Algorithms for Fast Linear System Solving and Rank Profile Computation

Shiyun (Sophia) Yang

Symbolic Computation Group
David R. Cheriton School of Computer Science
University of Waterloo

June 27, 2014

Rank profile

Given $A \in \mathsf{K}^{n \times m}$ over a finite field K.

- RankProfile: Compute the rank r and the lexicographically minimal lists $[i_1, i_2, \dots, i_r]$ of row indices of A such that these rows of A are linearly independent.

Pivot locations in the column echelon form:

$$\begin{bmatrix} \circledast & & & & & & & & \\ * & & & & & & & & \\ * & \circledast & & & & & & & \\ * & * & \circledast & & & & & & \\ * & * & * & \circledast & & & & & \\ * & * & * & * & \circledast & & & & \end{bmatrix}$$

Row rank profile: $[1, 3, 4, 5, 6]$

Applications: Gröbner basis computations, computational number theory, etc.

Linear system

- LinSys: Given $b \in \mathsf{K}^{n \times 1}$, compute a particular solution $x \in \mathsf{K}^{m \times 1}$ to $Ax = b$,

Consistent: x can be read off from the last column in the reduced row echelon form.

$$\left[\begin{array}{ccccccccc||c} 1 & * & & & * & * & & * & & x_1 \\ & & 1 & & * & * & & * & & x_3 \\ & & & 1 & * & * & & * & & x_4 \\ & & & & & & 1 & * & & x_7 \\ & & & & & & & & 1 & x_9 \\ & & & & & & & & & \end{array}\right]$$

Solution vector $x^T = \left[\begin{array}{ccccccccc} x_1 & 0 & x_3 & x_4 & 0 & 0 & x_7 & 0 & x_9 \end{array}\right]$

Linear system

- LinSys: Given $b \in \mathsf{K}^{n \times 1}$, compute a particular solution $x \in \mathsf{K}^{m \times 1}$ to $Ax = b$, or a certificate of inconsistency[1]: a row vector $u \in \mathsf{K}^{1 \times n}$ such that $uA = 0$ and $ub \neq 0$.

Inconsistent: u can be read off from the last row of the transformation matrix.

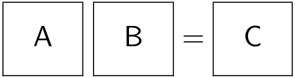
Transform $\left[\; A \;\|\; b \;|\; I \;\right] \Rightarrow \left[\; R \;\|\; * \;|\; U \;\right]$

$$\left[\begin{array}{ccccccccc||c|cccccc}
\circledast & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * \\
 & & \circledast & * & * & * & * & * & * & * & * & * & * & * & * & * \\
 & & & \circledast & * & * & * & * & * & * & * & * & * & * & * & * \\
 & & & & & & \circledast & * & * & * & * & * & * & * & * & * \\
 & & & & & & & & \circledast & * & * & * & * & * & * & * \\
 & & & & & & & & & \circledast & \boxed{*} & \boxed{*} & \boxed{*} & \boxed{*} & \boxed{*} & \boxed{*}
\end{array}\right]$$

[1]Giesbrecht, Lobo & Saunders (1998)

Notation

- **K :** a finite field.
- **Cost model:** counting scalar field operations of type $\{+, -, \times, /\}$ from K.
- $\boldsymbol{n}$ **:** row dimension of A.
- $\boldsymbol{m}$ **:** column dimension of A.
- $\boldsymbol{r}$ **:** rank of A.
- $\boldsymbol{\omega}$ **:** exponent of matrix multiplication, $2 < \omega \leq 3$. Multiply two $n \times n$ matrices

  in time $O(n^{\omega})$.
- $\boldsymbol{o(1)}$: hides log factors in the cost estimates.

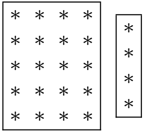
$$O(n \log n \log\log n) = (n)^{1+o(1)}$$

.

Notation

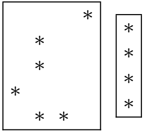
- **$|A|$:** number of nonzero entries of A.

  Dense matrices: $|A| \in \Theta(nm)$

  $cost : O(nm)$

  Sparse matrices: $|A| \in o(nm)$

  $cost : O(|A|)$

  In this talk, we assume $|A| \geq \max(n, m)$.

Notation

- $\boldsymbol{\mu(A)}$ **:** time required to multiply a vector by A in black box approach[2]. It follows a different cost model, in this talk, we have $\mu(A) \in O(|A|)$.

[2] Kaltofen & Saunders (1991)

Previous results for RANK and RANKPROFILE

Deterministic algorithm

- Dumas, Gautier & Pernet (2013); Jeannerod, Pernet & Storjohann (2013)
  - $O(nmr^{\omega-2}) \longleftarrow$ RANKPROFILE

Monte Carlo randomized algorithms

- Kaltofen & Saunders (1991); Chen, Eberly, Kaltofen, Saunders, Turner & Villard (2002)
  - $(r^\omega + nm)^{1+o(1)} \longleftarrow$ RANK
- Wiedemann (1986); Kaltofen & Saunders (1991); Eberly (2003)
  - $(\mu(A)\, r)^{1+o(1)}$ or $(|A|\, r)^{1+o(1)} \longleftarrow$ RANK
- Cheung, Kwok & Lau (2013)
  - $(r^\omega + |A|)^{1+o(1)} \longleftarrow$ RANK
  - computes a list of r linearly independent columns

This talk

- a Monte Carlo algorithm: $(r^\omega + |A|)^{1+o(1)} \longleftarrow$ RANKPROFILE

Previous results for LinSys

Deterministic algorithms

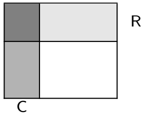
- Dumas, Gautier & Pernet (2013); Jeannerod, Pernet & Storjohann (2013)
  - $O(nmr^{\omega-2})$
- Mulders & Storjohann (2000)
  - $O((n+m)r^2)$

Las Vegas randomized algorithms

- Giesbrecht, Lobo & Saunders (1999); Eberly (2003)
  - $(\mu(A)\,r)^{1+o(1)}$ or $(|A|\,r)^{1+o(1)}$
- Cheung, Kwok & Lau (2013)
  - $(r^\omega + |A|)^{1+o(1)}$

This talk

- a Las Vegas algorithm: $2r^3 + (r^2 + n + m + |R| + |C|)^{1+o(1)}$
- examines at most $r+1$ rows and r columns of A:



Comparison with previous results for LinSys

For a class of input matrices $A \in \mathsf{K}^{n \times n}$ that have

- at most $O(n^{2/3})$ nonzero entries per row and column, and
- $r \in O(n^{1/3})$.

Black box approach:

- $O(n)$ additional space
- $(n^2)^{1+o(1)}$ time

Cheung, Kwok & Lau (2013):

- $O(n^{5/3})$ additional space
- $(n^{5/3})^{1+o(1)}$ time

Our approach:

- $O(n)$ additional space
- $(n)^{1+o(1)}$ time

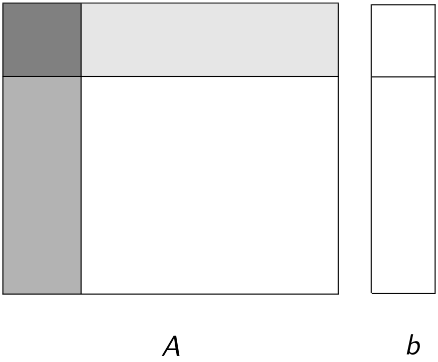
Outline

- Oracle linear solving [Mulders & Storjohann (2000)]
  - application to RankProfile
- Linear independence oracles
  - application to LinSys
  - application to RankProfile
- A relaxed algorithm for online matrix inversion
  - application to RankProfile

