Eliminations and echelon forms in exact linear algebra

Clément PERNET,

INRIA-MOAIS, Grenoble Université, France

East Coast Computer Algebra Day,
University of Waterloo, ON, Canada,
April 9, 2011

Introduction

Gaussian elimination in Computer Algebra

Linear system solving: over $\mathbb{Z}_p$, $\mathbb{Z}$, $\mathbb{Q}$ (Crypto, Number Theory)

Polynomial system solving: Gröbner basis (Robotics, Crypto)

Linear dependencies: rank, basis (of vector spaces, free modules, Krylov spaces, ...) (Number Theory)

Determinants: certificate of similarity (Graph Theory)

Motivation

Designing efficient dense gaussian elimination routines over an exact ring/field.

- Extensively studied for numerical computations
- Specificities of exact computations:
  - No partial/full pivoting
  - Rank profile matters
- size of coefficients (e.g. compressed in GF(2)) $\Rightarrow$asymmetry

Study originating from, the design of the libraries

- `FFLAS-FFPACK`: word size finite fields
- `M4RI`: GF(2)

Outline

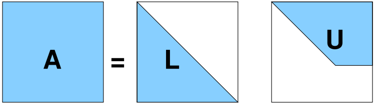
1. Decompositions and factorizations
   - Gaussian elimination based matrix decompositions
   - Relations between decompositions
2. Algorithms
   - Block recursive gaussian elimination
   - Memory allocations
   - Time complexity
3. Parallelization
4. Algorithms into practice: the case of GF(2)

Outline

1. Decompositions and factorizations
   - Gaussian elimination based matrix decompositions
   - Relations between decompositions
2. Algorithms
   - Block recursive gaussian elimination
   - Memory allocations
   - Time complexity
3. Parallelization
4. Algorithms into practice: the case of GF(2)

Outline

LU decomposition

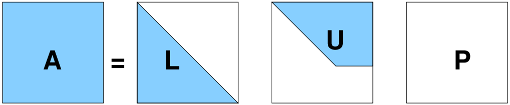


- L unit lower triangular,
- U non-sing upper triangular

Exists for

- matrices having the generic rank profile (every leading principal minor is non zero)

LUP, PLU decomposition

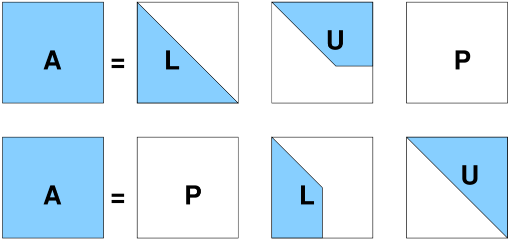


- P a permutation matrix

Exists for

- Any non-singular matrix
- Or any matrix with generic row rank profile

LUP, PLU decomposition

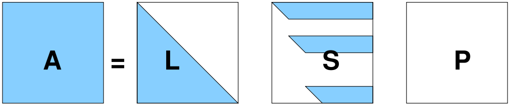


- *P* a permutation matrix

Exists for

- Any non-singular matrix
- Or any matrix with generic row rank profile

LSP, LQUP, PLUQ decompositions

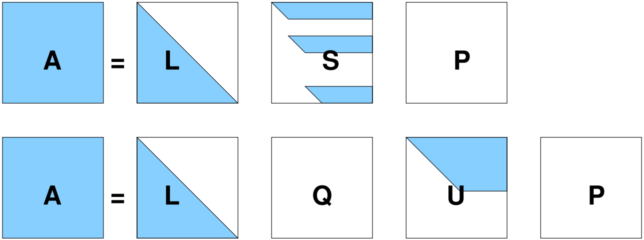


- S: semi-upper triangular,
- Q permutation matrix

Exists for

- any $m \times n$ matrix

LSP, LQUP, PLUQ decompositions

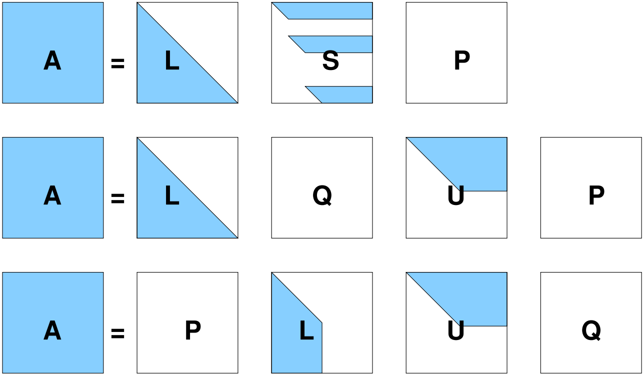


- S: semi-upper triangular,
- Q permutation matrix

Exists for

- any $m \times n$ matrix

LSP, LQUP, PLUQ decompositions



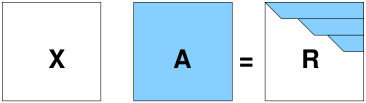
- S: semi-upper triangular,
- Q permutation matrix

Exists for

- any $m \times n$ matrix

Echelon form decomposition

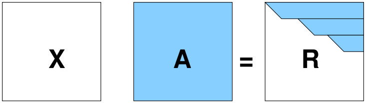
Row Echelon Form $XA = R$



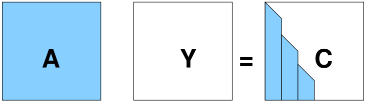
- X, Y: non-singular transformation matrices
- R, C: matrices in row/col echelon form

Echelon form decomposition

Row Echelon Form $XA = R$



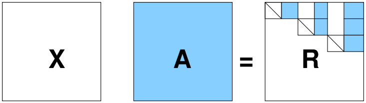
Column Echelon Form $AY = C$



- X, Y: non-singular transformation matrices
- R, C: matrices in row/col echelon form

Reduced echelon form decomposition

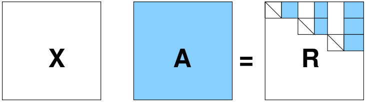
Row Reduced Echelon Form $XA = R$



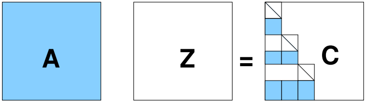
- X, Y: non-singular transformation matrices
- R, C: matrices in reduced row/col echelon form

Reduced echelon form decomposition

Row Reduced Echelon Form $XA = R$

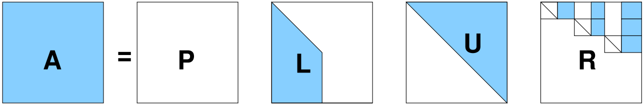


Column Reduced Echelon Form $AY = C$



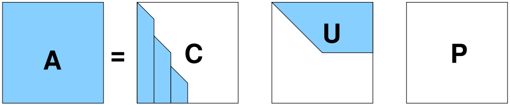
- X, Y: non-singular transformation matrices
- R, C: matrices in reduced row/col echelon form

Turing factorization



- PLU decomposition of X^{-1}, the inverse of the transformation matrix to REF

CUP and PLE decompositions

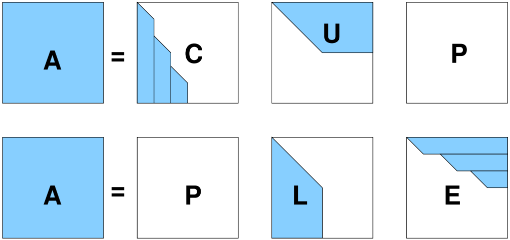


- C: column echelon form
- E: row echelon form

Exists for

- any $m \times n$ matrix

CUP and PLE decompositions



- C: column echelon form
- E: row echelon form

Exists for

- any $m \times n$ matrix

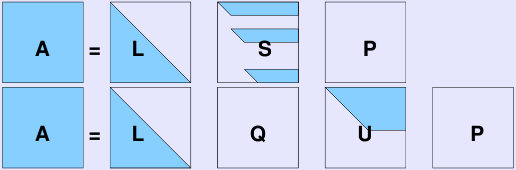
Outline

1. Decompositions and factorizations
   - Gaussian elimination based matrix decompositions
   - Relations between decompositions
2. Algorithms
   - Block recursive gaussian elimination
   - Memory allocations
   - Time complexity
3. Parallelization
4. Algorithms into practice: the case of GF(2)

Relations: up to permutations

From LSP to LQUP

$S = QU$



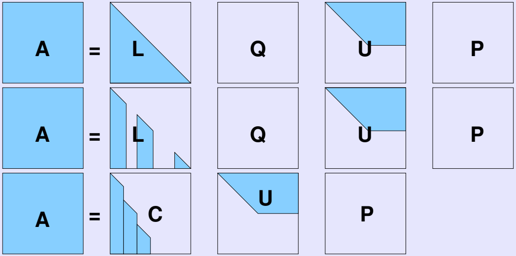
Fact

The first $r = rank(A)$ values of the permutation Q are monotonically increasing.

