Automatic Modeling and Ranking of Linear Algebra Algorithms

Paolo Bientinesi

AICES, RWTH Aachen
pauldj@aices.rwth-aachen.de

iWAPT 2012
7th International Workshop on Automatic Performance Tuning
July 17th, 2012
Kobe, Japan
One operation $\rightarrow$ multiple algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>alg-1</td>
<td></td>
</tr>
<tr>
<td>alg-2</td>
<td></td>
</tr>
<tr>
<td>alg-3</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>alg-n</td>
<td></td>
</tr>
</tbody>
</table>
Objective: Ranking

One operation $\rightarrow$ multiple algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>alg-1</td>
<td></td>
</tr>
<tr>
<td>alg-2</td>
<td></td>
</tr>
<tr>
<td>alg-3</td>
<td></td>
</tr>
<tr>
<td>alg-n</td>
<td></td>
</tr>
</tbody>
</table>

$\Rightarrow$

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>alg-4</td>
<td>27.0</td>
</tr>
<tr>
<td>alg-1</td>
<td>22.5</td>
</tr>
<tr>
<td>alg-n</td>
<td>15.5</td>
</tr>
<tr>
<td>alg-13</td>
<td>1.07</td>
</tr>
</tbody>
</table>
1 Motivation

2 Analytic Modeling

3 Modeling through Sampling

4 Results

5 Conclusions
**Tuning**

\[
\text{LU}(A)
\]

**Partition** \( A \rightarrow \begin{pmatrix} \frac{A_{TL}}{A_{BL}} & \frac{A_{TR}}{A_{BR}} \end{pmatrix} \)

*where* \( A_{TL} \) is \( 0 \times 0 \)

**While** \( \text{size}(A_{TL}) < \text{size}(A) \) **do**

**Repartition**

\[
\begin{pmatrix} \frac{A_{TL}}{A_{BL}} & \frac{A_{TR}}{A_{BR}} \end{pmatrix} \rightarrow \begin{pmatrix} \frac{A_{00}}{A_{10}} & \frac{A_{01}}{A_{11}} & \frac{A_{02}}{A_{12}} \end{pmatrix}
\]

*where* \( A_{11} \) is \( b \times b \)

\[
\begin{align*}
U_{01} & := L_{00}^{-1} A_{01} \\
L_{10} & := A_{10} U_{00}^{-1} \\
A_{11} & := \text{LU}(A_{11} - L_{10} U_{01})
\end{align*}
\]

**Continue**

\[
\begin{pmatrix} \frac{A_{TL}}{A_{BL}} & \frac{A_{TR}}{A_{BR}} \end{pmatrix} \leftarrow \begin{pmatrix} \frac{A_{00}}{A_{10}} & \frac{A_{01}}{A_{11}} & \frac{A_{02}}{A_{12}} \end{pmatrix}
\]

**endwhile**

- block size \( b \)?
- how many levels of recursion?
- recursive calls?
One Algorithm to rule them all? Not really

- serial
- multi-threaded
- distributed memory
- data parallel
- ooc
- unblocked
- blocked
- by blocks...
“One Algorithm to rule them all”?

Paolo Bientinesi (AICES, RWTH Aachen)
Automatic Ranking of Algorithms
July 17, 2012 5 / 35
“One Algorithm to rule them all”? Not really
Trilv: $X := L^{-1}$

Partition $\star \in \{L, X\}$ as
$$
\begin{pmatrix}
X_{TL} & 0 \\
X_{BL} & X_{BR}
\end{pmatrix}
$$
where $L_{TL}, X_{TL}$ are $0 \times 0$

While $\text{size}(L_{TL}) < \text{size}(L)$ do

Repartition
$$
\begin{align*}
& (X_{TL} | 0) \rightarrow \begin{pmatrix} X_{00} & 0 & 0 \\ X_{10} & X_{11} & 0 \\ X_{20} & X_{21} & X_{22} \end{pmatrix}, \quad \text{and} \quad
& (L_{TL} | 0) \rightarrow \begin{pmatrix} L_{00} & 0 & 0 \\ L_{10} & L_{11} & 0 \\ L_{20} & L_{21} & L_{22} \end{pmatrix}
\end{align*}
$$

