Algorithms and efficient implementations in exact linear algebra

Pascal Giorgi

soutenance HDR - 28 oct. 2019

Linear algebra is a widespread tool

- Assessing security in cryptography : NFS algo. [Lenstra(s) 93] 

- Coding theory in practice: list decoding [Guruswami, Sudan 98] 
- Weather forecast simulation 

Numerical linear algebra

⇒ approximated solutions : float

✓ dedicated hardware

✗ pb of stability

✓ mature developments

Exact linear algebra

⇒ exact solutions: $\mathbb{Z}, Z_p, \mathbb{Q}, \mathbb{Z}[X]$

✗ no dedicated hardware

✓ no stability issue

✗ slower development

Linear algebra is a widespread tool

- Assessing security in cryptography : NFS algo. [Lenstra(s) 93] 

- Coding theory in practice: list decoding [Guruswami, Sudan 98] 
- Weather forecast simulation 

Numerical linear algebra

$\Rightarrow$ approximated solutions : float

✓ dedicated hardware

✗ pb of stability

✓ mature developments

Exact linear algebra

$\Rightarrow$ exact solutions: $\mathbb{Z}, Z_p, \mathbb{Q}, \mathbb{Z}[X]$

✗ no dedicated hardware

✓ no stability issue

✓ improved over past 20 years

at the end of the talk

High performance linear algebra

- numerical computing $= O(n^3)$ classic algo. with optimized hardware
  $\Rightarrow$ efficient libraries BLAS, LAPACK (top 500 supercomputers)

High performance linear algebra

- numerical computing $= O(n^3)$ classic algo. with optimized hardware
  $\Rightarrow$ efficient libraries BLAS, LAPACK (top 500 supercomputers)

- exact computing $\neq$ numerical computing
  - reductions to core problems
  - adaptative implementations with thresholds

Matrix multiplication

- $O(n^{2.81})$ [Strassen 69]
- $\vdots$
- $O(n^{2.37})$ [Le Gall 2014]

Polynomial/Integer multiplication

- $O(d^{1.58})$ [Karatsuba 62]
- $\vdots$
- $O(d \log(d))$ [Harvey, Hoeven 2019]

High performance linear algebra

- numerical computing = $O(n^3)$ classic algo. with optimized hardware
  $\Rightarrow$ efficient libraries BLAS, LAPACK (top 500 supercomputers)

- exact computing $\neq$ numerical computing
  - reductions to core problems
  - adaptative implementations with thresholds

Matrix multiplication

- $O(n^{2.81})$ [Strassen 69]
- $\vdots$
- $O(n^{2.37})$ [Le Gall 2014]

Polynomial/Integer multiplication

- $O(d^{1.58})$ [Karatsuba 62]
- $\vdots$
- $O(d \log(d))$ [Harvey, Hoeven 2019]

$\Rightarrow$ finding best algo., reductions and thresholds is rather complicated

Exact linear algebra versatility

$$
\begin{bmatrix} 993 & 512 & 509 \\ 106 & 978 & 690 \\ 946 & 442 & 832 \end{bmatrix}^{-1} = \begin{cases} \begin{bmatrix} 648 & 98 & 16 \\ 648 & 839 & 305 \\ 31 & 193 & 516 \end{bmatrix} \text{ over } \mathbb{Z}_{997} \\ \\ \begin{bmatrix} \frac{14131}{9642515} & -\frac{11167}{19285030} & -\frac{8029}{19285030} \\ \frac{141137}{86782635} & \frac{172331}{173565270} & -\frac{157804}{86782635} \\ -\frac{219584}{86782635} & \frac{22723}{173565270} & \frac{458441}{173565270} \end{bmatrix} \text{ over } \mathbb{Q} \end{cases}
$$

expression swell → op. on entries can be more than $O(1)$

Exact linear algebra versatility

$$\begin{bmatrix} 993 & 512 & 509 \\ 106 & 978 & 690 \\ 946 & 442 & 832 \end{bmatrix}^{-1} = \begin{cases} \begin{bmatrix} 648 & 98 & 16 \\ 648 & 839 & 305 \\ 31 & 193 & 516 \end{bmatrix} \text{ over } \mathbb{Z}_{997} \\ \\ \begin{bmatrix} \frac{14131}{9642515} & -\frac{11167}{19285030} & -\frac{8029}{19285030} \\ \frac{141137}{86782635} & \frac{172331}{173565270} & -\frac{157804}{86782635} \\ -\frac{219584}{86782635} & \frac{22723}{173565270} & \frac{458441}{173565270} \end{bmatrix} \text{ over } \mathbb{Q} \end{cases}$$

expression swell → op. on entries can be more than O(1)

- algebraic *vs* bit (or word) complexity → different algo. reductions
- sparse *vs* dense approach

High performance libraries

Main goal:

implement fast algorithms

but take care of

constants, memory, parallelism

High performance libraries

Main goal:

implement fast algorithms

but take care of

constants, memory, parallelism

Our work in Exact Linear Algebra

- design "faster" algorithms
- provide their efficient implementation into libraries

$\Rightarrow$ 3 major libraries: Givaro [BCD+16]; FFLAS-FFPACK [DGP04; DGP08; DGLS18]; LinBox [DGG+02; BDG10; BDG+14]

Important dates in exact linear algebra

Major algorithmic breaktrough

- Gaussian Elimation in $O(n^\omega)$ with $\omega < 3$ [Strassen 69]
  $\Rightarrow$ influence algebraic complexity: reduction to matrix mult.
- Sparse linear algebra in $O(n^2)$ [Wiedemann 86, Coppersmith 90]
  $\Rightarrow$ provide iterative methods to finite fields
- Polynomial linear algebra in $\tilde{O}(n^\omega d)$ [Storjohann 02]
  $\Rightarrow$ influence bit complexity: reduction to matrix mult.

Outline

Dense linear algebra (post Strassen)

Sparse linear algebra - Polynomial linear algebra (a short round trip)

Beyond time: space and confidence

Outline

Dense linear algebra (post Strassen)

Sparse linear algebra - Polynomial linear algebra (a short round trip)

Beyond time: space and confidence

Dense linear algebra mod p

Problems reduces to matrix mult. : $O(n^\omega)$ op. in $\mathbb{Z}_p$, $\omega < 2.3728$ [Le Gall14]

$\Rightarrow$ linsys, det, inv [Strassen 69], rank [Ibarra et al. 82], minpoly, charpoly [Pernet et. al 07]

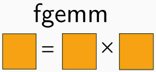
Dense linear algebra mod p

Problems reduces to matrix mult. : $O(n^\omega)$ op. in $\mathbb{Z}_p$, $\omega < 2.3728$ [Le Gall14]

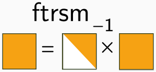
$\Rightarrow$ linsys, det, inv [Strassen 69], rank [Ibarra et al. 82], minpoly, charpoly [Pernet et. al 07]

FFLAS-FFPACK $\rightarrow$ reductions effective in practice [DGP08; DGP04; DGLS18]

- which matrix mult./reductions and how ?