Relations: up to permutations

From LQUP to CUP

$C = LQ$

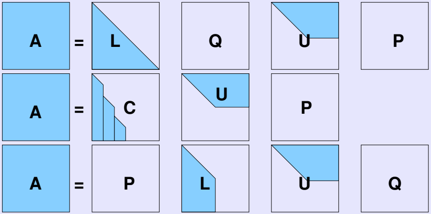


And to PLE using transposition: $PLE(A^T) = CUP(A)^T$

Relations:

From LQUP to PLUQ

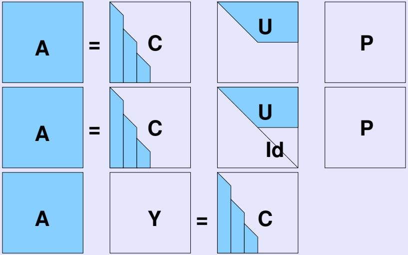
$$P \leftrightarrow Q, L \longleftarrow= Q^T L Q$$



Relations:

From CUP to ColumnEchelon form

$$Y = P^T \begin{bmatrix} U & \\ & I_{n-r} \end{bmatrix}^{-1}$$

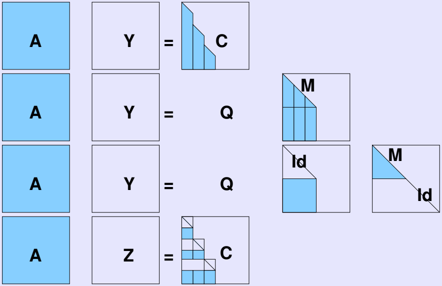


Similarly, from PLE to RowEchelon form

Relations:

From Column Echelon form to Reduced Column Echelon form

$$Z = Y \begin{bmatrix} M & \\ & I_{n-r} \end{bmatrix}^{-1}$$



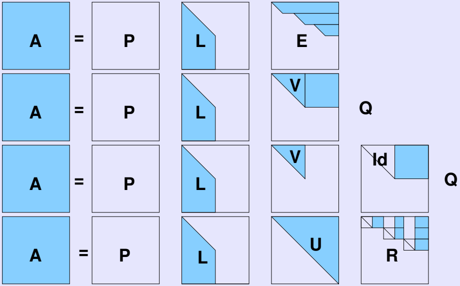
Similarly, from PLE to RowEchelon form

Relations:

From PLE to Turing

$$U = V,$$

$$R = \begin{bmatrix} V & \\ & I_{n-r} \end{bmatrix}^{-1} E$$



Outline

Algorithms: main types

Three ways to group operations:

1. simple iterative
   - Apply the standard Gaussian elimination in dimension n
   - Main loop **for** i=1 to n
2. block algorithms
   1. block iterative (Tile)
      - Apply Gaussian elimination in dimension n/k over blocks of size k
      - Main loop: **for** i=1 to n/k
   2. block recursive
      - Apply Gaussian elimination in dimension 2 recursively on blocks of size $n/2^i$
      - Main loop: **for** i=1 to 2

Type of algorithms

Data locality: prefer block algorithms

- cache aware: block iterative
- cache oblivious: block recursive

Base case efficieny: simple iterative

Asymptotic time complexity: block recursive

Parallelization: block iterative

Outline

Block recursive gaussian elimination

Author	Year	Computation	Requirement
Strassen	69	Inverse	gen. rank prof.
Bunch, Hopcroft	74	LUP	gen. row rank prof.
Ibarra, Moran, Hui	82	LSP, LQUP	none
Schönage, Keller-Gerig	85	StepForm	none
Storjohann	00	Echelon, RedEch	none
here	11	CUP,PLE,PLUQ	none

Block recursive gaussian elimination

Author	Year	Computation	Requirement
Strassen	69	Inverse	gen. rank prof.
Bunch, Hopcroft	74	LUP	gen. row rank prof.
Ibarra, Moran, Hui	82	LSP, LQUP	none
Schönage, Keller-Gerig	85	StepForm	none
Storjohann	00	Echelon, RedEch	none
here	11	CUP,PLE,PLUQ	none

Comparison according to

- No requirement on the input matrix

Block recursive gaussian elimination

Author	Year	Computation	Requirement
Strassen	69	Inverse	gen. rank prof.
Bunch, Hopcroft	74	LUP	gen. row rank prof.
Ibarra, Moran, Hui	82	LSP, LQUP	none
Schönage, Keller-Gerig	85	StepForm	none
Storjohann	00	Echelon, RedEch	none
here	11	CUP,PLE,PLUQ	none

Comparison according to

- No requirement on the input matrix
- Rank sensitive complexity

Block recursive gaussian elimination

Author	Year	Computation	Requirement
Strassen	69	Inverse	gen. rank prof.
Bunch, Hopcroft	74	LUP	gen. row rank prof.
Ibarra, Moran, Hui	82	LSP, LQUP	none
Schönage, Keller-Gerig	85	StepForm	none
Storjohann	00	Echelon, RedEch	none
here	11	CUP,PLE,PLUQ	none

Comparison according to

- No requirement on the input matrix
- Rank sensitive complexity
- Memory allocations
- Constant factor in the time complexity

Outline

Memory requirements:

Definition

In place = *output overrides the input and computation does not need extra memory (considering Matrix multiplication $C \leftarrow C + AB$ as a black box)*

Remark: a unit lower triangular and an upper triangular matrix can be stored on the same $m \times n$ storage!

Preliminaries

TRSM: TRiangular Solve with Matrix

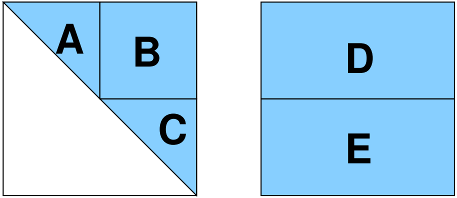
$$\begin{bmatrix} A & B \\ & C \end{bmatrix}^{-1} \begin{bmatrix} D \\ E \end{bmatrix} = \begin{bmatrix} A^{-1} & \\ & I \end{bmatrix} \begin{bmatrix} I & -B \\ & I \end{bmatrix} \begin{bmatrix} I & \\ & C^{-1} \end{bmatrix} \begin{bmatrix} D \\ E \end{bmatrix}$$

Preliminaries

TRSM: TRiangular Solve with Matrix

$$\begin{bmatrix} A & B \\ & C \end{bmatrix}^{-1} \begin{bmatrix} D \\ E \end{bmatrix} = \begin{bmatrix} A^{-1} & \\ & I \end{bmatrix} \begin{bmatrix} I & -B \\ & I \end{bmatrix} \begin{bmatrix} I & \\ & C^{-1} \end{bmatrix} \begin{bmatrix} D \\ E \end{bmatrix}$$

Compute $F = C^{-1}E$ (Recursive call)
Compute $G = D - BF$ (MM)
Compute $H = A^{-1}G$ (Recursive call)
Return $\begin{bmatrix} H \\ F \end{bmatrix}$



Preliminaries

TRSM: TRiangular Solve with Matrix

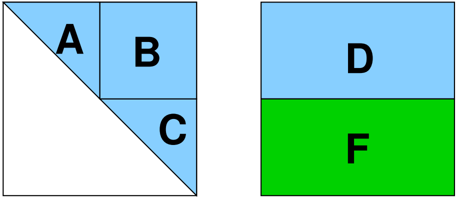
$$\begin{bmatrix} A & B \\ & C \end{bmatrix}^{-1} \begin{bmatrix} D \\ E \end{bmatrix} = \begin{bmatrix} A^{-1} & \\ & I \end{bmatrix} \begin{bmatrix} I & -B \\ & I \end{bmatrix} \begin{bmatrix} I & \\ & C^{-1} \end{bmatrix} \begin{bmatrix} D \\ E \end{bmatrix}$$

Compute $F = C^{-1}E$ (Recursive call)

Compute $G = D - BF$ (MM)

Compute $H = A^{-1}G$ (Recursive call)

Return $\begin{bmatrix} H \\ F \end{bmatrix}$

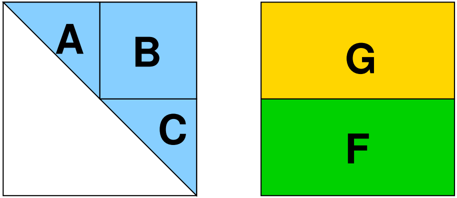


Preliminaries

TRSM: TRiangular Solve with Matrix

$$\begin{bmatrix} A & B \\ & C \end{bmatrix}^{-1} \begin{bmatrix} D \\ E \end{bmatrix} = \begin{bmatrix} A^{-1} & \\ & I \end{bmatrix} \begin{bmatrix} I & -B \\ & I \end{bmatrix} \begin{bmatrix} I & \\ & C^{-1} \end{bmatrix} \begin{bmatrix} D \\ E \end{bmatrix}$$

Compute $F = C^{-1}E$ (Recursive call)
Compute $G = D - BF$ (MM)
Compute $H = A^{-1}G$ (Recursive call)
Return $\begin{bmatrix} H \\ F \end{bmatrix}$



Preliminaries

TRSM: TRiangular Solve with Matrix

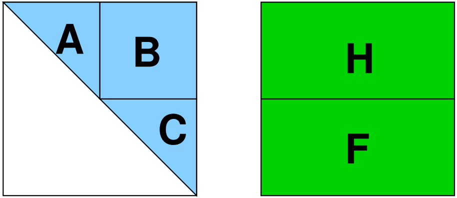
$$\begin{bmatrix} A & B \\ & C \end{bmatrix}^{-1} \begin{bmatrix} D \\ E \end{bmatrix} = \begin{bmatrix} A^{-1} & \\ & I \end{bmatrix} \begin{bmatrix} I & -B \\ & I \end{bmatrix} \begin{bmatrix} I & \\ & C^{-1} \end{bmatrix} \begin{bmatrix} D \\ E \end{bmatrix}$$

Compute $F = C^{-1}E$ (Recursive call)

Compute $G = D - BF$ (MM)

Compute $H = A^{-1}G$ (Recursive call)

