Given pivot matrix $P_{old} = A[\mathcal{I},\mathcal{J}]$ with $\tilde{\chi} < \chi$. To improve CI decomposition, add a new pivot:

$$P_{new} = \begin{pmatrix} A(\mathcal{I},\mathcal{J}) & A(\mathcal{I},y_0) \\ A(x_0,\mathcal{J}) & A(x_0,y_0) \end{pmatrix} = \begin{bmatrix} \boxed{P_{old}} & \\ & \text{new} \end{bmatrix} \qquad (1)$$

How do we find the 'best' new element $A(x_0, y_0)$?

Recall Schur determinant identity (TCI.3.29) [for proof: see (TCI.5.3-5)]:

$$\det[P_{new}] = \det[\overbrace{A(\mathcal{I},\mathcal{J})}^{P_{old}}]\,\det[\underbrace{\underbrace{A(x_0,y_0)}_{\text{new element}} - \underbrace{A(x_0,\mathcal{J})A^{-1}(\mathcal{I},\mathcal{J})A(\mathcal{I},y_0)}_{\text{'old' CI prediction for that element}}}_{\text{error of 'old' CI prediction}}] \qquad (2)$$

Maximum volume principle

To get best improvement, i.t. to maximize reduction in old error, add to P that new element $A(x_0, y_0)$ that maximizes error of 'old' CI prediction. By (2), that maximizes $|\det[P_{new}]|$, i.e. volume spanned by row or column vectors of pivot matrix.

Add new pivots until |error of 'old' CI prediction| < specified tolerance τ for all potential new pivots.

The resulting decomposition is denoted $CI_\tau(A)$ (subscript τ denotes 'specified tolerance')

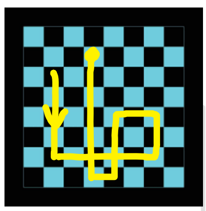
Finding truly optimum new element requires 'full search' of all potential new ones, with runtime $\mathcal{O}(m\,n)$

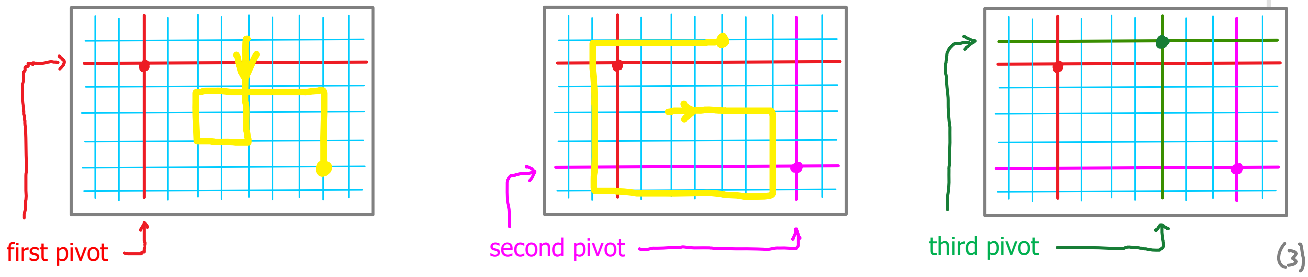
If A is a very large matrix, finding optimal new pivot can take very long time.

In practice, we don't insist on finding strictly optimal pivots. It suffices to obtain 'fairly good' pivots in shorter runtime (with better scaling, e.g. $\mathcal{O}(m+n)$), at the cost of somewhat (~10%) larger $\tilde{\chi}$ to reach specified tolerance.

Faster alternative strategy, that is successful in practice and achieves $\mathcal{O}(m+n)$ runtime:

'Rook search' (among non-pivot rows and columns):
search along randomly chosen initial column for the row yielding maximum old error, then along that row for the column yielding maximal old error, etc. Stop when "rook condition is established", i.e. when an element is found that maximizes old error along both its row and column; select that element as new pivot.





(3)

Let P_r denote pivot matrix after r steps of sequential rook search: $P_r = A[(i_1, \dots, i_r), (j_1, \dots, j_r)]$

Its elements satisfy 'rook conditions': each new pivot (i_r, j_r) maximizes old error along its row and column:

$$i_r = \mathrm{argmax}\big([A/P_{r-1}](\mathbb{I}, j_r)\big), \qquad j_r = \mathrm{argmax}\big([A/P_{r-1}](i_r, \mathbb{J})\big) \qquad (4)$$

Rook search is 'greedy' algorithm: makes locally-optimal choice instead of seeking globally-optimal solution.

