Online order basis algorithm and its application to block Wiedemann algorithm*

PASCAL GIORGI ROMAIN LEBRETON

University of Waterloo

March 21st, 2014

*. This document has been written using the GNU $\text{T}_\text{E}\text{X}_{\text{MACS}}$ text editor (see www.texmacs.org).

Outline

1. Order basis
    a. Definition
    b. Algorithms
2. Application to block Wiedemann algorithm
3. Contributions :
    a. Fast iterative order basis
    b. Fast online order basis
    c. Timings

Let $\mathbb{K}$ be a field and $F \in \mathbb{K}[[x]]^{m \times n}$.

Let (F, σ) be the $\mathbb{K}[x]$-module $\{v \in \mathbb{K}[[x]]^{1 \times m}$ such that $vF = 0 \bmod x^\sigma\}$.

Definition. *An (F, σ) order basis P is a basis of (F, σ) of minimal degree.*

Minimal degree ?

1. Row degree :

$$\mathrm{rdeg}(P_1, \ldots, P_n) = \max(\deg P_i), \quad \mathrm{rdeg}\left(\begin{pmatrix} \text{row } 1 \\ \vdots \\ \text{row } n \end{pmatrix}\right) = (\mathrm{rdeg}(\text{row } i))_{i=1\ldots n}$$

   Partial order $(v_1, \ldots, v_m) \leqslant (w_1, \ldots, w_m) \Leftrightarrow \forall i \quad v_i \leqslant w_i$ on the sorted vector

2. Shifted row degree : $\mathrm{rdeg}_{\vec{s}}(P_1, \ldots, P_n) = \max(\deg P_i + s_i)$ where $\vec{s} = (s_1, \ldots, s_n)$

   $\rightsquigarrow$ $(F, \sigma, \vec{s})$ order basis : minimal for the $\vec{s}$-row degree

Let $\mathbb{K}$ be a field and $F \in \mathbb{K}[[x]]^{m \times n}$.

Let (F, σ) be the $\mathbb{K}[x]$-module $\{v \in \mathbb{K}[[x]]^{1 \times m}$ such that $vF = 0 \bmod x^\sigma\}$.

Definition. *An (F, σ) order basis P is a basis of (F, σ) of minimal degree.*

Lemma. *There exists a basis P of minimal degree.*

Remark. Existence but no unicity ($\rightsquigarrow$ Popov form).

Example. (taken from Zhou thesis)

$$\underbrace{\begin{pmatrix} 1 & 0 & 1 & 1 \\ x & 1 & 1+x & 0 \\ 1 & x^2+x^3 & x & 0 \\ x^2 & 0 & x^3+x^4 & 0 \end{pmatrix}}_{(F,8,\vec{0})-\text{order basis over } \mathbb{F}_2} \underbrace{\begin{pmatrix} x+x^2+x^3+x^4+x^5+x^6 \\ 1+x+x^5+x^6+x^7 \\ 1+x^2+x^4+x^5+x^6+x^7 \\ 1+x+x^3+x^7 \end{pmatrix}}_{F \text{ in } \mathbb{F}_2[[x]]^{4\times 1}} = 0^{4\times 1} \bmod x^8$$

1. Base case $\sigma = 1$

Algorithm Basis : $F \bmod x \longrightarrow$ a $(F, 1)$ order basis

Basic idea :

If $\begin{pmatrix} K \\ S \end{pmatrix} (F \bmod x) = \begin{pmatrix} 0 \\ R \end{pmatrix}$ with R full rank

then $\begin{pmatrix} K \\ x\,S \end{pmatrix} (F \bmod x) = \begin{pmatrix} 0 \\ x\,R \end{pmatrix} = (0) \bmod x$

$\rightarrow$ $\begin{pmatrix} K \\ x\,S \end{pmatrix}$ is a basis of the module $(F, 1)$.

2. Splitting the order basis problem

Theorem.
Let P_1 *be a* $(F, \sigma_1, \vec{s})$ *order basis of* $\vec{s}$*-degree* $\vec{u}$.
Let P_2 *be a* $((P_1\, F \operatorname{div} x^{\sigma_1}), \sigma_2, \vec{u})$ *order basis of* $\vec{u}$*-degree* $\vec{v}$.

Then $P_2\, P_1$ *is a* $(F, \sigma_1 + \sigma_2, \vec{s})$ *order basis of* $\vec{s}$*-degree* $\vec{v}$.

Remarks.

- $P_1\, F = x^{\sigma_1} M$ where $M = (P_1\, F \operatorname{div} x^{\sigma_1}) \in \mathbb{K}[[x]]^{m \times n}$

  $P_2\, P_1\, F = x^{\sigma_1} P_2\, M = x^{\sigma_1}(x^{\sigma_2} M') = (0) \bmod x^{\sigma_1 + \sigma_2}$

- The module $(F, \sigma_1 + \sigma_2, \vec{s})$ is a subset of $(F, \sigma_1, \vec{s})$ of basis P_1

  $\rightsquigarrow$ Express the module $(F, \sigma_1 + \sigma_2, \vec{s})$ on the basis P_1 $\rightarrow$ reduce the problem

- Need of $\vec{s}$-row degree:

  The row total degree of $v\, P$ is the exactly the $\vec{s}$-row degree of v where $\vec{s}$ is the row degree of P $\square$

