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

Objective: Ranking

One operation $\rightarrow$ multiple algorithms

	Algorithm
Metric,	alg-1
	alg-2
	alg-3
	⋮
	alg-n

Objective: Ranking

One operation $\rightarrow$ multiple algorithms

Metric,

Algorithm
alg-1
alg-2
alg-3
⋮
alg-n

$\Longrightarrow$

Algorithm	Metric
alg-4	27.0
alg-1	22.5
alg-n	15.5
⋮	⋮
alg-13	1.07

Tuning

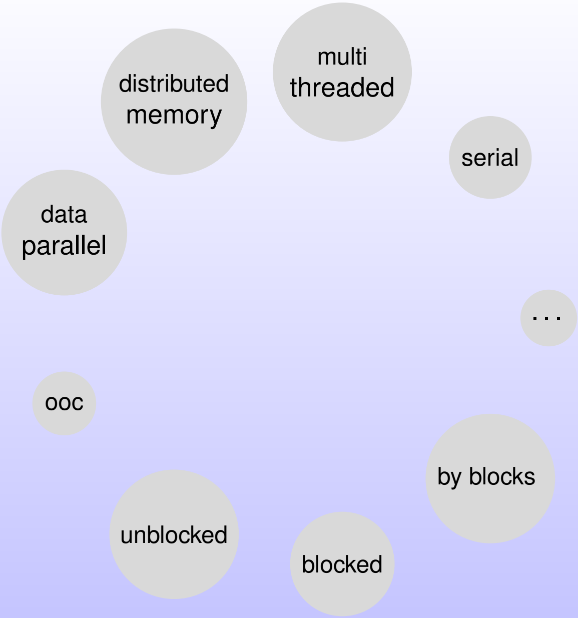
$\mathbf{LU}(\mathbf{A})$

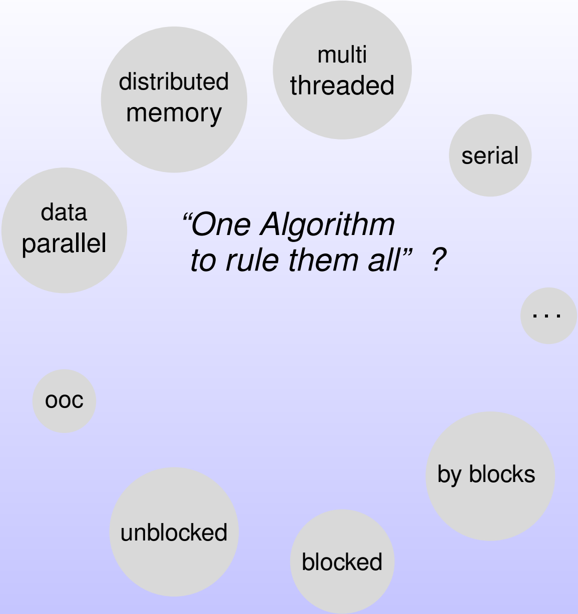
Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$
where A_{TL} is 0×0
While $size(A_{TL}) < size(A)$ **do**
Repartition
$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline A_{10} & A_{11} & A_{12} \\ A_{20} & A_{21} & A_{22} \end{array}\right)$$
where A_{11} is $b \times b$

$U_{01} := L_{00}^{-1} A_{01}$
$L_{10} := A_{10} U_{00}^{-1}$
$A_{11} := \text{LU}(A_{11} - L_{10} U_{01})$

Continue
$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ A_{10} & A_{11} & A_{12} \\ \hline A_{20} & A_{21} & A_{22} \end{array}\right)$$
endwhile

- block size b?
- how many levels of recursion?
- recursive calls?


distributed memory
multi threaded
serial
data parallel
. . .
ooc
by blocks
unblocked
blocked



"One Algorithm to rule them all" ?

distributed memory

multi threaded

serial

data parallel

“One Algorithm to rule them all” ?

. . .

ooc

Not really

by blocks

unblocked

blocked

Generation of algorithms: Cl1ck

TriInv: $X := L^{-1}$

Partition $\star \in \{L, X\}$ **as** $\left(\begin{array}{c|c} \star_{TL} & 0 \\ \hline \star_{BL} & \star_{BR} \end{array}\right)$ **where** L_{TL}, X_{TL} are 0×0

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

Repartition

$$\left(\begin{array}{c|c} X_{TL} & 0 \\ \hline X_{BL} & X_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} X_{00} & 0 & 0 \\ \hline X_{10} & X_{11} & 0 \\ \hline X_{20} & X_{21} & X_{22} \end{array}\right), \text{ and } \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline L_{10} & L_{11} & 0 \\ \hline L_{20} & L_{21} & L_{22} \end{array}\right)$$

Variant 1	Variant 2	Variant 3	Variant 4
$X_{10} := L_{10} X_{00}$	$X_{21} := L_{22}^{-1} L_{21}$	$X_{21} := L_{22}^{-1} L_{21}$	$X_{21} := L_{22}^{-1} L_{21}$
$X_{10} := -L_{11}^{-1} X_{10}$	$X_{21} := -X_{21} L_{11}^{-1}$	$X_{20} := X_{20} - X_{21} X_{10}$	$X_{20} := X_{20} - X_{21} X_{10}$
$X_{11} := L_{11}^{-1}$	$X_{11} := L_{11}^{-1}$	$X_{10} := L_{10} L_{00}$	$X_{10} := L_{10} L_{00}$
		$X_{11} := L_{11}^{-1}$	$X_{11} := L_{11}^{-1}$

Continue

$$\left(\begin{array}{c|c} X_{TL} & 0 \\ \hline X_{BL} & X_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} X_{00} & 0 & 0 \\ \hline X_{10} & X_{11} & 0 \\ \hline X_{20} & X_{21} & X_{22} \end{array}\right), \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline L_{10} & L_{11} & 0 \\ \hline L_{20} & L_{21} & L_{22} \end{array}\right)$$

endwhile

Generation of algorithms: Cl1ck

Sylvester equation: $AX + XB = C$

Partition $\star \in \{A, B, C\}$ **as** $\left(\begin{array}{c|c} \star_{TL} & \star_{TR} \\ \hline \star_{BL} & \star_{BR} \end{array}\right)$ **where** A_{BR}, B_{TL}, C_{BL} **are** 0×0

While $size(C_{TL}) < size(C)$ **do**

Repartition

$$\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} C_{00} & C_{01} & C_{02} \\ \hline C_{10} & C_{11} & C_{12} \\ \hline C_{20} & C_{21} & C_{22} \end{array}\right), \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline A_{10} & A_{11} & A_{12} \\ \hline A_{20} & A_{21} & A_{22} \end{array}\right), \ldots$$