Variant 1
$$
\begin{align*}
X_{10} & := L_{10} X_{00} \\
X_{10} & := -L_{11}^{-1} X_{10} \\
X_{11} & := L_{11}^{-1}
\end{align*}
$$

Variant 2
$$
\begin{align*}
X_{21} & := L_{22}^{-1} L_{21} \\
X_{21} & := -X_{21} L_{11}^{-1} \\
X_{11} & := L_{11}^{-1}
\end{align*}
$$

Variant 3
$$
\begin{align*}
X_{21} & := L_{22}^{-1} L_{21} \\
X_{20} & := X_{20} - X_{21} X_{10} \\
X_{10} & := L_{10} L_{00} \\
X_{11} & := L_{11}^{-1}
\end{align*}
$$

Variant 4
$$
\begin{align*}
X_{21} & := L_{22}^{-1} L_{21} \\
X_{20} & := X_{20} - X_{21} X_{10} \\
X_{10} & := L_{10} L_{00} \\
X_{11} & := L_{11}^{-1}
\end{align*}
$$

Continue
$$
\begin{align*}
& \left( \begin{pmatrix} X_{TL} & 0 \\ X_{BL} & X_{BR} \end{pmatrix} \right) \leftarrow \left( \begin{pmatrix} X_{00} & 0 & 0 \\ X_{10} & X_{11} & 0 \\ X_{20} & X_{21} & X_{22} \end{pmatrix} \right), \quad \text{and} \quad
& \left( \begin{pmatrix} L_{TL} & 0 \\ L_{BL} & L_{BR} \end{pmatrix} \right) \leftarrow \left( \begin{pmatrix} L_{00} & 0 & 0 \\ L_{10} & L_{11} & 0 \\ L_{20} & L_{21} & L_{22} \end{pmatrix} \right)
\end{align*}
$$

endwhile
Sylvester equation: \( AX + XB = C \)

<table>
<thead>
<tr>
<th>Partition ( \star \in {A, B, C} ) as ( \begin{pmatrix} \star_{TL} &amp; \star_{TR} \ \star_{BL} &amp; \star_{BR} \end{pmatrix} ) where ( A_{BR}, B_{TL}, C_{BL} ) are ( 0 \times 0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>While ( \text{size}(C_{TL}) &lt; \text{size}(C) ) do</td>
</tr>
<tr>
<td>Repartition</td>
</tr>
<tr>
<td>( \begin{pmatrix} C_{TL} &amp; C_{TR} \ C_{BL} &amp; C_{BR} \end{pmatrix} \rightarrow \begin{pmatrix} C_{00} &amp; C_{01} &amp; C_{02} \ C_{10} &amp; C_{11} &amp; C_{12} \ C_{20} &amp; C_{21} &amp; C_{22} \end{pmatrix} ), ( \begin{pmatrix} A_{TL} &amp; A_{TR} \ A_{BL} &amp; A_{BR} \end{pmatrix} \rightarrow \begin{pmatrix} A_{00} &amp; A_{01} &amp; A_{02} \ A_{10} &amp; A_{11} &amp; A_{12} \ A_{20} &amp; A_{21} &amp; A_{22} \end{pmatrix} ), ( \ldots )</td>
</tr>
</tbody>
</table>

**Variant 1**

- \( C_{10} := C_{10} - A_{12} C_{20} \)
- \( C_{10} := \Omega(A_{11}, B_{00}, C_{10}) \)
- \( C_{21} := C_{21} - C_{20} B_{01} \)
- \( C_{21} := \Omega(A_{22}, B_{11}, C_{21}) \)
- \( C_{11} := C_{11} - A_{12} C_{21} - C_{10} B_{01} \)
- \( C_{11} := \Omega(A_{11}, B_{11}, C_{11}) \)

**Variant 16**

- \( C_{11} := C_{11} - C_{10} B_{01} \)
- \( C_{11} := \Omega(A_{11}, B_{11}, C_{11}) \)
- \( C_{01} := C_{01} - C_{00} B_{01} - A_{01} C_{11} \)
- \( C_{01} := \Omega(A_{00}, B_{11}, C_{10}) \)
- \( C_{12} := C_{12} - C_{10} B_{02} - C_{11} B_{12} \)
- \( C_{12} := \Omega(A_{11}, B_{22}, C_{12}) \)
- \( C_{02} := C_{02} - A_{01} C_{12} \)