Oracle linear solving

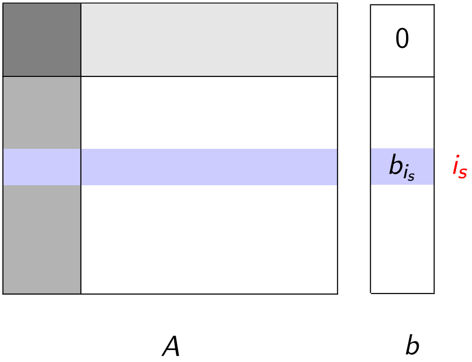
Mulders & Storjohann (2000)

At stage s



Oracle linear solving
Mulders & Storjohann (2000)

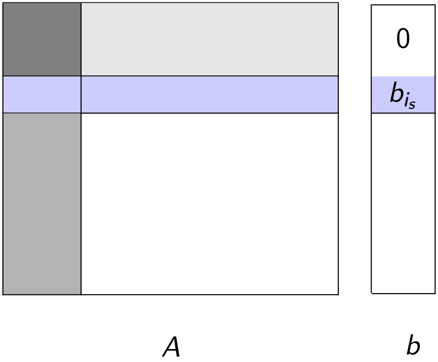
At stage s



Oracle linear solving

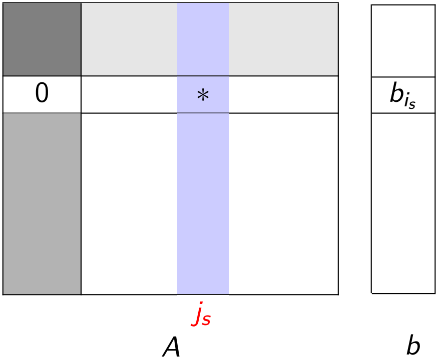
Mulders & Storjohann (2000)

At stage s



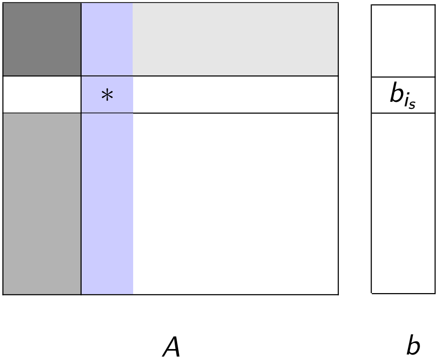
Oracle linear solving
Mulders & Storjohann (2000)

At stage s



Oracle linear solving
Mulders & Storjohann (2000)

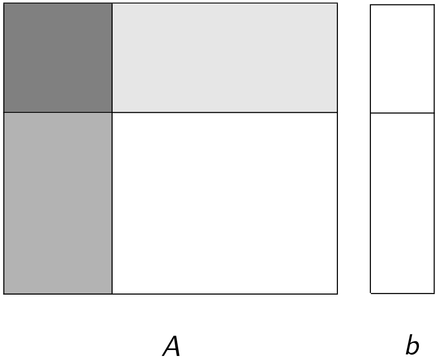
At stage s



Oracle linear solving

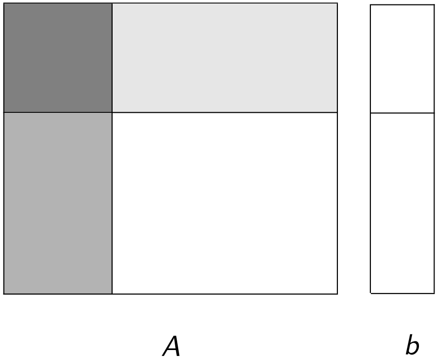
Mulders & Storjohann (2000)

Repeat for stage $s + 1$



Oracle linear solving
Mulders & Storjohann (2000)

Repeat for stage $s + 1$



Terminates with $s \leq r$, overall cost $O((n+m)r^2)$ for LinSys.

Contribution 1: Randomized rank profiles

- If b is chosen uniformly and randomly sampled from the column space of A, that is,
  - choose a $w \in \mathsf{K}^{m \times 1}$ uniformly and randomly, and compute $b = Aw$,

  then $[i_1, i_2, \ldots, i_s]$ is the row rank profile of A with probability at least $(1 - 1/\#\mathsf{K})^r$.
- This gives a Monte Carlo algorithm for RankProfile in time $O((n + m)r^2)$.
- The ideas of our improved algorithms for LinSys and RankProfile are the same, except that b is given explicitly for LinSys.
- We focus on algorithms for RankProfile in this presentation.

Goals

Starting complexity: $O((n+m)r^2)$

Goals

1. Decouple the cubic part of the time complexity:

$$2r^3 + (r^2 + nm)^{1+o(1)}$$

2. Exploit possible sparsity of A:

$$2r^3 + (r^2 + |A|)^{1+o(1)}$$

3. Incorporate fast matrix multiplication:

$$(r^\omega + |A|)^{1+o(1)}$$

Goals

Starting complexity: $O((n+m)r^2)$

Goals

1. Decouple the cubic part of the time complexity:

$$2r^3 + (r^2 + nm)^{1+o(1)}$$

2. Exploit possible sparsity of A:

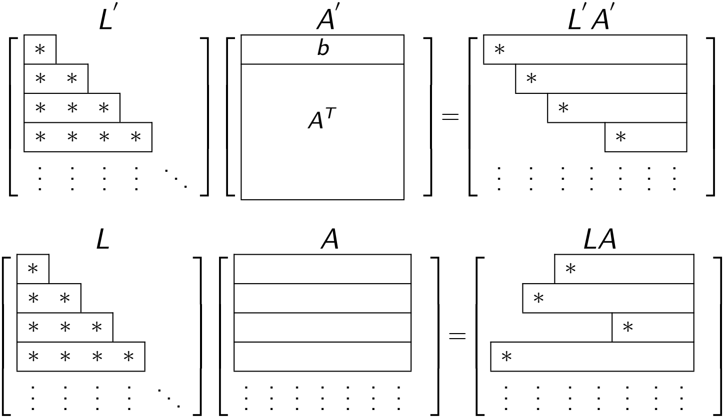
$$2r^3 + (r^2 + |A|)^{1+o(1)}$$

3. Incorporate fast matrix multiplication:

$$(r^\omega + |A|)^{1+o(1)}$$

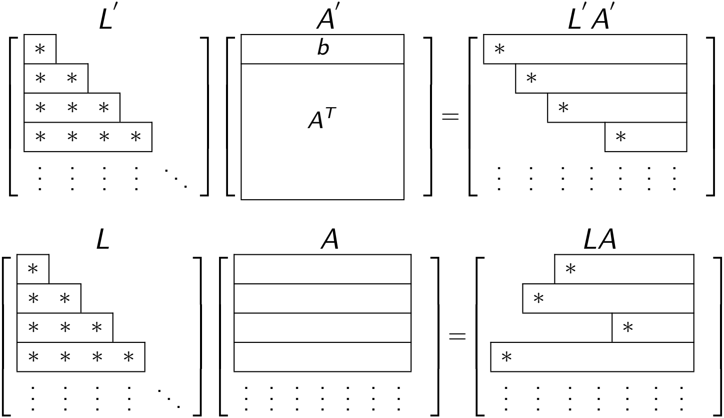
Linear independence oracles

After some simplifications, the steps in the oracle solver algorithm to find i_s and j_s "boil down" to the problem of finding the pivot locations in $L^{'}A^{'}$ and LA. $L^{'}$ and L are coming from Gaussian elimination.



Linear independence oracles

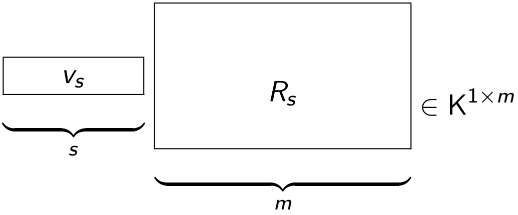
After some simplifications, the steps in the oracle solver algorithm to find i_s and j_s "boil down" to the problem of finding the pivot locations in $L^{'}A^{'}$ and LA. $L^{'}$ and L are coming from Gaussian elimination.