Input : $F \in \mathbb{K}[x]^{m \times n}, \sigma \in \mathbb{N}$

Output : A (F, σ) order basis P of F

1. Quadratic algorithm M-Basis

Iterative : $(F, 1) \to (F, 2) \to (F, 3) \to \cdots \to (F, \sigma)$

Algorithm M-Basis

1. $P_0 := \text{Basis}(F \bmod x)$
2. **for** $k = 1, \ldots, \sigma - 1$ **do**
3. $\quad F' := x^{-k} P_{k-1} F$
4. $\quad M_k := \text{Basis}(F' \bmod x)$
5. $\quad P_k := M_k P_{k-1}$
6. **return** $P_{\sigma-1}$

In terms of polynomial multiplication, naive multiplication $P_{\sigma-1} = M_{\sigma-1} (\cdots M_3 (M_2\, M_1))$ where each M_i is of degree one.

Input : $F \in \mathbb{K}[x]^{m \times n}, \sigma \in \mathbb{N}$

Output : A (F, σ) order basis P of F

2. Quasi-linear algorithm PM-Basis

Divide-and-conquer : $(F,1) \to (F,2) \to (F,4) \to \cdots \to (F,\sigma/2) \to (F,\sigma)$

Algorithm PM-Basis

1. **if** $\sigma = 1$ **then**
2. **return** Basis($F \bmod x$)
3. **else**
4. $P_{\text{low}} := \text{PM-Basis}(F, \lfloor \sigma/2 \rfloor)$ — First subproblem
5. $F' := \text{MiddleProduct}(P_{\text{low}}, F, \lfloor \sigma/2 \rfloor \ldots \sigma - 1)$ — Update problem
6. $P_{\text{high}} := \text{PM-Basis}(F', \lceil \sigma/2 \rceil)$ — Second subproblem
7. **return** $P_{\text{high}} \cdot P_{\text{low}}$ — Solve original problem

In terms of polynomial multiplication, binary multiplication tree.

1. Order basis
   a. Definition
   b. Algorithms

2. Application to block Wiedemann algorithm

3. Contributions :
   a. Fast iterative order basis
   b. Fast online order basis
   c. Timings

Algorithm - Wiedemann '86

Input: A sparse matrix $A \in \mathcal{M}_N(\mathbb{K})$
Output: Its minimal polynomial

1. Choose random $u, v \in \mathbb{K}^{N \times 1}$
2. Compute the sequence $S_i = {}^t u\, A^i\, v \in \mathbb{K}$ for $i = 0 \ldots 2\,N$
3. Return the minimal generating polynomial of the linear recursive sequence S (using Berlekamp-Massey / Padé approximants)

Applications :

- sparse linear system solving
- rank, determinant computation of sparse matrices

Algorithm - Block Wiedemann [Coppersmith '94]

Input: A sparse matrix $A \in \mathcal{M}_N(\mathbb{K})$
Output: Its minimal polynomial

1. Choose random $U, V \in \mathbb{K}^{N \times m}$
2. Compute the sequence $S_i = U^t A^i V \in \mathbb{K}^{m \times m}$ for $i = 0 \ldots 2N/m$**$+O(1)$?**
3. Compute the *left matrix generating polynomial* Π of the recursive sequence S

$$\forall j, \quad \sum_{i=0}^{d} \Pi_i S_{i+j} = 0^{m \times m} \quad \text{with } \Pi = \sum_{i=0}^{d} \Pi_i x^i$$

   (using Order basis / matrix-type Padé approximants)
4. Return the minimal polynomial of A (using the matrix generating polynomial)

Advantages of block Wiedemann algorithm :

- Better probability of success if $\mathbb{K}$ is a small field
- Enable parallelization of the algorithm (step 2)

Algorithm - Block Wiedemann [Coppersmith '94]

1. Choose random $U, V \in \mathbb{K}^{N \times m}$
2. Compute the sequence $S_i = {}^tU A^i V \in \mathbb{K}^{m \times m}$ for $i = 0 \ldots 2N/m$
3. Compute the *left matrix generating polynomial* of S using order basis
4. return the minimal polynomial of A

Problem :

The bound $2N/m$ is general but loose + Step 2 has dominant cost
$\rightsquigarrow$ Early termination strategy

Pros and cons of order basis algorithms ?

- for M-Basis
- for PM-Basis

Algorithm - Block Wiedemann using M-Basis with early termination
1. Choose random $U, V \in \mathbb{K}^{N \times m}$ 2. **for** $i = 0 \ldots 2N/m$ a. Update S from $[S_0, \ldots, S_{i-1}]$ to $[S_0, \ldots, S_i]$ b. Update order basis from order $i-1$ to order i using M-Basis c. **if** StopCriteria(S, order basis) **then** break 3. return the minimal polynomial of A

Will stop at $i = \sigma_{\mathsf{KV}} := \lceil \mu / m \rceil + \lceil \mu / n \rceil + O(1)$ - careful with formula : what is μ?

