High Performance and Reliable Algebraic Computing

Soutenance d'habilitation à diriger des recherches

Clément Pernet

Université Joseph Fourier (Grenoble 1)

November 25, 2014

Rapporteurs :	Daniel Augot, Mark Giesbrecht, Laura Grigori,	Examinateurs :	Jean-Guillaume Dumas, Jean-Charles Faugère, Erich L. Kaltofen, Brigitte Plateau.

Introduction

Computer Algebra

Computing exactly over $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathsf{GF}(q), \mathsf{K}[X]$.

- Symbolic manipulations.
- Applications where all digits matter:

- breaking Discrete Log Pb. in quasi-polynomial time [Barbulescu & *al.* 14],
- building modular form databases to test the BSD conjecture [Stein 12],
- formal verification of Hales' proof of Kepler conjecture [Hales 05].

Introduction

Computer Algebra

Computing exactly over $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathsf{GF}(q), \mathsf{K}[X]$.

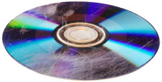
- Symbolic manipulations.
- Applications where all digits matter:

- breaking Discrete Log Pb. in quasi-polynomial time [Barbulescu & *al.* 14],
- building modular form databases to test the BSD conjecture [Stein 12],
- formal verification of Hales' proof of Kepler conjecture [Hales 05].

Efficiency mostly rely on linear algebra over $\mathbb{Z}$ and $\mathbb{Z}/p\mathbb{Z}$.

Introduction

Coding theory

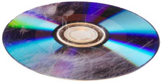


Protecting information against alteration:

- deep space communication,
- data storage,
- fault tolerance of large scale computations.

Introduction

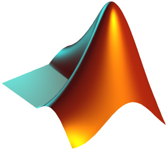
Coding theory



Protecting information against alteration:

- deep space communication,
- data storage,
- fault tolerance of large scale computations.

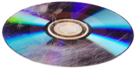
Numerical linear algebra



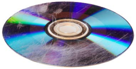
Computing fast with approximations:

- delivering flops to most scientific computations for over 60 years,
- `LinPack`: benchmark for the top 500 supercomputers,
- impacts nowadays computer architectures.

Interactions



Interactions

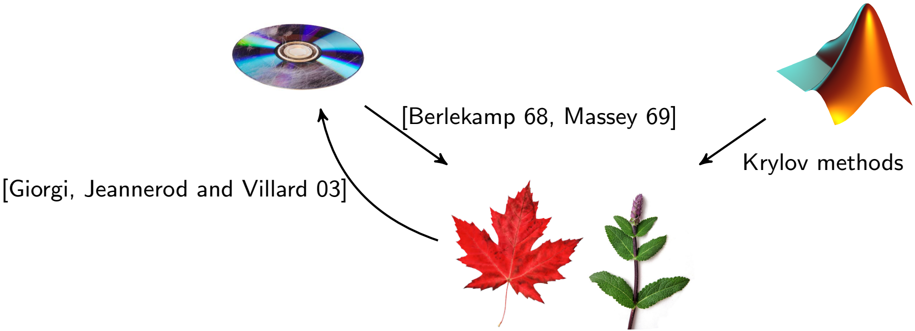


[Berlekamp 68, Massey 69]

Krylov methods

[Wiedemann 86]: sparse linear system solving over $\mathbb{F}_q$

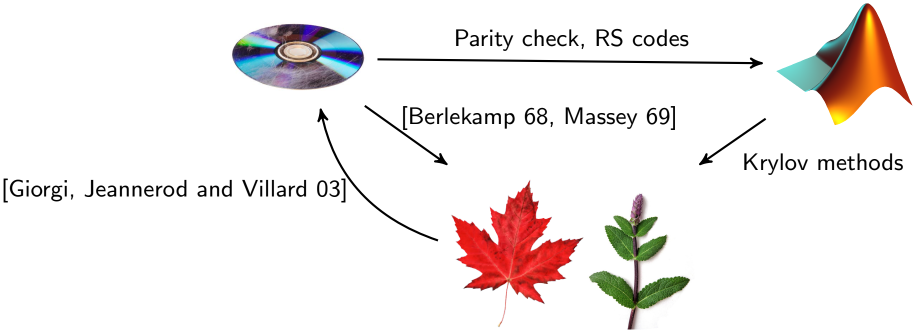
Interactions



[Wiedemann 86]: sparse linear system solving over $\mathbb{F}_q$

[Chowdhury & *al.* 14]: fast list decoding of Reed-Solomon codes

Interactions

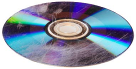


[Wiedemann 86]: sparse linear system solving over $\mathbb{F}_q$

[Chowdhury & *al.* 14]: fast list decoding of Reed-Solomon codes

[Huang and Abraham 84]: Algorithm Based Fault Tolerance (ABFT)

Interactions

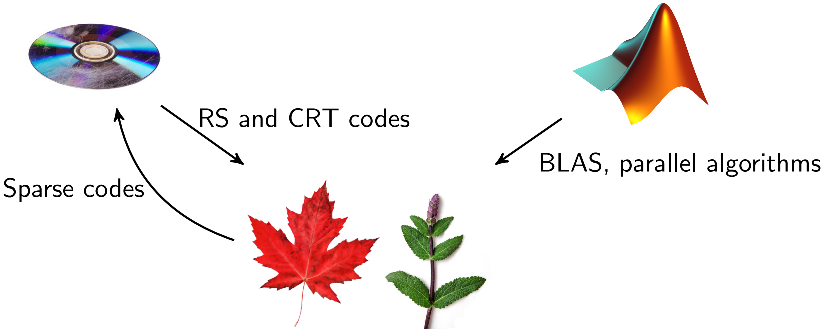


BLAS, parallel algorithms

Contributions:

- design of high performance linear algebra kernels,

Interactions



Contributions:

- design of high performance linear algebra kernels,
- fault tolerant computer algebra.

Outline

1. Design of High Performance Exact Linear Algebra Kernels
   - Matrix multiplication
   - Gaussian elimination
   - Rank profiles
   - Characteristic polynomial
2. Coding Theory for Fault Tolerant Computer Algebra
   - Approximation problems
   - Dense polynomial evaluation codes
   - Rational function codes
   - Sparse evaluation codes

Outline

1. Design of High Performance Exact Linear Algebra Kernels
   - Matrix multiplication
   - Gaussian elimination
   - Rank profiles
   - Characteristic polynomial
2. Coding Theory for Fault Tolerant Computer Algebra
   - Approximation problems
   - Dense polynomial evaluation codes
   - Rational function codes
   - Sparse evaluation codes

Reductions: linear algebra's arithmetic complexity

< 1969: $O(n^3)$ for everyone (Gauss, Householder, Danilevskĭi, etc)

Reductions: linear algebra's arithmetic complexity

< 1969: $O(n^3)$ for everyone (Gauss, Householder, Danilevskĭi, etc)

Matrix Product

[Strassen 69]: $O(n^{2.807})$

⋮

[Schönhage 81] $O(n^{2.52})$

⋮

[Coppersmith, Winograd 90] $O(n^{2.375})$

⋮

[Le Gall 14] $O(n^{2.3728639})$

$\rightsquigarrow \mathsf{MM}(n) = O(n^\omega)$

Reductions: linear algebra's arithmetic complexity

< 1969: $O(n^3)$ for everyone (Gauss, Householder, Danilevskĭi, etc)

Matrix Product

[Strassen 69]: $O(n^{2.807})$

$\vdots$

[Schönhage 81] $O(n^{2.52})$

$\vdots$

[Coppersmith, Winograd 90] $O(n^{2.375})$

$\vdots$

[Le Gall 14] $O(n^{2.3728639})$

$\rightsquigarrow$ $\text{MM}(n) = O(n^\omega)$

Other operations

[Strassen 69]: Inverse in $O(n^\omega)$

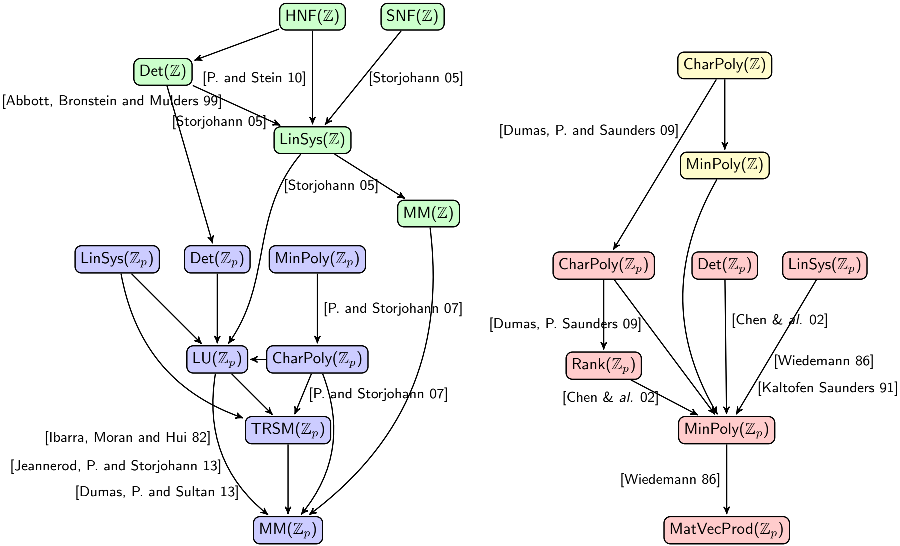
[Schönhage 72]: QR in $O(n^\omega)$

[Bunch, Hopcroft 74]: LU in $O(n^\omega)$

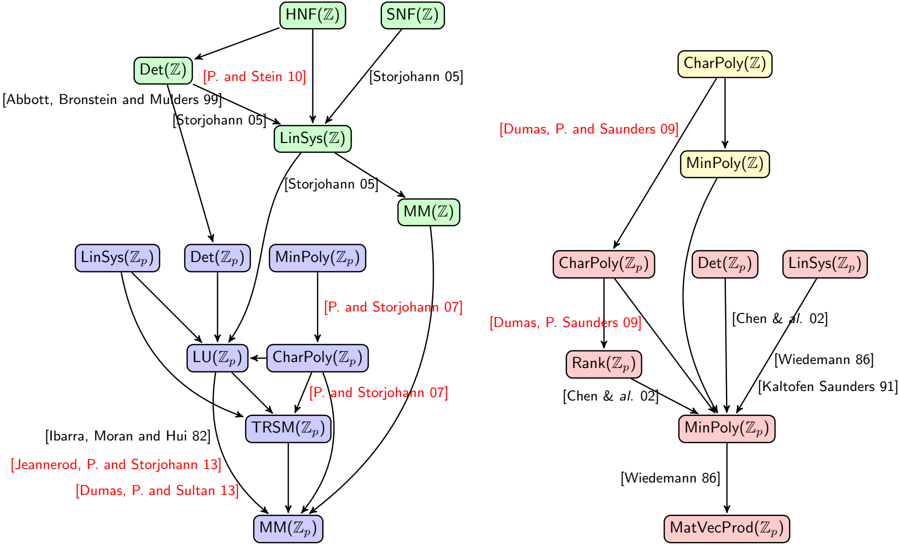
[Ibarra & *al.* 82]: Rank in $O(n^\omega)$

[Keller-Gehrig 85]: CharPoly in $O(n^\omega \log n)$

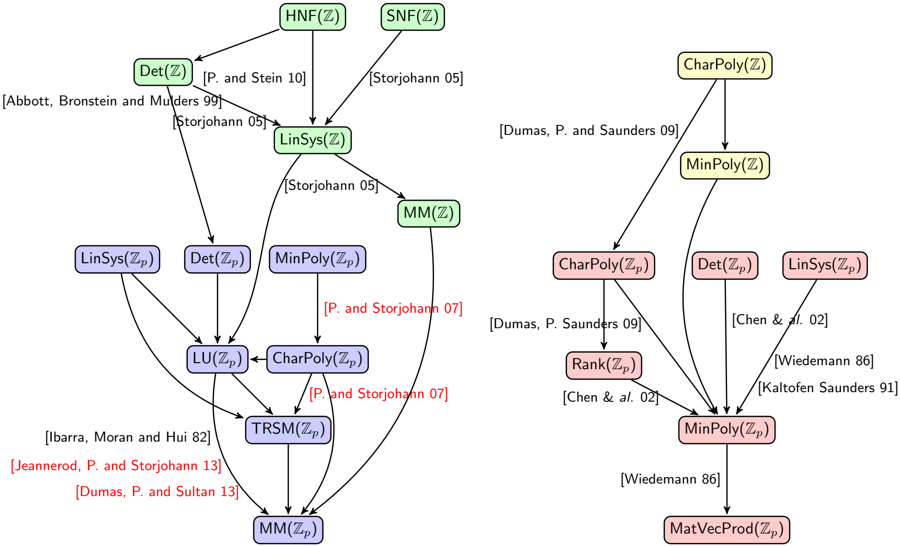
Reductions


HNF(ℤ)
SNF(ℤ)
Det(ℤ)
[P. and Stein 10]
[Storjohann 05]
[Abbott, Bronstein and Mulders 99]
[Storjohann 05]
LinSys(ℤ)
[Storjohann 05]
MM(ℤ)
LinSys($\mathbb{Z}_p$)
Det($\mathbb{Z}_p$)
MinPoly($\mathbb{Z}_p$)
[P. and Storjohann 07]
LU($\mathbb{Z}_p$)
CharPoly($\mathbb{Z}_p$)
[P. and Storjohann 07]
TRSM($\mathbb{Z}_p$)
[Ibarra, Moran and Hui 82]
[Jeannerod, P. and Storjohann 13]
[Dumas, P. and Sultan 13]
MM($\mathbb{Z}_p$)
CharPoly(ℤ)
[Dumas, P. and Saunders 09]
MinPoly(ℤ)
CharPoly($\mathbb{Z}_p$)
Det($\mathbb{Z}_p$)
LinSys($\mathbb{Z}_p$)
[Dumas, P. Saunders 09]
[Chen & al. 02]
Rank($\mathbb{Z}_p$)
[Wiedemann 86]
[Kaltofen Saunders 91]
[Chen & al. 02]
MinPoly($\mathbb{Z}_p$)
[Wiedemann 86]
MatVecProd($\mathbb{Z}_p$)