Return $\begin{bmatrix} H \\ F \end{bmatrix}$

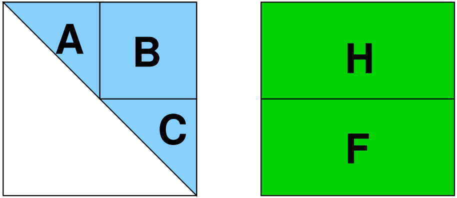


Preliminaries

TRSM: TRiangular Solve with Matrix

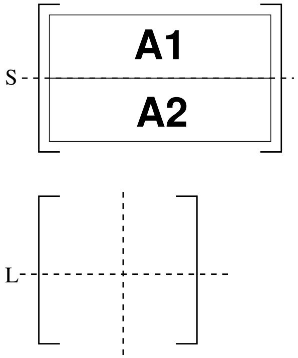
$$\begin{bmatrix} A & B \\ & C \end{bmatrix}^{-1} \begin{bmatrix} D \\ E \end{bmatrix} = \begin{bmatrix} {\color{red}A^{-1}} & \\ & I \end{bmatrix} \begin{bmatrix} I & -B \\ & I \end{bmatrix} \begin{bmatrix} I & \\ & {\color{red}C^{-}1} \end{bmatrix} \begin{bmatrix} D \\ E \end{bmatrix}$$

Compute $F = C^{-1}E$ (Recursive call)
Compute $G = D - BF$ (MM)
Compute $H = A^{-1}G$ (Recursive call)
Return $\begin{bmatrix} H \\ F \end{bmatrix}$



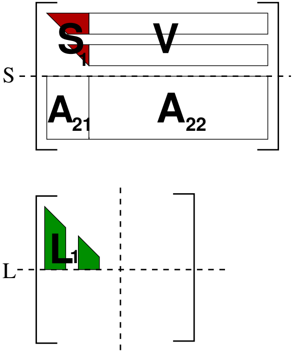
- $\mathcal{O}\left(n^{\omega}\right)$
- In place

The LSP algorithm



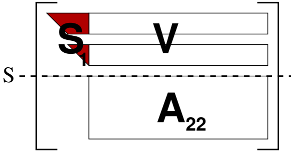
1. Split A Row-wise

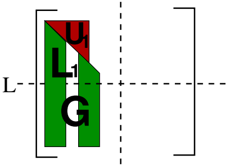
The LSP algorithm



1. Split A Row-wise
2. Recursive call on A_1

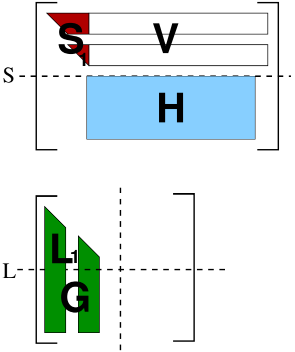
The LSP algorithm





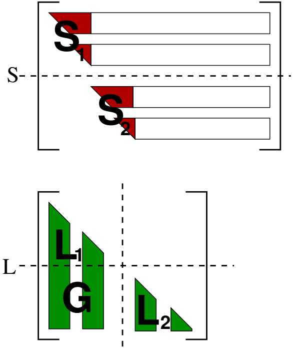
1. Split A Row-wise
2. Recursive call on A_1
3. $G \leftarrow A_{21} U_1^{-1}$ (trsm)

The LSP algorithm



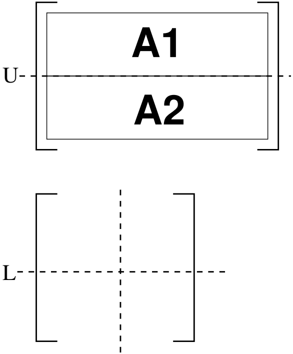
1. Split A Row-wise
2. Recursive call on A_1
3. $G \leftarrow A_{21} U_1^{-1}$ (trsm)
4. $H \leftarrow A_{22} - G \times V$ (MM)

The LSP algorithm



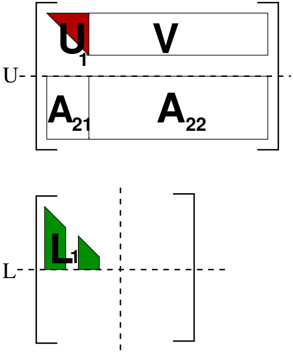
1. Split A Row-wise
2. Recursive call on A_1
3. $G \leftarrow A_{21} U_1^{-1}$ (trsm)
4. $H \leftarrow A_{22} - G \times V$ (MM)
5. Recursive call on H

The LQUP algorithm



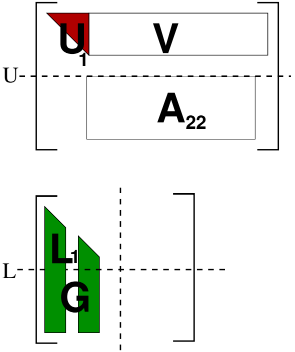
1. Split A Row-wise

The LQUP algorithm



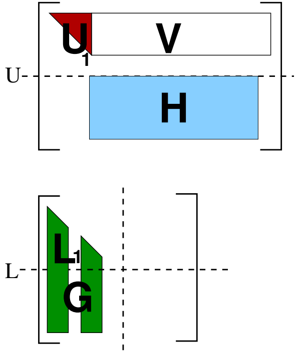
1. Split A Row-wise
2. Recursive call on A_1

The LQUP algorithm



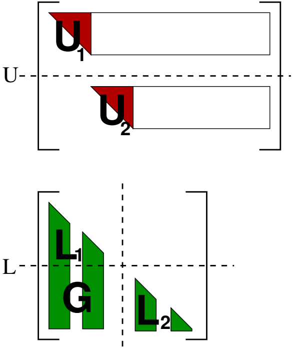
1. Split A Row-wise
2. Recursive call on A_1
3. $G \leftarrow A_{21} U_1^{-1}$ (`trsm`)

The LQUP algorithm



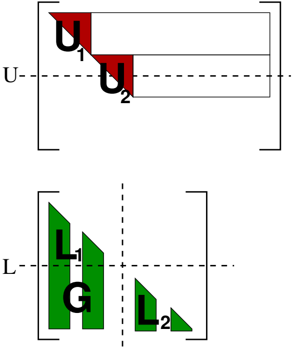
1. Split A Row-wise
2. Recursive call on A_1
3. $G \leftarrow A_{21} U_1^{-1}$ ($\texttt{trsm}$)
4. $H \leftarrow A_{22} - G \times V$ ($\texttt{MM1}$)

The LQUP algorithm



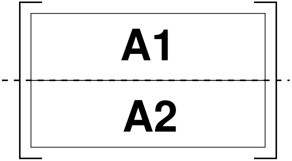
1. Split A Row-wise
2. Recursive call on A_1
3. $G \leftarrow A_{21} U_1^{-1}$ (trsm)
4. $H \leftarrow A_{22} - G \times V$ (MM1)
5. Recursive call on H

The LQUP algorithm



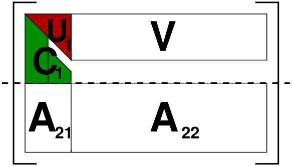
1. Split A Row-wise
2. Recursive call on A_1
3. $G \leftarrow A_{21} U_1^{-1}$ (trsm)
4. $H \leftarrow A_{22} - G \times V$ (MM1)
5. Recursive call on H

The CUP decomposition



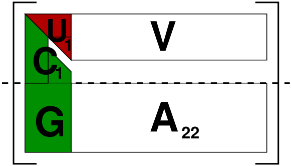
1. Split A Row-wise

The CUP decomposition



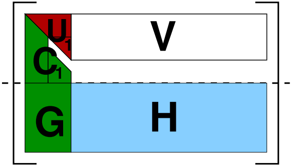
1. Split A Row-wise
2. Recursive call on A_1

The CUP decomposition



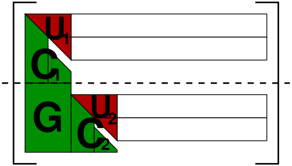
1. Split A Row-wise
2. Recursive call on A_1
3. $G \leftarrow A_{21} U_1^{-1}$ (`trsm`)

The CUP decomposition



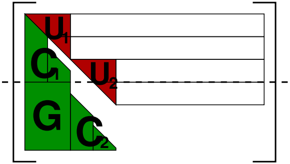
1. Split A Row-wise
2. Recursive call on A_1
3. $G \leftarrow A_{21} U_1^{-1}$ (`trsm`)
4. $H \leftarrow A_{22} - G \times V$ (MM)

The CUP decomposition



1. Split A Row-wise
2. Recursive call on A_1
3. $G \leftarrow A_{21} U_1^{-1}$ ($\texttt{trsm}$)
4. $H \leftarrow A_{22} - G \times V$ (MM)
5. Recursive call on H

The CUP decomposition



1. Split A Row-wise
2. Recursive call on A_1
3. $G \leftarrow A_{21} U_1^{-1}$ (`trsm`)
4. $H \leftarrow A_{22} - G \times V$ (MM)
5. Recursive call on H
6. Row permutations