**Continue**

- \( \begin{pmatrix} C_{TL} & C_{TR} \\ C_{BL} & C_{BR} \end{pmatrix} \leftarrow \begin{pmatrix} C_{00} & C_{01} & C_{02} \\ C_{10} & C_{11} & C_{12} \\ C_{20} & C_{21} & C_{22} \end{pmatrix} \), \( \begin{pmatrix} A_{TL} & A_{TR} \\ A_{BL} & A_{BR} \end{pmatrix} \leftarrow \begin{pmatrix} A_{00} & A_{01} & A_{02} \\ A_{10} & A_{11} & A_{12} \\ A_{20} & A_{21} & A_{22} \end{pmatrix} \), \( \ldots \)
**Generation of algorithms: CLAK**

**GWAS:**

\[ b_{ij} := \left( X_i^T M_j^{-1} X_i \right)^{-1} X_i^T M_j^{-1} y_j \]

| Algorithm 1 | Algorithm 2 | ... | Algorithm 20 | ...
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( LL^T = M )</td>
<td>( LL^T = M )</td>
<td>( ZWZ^T = \Phi )</td>
<td>( ZWZ^T = \Phi )</td>
<td></td>
</tr>
<tr>
<td>( X := L^{-1} X )</td>
<td>( X := L^{-1} X )</td>
<td>( D := (hW + (1-h)I)^{-1} )</td>
<td>( D := (hW + (1-h)I)^{-1} )</td>
<td></td>
</tr>
<tr>
<td>( S := X^T X )</td>
<td>( QR := X )</td>
<td>( KK^T = D )</td>
<td>( KK^T = D )</td>
<td></td>
</tr>
<tr>
<td>( GG^T = S )</td>
<td>( y := L^{-1} y )</td>
<td>( X := Z^T X )</td>
<td>( X := Z^T X )</td>
<td></td>
</tr>
<tr>
<td>( y := L^{-1} y )</td>
<td>( b := QR y )</td>
<td>( X := K^T X )</td>
<td>( X := K^T X )</td>
<td></td>
</tr>
<tr>
<td>( b := X^T y )</td>
<td>( b := R^{-1} b )</td>
<td>( QR := X )</td>
<td>( QR := X )</td>
<td></td>
</tr>
<tr>
<td>( b := G^{-1} b )</td>
<td>( b := G^{-1} b )</td>
<td>( y := L^{-1} y )</td>
<td>( y := L^{-1} y )</td>
<td></td>
</tr>
<tr>
<td>( b := G^{-T} b )</td>
<td>( b := G^{-T} b )</td>
<td>( b := R^{-1} b )</td>
<td>( b := R^{-1} b )</td>
<td></td>
</tr>
</tbody>
</table>
“O Brother, Where Art Thou?”
Wishlist

- **Speed**
  - No direct execution of the algorithm
  - Possibly no execution at all
- **Accuracy**
- **Automation**
Wishlist

- Speed
  - No direct execution of the algorithm
  - Possibly no execution at all
- Accuracy
- Automation

Approach: Performance Modeling

- Analytic Models
- Sampling
Wishlist

- Speed
  - No direct execution of the algorithm
  - Possibly no execution at all
- Accuracy
- Automation

Approach: Performance Modeling

- Analytic Models
- Sampling

Idea

- Exploit modularity: from kernels to algorithms
1. Motivation
2. Analytic Modeling
3. Modeling through Sampling
4. Results
5. Conclusions
Analytic modeling

- no execution of code
- models built from knowledge
Analytic modeling

- no execution of code
- models built from knowledge

**Model (simplified version)**

\[
\text{Time} = \alpha \#\text{flops} + \sum_i \beta_i \#\text{miss}_i
\]
Analytic modeling

- no execution of code
- models built from knowledge

Model (simplified version)

\[
\text{Time} = \alpha \ \#\text{flops} + \sum_i \beta_i \ \#\text{miss}_i
\]

- storage scheme
- size of the operands
- size and number of caches
- hardware & software prefetching
- how the algorithm traverses the operands
- size of cache-lines
- compilation level
- ...
Feasible?

Roman Iakymchuk

"Execution-less Performance Modeling"

Models for specific architecture, BLAS routine, implementation, . . .

Paolo Bientinesi (AICES, RWTH Aachen)
Feasible?

Roman Iakymchuk

“Execution-less Performance Modeling”
Feasible?