Variant 1	...	Variant 16
$C_{10} := C_{10} - A_{12} C_{20}$	$C_{10} := C_{10} - A_{12} C_{20}$	$C_{11} := C_{11} - C_{10} B_{01}$
$C_{10} := \Omega(A_{11}, B_{00}, C_{10})$	$C_{10} := \Omega(A_{11}, B_{00}, C_{10})$	$C_{11} := \Omega(A_{11}, B_{11}, C_{11})$
$C_{21} := C_{21} - C_{20} B_{01}$	$C_{11} := C_{11} - C_{10} B_{01} - A_{12} C_{21}$	$C_{01} := C_{01} - C_{00} B_{01} - A_{01} C_{11}$
$C_{21} := \Omega(A_{22}, B_{11}, C_{21})$	$C_{11} := \Omega(A_{11}, B_{11}, C_{11})$	$C_{01} := \Omega(A_{00}, B_{11}, C_{10})$
$C_{11} := C_{11} - A_{12} C_{21} - C_{10} B_{01}$	$C_{12} := C_{12} - C_{10} B_{02} - C_{11} B_{12}$	$C_{12} := C_{12} - C_{10} B_{02} - C_{11} B_{12}$
$C_{11} := \Omega(A_{11}, B_{11}, C_{11})$	$C_{12} := C_{12} - A_{12} C_{22}$	$C_{12} := \Omega(A_{11}, B_{22}, C_{12})$
	$C_{12} := \Omega(A_{11}, B_{22}, C_{12})$	$C_{02} := C_{02} - A_{01} C_{12}$

Continue

$$\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} C_{00} & C_{01} & C_{02} \\ \hline C_{10} & C_{11} & C_{12} \\ \hline C_{20} & C_{21} & C_{22} \end{array}\right), \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline A_{10} & A_{11} & A_{12} \\ \hline A_{20} & A_{21} & A_{22} \end{array}\right), \ldots$$

endwhile

Generation of algorithms: CLAK

GWAS: $b_{ij} := (X_i^T M_j^{-1} X_i)^{-1} X_i^T M_j^{-1} y_j$

Algorithm 1	Algorithm 2	...	Algorithm 20	...
$LL^T = M$	$LL^T = M$		$ZWZ^T = \Phi$	
$X := L^{-1}X$	$X := L^{-1}X$		$D := (hW + (1-h)I)^{-1}$	
$S := X^T X$	$QR := X$		$KK^T = D$	
$GG^T = S$	$y := L^{-1}y$		$X := Z^T X$	
$y := L^{-1}y$	$b := Q^T y$		$X := K^T X$	
$b := X^T y$	$b := R^{-1}b$		$QR := X$	
$b := G^{-1}b$			$y := L^{-1}y$	
$b := G^{-T}b$			$b := Q^T y$	
			$b := R^{-1}b$	

"O Brother, Where Art Thou?"

“O Brother, Where Art Thou?”

Wishlist

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

"O Brother, Where Art Thou?"

Wishlist

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

Approach: Performance Modeling

- Analytic Models
- Sampling

“O Brother, Where Art Thou?”

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

Analytic modeling

- no execution of code
- models built from knowledge

Analytic modeling

- no execution of code
- models built from knowledge

Model (simplified version)

$$\texttt{Time} = \alpha \,\#\texttt{flops} + \sum_i \beta_i \,\#\texttt{miss}_i$$

Analytic modeling

- no execution of code
- models built from knowledge

Model (simplified version)

$$\texttt{Time} = \alpha \, \#\texttt{flops} + \sum_i \beta_i \, \#\texttt{miss}_i$$

- 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?

Feasible?

Roman Iakymchuk

‘‘Execution-less Performance Modeling’’

Feasible?

Roman Iakymchuk

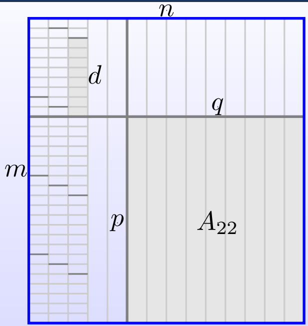
“Execution-less Performance Modeling”

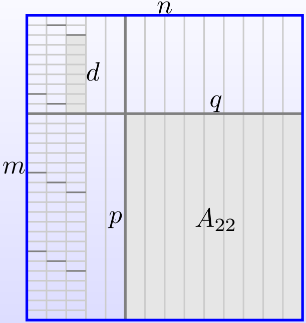
Models for specific architecture, BLAS routine, implementation, ...

Rank-k update

$A := A + xy^T$

GER, BLAS2





Rank-k update

$A := A + xy^T$

GER, BLAS2

L1 misses =

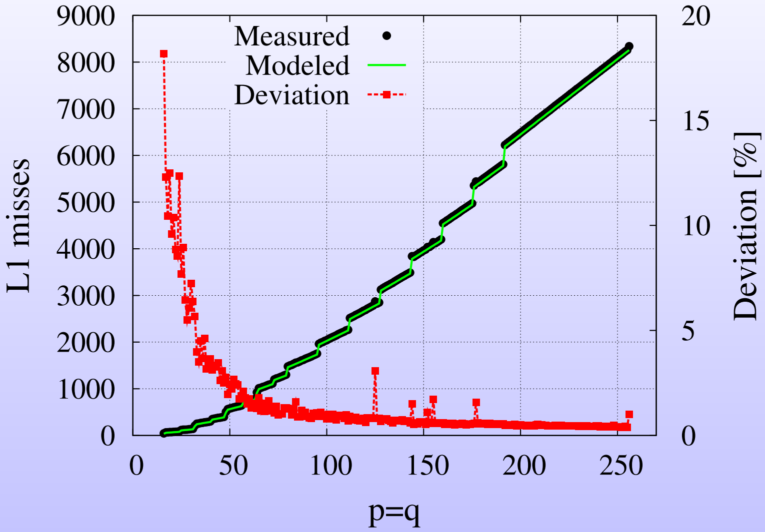
$$\begin{cases} \left\lceil \frac{p}{d} \right\rceil + \left\lceil \frac{q}{d} \right\rceil + \left\lfloor \frac{mq}{d} \right\rfloor, & \text{if } m - p < d \\ 2\left\lceil \frac{p}{d} \right\rceil + \left\lceil \frac{q}{d} \right\rceil + \sum_{i=1}^{q-1} \left(\left\lceil \frac{p + (mi \bmod d)}{d}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

with

$$\eta(i) = \min\left(d - 1, \left\lfloor \frac{m + (mi \bmod d)}{d} \right\rfloor - \left\lceil \frac{p + (mi \bmod d)}{d} \right\rceil \right)$$