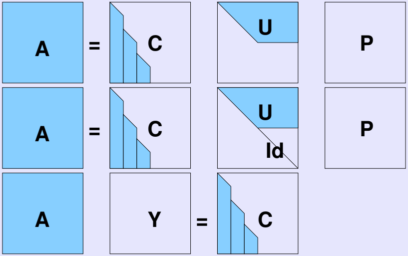
Memory: LSP vs LQUP vs PLUQ vs CUP

Decomposition	In place
LSP	N
LQUP	N
PLUQ	Y
CUP	Y

Echelon forms

From CUP to ColumnEchelon form

$$Y = P^T \begin{bmatrix} U_1 & U_2 \\ & I_{n-r} \end{bmatrix}^{-1}$$
$$= P^T \begin{bmatrix} U_1^{-1} & -U_1^{-1}U_2 \\ & I_{n-r} \end{bmatrix}$$



Additional operations:

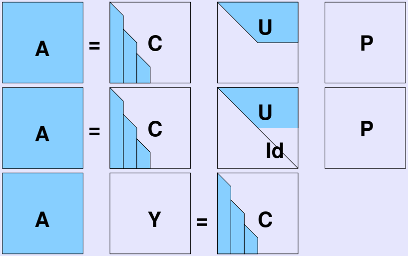
- $-U^{-1}U_2$ `trsm` (triangular system solve) in-place

Echelon forms

From CUP to ColumnEchelon form

$$Y = P^T \begin{bmatrix} U_1 & U_2 \\ & I_{n-r} \end{bmatrix}^{-1}$$

$$= P^T \begin{bmatrix} U_1^{-1} & -U_1^{-1} U_2 \\ & I_{n-r} \end{bmatrix}$$



Additional operations:

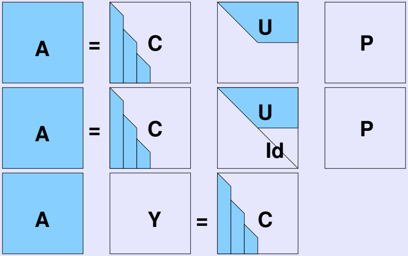
- $-U^{-1}U_2$ trsm (triangular system solve) in-place
- U_1^{-1}: trtri (triangular inverse)

Echelon forms

From CUP to ColumnEchelon form

$$Y = P^T \begin{bmatrix} U_1 & U_2 \\ & I_{n-r} \end{bmatrix}^{-1}$$

$$= P^T \begin{bmatrix} U_1^{-1} & -U_1^{-1} U_2 \\ & I_{n-r} \end{bmatrix}$$



Additional operations:

- $-U^{-1} U_2$ `trsm` (triangular system solve) in-place
- U_1^{-1}: `trtri` (triangular inverse) in-place

From LQUP to Column Echelon

TRTRI: triangular inverse

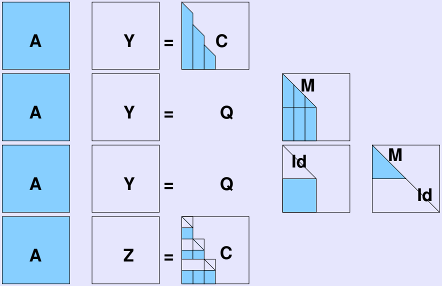
$$\begin{bmatrix} U_1 & U_2 \\ & U_3 \end{bmatrix}^{-1} = \begin{bmatrix} U_1^{-1} & -U_1^{-1} U_2 U_3^{-1} \\ & U_3^{-1} \end{bmatrix}$$

if $n = 1$ **then**
    $U \leftarrow U^{-1}$
else
    $U_2 \leftarrow U_3^{-1} U_2$ — TRSM
    $U_2 \leftarrow -U_2 U_3^{-1}$ — TRSM
    $U_1 \leftarrow U_1^{-1}$ — TRTRI
    $U_3 \leftarrow U_3^{-1}$ — TRTRI
end if

Reduced Echelon forms

From Column Echelon form to Reduced Column Echelon form

$$Z = Y \begin{bmatrix} M & \\ & I_{n-r} \end{bmatrix}^{-1}$$



Similarly, from PLE to RowEchelon form

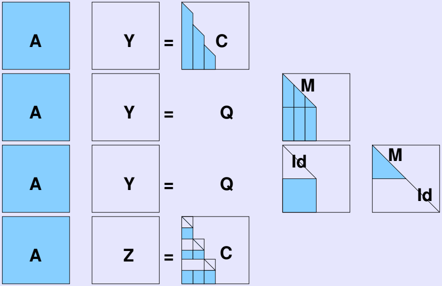
Again reduces to:

- $U^{-1}X$: TRSM, in-place
- U^{-1}: TRTRI, in-place

Reduced Echelon forms

From Column Echelon form to Reduced Column Echelon form

$$Z = Y \begin{bmatrix} M & \\ & I_{n-r} \end{bmatrix}^{-1}$$



Similarly, from PLE to RowEchelon form

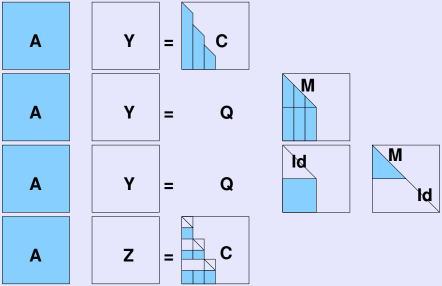
Again reduces to:

- $U^{-1}X$: TRSM, in-place
- U^{-1}: TRTRI, in-place
- UL: TRTRM,

Reduced Echelon forms

From Column Echelon form to Reduced Column Echelon form

$$Z = Y \begin{bmatrix} M & \\ & I_{n-r} \end{bmatrix}^{-1}$$



Similarly, from PLE to RowEchelon form

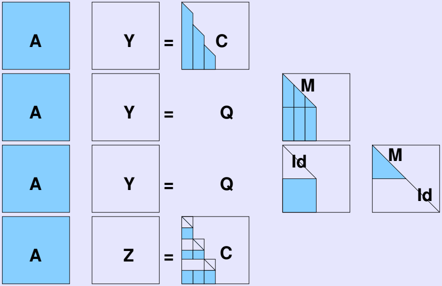
Again reduces to:

- $U^{-1}X$: TRSM, in-place
- U^{-1}: TRTRI, in-place
- UL: TRTRM,

Reduced Echelon forms

From Column Echelon form to Reduced Column Echelon form

$$Z = Y \begin{bmatrix} M & \\ & I_{n-r} \end{bmatrix}^{-1}$$



Similarly, from PLE to RowEchelon form

Again reduces to:

- $U^{-1}X$: TRSM, in-place
- U^{-1}: TRTRI, in-place
- UL: TRTRM, in-place

From Echelon to Reduced Echelon

TRTRM: triangular triangular multiplication

$$\begin{bmatrix} U_1 & U_2 \\ & U_3 \end{bmatrix} \begin{bmatrix} L_1 & \\ L_2 & L_3 \end{bmatrix} = \begin{bmatrix} U_1 L_1 + U_2 L_2 & U_2 L_3 \\ U_3 L_2 & U_3 L_3 \end{bmatrix}$$

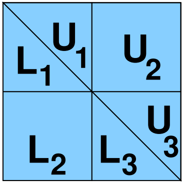
1: $X_1 \leftarrow U_1 L_1$ TRTRM
2: $X_1 \leftarrow X_1 + U_2 L_2$ MM
3: $X_2 \leftarrow U_2 L_3$ TRMM
4: $X_3 \leftarrow U_3 L_2$ TRMM
5: $X_4 \leftarrow U_3 L_3$ TRTRM

From Echelon to Reduced Echelon

TRTRM: triangular triangular multiplication

$$\begin{bmatrix} U_1 & U_2 \\ & U_3 \end{bmatrix} \begin{bmatrix} L_1 & \\ L_2 & L_3 \end{bmatrix} = \begin{bmatrix} U_1 L_1 + U_2 L_2 & U_2 L_3 \\ U_3 L_2 & U_3 L_3 \end{bmatrix}$$

1: $X_1 \leftarrow U_1 L_1$ TRTRM
2: $X_1 \leftarrow X_1 + U_2 L_2$ MM
3: $X_2 \leftarrow U_2 L_3$ TRMM
4: $X_3 \leftarrow U_3 L_2$ TRMM
5: $X_4 \leftarrow U_3 L_3$ TRTRM

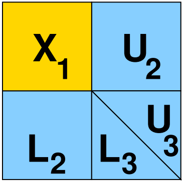


From Echelon to Reduced Echelon

TRTRM: triangular triangular multiplication

$$\begin{bmatrix} U_1 & U_2 \\ & U_3 \end{bmatrix} \begin{bmatrix} L_1 & \\ L_2 & L_3 \end{bmatrix} = \begin{bmatrix} U_1 L_1 + U_2 L_2 & U_2 L_3 \\ U_3 L_2 & U_3 L_3 \end{bmatrix}$$

1: $X_1 \leftarrow U_1 L_1$ TRTRM
2: $X_1 \leftarrow X_1 + U_2 L_2$ MM
3: $X_2 \leftarrow U_2 L_3$ TRMM
4: $X_3 \leftarrow U_3 L_2$ TRMM
5: $X_4 \leftarrow U_3 L_3$ TRTRM

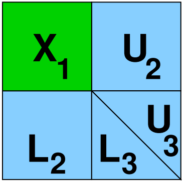


From Echelon to Reduced Echelon

TRTRM: triangular triangular multiplication

$$\begin{bmatrix} U_1 & U_2 \\ & U_3 \end{bmatrix} \begin{bmatrix} L_1 & \\ L_2 & L_3 \end{bmatrix} = \begin{bmatrix} U_1 L_1 + U_2 L_2 & U_2 L_3 \\ U_3 L_2 & U_3 L_3 \end{bmatrix}$$