Reductions


HNF($\mathbb{Z}$)
SNF($\mathbb{Z}$)
Det($\mathbb{Z}$)
[P. and Stein 10]
[Storjohann 05]
[Abbott, Bronstein and Mulders 99]
[Storjohann 05]
LinSys($\mathbb{Z}$)
[Storjohann 05]
MM($\mathbb{Z}$)
LinSys($\mathbb{Z}_p$)
Det($\mathbb{Z}_p$)
MinPoly($\mathbb{Z}_p$)
[P. and Storjohann 07]
LU($\mathbb{Z}_p$)
CharPoly($\mathbb{Z}_p$)
[P. and Storjohann 07]
TRSM($\mathbb{Z}_p$)
[Ibarra, Moran and Hui 82]
[Jeannerod, P. and Storjohann 13]
[Dumas, P. and Sultan 13]
MM($\mathbb{Z}_p$)
CharPoly($\mathbb{Z}$)
[Dumas, P. and Saunders 09]
MinPoly($\mathbb{Z}$)
CharPoly($\mathbb{Z}_p$)
Det($\mathbb{Z}_p$)
LinSys($\mathbb{Z}_p$)
[Chen & al. 02]
[Dumas, P. Saunders 09]
[Wiedemann 86]
Rank($\mathbb{Z}_p$)
[Kaltofen Saunders 91]
[Chen & al. 02]
MinPoly($\mathbb{Z}_p$)
[Wiedemann 86]
MatVecProd($\mathbb{Z}_p$)

Reductions


HNF($\mathbb{Z}$)
SNF($\mathbb{Z}$)
Det($\mathbb{Z}$)
[P. and Stein 10]
[Storjohann 05]
[Abbott, Bronstein and Mulders 99]
[Storjohann 05]
LinSys($\mathbb{Z}$)
[Storjohann 05]
MM($\mathbb{Z}$)
LinSys($\mathbb{Z}_p$)
Det($\mathbb{Z}_p$)
MinPoly($\mathbb{Z}_p$)
[P. and Storjohann 07]
LU($\mathbb{Z}_p$)
CharPoly($\mathbb{Z}_p$)
[P. and Storjohann 07]
TRSM($\mathbb{Z}_p$)
[Ibarra, Moran and Hui 82]
[Jeannerod, P. and Storjohann 13]
[Dumas, P. and Sultan 13]
MM($\mathbb{Z}_p$)
CharPoly($\mathbb{Z}$)
[Dumas, P. and Saunders 09]
MinPoly($\mathbb{Z}$)
CharPoly($\mathbb{Z}_p$)
Det($\mathbb{Z}_p$)
LinSys($\mathbb{Z}_p$)
[Dumas, P. Saunders 09]
[Chen & al. 02]
Rank($\mathbb{Z}_p$)
[Wiedemann 86]
[Kaltofen Saunders 91]
[Chen & al. 02]
MinPoly($\mathbb{Z}_p$)
[Wiedemann 86]
MatVecProd($\mathbb{Z}_p$)

Making theoretical reductions effective

Making theoretical reductions effective

Common mistrust

Fast linear algebra is

- ✗ never faster
- ✗ numerically unstable

Making theoretical reductions effective

Common mistrust

Fast linear algebra is

- ✗ never faster
- ✗ numerically unstable

Lucky coincidence

- ✓ building blocks **in theory** happen to be the most efficient routines **in practice**

$\rightsquigarrow$ reduction trees are still relevant

Making theoretical reductions effective

Common mistrust

Fast linear algebra is

- ✗ never faster
- ✗ numerically unstable

Lucky coincidence

- ✓ building blocks **in theory** happen to be the most efficient routines **in practice**

$\rightsquigarrow$ reduction trees are still relevant

Roadmap

1. Tune building blocks (MatMul)
2. Improve existing reductions (LU, Echelon)
   - ▹ leading constants
   - ▹ memory footprint
3. Produce new reduction schemes (CharPoly, Rank Profiles)

Design of parallel exact linear algebra

ANR HPAC project:

1. efficient kernels for exact linear algebra on SMP
2. DSL, runtime as a plugin and composition
3. attacking large scale challenges from cryptography

Design of parallel exact linear algebra

ANR HPAC project: **Ziad Sultan PhD. Thesis**

1. efficient kernels for exact linear algebra on SMP
2. DSL, runtime as a plugin and composition
3. attacking large scale challenges from cryptography

Design of parallel exact linear algebra

ANR HPAC project:

Ziad Sultan PhD. Thesis

1. **efficient kernels for exact linear algebra on SMP**
2. DSL, runtime as a plugin and composition
3. attacking large scale challenges from cryptography

Design of parallel exact linear algebra

ANR HPAC project: **Ziad Sultan PhD. Thesis**

1. **efficient kernels for exact linear algebra on SMP**
2. DSL, runtime as a plugin and composition
3. attacking large scale challenges from cryptography

Parallel numerical linear algebra

- cost invariant wrt. splitting
  - $O(n^3)$

  $\rightsquigarrow$ fine grain

  $\rightsquigarrow$ block iterative algorithms
- regular task load
- Numerical stability constraints

Design of parallel exact linear algebra

ANR HPAC project: **Ziad Sultan PhD. Thesis**

1. **efficient kernels for exact linear algebra on SMP**
2. DSL, runtime as a plugin and composition
3. attacking large scale challenges from cryptography

Parallel numerical linear algebra

- cost invariant wrt. splitting
  - $O(n^3)$

  $\rightsquigarrow$ fine grain

  $\rightsquigarrow$ block iterative algorithms
- regular task load
- Numerical stability constraints

Exact linear algebra specificities

- cost affected by the splitting
  - $O(n^\omega)$ for $w < 3$
  - modular reductions

  $\rightsquigarrow$ coarse grain

  $\rightsquigarrow$ recursive algorithms
- rank deficiencies

  $\rightsquigarrow$ unbalanced task loads

Design of parallel exact linear algebra

ANR HPAC project: **Ziad Sultan PhD. Thesis**

1. **efficient kernels for exact linear algebra on SMP**
2. DSL, runtime as a plugin and composition
3. attacking large scale challenges from cryptography

Parallel numerical linear algebra

- cost invariant wrt. splitting
  - $O(n^3)$

  $\rightsquigarrow$ fine grain

  $\rightsquigarrow$ block iterative algorithms
- regular task load
- Numerical stability constraints

Exact linear algebra specificities

- cost affected by the splitting
  - $O(n^\omega)$ for $w < 3$
  - modular reductions

  $\rightsquigarrow$ coarse grain

  $\rightsquigarrow$ recursive algorithms
- rank deficiencies

  $\rightsquigarrow$ unbalanced task loads

[Broquedis, Danjean and Gautier 12]: `libkomp` based on `XKaapi`

Matrix Multiplication over $\mathbb{Z}/p\mathbb{Z}$

Ingedients [Dumas, Gautier and P. 02]

- Compute over $\mathbb{Z}$ and delay modular reductions

$$\rightsquigarrow \quad k\left(\frac{p-1}{2}\right)^2 < 2^{\text{mantissa}}$$

Matrix Multiplication over $\mathbb{Z}/p\mathbb{Z}$

Ingedients [Dumas, Gautier and P. 02]

- Compute over $\mathbb{Z}$ and delay modular reductions

  $\rightsquigarrow$ $k\left(\frac{p-1}{2}\right)^2 < 2^{\text{mantissa}}$

- Fastest integer arithmetic: `double`, `float` (SIMD and pipeline)
- Cache optimizations

  $\rightsquigarrow$ numerical BLAS

Matrix Multiplication over $\mathbb{Z}/p\mathbb{Z}$

Ingedients [Dumas, Gautier and P. 02]

- Compute over $\mathbb{Z}$ and delay modular reductions

  $\rightsquigarrow 9^\ell \left\lfloor \frac{k}{2^\ell} \right\rfloor \left(\frac{p-1}{2}\right)^2 < 2^{\text{mantissa}}$

- Fastest integer arithmetic: `double`, `float` (SIMD and pipeline)
- Cache optimizations

  $\rightsquigarrow$ numerical BLAS

- Strassen-Winograd $6n^{2.807} + \ldots$

Matrix Multiplication over $\mathbb{Z}/p\mathbb{Z}$

Ingredients [Dumas, Gautier and P. 02]

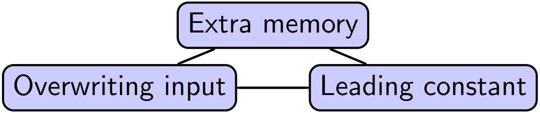
- Compute over $\mathbb{Z}$ and delay modular reductions

  $\rightsquigarrow$ $9^\ell \left\lfloor \frac{k}{2^\ell} \right\rfloor \left(\frac{p-1}{2}\right)^2 < 2^{\text{mantissa}}$
- Fastest integer arithmetic: `double`, `float` (SIMD and pipeline)
- Cache optimizations

  $\rightsquigarrow$ numerical BLAS
- Strassen-Winograd $6n^{2.807} + \dots$

with memory efficient schedules [Boyer, Dumas, P. and Zhou 09]

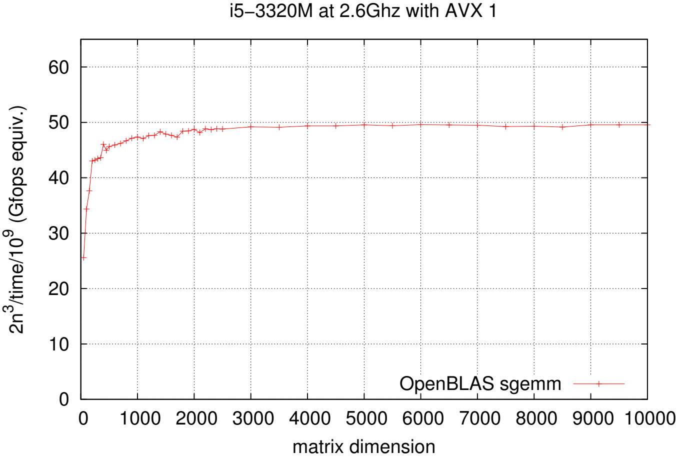
Tradeoffs:



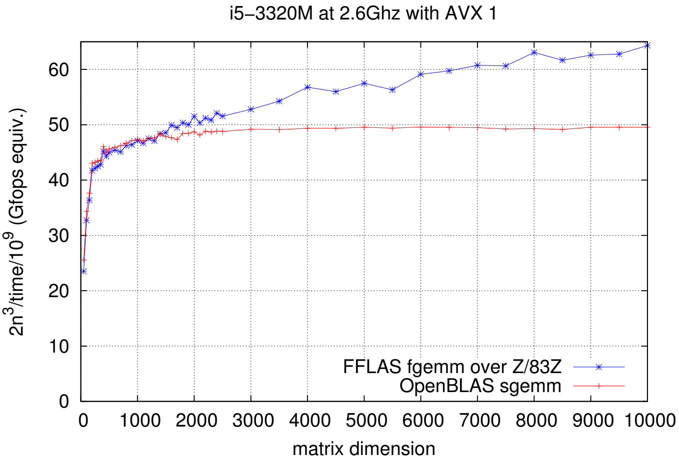
Fully in-place in

$7.2n^{2.807} + \dots$

Sequential Matrix Multiplication

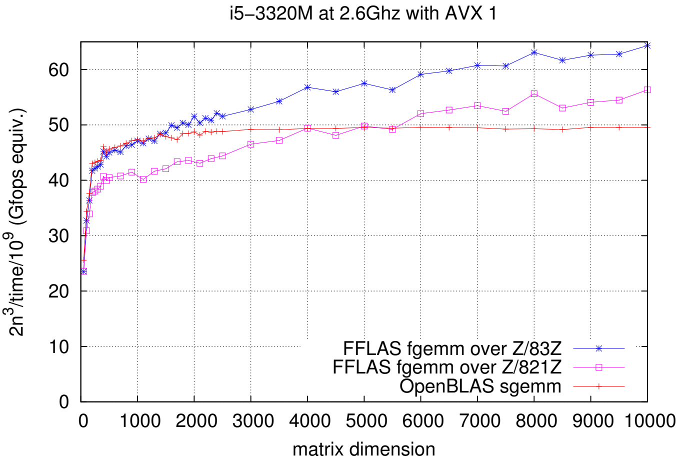


Sequential Matrix Multiplication



$p = 83$, $\rightsquigarrow$ 1 mod / 10000 mul.

