Efficient Computation of the Characteristic Polynomial

J-G. Dumas, C. Pernet, Z. Wan
{jgdumas,pernet}@imag.fr, wan@cis.udel.edu

ISSAC, 27th July 2005
Introduction

**Goal** Compute $\text{det}(\lambda I - A)$ over $\mathbb{Z}$
Introduction

Goal  Compute $\det(\lambda I - A)$ over $\mathbb{Z}$

Applications  Computational mathematics

- Matrix equivalence: via Frobenius normal form,
- Graph theory: cospectrality of graphs.
Introduction

Goal  Compute $\text{det}(\lambda I - A)$ over $\mathbb{Z}$

Applications  Computational mathematics

- Matrix equivalence: via Frobenius normal form,
- Graph theory: cospectrality of graphs.

Focus  on the design of algorithms

- Efficient in practice (time and memory)
- Mainly for dense matrices
- Probabilistic is enough if error probability $\epsilon \simeq 2^{-55}$
Outline

Efficient Computation of the Characteristic Polynomial

J-G. Dumas, C. Pernet, Z. Wan
Sommaire

Efficient Computation of the Characteristic Polynomial

J-G. Dumas, C. Pernet, Z. Wan
Global considerations

- Take benefit of the huge efforts for numerical computations: architecture, B.L.A.S.
  ⇒ cf. FFLAS & FFPACK
Global considerations

- Take benefit of the huge efforts for numerical computations: architecture, B.L.A.S.
  ⇒ cf. FFLAS & FFPACK
- Design of block algorithms to rely on matrix multiplication
Global considerations

- Take benefit of the huge efforts for numerical computations: architecture, B.L.A.S.
  ⇒ cf. FFLAS & FFPACK
- Design of block algorithms to rely on matrix multiplication
  Memory tuning: better data locality, cache optimizations
Global considerations

- Take benefit of the huge efforts for numerical computations: architecture, B.L.A.S.
  ⇒ cf. FFLAS & FFPACK
- Design of block algorithms to rely on matrix multiplication
  Memory tuning better data locality, cache optimizations
  Fast algorithms into practice $O(n^\omega)$ in theory but also proven useful in practice
<table>
<thead>
<tr>
<th>Sommaire</th>
</tr>
</thead>
<tbody>
<tr>
<td>Efficient Computation of the</td>
</tr>
<tr>
<td>Characteristic Polynomial</td>
</tr>
<tr>
<td>J-G. Dumas, C. Pernet, Z. Wan</td>
</tr>
</tbody>
</table>
Krylov’s method

**Definition**

\[ K = [v | Av | \ldots | A^n v] \]
Krylov’s method

**Definition**

\[ K = [v \mid Av \mid \ldots \mid A^d v \mid A^{d+1} v \mid \ldots \mid A^n v] \]
Krylov’s method

Definition

\[ K = [v | Av | \ldots | A^d v | A^{d+1} v | \ldots | A^n v] \]

\( X \)
Krylov’s method

Definition

\[ K = [v \mid Av \mid \ldots \mid A^d v \mid A^{d+1} v \mid \ldots \mid A^n v] \]

\[ \Rightarrow P_{\min}^{A,v}(X) = X^d - \sum_{i=1}^{d} y_i X^{i-1} \]
Krylov’s method

**Definition**

\[ K = \begin{bmatrix} v | Av | \ldots | A^d v | A^{d+1} v | \ldots | A^n v \end{bmatrix} \]

\[ \Rightarrow P_{A,v}^{A,v}(X) = X^d - \sum_{i=1}^{d} y_i X^{i-1} \]

**Fact**

\[ AX = XC_{P_{A,v}}^{A,v}_{\text{min}} \]

\[ C_{P_{A,v}}^{A,v}_{\text{min}} = \begin{pmatrix} 0 & m_0 \\ 1 & 0 & m_1 \\ \vdots & \vdots & \vdots \\ 1 & m_{d-1} \end{pmatrix} \]
Krylov’s method

Definition

\[ K = [v | Av | \ldots | A^d v | A^{d+1} v | \ldots | A^n v] \]

\[ \Rightarrow P_{min}^{A,v}(X) = X^d - \sum_{i=1}^{d} y_i X^{i-1} \]

Fact

\[ AX = XC_{P_{min}^{A,v}} \]

\[ C_{P_{min}^{A,v}} = \begin{pmatrix} 0 & m_0 \\ 1 & 0 & m_1 \\ \vdots & \vdots & \vdots \\ 1 & m_{d-1} \end{pmatrix} \]

If \( d = n \), \( \Rightarrow \) one gets \( P_{min}^{A,v} \) from \( C_{P_{min}^{A,v}} = X^{-1} AX \) [Krylov]
Krylov’s method

**Definition**

\[ K = [v|Av| \ldots |A^d v|A^{d+1} v| \ldots |A^n v] \]

\[ \Rightarrow P_{min}^{A,v}(X) = X^d - \sum_{i=1}^{d} y_i X^{i-1} \]