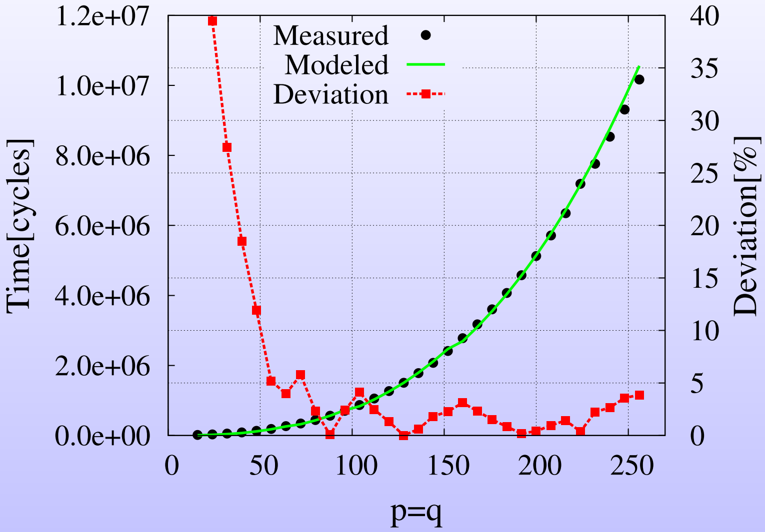
Accuracy

GER, GotoBLAS2



Predicting the execution time

LU factorization, unblocked



Analytic models

Wishlist

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

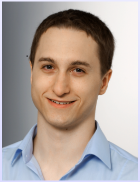
Wishlist

- Speed ✓ ✖
  - No direct execution of the algorithm ✓
  - Possibly no execution at all ✓
- Accuracy ✓ $\Rightarrow$ accurate ranking

Analytic models

Wishlist

- Speed ✓ ✖
  - No direct execution of the algorithm ✓
  - Possibly no execution at all ✓
- Accuracy ✓ $\Rightarrow$ accurate ranking
- Automation ✖



Elmar Peise

Modeling through sampling

Roadmap

- Sample the kernels

Modeling through sampling

Roadmap

- Sample the kernels
- Build polynomial models

Modeling through sampling

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 $\equiv$ querying

Sampling

A X = B

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

Sampling

A 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

```
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

Understanding the kernels

A X = B

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

Understanding the kernels

A X = B

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

- Not all arguments affect performance!

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

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

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

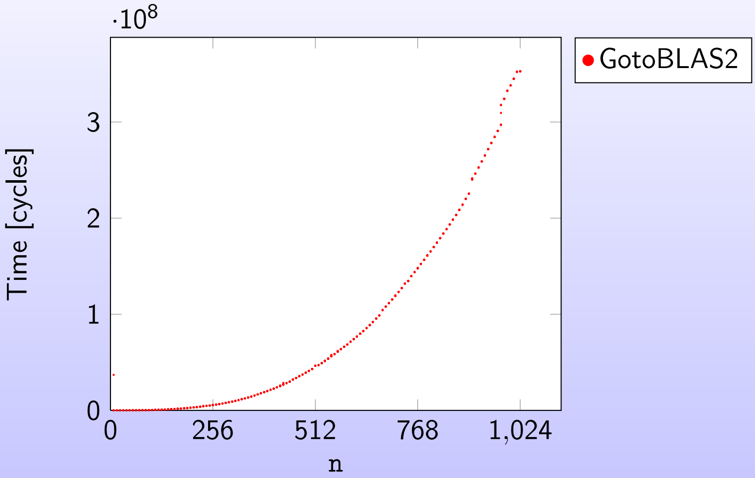
A X = B

```
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
- **Accuracy**: not for performance, for ranking!

