

# Extensions of Certain Graph-based Algorithms for Preconditioning

David Fritzsche

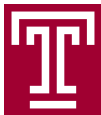
Temple  
University

Bergische Universität  
Wuppertal

with

Andreas Frommer and Daniel B. Szyld

Ninth Copper Mountain Conference on Iterative Methods  
April 2–7, 2006



# Overview

We consider a nonsingular linear system

$$Ax = b$$

with  $A \in \mathbb{R}^{n \times n}$ , sparse and nonsymmetric.

In this talk:

- ▶ We solve  $Ax = b$  iteratively (e.g. GMRES)
- ▶ block Jacobi / block Gauss-Seidel preconditioning
- ▶ **Main Topic:** An algorithm which finds (non-overlapping) sub-matrices, which are permuted to become diagonal blocks.

## Outline

1. Graph-based permutations
2. Preprocessing
3. Numerical experiments

# Basic Definitions

**Definition:** Let  $A \in \mathbb{R}^{n \times n}$ . Then  $G(A) = (V, E)$  is the *directed* graph of  $A$ , i.e.

- ▶ vertices  $V = \{1, \dots, n\}$
- ▶ edges  $E = \{(i, j) : a_{ij} \text{ is nonzero}\}$

**Definition:**

- ▶ Vertices  $i, j$  in  $G(A)$  are adjacent if  $a_{ij}$  **or**  $a_{ji}$  is nonzero.  
(direction of edges not important)
  - ▶  $i$  adjacent to  $W \subset V$  if  $i$  is adjacent to any  $j \in W$
- 

Finding dense “sub-matrices” in  $A$



Finding “well” connected “clusters” in  $G(A)$

# Overview of PABLO

PABLO (O'Neil, Szyld 1990) is a simple algorithm to find dense blocks.

## Basic Features

- ▶ Builds only one block at a time
- ▶ Considers only one vertex at a time for inclusion
- ▶ Checks all adjacent vertices for inclusion

## Other Features

- ▶ Time complexity:  $\mathcal{O}(n + nz(A))$ .  $\rightsquigarrow$  PABLO is fast
- ▶ Block sizes are not known a priori.
- ▶ The inclusion test can be varied  
 $\rightsquigarrow$  different versions (PABLO, TPABLO, XPABLO)

# Inclusion Test I: Structural Criteria

**General Situation:** Let  $B$  be the vertices currently in the block,  
 $i$  the eligible vertex (i.e.,  $i$  adjacent to some  $j \in B$ )

*We add  $i$  to  $B$  if the inclusion test  $\tau(i)$  is true*

Later: Definition of  $\tau$

Now: Basic criteria (predicates) used to define  $\tau$

## Fullness Criterion (FC)

The “fullness” of  $B \cup \{i\}$  is at least  $\alpha$  times the fullness of  $P$

$$\text{fullness} := \frac{\text{number of edges}}{\text{number of all possible edges}}$$

(typical values:  $\alpha \geq 1$ )

## Connectivity Criterion (CC)

A fraction of  $\beta$  or more of all edges of  $i$  goes into  $B$

(typical values:  $\beta \geq 0.5$ )

## Inclusion Test II: Threshold Criteria

**Observation:** So far only the structure of  $A$  was considered

**New:** Fix *threshold parameters*  $\delta$  and  $\gamma$ .

- ▶ Matrix entries  $< \delta$  are ignored  
(considered to be zero by PABLO)
- ▶ Matrix entries  $> \gamma$  are considered to be large.

Additionally to  $G = G(A)$  PABLO also looks at

$$G^\gamma := G \setminus \{\text{edges with represents matrix entries} \leq \gamma\}$$

### Threshold Fullness Criterion (TFC)

The fullness of  $B \cup \{i\}$  in  $G^\gamma$  is at least  $\theta$ .

### Threshold Connectivity Criterion (TCC)

A fraction of at least  $\zeta$  of the edges of  $i$  into  $B$  represent large entries.

## Inclusion Test III: Combining Criteria

The four basic criteria (FC, CC, TFC, TCC) are logically combined into the inclusion test  $\tau$ .

