A Relaxed Algorithm for Online Matrix Inversion

Arne Storjohann Shiyun Yang
University of Waterloo

1. Overview

Let K be a finite field. Assume we are given an $A \in \mathsf{K}^{n\times n}$ that has generic rank profile. Then there exists a sequence of pairs of matrices (R_i, L_i), $1 \le i \le n$, that gives a unique decomposition $A^{-1} = (R_nL_n)\cdots(R_2L_2)(R_1L_1)$. See an example for $n = 6$ on the right, where the matrix $L_1 = I_6$ is omitted. In this poster, we consider the following problem:

OnlineInverse: Suppose the rows of $A \in \mathsf{K}^{n\times n}$ with generic rank profile are given one at a time, from first to last. As soon as rows $1, 2, \ldots, s$ of A are given, produce the pair of matrices (R_s, L_s), for $s = 1, 2, \ldots, n$. Note: The iterative algorithm based on Gaussian elimination solves this problem in overall cost $2n^3 + O(n^2)$ field operations in K.

Result: By introducing a compressed representation of the inverse and using a relaxed approach, we give an algorithm that allows to incorporate matrix multiplication and solves problem OnlineInverse in total time $O(n^\omega)$ field operations, where ω is the exponent of matrix multiplication.

The full inverse decomposition for $n = 6$

$$A_6^{-1} = \begin{matrix} R_6 \\ \left[\begin{smallmatrix} 1 & & & & & \\ & 1 & & & & * \\ & & 1 & & & * \\ & & & 1 & & * \\ & & & & 1 & * \\ & & & & & * \end{smallmatrix}\right] \end{matrix} \begin{matrix} L_6 \\ \left[\begin{smallmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ * & * & * & * & * & 1 \end{smallmatrix}\right] \end{matrix} \begin{matrix} R_5 \\ \left[\begin{smallmatrix} 1 & & & & & \\ & 1 & & & & * \\ & & 1 & & & * \\ & & & 1 & & * \\ & & & & & * \\ & & & & & 1 \end{smallmatrix}\right] \end{matrix} \begin{matrix} L_5 \\ \left[\begin{smallmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ * & * & * & * & 1 & \\ & & & & & 1 \end{smallmatrix}\right] \end{matrix} \begin{matrix} R_4 \\ \left[\begin{smallmatrix} 1 & & & & & \\ & 1 & & & & * \\ & & 1 & & & * \\ & & & 1 & & * \\ & & & & & * \\ & & & & & 1 \end{smallmatrix}\right] \end{matrix} \begin{matrix} L_4 \\ \left[\begin{smallmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ * & * & * & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{smallmatrix}\right] \end{matrix} \begin{matrix} R_3 \\ \left[\begin{smallmatrix} 1 & & & & & \\ & 1 & & & & * \\ & & 1 & & & * \\ & & & & & * \\ & & & & 1 & \\ & & & & & 1 \end{smallmatrix}\right] \end{matrix} \begin{matrix} L_3 \\ \left[\begin{smallmatrix} 1 & & & & & \\ & 1 & & & & \\ & * & * & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{smallmatrix}\right] \end{matrix} \begin{matrix} R_2 \\ \left[\begin{smallmatrix} 1 & * & & & & \\ & * & & & & \\ & & 1 & & & \\ & & & & 1 & \\ & & & & & 1 \end{smallmatrix}\right] \end{matrix} \begin{matrix} L_2 \\ \left[\begin{smallmatrix} 1 & & & & & \\ & * & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{smallmatrix}\right] \end{matrix} \begin{matrix} R_1 \\ \left[\begin{smallmatrix} * & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{smallmatrix}\right] \end{matrix}$$

The relaxed/compressed inverse decomposition for $n = 6$

$$A_6^{-1} = \begin{matrix} R_{6\sim5} \\ \left[\begin{smallmatrix} 1 & & & & & \\ & 1 & & & & * \\ & & 1 & & & * \\ & & & 1 & & * \\ & & & & & * \\ & & & & & * \end{smallmatrix}\right] \end{matrix} \begin{matrix} L_{6\sim5} \\ \left[\begin{smallmatrix} 1 & & & & & \\ & 1 & & & & \\ & & & 1 & & \\ * & * & * & * & 1 & \\ * & * & * & * & & 1 \end{smallmatrix}\right] \end{matrix} \begin{matrix} R_{4\sim1} \\ \left[\begin{smallmatrix} * & * & * & * & & \\ * & * & * & * & & \\ * & * & * & * & & \\ * & * & * & * & & \\ & & & & 1 & \\ & & & & & 1 \end{smallmatrix}\right] \end{matrix} \begin{matrix} L_{4\sim1} \\ \left[\begin{smallmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{smallmatrix}\right] \end{matrix}$$