**Fact**

\[ AX = X C_{P_{min}^{A,v}} \]

\[ C_{P_{min}^{A,v}} = \begin{pmatrix} 0 & m_0 \\ 1 & 0 \\ \vdots & \vdots \\ 1 & m_{d-1} \end{pmatrix} \]

If \( d = n \) \( \Rightarrow \) one gets \( P_{min}^{A,v} \) from \( C_{P_{min}^{A,v}} = X^{-1} AX \) [Krylov]

If \( d < n \) complete \( X \) into \( \bar{X} \) invertible

\[ \Rightarrow \bar{X}^{-1} A \bar{X} = \begin{pmatrix} C_{P_{min}^{A,v}} & * \\ 0 & B \end{pmatrix} \]
Efficient Computation of the Characteristic Polynomial

J-G. Dumas, C. Pernet, Z. Wan
Using LUP factorization

Completion of $X$

Easier on the triangularized form (LUP):

$$P = X^T$$

$$B = A'_{22} - A'_{21} U_{1} - 1 U_{2}$$

where $A' = \begin{pmatrix} A'_{11} & A'_{12} \\ A'_{21} & A'_{22} \end{pmatrix} = P A_T P_T$.
Using LUP factorization

Completion of $X$

Easier on the triangularized form (LUP):

Compute

$$\begin{pmatrix} L_1 & 0 \\ L_2 & Id \end{pmatrix} \begin{pmatrix} U_1 & U_2 \\ 0 & 0 \end{pmatrix} P = X^T$$
Using LUP factorization

**Completion of \( X \)**

Easier on the triangularized form (LUP):

Compute \( \begin{pmatrix} L_1 & 0 \\ 0 & Id \end{pmatrix} \begin{pmatrix} U_1 & U_2 \\ 0 & Id \end{pmatrix} P = X^T \)
Using LUP factorization

Completion of $X$

Easier on the triangularized form (LUP):

Compute $\begin{pmatrix} L_1 & 0 \\ 0 & Id \end{pmatrix} \begin{pmatrix} U_1 & U_2 \\ 0 & Id \end{pmatrix} P = \overline{X}^T$

$\Rightarrow B = A'_{22} - A'_{21} U_1^{-1} U_2$

where $A' = \begin{pmatrix} A'_{11} & A'_{12} \\ A'_{21} & A'_{22} \end{pmatrix} = PA^T P^T.$
Using LUP factorization

**Completion of \( X \)**

Easier on the triangularized form (LUP):

Compute \[
\begin{pmatrix}
L_1 & 0 \\
0 & Id
\end{pmatrix}
\begin{pmatrix}
U_1 & U_2 \\
0 & Id
\end{pmatrix}
\] \( P = X^T \)

\[ B = A'_{22} - A'_{21} U_1^{-1} U_2 \]

where \( A' = \begin{pmatrix} A'_{11} & A'_{12} \\
A'_{21} & A'_{22} \end{pmatrix} = PA^T P^T. \)

**Minimal Polynomial** \( P_{\min}^{A,v}(X) = X^d - \sum_{i=1}^{d} y_i X^{i-1} \)

\[
X^T = \begin{bmatrix}
v^T \\
(Av)^T \\
(A^2 v)^T \\
\vdots \\
(A^d v)^T
\end{bmatrix}
= \begin{bmatrix}
L_{1,d} \\
L_{d+1}
\end{bmatrix}
. \begin{bmatrix}
U
\end{bmatrix}
. P
\]
Using LUP factorization

Completion of $X$

Easier on the triangularized form (LUP):

Compute $\begin{pmatrix} L_1 & 0 \\ 0 & Id \end{pmatrix} \begin{pmatrix} U_1 & U_2 \\ 0 & Id \end{pmatrix} P = \bar{X}^T$

$\Rightarrow B = A'_2 - A'_2 U_1^{-1} U_2$

where $A' = \begin{pmatrix} A'_{11} & A'_{12} \\ A'_{21} & A'_{22} \end{pmatrix} = PA^T P^T$.

Minimal Polynomial $P_{\text{min}}^A(X) = X^d - \sum_{i=1}^{d} y_i X^{i-1}$

$X^T = \begin{bmatrix} v^T \\ (Av)^T \\ (A^2v)^T \\ \vdots \\ (A^d v)^T \end{bmatrix} = \begin{bmatrix} L_{1,d} \\ L_{d+1} \end{bmatrix} \cdot U \cdot P$

$\Rightarrow y = L_{d+1} L_{1...d}^{-1}$ (in only $O(n^2)$ !)
LU-Krylov algorithm: \texttt{LUK}

\begin{align*}
\textbf{Require: } & \quad A \text{ a } n \times n \text{ matrix over a field} \\
\textbf{Ensure: } & \quad P_A^\text{char}(X) \text{ the characteristic polynomial of } A
\end{align*}
LU-Krylov algorithm: $\text{LUK}$

**Require:** $A$ a $n \times n$ matrix over a field

**Ensure:** $P^A_{\text{char}}(X)$ the characteristic polynomial of $A$

1: Pick a random vector $v$
2: Compute $X = [v | Av | A^2 v | \ldots | A^n v]$
LU-Krylov algorithm: \texttt{LUK}

**Require:** $A$ a $n \times n$ matrix over a field

**Ensure:** $P^A_{\text{char}}(X)$ the characteristic polynomial of $A$

1. Pick a random vector $\nu$
2. Compute $X = [\nu | A\nu | A^2\nu | \ldots | A^n\nu]$
3. Compute $(L, U, P) = \texttt{LUP}(X^T) (d = \text{rank}(X^T))$

Deterministic (although based on a probabilistic minpoly!)