Pros and cons of block Wiedemann using M-Basis:

M-Basis
Early termination possible Minimal knowledge required on S
Slow Step 2.b (non negligible)

Algorithm - Block Wiedemann using PM-Basis with early termination

1. Choose random $U, V \in \mathbb{K}^{N \times m}$
2. **for** $\ell = 0 \ldots \lceil \log_2(2N/m) \rceil$ \\\\Assume $2N/m = 2^k$
   a. Update S from $[S_0, \ldots, S_{2^{\ell-1}-1}]$ to $[S_0, \ldots, S_{2^{\ell}-1}]$
   b. Update order basis from order $2^{\ell-1}$ to order 2^{ℓ} using PM-Basis
   c. **if** StopCriteria(S, order basis) **then** break
3. return the minimal polynomial of A

Pros and cons of block Wiedemann using PM-Basis:

PM-Basis
Restrictive early termination (recursive algo) May require more knowledge on S
Fast Step 2.b (negligible)

$\rightsquigarrow$ Let's start by an iterative PM-Basis

for better early termination

1. Order basis
   a. Definition
   b. Algorithms

2. Application to block Wiedemann algorithm

3. Contributions :
   a. Fast iterative order basis
   b. Fast online order basis
   c. Timings

First objective :

Transform recursive PM-Basis into iterative algorithm

Why ? Enable early termination

Simpler subproblem :

Algorithm RecursiveAlgo

1. **if** $\sigma = 1$ **then** Base Case
2. **else**
3. Recursive Call 1
4. Update input
5. Recursive Call 2
6. **return** Multiply answers

$\rightarrow$

Algorithm BinaryMultiplicationTree

1. **if** $\sigma = 1$ **then** Base Case
2. **else**
3. Recursive Call 1
4. Recursive Call 2
5. **return** Multiply answers

On the blackboard :

Binary multiplication trees and their iterative version ($\sigma = 2^k$)

Algorithm BMT

Input: $M = [M_0, \ldots, M_{2^k-1}]$
Ouput: $M_{2^k-1} \cdots M_0$

1. **if** $\#M = 1$ **then** return $M[0]$
2. **else**
3. $\quad P_l = \text{BMT}([M_0, \ldots, M_{2^{k-1}-1}])$
4. $\quad P_h = \text{BMT}([M_{2^{k-1}}, \ldots, M_{2^k-1}])$
5. **return** $P_h \cdot P_l$

$\rightarrow$

Algorithm iBMT

Input: $M = [M_0, \ldots, M_{2^k-1}]$
Ouput: $M_{2^k-1} \cdots M_0$

1. $P = []$
2. **for** $i = 0 \ldots 2^{k-1}$ **do**
3. $\quad$ Add M_k at the beginning of P
4. $\quad$ **for** $i = 1 \ldots \nu_2(k)$
5. $\quad\quad$ Merge $P[0]$, $P[1]$ by multiplication
6. **return** $P[0]$

Remarks : At step 2^k, we have the product. Otherwise, only chunks of the product. These chunks are enough for what we need.

Derecursivation of PM-Basis when $\sigma = 2^r$:

Algorithm PM-Basis

Input : $F \in \mathbb{K}[x]^{m \times n}, \sigma \in \mathbb{N}$
Output : A (F, σ) order basis P of F

1. **if** $\sigma = 1$ **then**
2. **return** Basis($F \bmod x$)
3. **else**
4. $P_{\text{low}} := \text{PM-Basis}(F, \lfloor \sigma/2 \rfloor)$ — First subproblem
5. $F' := \text{MiddleProduct}(P_{\text{low}}, F, \lfloor \sigma/2 \rfloor \ldots \sigma - 1)$ — Update problem
6. $P_{\text{high}} := \text{PM-Basis}(F', \lceil \sigma/2 \rceil)$ — Second subproblem
7. **return** $P_{\text{high}} \cdot P_{\text{low}}$ — Solve original problem