Computing $L^{'}A^{'}$ and LA directly are expensive.

Linear independence oracles

At each stage s, finding j_s is equivalent to finding the first nonzero entry of



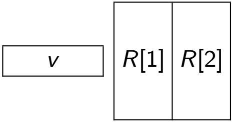
Computing $v_s R_s$ explicitly: $O(sm)$ field operations.

Use *linear independence oracle*: $O(s \log m)$ field operations.

Linear independence oracles

Example. $v \in \mathrm{K}^{1\times s}$, $R \in \mathrm{K}^{s\times 2}$.

- Require 2 dot products to determine if $vR = 0$.



- Idea: take random linear combination of columns of R. Choose $\alpha \in \mathrm{K}$ uniformly and randomly and compute

$$R_{1\sim 2} = R[1] + \alpha R[2]$$

  Require 1 dot product to determine if $vR = 0$ with high probability.

  - $vR_{1\sim 2} \neq 0 \implies vR \neq 0$.
  - If α well chosen, $vR_{1\sim 2} = 0 \implies vR = 0$.
  - α is good with probability $(1 - 1/\#\mathrm{K})$.

Linear independence oracles

T is a *linear independence oracle* for R based on $\alpha_1, \ldots, \alpha_{m-1}$.

$\alpha_1, \ldots, \alpha_{m-1} \in$ K be chosen uniformly and randomly.

$R_{a\sim b}$: a vector that is a linear combination of $R[a], R[a+1], \ldots, R[b]$.

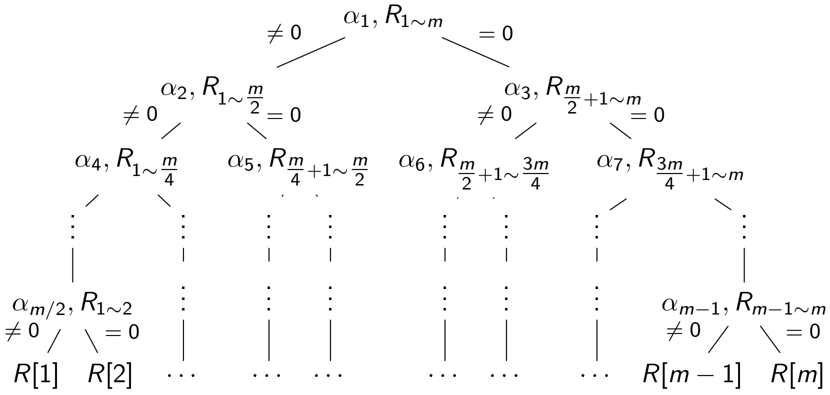


Figure : Oracle tree T for columns of R

Linear independence oracles

T is a *linear independence oracle* for R based on $\alpha_1, \ldots, \alpha_{m-1}$.

$\alpha_1, \ldots, \alpha_{m-1} \in \mathsf{K}$ be chosen uniformly and randomly.

$R_{a\sim b}$: a vector that is a linear combination of $R[a], R[a+1], \ldots, R[b]$.

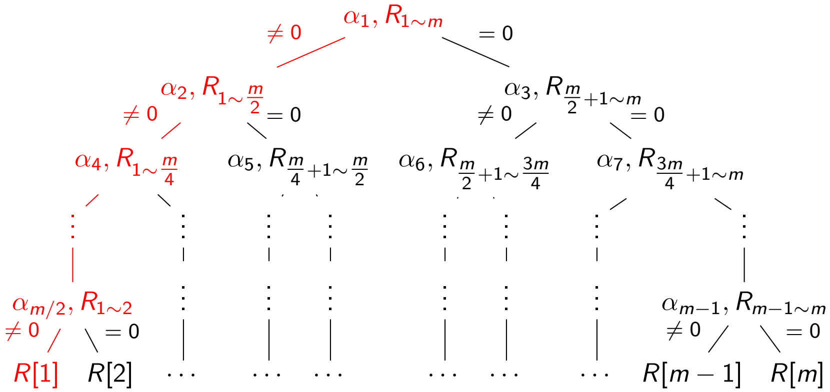


Figure : Oracle tree T for columns of R

Linear independence oracles

T is a *linear independence oracle* for R based on $\alpha_1, \ldots, \alpha_{m-1}$.

$\alpha_1, \ldots, \alpha_{m-1} \in$ K be chosen uniformly and randomly.

$R_{a\sim b}$: a vector that is a linear combination of $R[a], R[a+1], \ldots, R[b]$.

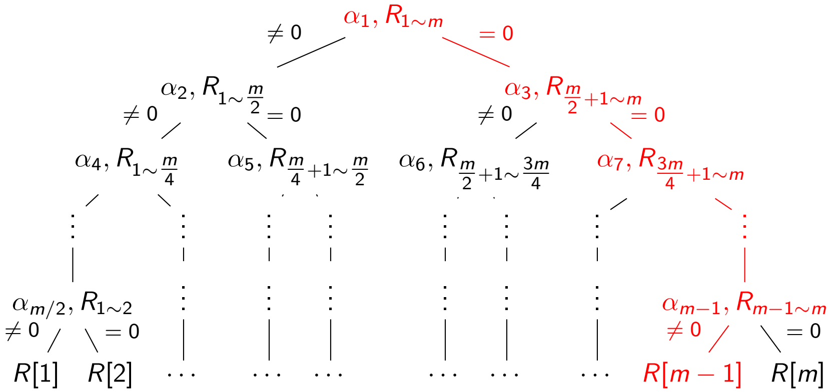


Figure : Oracle tree T for columns of R

Linear independence oracle

Cost

- The first nonzero entry in $v_s R_s$ can be found in $O(s \log m)$ field operations from K.

Probability of correctness

- T_s is correct with respect to v_s with probability at least

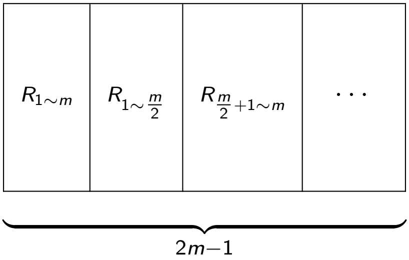
$$(1 - 1/\#\mathrm{K})^{\log_2 m}.$$

- $(T_s)_{1 \le s \le r}$, all based on the same $\alpha_1, \alpha_2, \ldots, \alpha_{m-1}$, are correct with respect to $(v_s)_{1 \le s \le r}$ with probability at least

$$(1 - r/\#\mathrm{K})^{\log_2 m}.$$

Linear independence oracles

Data structure for T

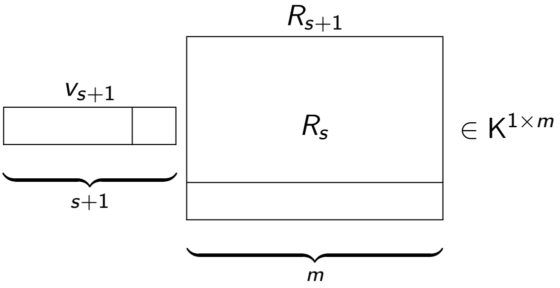


Linear independence oracles

Online construction

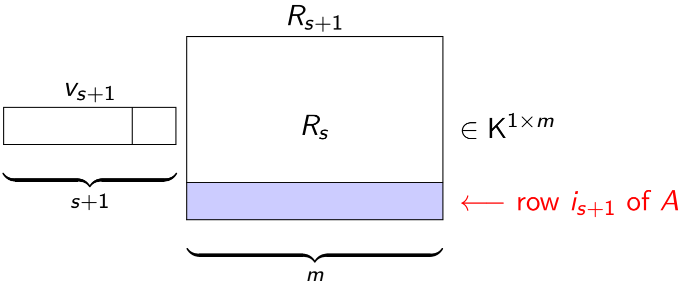
Construct T_s for $s = 0, 1, \ldots, r$ in succession.