1: $X_1 \leftarrow U_1 L_1$ TRTRM
2: $X_1 \leftarrow X_1 + U_2 L_2$ MM
3: $X_2 \leftarrow U_2 L_3$ TRMM
4: $X_3 \leftarrow U_3 L_2$ TRMM
5: $X_4 \leftarrow U_3 L_3$ TRTRM

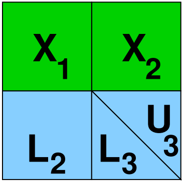


From Echelon to Reduced Echelon

TRTRM: triangular triangular multiplication

$$\begin{bmatrix} U_1 & U_2 \\ & U_3 \end{bmatrix} \begin{bmatrix} L_1 & \\ L_2 & L_3 \end{bmatrix} = \begin{bmatrix} U_1L_1 + U_2L_2 & U_2L_3 \\ U_3L_2 & U_3L_3 \end{bmatrix}$$

1: $X_1 \leftarrow U_1L_1$ TRTRM
2: $X_1 \leftarrow X_1 + U_2L_2$ MM
3: $X_2 \leftarrow U_2L_3$ TRMM
4: $X_3 \leftarrow U_3L_2$ TRMM
5: $X_4 \leftarrow U_3L_3$ TRTRM

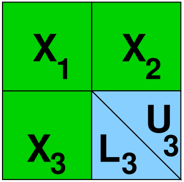


From Echelon to Reduced Echelon

TRTRM: triangular triangular multiplication

$$\begin{bmatrix} U_1 & U_2 \\ & U_3 \end{bmatrix} \begin{bmatrix} L_1 & \\ L_2 & L_3 \end{bmatrix} = \begin{bmatrix} U_1 L_1 + U_2 L_2 & U_2 L_3 \\ U_3 L_2 & U_3 L_3 \end{bmatrix}$$

1: $X_1 \leftarrow U_1 L_1$ TRTRM
2: $X_1 \leftarrow X_1 + U_2 L_2$ MM
3: $X_2 \leftarrow U_2 L_3$ TRMM
4: $X_3 \leftarrow U_3 L_2$ TRMM
5: $X_4 \leftarrow U_3 L_3$ TRTRM

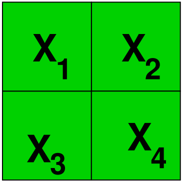


From Echelon to Reduced Echelon

TRTRM: triangular triangular multiplication

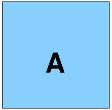
$$\begin{bmatrix} U_1 & U_2 \\ & U_3 \end{bmatrix} \begin{bmatrix} L_1 & \\ L_2 & L_3 \end{bmatrix} = \begin{bmatrix} U_1 L_1 + U_2 L_2 & U_2 L_3 \\ U_3 L_2 & U_3 L_3 \end{bmatrix}$$

1: $X_1 \leftarrow U_1 L_1$ TRTRM
2: $X_1 \leftarrow X_1 + U_2 L_2$ MM
3: $X_2 \leftarrow U_2 L_3$ TRMM
4: $X_3 \leftarrow U_3 L_2$ TRMM
5: $X_4 \leftarrow U_3 L_3$ TRTRM

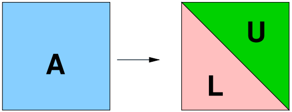


- $\mathcal{O}(n^\omega)$
- In place

Example: in place matrix inversion



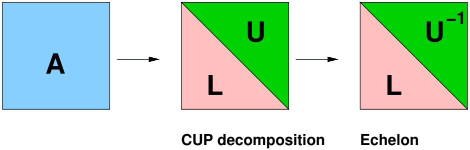
Example: in place matrix inversion



CUP decomposition

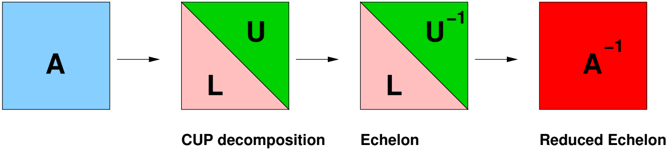
$$A = LU$$

Example: in place matrix inversion



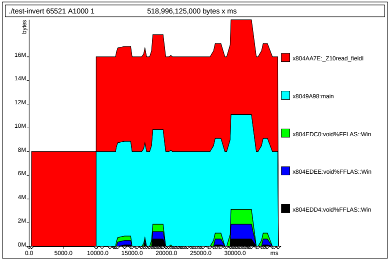
$$AU^{-1} = L$$

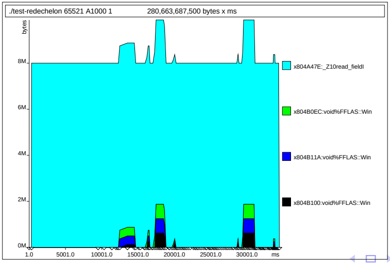
Example: in place matrix inversion



$$A(U^{-1}L^{-1}) = I$$

Experiments





Direct computation of the Reduced Echelon form

- Strassen 69: inverse of generic matrices
- Storjohann 00: `Gauss-Jordan` generalization for any rank profile

Matrix Inversion [Strassen 69]

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} & \\ & I \end{bmatrix} \begin{bmatrix} I & -B \\ & I \end{bmatrix} \begin{bmatrix} I & \\ & (D - CA^{-1}B)^{-1} \end{bmatrix} \begin{bmatrix} I & \\ CA^{-1} & I \end{bmatrix}$$

Direct computation of the Reduced Echelon form

- Strassen 69: inverse of generic matrices
- Storjohann 00: `Gauss-Jordan` generalization for any rank profile

Matrix Inversion [Strassen 69]

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} & \\ & I \end{bmatrix} \begin{bmatrix} I & -B \\ & I \end{bmatrix} \begin{bmatrix} I & \\ & (D - CA^{-1}B)^{-1} \end{bmatrix} \begin{bmatrix} I & \\ CA^{-1} & I \end{bmatrix}$$

1: Compute $E = A^{-1}$ (Recursive call)
2: Compute $F = D - CEB$ (MM)
3: Compute $G = F^{-1}$ (Recursive call)
4: Compute $H = -EB$ (MM)
5: Compute $J = HG$ (MM)
6: Compute $K = CE$ (MM)
7: Compute $L = E + JK$ (MM)
8: Compute $M = GK$ (MM)
9: Return $\begin{bmatrix} E & J \\ M & G \end{bmatrix}$

Strassen-Storjohann's `Gauss-Jordan` elimination

Problem

Needs to perform operations of the form $A \leftarrow AB$

$\Rightarrow$ *not doable in place by a usual matrix multiplication algorithm*

Strassen-Storjohann's `Gauss-Jordan` elimination

Problem

Needs to perform operations of the form $A \leftarrow AB$
$\Rightarrow$ not doable in place by a usual matrix multiplication algorithm

Workaround [Storjohann]:

1. Decompose $B = LU$ `LU`
2. $A \leftarrow AL$ `trmm`
3. $A \leftarrow AU$ `trmm`

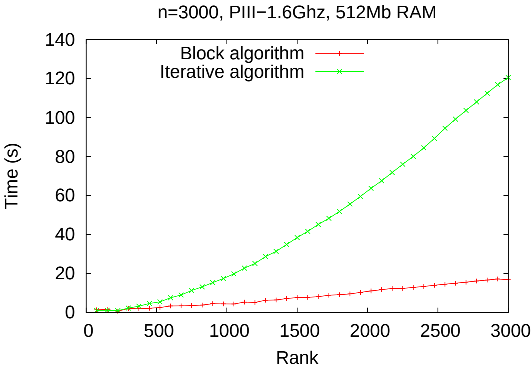
Outline

1. Decompositions and factorizations
   - Gaussian elimination based matrix decompositions
   - Relations between decompositions
2. Algorithms
   - Block recursive gaussian elimination
   - Memory allocations
   - Time complexity
3. Parallelization
4. Algorithms into practice: the case of GF(2)

Rank sensitive time complexity

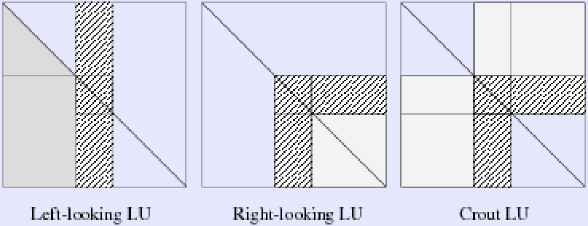
Fact

Algorithms LSP, CUP, LQUP, PLUQ, ... have a rank sensitive computation time: $\mathcal{O}\left(mnr^{\omega-2}\right)$



Rank sensitivity and Left/Right/Crout Looking variants

Right looking / Left looking/ Crout variants



- does not affect 2x2 splitting
- for block iterative: always the same rank sensitive complexity:

$$2n^2r - 2nr^2 + 2/3r^3$$

Time complexity: comparing constants

$$\mathcal{O}(n^\omega) = C_\omega n^3$$

Algorithm	Constant C_ω	C_3	$C_{\log_2 7}$	in-place
MM	C_ω	2	6	$\times$
TRSM	$\frac{C_\omega}{2^{\omega-1}-2}$	1	4	$\vee$
TRTRI	$\frac{C_\omega}{(2^{\omega-1}-2)(2^{\omega-1}-1)}$	$\frac{1}{3} \approx 0.33$	$\frac{8}{5} = 1.6$	$\vee$
TRTRM, CUP PLUQ LQUP,	$\frac{C_\omega}{2^{\omega-1}-2} - \frac{C_\omega}{2^\omega-2}$	$\frac{2}{3} \approx 0.66$	$\frac{14}{5} = 2.8$	$\vee$
Echelon	$\frac{C_\omega}{2^{\omega-2}-1} - \frac{3C_\omega}{2^\omega-2}$	1	$\frac{22}{5} \approx 4.4$	$\vee$
RedEchelon	$\frac{C_\omega(2^{\omega-1}+2)}{(2^{\omega-1}-2)(2^{\omega-1}-1)}$	2	$\frac{44}{5} = 8.8$	$\vee$
StepForm	$\frac{5C_\omega}{2^{\omega-1}-1} + \frac{C_\omega}{(2^{\omega-1}-1)(2^{\omega-2}-1)}$	4	$\frac{76}{5} = 15.2$	$\times$
GJ*	$\frac{C_\omega}{2^{\omega-2}-1}$	2	8	$\times$