**PABLO** (O'Neil, Szyld 1990)

- ▶  $\tau = FC \vee CC$

**TPABLO** (Choi, Szyld 1996)

- ▶ TPABLO1:  $\tau = (FC \vee CC) \wedge TCC$

- ▶ TPABLO2:  $\tau = (FC \vee CC) \wedge TFC$  with  $\theta = 1$

**XPABLO** (Fritzsche 2004)

- ▶  $\tau = FC \vee TCC$

**Other Parameters**

- ▶ *minbs*: Minimum size of a block

- ▶ *maxbs*: Maximum size of a block

# PABLO based Preconditioners

**Idea:** Use the dense blocks found by PABLO for preconditioning.

- ▶  $M_J$  := Block Jacobi preconditioner
- ▶  $M_G$  := Block Gauss-Seidel preconditioner

## Advantages

- ▶  $M_J/M_G$  is “substantial” part of  $A$  if PABLO “worked”
- ▶ We can use dense BLAS/LAPACK subroutines

## Jacobi or Gauss-Seidel

- ▶ In a non-parallel setting Gauss-Seidel has the same cost as Jacobi, but gives better convergence
- ▶ Block Jacobi preconditioning can be done in parallel  
     $\rightsquigarrow$  may be better in some applications

# Preprocessing: Scaling and Transversals

We need to preprocess the matrix:

- ▶ PABLO does not change the diagonal values
- ▶ Without scaling it is very difficult to find good threshold parameters.

We use

**(MPS, MC64)** (Duff, Koster 2001)

MC64 finds a nonsymmetric permutation  $P$  and a scaling  $(R, C)$ ,  
s.t.

$$(PRAC)_{ii} = 1 \quad \text{for } i = 1, \dots, n$$

$$(PRAC)_{ij} \leq 1 \quad \text{for } i \neq j$$

# Choosing the Parameters

The best choice for the parameters depends on the system to solve and can be quite different from the recommendations given here.

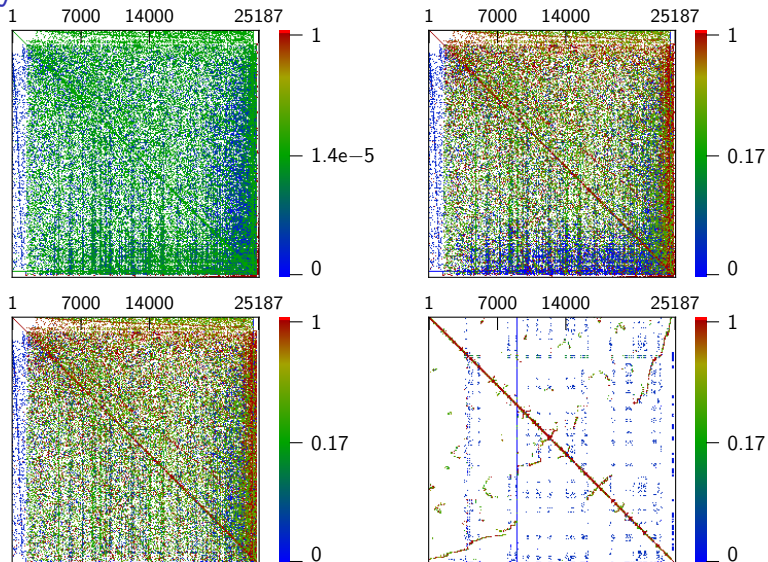
## XPABLO default parameters

- ▶  $\tau = (\text{FC}) \vee (\text{TCC})$
- ▶  $\alpha = 1.1$  and  $\zeta = 1/2n$
- ▶  $\text{minbs} = 10$  and  $\text{maxbs} = 50$
- ▶  $\delta = 0.05$  and  $\gamma = \sum_{i,j} |a_{ij}| / \text{nz}$

# Numerical Results I: Test Matrices

matrix	$n$	$nz$	application
CIRCUIT_4	80209	307604	circuit simulation
ECL32	51993	380415	semiconductor device simulation
HCIRCUIT	105676	513072	circuit design
IGBT3	10938	130500	semiconductor device simulation
MEMPLUS	17758	99147	circuit design
MULT_DCOP_01	25187	193276	circuit simulation
MULT_DCOP_02	25187	193276	circuit simulation
MULT_DCOP_03	25187	193216	circuit simulation
NMOS3	18588	237130	semiconductor device simulation
WANG3	26064	177168	semiconductor device simulation
ZHA01	33861	166453	electromagnetic
M_XC	35819	188564	circuit simulation
M_XCRCX	119193	593608	circuit simulation