At stage $s + 1$



Construct T_s for $s = 0, 1, \ldots, r$ in succession.

At stage $s + 1$



Linear independence oracles

Online construction

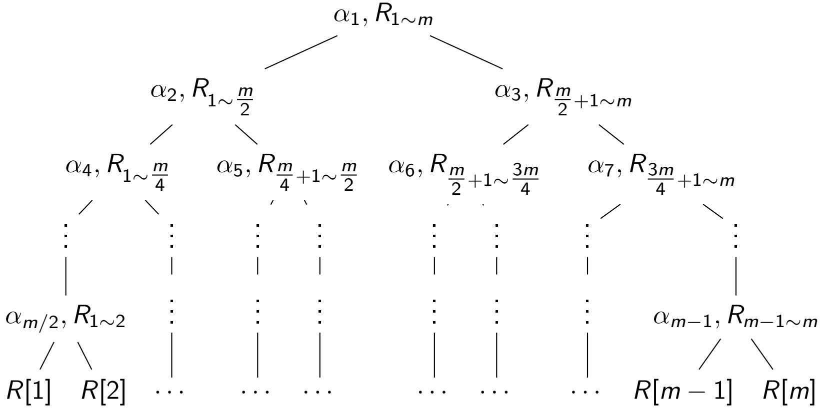
Data structure for T_{s+1}

$R_{1\sim m}$	$R_{1\sim \frac{m}{2}}$	$R_{\frac{m}{2}+1\sim m}$	$\cdots$
			$\cdots$

$2m-1$

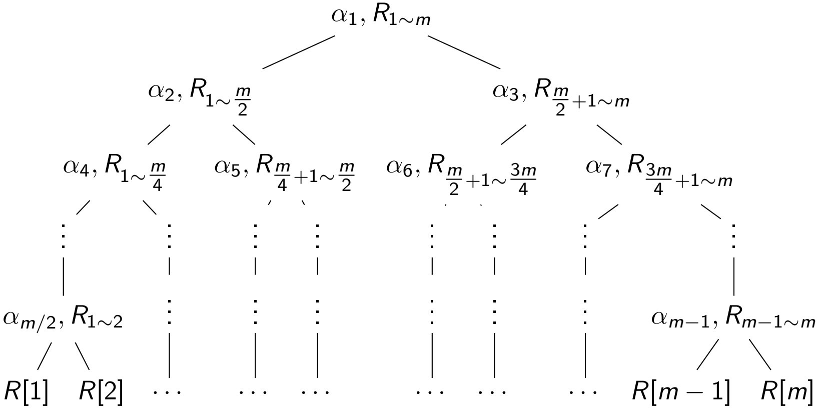
$\alpha_1, \ldots, \alpha_{m-1} \in \mathsf{K}$ be chosen uniformly and randomly.

T_{s+1}: Append the $(s+1)^{th}$ row to T_s in a bottom up fashion.



$\alpha_1, \ldots, \alpha_{m-1} \in \mathsf{K}$ be chosen uniformly and randomly.

T_{s+1}: Append the $(s+1)^{th}$ row to T_s in a bottom up fashion.



Cost: $O(m)$

Rank Profile

Theorem

There exists a randomized algorithm for RANKPROFILE *that has:*

1. $n + 2m - 2$ *random choices from* K *are required.*
2. *Probability of correctness at least*

$$\left(1 - \frac{1}{\#\mathsf{K}}\right)^r \left(1 - \frac{r}{\#\mathsf{K}}\right)^{\lceil \log_2 n \rceil + \lceil \log_2 m \rceil}$$

3. *The running time is bounded by*

$$\underbrace{2r^3}_{\textit{Inverse}} + O\left(\underbrace{nm}_{b=Aw} + \underbrace{r^2(\log n + \log m)}_{\textit{Use LIOs}} + \underbrace{(n+m)r}_{\textit{Build LIOs}}\right)$$

field operations in K.

Goals

Starting complexity: $O((n+m)r^2)$

Goals

1. Decouple the cubic part of the time complexity:

$$2r^3 + (r^2 + nm)^{1+o(1)}$$

2. Exploit possible sparsity of A:

$$2r^3 + (r^2 + |A|)^{1+o(1)}$$

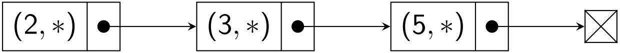
3. Incorporate fast matrix multiplication:

$$(r^\omega + |A|)^{1+o(1)}$$

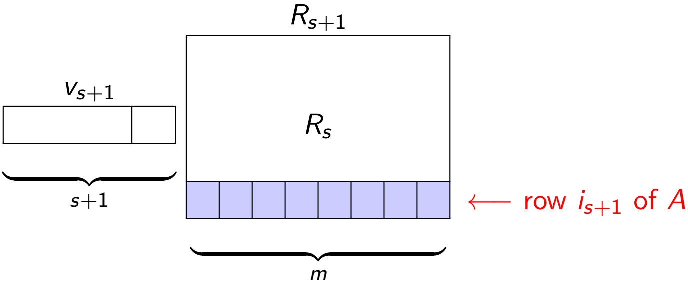
Exploit possible sparsity of A

Use a sparse representation for T.

Example. $R_{a\sim b} = \begin{bmatrix} & * & * & & * & \end{bmatrix}^T$ is represented as



Recall the construction of T_{s+1}



Only nonzero elements of row i_{s+1} of A modifies the associated vectors in T_{s+1}.

Exploit possible sparsity of A

Example. $r = 3$ and $m = 8$ Stage 0

$R_0 = \emptyset$

Cost: $O(m)$

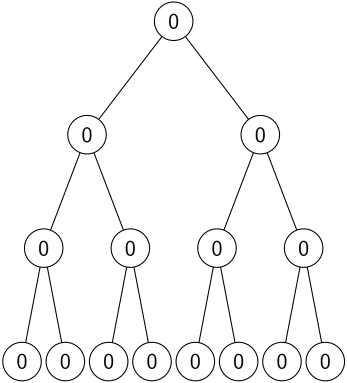
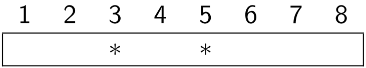