Roman Iakymchuk

“Execution-less Performance Modeling”

Models for specific architecture, BLAS routine, implementation, . . .
Example: GotoBLAS

Rank-k update

\[ A := A + xy^T \]

GER, BLAS2

\[
\begin{array}{c}
\begin{array}{c}
\begin{array}{cc}
\hline
m & p \\
\hline
A_{22} & \\
\hline
\end{array}
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\begin{array}{c}
\begin{array}{cc}
\hline
n & d \\
\hline
q & \\
\hline
\end{array}
\end{array}
\end{array}
\]

\[
L_1 \text{ misses } = \begin{cases} \left\lceil p \cdot d \right\rceil + \left\lceil q \cdot d \right\rceil + \lfloor m \cdot q \cdot d \rfloor, & \text{if } m - p < \frac{d}{2} \\ \left\lceil p \cdot d \right\rceil + \left\lceil q \cdot d \right\rceil + q - 1 \sum_{i=1}^{m} \left( \left\lceil p + (m_i \mod d) \cdot d \right\rceil + \eta(i) \right), & \text{otherwise} \end{cases}
\]

with \( \eta(i) = \min(d - 1, \lfloor m + (m_i \mod d) \cdot d \rfloor - \left\lceil p + (m_i \mod d) \cdot d \right\rceil) \)
Example: GotoBLAS

Rank-k update

\[ A := A + xy^T \]

GER, BLAS2

L1 misses =

\[
\begin{cases}
\left\lceil \frac{p}{d} \right\rceil \quad + \quad \left\lceil \frac{q}{d} \right\rceil \quad + \quad \left\lfloor \frac{mq}{d} \right\rfloor, & \text{if } m - p < d \\
2 \left\lceil \frac{p}{d} \right\rceil \quad + \quad \left\lceil \frac{q}{d} \right\rceil \quad + \quad \sum_{i=1}^{q-1} \left( \left\lceil \frac{p + (mi \mod d)}{d} \right\rceil + \eta(i) \right), & \text{otherwise}
\end{cases}
\]

with

\[
\eta(i) = \min \left( d - 1, \left\lfloor \frac{m + (mi \mod d)}{d} \right\rfloor - \left\lfloor \frac{p + (mi \mod d)}{d} \right\rfloor \right)
\]
Predicting the execution time

LU factorization, unblocked

![Graph showing measured and modeled execution time deviations](image)
Analytic models

Wishlist

No direct execution of the algorithm
Possibly no execution at all

Accuracy
⇒ accurate ranking

Automation
Analytic models

Wishlist

- Speed ✓ ✗
Analytic models

Wishlist

- Speed ✓ ✗
  - No direct execution of the algorithm ✓
Analytic models

Wishlist

- Speed ✓ ✗
  - No direct execution of the algorithm ✓
  - Possibly no execution at all ✓
Analytic models

Wishlist

- Speed ✓✗
  - No direct execution of the algorithm ✓
  - Possibly no execution at all ✓

- Accuracy ✓ ⇒ accurate ranking
Analytic models

Wishlist

- Speed ✔️ ✗
  - No direct execution of the algorithm ✔️
  - Possibly no execution at all ✔️
- Accuracy ✔️ ⇒ accurate ranking
- Automation ✗
1. Motivation
2. Analytic Modeling
3. Modeling through Sampling
4. Results
5. Conclusions
Roadmap

- Sample the kernels
Modeling through sampling

Roadmap
- Sample the kernels
- Build polynomial models
Roadmap

- Sample the kernels
- Build polynomial models
- Create a database
Modeling through sampling

Roadmap

- Sample the kernels
- Build polynomial models
- Create a database
- Algorithm execution ≡ querying
Sampling

A X = B

dtrsm(side, uplo, transA, diag, m, n, alpha, A, ldA, B, ldB)
Sampling

\[ A \times X = B \]

dtrsm(side, uplo, transA, diag, m, n, alpha, A, ldA, B, ldB)

blind sampling ⇒ curse of dimensionality ⇒ intractable low accuracy
Sampling

\[ A X = B \]

\[
\text{dtrsm(side, uplo, transA, diag, m, n, alpha, A, ldA, B, ldB)}
\]

blind sampling ⇒ curse of dimensionality ⇒ intractable
low accuracy

Solution:
- Understand the kernels
- Integrate knowledge into the modeling and models
A X = B

dtrsm(side, uplo, transA, diag, m, n, alpha, A, ldA, B, ldB)
A X = B

dtrsm(side, uplo, transA, diag, m, n, alpha, A, ldA, B, ldB)

Not all arguments affect performance!
Not all arguments affect performance!