On the blackboard : Computation tree of PM-Basis for $\sigma = 4$

Step i : Computations until M_i and after Step $i - 1$

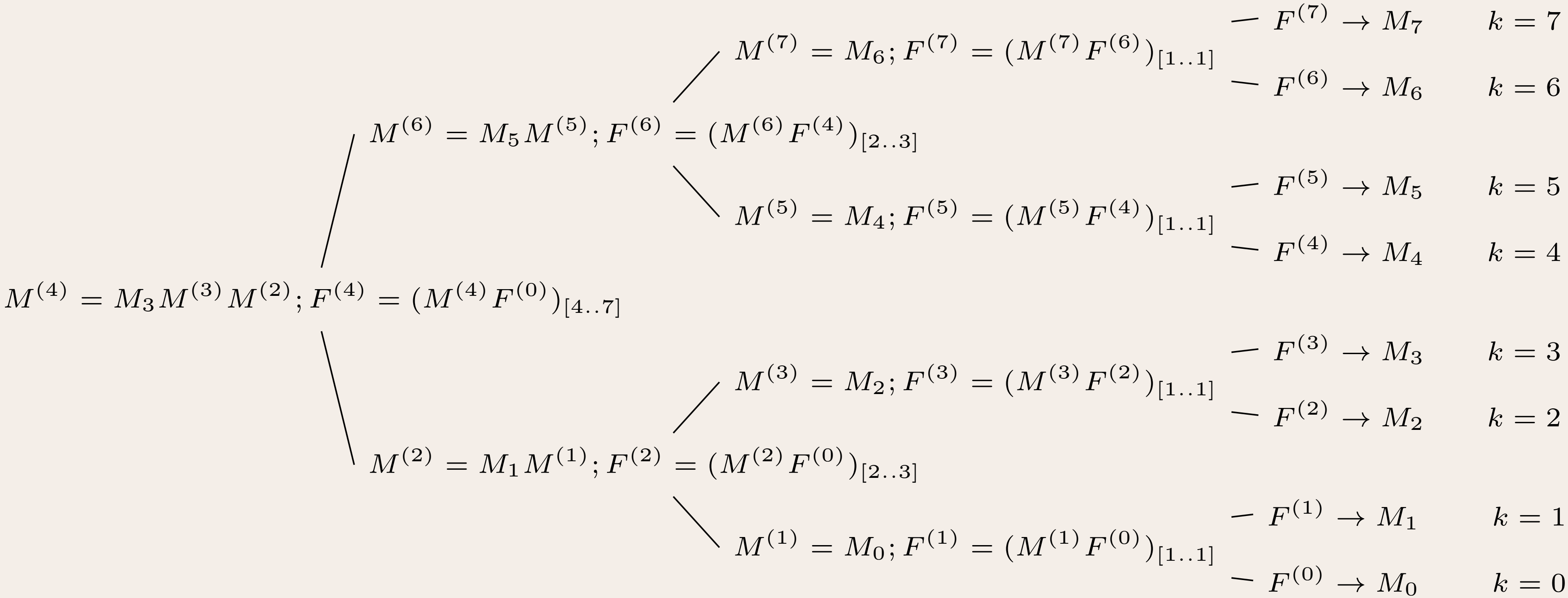


Figure. Computation tree of PM-Basis

Definition : If $\deg(A) \leqslant d$, then $\mathrm{MP}(A, B, d\ldots h) := (A\,B)_{d\ldots h}$.

Computations of PM-Basis(F, 2^k) :

1.	$M_0 = \text{Basis}(F_0)$	Step 0
2.	$F^{(1)} = \text{MP}(M_0, F, 1 \ldots 1)$	Step 1
3.	$M_1 = \text{Basis}\left(\left(F^{(1)}\right)_0\right)$	
4.	$M^{(2)} = M_1 \cdot M_0$	Step 2
5.	$F^{(2)} = \text{MP}\left(M^{(2)}, F, 2 \ldots 3\right)$	
6.	$M_2 = \text{Basis}\left(\left(F^{(2)}\right)_0\right)$	
7.	$F^{(3)} = \text{MP}\left(M_2, F^{(1)}, 1 \ldots 1\right)$	Step 3
8.	$M_3 = \text{Basis}\left(\left(F^{(3)}\right)_0\right)$	
9.	$M^{(4)} = M_3 \cdot M_2$	Step 4
10.	$M^{(4)} = M^{(4)} \cdot M^{(2)}$	
11.	$F^{(3)} = \text{MP}\left(M^{(4)}, F, 4 \ldots 7\right)$	
12.	$M_4 = \text{Basis}\left(\left(F^{(4)}\right)_0\right)$	

Algorithm iPM-Basis - Step *k*

1. $v = \nu_2(k)$
2. One step of iterative multiplication tree on M_i
3. $F^{(k)} = \text{MP}\left(M^{(k)}, F^{(k-2^v)}, 2^v \ldots 2^{v+1} - 1\right)$
4. $M_k = \text{Basis}\left(F^{(k)} \bmod x\right)$

Algorithm - Block Wiedemann using iPM-Basis with early termination

1. Choose random $U, V \in \mathbb{K}^{N \times m}$
2. **for** $\ell = 0 \ldots \lceil \log_2(2N/m) \rceil$ \\\\Assume $2N/m = 2^k$
   a. Update S from $[S_0, \ldots, S_{2^{\ell-1}-1}]$ to $[S_0, \ldots, S_{2^\ell - 1}]$
   b. **for** $i = 2^{\ell-1} \ldots 2^\ell - 1$
      i. Update order basis from order $i-1$ to order i using iPM-Basis
      ii. **if** StopCriteria(S, order basis) **then** break
3. return the minimal polynomial of A

Problem :

The bound $2N/m$ is general but loose + Step 2 has dominant cost

Pros and cons of order basis algorithms

M-Basis	PM-Basis	iPM-Basis
Early termination Minimal knowledge on S	No early termination More knowledge on S	Early termination More knowledge on S
Slow Step 3	Fast Step 3	Fast Step 3

At stake : We could gain a constant factor up to 2.

1. Order basis
   a. Definition
   b. Algorithms

2. Application to block Wiedemann algorithm

3. Contributions :
   a. Fast iterative order basis
   b. Fast online order basis
   c. Timings

Definition. (Online algorithm $\sim$ minimal knowledge on the input)

Let $F = \sum_{i \in \mathbb{N}} F_i x^i \in \mathbb{K}[[x]]^{m \times n}$.
An order basis algorithm is online if it reads at most $F_0, \ldots, F_k$ when computing M_k.

Example :

- M-Basis is online:

  Computation of M_k requires $(x^{-k} P_{k-1} F) \bmod x$, which involve only $F_0, \ldots, F_k$.

- iPM-Basis (and PM-Basis) are off-line

  Step 2^k: $F^{(2^k)} = \mathrm{MP}\Big(M^{(2^k)}, F, 2^k \ldots 2^{k+1} - 1\Big)$ requires $F_0, \ldots, F_{2^{k+1}-1}$!