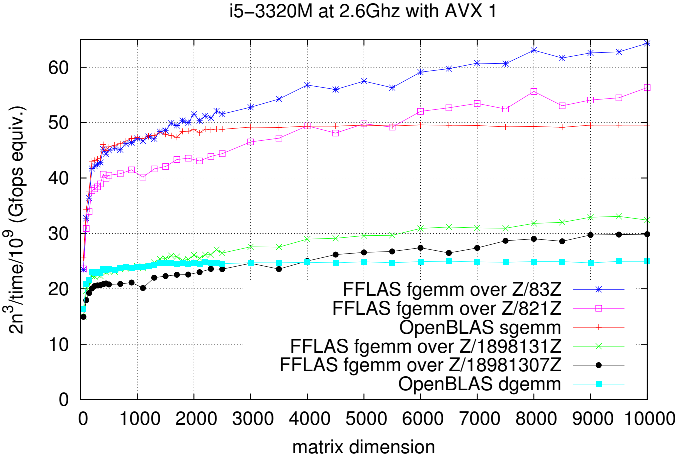
Sequential Matrix Multiplication



$p = 83$, $\rightsquigarrow$ 1 mod / 10000 mul.

$p = 821$, $\rightsquigarrow$ 1 mod / 100 mul.

Sequential Matrix Multiplication



$p = 83$, $\rightsquigarrow$ 1 mod / 10000 mul.

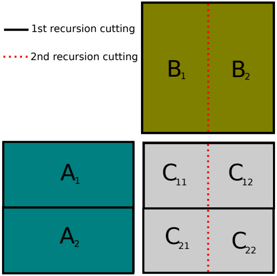
$p = 821$, $\rightsquigarrow$ 1 mod / 100 mul.

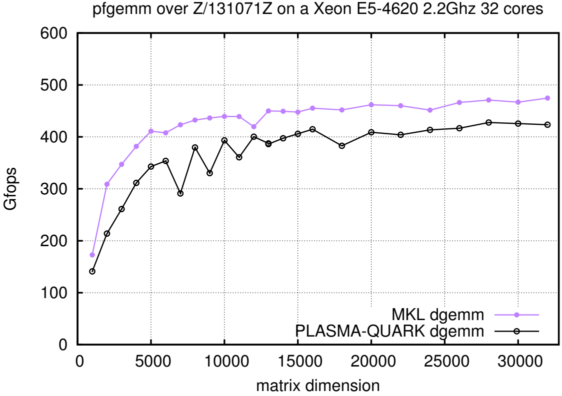
$p = 1898131$, $\rightsquigarrow$ 1 mod / 10000 mul.

$p = 18981307$, $\rightsquigarrow$ 1 mod / 100 mul.

Parallel matrix multiplication

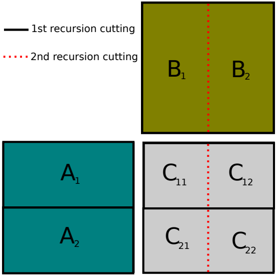
Dumas, Gautier, P. and Sultan 14

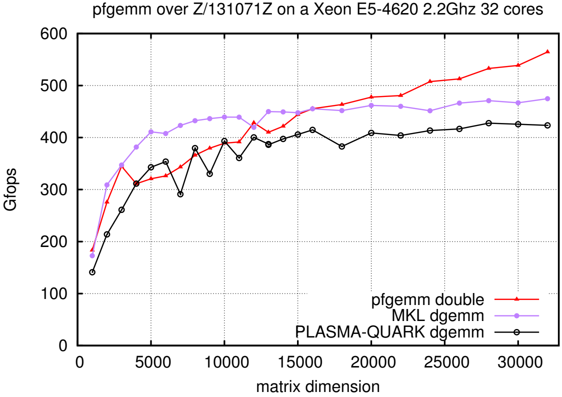




Parallel matrix multiplication

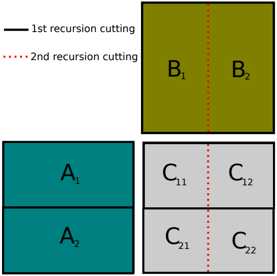
Dumas, Gautier, P. and Sultan 14

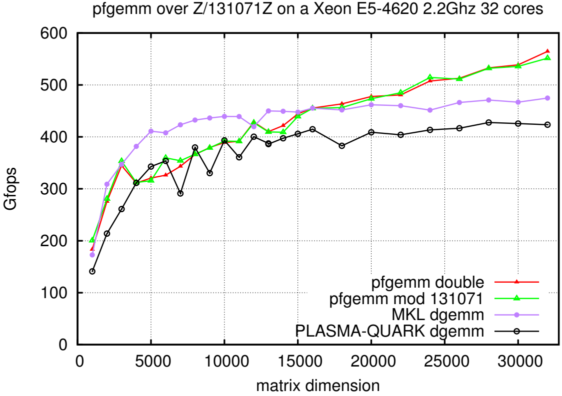




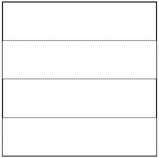
Parallel matrix multiplication

Dumas, Gautier, P. and Sultan 14

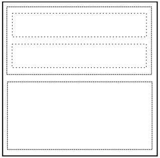




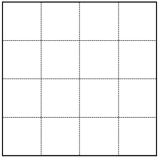
Gaussian elimination



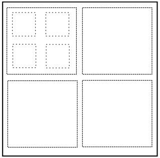
Slab iterative
`LAPACK`



Slab recursive
`FFLAS-FFPACK`

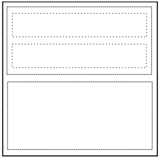


Tile iterative
`PLASMA`

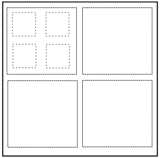


Tile recursive
`FFLAS-FFPACK`

Gaussian elimination



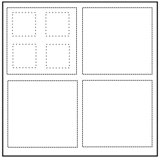
Slab recursive
FFLAS-FFPACK



Tile recursive
FFLAS-FFPACK

- Prefer recursive algorithms

Gaussian elimination



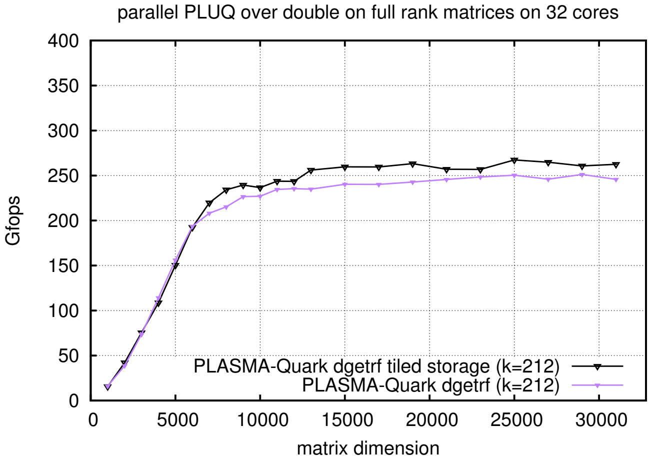
Tile recursive
FFLAS-FFPACK

- Prefer recursive algorithms
- Better data locality

Full rank Gaussian elimination

Dumas, Gautier, P. and Sultan 14

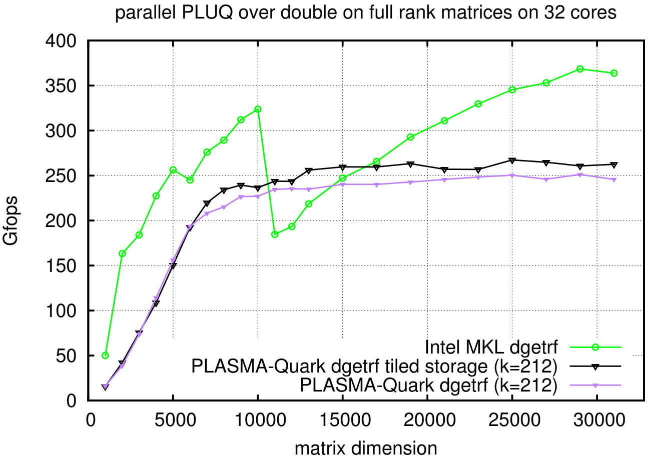
Comparing numerical efficiency (no modulo)



Full rank Gaussian elimination

Dumas, Gautier, P. and Sultan 14

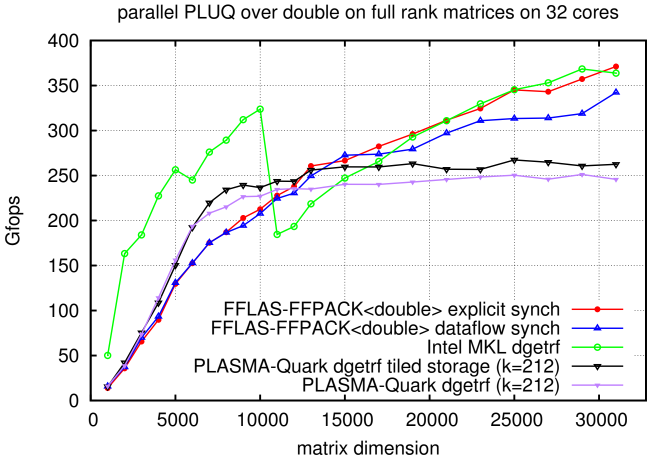
Comparing numerical efficiency (no modulo)



Full rank Gaussian elimination

Dumas, Gautier, P. and Sultan 14

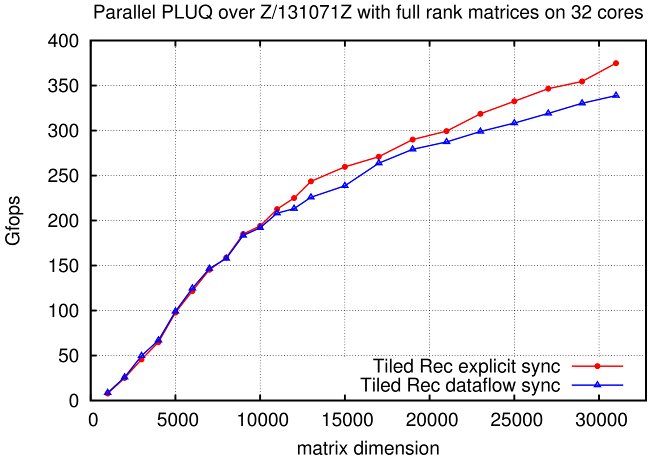
Comparing numerical efficiency (no modulo)



Full rank Gaussian elimination

Dumas, Gautier, P. and Sultan 14

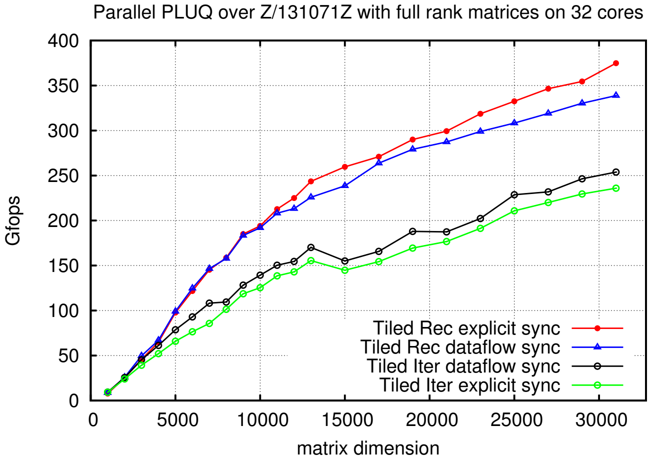
Over the finite field $\mathbb{Z}/131071\mathbb{Z}$



Full rank Gaussian elimination

Dumas, Gautier, P. and Sultan 14

Over the finite field $\mathbb{Z}/131071\mathbb{Z}$



Rank profiles

Definition (Row Rank Profile: RowRP)

Given $A \in \mathsf{K}^{m \times n}, r = \operatorname{rank}(A)$.

informally: *first* r linearly independent rows

formally: lexico-minimal sub-sequence of $(1, \ldots, m)$ of r indices of linearly independant rows.

Example

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

Rank profiles

Definition (Row Rank Profile: RowRP)

Given $A \in \mathsf{K}^{m\times n}, r = \mathrm{rank}(A)$.

informally: *first* r linearly independent rows

formally: lexico-minimal sub-sequence of $(1,\ldots,m)$ of r indices of linearly independant rows.