- Polynomial models, piecewise defined
A X = B

dtrsm(side, uplo, transA, diag, m, n, alpha, A, ldA, B, ldB)

- Not all arguments affect performance!
- Polynomial models, piecewise defined
- Discrete cases, multiple models
Understanding the kernels

A X = B

dtrsm(side, uplo, transA, diag, m, n, alpha, A, ldA, B, ldB)

- Not all arguments affect performance!
- Polynomial models, piecewise defined
- Discrete cases, multiple models
- Fluctuations $\Rightarrow$ need for stochastic quantities
### Understanding the kernels

$$AX = B$$

```c
dtrsm(side, uplo, transA, diag, m, n, alpha, A, ldA, B, ldb)
```

- Not all arguments affect performance!
- Polynomial models, piecewise defined
- Discrete cases, multiple models
- Fluctuations $\Rightarrow$ need for stochastic quantities
- **Accuracy**: not for performance, for ranking!
Size arguments

dtrsm(L, L, N, N, m, n, .5, L, 2500, B, 2500)

Time [cycles]

GotoBLAS2

Paolo Bientinesi (AICES, RWTH Aachen)
Automatic Ranking of Algorithms
July 17, 2012 21 / 35
Size arguments

dtrsm(L, L, N, N, m, n, .5, L, 2500, B, 2500)

\[
\begin{align*}
\text{Time [cycles]} & \quad 0 \quad 256 \quad 512 \quad 768 \quad 1,024 \\
0 \quad 1 \quad 2 \quad 3 \quad \cdot 10^8 \quad n
\end{align*}
\]

GotoBLAS2
LS model
Piecewise Polynomials

\[ \text{dtrsm}(L, L, N, N, m, n, .5, L, 2500, B, 2500) \]

Absolute error [cycles]

GotoBLAS2

Paolo Bientinesi (AICES, RWTH Aachen)
Flags

\[ \text{dtrsm}(L, L, N, N, m, n, .5, L, 2500, B, 2500) \]

\[
\begin{array}{c|c|c|c}
\hline
n & 0 & 256 & 512 & 768 & 1,024 \\
\hline
\text{Time [cycles]} & 0 & \ldots & 10^8 \\
\hline
\end{array}
\]

GotoBLAS2
- \( \text{side} = L, \text{uplo} = L \)
- \( \text{side} = L, \text{uplo} = U \)
- \( \text{side} = R, \text{uplo} = L \)
- \( \text{side} = R, \text{uplo} = U \)
Independent models

dtrsm(L, L, N, N, m, n, .5, L, 2500, B, 2500)

Absolute error [cycles]

GotoBLAS2
- side = L, uplo = L
- side = L, uplo = U
- side = R, uplo = L
- side = R, uplo = U

Paolo Bientinesi (AICES, RWTH Aachen)
Variability $\Rightarrow$ statistical info

DGEMM

![Graph showing performance comparison of GotoBLAS2, MKL, and ATLAS](image)

- GotoBLAS2
- MKL
- ATLAS

Paolo Bientinesi (AICES, RWTH Aachen)
Automatic Ranking of Algorithms
July 17, 2012 23 / 35
Building the models

- Two tools
  - Sampler
  - Modeler
Building the models

- Two tools
  - Sampler
  - Modeler

- Two modeling strategies
  - Expansion
  - Adaptive refinement
Model Expansion

\[ \begin{array}{c}
\text{n} \\
\text{m}
\end{array} \]

- Completed region
- Paolo Bientinesi (AICES, RWTH Aachen)
- Automatic Ranking of Algorithms
- July 17, 2012 25 / 35
Model Expansion

\[ n \]

\[ m \]

completed region
Model Expansion

completed region

\[ n \]

\[ m \]
Model Expansion

\[ \begin{array}{c}
\text{completed} \\
\text{region}
\end{array} \]

\[ \begin{array}{c}
\text{m} \\
n
\end{array} \]
Model Expansion

completed region

Paolo Bientinesi (AICES, RWTH Aachen)