$\longrightarrow$ subcubic algo., architecture optim. (BLAS)



$\longrightarrow$ minimize mod p

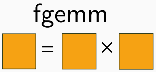
Dense linear algebra mod p

Problems reduces to matrix mult. : $O(n^\omega)$ op. in $\mathbb{Z}_p$, $\omega < 2.3728$ [Le Gall14]

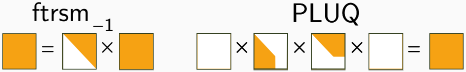
$\Rightarrow$ linsys, det, inv [Strassen 69], rank [Ibarra et al. 82], minpoly, charpoly [Pernet et. al 07]

FFLAS-FFPACK $\rightarrow$ reductions effective in practice [DGP08; DGP04; DGLS18]

- which matrix mult./reductions and how ?



$\longrightarrow$ subcubic algo., architecture optim. (BLAS)



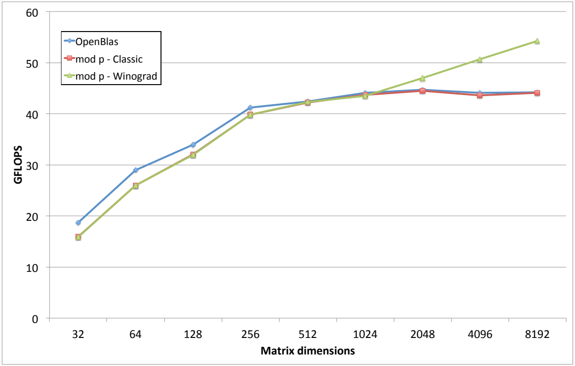
$\longrightarrow$ minimize mod p

- optimized half wordsize prime $p \Rightarrow$ larger p throught CRT

Matrix multiplication $\bmod p$ ($< 26 bits$)

FFLAS `fgemm` $\Rightarrow$ rely on numerical computation [Dumas et. al 02],[DGP08]

- delayed reductions mod p ✓ $O(n^2)$
- adaptative multiplication over $\mathbb{Z}$
  - ↪ t levels of Strassen-Winograd if $9^t \lfloor \frac{n}{2^t} \rfloor (p-1)^2 < 2^{53}$ ✓ $\omega < 3$
  - ↪ use BLAS as base case ✓ cache+simd



benchmark on Intel i7-4960HQ (2014), $p < 20$ bits

Matrix multiplication $\bmod p$ ($\geq 64 bits$)

No more native op. (e.g. $\mathbb{Z}_{1267650600228229401496703205653}$)
$\Rightarrow$ GMP library $\rightarrow$ too costly

Matrix multiplication $\bmod p$ ($\geq 64 bits$)

No more native op. (e.g. $\mathbb{Z}_{1267650600228229401496703205653}$)
$\Rightarrow$ GMP library $\rightarrow$ too costly

Most efficient solutions $\Rightarrow$ reduction to smaller prime(s) matrix mult.

- convert problem to polynomial matrix mult. $\bmod q$ (Kronecker)
  $\mathbb{Z} \rightarrow \mathbb{Z}_m \rightarrow \mathbb{Z}_q[X]_{<d} \rightarrow \mathbb{Z} \rightarrow \mathbb{Z}_p$

- convert problem to many matrix multiplication $\bmod p_i$ (CRT)
  $\mathbb{Z} \rightarrow \mathbb{Z}_m \rightarrow \underbrace{\mathbb{Z}_{p_1 \times \cdots \times p_d} \rightarrow \mathbb{Z}_{p_1} \times \cdots \times \mathbb{Z}_{p_d} \rightarrow \mathbb{Z}}_{\text{RNS conversions}} \rightarrow \mathbb{Z}_p$
  $AB \bmod (p_1 \times \cdots \times p_d) \longleftrightarrow (AB \bmod p_1, \ldots, AB \bmod p_d)$

Matrix multiplication $\bmod p$ ($\geq 64 bits$)

No more native op. (e.g. $\mathbb{Z}_{1267650600228229401496703205653}$)
$\Rightarrow$ GMP library $\rightarrow$ too costly

Most efficient solutions $\Rightarrow$ reduction to smaller prime(s) matrix mult.

- convert problem to polynomial matrix mult. $\bmod q$ (Kronecker)
  $\mathbb{Z} \rightarrow \mathbb{Z}_m \rightarrow \mathbb{Z}_q[X]_{<d} \rightarrow \mathbb{Z} \rightarrow \mathbb{Z}_p$

- convert problem to many matrix multiplication $\bmod p_i$ (CRT)
  $$\mathbb{Z} \rightarrow \mathbb{Z}_m \rightarrow \underbrace{\mathbb{Z}_{p_1 \times \cdots \times p_d} \rightarrow \mathbb{Z}_{p_1} \times \cdots \times \mathbb{Z}_{p_d} \rightarrow \mathbb{Z}}_{\text{RNS conversions}} \rightarrow \mathbb{Z}_p$$
  $$AB \bmod (p_1 \times \cdots \times p_d) \longleftrightarrow (AB \bmod p_1, \ldots, AB \bmod p_d)$$

How to improve the reduction ? especially RNS

RNS conversions in practice

Fast RNS conversions $O(d \log(d) \log\log(d))$ word op. [Borodin, Moenck 74]

$\Rightarrow$ hard to optimize in practice

RNS conversions in practice

Fast RNS conversions $O(d \log(d) \log\log(d))$ word op. [Borodin, Moenck 74]
$\Rightarrow$ hard to optimize in practice

Naive RNS conversions $O(d^2)$ word op.