Compared to full pivoting, rook pivoting has
(i) computational cost for finding one new pivot is reduced from $\mathcal{O}(m \cdot n)$ to $\mathcal{O}(\max(m,n))$; (5)
(ii) comparable robustness [Poole2000];
(iii) almost as good convergence of CI in practice.

Issues:

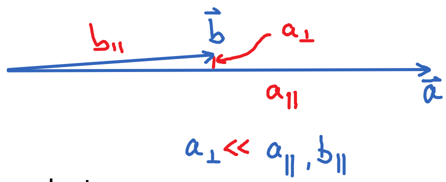
(i) 'Ergodicity problem': Rook pivoting may miss relevant parts of matrix (miss some linearly independent rows and columns), yielding suboptimal CI decomposition (better ones having same rank exisit).
Partial remedy: perform several rook pivot searches in parallel $\Longrightarrow$ 'block rook pivoting'.

(ii) 'Increasing linear dependence' (problematic for any 'accumulative' CI scheme, which keeps adding pivots but never discards pivots that turn out to be suboptimal): newly added rows and columns may increasingly be almost linearly dependent (almost parallel) to old ones. Then computation of P^{-1} becomes unstable:

When inverting P, the relative error in P^{-1} can be estimated as (original error in P) * (condition number).
'condition number': $\kappa(P) = s_{max}/s_{min}$ (singular values of P) [Golub2013, Section 2.6 (pp. 87- 90)]
In practice, (original error in P) $\simeq$ floating point error; computation of P^{-1} becomes unstable for $\kappa(P) > 10^{14}$

$\boxed{\text{So, large } \kappa(P) \Rightarrow \text{ instabilities for computing } P^{-1}}$

Large condition numbers arise
- if P has two almost parallel columns (so that their orthogonal components are much smaller than their parallel ones), or
- if $\|\cdot\|_2$ length of some columns is much larger than of others.



This observation generalizes to sets of columns that are almost linearly dependent.

Strategies for ameliorating / avoiding problem of increasing linear dependence:

1. When adding new pivots, discard suboptimal earlier pivots $\Longrightarrow$ 'block rook pivoting' (see below).

2. avoid explicit computation of P^{-1} $\Longrightarrow$ partial rank-revealing LU (prrLU) decomposition (see TCI.5).

(iii) 'Update issue': in TCI applications (see TCI.7-10), one routinely encounters the following situation:
having found a CI factorization of a $A[\mathbb{I}, \mathbb{J}]$ with 'good' pivot matrix $A[\mathcal{I}, \mathcal{J}]$,
the matrix A itself is updated by adding some rows and columns: $A \longrightarrow A'[\mathbb{I} \oplus \mathbb{I}', \mathbb{J} \oplus \mathbb{J}']$
Then, we seek a CI of the enlarged matrix A'. We would like to initialize search for new/better pivots by starting from the old pivots $\mathcal{I}, \mathcal{J}$, but expect that some of them will not be optimal for A', so we need a way to discard bad ones.

Block rook search

Goal: update list of pivots in a manner that can both add new ones and discards old ones if they are found to be suboptimal; for efficiency, use old pivots as starting point for more/better pivots.

Strategy: do parallel search of several rows or columns, starting from existing pivot rows and columns

Input: matrix $A(\mathbb{I},\mathbb{J})$ and 'old' pivot matrix $A(\mathcal{I},\mathcal{J})$ defined by old pivot lists $\mathcal{I},\mathcal{J}$ of length $\tilde{\chi}$.
Alternatingly search for better pivots along columns and rows in odd or even iterations, respectively.