*: GJ: GaussJordan alg of [Storjohann00] computing the reduced echelon form

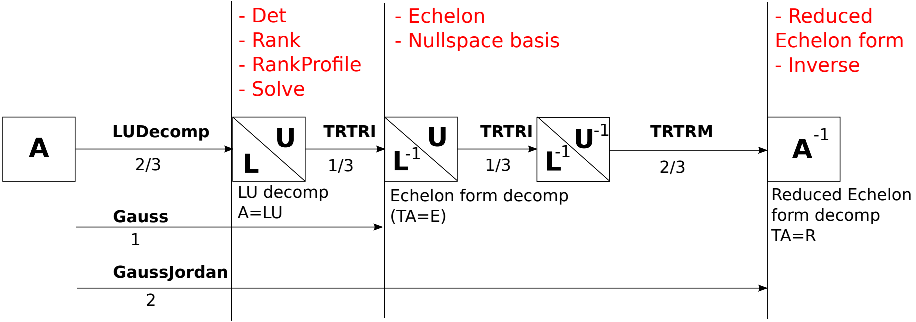
Applications to standard linalg problems

Problem	Using	C_ω	C_3	$C_{\log_2 7}$	In place

`Rank`
`RankProfile`
`IsSingular`
`Det`
`Solve`
	`GJ`
	`CUP`
		$\frac{C_\omega}{2^{\omega-2}-1}$
		$\frac{C_\omega}{2^{\omega-1}-2} - \frac{C_\omega}{2^\omega-2}$
			2
			0.66
				8
				2.8
					$\times$
					V

`Inverse`	`GJ` `CUP`	$\frac{C_\omega}{2^{\omega-2}-1}$ $\frac{C_\omega(2^{\omega-1}+2)}{(2^{\omega-1}-2)(2^{\omega-1}-1)}$	2 2	8 8.8	$\times$ V

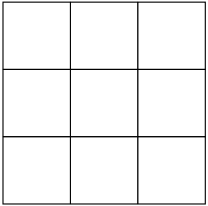
Summary


- Det
- Rank
- RankProfile
- Solve
- Echelon
- Nullspace basis
- Reduced Echelon form
- Inverse
A
LUDecomp
2/3
L U
TRTRI
1/3
L-1 U
TRTRI
1/3
L-1 U-1
TRTRM
2/3
A-1
LU decomp
A=LU
Echelon form decomp
(TA=E)
Reduced Echelon form decomp
TA=R
Gauss
1
GaussJordan
2

Outline

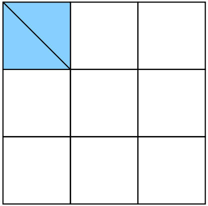
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



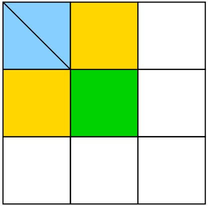
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of *n* on the diagonal



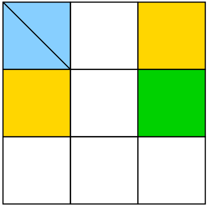
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



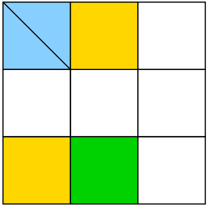
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



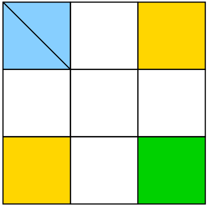
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



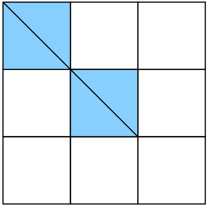
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



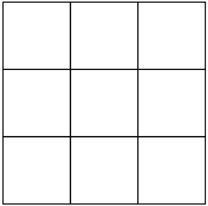
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



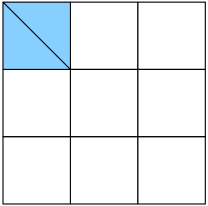
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



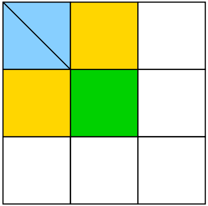
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



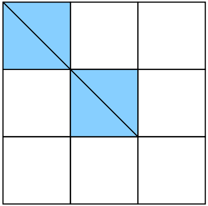
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



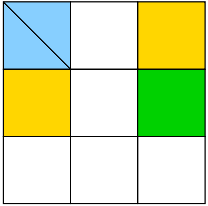
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



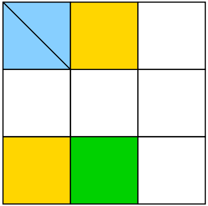
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



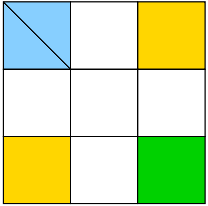
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



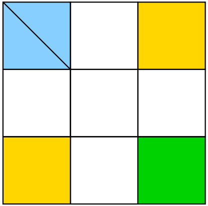
Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



Parallelization

- Using block iterative algorithm
- Parallelizing the matrix multiplication updates
- Always a critical path of n on the diagonal



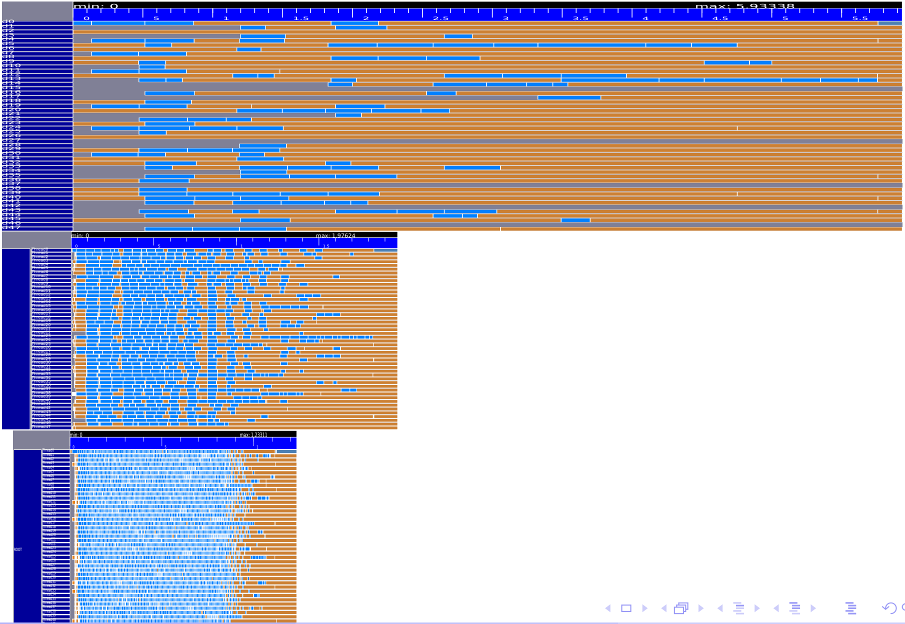
Scheduling tasks

- DAG, static scheduling, ...
- Dynamic scheduling, work-stealing

Advantage of exact computations:

- No need to update pivots (local pivoting) $\Rightarrow$ less ops
- Multi-frontal approach made easier

Dynamic scheduling with Kaapi



Outline

Dedicated implementations over GF(2)

The `M4RI` library [M Albrecht & Al.]:

Dense basic linear algebra over GF(2)

- Packed representation: `unsigned long long` is a vector of 64 coefficients
- Matrix multiplication:
  - Gray code table lookup (Methode of the 4 russians)
    $\Rightarrow \mathcal{O}\left(n^3/\log n\right)$
  - Strassen on the coarse grain.
    $\Rightarrow \mathcal{O}\left((n/k)^\omega k^3/\log k\right) = \mathcal{O}\left(n^\omega\right)$

Design of the gaussian elimination routine

PLE vs CUP

- Row major storage $\Rightarrow$easier to permute pivots along columns
  $\Rightarrow$PLE rather than CUP

PLE vs PLUQ

- PLUQ involves more back and forth pivoting
- Compact LAPACK representation of permutation: product of transpositions
  - Not possible to maintain for P in $\mathcal{O}(1)$ PLUQ