RNS conversions in practice

Fast RNS conversions $O(d\log(d)\log\log(d))$ word op. [Borodin, Moenck 74]
$\Rightarrow$ hard to optimize in practice

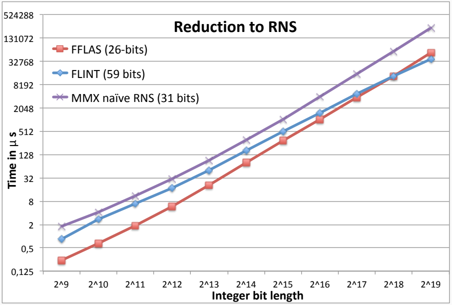
Naive RNS conversions $O(d^2)$ word op.
$\Rightarrow$ can be reduced to matrix mult. for many conversion [DGLS18]

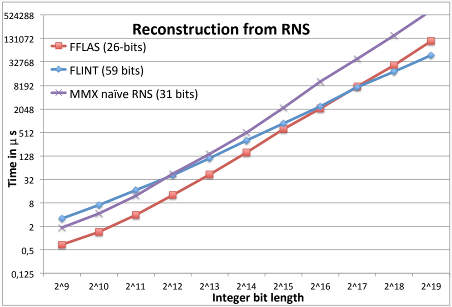
- pseudo-reduction:

$$A_0 + A_1\beta + \cdots A_{d-1}\beta^{d-1} \longrightarrow \begin{bmatrix} A_0 & \dots & A_{d-1} \end{bmatrix} \times (\beta^i \bmod p_j)_{i,j}$$
$$O(d) \longrightarrow O(\log d)$$

- r RNS conversions: $O(\mathbf{rd}^{\omega-\mathbf{1}}) + O(d^2)$ word op. (practical $\omega \simeq 2.81$)

Simultaneous conversions with RNS: in practice

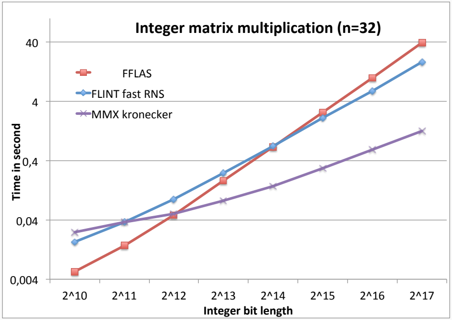


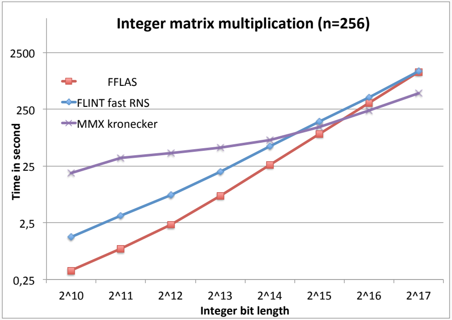


benchmark on Intel Xeon-E52697 (2017) - for matrix multiplication ($n = 128$)

- for good β and $p_j < 2^{26} \Rightarrow$ re-use BLAS (and possibly Strassen)
- can extend the p_j without sacrifying performance [DGLS18]

FFLAS integer matrix multiplication





benchmark on Intel Xeon-E52697 (2017)

- our solution: reduce everything to matrix mult. $O(n^\omega d + n^2 d^{\omega-1})$
- over $\mathbb{Z}_p \Rightarrow$ reduce afterward (slight slowdown)

Dense linear algebra modulo p: in practice

FFLAS approach: algo. reduction to `fgemm` [DGP04; DGP08]
$\Rightarrow$ but minimize modular reductions, only $O(n^2)$

Dense linear algebra modulo p: in practice

FFLAS approach: algo. reduction to `fgemm` [DGP04; DGP08]
$\Rightarrow$ but minimize modular reductions, only $O(n^2)$

Example with ftrsm:

$$\begin{bmatrix} A_1 & A_2 \\ & A_3 \end{bmatrix} \times \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}$$

- $A_3 X_2 = B_2 \bmod p$
- $D = B_1 - A_2 X_2$ over $\mathbb{Z}$
- $A_1 X_1 = D \bmod p$

reduce r.h.s $\bmod p$ when $n = 1$

Dense linear algebra modulo p: in practice

FFLAS approach: algo. reduction to `fgemm` [DGP04; DGP08]

$\Rightarrow$ but minimize modular reductions, only $O(n^2)$

Example with ftrsm:

$$\begin{bmatrix} A_1 & A_2 \\ & A_3 \end{bmatrix} \times \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}$$

- $A_3 X_2 = B_2 \bmod p$
- $D = B_1 - A_2 X_2$ over $\mathbb{Z}$
- $A_1 X_1 = D \bmod p$

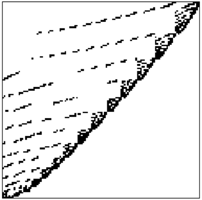
reduce r.h.s $\bmod p$ when $n = 1$

- p multi-precision : only $O(n^2)$ RNS conversions [G. HDR 19]
  $T(n,n) = 2T(n/2,n) + n^\omega d + \cancel{n^2 d^{\omega-1}} \rightarrow$ save a $\log(n)$

✓ practical performance ~ `fgemm` (matching up with constants)

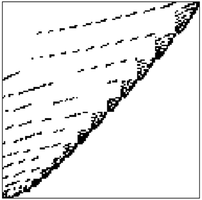
Outline

Exact sparse linear algebra



- $O(n)$ non zero elts among n^2
- Gauss fill-in : still $O(n^\omega)$ op in $\mathbb{K}$

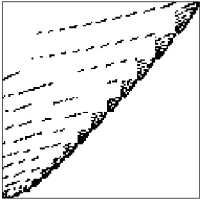
Exact sparse linear algebra



- $O(n)$ non zero elts among n^2
- Gauss fill-in : still $O(n^\omega)$ op in $\mathbb{K}$

Wiedemann (1986): $O(n^2)$ using minimal polynomial on $(uA^i v)_{i\geq 0}$
$\Rightarrow$ reduce to SpMV and polynomial GCD

Exact sparse linear algebra



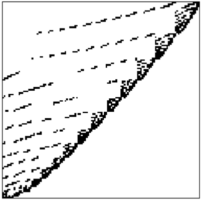
- $O(n)$ non zero elts among n^2
- Gauss fill-in : still $O(n^\omega)$ op in $\mathbb{K}$

Wiedemann (1986): $O(n^2)$ using minimal polynomial on $(uA^i v)_{i\geq 0}$
$\Rightarrow$ reduce to SpMV and polynomial GCD

Coppersmith (1990) : block variant
$\Rightarrow$ trade a sequence of integers to a shorter one of matrices $(UA^i V)_{i\geq 0}$

- more parallelism/ better proba. $\rightarrow$ great tool in crypto
- need fast algo./implem. on the degree (e.g. crypto [Thomé 02])

Exact sparse linear algebra



- $O(n)$ non zero elts among n^2
- Gauss fill-in : still $O(n^\omega)$ op in $\mathbb{K}$