Initialization: construct list of $2\tilde{\chi}$ candidate columns $\mathcal{J}' = \mathcal{J} \cup \{\tilde{\chi}$ new random indices $\in \mathbb{J}\setminus\mathcal{J}\}$ (6)

Odd iterations: construct column matrix $C = A(\mathbb{I},\mathcal{J}')$, do full-search CI$_\tau$ decomposition thereof,

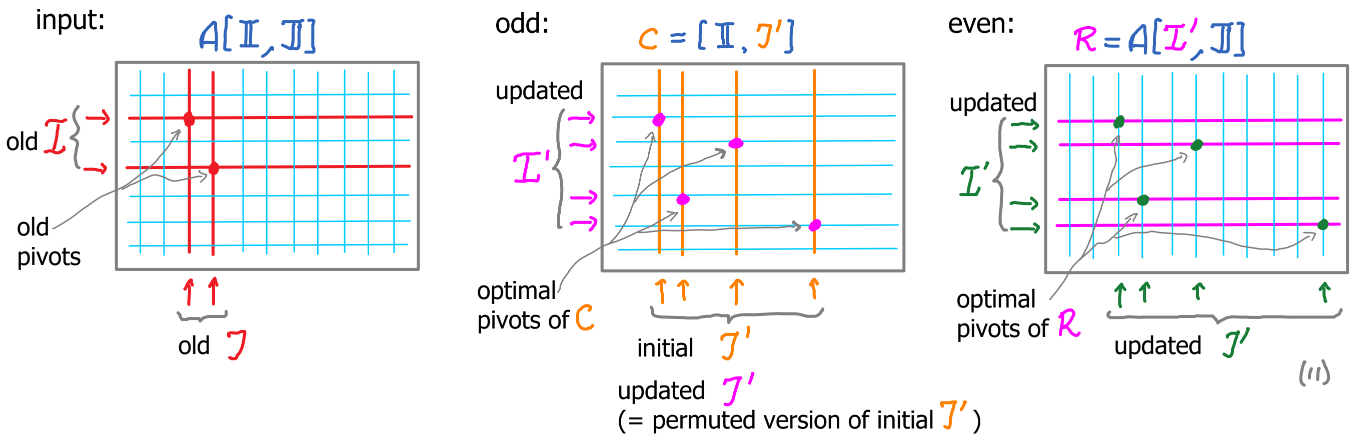
define updated candidate lists $\mathcal{I}',\mathcal{J}' \longleftarrow$ pivots of $CI_\tau(C)$ (7)

Even iterations: construct row matrix $R = A(\mathcal{I}',\mathbb{J})$, do full-search CI$_\tau$ decomposition thereof,

define updated candidate lists $\mathcal{I}',\mathcal{J}' \longleftarrow$ pivots of $CI_\tau(R)$ (8)

Stop when pivot lists $\mathcal{I}',\mathcal{J}'$ no longer change, then update $(\mathcal{I},\mathcal{J}) \leftarrow (\mathcal{I}',\mathcal{J}')$ (9)

Output: new pivot lists $(\mathcal{I},\mathcal{J})$ of up to $2\tilde{\chi}$ elements each that define $P_{new} = A(\mathcal{I},\mathcal{J})$ (10)



(11)

Remarks:

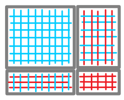
[Fernandez2025, App. A.2]

- It can be shown that block rook search always terminates, and when it does, rook conditions (4) are satisfied.
- Block rook search not only adds new pivots, but also <u>discards old pivots</u>, if better ones become available.
- Block rook search explores $O(\tilde{\chi} m + \tilde{\chi} n)$ candidate pivots, compared to only $O(m+n)$ for rook search, and thus takes longer by a factor $\mathcal{O}(\tilde{\chi})$; but in return it yields $\mathcal{O}(\tilde{\chi})$ new pivots, rather than just 1.

5. Properties of Schur complement

[Fernandez2025, Sec. 3.2]

TCI.5

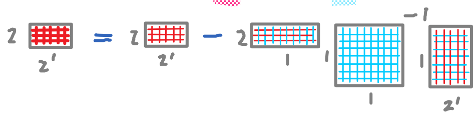
Consider block matrix: $A = \begin{pmatrix} A_{11} & A_{12'} \\ A_{21} & A_{22'} \end{pmatrix} =$  with A_{11} assumed square and invertible. (1)

Definition: 'Schur complement of A_{11} in A :

$$\boxed{[A/A_{11}] \equiv A_{22'} - A_{21}(A_{11})^{-1}A_{12'}} \qquad (2)$$

note index structure: 2's outside, 1's inside

[prime on 22' indicates that $A_{22'}$ block can be rectangular with dim(2) $\neq$ dim(2')]

block dimensions match: 

Schur complement $[A/A_{11}]$ has dimensions of $A_{22'}$, but includes information from other three blocks.

$[A/A_{11}]$ is result of 'eliminating' block A_{11} from A, or of 'projecting' or 'restricting' A to space of A_{22}.