  $\Rightarrow$PLE rater than PLUQ

Design of the gaussian elimination routine

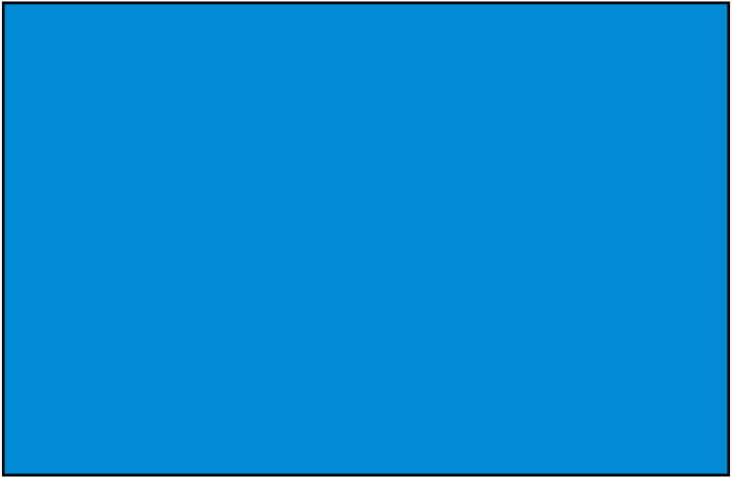
Structure of the algorithm

- block recursive PLE on the coarse grain ($\mathcal{O}(n^\omega)$)
- block iterative PLE with block size $k = \log n$
- simple iterative PLE on cache fitting blocks

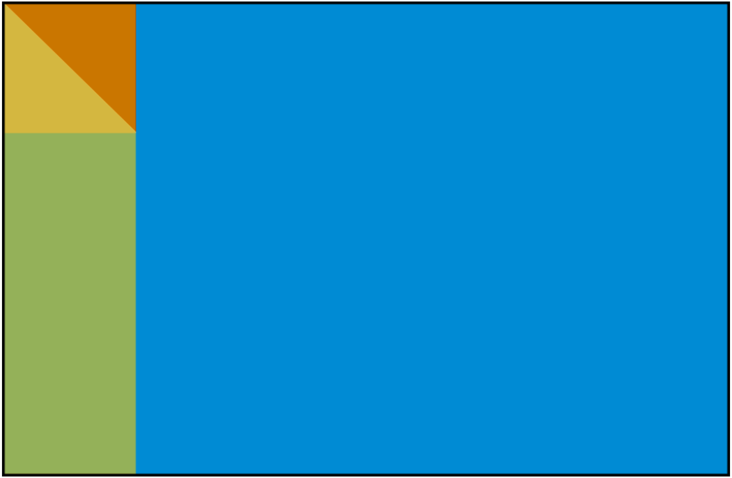
Additional tricks:

- With gray code tables: inverting means reverse table look-up
  $\Rightarrow$ `TRSM` as efficient as `MM`
- ...