Wiedemann (1986): $O(n^2)$ using minimal polynomial on $(uA^iv)_{i\geq 0}$
$\Rightarrow$ reduce to SpMV and polynomial GCD

Coppersmith (1990) : block variant
$\Rightarrow$ trade a sequence of integers to a shorter one of matrices $(UA^iV)_{i\geq 0}$

- more parallelism/ better proba. → great tool in crypto
- need fast algo./implem. on the degree (e.g. crypto [Thomé 02])

How to improve in theory/practice:
→ fast linear algebra with polynomials (fast on degree/dim)

Fast linear algebra with polynomials

Fast polynomial matrix mult. over $\mathbb{K}[X] \simeq O(n^\omega d + n^2 d \log(d))$ op in $\mathbb{K}$

Fast linear algebra with polynomials

Fast polynomial matrix mult. over $\mathbb{K}[X] \simeq O(n^{\omega} d + n^2 d \log(d))$ op in $\mathbb{K}$

Algorithmic reductions to matrix. mult: $\tilde{O}(n^{\omega} d)$

- linsys, Smith form, det [Storjohann 02]

Fast linear algebra with polynomials

Fast polynomial matrix mult. over $\mathbb{K}[X] \simeq O(n^\omega d + n^2 d \log(d))$ op in $\mathbb{K}$

Algorithmic reductions to matrix. mult: $\tilde{O}(n^\omega d)$

- linsys, Smith form, det [Storjohann 02]
- minimal approximant basis [GJV03]
  reduced basis of $\mathbb{K}[X]$-module , e.g. $\{qF = 0 \bmod X^d\}$
  $\Rightarrow$ central step in block Wiedemann method

Fast linear algebra with polynomials

Fast polynomial matrix mult. over $\mathbb{K}[X] \simeq O(n^\omega d + n^2 d \log(d))$ op in $\mathbb{K}$

Algorithmic reductions to matrix. mult: $\tilde{O}(n^\omega d)$

- linsys, Smith form, det [Storjohann 02]
- minimal approximant basis [GJV03]
  reduced basis of $\mathbb{K}[X]$-module , e.g. $\{qF = 0 \bmod X^d\}$
  $\Rightarrow$ central step in block Wiedemann method
- many derived reductions:
  - row reduction [GJV03]; Hermite [Gupta, Storjohann 11; Labahn et. al 17]
  - Popov [Beckerman et. al 06; Neiger 16]; inverse [Jeannerod, Villard 05; Zhou et. al 15]
  - rank, nullspace [Storjohann, Villard 05; Zhou et. al 12]

Fast minimal approximant basis in LinBox

state of the art implem. of PM-Basis algo. [GJV03] ($\mathbb{K} = \mathbb{Z}_p$)

- base case : kernel basis over $\mathbb{Z}_p$ → FFLAS-FFPACK
- fast matrix mult. in $\mathbb{Z}_p[X]$ [DEGU07; GL14] (ANR HPAC)

⇒ practical algo. : $\log(d)$ slowdown from matrix mult.

Fast minimal approximant basis in LinBox

state of the art implem. of PM-Basis algo. [GJV03] ($\mathbb{K} = \mathbb{Z}_p$)

- base case : kernel basis over $\mathbb{Z}_p$ → FFLAS-FFPACK
- fast matrix mult. in $\mathbb{Z}_p[X]$ [DEGU07; GL14] (ANR HPAC)

⇒ practical algo. : $\log(d)$ slowdown from matrix mult.

Application: block Wiedemann rank mod 65537 [DEGU07]
↪ matrix size 2 million with 40 million non-zero ; 35 days (50 cores)

Fast minimal approximant basis in LinBox

state of the art implem. of PM-Basis algo. [GJV03] ($\mathbb{K} = \mathbb{Z}_p$)

- base case : kernel basis over $\mathbb{Z}_p \rightarrow$ FFLAS-FFPACK
- fast matrix mult. in $\mathbb{Z}_p[X]$ [DEGU07; GL14] (ANR HPAC)

$\Rightarrow$ practical algo. : $\log(d)$ slowdown from matrix mult.

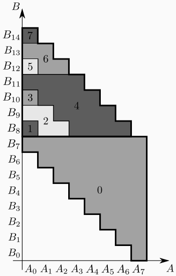
Application: block Wiedemann rank mod 65537 [DEGU07]
$\hookrightarrow$ matrix size 2 million with 40 million non-zero ; 35 days (50 cores)
$\Rightarrow$ some (UA^iV) are not necessary when rank deficient

Fast online minimal approximant basis

Variant of PM-Basis minimizing dependency on F [GL14]

- iterative DAC algo.
- half line middle product for updating $F \to \tilde{O}(n^\omega d)$

$\Rightarrow$ practical algo. : $\log^2(d)$ slowdown from matrix mult.

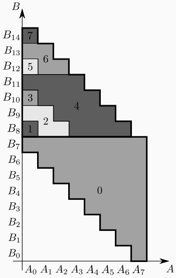


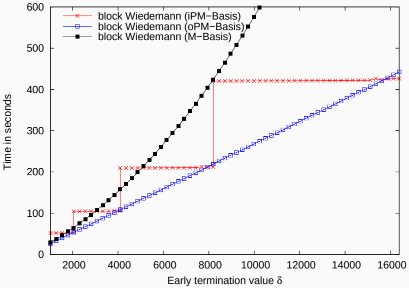
Fast online minimal approximant basis

Variant of PM-Basis minimizing dependency on F [GL14]

- iterative DAC algo.
- half line middle product for updating $F \to \tilde{O}(n^\omega d)$

$\Rightarrow$ practical algo. : $\log^2(d)$ slowdown from matrix mult.



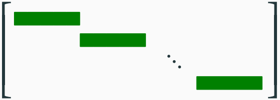


Main application:
rank deficiency with block Wiedemann
E.T. $\Rightarrow$ reduce staircase effect

Improving sparse linear algebra

key point: re-use fast polynomial linear algebra [EGG+06; EGG+07]

- prove existence of sparse projections for block Wiedemann

$U =$  with  a vector of $1's$

- blackbox matrix inverse over $\mathbb{K}$: $\tilde{O}(n^{2.5})$ op in $\mathbb{K}$

  $\Rightarrow$ sparse projection and PM-Basis

- solve $Ax = b$ over $\mathbb{Q}$ in $\tilde{O}(n^{2.5} \log \|A\|)$ word op.
  - improve classical approaches: $\tilde{O}(n^3 \log \|A\|)$ word op.
  - practical algo → thanks to PM-Basis (dim $\simeq$ deg)