2. Iterative Approach

Let A_s be the leading $s \times s$ submatrix of A. Suppose we have computed A_{s-1}^{-1} for some $s > 0$. Then Gaussian elimination will produce a pair of $s \times s$ matrices $\bar{L}_s$ and $\bar{R}_s$ such that

$$\begin{matrix} I_s \\ \left[\begin{array}{c|c} I_{s-1} & \\ \hline & 1 \end{array}\right] \end{matrix} = \underbrace{\begin{matrix} \bar{R}_s \\ \left[\begin{array}{c|c} I_{s-1} & * \\ \hline & * \end{array}\right] \end{matrix} \begin{matrix} \bar{L}_s \\ \left[\begin{array}{c|c} I_{s-1} & \\ \hline * & 1 \end{array}\right] \end{matrix} \begin{matrix} \\ \left[\begin{array}{c|c} A_{s-1}^{-1} & \\ \hline & 1 \end{array}\right] \end{matrix}}_{A_s^{-1}} \begin{matrix} A_s \\ \left[\begin{array}{c|c} A_{s-1} & u_s \\ \hline * & * \end{array}\right] \end{matrix}$$

$$= \begin{matrix} \bar{R}_s \\ \left[\begin{array}{c|c} I_{s-1} & * \\ \hline & * \end{array}\right] \end{matrix} \begin{matrix} \bar{L}_s \\ \left[\begin{array}{c|c} I_{s-1} & \\ \hline * & 1 \end{array}\right] \end{matrix} \begin{matrix} \\ \left[\begin{array}{c|c} I_{s-1} & A_{s-1}^{-1}u_s \\ \hline * & * \end{array}\right] \end{matrix}$$

Once $A_{s-1}^{-1}u_s$ has been computed, the matrices $\bar{L}_s$ and $\bar{R}_s$ can be computed in a further $O(s)$ field operations. Then we can recover $R_s = \mathrm{diag}(\bar{R}_s, I_{n-s})$ and $L_s = \mathrm{diag}(\bar{L}_s, I_{n-s})$, $1 \le s \le n$. A careful analysis shows that this iterative approach solves problem OnlineInverse in overall cost $\mathbf{2n^3 + O(n^2)}$ field operations in K.

3. Relax, but Anticipate

To incorporate matrix multiplication we adopt two ideas used in relaxed [3] and online [2] algorithms.

1. **Use a *relaxed* representation for A_s^{-1}.** To explicitly store each A_s^{-1} for $s = 1, 2, 3, \ldots$ as a dense matrix is too expensive, but the representation as a product of s pairs of structured matrices is too lazy. Instead, we represent A_s^{-1} as the product of HammingWeight$(s) \le \lceil \log s \rceil$ pair of structured matrices. The key observation is that for $j \ge i$, $(R_jL_j)(R_{j-1}L_{j-1})\cdots(R_iL_i)$ can be compressed to

$$R_{j\sim i}L_{j\sim i} = \begin{bmatrix} I_{i-1} & & \\ & \cdots & \\ & & I_{n-j} \end{bmatrix} \begin{bmatrix} I_{i-1} & & \\ & 1 & \\ & \vdots & \ddots \\ & & & 1 \\ & & & & I_{n-j} \end{bmatrix}.$$

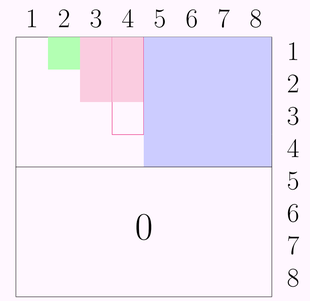
Example. The relaxed representation of A_s^{-1} for $1 \le s \le 8$.