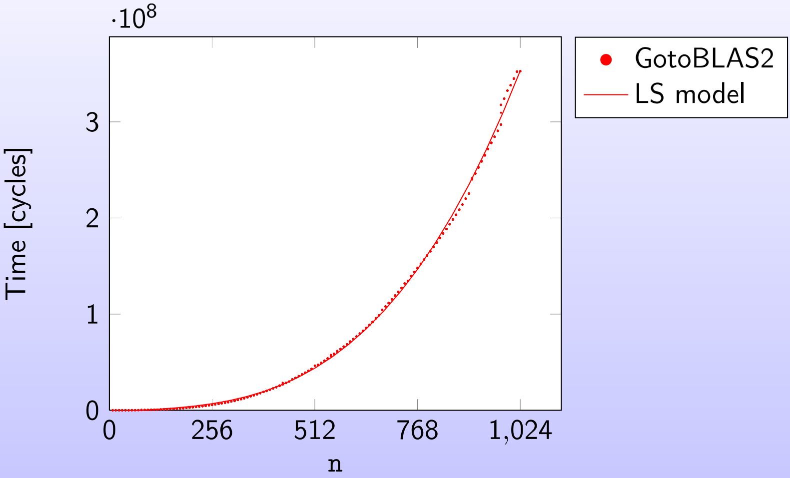
Size arguments

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



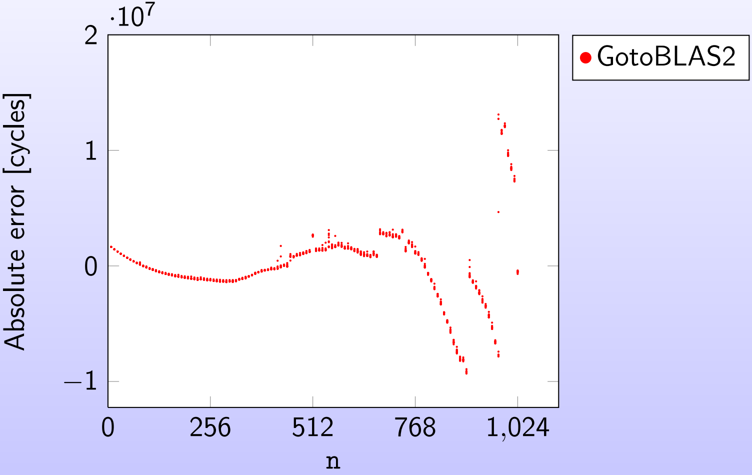
Size arguments

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



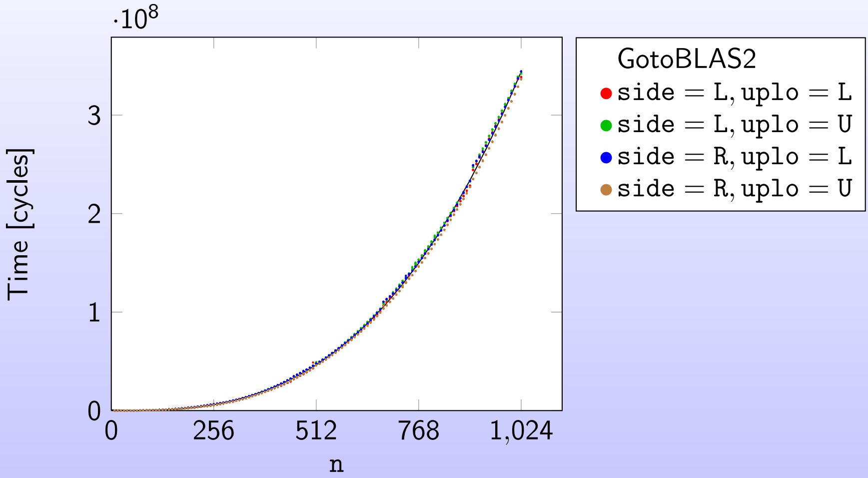
⇒ Piecewise Polynomials

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



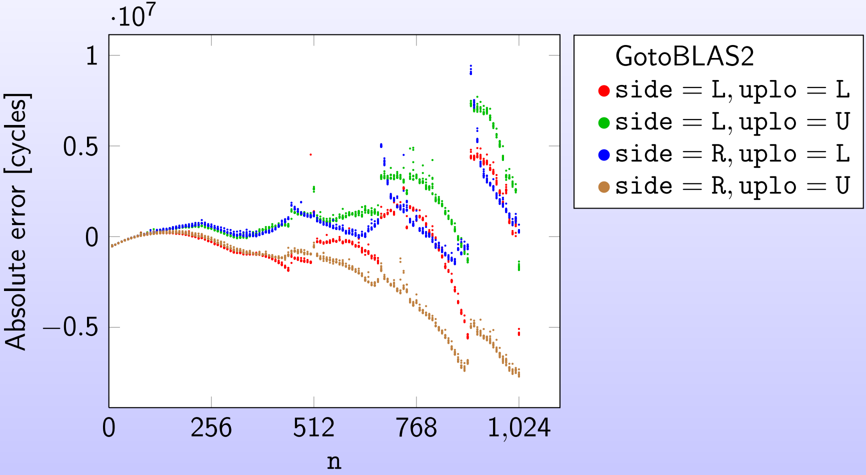
Flags

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



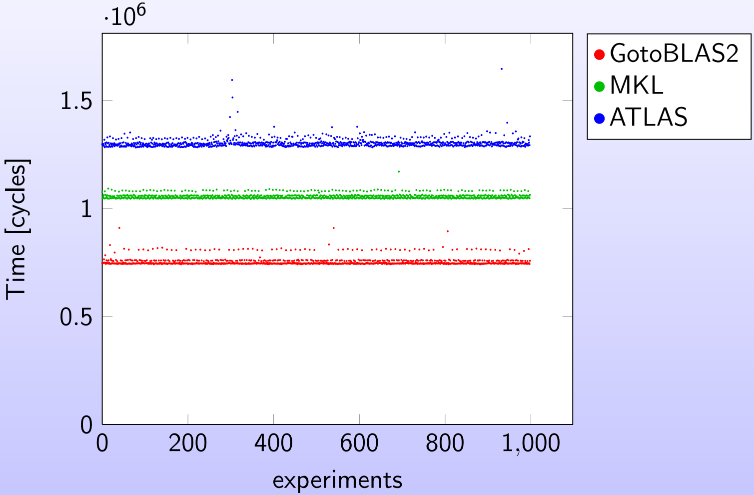
⇒ Independent models

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



Variability ⇒ statistical info

DGEMM



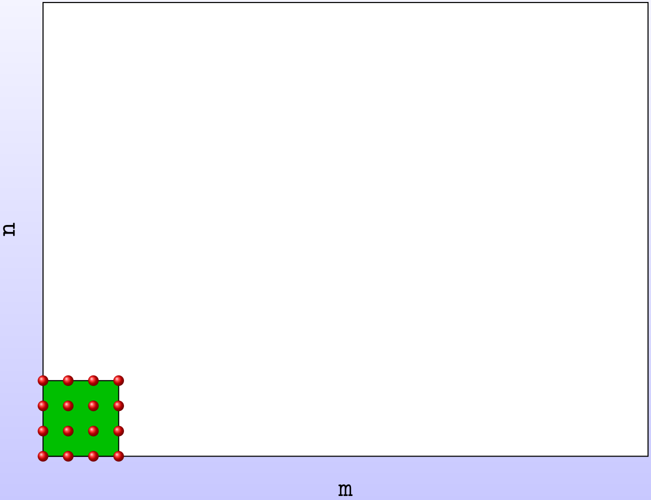
Building the models

- Two tools
  - Sampler
  - Modeler

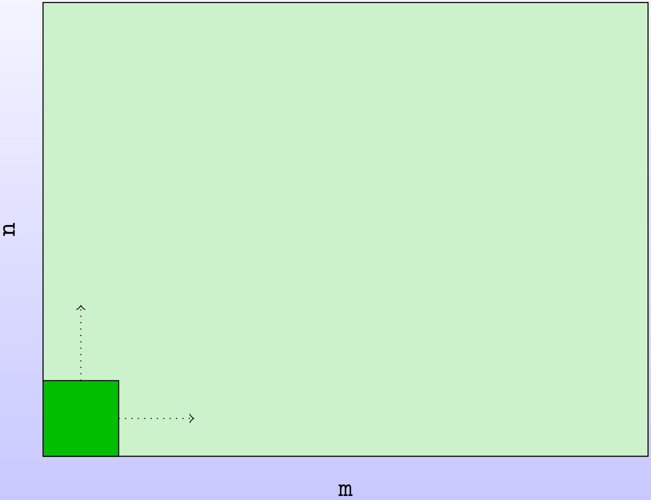
Building the models

- Two tools
  - Sampler
  - Modeler
- Two modeling strategies
  - Expansion
  - Adaptive refinement