Useful relations involving the Schur complement:

(0) 'Factorization property': A can be factorized as follows:

$$A = \begin{pmatrix} A_{11} & A_{12'} \\ A_{21} & A_{22'} \end{pmatrix} = \begin{pmatrix} \mathbb{1}_{11} & 0 \\ A_{21}A_{11}^{-1} & \mathbb{1}_{22} \end{pmatrix} \begin{pmatrix} A_{11} & 0 \\ 0 & [A/A_{11}] \end{pmatrix} \begin{pmatrix} \mathbb{1}_{11} & A_{11}^{-1}A_{12'} \\ 0 & \mathbb{1}_{2'2'} \end{pmatrix} \qquad (3)$$

check this by multiplying out:

$$= \begin{pmatrix} \mathbb{1}_{11} & 0 \\ A_{21}A_{11}^{-1} & \mathbb{1}_{22} \end{pmatrix} \begin{pmatrix} A_{11} & A_{12'} \\ 0 & [A/A_{11}] \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12'} \\ A_{21} & \underbrace{A_{21}\cancel{(A_{11})^{-1}}A_{12'} + [A/A_{11}]}_{\overset{(2)}{=} A_{22'}} \end{pmatrix}$$

(i) Schur determinant identity:

(if A_{11} and A_{22} are square and A_{11} is invertible)

$$\boxed{\det A = \det A_{11} \det[A/A_{11}]} \qquad (4)$$

Determinant of (3):

$$\det A = \det \begin{pmatrix} \mathbb{1}_{11} & 0 \\ A_{21}A_{11}^{-1} & \mathbb{1}_{22} \end{pmatrix} \det \begin{pmatrix} A_{11} & 0 \\ 0 & [A/A_{11}] \end{pmatrix} \det \begin{pmatrix} \mathbb{1}_{11} & A_{11}^{-1}A_{12} \\ 0 & \mathbb{1}_{22} \end{pmatrix} = \det A_{11} \det[A/A_{11}] \qquad (5)$$

(ii) Inverse of Schur complement:

$$\boxed{(A^{-1})_{22} = [A/A_{11}]^{-1}} \qquad (6)$$

Proof of (ii): Invert (3):

$$A^{-1} = \begin{pmatrix} \mathbb{1}_{11} & -A_{11}^{-1}A_{12} \\ 0 & \mathbb{1}_{22} \end{pmatrix} \begin{pmatrix} A_{11}^{-1} & 0 \\ 0 & [A/A_{11}]^{-1} \end{pmatrix} \begin{pmatrix} \mathbb{1}_{11} & 0 \\ -A_{21}A_{11}^{-1} & \mathbb{1}_{22} \end{pmatrix} \qquad (7)$$

The 22-component of (7) yields (6).

(iii) Remark for physicists: (6) can be used to derive self-energy of matrix Green's function:

Consider the 'Hamiltonian' matrix

$$H = H_0 + V = \begin{pmatrix} H_{11} & 0 \\ 0 & H_{22} \end{pmatrix} + \begin{pmatrix} 0 & H_{12} \\ H_{21} & 0 \end{pmatrix} \qquad (8)$$

Its 'Green's function' is the matrix

$$G(E) = (E - H)^{-1} \qquad (9)$$

It's 'projection' or 'restriction' to 22-space is:

$$\boxed{\begin{aligned}[G(E)]_{22} &= (E - H_{22} - \Sigma)^{-1} \qquad (10)\\ \Sigma &= H_{21}(E-H_{11})^{-1}H_{12} \qquad (11)\end{aligned}}$$

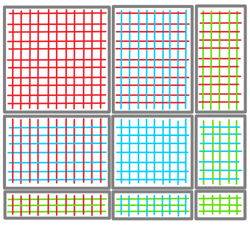
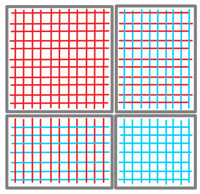
with 'self-energy'

Proof of (10), (11):

$$[G(E)]_{22} \overset{(9)}{=} [(E-H)^{-1}]_{22} \overset{(6)}{=} [(E-H)/(E-H)_{11}]^{-1} = \big[(E-H)_{22} - \underbrace{H_{21}[(E-H)_{11}]^{-1}H_{12}}_{\Sigma}\big]^{-1} \qquad (12)$$