Automatic Ranking of Algorithms

July 17, 2012
Model Expansion

$n$

$m$

completed region
Adaptive Refinement

dtrsm(L, L, N, N, m, n, .5, L, 2500, B, 2500)
Adaptive Refinement

dtrsm(L, L, N, N, m, n, .5, L, 2500, B, 2500)
Adaptive Refinement

dtrsm(L, L, N, N, m, n, .5, L, 2500, B, 2500)
Adaptive Refinement

dtrsm(L, L, N, N, m, n, .5, L, 2500, B, 2500)
Adaptive Refinement

dtrsm(L, L, N, N, m, n, .5, L, 2500, B, 2500)
Adaptive Refinement

dtrsm(L, L, N, N, m, n, .5, L, 2500, B, 2500)
From algorithm to prediction

TriInv_1('L’,300,A,300,100)

Partition \( L \rightarrow \begin{pmatrix} L_{TL} & 0 \\ L_{BL} & L_{BR} \end{pmatrix} \)

where \( L_{TL} \) is \( 0 \times 0 \)

While \( \text{size}(L_{TL}) < \text{size}(L) \) do

Repartition

\( \begin{pmatrix} L_{TL} & 0 \\ L_{BL} & L_{BR} \end{pmatrix} \rightarrow \begin{pmatrix} L_{00} & 0 & 0 \\ L_{10} & L_{11} & 0 \\ L_{20} & L_{21} & L_{22} \end{pmatrix} \)

where \( L_{11} \) is \( b \times b \)

\( L_{10} := \text{TRMM}(L_{10}, L_{00}) \)
\( L_{10} := \text{TRSM}(-L_{11} L_{10}) \)
\( L_{11} := \text{trinv}(L_{11}) \)

Continue

\( \begin{pmatrix} L_{TL} & 0 \\ L_{BL} & L_{BR} \end{pmatrix} \leftarrow \begin{pmatrix} L_{00} & 0 & 0 \\ L_{10} & L_{11} & 0 \\ L_{20} & L_{21} & L_{22} \end{pmatrix} \)

endwhile
From algorithm to prediction

\[
\text{TriInv}_1('L', 300, A, 300, 100)
\]

\[
\text{Partition } L \rightarrow \begin{pmatrix} L_{TL} & 0 \\ L_{BL} & L_{BR} \end{pmatrix}
\]

where \( L_{TL} \) is \( 0 \times 0 \)

\[
\text{While } \text{size}(L_{TL}) < \text{size}(L) \text{ do}
\]

\[
\text{Repartition } \begin{pmatrix} L_{TL} & 0 \\ L_{BL} & L_{BR} \end{pmatrix} \rightarrow \begin{pmatrix} L_{00} & 0 & 0 \\ L_{10} & L_{11} & 0 \\ L_{20} & L_{21} & L_{22} \end{pmatrix}
\]

where \( L_{11} \) is \( b \times b \)

\[
L_{10} := \text{TRMM}(L_{10}, L_{00})
\]

\[
L_{10} := \text{TRSM}(-L_{11}L_{10})
\]

\[
L_{11} := \text{trinv}(L_{11})
\]

\[
\text{Continue } \begin{pmatrix} L_{TL} & 0 \\ L_{BL} & L_{BR} \end{pmatrix} \leftarrow \begin{pmatrix} L_{00} & 0 & 0 \\ L_{10} & L_{11} & 0 \\ L_{20} & L_{21} & L_{22} \end{pmatrix}
\]

endwhile

dtrmm(100, 0, 1, 300, 300)

dtrsm(100, 0, -1, 300, 300)

triinv_1('L', 100, 300, 1)

dtrmm(100, 100, 1, 300, 300)

dtrsm(100, 100, -1, 300, 300)

triinv_1('L', 100, 300, 1)

dtrmm(100, 200, 1, 300, 300)

dtrsm(100, 200, -1, 300, 300)

triinv_1('L', 100, 300, 1)
1. Motivation
2. Analytic Modeling
3. Modeling through Sampling
4. Results
5. Conclusions

- TriInv: efficiency
- TriInv: block size tuning
- Sylvester Equation
- GWAS
$X := L^{-1}$
$X := L^{-1}$
Statistics

\[ X := L^{-1} \]