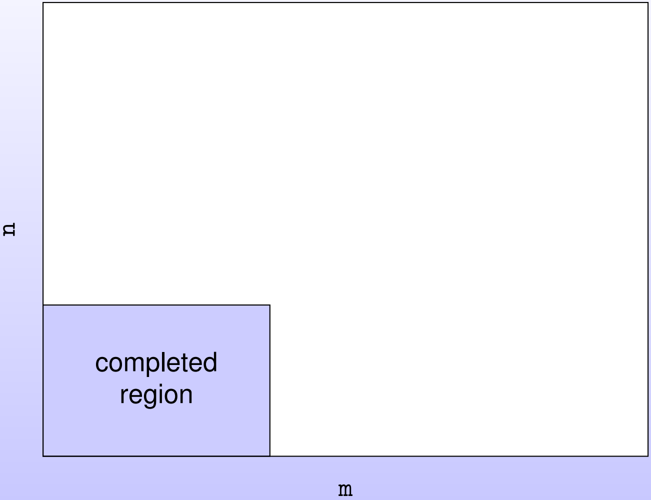
Model Expansion



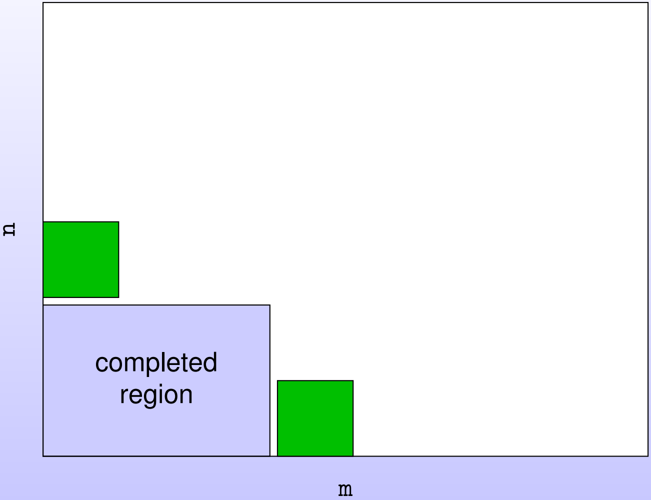
Model Expansion



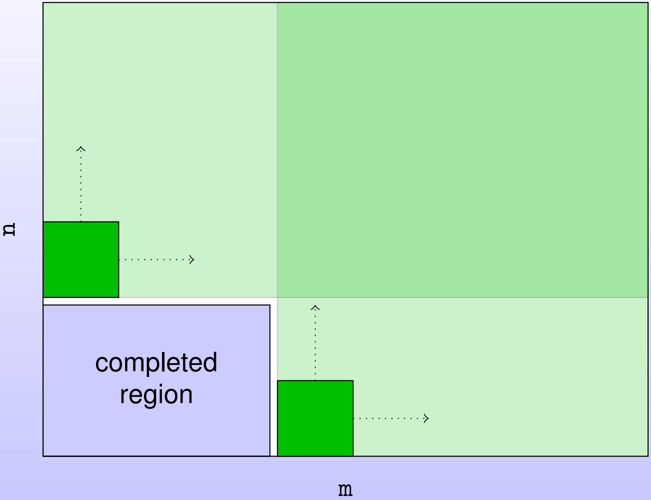
Model Expansion



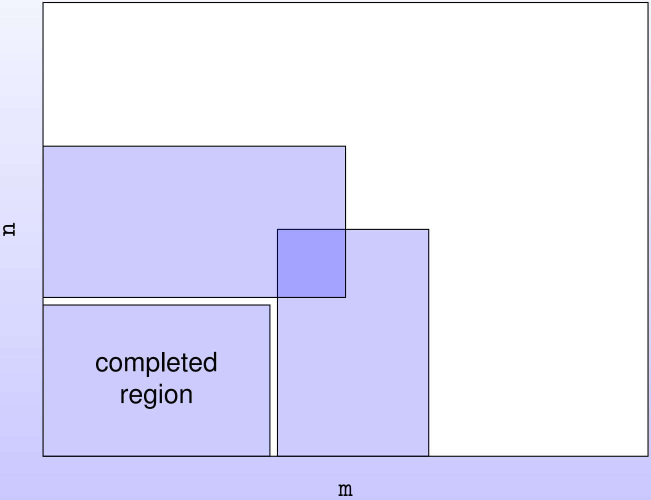
Model Expansion



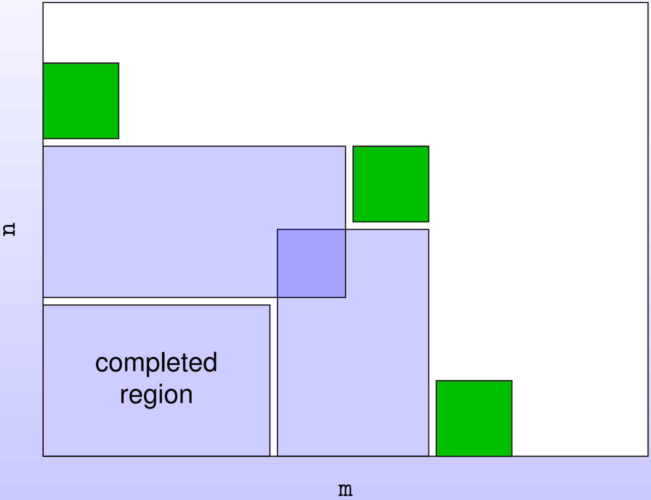
Model Expansion



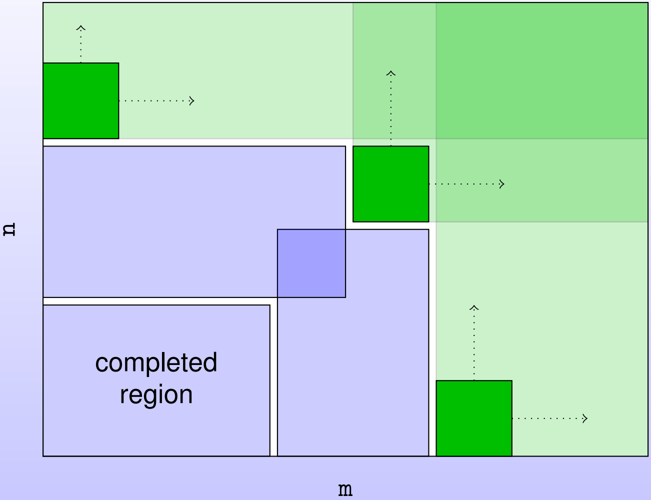
Model Expansion



Model Expansion

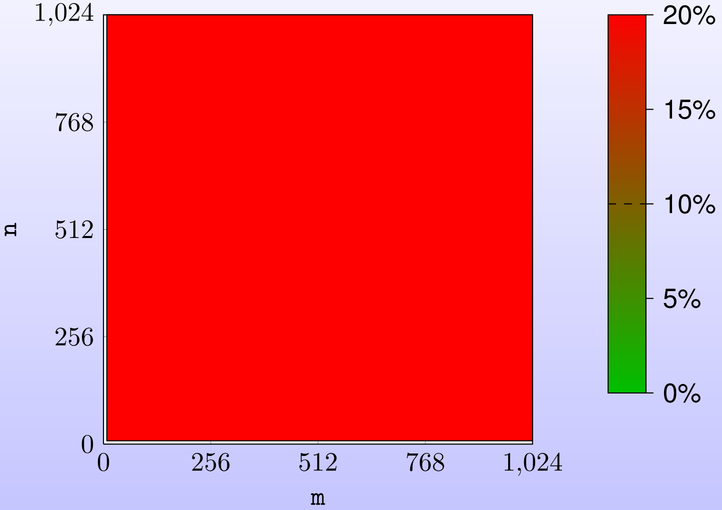


Model Expansion



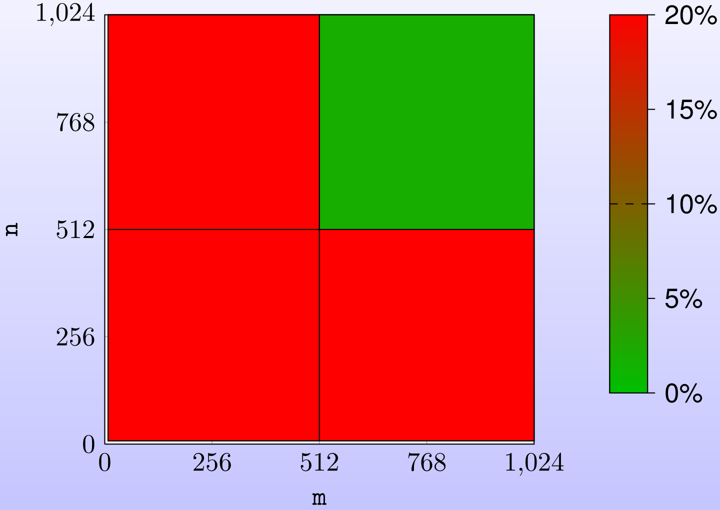
Adaptive Refinement

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



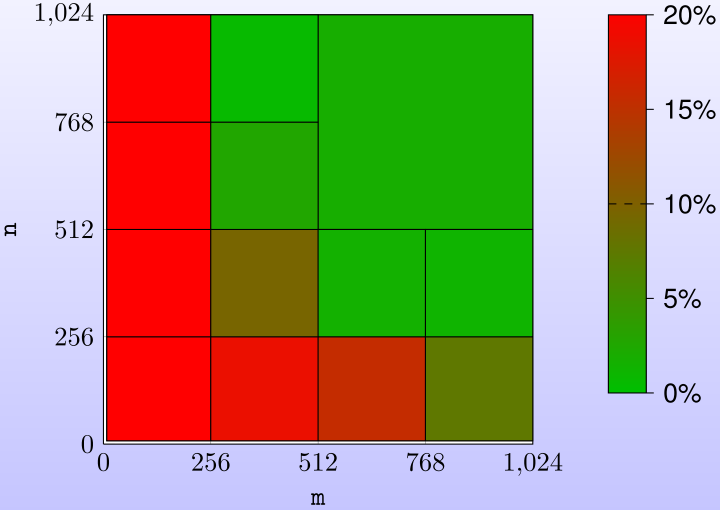
Adaptive Refinement

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



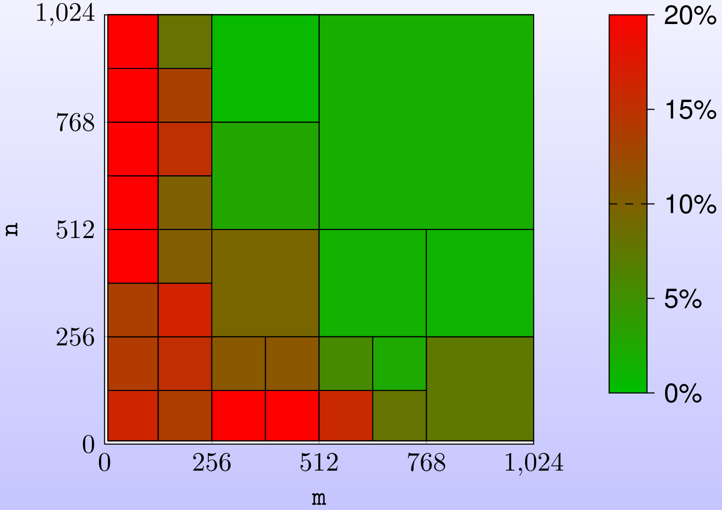
Adaptive Refinement

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



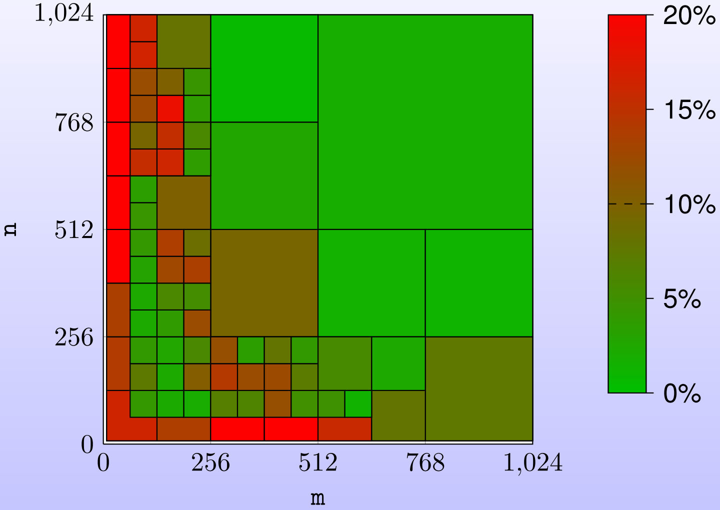
Adaptive Refinement

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