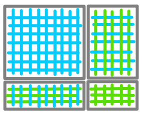
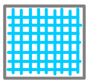
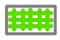
(iv) Schur quotient rule: When successively 'eliminating' more than one block, the order does not matter

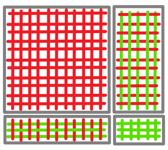
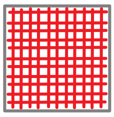
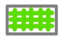
Consider A with subblock B: red, blue, green: rows & columns labeled 1, 2, 3

$$A = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} =$$  $$B \equiv \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} =$$  (13)

Assume A_{11}, A_{22} are square and invertible.

thin lines: original matrix
thicker lines: Schur complement
even thicker lines: two-fold Schur complement

$[A/A_{11}]$ =  , $[B/A_{11}]$ =  , $\big[[A/A_{11}]/[B/A_{11}]\big]$ =  (14)

$[A/A_{22}]$ =  , $[B/A_{22}]$ =  , $\big[[A/A_{22}]/[B/A_{22}]\big]$ =  (15)

Schur quotient rule: $$\big[[A/\cancel{A_{11}}]/[B/\cancel{A_{11}}]\big] = [A/B] = \big[[A/\cancel{A_{22}}]/[B/\cancel{A_{22}}]\big] \qquad (16)$$

For a proof of (iv), see [Fernandez2024, App. A.1] reminiscent of $(a/\cancel{c})/(b/\cancel{c}) = a/b$ (17)

Since order of block elimination does not matter, use a simpler notation:

$$\boxed{[[A/1]/2] = [A/(1,2)] = [[A/2]/1]} \qquad (18)$$

Here, /1 or /2 denotes elimination of square 11- or 22-block (w.r.t. original 3x3 block matrix), and /(1,2) the elimination of the square 2x2 block containing both.

Permutations of rows and columns in 11- and 22-blocks can be taken before or after taking the Schur complement $[A/(1,2)]$ without affecting the result.

For matrices involving larger number of blocks, iterative application of Schur quatient rule to successively eliminate blocks 11 to XX gives:

$$\boxed{\big[[[A/1]/2]\dots\big]/x\big] = [A/(1,2,\dots,x)]} \qquad (19)$$

(v) Restriction of Schur complement:

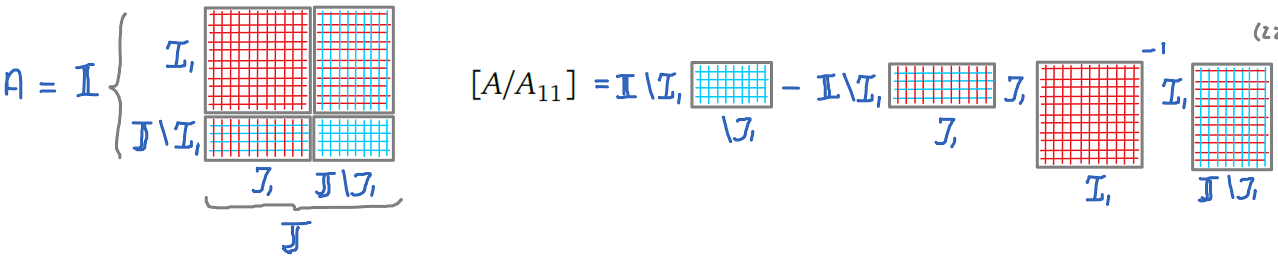
Restriction of Schur complement to limited numbers of rows and columns is equal to Schur complement of full matrix restricted to those rows and columns (plus the pivots):

Let $\mathcal{I}_1, \mathcal{J}_1$ specify Schur complement; its restriction to its $\mathcal{I}_2$ -rows and $\mathcal{J}_2$ -columns is given by:

$$\boxed{[A(\mathbb{I},\mathbb{J})/A(\mathcal{I}_1,\mathcal{J}_1)](\mathcal{I}_2,\mathcal{J}_2) = [A(\mathcal{I}_1 \cup \mathcal{I}_2, \mathcal{J}_1 \cup \mathcal{J}_2)/A(\mathcal{I}_1,\mathcal{J}_1)]} \quad \begin{array}{l} \mathcal{I}_1, \mathcal{I}_2 \subseteq \mathbb{I}, \quad \mathcal{J}_1, \mathcal{J}_2 \subseteq \mathbb{J} \\ \mathcal{I}_2 \subset \mathbb{I} \setminus \mathcal{I}_1, \quad \mathcal{J}_2 \subset \mathbb{J} \setminus \mathcal{J}_1 \end{array} \qquad (20)$$