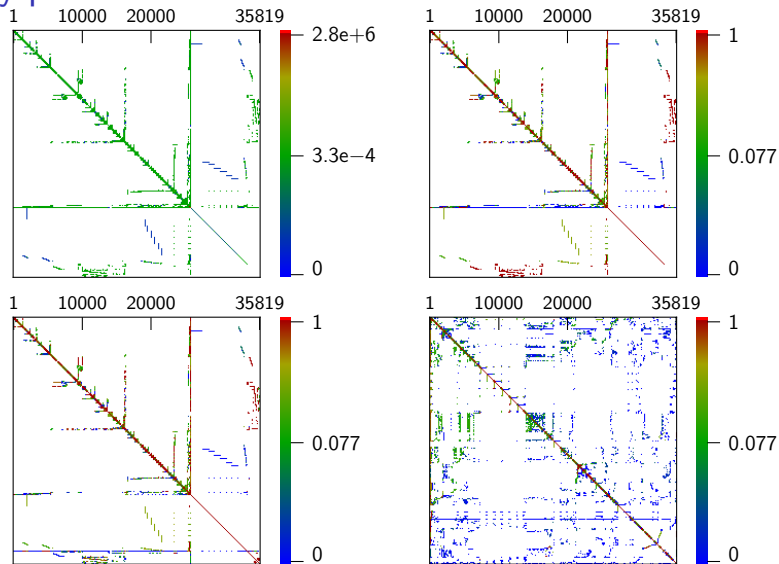
# Spy Plots: MULT\_DCOP\_01



Top left: original. Top right: scaled (and not yet permuted) by MC64.

Bottom left: after MC64. Bottom right: after MC64 and XPABLO

## Spy plots II: M\_XC



Top left: original. Top right: scaled (and not yet permuted) by MC64.

Bottom left: after MC64. Bottom right: after MC64 and XPABLO

# Measurements using GMRES(50)

Matrix	Times in seconds				Total Iterations			
	XPABLO		ILUT		XPABLO		ILUT	
	J	GS	I1	I2	J	GS	I1	I2
CIRCUIT_4	-	<b>2.1</b>	11	4.2	-	<b>36</b>	381	128
ECL32	-	14	2.6	<b>2</b>	-	408	131	<b>33</b>
HCIRCUIT	6.5	3.4	0.88	<b>0.57</b>	89	40	20	<b>8</b>
IGBT3	-	4	0.32	<b>0.21</b>	-	577	73	<b>23</b>
MEMPLUS	0.29	0.19	0.14	<b>0.1</b>	28	<b>12</b>	31	13
MULT_DCOP_01	0.74	<b>0.5</b>	0.74	0.76	41	21	7	<b>5</b>
MULT_DCOP_02	0.76	<b>0.45</b>	0.66	0.65	54	26	7	<b>5</b>
MULT_DCOP_03	0.29	<b>0.23</b>	0.65	0.64	15	7	7	<b>5</b>
NMOS3	4.8	2	0.35	<b>0.33</b>	450	150	34	<b>17</b>
WANG3	3.8	2	<b>0.43</b>	0.88	323	147	46	<b>23</b>
ZHA01	0.77	0.51	<b>0.2</b>	0.37	29	17	6	<b>4</b>
M_XC	-	<b>0.48</b>	0.6	0.49	-	<b>16</b>	41	21
M_XCRCX	-	<b>11</b>	19	11	-	<b>110</b>	310	150

## Preconditioners:

- J: MC64 + XPABLO + block Jacobi
- GS: MC64 + XPABLO + block Gauss-Seidel
- I1: MC64 + RCM + ILUT( $10^{-2}$ , 5)
- I2: MC64 + RCM + ILUT( $10^{-3}$ , 10)

# Conclusions

- ▶ not a black-box method
- ▶ competitive with ILUT when good parameters can be predicted
- ▶ less overhead than ILUT
- ▶ easy parallelization with block Jacobi preconditioning

## More Information

- ▶ <http://math.temple.edu/~daffi/pablo>
- ▶ Report coming soon