Intensive use of matrix product...

... but only $2.666 \cdot n^3$ algebraic complexity
LU-Krylov algorithm: LUK

**Require:** $A$ a $n \times n$ matrix over a field

**Ensure:** $P^A_{\text{char}}(X)$ the characteristic polynomial of $A$

1. Pick a random vector $v$
2. Compute $X = [v | Av | A^2 v | \ldots | A^n v]$
3. Compute $(L, U, P) = \text{LUP}(X^T)$ ($d = \text{rank}(X^T)$)
4. Solve $y^T L_{1 \ldots d} = L_{d+1}$
5. Set $P^A_{\text{min}}(X) = X^d - \sum_{i=1}^{d} y_i X^{i-1}$
LU-Krylov algorithm: **LUK**

**Require:** \( A \) a \( n \times n \) matrix over a field

**Ensure:** \( P_A^{\text{char}}(X) \) the characteristic polynomial of \( A \)

1: Pick a random vector \( v \)

2: Compute \( X = [v|Av|A^2v|\ldots|A^nv] \)

3: Compute \( (L, U, P) = \text{LUK}(X^T) (d = \text{rank}(X^T)) \)

4: Solve \( y^TL_1\ldots d = L_{d+1} \)

5: Set \( P_{\text{min}}^{A,v}(X) = X^d - \sum_{i=1}^d y_i X^{i-1} \)

6: **if** \( d = n \) **then**

7: return \( P_A^{\text{char}} = P_{\text{min}}^{A,v} \)

8: **else**

11: **end if**
LU-Krylov algorithm: \( \text{LUK} \)

**Require:** \( A \) a \( n \times n \) matrix over a field

**Ensure:** \( P_{\text{char}}^A(X) \) the characteristic polynomial of \( A \)

1: Pick a random vector \( v \)
2: Compute \( X = [v|Av|A^2v|\ldots|A^n v] \)
3: Compute \((L, U, P) = \text{LUP}(X^T) \) \((d = \text{rank}(X^T))\)
4: Solve \( y^T L_{1\ldots d} = L_{d+1} \)
5: Set \( P_{\text{min}}^A(X) = X^d - \sum_{i=1}^d y_i X^{i-1} \)
6: **if** \( (d = n) \) **then**
7: return \( P_{\text{char}}^A = P_{\text{min}}^A \)
8: **else**
9: \( A' = PA^T P^T; B = A'_{22} - A'_{21} S_1^{-1} S_2 \)
11: **end if**
LU-Krylov algorithm: \( \text{LUK} \)

**Require:** \( A \) a \( n \times n \) matrix over a field

**Ensure:** \( P^A_{\text{char}}(X) \) the characteristic polynomial of \( A \)

1: Pick a random vector \( v \)
2: Compute \( X = [v|Av|A^2v|\ldots|A^n v] \)
3: Compute \( (L, U, P) = \text{LUP}(X^T) \ (d = \text{rank}(X^T)) \)
4: Solve \( y^T L_{1\ldots d} = L_{d+1} \)
5: Set \( P^A_{\text{min}}(X) = X^d - \sum_{i=1}^{d} y_i X^{i-1} \)
6: **if** \( (d = n) \) **then**
7: \( \text{return } P^A_{\text{char}} = P^A_{\text{min}} \)
8: **else**
9: \( A' = PA^T P^T; \ B = A'_{22} - A'_{21} S_1^{-1} S_2 \)
10: \( \text{return } P^A_{\text{char}}(X) = P^A_{\text{min}}(X) \times \text{LUK}(B) \)
11: **end if**
LU-Krylov algorithm: \texttt{LUK}

\begin{algorithm}
\textbf{Require:} $A$ a $n \times n$ matrix over a field
\textbf{Ensure:} $P_{\text{char}}^A(X)$ the characteristic polynomial of $A$

1: Pick a random vector $v$
2: Compute $X = [v | Av | A^2 v | \ldots | A^n v]$
3: Compute $(L, U, P) = \text{LUP}(X^T) \ (d = \text{rank}(X^T))$
4: Solve $y^T L_{1 \ldots d} = L_{d+1}$
5: Set $P_{\text{min}}^{A,v}(X) = X^d - \sum_{i=1}^{d} y_i X^{i-1}$
6: \textbf{if} $(d = n)$ \textbf{then}
7: \quad return $P_{\text{char}}^A = P_{\text{min}}^{A,v}$
8: \textbf{else}
9: \quad $A' = PA^T P^T; B = A'_{22} - A'_{21} S_1^{-1} S_2$
10: \quad return $P_{\text{char}}^A(X) = P_{\text{min}}^{A,v}(X) \times \text{LUK}(B)$
11: \textbf{end if}
\end{algorithm}

- Deterministic (although based on a probabilistic \texttt{minpoly} !)
- Intensive use of matrix product...