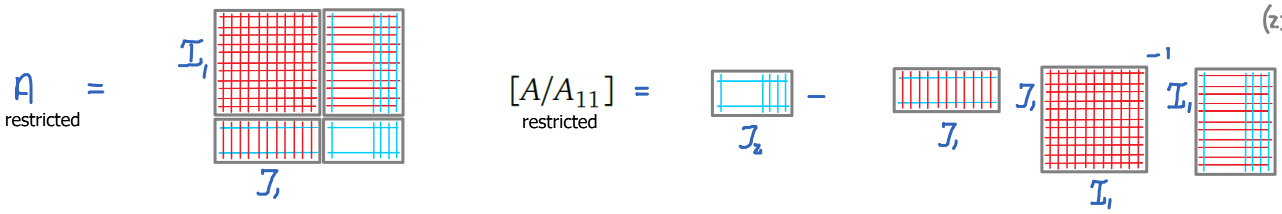
(v) follows directly from definition of Schur complement: $[A/A_{11}] \overset{(2)}{=} A_{22'} - A_{21}(A_{11})^{-1}A_{12'}$ (21)

(22)



For restricted version: retain only $\subset \mathbb{I}\setminus\mathcal{I}_1$ rows (blue) and only $\subset \mathcal{J}\setminus\mathcal{J}_1$ columns (blue):

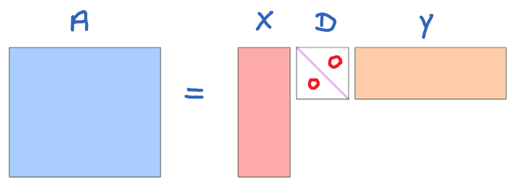
(23)



6. Partial rank-revealing LU decomposition (prrLU)

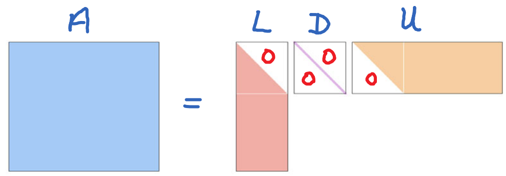
Decomposition $A = XDY$ is 'rank-revealing' if both X and Y are well-conditioned and D is diagonal.

rank χ = number of nonzero entries on diagonal of D



(1)

Standard LU decomposition: $A = LDU$
with L lower-triangular, U upper-triangular



(2)

Remark: LU decomposition implements Gaussian elimination for solving linear system of equations:

To solve $Ax = b$ for x, using $A = LU$, first solve $Ly = b$ for y, then $Ux = y$ for x. (3)

(4)

Partial rank-revealing LU (prrLU) algorithm computes LU decomposition in manner that is
(i) rank-revealing: largest remaining element is used for next pivot;
(ii) partial: process is stopped after constructing first $\tilde{\chi}$ columns of L and rows of U.

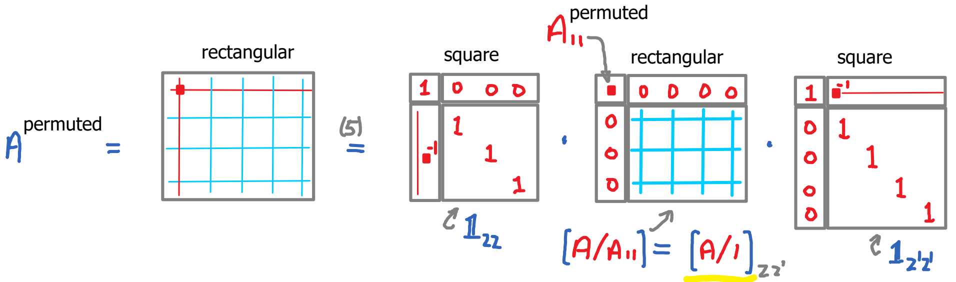
Recall general factorization of block matrix: [here, prime on second 2' indicates that dim(2') may differ from dim(2)]

$$A = \underset{\text{rectangular}}{\begin{pmatrix} A_{11} & A_{12'} \\ A_{21} & A_{22'} \end{pmatrix}} = \underset{\text{square}}{\begin{pmatrix} \mathbb{1}_{11} & 0 \\ A_{21}A_{11}^{-1} & \mathbb{1}_{22} \end{pmatrix}} \underset{\text{rectangular}}{\begin{pmatrix} A_{11} & 0 \\ 0 & [A/A_{11}]_{22'} \end{pmatrix}} \underset{\text{square}}{\begin{pmatrix} \mathbb{1}_{11} & A_{11}^{-1}A_{12'} \\ 0 & \mathbb{1}_{2'2'} \end{pmatrix}} \qquad (5)$$