Figure : T_0

Exploit possible sparsity of A

Example. $r = 3$ and $m = 8$ Stage 1



R_1

Cost: $O(2 \log m)$

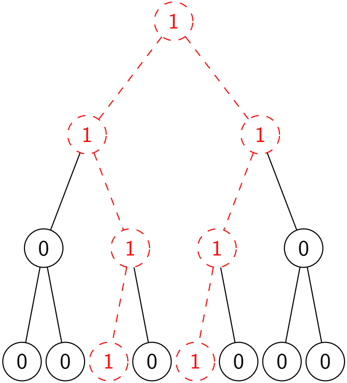


Figure : T_1

Exploit possible sparsity of A

Example. $r = 3$ and $m = 8$ Stage 2

1	2	3	4	5	6	7	8
		*		*			
			*	*			

R_2

Cost: $O(2 \log m)$

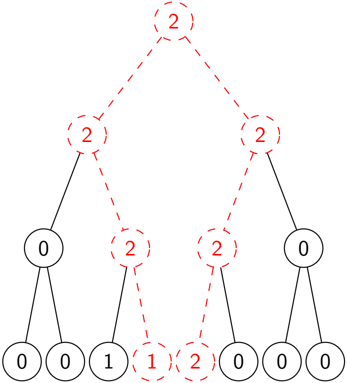


Figure : T_2

Exploit possible sparsity of A

Example. $r = 3$ and $m = 8$ Stage 3

1	2	3	4	5	6	7	8
		*		*			
			*	*			
	*			*			*

R_2

Cost: $O(3 \log m)$

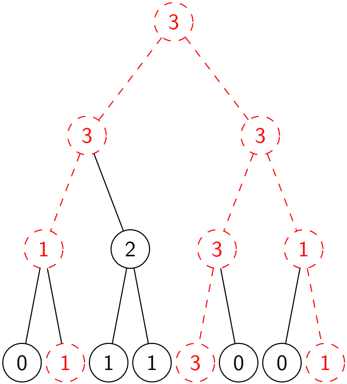


Figure : T_3

Exploit possible sparsity of A

Example. $r = 3$ and $m = 8$ Stage 3

1	2	3	4	5	6	7	8
		*		*			
			*	*			
	*			*			*

R_2

Cost: $O(3 \log m)$

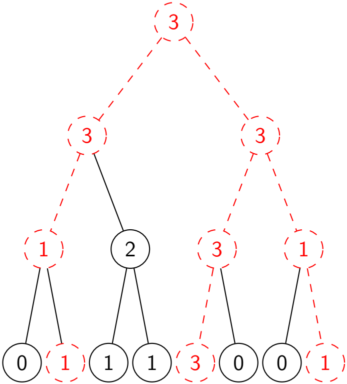


Figure : T_3

Overall cost: $O(m + |R| \log m)$

Exploit possible sparsity of A

Theorem

There exists a randomized algorithm for RANKPROFILE *that has:*

1. $n + 2m - 2$ *random choices from* K *are required.*
2. *Probability of correctness at least*

$$\left(1 - \frac{1}{\#\mathrm{K}}\right)^r \left(1 - \frac{r}{\#\mathrm{K}}\right)^{\lceil \log_2 n \rceil + \lceil \log_2 m \rceil}$$

3. *The running time is bounded by*

$$\underbrace{2r^3}_{\textit{Inverse}} + O\left(\underbrace{|A|}_{b=Aw} + \underbrace{r^2(\log n + \log m)}_{\textit{Use LIOs}} + \underbrace{n + |C| \log n + m + |R| \log m}_{\textit{Build LIOs}}\right)$$

field operations in K.

Goals

Starting complexity: $O((n+m)r^2)$

Goals

1. Decouple the cubic part of the time complexity:

$$2r^3 + (r^2 + nm)^{1+o(1)}$$

2. Exploit possible sparsity of A:

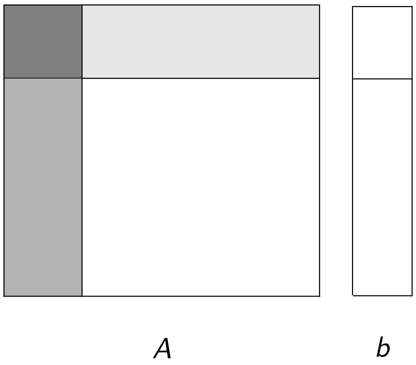
$$2r^3 + (r^2 + |A|)^{1+o(1)}$$

3. Incorporate fast matrix multiplication:

$$(r^\omega + |A|)^{1+o(1)}$$

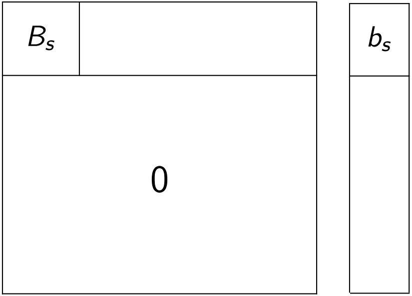
Incorporate fast matrix multiplication

The leading term $2r^3$ arises from computing the inverse of the leading $s \times s$ submatrix for $s = 1, 2, \ldots, r$.



Incorporate fast matrix multiplication

The leading term $2r^3$ arises from computing the inverse of the leading $s \times s$ submatrix for $s = 1, 2, \ldots, r$.

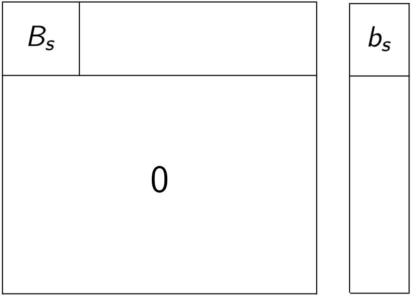


Work matrix B

b

Incorporate fast matrix multiplication

The leading term $2r^3$ arises from computing the inverse of the leading $s \times s$ submatrix for $s = 1, 2, \ldots, r$.



Work matrix B $\qquad$ b

These inverses are used to compute a sequence of subsystem solutions $B_s^{-1} b_s$ for $s = 1, 2, \ldots, r$.

Full inverse decomposition

We give a unique decomposition for the inverse

$$B_s^{-1} = (R_s L_s) \cdots (R_2 L_2)(R_1 L_1)$$

Example. $B_6^{-1} =$

$$\underbrace{\begin{bmatrix} 1 &&&&& * \\ & 1 &&&& * \\ && 1 &&& * \\ &&& 1 && * \\ &&&& 1 & * \\ &&&&& * \end{bmatrix}}_{R_6} \underbrace{\begin{bmatrix} 1 &&&&& \\ & 1 &&&& \\ && 1 &&& \\ &&& 1 && \\ &&&& 1 & \\ * & * & * & * & * & 1 \end{bmatrix}}_{L_6} \underbrace{\begin{bmatrix} 1 &&&& * & \\ & 1 &&& * & \\ && 1 && * & \\ &&& 1 & * & \\ &&&& * & \\ &&&&& 1 \end{bmatrix}}_{R_5} \underbrace{\begin{bmatrix} 1 &&&&& \\ & 1 &&&& \\ && 1 &&& \\ &&& 1 && \\ * & * & * & * & 1 & \\ &&&&& 1 \end{bmatrix}}_{L_5} \underbrace{\begin{bmatrix} 1 &&& * && \\ & 1 && * && \\ && 1 & * && \\ &&& * && \\ &&&& 1 & \\ &&&&& 1 \end{bmatrix}}_{R_4} \underbrace{\begin{bmatrix} 1 &&&&& \\ & 1 &&&& \\ && 1 &&& \\ * & * & * & 1 && \\ &&&& 1 & \\ &&&&& 1 \end{bmatrix}}_{L_4} \cdots \underbrace{\begin{bmatrix} * &&&&& \\ & 1 &&&& \\ && 1 &&& \\ &&& 1 && \\ &&&& 1 & \\ &&&&& 1 \end{bmatrix}}_{R_1} \underbrace{\begin{bmatrix} 1 &&&&& \\ & 1 &&&& \\ && 1 &&& \\ &&& 1 && \\ &&&& 1 & \\ &&&&& 1 \end{bmatrix}}_{L_1}$$

To compute a sequence of subsystem solutions $B_s^{-1} b_s$ for $s = 1, 2, \ldots, r$, it is sufficient to solve the following problem.

- OnlineInverse: Suppose the rows of $B \in \mathsf{K}^{r \times r}$ with generic rank profile are given one at a time, from first to last. As soon as rows $1, 2, \ldots, r$ of B are given, the pair of matrices (R_s, L_s) should be produced, for $s = 1, 2, \ldots, r$.

Iterative algorithm for OnlineInverse: $2r^3 + O(r^2)$

Full inverse decomposition

We give a unique decomposition for the inverse

$$B_s^{-1} = (R_s L_s) \cdots (R_2 L_2)(R_1 L_1)$$

Example. $B_6^{-1} =$

$$\underset{R_6}{\begin{bmatrix} 1 & & & & & * \\ & 1 & & & & * \\ & & 1 & & & * \\ & & & 1 & & * \\ & & & & 1 & * \\ & & & & & * \end{bmatrix}} \underset{L_6}{\begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ * & * & * & * & * & 1 \end{bmatrix}} \underset{R_5}{\begin{bmatrix} 1 & & & & * & \\ & 1 & & & * & \\ & & 1 & & * & \\ & & & 1 & * & \\ & & & & * & \\ & & & & & 1 \end{bmatrix}} \underset{L_5}{\begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ * & * & * & * & 1 & \\ & & & & & 1 \end{bmatrix}} \underset{R_4}{\begin{bmatrix} 1 & & & * & & \\ & 1 & & * & & \\ & & 1 & * & & \\ & & & * & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix}} \underset{L_4}{\begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ * & * & * & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix}} \cdots \underset{R_1}{\begin{bmatrix} * & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix}} \underset{L_1}{\begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix}}$$

To compute a sequence of subsystem solutions $B_s^{-1} b_s$ for $s = 1, 2, \ldots, r$, it is sufficient to solve the following problem.