... but only $2.666n^3$ algebraic complexity
Efficient Computation of the Characteristic Polynomial

J-G. Dumas, C. Pernet, Z. Wan
Comparison with the branching algorithm

The branching algorithm

- Also based on Krylov iterates and elimination
- Handles every blocks at once with matrix product
- The best in theory: $O(n^\omega \log(n)) = O(n^\omega \log(k_{\text{max}}))$
Comparison with the branching algorithm

The branching algorithm

- Also based on Krylov iterates and elimination
- Handles every blocks at once with matrix product
- The best in theory: $O(n^\omega \log(n))$ ($= O(n^\omega \log(k_{\text{max}}))$)

Comparison with different number of blocks
- the $\log$ factor and the constant penalize the gain of grouping operations into matrix product
Comparison with the fast algorithm

The Fast algorithm

- Only valid for generic matrices
- Optimal complexity $T = O(n^\omega)$
- Constant 2.794 with $\omega = 3$ close to the 2.666 of LUK

- slowness for small matrices
- faster for large matrices (only matrix products)

⇒ advocates for a generalization or hybrid algorithm.
Comparison with the fast algorithm

The Fast algorithm
- Only valid for generic matrices
- Optimal complexity $T = O(n^\omega)$
- Constant 2.794 with $\omega = 3$ close to the 2.666 of LUK

- slower for small matrices (constant)
- faster for large matrices (only matrix products)
Comparison with the fast algorithm

The Fast algorithm

- Only valid for generic matrices
- Optimal complexity $T = O(n^\omega)$
- Constant 2.794 with $\omega = 3$ close to the 2.666 of LUK

- Slower for small matrices (constant)
- Faster for large matrices (only matrix products)

⇒ advocates for a generalization or hybrid algorithm.
Sommaire

Efficient Computation of the Characteristic Polynomial
J-G. Dumas, C. Pernet, Z. Wan
Generalities

Several approaches:

Ring operations

- without divisions \([Berkowitz84, Kaltofen92]\)
- with exact divisions \([Abdeljaoued-Malaschonock01]\)
Generalities

Several approaches:

Ring operations

- without divisions [Berkowitz84, Kaltofen92]
- with exact divisions [Abdeljaoued-Malaschonock01]

Finite fields and chinese remaindering: Folklore
Generalities

Several approaches:

Ring operations

- without divisions [Berkowitz84, Kaltofen92]
- with exact divisions [Abdeljaoued-Malaschonock01]

Finite fields and chinese remaindering: Folklore

Lifting and gcd free basis: [Storjohann00]
Generalities

Several approaches:

**Ring operations**

- without divisions \([\text{Berkowitz84}, \text{Kaltofen92}]\)
- with exact divisions \([\text{Abdeljaoued-Malaschonock01}]\)

**Finite fields and chinese remaindering** :  \(\text{Folklore}\)

**Lifting and gcd free basis** :  \([\text{Storjohann00}]\)

**Combination Block-Wiedemann+BSGS** \([\text{Kaltofen-Villard04}]\)
Efficient Computation of the Characteristic Polynomial

J-G. Dumas, C. Pernet, Z. Wan
Bound on the coefficients

Chinese Remainder Algorithm

Principle: Several computations modulo random word size primes $p_i$

Correctness: If $\beta$ is bounds the result: correctness if $\prod_i p_i \leq \beta$
Bound on the coefficients

### Chinese Remainder Algorithm

**Principle**  Several computations modulo random word size primes $p_i$

**Correctness**  If $\beta$ is bounds the result: correctness if $\prod_i p_i \leq \beta$

- for $\text{det}$: use Hadamard's bound
  $$\log(|d|) \leq \frac{n}{2} (\log(n) + \log(\|A\|^2))$$
Bound on the coefficients

Chinese Remainder Algorithm

Principle Several computations modulo random word size primes $p_i$

Correctness If $\beta$ is bounds the result: correctness if $\prod_i p_i \leq \beta$

- for $\det$ : use Hadamard’s bound
  $\Rightarrow \log(|d|) \leq \frac{n}{2}(\log(n) + \log(\|A\|^2))$

- for $\text{charpoly}$:
  $\beta = \max_{i=0..\sqrt{1+4en-1}/2e} \left( \binom{n}{i} \sqrt{(n-i) \log(\|A\|^2)^{n-i}} \right)$
  $\Rightarrow \log(|c_i|) \leq \frac{n}{2}(\log(n) + \log(\|A\|^2) + 0.21163175)$
Example

\[
A = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & -1 & -1 & -1 & -1 \\
1 & -1 & 1 & -1 & -1 & -1 \\
1 & -1 & -1 & 1 & -1 & -1 \\
1 & -1 & -1 & -1 & 1 & 1
\end{bmatrix}
\]