(TCI.4.3)

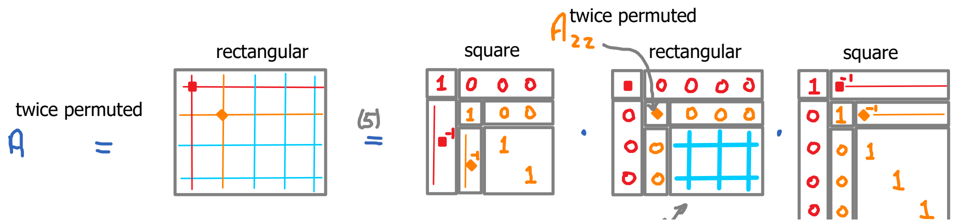
prrLU algorithm:

Permute rows and columns of A to move largest element (in modulus) = first pivot to top-left 11 position then apply (5) with 11-block of size 1x1, containing first pivot:

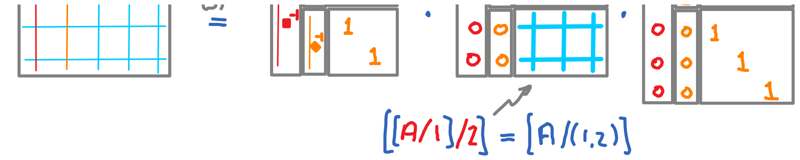


(6)

Permute rows and columns of $[A/1]$ to move largest element (in modulus) = second pivot to top-left 11 position then apply (5) with 11-block of size 1x1, containing second pivot:

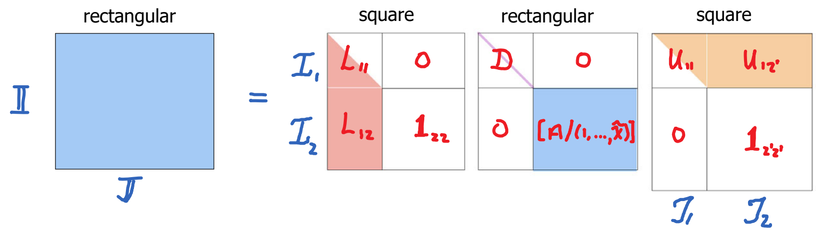


(7)

A =  (7)

After $\tilde{\chi}$ steps, we obtain prrLU decomposition of the form

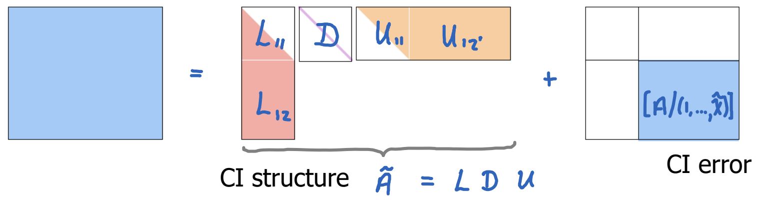
$$A^{\tilde{\chi}\text{ times permuted}} = \begin{pmatrix} L_{11} & 0 \\ L_{21} & \mathbb{1}_{22} \end{pmatrix} \begin{pmatrix} D & 0 \\ 0 & [A/(1,\ldots,\tilde{\chi})] \end{pmatrix} \begin{pmatrix} U_{11} & U_{12'} \\ 0 & \mathbb{1}_{2'2'} \end{pmatrix} \quad (8)$$

 (9)