Outline

Improving the confidence in computed results

We need to trust the results from libraries or Google cloud

Improving the confidence in computed results

We need to trust the results from libraries or Google cloud

How to sustain confidence:

- prove the code (COQ) or its execution (interactive proof)
- verify the solution ⇒ e.g. $Ax = b$ or $uC = (uA)B$ [Freivalds 79]

Improving the confidence in computed results

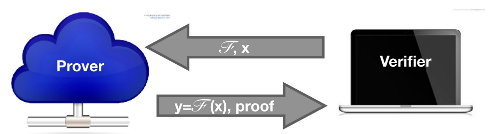
We need to trust the results from libraries or Google cloud

How to sustain confidence:

- prove the code (COQ) or its execution (interactive proof)
- verify the solution ⇒ e.g. $Ax = b$ or $uC = (uA)B$ [Freivalds 79]

last solution is adapted to linear algebra (no linear complexity)

Verified computation model

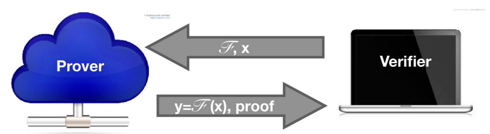


- generating proof negligible
- verifying proof easier than $\mathcal{F}(x)$

Generic approaches in linear algebra [Goldwasser et. al 08, Kaltofen et. al 11]

$\Rightarrow$ non optimal prover or verifier time

Verified computation model



- generating proof negligible
- verifying proof easier than $\mathcal{F}(x)$

Generic approaches in linear algebra [Goldwasser et. al 08, Kaltofen et. al 11]
$\Rightarrow$ non optimal prover or verifier time

Optimal verification algorithms exist [Dumas, Kaltofen 14]:

e.g. over $\mathbb{Z}_p$, $\quad \mathrm{rank}(A) = r \iff A = PLUQ$ and $\mathrm{rank}(U) = r$

$\Rightarrow$ optimal prover/verifier time + "independency" from implementation

Optimal certificates in linear algebra

Most classical problems have an "optimal" probabilistic certificate:

- sparse, dense over $\mathbb{K}$, $\mathbb{Z}$ and $\mathbb{Q}$
- very often with interactive protocols

Optimal certificates in linear algebra

Most classical problems have an "optimal" probabilistic certificate:

- sparse, dense over $\mathbb{K}$, $\mathbb{Z}$ and $\mathbb{Q}$
- very often with interactive protocols

Our concerns: computation over $\mathbb{K}[X]$

- minimal approximant basis as a good candidate
- is interactivity really required ?

verifying $A(x)B(x) = 0 \bmod X^k$ not so easy

Our main tool: multiplications as linear maps

Example:

$$f = 3X^2 + 2X + 1, \quad g = X^2 + 2X + 4 \qquad \Rightarrow \qquad \begin{bmatrix} 1 & & \\ 2 & 1 & \\ 3 & 2 & 1 \\ & 3 & 2 \\ & & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 10 \\ 17 \\ 8 \\ 3 \end{bmatrix}$$

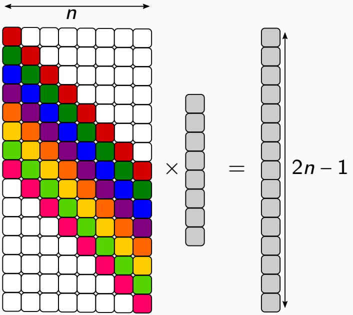
$$fg = 3X^4 + 8X^3 + 17X^2 + 10X + 4$$

Our main tool: multiplications as linear maps

Example:

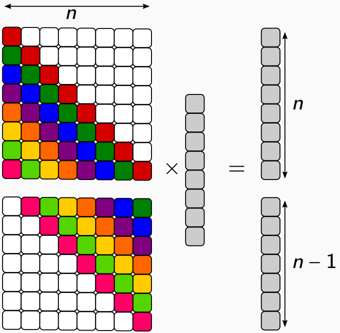
$$f = 3X^2 + 2X + 1, \quad g = X^2 + 2X + 4 \quad \Rightarrow \quad \begin{bmatrix} 1 & & \\ 2 & 1 & \\ 3 & 2 & 1 \\ & 3 & 2 \\ & & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 10 \\ 17 \\ 8 \\ 3 \end{bmatrix}$$

$$fg = 3X^4 + 8X^3 + 17X^2 + 10X + 4$$



Full product (FP)

$\Longrightarrow$



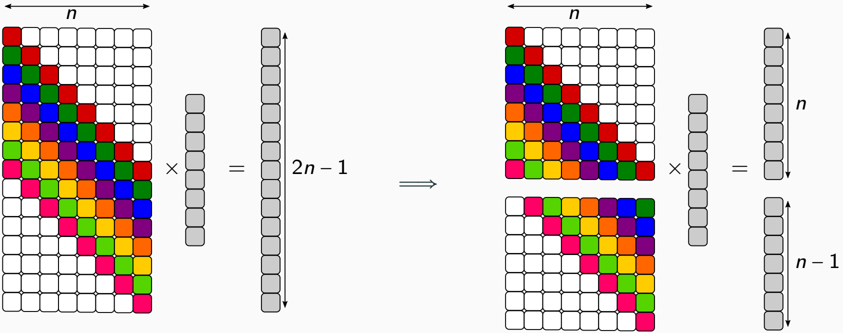
Short products (SP_{lo}, SP_{hi})

Our main tool: multiplications as linear maps

Example:

$$f = 3X^2 + 2X + 1, \quad g = X^2 + 2X + 4 \qquad \Rightarrow \qquad \begin{bmatrix} 1 & & \\ 2 & 1 & \\ 3 & 2 & 1 \\ & 3 & 2 \\ & & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 10 \\ 17 \\ 8 \\ 3 \end{bmatrix}$$

$$fg = 3X^4 + 8X^3 + 17X^2 + 10X + 4$$



Full product (FP) — Short products (SP_{lo}, SP_{hi})

→ eval. $X = \alpha \quad \simeq \quad$ mult. by $\begin{bmatrix} 1 & \alpha & \dots & \alpha^{n-1} \end{bmatrix}$

Verifying truncated polynomial products

Low short product over $\mathbb{K}[X]$:
verify $C(X) = A(X)B(X) \bmod X^k$ in optimal time [Gio18]

$$\begin{bmatrix}1 & \alpha & \dots & \alpha^{k-1}\end{bmatrix}\begin{bmatrix} a_0 & & & \\ a_1 & \ddots & & \\ \vdots & \ddots & \ddots & \\ a_{k-1} & \cdots & a_1 & a_0 \end{bmatrix}\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{k-1}\end{bmatrix} = \begin{bmatrix}1 & \alpha & \dots & \alpha^{k-1}\end{bmatrix}\begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{k-1}\end{bmatrix}$$