Definition. (Online algorithm $\sim$ minimal knowledge on the input)

Let $F = \sum_{i \in \mathbb{N}} F_i x^i \in \mathbb{K}[[x]]^{m \times n}$.
An order basis algorithm is online *if it reads at most* $F_0, \ldots, F_k$ *when computing* M_k.

Goal : Turn iPM-Basis into an online algorithm

Algorithm iPM-Basis - step k
1. $v = \nu_2(k)$ 2. One step of iterative multiplication tree on M_i 3. Update the problem $F^{(k)} = \mathrm{MP}\left(M^{(k)}, F^{(k-2^v)}, 2^v \ldots 2^{v+1} - 1\right)$ 4. $M_k = \mathrm{Basis}\left(F^{(k)} \bmod x\right)$

On the blackboard :

- iPM-Basis middle products and how to make them online
- Necessity of an online middle product

Definition : If $\deg(A) \leqslant d$, then $\mathrm{MP}(A, B, d \ldots h) := (A\,B)_{d \ldots h}$.

Definition. *An middle product algorithm is* shifted online *if at each step it requires minimal knowledge on A and B.*

In practice, the ith coefficient of $\mathrm{MP}(A, B, d \ldots h)$ must use at most $A_0, \ldots, A_{d+i}$ and $B_0, \ldots, B_{d+i}$.

Example: $\mathrm{MP}(A, B, 3 \ldots 6) = \mathrm{MP}(A_{0 \ldots 3}, B_{0 \ldots 6}, 3 \ldots 6)$ on the blackboard

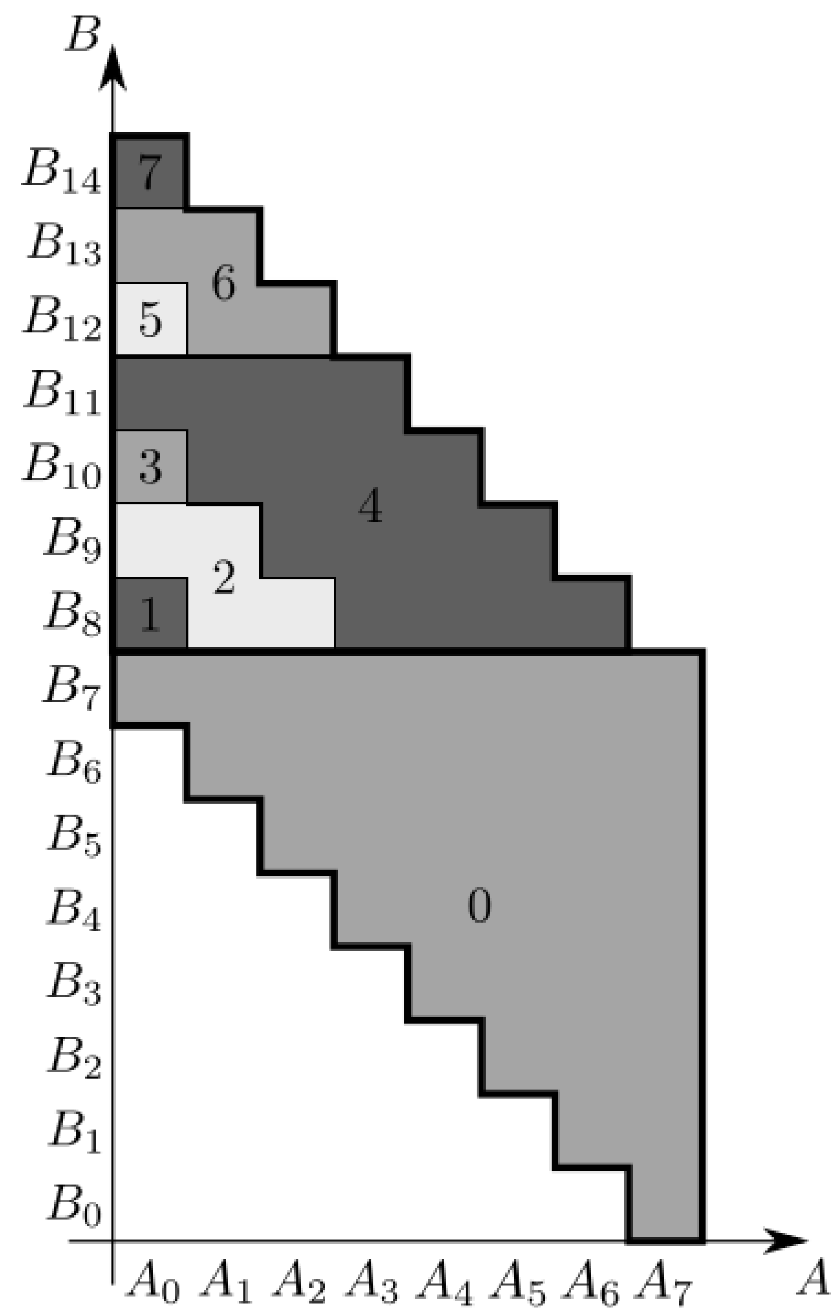
Shifted online middle product

Another example: $\mathrm{MP}(A_{0\ldots7}, B_{0\ldots14}, 7\ldots14)$

Shifted online algorithm :