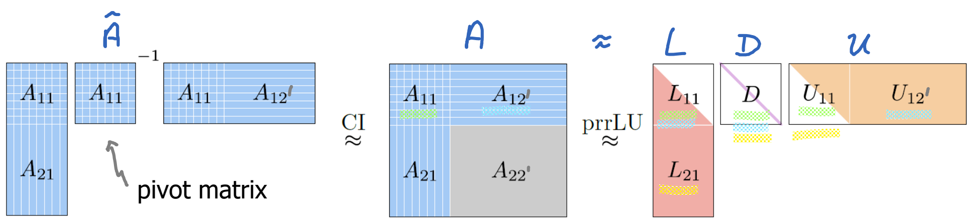
L_{11} and U_{11} are lower- and upper-triangular, with diagonal entries = 1.
D (shorthand for D_{11}) is diagonal, containing the maximal (in modulus) elements from each step.
Block subscripts label rows, columns with indices given by
1: $\mathcal{I}_1 = \mathcal{J}_1 = \{1, .., \tilde{\chi}\}$, (10a)
2: $\mathcal{I}_2 = \mathbb{I} \backslash \mathcal{I}_1$ (10b)
2': $\mathcal{J}_{2'} = \mathbb{J} \backslash \mathcal{J}_1$ (10c)

When Schur complement becomes zero, after χ steps, scheme terminates, identifying rank(A) = χ

For any $\chi < \tilde{\chi}$ (9) can be recast as:

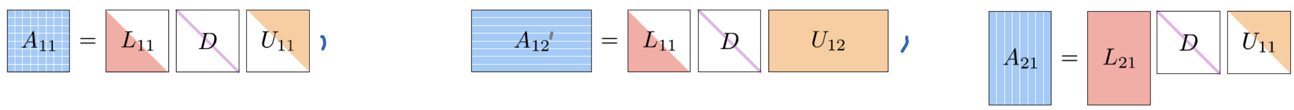
A =  (11)

Identify the CI ingredients:

 (12)

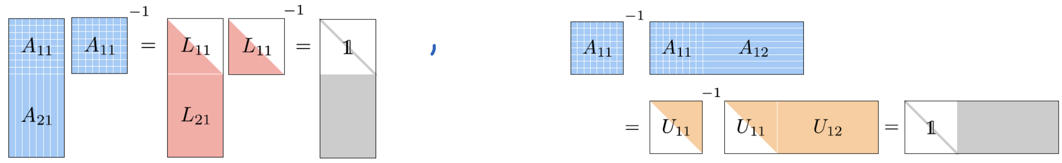
$$\begin{pmatrix} A_{11} \\ A_{12} \end{pmatrix} (A_{11})^{-1} (A_{11} \ \ A_{12'}) = \begin{pmatrix} L_{11} \\ L_{21} \end{pmatrix} D \,(U_{11} \ \ U_{12'}) = \begin{pmatrix} L_{11} D U_{11} \\ L_{21} D U_{11} \end{pmatrix} (L_{11} D U_{11})^{-1} (L_{11} D U_{11} \ \ L_{11} D U_{12'}) \quad (13)$$

$A_{11} = P = L_{11} D U_{11}$ $\qquad$ $A_{21} = L_{21} D U_{11}$ $\qquad$ $A_{12'} = L_{11} D U_{12'}$ (14)



$(A_{11})(A_{11})^{-1} = (\mathbb{1}_{11}$ $\qquad$ $(A_{..})^{-1}(A_{..} \ A_{..'}) = (\mathbb{1}_{..} \ \ U_{..}^{-1} U_{..'})$ (15)

$$\begin{pmatrix} A_{11} \\ A_{12} \end{pmatrix} (A_{11})^{-1} = \begin{pmatrix} \mathbb{1}_{11} \\ L_{21} L_{11}^{-1} \end{pmatrix} , \qquad (A_{11})^{-1} \begin{pmatrix} A_{11} & A_{12} \end{pmatrix} = \begin{pmatrix} \mathbb{1}_{11} & U_{11}^{-1} U_{12} \end{pmatrix} \tag{15}$$



(16)

prrLU is updatable: new rows and columns can be added easily.

Maximal pivot strategy of prrLU eliminates largest contribution to next Schur complement, hence reducing CI error. Hence, it is a simple, greedy algorithm for building near-maximum volume submatrix.

Search for maximal pivot in current Schur complement can be performed as full search, rook search or block rook search (see TCI.4).

Advantages of prrLU over direct CI:
- numerical stability - since construction and inversion of ill-conditioned pivot matrices is avoided.
- prrLU is more stable than QR-stabilization approach to CI.
- combination prrLU + block rook search allows efficient updates and has been found to work very reliably.