\[
\Rightarrow P_A^{\text{char}}(X) = X^5 - 5X^4 + 40X^2 - 80X + 48
\]
Example

\[ A = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 \\
1 & 1 & -1 & -1 & -1 \\
1 & -1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & -1 \\
1 & -1 & -1 & -1 & 1
\end{bmatrix} \]

\[ \Rightarrow P_{\text{char}}^A(X) = X^5 - 5X^4 + 40X^2 - 80X + 48 \]

\[ \max_i(|c_i|) = 80 = \binom{5}{1} \sqrt{4^4} \]
Example

\[
A = \begin{pmatrix}
1 & 1 & 1 & 1 & 1 \\
1 & 1 & -1 & -1 & -1 \\
1 & -1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & -1 \\
1 & -1 & -1 & -1 & 1 \\
\end{pmatrix}
\]

\[P_{\text{char}}^A(X) = X^5 - 5X^4 + 40X^2 - 80X + 48\]

- \(\max_i(|c_i|) = 80 = \binom{5}{1} \sqrt{4^4}\)
- Hadamard’s bound : 55.9
Example

\[ A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & -1 \\ 1 & -1 & -1 & -1 & 1 \end{bmatrix} \]

\[ \Rightarrow P_{\text{char}}^A(X) = X^5 - 5X^4 + 40X^2 - 80X + 48 \]

- \( \max_i(|c_i|) = 80 = \binom{5}{1} \sqrt{4}^4 \)
- Hadamard’s bound : 55.9
- bound in [Giesbrecht-Storjohann02] : 21792.7
Example

\[ A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & -1 \\ 1 & -1 & -1 & -1 & 1 \end{bmatrix} \]

\[ P^A_{\text{char}}(X) = X^5 - 5X^4 + 40X^2 - 80X + 48 \]

- \( \max_i(|c_i|) = 80 = \binom{5}{1} \sqrt{4^4} \)
- Hadamard’s bound : 55.9
- bound in [Giesbrecht-Storjohann02] : 21792.7
- Our bound : 80.66661.
Sommaire
Early termination

- Stop as soon as the reconstructed polynomial remains unchanged
- Probability of failure $< 1/p$ (rough majoration)
Early termination

- Stop as soon as the reconstructed polynomial remains unchanged
- Probability of failure $< 1/p$ (rough majoration)

Improvement:

Algorithm CIA

- CRA on the minimal polynomial (stops earlier)
- Recovery of the characteristic polynomial by:
  - Factorization of $P_{\min}^A$ over $\mathbb{Z}$ via Hensel Lifting
  - One computation of the characteristic polynomial mod $p$
  - Recovery of the multiplicities by divisions in $\mathbb{Z}_p[X]$
Properties of \texttt{CIA}

**Properties**

- Las Vegas if \texttt{minpoly} is deterministic
  \[ \Rightarrow \text{test if } \sum_i \alpha_i d_i = n \]
Properties of CIA

- Las Vegas if $\text{minpoly}$ is deterministic
  \[ \Rightarrow \text{test if } \sum_i \alpha_i d_i = n \]

- Otherwise MonteCarlo with many failure detections
  \[ \Rightarrow \text{test if } \alpha_i = 0 \text{ implies } P_{\text{min}}^A \nmid P_{\text{char}}^A \mod p \]
  \[ \Rightarrow \text{test if } \text{Trace}(A) = a_{n-1} \text{ implies } P_{\text{min}}^A \nmid P_{\text{char}}^A \]
Properties of CIA

Properties

- Las Vegas if minpoly is deterministic
  \[ \Rightarrow \text{test if } \sum_i \alpha_i d_i = n \]

- Otherwise MonteCarlo with many failure detections
  \[ \Rightarrow \text{test if } \alpha_i = 0 \text{ implies } p^A_{\min} \nmid p^A_{\text{char}} \pmod{p} \]
  \[ \Rightarrow \text{test if } \text{Trace}(A) = a_{n-1} \text{ implies } p^A_{\min} \nmid p^A_{\text{char}} \]

- Also adapted to sparse computations (using Wiedemann minpoly), although it requires one dense modular computation
Efficient Computation of the Characteristic Polynomial

J-G. Dumas, C. Pernet, Z. Wan
Experiments with random dense matrices

On an athlon-1.8Ghz with 2Gb of RAM

<table>
<thead>
<tr>
<th>n</th>
<th>100</th>
<th>200</th>
<th>400</th>
<th>800</th>
<th>1200</th>
<th>1500</th>
<th>3000</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>400</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>800</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1200</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1500</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Experiments with random dense matrices

On an athlon-1.8Ghz with 2Gb of RAM

<table>
<thead>
<tr>
<th>n</th>
<th>Maple</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>163s</td>
</tr>
<tr>
<td>200</td>
<td>3355s</td>
</tr>
<tr>
<td>400</td>
<td>74970s</td>
</tr>
<tr>
<td>800</td>
<td></td>
</tr>
<tr>
<td>1200</td>
<td></td>
</tr>
<tr>
<td>1500</td>
<td></td>
</tr>
<tr>
<td>3000</td>
<td></td>
</tr>
</tbody>
</table>
### Experiments with random dense matrices

