n$, $1 \leq l, j \leq N$, $d_{lj}^{(h)} := \alpha_h^{(j)} t_l(P_j)$, the evaluation of $x_h t_l$ at the point P_j.

Lundqvist

$\mathbf{X} = \{P_1, ..., P_N\}$, $I := I(\mathbf{X}) \triangleleft \mathbf{k}[x_1, ..., x_n]$; $\mathrm{N} = \{\mathrm{t}_1, ..., \mathrm{t}_\mathrm{N}\} \subset \mathbf{k}[\mathrm{x}_1, ..., \mathrm{x}_\mathrm{n}]$ s.t. $[\mathrm{N}] = \{[\mathrm{t}_1], ..., [\mathrm{t}_\mathrm{N}]\}$ is a basis for $A := \mathbf{k}[x_1, ..., x_n]/I$. Then, for each $f \in \mathbf{k}[x_1, ..., x_n]$ we have

$$\mathbf{Nf}(f, \mathrm{N}) = (\mathrm{t}_1, ..., \mathrm{t}_\mathrm{N})(\mathrm{N}(\mathbf{X})^{-1})^\mathrm{t}(\mathrm{f}(\mathrm{P}_1), ..., \mathrm{f}(\mathrm{P}_\mathrm{N}))^\mathrm{t},$$

where $\mathrm{Nf}(f, \mathrm{N})$ is the normal form of f w.r.t. N.

For $1 \leq l \leq N$, the l-th row of A_{x_h} is the normal form of $x_h t_l$:

$$\mathrm{Nf}(x_h t_l, \mathrm{N(I)}) = \sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{a}_{\mathrm{li}} \mathrm{t}_\mathrm{i} = (\mathrm{t}_1, ..., \mathrm{t}_\mathrm{N})\mathrm{C}^\mathrm{t}(\mathrm{x}_\mathrm{h}\mathrm{t}_\mathrm{l}(\mathrm{P}_1), ..., \mathrm{x}_\mathrm{h}\mathrm{t}_\mathrm{l}(\mathrm{P}_\mathrm{N}))^\mathrm{t} =$$

$$(t_1, ..., t_N)C^t(d_{l1}^{(h)}, ..., d_{lN}^{(h)})^t = \sum_i \left(\sum_{j=1}^{N} d_{lj}^{(h)} c_{ji} \right) t_i.$$

This trivially implies that $A_{x_h} = D^{(h)}C = D^{(h)}B^{-1}$.

Computing B^{-1}.

Gaussian column-reduction of $\left(\frac{B}{I}\right)$.

At each step

$$\left(\frac{B}{I}\right) \to \left(\frac{E}{F}\right)$$

it holds $E = BF$ So $E = 1 \implies F = B^{-1}$.

We border B obtaining $B' := \begin{pmatrix} & B & & b_{1N} \\ & & & \vdots \\ & & & b_{N-1N} \\ b_{N1} & \cdots & b_{NN-1} & b_{NN} \end{pmatrix}$ and

properly border $\left(\frac{I}{C}\right)$ as $\left(\begin{array}{cccc} & I & & b_{1N} \\ & & & \vdots \\ & & & b_{N-1N} \\ f_{N1} & \cdots & f_{NN-1} & b_{NN} \\ \hline & & & 0 \\ & C & & \vdots \\ & & & 0 \\ 0 & \cdots 0 & & 1 \end{array}\right)$ where

$$(f_{N1} \cdots f_{NN-1}) = (b_{N1} \cdots b_{NN-1})\, C$$

For each point i we know the last σ-value $s(i)$ and σ-antecedent $P_{l(i)}$ $\boxed{t_i = x_{s(i)} t_{l(i)}}$

We perform the following computations

- $b_{1N} := 1$
- for $i = 2 \cdots N-1$, $b_{iN} := b_{l(i)N} a_{s(i)N}$
- for $j = 1 \cdots N$, $b_{Nj} := b_{l(N)j} a_{s(N)N}$

  border B

For each point i we know the last σ-value $s(i)$ and σ-antecedent $P_{l(i)}$ $\boxed{t_i = x_{s(i)} t_{l(i)}}$