- OnlineInverse: Suppose the rows of $B \in \mathsf{K}^{r \times r}$ with generic rank profile are given one at a time, from first to last. As soon as rows $1, 2, \ldots, r$ of B are given, the pair of matrices (R_s, L_s) should be produced, for $s = 1, 2, \ldots, r$.

Iterative algorithm for OnlineInverse: $2r^3 + O(r^2)$

How to incorporate matrix multiplication?

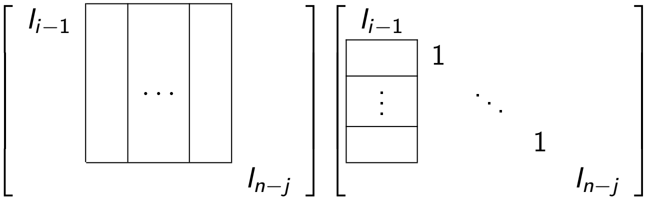
Relax, but anticipate

We adopt two ideas used in relaxed and online algorithms.

1. **Use a *relaxed* representation for B_s^{-1}.**

   Key observation: $(R_j L_j)(R_{j-1} L_{j-1}) \cdots (R_i L_i)$ can be expressed as

$R_{j\sim i} L_{j\sim i} =$



Relax, but anticipate

Represent B_s^{-1} as the product of $\mathrm{HammingWeight}(s) \leq \lceil \log s \rceil$ pair of structured matrices.

Example. The relaxed representation of B_s^{-1} for $1 \leq s \leq 8$.

s	**Relaxed representation of B_s^{-1}**
$1 = (1)_2$	$(R_{1\sim 1} L_{1\sim 1})$
$2 = (10)_2$	$(R_{2\sim 1} L_{2\sim 1})$
$3 = (11)_2$	$(R_{3\sim 3} L_{3\sim 3})(R_{2\sim 1} L_{2\sim 1})$
$4 = (100)_2$	$(R_{4\sim 1} L_{4\sim 1})$
$5 = (101)_2$	$(R_{5\sim 5} L_{5\sim 5})(R_{4\sim 1} L_{4\sim 1})$
$6 = (110)_2$	$(R_{6\sim 5} L_{6\sim 5})(R_{4\sim 1} L_{4\sim 1})$
$7 = (111)_2$	$(R_{7\sim 7} L_{7\sim 7})(R_{6\sim 5} L_{6\sim 5})(R_{4\sim 1} L_{4\sim 1})$
$8 = (1000)_2$	$(R_{8\sim 1} L_{8\sim 1})$

Example.

$$B_6^{-1} = \overset{R_{6\sim5}}{\begin{bmatrix} 1 & & & & * & * \\ & 1 & & & * & * \\ & & 1 & & * & * \\ & & & 1 & * & * \\ & & & & * & * \\ & & & & * & * \end{bmatrix}} \overset{L_{6\sim5}}{\begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ * & * & * & * & 1 & \\ * & * & * & * & & 1 \end{bmatrix}} \overset{R_{4\sim1}}{\begin{bmatrix} * & * & * & * & & \\ * & * & * & * & & \\ * & * & * & * & & \\ * & * & * & * & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix}} \overset{L_{4\sim1}}{\begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix}}$$

$$B_7^{-1} = \overset{R_7}{\begin{bmatrix} 1 & & & & & & * \\ & 1 & & & & & * \\ & & 1 & & & & * \\ & & & 1 & & & * \\ & & & & 1 & & * \\ & & & & & 1 & * \\ & & & & & & * \end{bmatrix}} \overset{L_7}{\begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ * & * & * & * & * & * & 1 \end{bmatrix}} \overset{R_{6\sim5}}{\begin{bmatrix} 1 & & & & * & * & \\ & 1 & & & * & * & \\ & & 1 & & * & * & \\ & & & 1 & * & * & \\ & & & & * & * & \\ & & & & * & * & \\ & & & & & & 1 \end{bmatrix}} \overset{L_{6\sim5}}{\begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ * & * & * & * & 1 & & \\ * & * & * & * & & 1 & \\ & & & & & & 1 \end{bmatrix}} \overset{R_{4\sim1}}{\begin{bmatrix} * & * & * & * & & & \\ * & * & * & * & & & \\ * & * & * & * & & & \\ * & * & * & * & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \end{bmatrix}} \overset{L_{4\sim1}}{\begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \end{bmatrix}}$$

$$B_8^{-1} = \overset{R_{8\sim1}}{\begin{bmatrix} * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{bmatrix}} \overset{L_{8\sim1}}{\begin{bmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{bmatrix}}$$

Relax, but anticipate

The relaxed representation is constructed in an incremental fashion.

Example.

$$\begin{aligned} B_8^{-1} &= (R_8 L_8)\ B_7^{-1} \\ &= (R_8 L_8)\ (R_7 L_7)\ (R_{6\sim5} L_{6\sim5})\ (R_{4\sim1} L_{4\sim1}) \\ &= (R_{8\sim7} L_{8\sim7})\ (R_{6\sim5} L_{6\sim5})\ (R_{4\sim1} L_{4\sim1}) \\ &= (R_{8\sim5} L_{8\sim5})\ (R_{4\sim1} L_{4\sim1}) \\ &= (R_{8\sim1} L_{8\sim1}) \end{aligned}$$

Relax, but anticipate

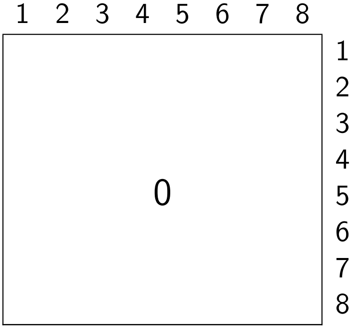
2. ***Anticipate* computations**.

   To compute the pair (R_s, L_s), we need to apply the inverse B_{s-1}^{-1} to column s of B.

   At stage $s-1$, apply parts of our representation for B_{s-1}^{-1} to multiple columns of B such that column s of B have been premultiplied with B_{s-1}^{-1} at the beginning of stage s.

Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

Stage 1

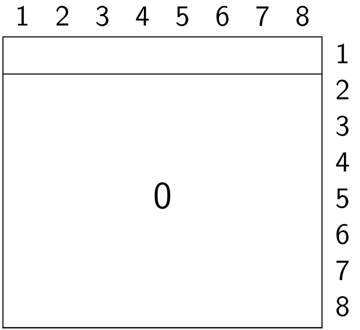


B

Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

Stage 1

- The first row of B is given.



B

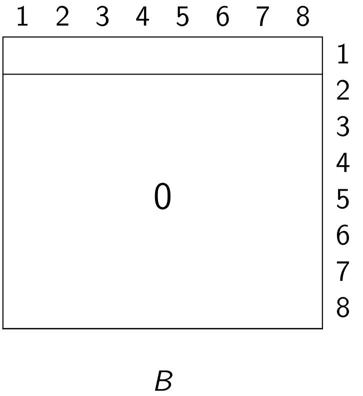
Relax, but anticipate

Example. The computations of the first four stages for $B \in \mathsf{K}^{8\times 8}$.