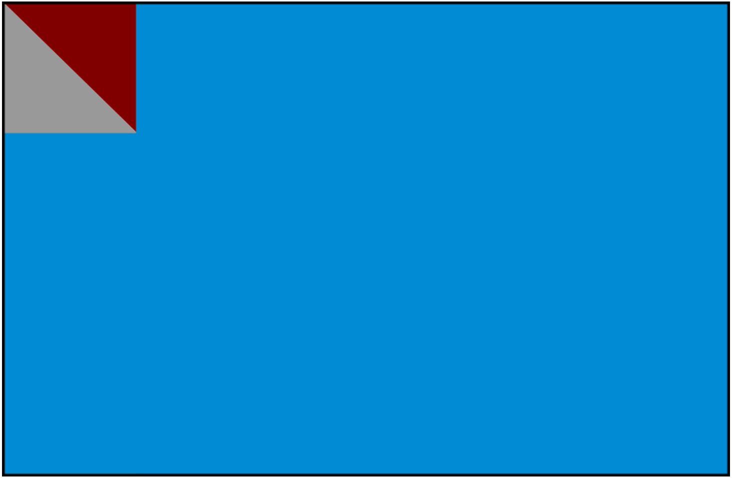
Visualisation



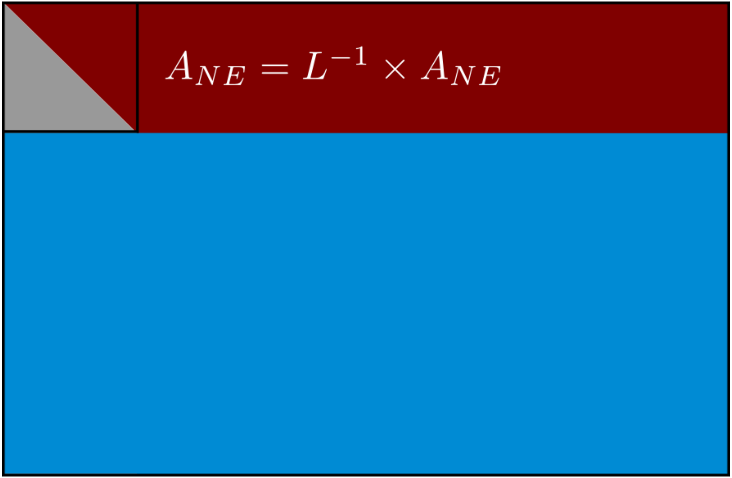
Visualisation



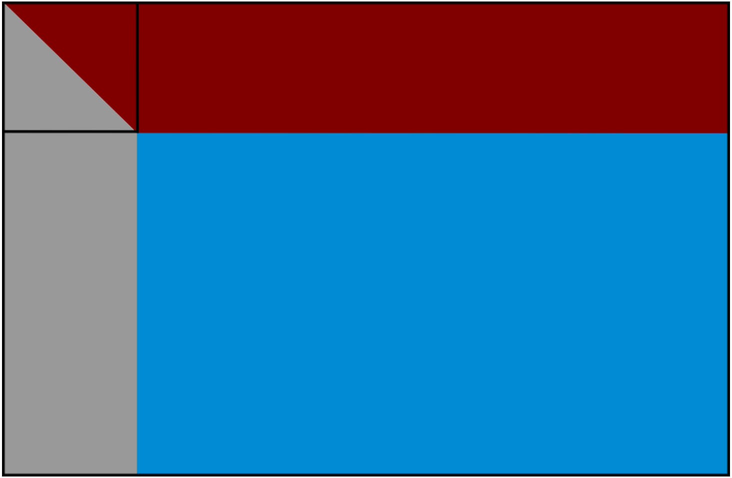
Visualisation



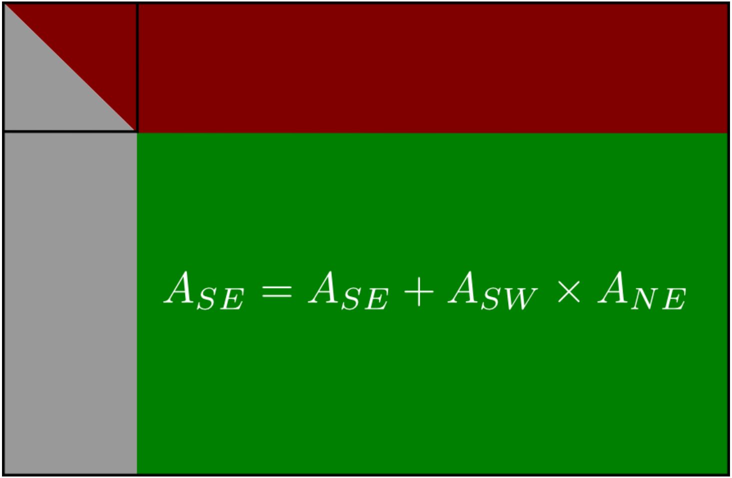
Visualisation



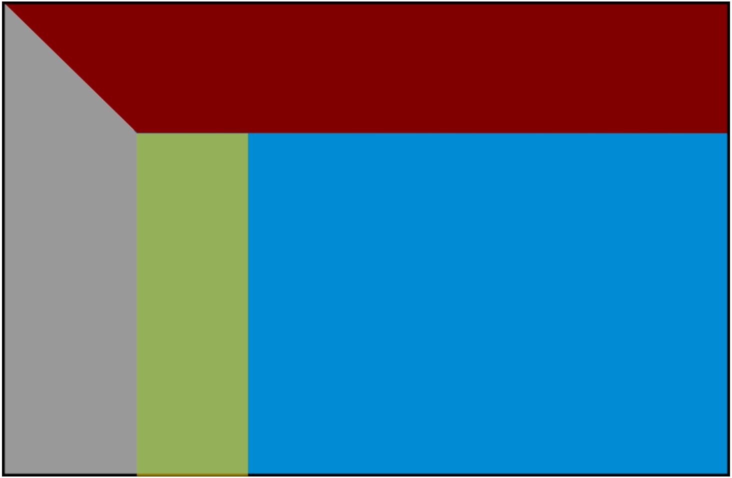
Visualisation



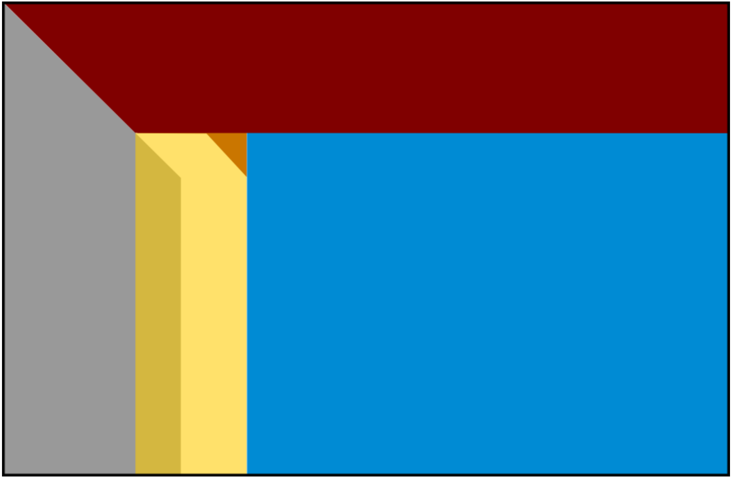
Visualisation



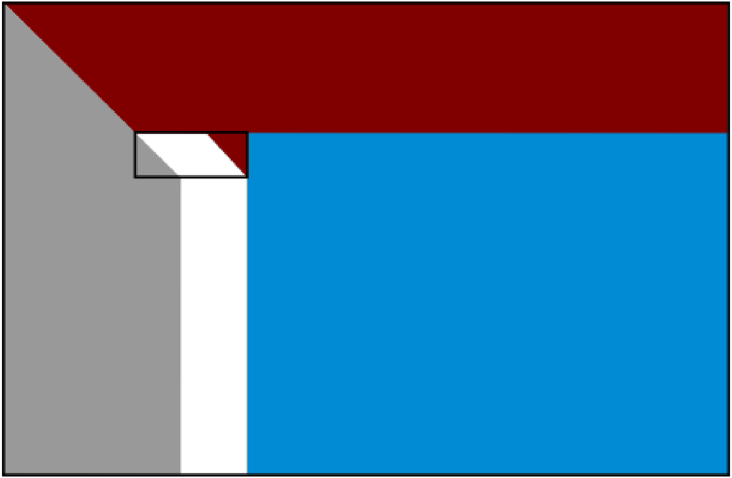
Visualisation



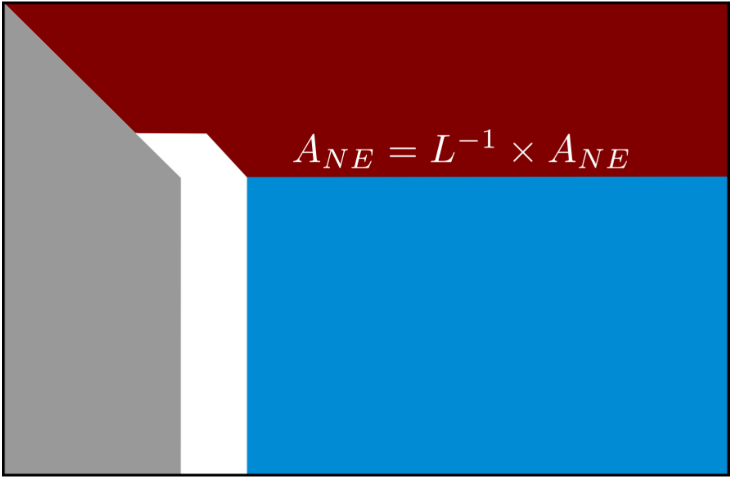
Visualisation



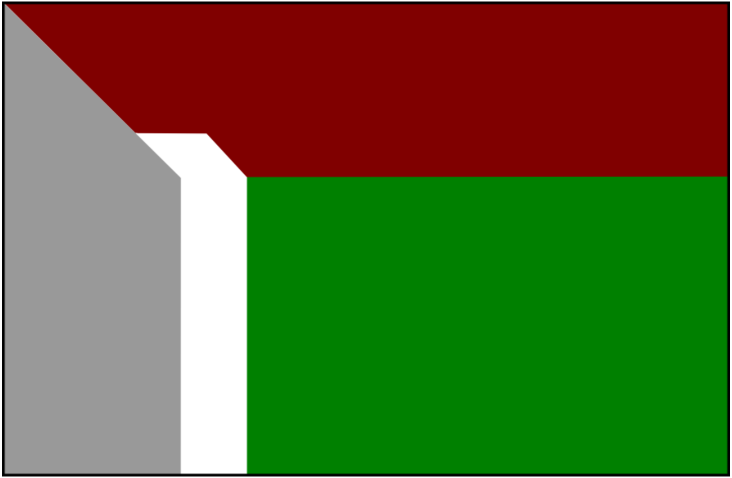
Visualisation



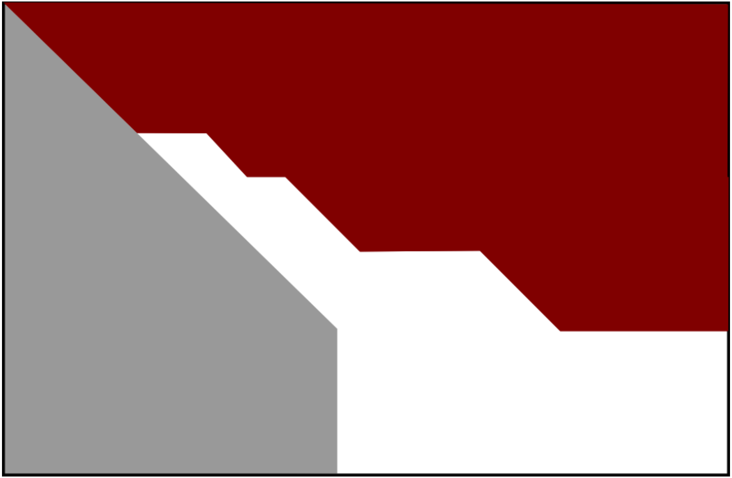
Visualisation



Visualisation



Visualisation



Results: Reduced Row Echelon Form

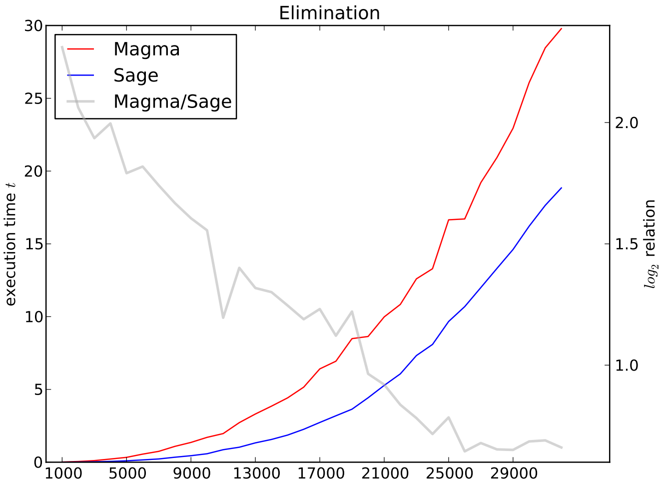


Figure: 2.66 Ghz Intel i7, 4GB RAM

Results: Row Echelon Form

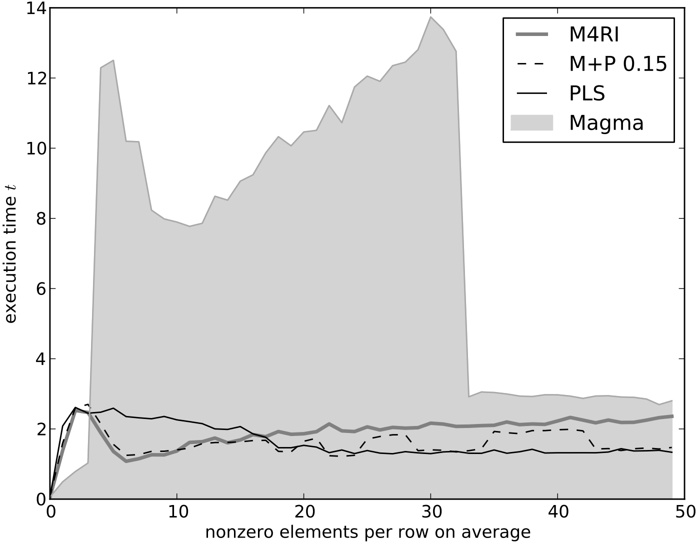
Using one core we can compute the echelon form of a $500,000 \times 500,000$ dense random matrix over $\mathbb{F}_2$ in

$$9711.42 \text{ seconds } = 2.7 \text{ hours.}$$

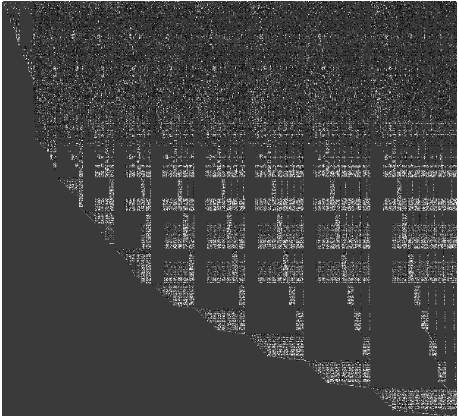
Using four cores decomposition we can compute the echelon form of a random dense $500,000 \times 500,000$ matrix in

$$3806.28 \text{ seconds } = 1.05 \text{ hours.}$$

Work-in-Progress: Sensitivity to Sparsity



Work-in-Progress: Gröbner Basis Linear Algebra



			64-bit Debian/GNU Linux, 2.6Ghz Opteron)				
Problem	Matrix Dimension	Density	Magma 2.15-10	M4RI 20100324	PLS 20100324	M+P 0.15 20100429	M+P 0.20 20100429
HFE 25	$12,307 \times 13,508$	0.076	4.57s	3.28s	3.45s	**3.03s**	3.21s
HFE 30	$19,907 \times 29,323$	0.067	33.21s	**23.72s**	25.42s	23.84s	25.09s
HFE 35	$29,969 \times 55,800$	0.059	278.58s	126.08s	159.72s	154.62s	**119.44s**
MXL	$26,075 \times 26,407$	0.185	76.81s	23.03s	19.04s	**17.91s**	18.00s

Work-in-Progress: Multi-core Support