On an athlon-1.8Ghz with 2Gb of RAM

<table>
<thead>
<tr>
<th>$n$</th>
<th>Maple</th>
<th>Magma</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>163s</td>
<td>0.34s</td>
</tr>
<tr>
<td>200</td>
<td>3355s</td>
<td>4.45s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11.1Mb</td>
</tr>
<tr>
<td>400</td>
<td>74970s</td>
<td>69.8s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>56Mb</td>
</tr>
<tr>
<td>800</td>
<td>1546s</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>403Mb</td>
</tr>
<tr>
<td>1200</td>
<td>8851s</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1368Mb</td>
</tr>
<tr>
<td>1500</td>
<td>MT</td>
<td></td>
</tr>
<tr>
<td>3000</td>
<td>MT</td>
<td></td>
</tr>
</tbody>
</table>
**Experiments with random dense matrices**

On an athlon-1.8Ghz with 2Gb of RAM

<table>
<thead>
<tr>
<th>$n$</th>
<th>Maple</th>
<th>Magma</th>
<th>ILUK-det</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>163s</td>
<td>0.34s</td>
<td>0.23s</td>
</tr>
<tr>
<td>200</td>
<td>3355s</td>
<td>4.45s</td>
<td>3.95s</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>11.1Mb</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.5Mb</td>
</tr>
<tr>
<td>400</td>
<td>74970s</td>
<td>69.8s</td>
<td>91.4s</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>56Mb</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10.1Mb</td>
</tr>
<tr>
<td>800</td>
<td>1546s</td>
<td>1409s</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>403Mb</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>36.3Mb</td>
</tr>
<tr>
<td>1200</td>
<td>8851s</td>
<td>7565s</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1368Mb</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>81Mb</td>
</tr>
<tr>
<td>1500</td>
<td>MT</td>
<td>21010s</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>136Mb</td>
</tr>
<tr>
<td>3000</td>
<td>MT</td>
<td>349494s</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>521Mb</td>
</tr>
</tbody>
</table>
Experiments with random dense matrices

On an athlon-1.8Ghz with 2Gb of RAM

<table>
<thead>
<tr>
<th>$n$</th>
<th>Maple</th>
<th>Magma</th>
<th>ILUK-det</th>
<th>CIA</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>163s</td>
<td>0.34s</td>
<td>0.23s</td>
<td>0.20s</td>
</tr>
<tr>
<td>200</td>
<td>3355s</td>
<td>4.45s</td>
<td>3.95s</td>
<td>3.25s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11.1Mb</td>
<td>3.5Mb</td>
<td>3.5Mb</td>
</tr>
<tr>
<td>400</td>
<td>74970s</td>
<td>69.8s</td>
<td>91.4s</td>
<td>71.74s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>56Mb</td>
<td>10.1Mb</td>
<td>10.1Mb</td>
</tr>
<tr>
<td>800</td>
<td>1546s</td>
<td>1409s</td>
<td>1110s</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>403Mb</td>
<td>36.3Mb</td>
<td>36.3Mb</td>
</tr>
<tr>
<td>1200</td>
<td>8851s</td>
<td>7565s</td>
<td>5999s</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1368Mb</td>
<td>81Mb</td>
<td>81Mb</td>
</tr>
<tr>
<td>1500</td>
<td>MT</td>
<td>21010s</td>
<td>16705s</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>136Mb</td>
<td>136Mb</td>
<td></td>
</tr>
<tr>
<td>3000</td>
<td>MT</td>
<td>349494s</td>
<td>286466s</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>521Mb</td>
<td>521Mb</td>
<td></td>
</tr>
</tbody>
</table>
Experiments with structured matrices

On a athlon-1.8Ghz with 2Gb of RAM.

| Matrix |  
|-------|---
| $n$   |   
| $d$   |   
| $\omega$ |   

<table>
<thead>
<tr>
<th>Method</th>
<th>Computation time in seconds.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magma-prob</td>
<td>1.14</td>
</tr>
<tr>
<td>Magma-det</td>
<td>1.31</td>
</tr>
<tr>
<td>ILUK-det</td>
<td>1.1</td>
</tr>
<tr>
<td>CIA-sparse</td>
<td>0.32</td>
</tr>
<tr>
<td>CIA-dense</td>
<td>1.22</td>
</tr>
</tbody>
</table>

Efficient Computation of the Characteristic Polynomial

J-G. Dumas, C. Pernet, Z. Wan
Experiments with structured matrices

On a athlon-1.8Ghz with 2Gb of RAM.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>300</td>
</tr>
<tr>
<td>$d$</td>
<td>75</td>
</tr>
<tr>
<td>$\omega$</td>
<td>1.9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Computation time in seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magma-prob</td>
<td>1.14</td>
</tr>
<tr>
<td>Magma-det</td>
<td>1.31</td>
</tr>
<tr>
<td>ILUK-det</td>
<td>1.1</td>
</tr>
<tr>
<td>CIA-sparse</td>
<td>0.32</td>
</tr>
<tr>
<td>CIA-dense</td>
<td>1.22</td>
</tr>
</tbody>
</table>
Experiments with structured matrices