Stage 1

- The first row of B is given.
- Compute $(R_1 L_1)$ of shape:

$$[*]\,[1]$$



Relax, but anticipate

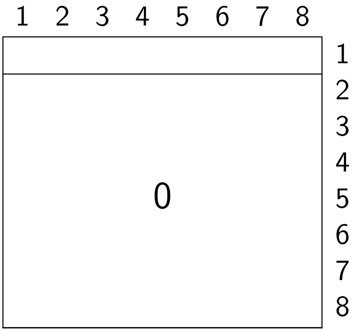
Example. The computations of the first four stages for $B \in \mathsf{K}^{8\times 8}$.

Stage 1

- The first row of B is given.
- Compute $(R_1 L_1)$ of shape:

$$[*]\,[1]$$

- $B_1^{-1} = (R_1 L_1)$.



B

Relax, but anticipate

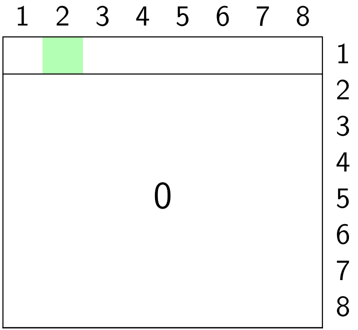
Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

Stage 1

- The first row of B is given.
- Compute $(R_1 L_1)$ of shape:

$$[*]\,[1]$$

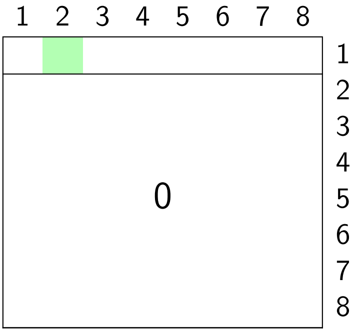
- $B_1^{-1} = (R_1 L_1)$.
- Apply $R_1 L_1$ to column 2 of B.



B

Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

Stage 2

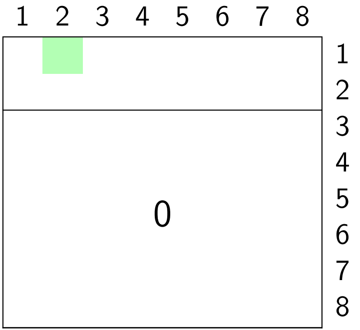


B

Example. The computations of the first four stages for $B \in \mathsf{K}^{8\times 8}$.

Stage 2

- The 2^{nd} row of B is given.



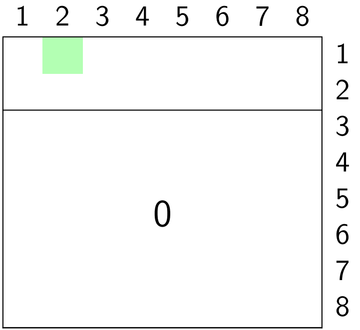
B

Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

Stage 2

- The 2^{nd} row of B is given.
- Compute $(R_2 L_2)$ of shape:

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



B

Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

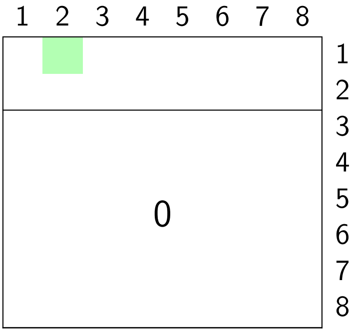
Stage 2

- The 2^{nd} row of B is given.
- Compute $(R_2 L_2)$ of shape:

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

- Compress $(R_2 L_2)(R_1 L_1) = (R_{2\sim 1} L_{2\sim 1})$

$$\begin{bmatrix} 1 & * \\ & * \end{bmatrix} \begin{bmatrix} 1 & \\ * & 1 \end{bmatrix} \begin{bmatrix} * & \\ & 1 \end{bmatrix} \begin{bmatrix} 1 & \\ & 1 \end{bmatrix} = \begin{bmatrix} * & * \\ * & * \end{bmatrix} \begin{bmatrix} 1 & \\ & 1 \end{bmatrix}$$



B

Relax, but anticipate

Example. The computations of the first four stages for $B \in \mathsf{K}^{8\times 8}$.

Stage 2

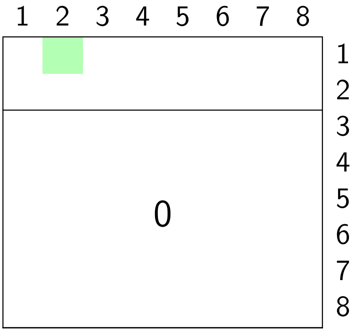
- The 2^{nd} row of B is given.
- Compute $(R_2 L_2)$ of shape:

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

- Compress $(R_2 L_2)(R_1 L_1) = (R_{2\sim 1} L_{2\sim 1})$

$$\begin{bmatrix} 1 & * \\ & * \end{bmatrix} \begin{bmatrix} 1 & \\ * & 1 \end{bmatrix} \begin{bmatrix} * & \\ & 1 \end{bmatrix} \begin{bmatrix} 1 & \\ & 1 \end{bmatrix} = \begin{bmatrix} * & * \\ * & * \end{bmatrix} \begin{bmatrix} 1 & \\ & 1 \end{bmatrix}$$

- $B_2^{-1} = (R_{2\sim 1} L_{2\sim 1})$.



B

Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

Stage 2

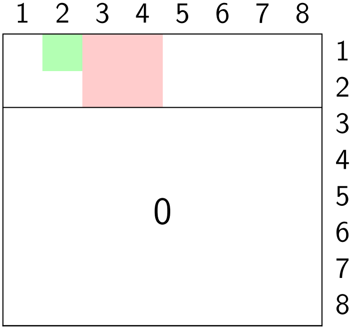
- The 2^{nd} row of B is given.
- Compute $(R_2 L_2)$ of shape:

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

- Compress $(R_2 L_2)(R_1 L_1) = (R_{2\sim 1} L_{2\sim 1})$

$$\begin{bmatrix} 1 & * \\ & * \end{bmatrix} \begin{bmatrix} 1 & \\ * & 1 \end{bmatrix} \begin{bmatrix} * & \\ & 1 \end{bmatrix} \begin{bmatrix} 1 & \\ & 1 \end{bmatrix} = \begin{bmatrix} * & * \\ * & * \end{bmatrix} \begin{bmatrix} 1 & \\ & 1 \end{bmatrix}$$

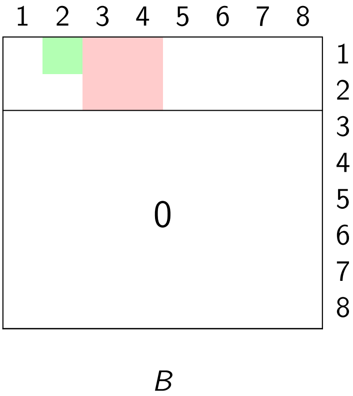
- $B_2^{-1} = (R_{2\sim 1} L_{2\sim 1})$.
- Apply $(R_{2\sim 1} L_{2\sim 1})$ to columns 3,4 of B.



B

Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

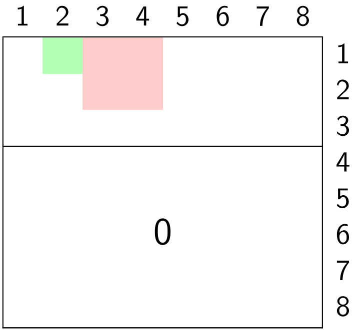
Stage 3



Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

Stage 3

- The 3^{rd} row of B is given.



B

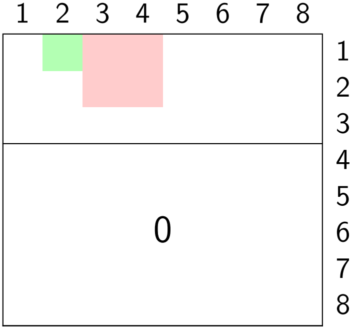
Relax, but anticipate

Example. The computations of the first four stages for $B \in \mathsf{K}^{8\times 8}$.