s	**Relaxed representation of A_s^{-1}**
$1 = (1)_2$	$(R_{1\sim1}L_{1\sim1})$
$2 = (10)_2$	$(R_{2\sim1}L_{2\sim1})$
$3 = (11)_2$	$(R_{3\sim3}L_{3\sim3})(R_{2\sim1}L_{2\sim1})$
$4 = (100)_2$	$(R_{4\sim1}L_{4\sim1})$
$5 = (101)_2$	$(R_{5\sim5}L_{5\sim5})(R_{4\sim1}L_{4\sim1})$
$6 = (110)_2$	$(R_{6\sim5}L_{6\sim5})(R_{4\sim1}L_{4\sim1})$
$7 = (111)_2$	$(R_{7\sim7}L_{7\sim7})(R_{6\sim5}L_{6\sim5})(R_{4\sim1}L_{4\sim1})$
$8 = (1000)_2$	$(R_{8\sim1}L_{8\sim1})$

2. ***Anticipate* computations**. The dominant cost of computing the pair (R_s, L_s) is to compute the matrix×vector product $A_{s-1}^{-1}u_s$. Instead, at stage $s-1$, we apply parts of our relaxed representation of A_{s-1}^{-1} to multiple columns of A, and apply compressions to matrices with equal size blocks to update the relaxed representation of the inverse. This effectively incorporates matrix multiplication.

Example. The computations of the first four stages for $A \in \mathsf{K}^{8\times8}$.

0. Initialize B to be the 8×8 zero matrix.
1. Copy row 1 of A to row 1 of B.
   Compute (R_1, L_1).
   Apply R_1L_1 to column 2 of B (■).
2. Copy row 2 of A to row 2 of B.
   Compute (R_2, L_2).
   Compress $(R_2L_2)(R_1L_1) = R_{2\sim1}L_{2\sim1}$.
   Apply $R_{2\sim1}L_{2\sim1}$ to columns 3,4 of B (■).
3. Copy row 3 of A to row 3 of B.
   Compute (R_3, L_3).
   Apply R_3L_3 to column 4 of B (□).
4. Copy row 4 of A to row 4 of B.
   Compute (R_4, L_4).
   Compress $(R_4L_4)(R_3L_3) = R_{4\sim3}L_{4\sim3}$.
   Compress $(R_{4\sim3}L_{4\sim3})(R_{2\sim1}L_{2\sim1}) = R_{4\sim1}L_{4\sim1}$.
   Apply $R_{4\sim1}L_{4\sim1}$ to columns 5,6,7,8 of B (■).



Work matrix B

4. Application

Let $A \in \mathsf{K}^{n\times m}$ of (unknown) rank r be given. Our ISSAC 2014 paper [5, Theorem 19] gives an algorithm that computes the row rank profile of A in a Monte Carlo fashion in time $2r^3 + (r^2 + |A|)^{1+o(1)}$, where $|A| \ge \max(n, m)$ is an upper bound on the number of nonzero entries of A and the $o(1)$ in the cost estimates captures some missing $\log n$ and $\log m$ factors. The leading term $2r^3$ in the cost estimate arises from the iterative approach mentioned above to solve problem OnlineInverse.

Incorporating the relaxed algorithm for OnlineInverse described above, together with a Toeplitz preconditioner [4], we can compute the row rank profile of a full column rank matrix A in time $(r^\omega + |A|)^{1+o(1)}$. Combined with the algorithm supporting [1, Theorem 2.11] for computing a maximal rank subset of of linearly independent columns, this gives an algorithm that computes **the row rank profile of an arbitrary A** with high probability in time $\mathbf{(r^\omega + |A|)^{1+o(1)}}$.

References

[1] H. Y. Cheung, T. C. Kwok, and L. C. Lau. Fast matrix rank algorithms and applications. *Journal of the ACM*, 60(5):733–751, 2013. Article No. 31.

[2] P. Giorgi and R. Lebreton. Online order basis algorithm and its impact on block Wiedemann algorithm. In *ISSAC'14*. ACM Press, New York, 2014.

[3] J. van der Hoeven. Relax, but don't be too lazy. *Journal of Symbolic Computation*, 36(6):479–542, 2002.

[4] E. Kaltofen and B. D. Saunders. On Wiedemann's method of solving sparse linear systems. In *Proc. AAECC-9, Lecture Notes in Comput. Sci., vol. 539*, pages 29–38, 1991.

[5] A. Storjohan and S. Yang. Linear independence oracles and applications to rectangular and low rank linear systems. In *ISSAC'14*. ACM Press, New York, 2014.