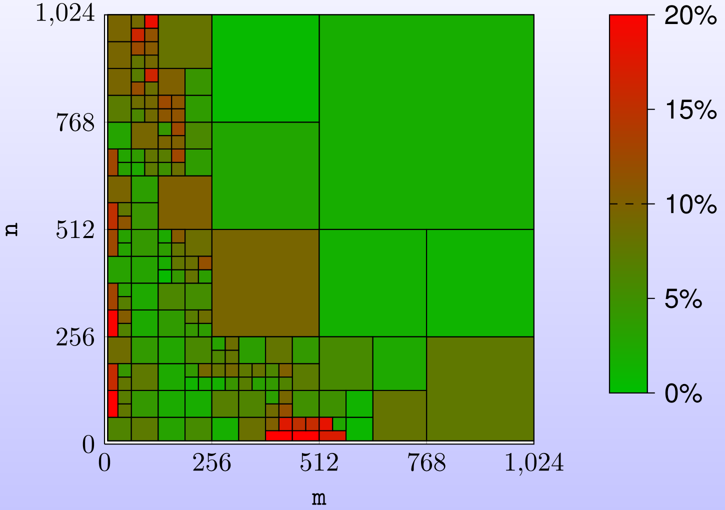
Adaptive Refinement

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



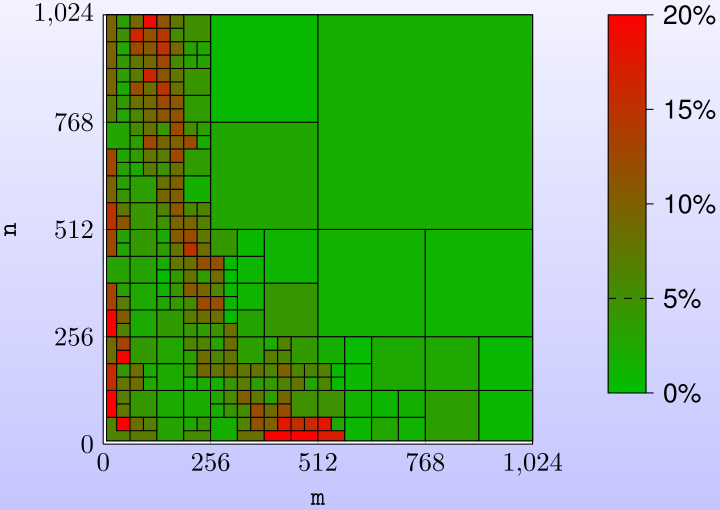
Adaptive Refinement

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



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 $L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$

where L_{TL} is 0×0

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

Repartition

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline L_{10} & L_{11} & 0 \\ \hline L_{20} & L_{21} & L_{22} \end{array}\right)$$

where L_{11} is $b \times b$

$L_{10} := \texttt{TRMM}(L_{10}, L_{00})$

$L_{10} := \texttt{TRSM}(-L_{11} L_{10})$

$L_{11} := \texttt{trinv}(L_{11})$

Continue

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline L_{10} & L_{11} & 0 \\ \hline L_{20} & L_{21} & L_{22} \end{array}\right)$$

endwhile

From algorithm to prediction

TriInv_1('L',300,A,300,100)