Step i can read only $B_0, \ldots, B_{i+7}$

Step	Computation
0	$C \;= (A\,B_{0\ldots7})_{7\ldots14}$
1	$C \mathrel{+}= x\;\mathrm{MP}(A_0, B_8, 0)$
2	$C \mathrel{+}= x^2\,\mathrm{MP}(A_{0\ldots2}, B_{8\ldots9}, 1\ldots2)$
3	$C \mathrel{+}= x^3\,\mathrm{MP}(A_0, B_{10}, 0)$
4	$C \mathrel{+}= x^4\,\mathrm{MP}(A_{0\ldots6}, B_{8\ldots11}, 3\ldots6)$
5	$C \mathrel{+}= x^5\,\mathrm{MP}(A_0, B_{12}, 0)$
6	$C \mathrel{+}= x^6\,\mathrm{MP}(A_{0\ldots2}, B_{12\ldots13}, 1\ldots2)$
7	$C \mathrel{+}= x^7\,\mathrm{MP}(A_0, B_{14}, 0)$



Goal : Turn iPM-Basis into an online algorithm

Algorithm oPM-Basis - step k
1. $v = \nu_2(k)$ 2. One step of iterative multiplication tree on M_i 3. Compute one more term of previous online middle products 4. Compute first term of $F^{(k)} = \text{onlineMP}\left(M^{(k)}, F^{(k-2^v)}, 2^v \ldots 2^{v+1}-1\right)$ 5. $M_k = \text{Basis}\left(F^{(k)} \bmod x\right)$

Theorem. oPM-Basis *is an online order basis algorithm which is quasi-linear in the order* σ*.*

Algorithm - Block Wiedemann using oPM-Basis with early termination
1. Choose random $U, V \in \mathbb{K}^{N \times m}$
2. **for** $i = 0 \ldots 2N/m$
a. Update S from $[S_0, \ldots, S_{i-1}]$ to $[S_0, \ldots, S_i]$
b. Update order basis from order $i-1$ to order i using oPM-Basis
c. **if** StopCriteria(S, order basis) **then** break
3. return the minimal polynomial of A

Pros and cons of order basis algorithms in block Wiedemann :

M-Basis	PM-Basis	oPM-Basis
Early termination Minimal knowledge on S	No early termination More knowledge on S	Early termination Minimal knowledge on S
Slow Step 3	Fast Step 3	Fast Step 3

1. Order basis
    a. Definition
    b. Algorithms

2. Application to block Wiedemann algorithm

3. Contributions :
    a. Fast iterative order basis
    b. Fast online order basis
    c. Timings

Related work : Polynomial matrix multiplication

Tailored implementation for multi-core architectures

- Polynomial aspects : DFT cache friendly, SSE4 instructions
- Matrix aspects : use of BLAS, two data structures

m, d	MMX	FLINT 2.4	our code
16, 1024	**0.02 s**	0.26 s	0.03 s
16, 2048	**0.04 s**	0.70 s	0.06 s
16, 4096	**0.09 s**	1.68 s	0.13 s
16, 8192	**0.20 s**	4.52 s	0.28 s
128, 512	1.00 s	26.21 s	**0.82 s**
256, 256	4.00 s	36.71 s	**1.75 s**
512, 512	69.19 s	465.66 s	**19.64 s**
1024, 64	71.36 s	115.52 s	**13.95 s**
2048, 32	298.27 s	263.88 s	**48.90 s**

Table. Computation times of polynomial matrix multiplication in $\mathbb{F}_p[x]^{m\times m}$ with degree d and p a 23-bit FFT prime compared to Mathemagix (MMX) and FLINT.