We perform the following computations

- $b_{1N} := 1$
- for $i = 2 \cdots N-1$, $b_{iN} := b_{l(i)N} a_{s(i)N}$
- for $j = 1 \cdots N$, $b_{Nj} := b_{l(N)j} a_{s(N)N}$
  border B
- for $i = 1 \cdots N-1$, $1 \leq h \leq n$, $d_{iN}^{(h)} := d_{l(i)N}^{(h)} a_{s(i)N}$
- for $j = 1 \cdots N$, $1 \leq h \leq n$, $d_{Nj}^{(h)} := d_{l(N)j}^{(h)} a_{s(N)N}$
  border D

For each point i we know the last σ-value $s(i)$ and σ-antecedent $P_{l(i)}$ $\boxed{t_i = x_{s(i)} t_{l(i)}}$

We perform the following computations

- $b_{1N} := 1$
- for $i = 2 \cdots N - 1, b_{iN} := b_{l(i)N} a_{s(i)N}$
- for $j = 1 \cdots N, b_{Nj} := b_{l(N)j} a_{s(N)N}$
  border B
- for $i = 1 \cdots N - 1, 1 \leq h \leq n, d^{(h)}_{iN} := d^{(h)}_{l(i)N} a_{s(i)N}$
- for $j = 1 \cdots N, 1 \leq h \leq n, d^{(h)}_{Nj} := d^{(h)}_{l(N)j} a_{s(N)N}$
  border D
- for $i = 1 \cdots N - 1, f_{Ni} := \sum_j b_{Nj} c_{ji}$
  border C

- for $i = 1 \cdots N - 1, g_{iN} := \sum_j c_{ij} b_{jN}$
- $h_{NN} := f_{NN} - \sum_j f_{Nj} b_{jN}$
- $c_{iN} := \frac{g_{iN}}{h_{NN}}, 1 \le i \le N$
- $c_{ij} := c'_{ij} - f_{Nj} c_{iN} 1 \le i \le N, 1 \le j < N$

  computing $C = B^{-1}$

- for $i = 1 \cdots N-1$, $g_{iN} := \sum_j c_{ij} b_{jN}$
- $h_{NN} := f_{NN} - \sum_j f_{Nj} b_{jN}$
- $c_{iN} := \frac{g_{iN}}{h_{NN}}, 1 \leq i \leq N$
- $c_{ij} := c'_{ij} - f_{Nj} c_{iN} 1 \leq i \leq N, 1 \leq j < N$

  computing $C = B^{-1}$
- for $1 \leq l < N, 1 \leq h \leq n$, $a_{lN}^{(h)} := \sum_i d_{li}^{(h)} c_{iN}$,
- for $1 \leq j < N, 1 \leq h \leq n$, $a_{Nj}^{(h)} := \sum_i d_{Ni}^{(h)} c_{ij}$,

  $A^{(h)} = CD^{(h)}$

Example

For $\mathbf{X} = \{P_1 = (1,0), P_2 = (0,1), P_3 = (0,2)\}$:

P_1: $B = C = 1$ and $D^{(1)} = (1) = A_x$, $D^{(2)} = (0) = A_y$.

P_2: $B'' = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}, \left(\frac{I''}{C''}\right) = \left(\begin{array}{cc} 1 & 1 \\ 1 & 0 \\ \hline 1 & 0 \\ 0 & 1 \end{array}\right) \to B = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}, C = B^{-1} = \begin{pmatrix} 0 & 1 \\ 1 & -1 \end{pmatrix}$

$$D^{(1)} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}, A_x = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}, D^{(2)} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, A_y = \begin{pmatrix} 1 & -1 \\ 0 & 0 \end{pmatrix}.$$

P_3: $B'' = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 2 \end{pmatrix}, C'' = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, I'' = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & -1 & 2 \end{pmatrix} \to$

$$C = B^{-1} = \begin{pmatrix} 0 & 1 & 0 \\ 2 & -2 & -1 \\ -1 & 1 & 1 \end{pmatrix}. D^{(1)} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, A_x = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

$$D^{(2)} = \begin{pmatrix} 0 & 1 & 2 \\ 0 & 0 & 0 \\ 0 & 1 & 4 \end{pmatrix}, A_y = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 2 & -2 & 3 \end{pmatrix}.$$

Thank you for your attention!