Stage 3

- The 3^{rd} row of B is given.
- Compute $(R_3 L_3)$ of shape:

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



B

Relax, but anticipate

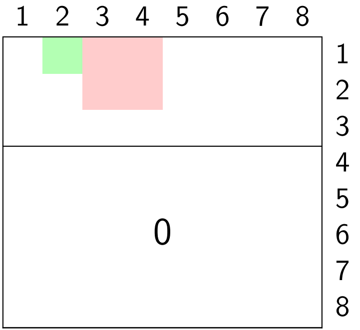
Example. The computations of the first four stages for $B \in \mathsf{K}^{8\times 8}$.

Stage 3

- The 3^{rd} row of B is given.
- Compute $(R_3 L_3)$ of shape:

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

- $B_3^{-1} = (R_3 L_3)(R_{2\sim 1} L_{2\sim 1})$.



B

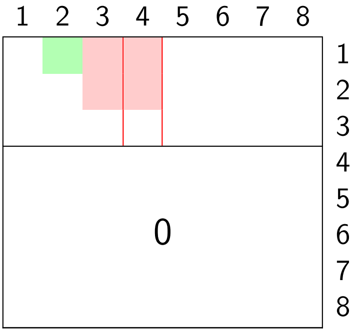
Example. The computations of the first four stages for $B \in \mathsf{K}^{8\times 8}$.

Stage 3

- The 3^{rd} row of B is given.
- Compute $(R_3 L_3)$ of shape:

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

- $B_3^{-1} = (R_3 L_3)(R_{2\sim 1} L_{2\sim 1})$.
- Apply $(R_3 L_3)$ to column 4 of B.

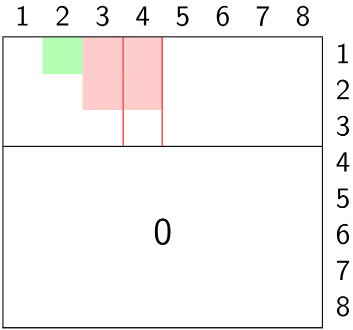


B

Relax, but anticipate

Example. The computations of the first four stages for $B \in \mathsf{K}^{8\times 8}$.

Stage 4

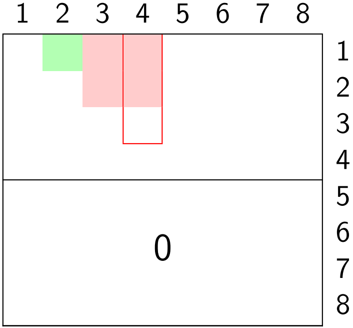


B

Example. The computations of the first four stages for $B \in \mathsf{K}^{8\times 8}$.

Stage 4

- The 4^{th} row of B is given.



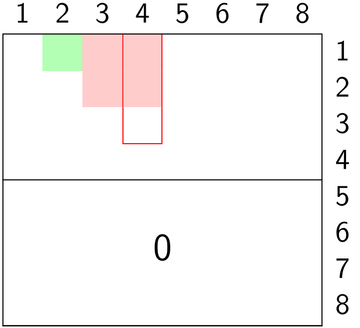
B

Example. The computations of the first four stages for $B \in \mathsf{K}^{8\times 8}$.

Stage 4

- The 4^{th} row of B is given.
- Compute $(R_4 L_4)$ of shape:

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



B

Relax, but anticipate

Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

Stage 4

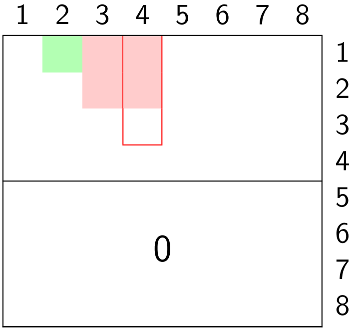
- The 4^{th} row of B is given.
- Compute $(R_4 L_4)$ of shape:

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

- Compress $(R_4 L_4)(R_3 L_3) = (R_{4\sim 3} L_{4\sim 3})$

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

$$= \begin{bmatrix} 1 & & * & * \\ & 1 & * & * \\ & & * & * \\ & & * & * \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ * & * & 1 & \\ * & * & & 1 \end{bmatrix}$$



B

Relax, but anticipate

Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

Stage 4

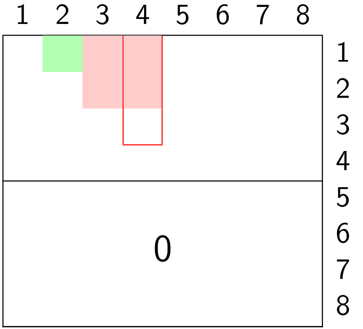
- The 4^{th} row of B is given.
- Compute $(R_4 L_4)$ of shape:

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

- Compress
$(R_{4\sim 3} L_{4\sim 3})(R_{2\sim 1} L_{2\sim 1}) = (R_{4\sim 1} L_{4\sim 1})$

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

$$= \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix}$$



B

Relax, but anticipate

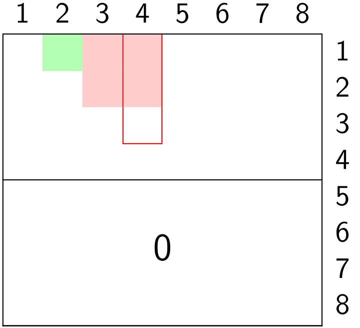
Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

Stage 4

- The 4^{th} row of B is given.
- Compute $(R_4 L_4)$ of shape:

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

- $B_4^{-1} = (R_{4\sim 1} L_{4\sim 1})$.



B

Relax, but anticipate

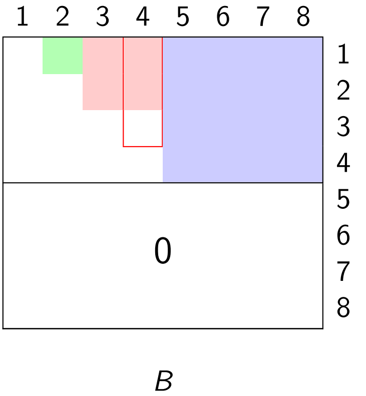
Example. The computations of the first four stages for $B \in \mathrm{K}^{8\times 8}$.

Stage 4

- The 4^{th} row of B is given.
- Compute $(R_4 L_4)$ of shape:

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

- $B_4^{-1} = (R_{4\sim 1} L_{4\sim 1})$.
- Apply $R_{4\sim 1} L_{4\sim 1}$ to columns 5,6,7,8 of B.



Relax, but anticipate

- At stages $2^k = 1, 2, 4, \ldots$ the explicit inverse has been computed.
- The number of compressions done at stage s is equal to the maximal $c \in \mathbb{Z}$ such that $2^c \mid s$, thus some stages are more costly than others.
- By taking into account the special structure of the $(R_{j\sim i} L_{j\sim i})$ matrices, an amortized analysis of the above approach yields an algorithm for OnlineInverse with overall running time bounded by $O(r^\omega)$ field operations from K.

Rank profile algorithm

Steps:

- Use Cheung, Kwok & Lau's (2013) algorithm to find a subset of r linearly independent columns of A (with high probability).
- Use a Toeplitz preconditioner L such that the leading $s \times s$ submatrix of BL, $1 \leq s \leq r$, are nonsingular.
- Incorporate the relaxed approach for OnlineInverse into the oracle rank profile algorithm.

Overall running time: $(r^{\omega} + n + m + |A|)^{1+o(1)}$

Future work

- Our algorithm for LinSys has overall running time

$$2r^3 + (r^2 + n + m + |R| + |C|)^{1+o(1)},$$

an open problem is to reduce the leading term $2r^3$ to $O(r^\omega)$ by incorporating fast matrix multiplication.

- Find other applications for our relaxed algorithm for online matrix inversion.