Timings

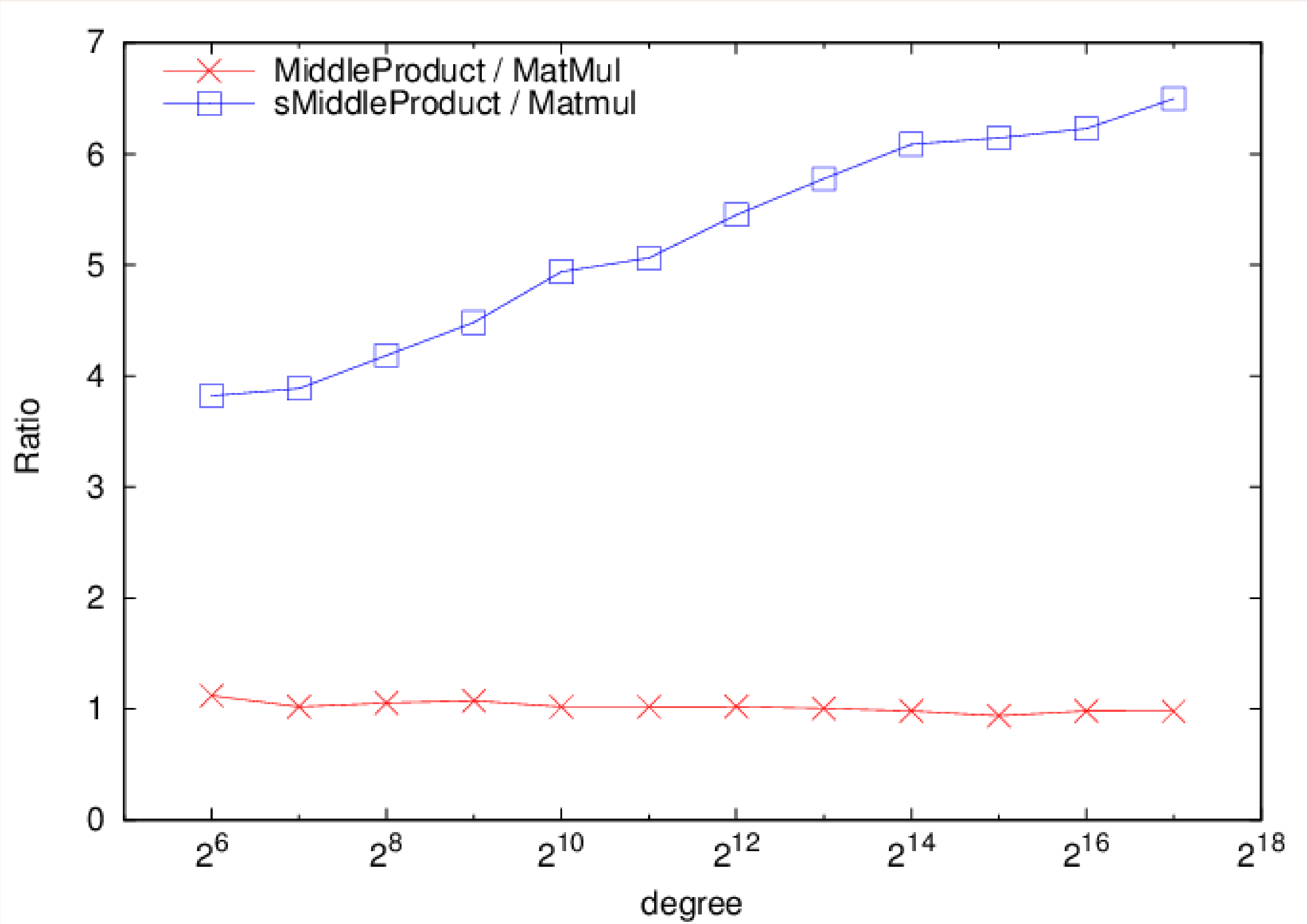


Figure. Relative performance of different middle products against polynomial matrix multiplication over $\mathbb{F}_p[x]^{16\times16}$ with p a 23-bit FFT prime