Example

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

Rank = 3
RowRP = {1,2,4}

Rank profiles

Definition (Row Rank Profile: RowRP)

Given $A \in \mathsf{K}^{m\times n}, r = \text{rank}(A)$.

informally: *first* r linearly independent rows

formally: lexico-minimal sub-sequence of $(1,\ldots,m)$ of r indices of linearly independant rows.

Example

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

Rank = 3
RowRP = {1,2,4}
ColRP = {1,2,3}

Rank profiles

Definition (Row Rank Profile: RowRP)

Given $A \in \mathsf{K}^{m\times n}, r = \mathsf{rank}(A)$.

informally: *first* r linearly independent rows

formally: lexico-minimal sub-sequence of $(1,\ldots,m)$ of r indices of linearly independant rows.

Example

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

Rank = 3

RowRP = {1,2,4}

ColRP = {1,2,3} → Generic ColRP.

Rank profiles

Definition (Row Rank Profile: RowRP)

Given $A \in \mathsf{K}^{m \times n}, r = \mathsf{rank}(A)$.

informally: *first* r linearly independent rows

formally: lexico-minimal sub-sequence of $(1, \ldots, m)$ of r indices of linearly independant rows.

Example

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

Rank = 3
RowRP = {1,2,4}
ColRP = {1,2,3} → Generic ColRP.

- Major invariant of a matrix (echelon form)
- Gröbner basis computations (Macaulay matrix) [Faugère 99, 02]
- Krylov methods

Computing rank profiles

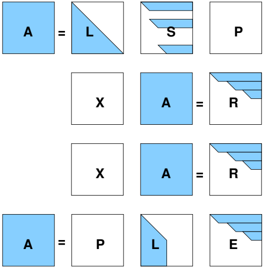
Via Gaussian elimination revealing echelon forms:

[Ibarra, Moran and Hui 82]

[Keller-Gehrig 85]

[Storjohann 00]

[Jeannerod, P. and Storjohann 13]



Computing rank profiles

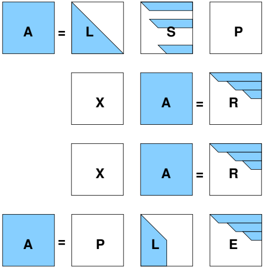
Via Gaussian elimination revealing echelon forms:

[Ibarra, Moran and Hui 82]

[Keller-Gehrig 85]

[Storjohann 00]

[Jeannerod, P. and Storjohann 13]



Lessons learned (or what we thought was necessary):

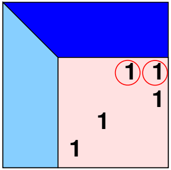
- treat rows in order
- exhaust all columns before considering the next row
- **slab** block splitting required (recursive or iterative)
  $\rightsquigarrow$ similar to partial pivoting

Pivoting strategies revealing rank profiles

Pivot Search

Pivot's (i, j) position minimizes some pre-order:

Row order: any non-zero on the first non-zero row



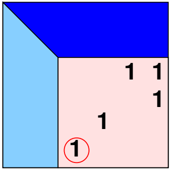
Search	Row perm.	Col. perm.	RowRP	ColRP	Tiles possible
Row order					

Pivoting strategies revealing rank profiles

Pivot Search

Pivot's (i,j) position minimizes some pre-order:

Row/Col order: any non-zero on the first non-zero row/col



Search	Row perm.	Col. perm.	RowRP	ColRP	Tiles possible
Row order Col. order					

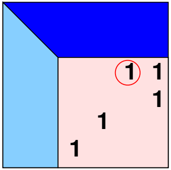
Pivoting strategies revealing rank profiles

Pivot Search

Pivot's (i, j) position minimizes some pre-order:

Row/Col order: any non-zero on the first non-zero row/col

Lex order: first non-zero on the first non-zero row



Search	**Row perm.**	**Col. perm.**	RowRP	ColRP	Tiles possible
Row order Col. order					
Lexico.					

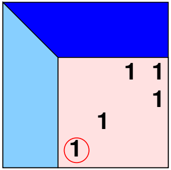
Pivoting strategies revealing rank profiles

Pivot Search

Pivot's (i, j) position minimizes some pre-order:

Row/Col order: any non-zero on the first non-zero row/col

Lex/RevLex order: first non-zero on the first non-zero row/col



Search	**Row perm.**	**Col. perm.**	RowRP	ColRP	Tiles possible
Row order Col. order					
Lexico.					
Rev. lex.					

Pivoting strategies revealing rank profiles

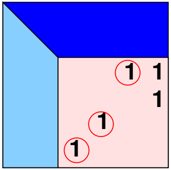
Pivot Search

Pivot's (i, j) position minimizes some pre-order:

Row/Col order: any non-zero on the first non-zero row/col

Lex/RevLex order: first non-zero on the first non-zero row/col

Product order: first non-zero in the (i, j) leading sub-matrix



Search	**Row perm.**	**Col. perm.**	RowRP	ColRP	Tiles possible
Row order Col. order					
Lexico.					
Rev. lex.					
Product					

Pivoting strategies revealing rank profiles

Pivot Search

Pivot's (i,j) position minimizes some pre-order:

Row/Col order: any non-zero on the first non-zero row/col

Lex/RevLex order: first non-zero on the first non-zero row/col

Product order: first non-zero in the (i,j) leading sub-matrix

Permutation

- Transpositions

Search	Row perm.	Col. perm.	RowRP	ColRP	Tiles possible
Row order	Transposition	Transposition	✓		✗
Col. order	Transposition	Transposition		✓	✗
Lexico.	Transposition	Transposition	✓		✗
Rev. lex.	Transposition	Transposition		✓	✗
Product					

Pivoting strategies revealing rank profiles

Pivot Search

Pivot's (i, j) position minimizes some pre-order:

Row/Col order: any non-zero on the first non-zero row/col

Lex/RevLex order: first non-zero on the first non-zero row/col

Product order: first non-zero in the (i, j) leading sub-matrix

Permutation

- Transpositions
- Cylcic Rotations

Search	Row perm.	Col. perm.	RowRP	ColRP	Tiles possible
Row order	Transposition	Transposition	✓		✗
Col. order	Transposition	Transposition		✓	✗
Lexico.	Transposition	Transposition	✓		✗
Rev. lex.	Transposition	Transposition		✓	✗
Product	Rotation	Transposition	✓		✓
Product	Transposition	Rotation		✓	✓

Pivoting strategies revealing rank profiles

Pivot Search

Pivot's (i,j) position minimizes some pre-order:

Row/Col order: any non-zero on the first non-zero row/col

Lex/RevLex order: first non-zero on the first non-zero row/col

Product order: first non-zero in the (i,j) leading sub-matrix

Permutation

- Transpositions
- Cylcic Rotations

Search	**Row perm.**	**Col. perm.**	RowRP	ColRP	All RPs	Tiles possible
Row order	Transposition	Transposition	✓			✗
Col. order	Transposition	Transposition		✓		✗
Lexico.	Transposition	Transposition	✓			✗
Rev. lex.	Transposition	Transposition		✓		✗
Product	Rotation	Transposition	✓			✓
Product	Transposition	Rotation		✓		✓
Product	Rotation	Rotation	✓	✓	✓	✓

Pivoting strategies revealing rank profiles

Pivot Search

Pivot's (i, j) position minimizes some pre-order:

Row/Col order: any non-zero on the first non-zero row/col

Lex/RevLex order: first non-zero on the first non-zero row/col

Product order: first non-zero in the (i, j) leading sub-matrix

Permutation

- Transpositions
- Cylcic Rotations

Search	Row perm.	Col. perm.	RowRP	ColRP	All RPs	Tiles possible
Row order	Transposition	Transposition	✓			✗
Col. order	Transposition	Transposition		✓		✗
Lexico.	Transposition	Transposition	✓			✗
Lexico.	Transposition	Rotation	✓	✓	✓	✗
Rev. lex.	Transposition	Transposition		✓		✗
Rev. lex.	Rotation	Transposition	✓	✓	✓	✗
Product	Rotation	Transposition	✓			✓
Product	Transposition	Rotation		✓		✓
Product	Rotation	Rotation	✓	✓	✓	✓

Pivoting strategies revealing rank profiles

Pivot Search

Pivot's (i, j) position minimizes some pre-order:

Row/Col order: any non-zero on the first non-zero row/col

Lex/RevLex order: first non-zero on the first non-zero row/col

Product order: first non-zero in the (i, j) leading sub-matrix

Permutation

- Transpositions
- Cylcic Rotations

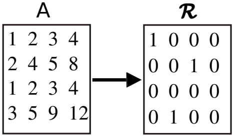
Search	**Row perm.**	**Col. perm.**	RowRP	ColRP	All RPs	Tiles possible
Row order	Transposition	Transposition	✓			✗
Col. order	Transposition	Transposition		✓		✗
Lexico.	Transposition	Transposition	✓			✗
Lexico.	Transposition	Rotation	✓	✓	✓	✗
Rev. lex.	Transposition	Transposition		✓		✗
Rev. lex.	Rotation	Transposition	✓	✓	✓	✗
Product	Rotation	Transposition	✓			✓
Product	Transposition	Rotation		✓		✓
Product	Rotation	Rotation	✓	✓	✓	✓

Computing all rank profiles at once

Dumas, P. and Sultan 13

Definition (Rank Profile matrix)

The unique $\mathcal{R}_A \in \{0,1\}^{m\times n}$ such that any pair of (i,j)-leading sub-matrix of $\mathcal{R}_A$ and of A have the same rank.



Computing all rank profiles at once

Dumas, P. and Sultan 13

Definition (Rank Profile matrix)

The unique $\mathcal{R}_A \in \{0,1\}^{m\times n}$ such that any pair of (i,j)-leading sub-matrix of $\mathcal{R}_A$ and of A have the same rank.

Theorem

- *RowRP and ColRP read directly on $\mathcal{R}(A)$*
- *Same holds for any (i,j)-leading submatrix.*

A

1	2	3	4
2	4	5	8
1	2	3	4
3	5	9	12

→

$\mathcal{R}$

1	0	0	0
0	0	1	0
0	0	0	0
0	1	0	0

RowRP $= \{1\}$
ColRP $= \{1\}$

Computing all rank profiles at once

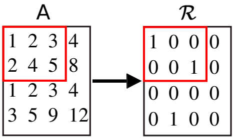
Dumas, P. and Sultan 13

Definition (Rank Profile matrix)

The unique $\mathcal{R}_A \in \{0,1\}^{m\times n}$ such that any pair of (i,j)-leading sub-matrix of $\mathcal{R}_A$ and of A have the same rank.

Theorem

- *RowRP and ColRP read directly on $\mathcal{R}(A)$*
- *Same holds for any (i,j)-leading submatrix.*



RowRP = {1,2}
ColRP = {1,3}

Computing all rank profiles at once

Dumas, P. and Sultan 13

Definition (Rank Profile matrix)

The unique $\mathcal{R}_A \in \{0,1\}^{m\times n}$ such that any pair of (i,j)-leading sub-matrix of $\mathcal{R}_A$ and of A have the same rank.

Theorem

- *RowRP and ColRP read directly on $\mathcal{R}(A)$*
- *Same holds for any (i,j)-leading submatrix.*

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 5 & 8 \\ 1 & 2 & 3 & 4 \\ 3 & 5 & 9 & 12 \end{bmatrix} \longrightarrow \mathcal{R} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

RowRP $= \{1,4\}$
ColRP $= \{1,2\}$

Computing all rank profiles at once

Dumas, P. and Sultan 13

Definition (Rank Profile matrix)

The unique $\mathcal{R}_A \in \{0,1\}^{m \times n}$ such that any pair of (i,j)-leading sub-matrix of $\mathcal{R}_A$ and of A have the same rank.

Theorem

- *RowRP and ColRP read directly on $\mathcal{R}(A)$*
- *Same holds for any (i,j)-leading submatrix.*

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 5 & 8 \\ 1 & 2 & 3 & 4 \\ 3 & 5 & 9 & 12 \end{bmatrix} \longrightarrow \mathcal{R} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

RowRP = {1,4}
ColRP = {1,2}

$$A = PLUQ = P \begin{bmatrix} L & 0 \\ M & I_{m-r} \end{bmatrix} \quad \begin{bmatrix} I_r & \\ & 0 \end{bmatrix} \quad \begin{bmatrix} U & V \\ & I_{n-r} \end{bmatrix} Q$$

Computing all rank profiles at once

Dumas, P. and Sultan 13

Definition (Rank Profile matrix)

The unique $\mathcal{R}_A \in \{0,1\}^{m\times n}$ such that any pair of (i,j)-leading sub-matrix of $\mathcal{R}_A$ and of A have the same rank.