Efficiency

<table>
<thead>
<tr>
<th>Measurement</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>median</td>
<td>standard dev.</td>
</tr>
</tbody>
</table>

- variant 1
- variant 2
- variant 3
- variant 4

Paolo Bientinesi (AICES, RWTH Aachen)
Tuning: block size

\[ X := L^{-1} \]
Tuning: block size

\[ X := L^{-1} \]

![Graph showing efficiency vs blocksize for different variants.](image)

- **variant 1**
- **variant 2**
- **variant 3**
- **variant 4**

- Measurements
- Prediction:
  - median
  - standard dev.
Sylvester equation – 16 variants

$AX + XB = C$
## Sylvester equation – 16 variants

\[ AX + XB = C \]

<table>
<thead>
<tr>
<th>Variant</th>
<th>Efficiency predicted</th>
<th>Efficiency measured</th>
</tr>
</thead>
<tbody>
<tr>
<td>Var-1</td>
<td>27.03%</td>
<td>24.04%</td>
</tr>
<tr>
<td>Var-2</td>
<td>22.52%</td>
<td>21.07%</td>
</tr>
<tr>
<td>Var-5</td>
<td>15.51%</td>
<td>18.82%</td>
</tr>
<tr>
<td>Var-6</td>
<td>13.72%</td>
<td>18.51%</td>
</tr>
<tr>
<td>Var-16</td>
<td>1.79%</td>
<td>2.21%</td>
</tr>
<tr>
<td>Var-3</td>
<td>1.52%</td>
<td>1.52%</td>
</tr>
<tr>
<td>Var-4</td>
<td>1.50%</td>
<td>1.45%</td>
</tr>
<tr>
<td>Var-8</td>
<td>1.49%</td>
<td>1.37%</td>
</tr>
<tr>
<td>Var-10</td>
<td>1.43%</td>
<td>1.53%</td>
</tr>
<tr>
<td>Var-15</td>
<td>1.43%</td>
<td>1.52%</td>
</tr>
<tr>
<td>Var-9</td>
<td>1.40%</td>
<td>1.48%</td>
</tr>
<tr>
<td>Var-14</td>
<td>1.34%</td>
<td>1.33%</td>
</tr>
<tr>
<td>Var-12</td>
<td>1.29%</td>
<td>1.43%</td>
</tr>
<tr>
<td>Var-7</td>
<td>1.06%</td>
<td>1.16%</td>
</tr>
<tr>
<td>Var-11</td>
<td>1.04%</td>
<td>1.07%</td>
</tr>
<tr>
<td>Var-13</td>
<td>1.01%</td>
<td>1.01%</td>
</tr>
</tbody>
</table>
$$b := (X^T M^{-1} X)^{-1} X^T M^{-1} y$$
GWAS

\[ b := (X^T M^{-1} X)^{-1} X^T M^{-1} y \]
Modeling through sampling

Wishlist
Wishlist

- Speed ✓
Wishlist

- Speed ✓
  - No direct execution of the algorithm ✓
Wishlist

- Speed ✓
  - No direct execution of the algorithm ✓
  - Possibly no execution at all ❌
Modeling through sampling

Wishlist

- Speed ✓
  - No direct execution of the algorithm ✓
  - Possibly no execution at all ✗

- Accuracy ✓  ⇒ accurate ranking
Wishlist

- Speed ✓
  - No direct execution of the algorithm ✓
  - Possibly no execution at all ✗

- Accuracy ✓ ⇒ accurate ranking

- Automation ✓
1 Motivation
2 Analytic Modeling
3 Modeling through Sampling
4 Results
5 Conclusions
Ranking of algorithms

- Request: no direct execution
- Solutions:
  - Analytic models
  - Models through samples
- Accuracy in the models vs. accuracy in the ranking

What's next? . . . we just started!

Extrapolation, MPI, sparse computations, . . .

Financial support from the Deutsche Forschungsgemeinschaft (German Research Association) through grant GSC 111 is gratefully acknowledged.
Conclusions

**Ranking of algorithms**

- Request: no direct execution
- Solutions:
  - Analytic models
  - Models through samples
- Accuracy in the models vs. accuracy in the ranking

**What's next? . . . we just started!**

- Extrapolation, MPI, sparse computations, . . .

---

Financial support from the Deutsche Forschungsgemeinschaft (German Research Association) through grant GSC 111 is gratefully acknowledged.