Verifying truncated polynomial products

Low short product over $\mathbb{K}[X]$:

verify $C(X) = A(X)B(X) \bmod X^k$ in optimal time [Gio18]

$$\begin{bmatrix} 1 & \alpha & \dots & \alpha^{k-1} \end{bmatrix} \begin{bmatrix} a_0 & & & \\ a_1 & \ddots & & \\ \vdots & \ddots & \ddots & \\ a_{k-1} & \cdots & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{k-1} \end{bmatrix} = C(\alpha)$$

Verifying truncated polynomial products

Low short product over $\mathbb{K}[X]$:
verify $C(X) = A(X)B(X) \bmod X^k$ in optimal time [Gio18]

$$\begin{bmatrix} A(\alpha) & \dots & \alpha^{k-j}(A \bmod X^j)(\alpha) & \dots & \alpha^{k-1}a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_k \end{bmatrix} = C(\alpha)$$

Verifying truncated polynomial products

Low short product over $\mathbb{K}[X]$:
verify $C(X) = A(X)B(X) \bmod X^k$ in optimal time [Gio18]

$$\left[A(\alpha) \ \dots \ \alpha^{k-j}(A \bmod X^j)(\alpha) \ \dots \ \alpha^{k-1}a_0\right] \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_k \end{bmatrix} = C(\alpha)$$

- $O(k)$ op. in $\mathbb{K}$ → evaluating $\alpha^{k-1}\mathrm{rev}(A)$ at $X = \alpha^{-1}$
- no interactivity

⇒ similar result for Toeplitz matrix: e.g. middle product
⇒ optimal also for polynomial matrix mult. → use Freivald

Verifying minimal approximant basis

Optimal certificate [GN18]

$P \in \mathbb{K}[X]^{n\times n}$ is a minimal approximant basis of $\mathcal{A}_d(F)$ iff

- P is row-reduced (**minimality**)
- $PF = 0 \bmod X^d$ (**approximant**)
- $\det(P) = X^\delta$
- $\begin{bmatrix} P(0) & C(0) \end{bmatrix}$ is full rank with $C = PFX^{-d}$
  $\Rightarrow C(0)$ as certificate

Verifying minimal approximant basis

Optimal certificate [GN18]

$P \in \mathbb{K}[X]^{n\times n}$ is a minimal approximant basis of $\mathcal{A}_d(F)$ iff

- P is row-reduced (**minimality**) ✓ linear algebra over $\mathbb{K}$
- $PF = 0 \bmod X^d$ (**approximant**) ✓ trunc. polynomial product
- $\det(P) = X^\delta$ ✓ linear algebra over $\mathbb{K}$
- $\begin{bmatrix} P(0) & C(0) \end{bmatrix}$ is full rank with $C = PFX^{-d}$ ✓ linear algebra over $\mathbb{K}$
  $\Rightarrow C(0)$ as certificate ✓ trunc. polynomial product

Verifying minimal approximant basis

Optimal certificate [GN18]

$P \in \mathbb{K}[X]^{n\times n}$ is a minimal approximant basis of $\mathcal{A}_d(F)$ iff

- P is row-reduced (**minimality**) ✓ linear algebra over $\mathbb{K}$
- $PF = 0 \bmod X^d$ (**approximant**) ✓ trunc. polynomial product
- $\det(P) = X^\delta$ ✓ linear algebra over $\mathbb{K}$
- $\begin{bmatrix} P(0) & C(0) \end{bmatrix}$ is full rank with $C = PFX^{-d}$ ✓ linear algebra over $\mathbb{K}$
  $\Rightarrow C(0)$ as certificate ✓ trunc. polynomial product

- prover: P in $\tilde{O}(n^\omega d)$ and $C(0)$ in $O(n^\omega d)$
- verifier: almost $O(\mathrm{size}(P) + \mathrm{size}(F))$, almost no-interactivity

$\Rightarrow$ work for non-uniform truncation: $\bmod X^{\mathbf{d}}$ with $\mathbf{d} = (d_1, \ldots, d_m)$

Minimizing memory requirement

Fast algorithms usually trade time by memory:

Minimizing memory requirement

Fast algorithms usually trade time by memory:

$$\text{polynomial mult.} = \begin{cases} \text{naive} \rightarrow O(n^2) \text{ time and } O(1) \text{ extra space} \\ \text{Karatsuba} \rightarrow O(n^{1.58}) \text{ time and } O(n) \text{extra space} \end{cases}$$

$\Rightarrow$ data can be ~ 1To (e.g. crypto DLP)

Minimizing memory requirement

Fast algorithms usually trade time by memory:

$$\text{polynomial mult.} = \begin{cases} \text{naive} \rightarrow O(n^2) \text{ time and } O(1) \text{ extra space} \\ \text{Karatsuba} \rightarrow O(n^{1.58}) \text{ time and } O(n) \text{extra space} \end{cases}$$

$\Rightarrow$ data can be ~ 1To (e.g. crypto DLP)

Our concern: generic approach for in-place multiplication ?

Algorithms	Time complexity	Space complexity
naive	$2n^2 + 2n - 1$	O(1)
Karatsuba ['62]	$< 6.5n^{1.58}$	$\leq 2n + 5\log(n)$
Karatsuba [(Roche'09)]	$< 10n^{1.58}$	$\leq 5\log(n)$
Toom-3 ['63]	$< \frac{73}{4}n^{1.46}$	$\leq 2n + 5\log_3(n)$
FFT [CT'65]	$9n\log(2n)$	$2n$
FFT [Roche'09]	$11n\log(2n)$	$O(1)$

Our technique: again, multiplication as a linear map

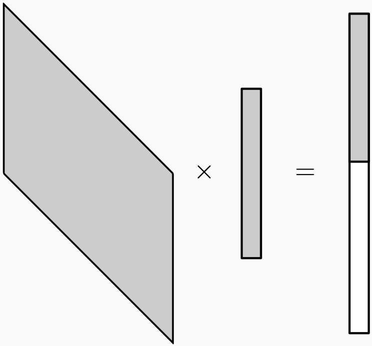
In-place polynomial multiplication

A first positive answer from [GGR19]

Given algo. with $\mathsf{M}(n)$ time complexity and cn space complexity

- in-place full product (half-additive) in time $O((2c+7)\mathsf{M}(n))$

$$(f_0 + X^k \hat{f}) \cdot (g_0 + X^k \hat{g}) = f_0 g_0 + X^k (f_0 \hat{g} + \hat{f} g_0) + X^{2k} \hat{f} \hat{g}$$