Timings

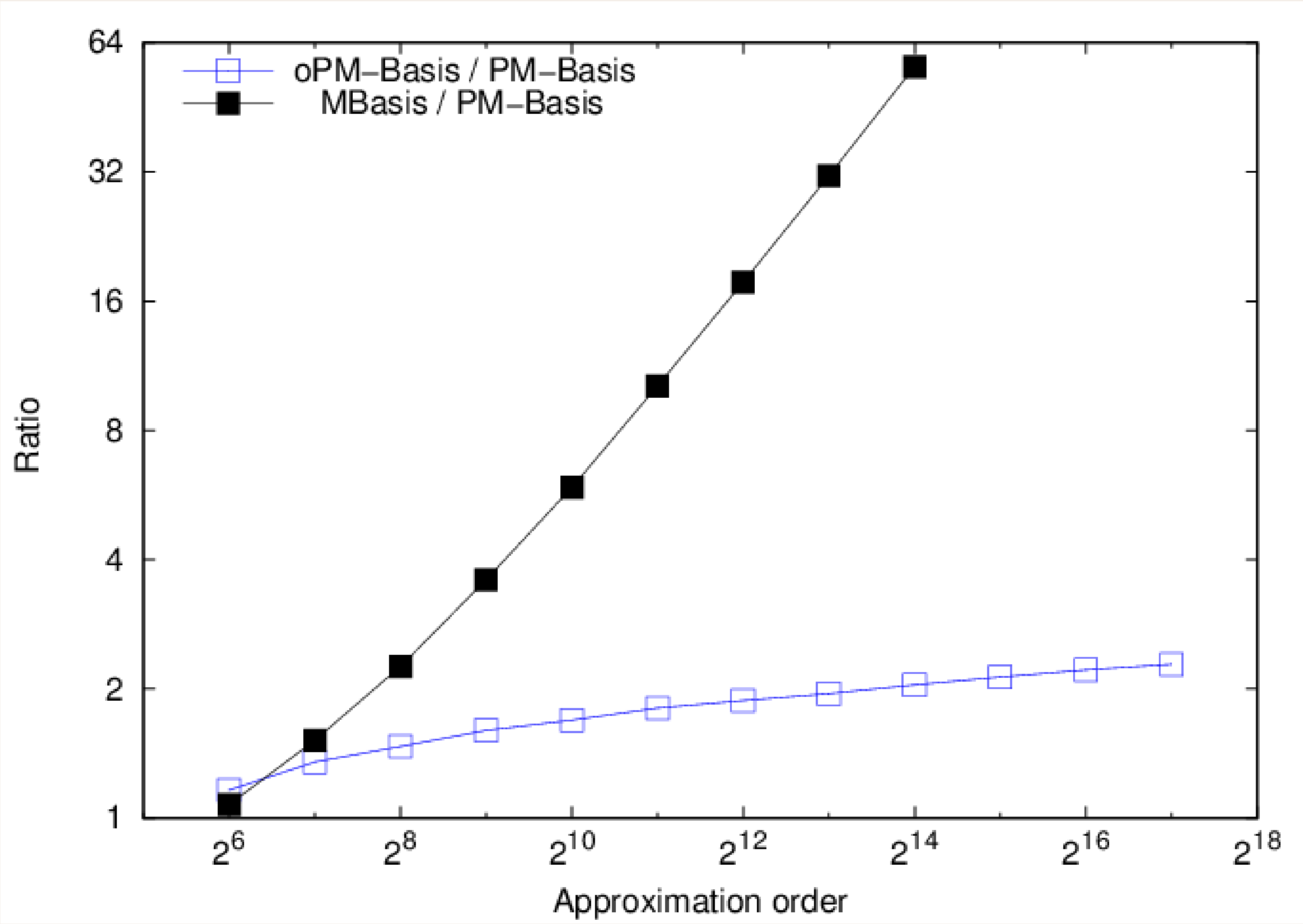


Figure. Relative performance of different middle products against polynomial matrix multiplication over $\mathbb{F}_p[x]^{16\times16}$ with p a 23-bit FFT prime

Setting : $A \in GL_{2^{17}}(\mathbb{F}_p)$ with 20 elements / row, $m = 16$, loose bound $\sigma = 2^{14}$

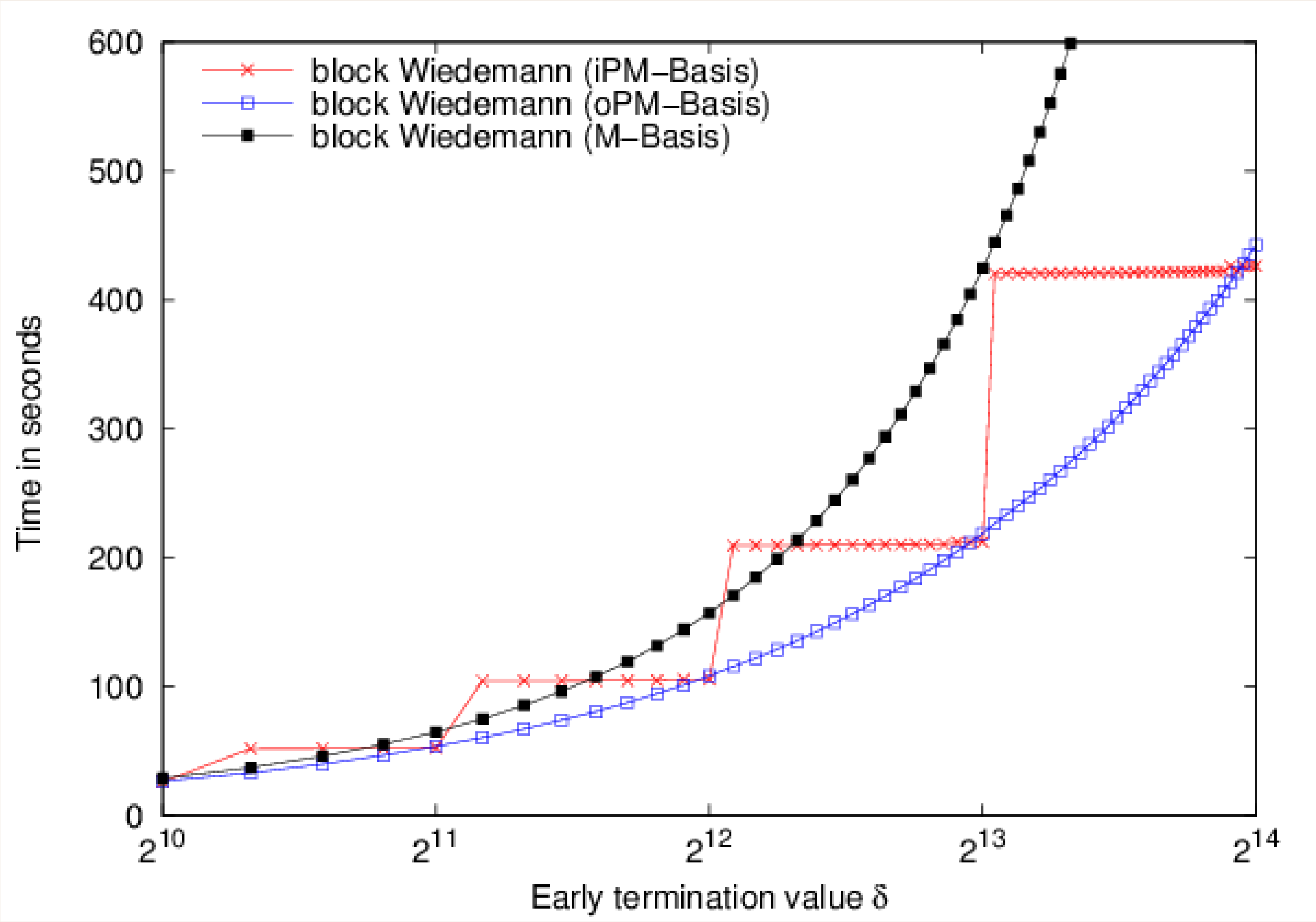


Figure. Computation times of early termination in block Wiedemann using order basis algorithm.