On a athlon-1.8Ghz with 2Gb of RAM.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>A</th>
<th>$U^{-1}AU$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>$d$</td>
<td>75</td>
<td>75</td>
</tr>
<tr>
<td>$\omega$</td>
<td>1.9</td>
<td>300</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magma-prob</td>
<td>1.14</td>
</tr>
<tr>
<td>Magma-det</td>
<td>1.31</td>
</tr>
<tr>
<td>ILUK-det</td>
<td>1.1</td>
</tr>
<tr>
<td>CIA-sparse</td>
<td>0.32</td>
</tr>
<tr>
<td>CIA-dense</td>
<td>1.22</td>
</tr>
</tbody>
</table>

Computation time in seconds.
Experiments with structured matrices

On a athlon-1.8Ghz with 2Gb of RAM.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>$A$</th>
<th>$U^{-1}AU$</th>
<th>$A^TA$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>300</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>$d$</td>
<td>75</td>
<td>75</td>
<td>21</td>
</tr>
<tr>
<td>$\omega$</td>
<td>1.9</td>
<td>300</td>
<td>2.95</td>
</tr>
<tr>
<td>Magma-prob</td>
<td>1.14</td>
<td>7.11</td>
<td>0.23</td>
</tr>
<tr>
<td>Magma-det</td>
<td>1.31</td>
<td>10.55</td>
<td>0.24</td>
</tr>
<tr>
<td>ILUK-det</td>
<td>1.1</td>
<td>93.3</td>
<td>64.87</td>
</tr>
<tr>
<td>CIA-sparse</td>
<td>0.32</td>
<td>4.32</td>
<td>0.81</td>
</tr>
<tr>
<td>CIA-dense</td>
<td>1.22</td>
<td>1.3</td>
<td>0.87</td>
</tr>
</tbody>
</table>

Computation time in seconds.
Experiments with structured matrices

On a athlon-1.8Ghz with 2Gb of RAM.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>$A$</th>
<th>$U^{-1}AU$</th>
<th>$A' A$</th>
<th>$B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>300</td>
<td>300</td>
<td>300</td>
<td>600</td>
</tr>
<tr>
<td>$d$</td>
<td>75</td>
<td>75</td>
<td>21</td>
<td>424</td>
</tr>
<tr>
<td>$\omega$</td>
<td>1.9</td>
<td>300</td>
<td>2.95</td>
<td>4</td>
</tr>
<tr>
<td>Magma-prob</td>
<td>1.14</td>
<td>7.11</td>
<td>0.23</td>
<td>6.4</td>
</tr>
<tr>
<td>Magma-det</td>
<td>1.31</td>
<td>10.55</td>
<td>0.24</td>
<td>6.4</td>
</tr>
<tr>
<td>ILUK-det</td>
<td>1.1</td>
<td>93.3</td>
<td>64.87</td>
<td>68.4</td>
</tr>
<tr>
<td>CIA-sparse</td>
<td>0.32</td>
<td>4.32</td>
<td>0.81</td>
<td>4.4</td>
</tr>
<tr>
<td>CIA-dense</td>
<td>1.22</td>
<td>1.3</td>
<td>0.87</td>
<td>38.9</td>
</tr>
</tbody>
</table>

Computation time in seconds.
Experiments with structured matrices

On a athlon-1.8Ghz with 2Gb of RAM.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>$A$</th>
<th>$U^{-1}AU$</th>
<th>$A^T A$</th>
<th>$B$</th>
<th>$U^{-1}BU$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>300</td>
<td>300</td>
<td>300</td>
<td>600</td>
<td>600</td>
</tr>
<tr>
<td>$d$</td>
<td>75</td>
<td>75</td>
<td>21</td>
<td>424</td>
<td>424</td>
</tr>
<tr>
<td>$\omega$</td>
<td>1.9</td>
<td>300</td>
<td>2.95</td>
<td>4</td>
<td>600</td>
</tr>
<tr>
<td>Magma-prob</td>
<td>1.14</td>
<td>7.11</td>
<td>0.23</td>
<td>6.4</td>
<td>184.7</td>
</tr>
<tr>
<td>Magma-det</td>
<td>1.31</td>
<td>10.55</td>
<td>0.24</td>
<td>6.4</td>
<td>185</td>
</tr>
<tr>
<td>ILUK-det</td>
<td>1.1</td>
<td>93.3</td>
<td>64.87</td>
<td>68.4</td>
<td>2305</td>
</tr>
<tr>
<td>CIA-sparse</td>
<td>0.32</td>
<td>4.32</td>
<td>0.81</td>
<td>4.4</td>
<td>352.6</td>
</tr>
<tr>
<td>CIA-dense</td>
<td>1.22</td>
<td>1.3</td>
<td>0.87</td>
<td>38.9</td>
<td>42.6</td>
</tr>
</tbody>
</table>