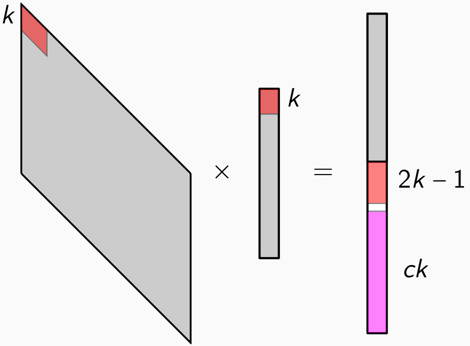
In-place polynomial multiplication

A first positive answer from [GGR19]

Given algo. with $\mathsf{M}(n)$ time complexity and cn space complexity

- in-place full product (half-additive) in time $O((2c+7)\mathsf{M}(n))$

$$(f_0 + X^k \hat{f}) \cdot (g_0 + X^k \hat{g}) = f_0 g_0 + X^k (f_0 \hat{g} + \hat{f} g_0) + X^{2k} \hat{f} \hat{g}$$



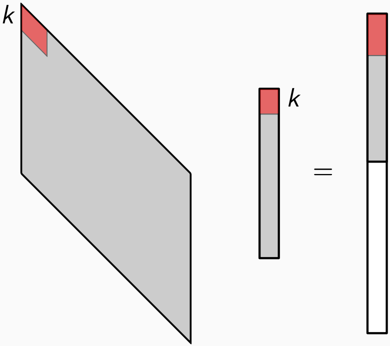
In-place polynomial multiplication

A first positive answer from [GGR19]

Given algo. with $\mathsf{M}(n)$ time complexity and cn space complexity

- in-place full product (half-additive) in time $O((2c+7)\mathsf{M}(n))$

$$(f_0 + X^k \hat{f}) \cdot (g_0 + X^k \hat{g}) = f_0 g_0 + X^k (f_0 \hat{g} + \hat{f} g_0) + X^{2k} \hat{f} \hat{g}$$



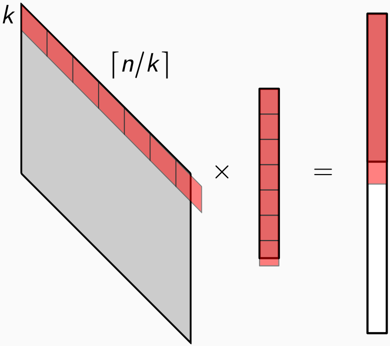
In-place polynomial multiplication

A first positive answer from [GGR19]

Given algo. with $\mathsf{M}(n)$ time complexity and cn space complexity

- in-place full product (half-additive) in time $O((2c+7)\mathsf{M}(n))$

$$(f_0 + X^k \hat{f}) \cdot (g_0 + X^k \hat{g}) = f_0 g_0 + X^k (f_0 \hat{g} + \hat{f} g_0) + X^{2k} \hat{f} \hat{g}$$



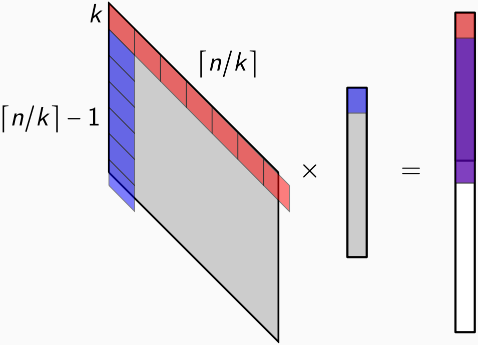
In-place polynomial multiplication

A first positive answer from [GGR19]

Given algo. with $\mathsf{M}(n)$ time complexity and cn space complexity

- in-place full product (half-additive) in time $O((2c+7)\mathsf{M}(n))$

$$(f_0 + X^k \hat{f}) \cdot (g_0 + X^k \hat{g}) = f_0 g_0 + X^k (f_0 \hat{g} + \hat{f} g_0) + X^{2k} \hat{f} \hat{g}$$



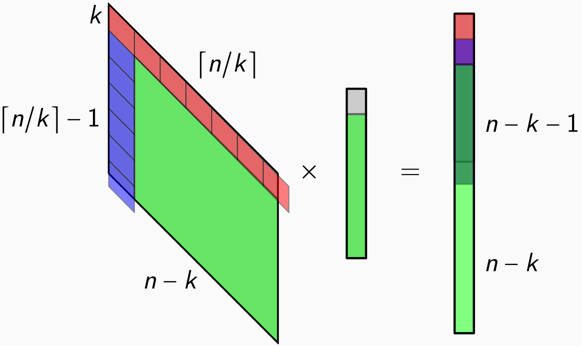
In-place polynomial multiplication

A first positive answer from [GGR19]

Given algo. with $\mathsf{M}(n)$ time complexity and cn space complexity

- in-place full product (half-additive) in time $O((2c+7)\mathsf{M}(n))$

$$(f_0 + X^k\hat{f})\cdot(g_0 + X^k\hat{g}) = f_0g_0 + X^k(f_0\hat{g} + \hat{f}g_0) + X^{2k}\hat{f}\hat{g}$$



In-place polynomial multiplication

A first positive answer from [GGR19]

Given algo. with $\mathsf{M}(n)$ time complexity and cn space complexity

- in-place full product (half-additive) in time $O((2c+7)\mathsf{M}(n))$
- in-place short product in time $O((2c+5)\mathsf{M}(n))$
- in-place middle product in time $O(\mathsf{M}(n)\log(n))$

Take-home message

Efficient dense linear algebra mod p

fast integer matrix multiplication + delayed mod p

$\Rightarrow$ influence libraries in computer algebra (e.g. Maple, SageMath, ...)

Efficient reductions in theory/practice

Polynomial linear algebra	$\Leftarrow$	High Order lifting
$\Uparrow$		$\Uparrow$
PM-Basis algorithm†	$\Leftarrow$	**polynomial matrix mult.** †‡
$\Downarrow$		$\Uparrow$
Sparse linear algebra (BW)		**FFT +** `fgemm`**+ RNS**

(†) with optimal certificate; (‡) with in-place variant

Perspectives

Toward better space complexity

- in-place reductions to polynomial mult.
  - series inversion, division
  - multipoint evaluation/interpolation
  - any linear maps $\Rightarrow$ find generic approach
- hard problems: composition of linear maps, use input as output

$\Rightarrow$ applicable in practice ? usefullness in BW ?