Theorem

- *RowRP and ColRP read directly on $\mathcal{R}(A)$*
- *Same holds for any (i,j)-leading submatrix.*

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 5 & 8 \\ 1 & 2 & 3 & 4 \\ 3 & 5 & 9 & 12 \end{bmatrix} \longrightarrow \mathcal{R} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

RowRP = {1,4}
ColRP = {1,2}

$$A = PLUQ = P\begin{bmatrix} L & 0 \\ M & I_{m-r} \end{bmatrix} P^T P \begin{bmatrix} I_r & \\ & 0 \end{bmatrix} QQ^T \begin{bmatrix} U & V \\ & I_{n-r} \end{bmatrix} Q$$

Computing all rank profiles at once

Dumas, P. and Sultan 13

Definition (Rank Profile matrix)

The unique $\mathcal{R}_A \in \{0,1\}^{m\times n}$ such that any pair of (i,j)-leading sub-matrix of $\mathcal{R}_A$ and of A have the same rank.

Theorem

- *RowRP and ColRP read directly on $\mathcal{R}(A)$*
- *Same holds for any (i,j)-leading submatrix.*

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 5 & 8 \\ 1 & 2 & 3 & 4 \\ 3 & 5 & 9 & 12 \end{bmatrix} \longrightarrow \mathcal{R} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

RowRP = {1,4}
ColRP = {1,2}

$$A = PLUQ = \underbrace{P\begin{bmatrix} L & 0 \\ M & I_{m-r} \end{bmatrix} P^T}_{\overline{L}} \underbrace{P\begin{bmatrix} I_r & \\ & 0 \end{bmatrix} Q}_{\Pi_{P,Q}} \underbrace{Q^T \begin{bmatrix} U & V \\ & I_{n-r} \end{bmatrix} Q}_{\overline{U}}$$

Computing all rank profiles at once

Dumas, P. and Sultan 13

Definition (Rank Profile matrix)

The unique $\mathcal{R}_A \in \{0,1\}^{m\times n}$ such that any pair of (i,j)-leading sub-matrix of $\mathcal{R}_A$ and of A have the same rank.

Theorem

- *RowRP and ColRP read directly on $\mathcal{R}(A)$*
- *Same holds for any (i,j)-leading submatrix.*

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 5 & 8 \\ 1 & 2 & 3 & 4 \\ 3 & 5 & 9 & 12 \end{bmatrix} \longrightarrow \mathcal{R} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

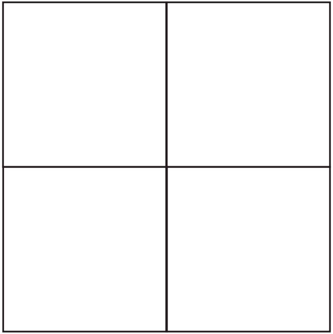
RowRP = {1,4}
ColRP = {1,2}

$$A = PLUQ = \underbrace{P\begin{bmatrix} L & 0 \\ M & I_{m-r} \end{bmatrix} P^T}_{\overline{L}} \underbrace{P\begin{bmatrix} I_r & \\ & 0 \end{bmatrix} Q}_{\Pi_{P,Q}} \underbrace{Q^T \begin{bmatrix} U & V \\ & I_{n-r} \end{bmatrix} Q}_{\overline{U}}$$

With appropriate pivoting: $\Pi_{P,Q} = \mathcal{R}(A)$

A tiled recursive algorithm

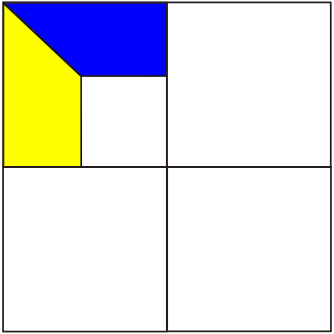
Dumas, P. and Sultan 13



2×2 block splitting

A tiled recursive algorithm

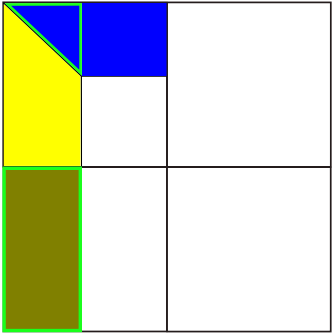
Dumas, P. and Sultan 13



Recursive call

A tiled recursive algorithm

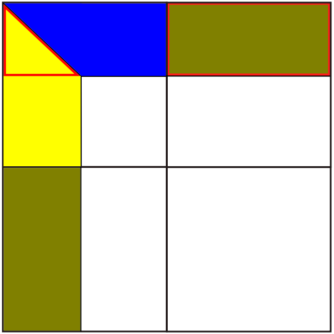
Dumas, P. and Sultan 13



TRSM: $B \leftarrow BU^{-1}$

A tiled recursive algorithm

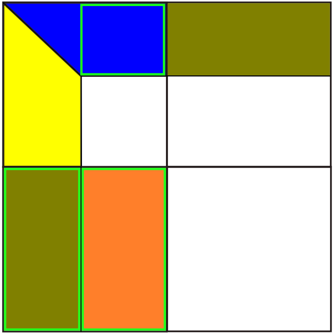
Dumas, P. and Sultan 13



TRSM: $B \leftarrow L^{-1}B$

A tiled recursive algorithm

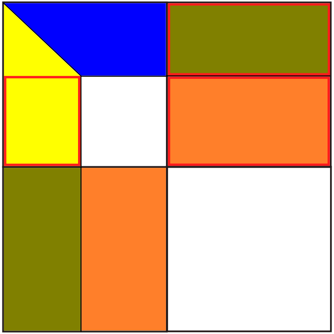
Dumas, P. and Sultan 13



`MatMul`: $C \leftarrow C - A \times B$

A tiled recursive algorithm

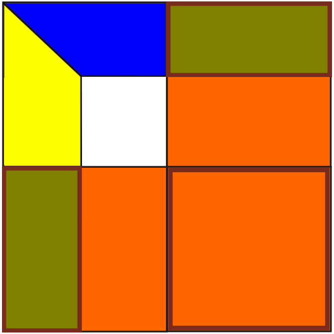
Dumas, P. and Sultan 13



MatMul: $C \leftarrow C - A \times B$

A tiled recursive algorithm

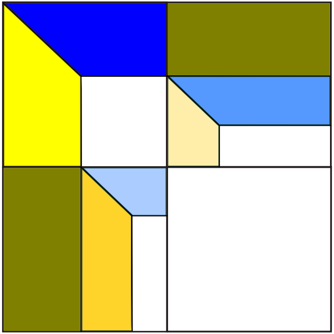
Dumas, P. and Sultan 13



MatMul: $C \leftarrow C - A \times B$

A tiled recursive algorithm

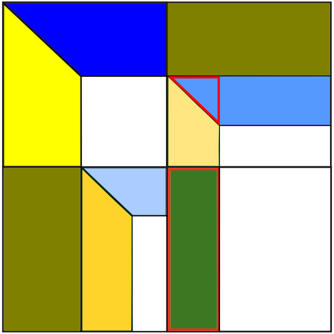
Dumas, P. and Sultan 13



2 independent recursive calls

A tiled recursive algorithm

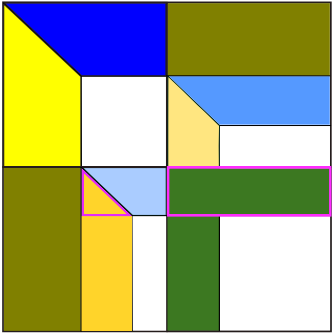
Dumas, P. and Sultan 13



TRSM: $B \leftarrow BU^{-1}$

A tiled recursive algorithm

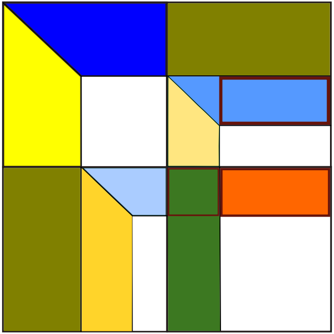
Dumas, P. and Sultan 13



TRSM: $B \leftarrow L^{-1}B$

A tiled recursive algorithm

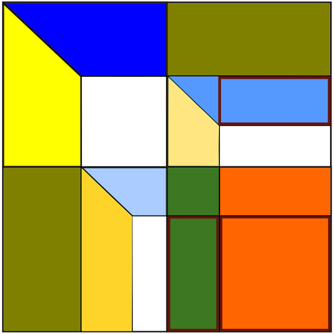
Dumas, P. and Sultan 13



`MatMul`: $C \leftarrow C - A \times B$

A tiled recursive algorithm

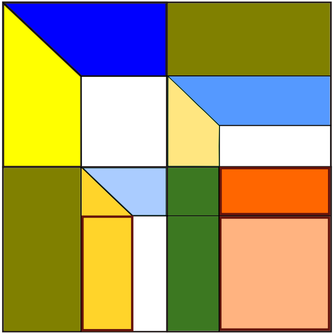
Dumas, P. and Sultan 13



MatMul: $C \leftarrow C - A \times B$

A tiled recursive algorithm

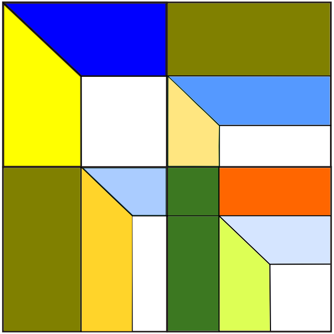
Dumas, P. and Sultan 13



MatMul: $C \leftarrow C - A \times B$

A tiled recursive algorithm

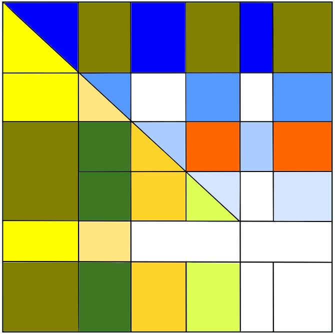
Dumas, P. and Sultan 13



Recursive call

A tiled recursive algorithm

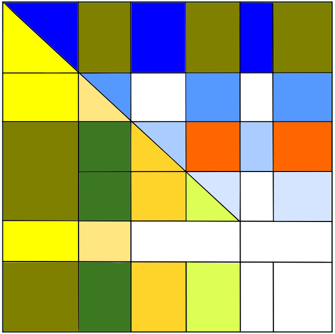
Dumas, P. and Sultan 13



Puzzle game (block cyclic rotations)

A tiled recursive algorithm

Dumas, P. and Sultan 13



- $O(mnr^{\omega-2})$ (degenerating to $2/3n^3$)
- computing col. and row rank profiles of all leading sub-matrices
- fewer modular reductions than slab algorithms
- rank deficiency introduces parallelism

Computing the characteristic polynomial

Motivation

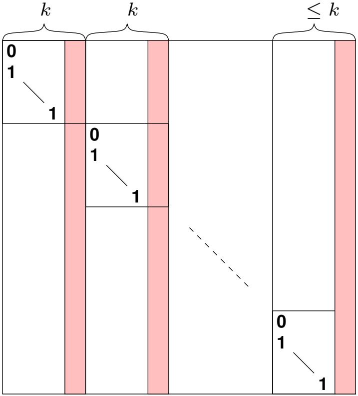
- Connection with the Frobenius normal form
- Krylov methods at large
- Graph invariants
- Crucial step in modular form computations

The last missing reduction

[Danilevskiĭ 37], [Hessenberg 42]	CharPoly, deterministic	$O(n^3)$
[Keller-Gehrig 85]	CharPoly, deterministic	$O(n^\omega \log n)$
[Giesbrecht 93]	Frobenius form, Las-Vegas probabilistic	$O(n^\omega \log n)$
[Augot, Camion 94]	Frobenius form, deterministic	$O(n^3 \#\text{inv factors})$
[Storjohann 00]	Frobenius form, deterministic	$O(n^3)$ or $O(n^\omega \log n \log\log n)$

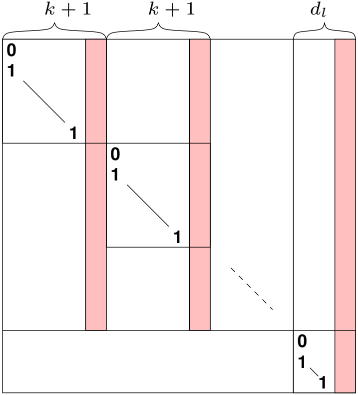
P. and Storjohann ISSAC'07

k-shifted form:



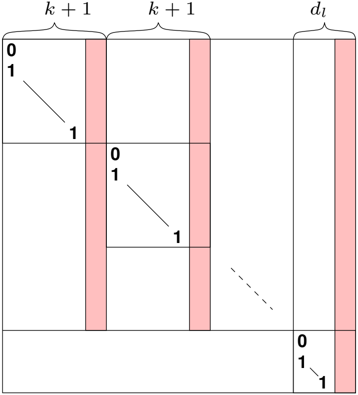
P. and Storjohann ISSAC'07

$k+1$-shifted form:



P. and Storjohann ISSAC'07

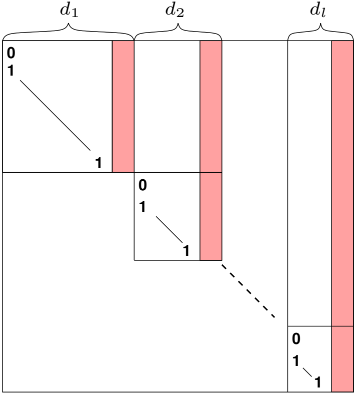
$k+1$-shifted form:



- From k to $k+1$-shifted in $O(n(\frac{n}{k})^{\omega-1})$
- Compute iteratively from a 1-shifted form
- Invariant factors appear by increasing degree

P. and Storjohann ISSAC'07

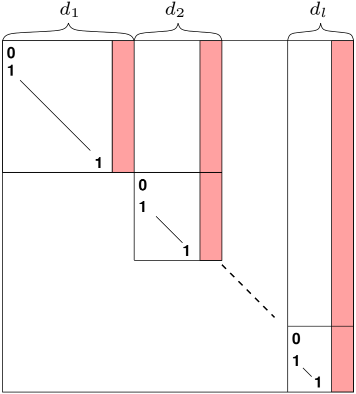
Hessenberg polycyclic:



- From k to $k+1$-shifted in $O(n(\frac{n}{k})^{\omega-1})$
- Compute iteratively from a 1-shifted form
- Invariant factors appear by increasing degree
- Until the Hessenberg polycyclic form

P. and Storjohann ISSAC'07

Hessenberg polycyclic:

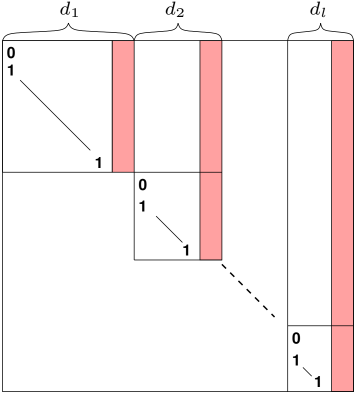


- From k to $k+1$-shifted in $O(n(\frac{n}{k})^{\omega-1})$
- Compute iteratively from a 1-shifted form
- Invariant factors appear by increasing degree
- Until the Hessenberg polycyclic form

$$n^\omega \sum_{k=1}^{n} \left(\frac{1}{k}\right)^{\omega-1} \leq \zeta(\omega-1)n^\omega = O(n^\omega)$$

P. and Storjohann ISSAC'07

Hessenberg polycyclic:



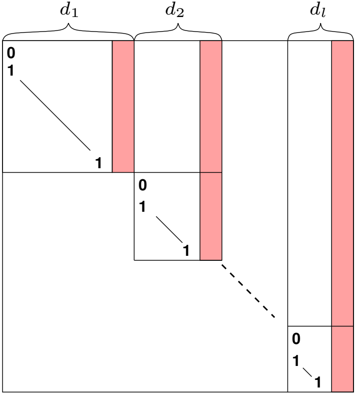
- From k to $k+1$-shifted in $O(n(\frac{n}{k})^{\omega-1})$
- Compute iteratively from a 1-shifted form
- Invariant factors appear by increasing degree
- Until the Hessenberg polycyclic form

$$n^\omega \sum_{k=1}^{n} \left(\frac{1}{k}\right)^{\omega-1} \leq \zeta(\omega-1)n^\omega = O(n^\omega)$$

- Generalized to the Frobenius form as well
- Transformation matrix in $O(n^\omega \log\log n)$

P. and Storjohann ISSAC'07

Hessenberg polycyclic:



- From k to $k+1$-shifted in $O(n(\frac{n}{k})^{\omega-1})$
- Compute iteratively from a 1-shifted form
- Invariant factors appear by increasing degree
- Until the Hessenberg polycyclic form

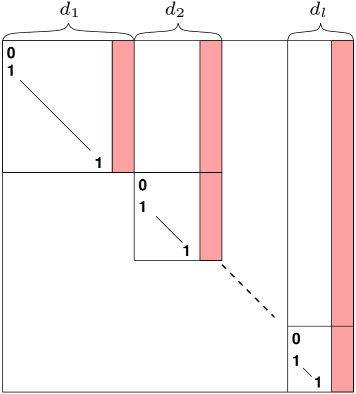
$$n^\omega \sum_{k=1}^{n} \left(\frac{1}{k}\right)^{\omega-1} \leq \zeta(\omega-1)n^\omega = O(n^\omega)$$

- Generalized to the Frobenius form as well
- Transformation matrix in $O(n^\omega \log\log n)$

n	1000	2000	5000	10000
magma-v2.19-9	1.38s	24.28s	332.7s	2497s
fflas-ffpack	0.532s	2.936s	32.71s	219.2s

P. and Storjohann ISSAC'07

Hessenberg polycyclic:



- From k to $k+1$-shifted in $O(n(\frac{n}{k})^{\omega-1})$
- Compute iteratively from a 1-shifted form
- Invariant factors appear by increasing degree
- Until the Hessenberg polycyclic form

$$n^\omega \sum_{k=1}^{n} \left(\frac{1}{k}\right)^{\omega-1} \leq \zeta(\omega-1)n^\omega = O(n^\omega)$$

- Generalized to the Frobenius form as well
- Transformation matrix in $O(n^\omega \log\log n)$

n	1000	2000	5000	10000
`magma-v2.19-9`	1.38s	24.28s	332.7s	2497s
`fflas-ffpack`	0.532s	2.936s	32.71s	219.2s

$\times 7.5$

$\times 6.7$

Outline

Fault Tolerance

Reliability of large scale distributed computing

	Peak	Mean Time To Error	Mean Time To Failure
Blue Waters	14 Pflops	15min	$\approx 1/2$ day
Tsubame 2	2.3 Pflops	?	15.8h

- Disk crash, hardware/software failures $\rightsquigarrow$ hard errors
- Bitflip in main or cache memory $\rightsquigarrow$ soft/silent errors

Fault Tolerance

Reliability of large scale distributed computing

	Peak	Mean Time To Error	Mean Time To Failure
Blue Waters	14 Pflops	15min	$\approx 1/2$ day
Tsubame 2	2.3 Pflops	?	15.8h

- Disk crash, hardware/software failures $\rightsquigarrow$ hard errors
- Bitflip in main or cache memory $\rightsquigarrow$ soft/silent errors

Fault Tolerance

Reliability of large scale distributed computing

	Peak	Mean Time To Error	Mean Time To Failure
Blue Waters	14 Pflops	15min	$\approx 1/2$ day
Tsubame 2	2.3 Pflops	?	15.8h

- Disk crash, hardware/software failures $\rightsquigarrow$ hard errors
- Bitflip in main or cache memory $\rightsquigarrow$ soft/silent errors

Trust in outsourced computations (P2P, Cloud, Volunteer, etc)

Byzantine error model:

- a corrupted node is not always wrong
- black-listing is not an option

Fault Tolerance

Reliability of large scale distributed computing

	Peak	Mean Time To Error	Mean Time To Failure
Blue Waters	14 Pflops	15min	$\approx$ 1/2 day
Tsubame 2	2.3 Pflops	?	15.8h

- Disk crash, hardware/software failures $\rightsquigarrow$ hard errors
- Bitflip in main or cache memory $\rightsquigarrow$ soft/silent errors

Trust in outsourced computations (P2P, Cloud, Volunteer, etc)

Byzantine error model:

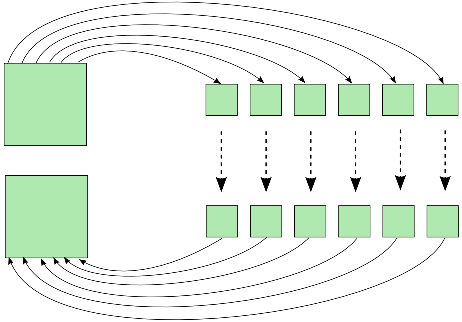
- a corrupted node is not always wrong
- black-listing is not an option

Algorithm Based Fault Tolerance:

exploit the algebraic specificity of the algorithm to embed redundancy.

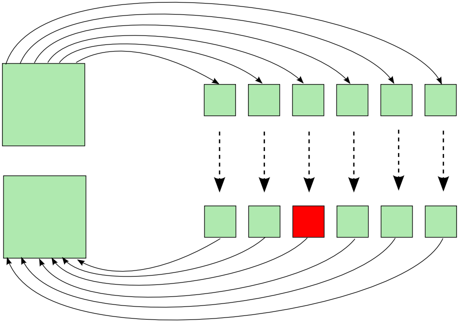
ABFT using error correcting codes

Computations $\Leftrightarrow$ Communication



ABFT using error correcting codes

Unsecure Computations ⇔ Noisy Communication



⇝ Choice of the parallelization algorithm determines

- the communication channel
- the error model

Evaluation-interpolation schemes

Polynomial evaluation

$$\begin{array}{rccc} \mathsf{Ev}_{(x_0,\dots,x_{n-1})}: & \mathsf{K}_{<n}[X] & \longrightarrow & \mathsf{K}^n \\ & f & \longmapsto & (f(x_0),\dots,f(x_{n-1})) \end{array}$$

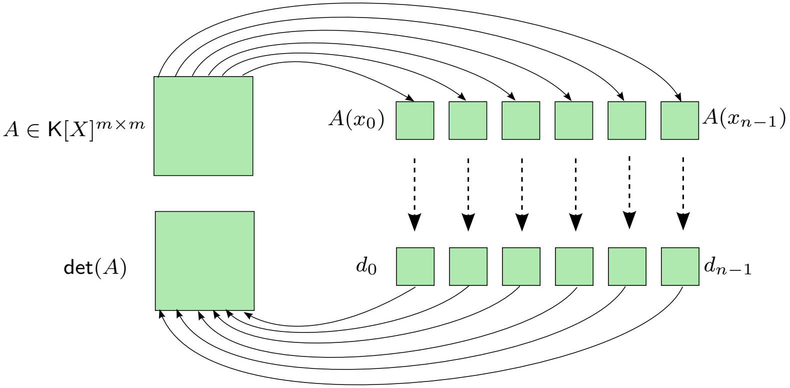
for $x_0,\dots,x_{n-1}$ distinct.

Evaluation-interpolation schemes

Polynomial evaluation

$$\begin{array}{rccc} \mathsf{Ev}_{(x_0,\dots,x_{n-1})}: & \mathsf{K}_{<n}[X] & \longrightarrow & \mathsf{K}^n \\ & f & \longmapsto & (f(x_0),\dots,f(x_{n-1})) \end{array}$$

for $x_0,\dots,x_{n-1}$ distinct.

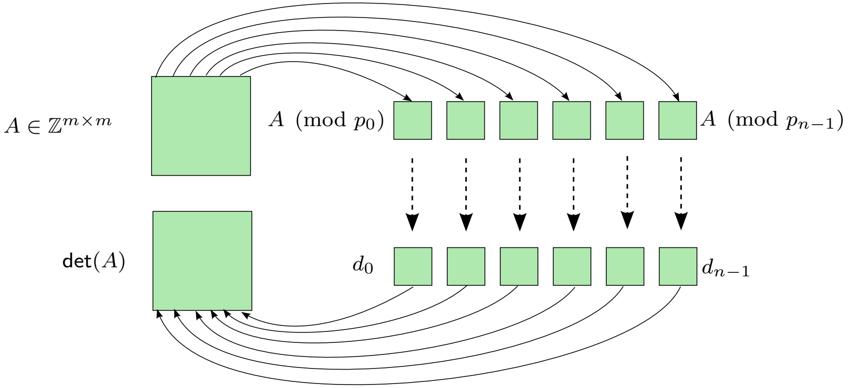


Evaluation-interpolation schemes

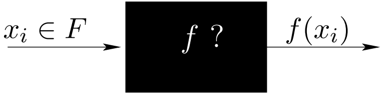
Chinese Remainder Theorem

$$\begin{array}{rccc} \mathrm{Ev}_{(p_0,\dots,p_{n-1})}: & \mathbb{Z}_{<p_0\times\dots\times p_{n-1}} & \longrightarrow & \mathbb{Z}_{p_0}\times\dots\times\mathbb{Z}_{p_{n-1}} \\ & m & \longmapsto & (m \bmod p_0, \dots, m \bmod p_{n-1}) \end{array}$$

for $p_1, \dots, p_n$ pairwise co-prime.



Making evaluation-interpolation schemes fault tolerant



Problem

Recover an unknown function f, given as a black-box, from its evaluations.

Making evaluation-interpolation schemes fault tolerant

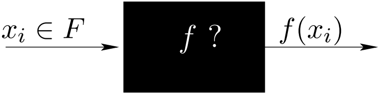
Problem

Recover an unknown function f, given as a black-box, from its evaluations.

Additional knowledge on the model

Dense polynomial: degree bound

Making evaluation-interpolation schemes fault tolerant



$f = \sum_{i=1}^{t} c_i X^{d_i}$

Problem

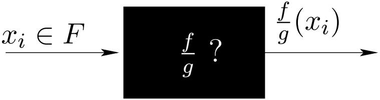
Recover an unknown function f, given as a black-box, from its evaluations.

Additional knowledge on the model

Dense polynomial: degree bound

Sparse polynomial: support unknown, bound on sparsity

Making evaluation-interpolation schemes fault tolerant



$$f = \sum_{i=0}^{d_f} f_i X^i, \; g = \sum_{i=0}^{d_g} g_i X^i$$

Problem

Recover an unknown function f, given as a black-box, from its evaluations.