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$

where L_{TL} is 0×0

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

Repartition

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline L_{10} & L_{11} & 0 \\ \hline L_{20} & L_{21} & L_{22} \end{array}\right)$$

where L_{11} is $b \times b$

$L_{10} := \texttt{TRMM}(L_{10}, L_{00})$

$L_{10} := \texttt{TRSM}(-L_{11} L_{10})$

$L_{11} := \texttt{trinv}(L_{11})$

Continue

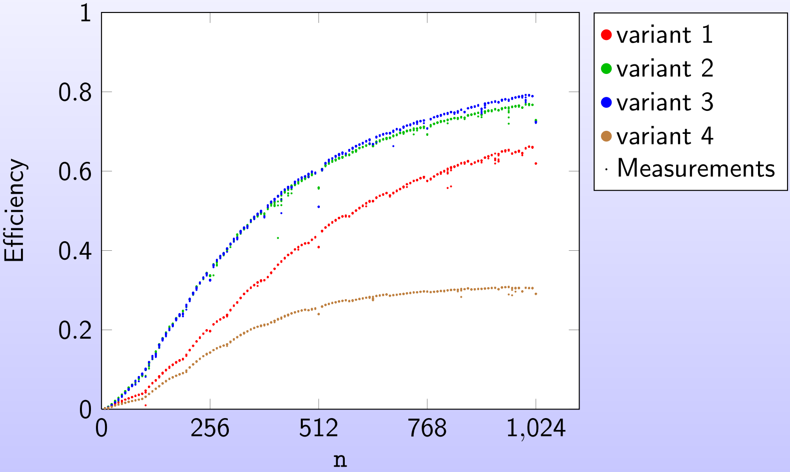
$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline L_{10} & L_{11} & 0 \\ \hline L_{20} & L_{21} & L_{22} \end{array}\right)$$