Perspectives

Toward better space complexity

- in-place reductions to polynomial mult.
  - series inversion, division ✓ $O(\mathsf{M}(d))$
  - multipoint evaluation/interpolation ✓ $O(\mathsf{M}(d)\log^3(d))$
  - any linear maps ⇒ find generic approach
- hard problems: composition of linear maps, use input as output

⇒ applicable in practice ? usefullness in BW ?

Perspectives

Toward better space complexity

- in-place reductions to polynomial mult.
  - series inversion, division ✓ $O(\mathsf{M}(d))$
  - multipoint evaluation/interpolation ✓ $O(\mathsf{M}(d)\log^3(d))$
  - any linear maps ⇒ find generic approach
- hard problems: composition of linear maps, use input as output

⇒ applicable in practice ? usefullness in BW ?

Optimal certificates in linear algebra

- linear maps: what transposition tells us about certification ?
- extension to other bases of $\mathbb{K}[X]$-submodule (e.g. interpolant basis)
  ⇒ modular multiplication ?

Perspectives

Further software improvements

- effective reduction to polynomial matrix mult. [Hyun, Neiger, Schost 19]
- linear algebra mod p for "medium range" p

Sparse polynomial arithmetic

starting thesis (10/2019): Armelle Perret Du Cray

$\Rightarrow$ practical quasi-linear time algo. (proba.)

- product certification, multiplication
- division and harder problems (e.g. gcd, factorization)

Perspectives

Further software improvements

- effective reduction to polynomial matrix mult. [Hyun, Neiger, Schost 19]
- linear algebra $\bmod p$ for "medium range" p

Sparse polynomial arithmetic

starting thesis (10/2019): Armelle Perret Du Cray

$\Rightarrow$ practical quasi-linear time algo. (proba.)

- product certification, multiplication ✓ quasi-linear
- division and harder problems (e.g. gcd, factorization)

Thank you !!!

References

[GGR19] P. Giorgi, B. Grenet, and D. S. Roche. "Generic reductions for in-place polynomial multiplication". In: *Proceedings of the 2019 international symposium on symbolic and algebraic computation*. ISSAC'19. To appear at ISSAC'19. 2019.

[DGLS18] J. Doliskani, P. Giorgi, R. Lebreton, and É. Schost. "Simultaneous conversions with the Residue Number System using linear algebra". In: *ACM Transactions on Mathematical Software* 44.3 (2018), 27:1–27:21. DOI: 10.1145/3145573.

[Gio18] P. Giorgi. "A probabilistic algorithm for verifying polynomial middle product in linear time". In: *Information Processing Letters* 139 (2018), pp. 30–34. DOI: 10.1016/j.ipl.2018.06.014.

[GN18] P. Giorgi and V. Neiger. "Certification of minimal approximant bases". In: *Proceedings of the 2018 international symposium on symbolic and algebraic computation*. ISSAC'18. ACM, 2018, pp. 67–174. DOI: 10.1145/3208976.3208991.

[BCD+16] A. Breust, C. Chabot, J.-G. Dumas, L. Fousse, and P. Giorgi. "Recursive double-size fixed precision arithmetic". In: *ICMS: International Congress on Mathematical Software*. Vol. 9725. Lecture Notes in Computer Science. 2016, pp. 223–231. DOI: 10.1007/978-3-319-42432-3_28.

[BDG+14] B. Boyer, J.-G. Dumas, P. Giorgi, C. Pernet, and B.-D. Saunders. "Elements of Design for Containers and Solutions in the LinBox Library". In: *ICMS: International Congress on Mathematical Software*. Vol. 8592. Lecture Notes in Computer Science. 2014, pp. 654–662. DOI: 10.1007/978-3-662-44199-2_98.

[GL14] P. Giorgi and R. Lebreton. "Online Order Basis and its impact on Block Wiedemann Algorithm". In: *Proceedings of the 2014 international symposium on symbolic and algebraic computation*. ISSAC'14. ACM, 2014, pp. 202–209. DOI: 10.1145/2608628.2608647.

[BDG10] B. Boyer, J.-G. Dumas, and P. Giorgi. "Exact Sparse Matrix-Vector Multiplication on GPU's and Multicore Architectures". In: *Proceedings of PASCO'10: Parallel Symbolic Computation*. ACM, 2010, pp. 80–88. DOI: 10.1145/1837210.1837224.

[DGP08] J.-G. Dumas, P. Giorgi, and C. Pernet. "Dense Linear Algebra over Finite Fields: the FFLAS and FFPACK package". In: *ACM Transactions on Mathematical Software* 35 (2008), 19:1–19:42. DOI: 10.1145/1391989.1391992.

[DEGU07] J.-G. Dumas, P. Elbaz-Vincent, P. Giorgi, and A. Urbánska. "Parallel computation of the rank of large sparse matrices from algebraic K-theory". In: *Proceedings of the 2007 international workshop on parallel symbolic computation*. PASCO '07. ACM, 2007, pp. 43–52. DOI: 10.1145/1278177.1278186.

[EGG+07] W. Eberly, M. Giesbrecht, P. Giorgi, A. Storjohann, and G. Villard. "Faster inversion and other black box matrix computations using efficient block projections". In: *Proceedings of the 2007 international symposium on symbolic and algebraic computation*. ISSAC '07. ACM, 2007, pp. 143–150. DOI: 10.1145/1277548.1277569.

[EGG+06] W. Eberly, M. Giesbrecht, P. Giorgi, A. Storjohann, and G. Villard. "Solving sparse rational linear systems". In: *Proceedings of the 2006 international symposium on symbolic and algebraic computation*. ISSAC '06. ACM, 2006, pp. 63–70. DOI: 10.1145/1145768.1145785.

[DGP04] J.-G. Dumas, P. Giorgi, and C. Pernet. "FFPACK: finite field linear algebra package". In: *Proceedings of the 2004 international symposium on symbolic and algebraic computation*. ISSAC '04. ACM, 2004, pp. 119–126. DOI: 10.1145/1005285.1005304.

[GJV03] P. Giorgi, C.-P. Jeannerod, and G. Villard. "On the complexity of polynomial matrix computations". In: *Proceedings of the 2003 international symposium on symbolic and algebraic computation*. ISSAC '03. ACM, 2003, pp. 135–142. DOI: 10.1145/860854.860889.

[DGG+02] J.-G. Dumas, T. Gautier, M. Giesbrecht, P. Giorgi, B. Hovinen, E. Kaltofen, B. Saunders, W. J. Turner, and G. Villard. "Linbox: A Generic Library For Exact Linear Algebra". In: *Proceedings of the 2002 International Congress of Mathematical Software*. World Scientific, 2002.