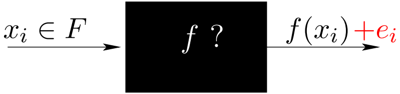
Additional knowledge on the model

Dense polynomial: degree bound

Sparse polynomial: support unknown, bound on sparsity

Dense rational function: degree bounds

Making evaluation-interpolation schemes fault tolerant



Problem

Recover an unknown function f, given as a black-box, from its evaluations.

Additional knowledge on the model

Dense polynomial: degree bound

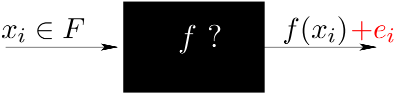
Sparse polynomial: support unknown, bound on sparsity

Dense rational function: degree bounds

Trust in the evaluations

- errors (outliers)
- approximations: numerical noise

Making evaluation-interpolation schemes fault tolerant



Problem

Recover an unknown function f, given as a black-box, from its evaluations.

Additional knowledge on the model

Dense polynomial: degree bound

Sparse polynomial: support unknown, bound on sparsity

Dense rational function: degree bounds

Trust in the evaluations

- errors (outliers)
- approximations: numerical noise

Rational function reconstruction

Problem (RFR: Rational Function Reconstruction)

Given $A, B \in \mathsf{K}[X]$ *with* $\deg B < \deg A = n$,
Find $f \in \mathsf{K}_{\leq d_f}[X]$, $g \in \mathsf{K}_{\leq n-d_f-1}[X]$ *such that*

$$f = gB \mod A.$$

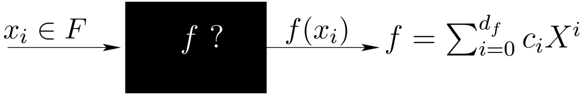
Fact

The Extended Euclidean Algo. run on (A, B) *and terminated when* $\deg f_j \leq d_f < \deg f_{j-1}$, *produces* $f_j = u_j A + v_j B$ *s.t.*

1. (f_j, v_j) *is a solution to the RFR problem.*
2. *it is minimal: any other solution* (f, g) *is of the form*

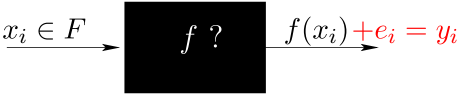
$$f = qf_j, \quad g = qv_j \quad \text{for } q \in \mathsf{K}[X].$$

Dense polynomial interpolation



without error: polynomial interpolation (Lagrange, Newton, etc).

Dense polynomial interpolation with errors



$$f = \sum_{i=0}^{d_f} c_i X^i$$

without error: polynomial interpolation (Lagrange, Newton, etc).

with errors: Reed-Solomon decoding

- Erroneous interpolant: $h = \mathsf{Interp}((y_i, x_i))$
- Error locator polynomial: $\Lambda = \prod_{e_i \neq 0} (X - x_i)$

$$\Lambda f = \Lambda h \mod \prod_{i=0}^{n-1} (X - x_i)$$

Dense polynomial interpolation with errors

$x_i \in F$ → **f ?** → $f(x_i) + e_i = y_i$ $f = \sum_{i=0}^{d_f} c_i X^i$

without error: polynomial interpolation (Lagrange, Newton, etc).

with errors: Reed-Solomon decoding

- Erroneous interpolant: $h = \text{Interp}((y_i, x_i))$
- Error locator polynomial: $\Lambda = \prod_{e_i \neq 0}(X - x_i)$

$$\underbrace{\Lambda f}_{N} = \underbrace{\Lambda}_{D} h \mod \prod_{i=0}^{n-1}(X - x_i)$$

Rational Reconstruction Problem:

$(\Lambda f, \Lambda)$ is a minimal solution ⇝ computed by Ext. Euclidean Algorithm

$$f = f_j / v_j.$$

Correction capacity

Unique decoding of t errors whenever: $n \geq \deg f + 2E + 1$

Bounding the degree

- $\deg f$ rarely known *a priori*; bound $d_f \geq \deg f$ often pessimistic
- Early termination:

  without errors: add evaluations until interpolant stabilizes

  with errors: no stabilization

Correction capacity

Unique decoding of t errors whenever: $n \geq \deg f + 2E + 1$

Bounding the degree

- $\deg f$ rarely known *a priori*; bound $d_f \geq \deg f$ often pessimistic
- Early termination:

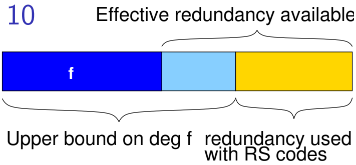
  without errors: add evaluations until interpolant stabilizes

  with errors: no stabilization

Parameter oblivious decoding

Khonji, P., Roch, Roche and Stalinsky 10

$\rightsquigarrow$ how to use all available redundancy?



Correction capacity

Unique decoding of t errors whenever: $n \geq \deg f + 2E + 1$

Bounding the degree

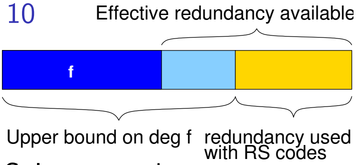
- $\deg f$ rarely known *a priori*; bound $d_f \geq \deg f$ often pessimistic
- Early termination:

  without errors: add evaluations until interpolant stabilizes

  with errors: no stabilization

Parameter oblivious decoding

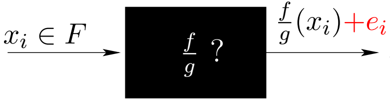
Khonji, P., Roch, Roche and Stalinsky 10



⇝ how to use all available redundancy?

⇝ list decoder exploring all length n Reed-Solomon codes

Dense rational function interpolation with errors



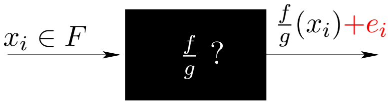
$f = \sum_{i=0}^{d_f} f_i X^i, \ g = \sum_{i=0}^{d_g} g_i X^i$

$$\underbrace{\Lambda f}_{N} = \underbrace{\bar{\Lambda}\bar{g}}_{D}\, h \mod \prod_{y_i \neq \infty} (X - x_i)$$

Rational Reconstruction Problem

$(\Lambda f, \bar{\Lambda}\bar{g})$ is a minimal solution $\rightsquigarrow$ computed by Ext. Euclidean Algorithm.

Dense rational function interpolation with errors



$f = \sum_{i=0}^{d_f} f_i X^i, \; g = \sum_{i=0}^{d_g} g_i X^i$

$$\underbrace{\Lambda f}_{N} = \underbrace{\bar{\Lambda}\bar{g}}_{D} \, h \mod \prod_{y_i \neq \infty} (X - x_i)$$

Rational Reconstruction Problem

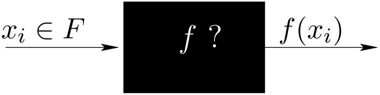
$(\Lambda f, \bar{\Lambda}\bar{g})$ is a minimal solution $\rightsquigarrow$ computed by Ext. Euclidean Algorithm.

Correction capacity

Unique decoding of E errors whenever $n \geq d_f + d_g + 2E + 1$

- smoothly supports evaluations at poles (even erroneous ones)
- parameter oblivious decoding applies

Sparse interpolation

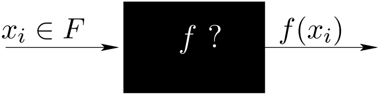


$f = \sum_{i=1}^{t} c_i X^{d_i}$

Without error: [Prony 1795] [Ben-Or and Tiwari 88]

- sample in a geometric progression: $y_i = f(\alpha^i)$

Sparse interpolation

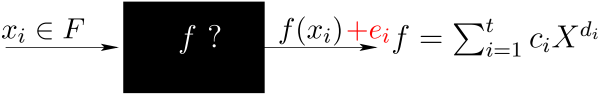


$$f = \sum_{i=1}^{t} c_i X^{d_i}$$

Without error: [Prony 1795] [Ben-Or and Tiwari 88]

- sample in a geometric progression: $y_i = f(\alpha^i)$
- [Blahut'84]: the seq. $(y_0, y_1, ...)$ has linear complexity t
- and is generated by $\Lambda(X) = \prod_{i=1}^{t}(X - \alpha^{d_i})$
- Berlekamp-Massey algo. on $2t$ terms $\rightsquigarrow d_i$
- Vandermonde system $\rightsquigarrow c_i$

Sparse interpolation with errors



Without error: [Prony 1795] [Ben-Or and Tiwari 88]

- sample in a geometric progression: $y_i = f(\alpha^i)$
- [Blahut'84]: the seq. $(y_0, y_1, \ldots)$ has linear complexity t
- and is generated by $\Lambda(X) = \prod_{i=1}^{t}(X - \alpha^{d_i})$
- Berlekamp-Massey algo. on $2t$ terms $\rightsquigarrow d_i$
- Vandermonde system $\rightsquigarrow c_i$

With errors: rule of thumb:

- find a clean sub-sequence of $2t$ terms free of error

Unique decoding by majority rule Berlekamp-Massey

Comer, Kaltofen and P. 12

Necessary condition for unique decoding: $n \geq 2t(2E+1)$

(a_i)	$\Lambda(z)$	$f(z)$
$x = (\overbrace{\overline{0}}^{(t-1) zeros}, 1, \overline{0}, 1, \ldots, \overline{0}, 1)$	$z^t - 1$	$\frac{1}{t}\sum_{i=0}^{t-1} z^{2i\frac{m}{2t}}$

Unique decoding by majority rule Berlekamp-Massey

Comer, Kaltofen and P. 12

Necessary condition for unique decoding: $n \geq 2t(2E+1)$

(a_i)	$\Lambda(z)$	$f(z)$
$x = (\overbrace{\bar{0}}^{(t-1)zeros}, 1, \bar{0}, 1, \ldots, \bar{0}, 1)$	$z^t - 1$	$\frac{1}{t}\sum_{i=0}^{t-1} z^{2i\frac{m}{2t}}$
$y = (\underbrace{\bar{0}}_{(t-1)zeros}, 1, \bar{0}, -1, \ldots, \bar{0}, -1)$	$z^t + 1$	$\frac{-1}{t}\sum_{i=0}^{t-1} z^{(2i+1)\frac{m}{2t}}$

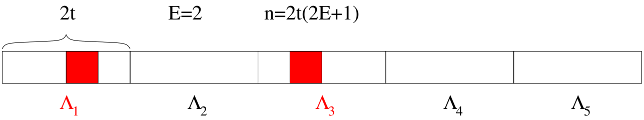
Unique decoding by majority rule Berlekamp-Massey

Comer, Kaltofen and P. 12

Necessary condition for unique decoding: $n \geq 2t(2E+1)$

(a_i)	$\Lambda(z)$	$f(z)$
$x = (\overbrace{\overline{0}}^{(t-1)zeros}, 1, \overline{0}, 1, \dots, \overline{0}, 1)$	$z^t - 1$	$\frac{1}{t}\sum_{i=0}^{t-1} z^{2i\frac{m}{2t}}$
$y = (\underbrace{\overline{0}}_{(t-1)zeros}, 1, \overline{0}, -1, \dots, \overline{0}, -1)$	$z^t + 1$	$\frac{-1}{t}\sum_{i=0}^{t-1} z^{(2i+1)\frac{m}{2t}}$

Sufficient condition for unique decoding: $n \leq 2t(2E+1)$



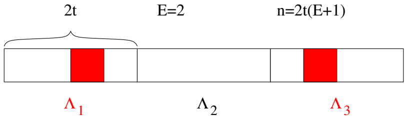
Unique decoding by majority rule Berlekamp-Massey

Comer, Kaltofen and P. 12

Necessary condition for unique decoding: $n \geq 2t(2E+1)$

$$\begin{array}{l|l|l} (a_i) & \Lambda(z) & f(z) \\ x = (\overbrace{\bar{0}}^{(t-1)zeros}, 1, \bar{0}, 1, \dots, \bar{0}, 1) & z^t - 1 & \frac{1}{t}\sum_{i=0}^{t-1} z^{2i\frac{m}{2t}} \\ y = (\underbrace{\bar{0}}_{(t-1)zeros}, 1, \bar{0}, -1, \dots, \bar{0}, -1) & z^t + 1 & \frac{-1}{t}\sum_{i=0}^{t-1} z^{(2i+1)\frac{m}{2t}} \end{array}$$

Sufficient condition for list decoding: $n \leq 2t(E+1)$



List decoding: using affine sub-sequences

Kaltofen and P. 14

$f(\alpha^0)$	$f(\alpha^1)$	$f(\alpha^2)$	$f(\alpha^3)$	$f(\alpha^4)$	$f(\alpha^5)$	$f(\alpha^6)$	$f(\alpha^7)$	$f(\alpha^8)$
$f(\alpha^0)$		$f(\alpha^2)$		$f(\alpha^4)$		$f(\alpha^6)$		$f(\alpha^8)$
	$f(\alpha^1)$		$f(\alpha^3)$		$f(\alpha^5)$		$f(\alpha^7)$	
$f(\alpha^0)$			$f(\alpha^3)$			$f(\alpha^6)$		
	$f(\alpha^1)$			$f(\alpha^4)$			$f(\alpha^7)$	
		$f(\alpha^2)$			$f(\alpha^5)$			$f(\alpha^8)$