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)
```

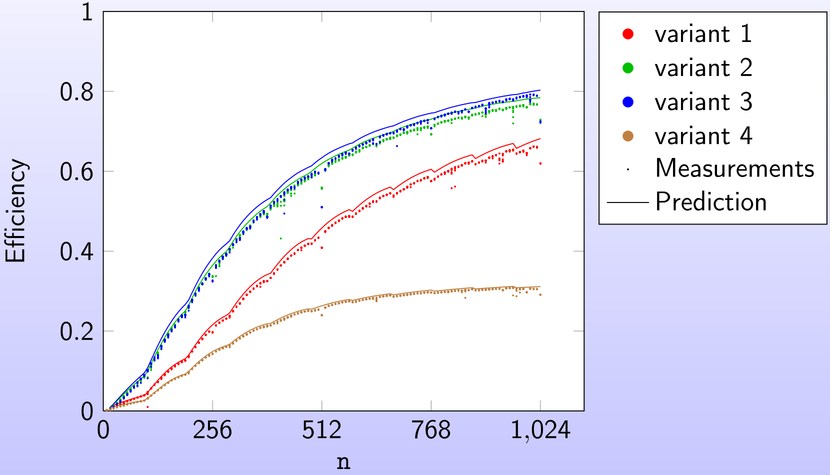
Efficiency

$X := L^{-1}$



Ranking

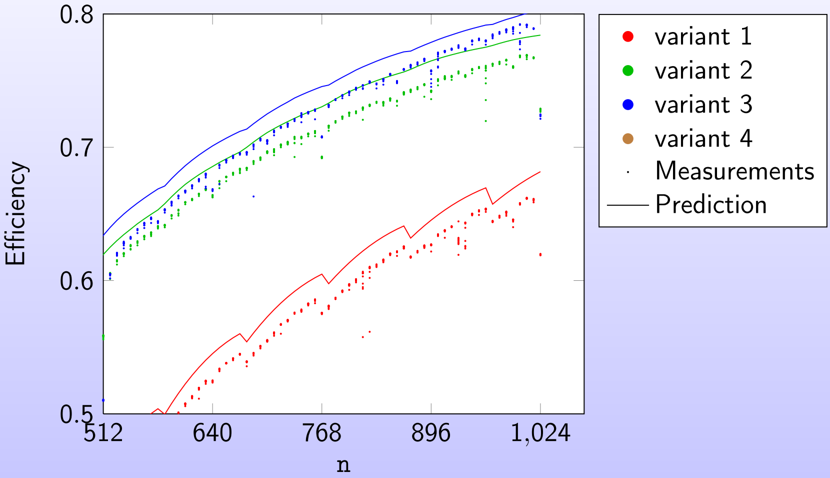
$X := L^{-1}$



Efficiency
1
0.8
0.6
0.4
0.2
0
0
256
512
768
1,024
n
variant 1
variant 2
variant 3
variant 4
Measurements
Prediction

Zoom

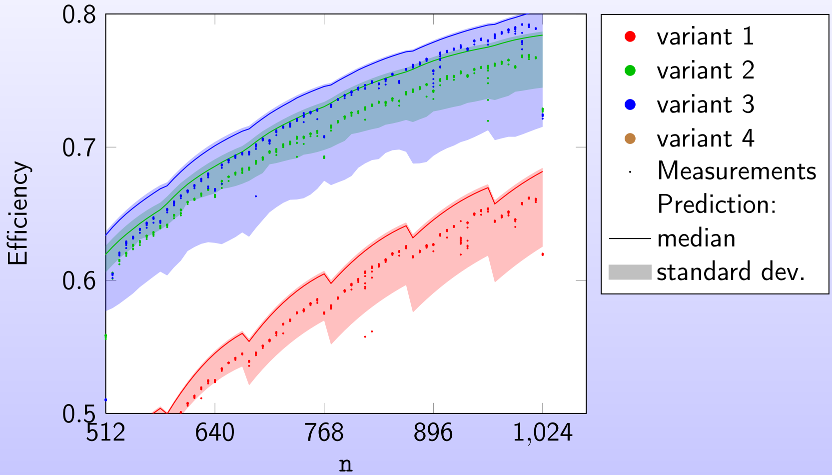
$X := L^{-1}$



0.8
0.7
0.6
0.5
Efficiency
512
640
768
896
1,024
n
variant 1
variant 2
variant 3
variant 4
Measurements
Prediction

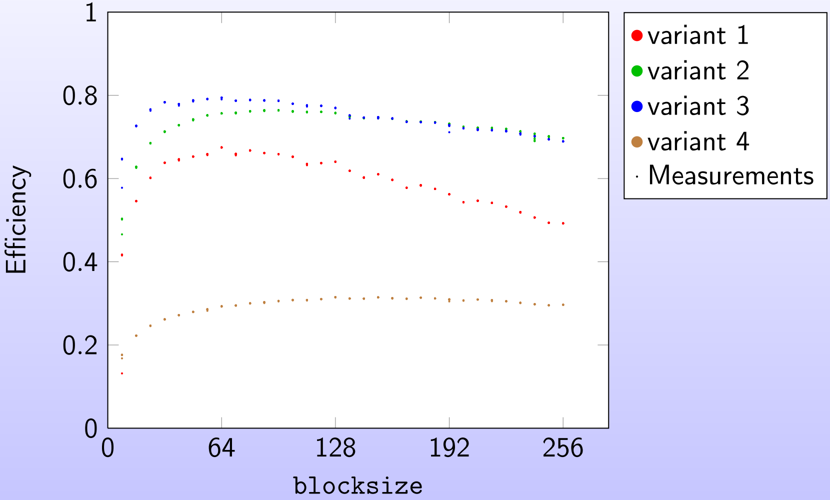
Statistics

$X := L^{-1}$



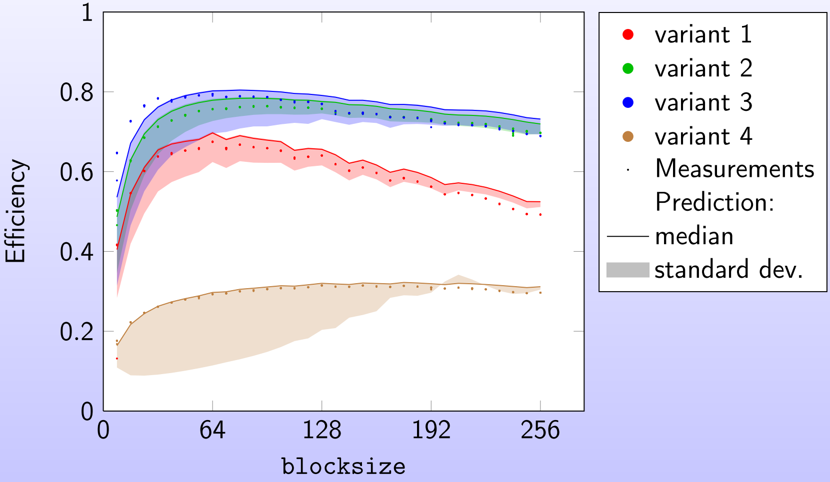
Tuning: block size

$X := L^{-1}$



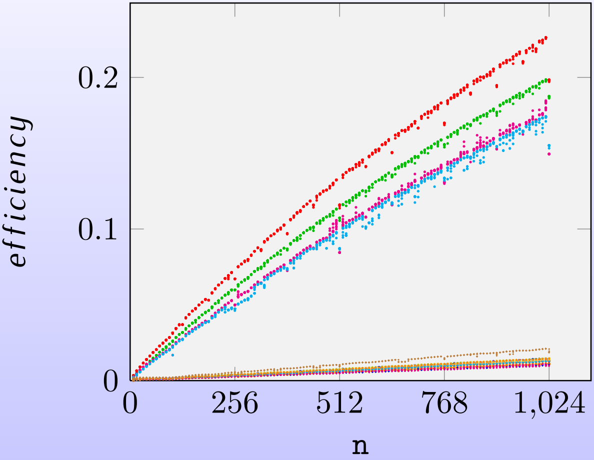
Tuning: block size

$X := L^{-1}$



Sylvester equation – 16 variants

$AX + XB = C$



Sylvester equation – 16 variants

$AX + XB = C$

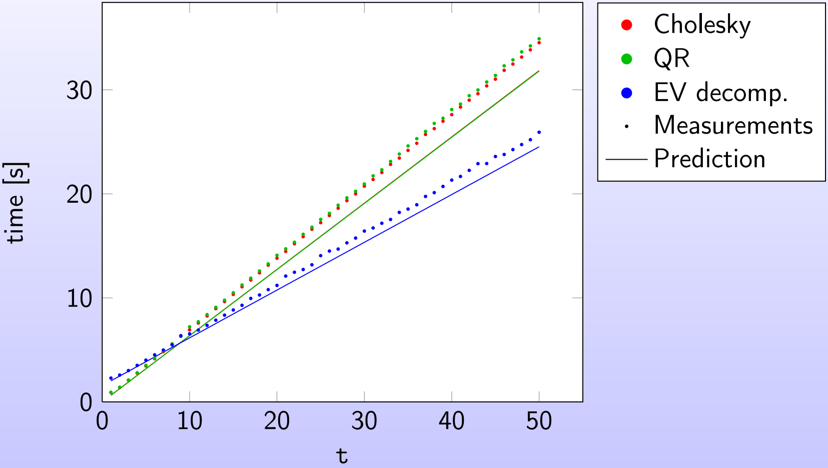
Variant	Efficiency predicted	Efficiency measured
Var-1	27.03%	24.04%
Var-2	22.52%	21.07%
Var-5	15.51%	18.82%
Var-6	13.72%	18.51%
Var-16	1.79%	2.21%
Var-3	1.52%	1.52%
Var-4	1.50%	1.45%
Var-8	1.49%	1.37%
Var-10	1.43%	1.53%
Var-15	1.43%	1.52%
Var-9	1.40%	1.48%
Var-14	1.34%	1.33%
Var-12	1.29%	1.43%
Var-7	1.06%	1.16%
Var-11	1.04%	1.07%
Var-13	1.01%	1.01%

GWAS

$b := (X^T M^{-1} X)^{-1} X^T M^{-1} y$

GWAS

$b := (X^T M^{-1} X)^{-1} X^T M^{-1} y$



Modeling through sampling

Wishlist

Modeling through sampling

Wishlist

- Speed ✓

Modeling through sampling

Wishlist

- Speed ✓
  - No direct execution of the algorithm ✓

Modeling through sampling

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 ✓ $\Rightarrow$ accurate ranking

Modeling through sampling

Wishlist

- Speed ✓
  - No direct execution of the algorithm ✓
  - Possibly no execution at all ✖
- Accuracy ✓ $\Rightarrow$ accurate ranking
- Automation ✓

Conclusions

Ranking of algorithms

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

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, . . .

Deutsche Forschungsgemeinschaft DFG

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