Computation time in seconds.
Experiments with structured matrices

On an Athlon-1.8Ghz with 2Gb of RAM.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>$A$</th>
<th>$U^{-1}AU$</th>
<th>$A^TA$</th>
<th>$B$</th>
<th>$U^{-1}BU$</th>
<th>$B^TB$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>300</td>
<td>300</td>
<td>300</td>
<td>600</td>
<td>600</td>
<td>600</td>
</tr>
<tr>
<td>$d$</td>
<td>75</td>
<td>75</td>
<td>21</td>
<td>424</td>
<td>424</td>
<td>8</td>
</tr>
<tr>
<td>$\omega$</td>
<td>1.9</td>
<td>300</td>
<td>2.95</td>
<td>4</td>
<td>600</td>
<td>13</td>
</tr>
<tr>
<td>Magma-prob</td>
<td>1.14</td>
<td>7.11</td>
<td>0.23</td>
<td>6.4</td>
<td>184.7</td>
<td>6.04</td>
</tr>
<tr>
<td>Magma-det</td>
<td>1.31</td>
<td>10.55</td>
<td>0.24</td>
<td>6.4</td>
<td>185</td>
<td>6.07</td>
</tr>
<tr>
<td>ILUK-det</td>
<td>1.1</td>
<td>93.3</td>
<td>64.87</td>
<td>68.4</td>
<td>2305</td>
<td>155.3</td>
</tr>
<tr>
<td>CIA-sparse</td>
<td>0.32</td>
<td>4.32</td>
<td>0.81</td>
<td>4.4</td>
<td>352.6</td>
<td>2.15</td>
</tr>
<tr>
<td>CIA-dense</td>
<td>1.22</td>
<td>1.3</td>
<td>0.87</td>
<td>38.9</td>
<td>42.6</td>
<td>2.57</td>
</tr>
</tbody>
</table>

Computation time in seconds.
Other sparse matrices

On a athlon-1.8Ghz with 2Gb of RAM.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>$n$</th>
<th>$\omega$</th>
</tr>
</thead>
<tbody>
<tr>
<td>magma-prob</td>
<td>552</td>
<td>7.6</td>
</tr>
<tr>
<td>CIA-sparse</td>
<td>500</td>
<td>16.9</td>
</tr>
<tr>
<td>CIA-dense</td>
<td>1260</td>
<td>3</td>
</tr>
</tbody>
</table>

Computation time in seconds.
Other sparse matrices

On a athlon-1.8Ghz with 2Gb of RAM.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>TF12</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>552</td>
</tr>
<tr>
<td>$\omega$</td>
<td>7.6</td>
</tr>
<tr>
<td>magma-prob</td>
<td>10.12s</td>
</tr>
<tr>
<td>CIA-sparse</td>
<td>6.8s</td>
</tr>
<tr>
<td>CIA-dense</td>
<td>61.77s</td>
</tr>
</tbody>
</table>

Computation time in seconds.
Other sparse matrices

On a athlon-1.8Ghz with 2Gb of RAM.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>TF12</th>
<th>Tref500</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>552</td>
<td>500</td>
</tr>
<tr>
<td>$\omega$</td>
<td>7.6</td>
<td>16.9</td>
</tr>
<tr>
<td>magma-prob</td>
<td>10.12s</td>
<td>112s</td>
</tr>
<tr>
<td>CIA-sparse</td>
<td>6.8s</td>
<td>65.14s</td>
</tr>
<tr>
<td>CIA-dense</td>
<td>61.77s</td>
<td>372.6s</td>
</tr>
</tbody>
</table>

Computation time in seconds.
Other sparse matrices

On a athlon-1.8Ghz with 2Gb of RAM.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>TF12</th>
<th>Tref500</th>
<th>mk9b3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>552</td>
<td>500</td>
<td>1260</td>
</tr>
<tr>
<td>$\omega$</td>
<td>7.6</td>
<td>16.9</td>
<td>3</td>
</tr>
<tr>
<td>magma-prob</td>
<td>10.12s</td>
<td>112s</td>
<td>48.4s</td>
</tr>
<tr>
<td>CIA-sparse</td>
<td>6.8s</td>
<td>65.14s</td>
<td>31.25s</td>
</tr>
<tr>
<td>CIA-dense</td>
<td>61.77s</td>
<td>372.6s</td>
<td>433s</td>
</tr>
</tbody>
</table>

Computation time in seconds.
Sommaire

Efficient Computation of the Characteristic Polynomial

J-G. Dumas, C. Pernet, Z. Wan
Toward a truly sparse algorithm

Find the multiplicities by different techniques:
- compute some $\text{rank}(P_i(A))$ where $P_i$ is an irreducible factor of $P_{\min}^A$
- combinatorial search
- sieve remaining solutions by some evaluation of $P_{\text{char}}^A$
- ...
Toward a truly sparse algorithm

Find the multiplicities by different techniques:
- compute some \( \text{rank}(P_i(A)) \) where \( P_i \) is an irreducible factor of \( P_{\min}^A \)
- combinatorial search
- sieve remaining solutions by some evaluation of \( P_{\text{char}}^A \)
- ...

Applied to a graph theory problem:
\( \Rightarrow \) compute the characteristic polynomial of a 7140 \( \times \) 7140 sparse adjacency matrix in 1h4’ on a P4-2.4Ghz
Thank you!