List decoding: using affine sub-sequences

Kaltofen and P. 14

$f(\alpha^0)$	$f(\alpha^1)$	$f(\alpha^2)$	$f(\alpha^3)$	$f(\alpha^4)$	$f(\alpha^5)$	$f(\alpha^6)$	$f(\alpha^7)$	$f(\alpha^8)$	$= \mathsf{Ev}(f, \alpha)$
$f(\alpha^0)$		$f(\alpha^2)$		$f(\alpha^4)$		$f(\alpha^6)$		$f(\alpha^8)$	$= \mathsf{Ev}(f, \alpha^2)$
	$f(\alpha^1)$		$f(\alpha^3)$		$f(\alpha^5)$		$f(\alpha^7)$		$= \mathsf{Ev}(f \circ (\alpha x), \alpha^2)$
$f(\alpha^0)$			$f(\alpha^3)$			$f(\alpha^6)$			$= \mathsf{Ev}(f, \alpha^3)$
	$f(\alpha^1)$			$f(\alpha^4)$			$f(\alpha^7)$		$= \mathsf{Ev}(f \circ (\alpha x), \alpha^3)$
		$f(\alpha^2)$			$f(\alpha^5)$			$f(\alpha^8)$	$= \mathsf{Ev}(f \circ (\alpha^2 x), \alpha^3)$

List decoding: using affine sub-sequences

Kaltofen and P. 14

$f(\alpha^0)$	$f(\alpha^1)$	$f(\alpha^2)$	$f(\alpha^3)$	$f(\alpha^4)$	$f(\alpha^5)$	$f(\alpha^6)$	$f(\alpha^7)$	$f(\alpha^8)$	$= \mathsf{Ev}(f, \alpha)$
$f(\alpha^0)$		$f(\alpha^2)$		$f(\alpha^4)$		$f(\alpha^6)$		$f(\alpha^8)$	$= \mathsf{Ev}(f, \alpha^2)$
	$f(\alpha^1)$		$f(\alpha^3)$		$f(\alpha^5)$		$f(\alpha^7)$		$= \mathsf{Ev}(f \circ (\alpha x), \alpha^2)$
$f(\alpha^0)$			$f(\alpha^3)$			$f(\alpha^6)$			$= \mathsf{Ev}(f, \alpha^3)$
	$f(\alpha^1)$			$f(\alpha^4)$			$f(\alpha^7)$		$= \mathsf{Ev}(f \circ (\alpha x), \alpha^3)$
		$f(\alpha^2)$			$f(\alpha^5)$			$f(\alpha^8)$	$= \mathsf{Ev}(f \circ (\alpha^2 x), \alpha^3)$

List decoding: using affine sub-sequences

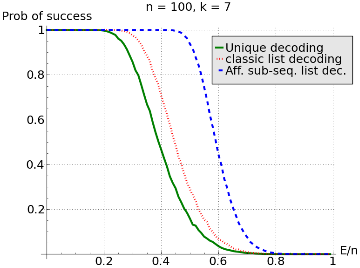
Kaltofen and P. 14

$f(\alpha^0)$	$f(\alpha^1)$	$f(\alpha^2)$	$f(\alpha^3)$	$f(\alpha^4)$	$f(\alpha^5)$	$f(\alpha^6)$	$f(\alpha^7)$	$f(\alpha^8)$	$= \mathsf{Ev}(f, \alpha)$
$f(\alpha^0)$		$f(\alpha^2)$		$f(\alpha^4)$		$f(\alpha^6)$		$f(\alpha^8)$	$= \mathsf{Ev}(f, \alpha^2)$
	$f(\alpha^1)$		$f(\alpha^3)$		$f(\alpha^5)$		$f(\alpha^7)$		$= \mathsf{Ev}(f \circ (\alpha x), \alpha^2)$
$f(\alpha^0)$			$f(\alpha^3)$			$f(\alpha^6)$			$= \mathsf{Ev}(f, \alpha^3)$
	$f(\alpha^1)$			$f(\alpha^4)$			$f(\alpha^7)$		$= \mathsf{Ev}(f \circ (\alpha x), \alpha^3)$
		$f(\alpha^2)$			$f(\alpha^5)$			$f(\alpha^8)$	$= \mathsf{Ev}(f \circ (\alpha^2 x), \alpha^3)$

List decoding: using affine sub-sequences

Kaltofen and P. 14

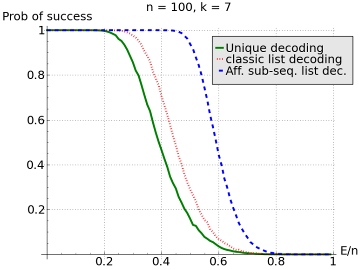
$f(\alpha^0)$	$f(\alpha^1)$	$f(\alpha^2)$	$f(\alpha^3)$	$f(\alpha^4)$	$f(\alpha^5)$	$f(\alpha^6)$	$f(\alpha^7)$	$f(\alpha^8)$	$= \mathrm{Ev}(f, \alpha)$
$f(\alpha^0)$		$f(\alpha^2)$		$f(\alpha^4)$		$f(\alpha^6)$		$f(\alpha^8)$	$= \mathrm{Ev}(f, \alpha^2)$
	$f(\alpha^1)$		$f(\alpha^3)$		$f(\alpha^5)$		$f(\alpha^7)$		$= \mathrm{Ev}(f \circ (\alpha x), \alpha^2)$
$f(\alpha^0)$			$f(\alpha^3)$			$f(\alpha^6)$			$= \mathrm{Ev}(f, \alpha^3)$
	$f(\alpha^1)$			$f(\alpha^4)$			$f(\alpha^7)$		$= \mathrm{Ev}(f \circ (\alpha x), \alpha^3)$
		$f(\alpha^2)$			$f(\alpha^5)$			$f(\alpha^8)$	$= \mathrm{Ev}(f \circ (\alpha^2 x), \alpha^3)$



List decoding: using affine sub-sequences

Kaltofen and P. 14

$f(\alpha^0)$	$f(\alpha^1)$	$f(\alpha^2)$	$f(\alpha^3)$	$f(\alpha^4)$	$f(\alpha^5)$	$f(\alpha^6)$	$f(\alpha^7)$	$f(\alpha^8)$	$= \text{Ev}(f, \alpha)$
$f(\alpha^0)$		$f(\alpha^2)$		$f(\alpha^4)$		$f(\alpha^6)$		$f(\alpha^8)$	$= \text{Ev}(f, \alpha^2)$
	$f(\alpha^1)$		$f(\alpha^3)$		$f(\alpha^5)$		$f(\alpha^7)$		$= \text{Ev}(f \circ (\alpha x), \alpha^2)$
$f(\alpha^0)$			$f(\alpha^3)$			$f(\alpha^6)$			$= \text{Ev}(f, \alpha^3)$
	$f(\alpha^1)$			$f(\alpha^4)$			$f(\alpha^7)$		$= \text{Ev}(f \circ (\alpha x), \alpha^3)$
		$f(\alpha^2)$			$f(\alpha^5)$			$f(\alpha^8)$	$= \text{Ev}(f \circ (\alpha^2 x), \alpha^3)$



Difficult worst case analysis

- [Erdös and Turan 36]: size of the largest subset of $\{1 \ldots n\}$ not containing k terms in arithmetic progression
- [Szeremedi 75]: arithmetic prog. are dense

$$n - \frac{n}{(\log\log n)^{1/2^{2^{k+9}}}} \leq E \leq \frac{n}{k-2} \log_k \frac{n}{k-2}.$$

Towards better decoding capacities

Unique decoding: $n \geq 2t(2E+1)$

List decoding (basic): $n \geq 2t(E+1)$

List decoding (affine subsequence): $n \geq 2t\frac{E}{\log E}$

Towards better decoding capacities

Unique decoding: $n \geq 2t(2E+1)$

List decoding (basic): $n \geq 2t(E+1)$

List decoding (affine subsequence): $n \geq 2t\frac{E}{\log E}$

Improved conditions for unique decoding

Descartes' rule of signs: Over $\mathsf{K} = \mathbb{R}_{>0}$: $n \geq 2t + 2E + 1$

Irreducibility of cyclotomic polynomials:

- Over $\mathsf{K} = \mathbb{C}$: $n \geq 2t\frac{\log \deg f}{\log 2t} + 2E + 1$
- Over $\mathbb{F}_q^{(p_1)} \times \cdots \times \mathbb{F}_q^{(p_n)}$: $n \geq 2t\frac{\log \deg f}{\log 2t} + 2E + 1$

- No known decoding algorithm
- Makes the list decoding algo. a unique decoding one

Conclusion

Design framework for high performance exact linear algebra

Asymptotic reduction > algorithm tuning > building block implementation

- Favor **tiled recursive** algorithms
  $\rightsquigarrow$ **architecture oblivious vs aware** algorithms [Gustavson 07]

Conclusion

Design framework for high performance exact linear algebra

Asymptotic reduction > algorithm tuning > building block implementation

- Favor **tiled recursive** algorithms
  $\rightsquigarrow$ **architecture oblivious vs aware** algorithms [Gustavson 07]
- New pivoting strategies revealing **all rank profile informations**
  $\rightsquigarrow$ **tournament pivoting**? [Demmel, Grigori and Xiang 11]
  $\rightsquigarrow$ $O(r^\omega + (m+n+|A|)^{1+o(n)})$? [Storjohann and Yang 14]

Conclusion

Design framework for high performance exact linear algebra

Asymptotic reduction > algorithm tuning > building block implementation

- Favor **tiled recursive** algorithms
  $\rightsquigarrow$ **architecture oblivious vs aware** algorithms [Gustavson 07]
- New pivoting strategies revealing **all rank profile informations**
  $\rightsquigarrow$ **tournament pivoting**? [Demmel, Grigori and Xiang 11]
  $\rightsquigarrow$ $O(r^\omega + (m+n+|A|)^{1+o(n)})$? [Storjohann and Yang 14]
- **Recursive tasks** and **coarse grain** parallelization
  $\rightsquigarrow$ Light weight task workstealing management required (`libkomp`)
  $\rightsquigarrow$ Need for an improved recursive **dataflow** scheduling

Conclusion

Fault tolerance based on evaluation codes

- RS and CRT codes extended to rational fractions
  $\rightsquigarrow$ smooth generalization
- Parameter oblivious decoding for early termination schemes
  $\rightsquigarrow$ parameter oblivious **list-decoding**? [Wu 08]
- Sparse evaluation codes
  $\rightsquigarrow$ Gap between best correction radius and existing algorithm

Perspectives

Large scale distributed exact linear algebra

- reducing communications [Demmel, Grigori and Xiang 11]
- sparse and hybrid (Boyer and Vialla) [Faugère and Lachartre 10]
- combine genericity and efficiency to attack crypto. challenges

Perspectives

Large scale distributed exact linear algebra

- reducing communications [Demmel, Grigori and Xiang 11]
- sparse and hybrid (Boyer and Vialla) [Faugère and Lachartre 10]
- combine genericity and efficiency to attack crypto. challenges

Polynomial matrix arithmetic for coding theory

- State of the art implementations in LinBox [Giorgi and Lebreton 14]
- Coding theory tools in Sage (Lucas)
- Further joint developments

Perspectives

Large scale distributed exact linear algebra

- reducing communications [Demmel, Grigori and Xiang 11]
- sparse and hybrid (Boyer and Vialla) [Faugère and Lachartre 10]
- combine genericity and efficiency to attack crypto. challenges

Polynomial matrix arithmetic for coding theory

- State of the art implementations in LinBox [Giorgi and Lebreton 14]
- Coding theory tools in Sage (Lucas)
- Further joint developments

Symbolic-numeric computation

- smooth transition between **noise** and **errors** for sparse codes [Comer Kaltofen P. 12], [Kaltofen Yang 13-14]
  $\rightsquigarrow$ improve decoding capacities and efficiency
  $\rightsquigarrow$ extend to larger classes of codes
- High precision floating point linear algebra via exact rational arithmetic

Perspectives

Large scale distributed exact linear algebra

- reducing communications [Demmel, Grigori and Xiang 11]
- sparse and hybrid (Boyer and Vialla) [Faugère and Lachartre 10]
- combine genericity and efficiency to attack crypto. challenges

Polynomial matrix arithmetic for coding theory

- State of the art implementations in LinBox [Giorgi and Lebreton 14]
- Coding theory tools in Sage (Lucas)
- Further joint developments

Symbolic-numeric computation

- smooth transition between **noise** and **errors** for sparse codes [Comer Kaltofen P. 12], [Kaltofen Yang 13-14]
  $\rightsquigarrow$ improve decoding capacities and efficiency
  $\rightsquigarrow$ extend to larger classes of codes
- High precision floating point linear algebra via exact rational arithmetic

Thank you