Compiling Linear Algebra Expressions to High-Performance Code

Henrik Barthels, **Paolo Bientinesi**
Aachen Institute for Computational Engineering Science
RWTH Aachen University

8th International Workshop on Parallel Symbolic Computation
July 23, 2017
Kaiserslautern, Germany

$$x := A(B^T B + A^T R^T \Lambda R A)^{-1} B^T B A^{-1} y$$

exponential transient excision

$$q := u - U(P^T U)^{-1} P^T u$$

reduced basis methodology for parametric PDEs

$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

probabilistic Nordsieck method for ODEs

$$E := Q^{-1} U (I + U^T Q^{-1} U)^{-1} U^T$$

L1-norm minimization on manifolds

$$\begin{cases} x_{k|k-1} &= F x_{k-1|k-1} + Bu \\ P_{k|k-1} &= F P_{k-1|k-1} F^T + Q \\ x_{k|k} &= x_{k|k-1} + P_{k|k-1} H^T \times (H P_{k|k-1} H^T + R)^{-1} (z_k - H x_{k|k-1}) \\ P_{k|k} &= P_{k|k-1} - P_{k|k-1} H^T \times (H P_{k|k-1} H^T + R)^{-1} H P_{k|k-1} \end{cases}$$

Kalman filter

$$x := A(B^TB + A^TR^T\Lambda RA)^{-1}B^TBA^{-1}y$$

exponential transient excision

$$q := u - U(P^TU)^{-1}P^Tu$$

reduced basis methodology for parametric PDEs

$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T(HC_\dagger H^T)^{-1} \end{cases}$$

probabilistic Nordsieck method for ODEs

how to EFFICIENTLY compute these expressions?

$$E := Q^{-1}U(I + U^TQ^{-1}U)^{-1}U^T$$

L1-norm minimization on manifolds

$$\begin{cases} x_{k|k-1} &= Fx_{k-1|k-1} + Bu \\ P_{k|k-1} &= FP_{k-1|k-1}F^T + Q \\ x_{k|k} &= x_{k|k-1} + P_{k|k-1}H^T \times (HP_{k|k-1}H^T + R)^{-1}(z_k - Hx_{k|k-1}) \\ P_{k|k} &= P_{k|k-1} - P_{k|k-1}H^T \times (HP_{k|k-1}H^T + R)^{-1}HP_{k|k-1} \end{cases}$$

Kalman filter

$$x := A(B^T B + A^T R^T \Lambda R A)^{-1} B^T B A^{-1} y$$

exponential transient excision

$$q := u - U(P^T U)^{-1} P^T u$$

reduced basis methodology for parametric PDEs

$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

Role of symbolic calculations in high-performance numerical code

probabilistic Nordsieck method for ODEs

$$E := Q^{-1} U (I + U^T Q^{-1} U)^{-1} U^T$$

L1-norm minimization on manifolds

$$\begin{cases} x_{k|k-1} &= F x_{k-1|k-1} + Bu \\ P_{k|k-1} &= F P_{k-1|k-1} F^T + Q \\ x_{k|k} &= x_{k|k-1} + P_{k|k-1} H^T \times (H P_{k|k-1} H^T + R)^{-1} (z_k - H x_{k|k-1}) \\ P_{k|k} &= P_{k|k-1} - P_{k|k-1} H^T \times (H P_{k|k-1} H^T + R)^{-1} H P_{k|k-1} \end{cases}$$

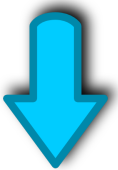
Kalman filter

$$x := A(B^T B + A^T R^T \Lambda R A)^{-1} B^T B A^{-1} y$$

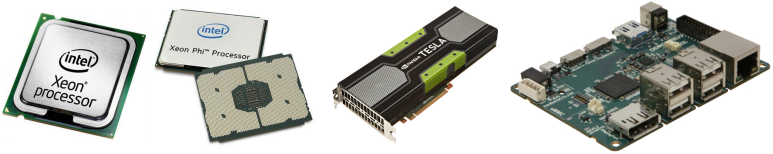
$$\begin{cases} C_\dagger := P C P^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

$$E := Q^{-1} U (I + U^T Q^{-1} U)^{-1} U^T$$

...



?



MUL ADD

MOV

MOVAPD

VFMADDPD ...

$$x := A(B^T B + A^T R^T \Lambda R A)^{-1} B^T B A^{-1} y$$

$$\begin{cases} C_\dagger := P C P^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

$$E := Q^{-1} U (I + U^T Q^{-1} U)^{-1} U^T \qquad \ldots$$

$$y := \alpha x + y \qquad LU = A \qquad \cdots \qquad C := \alpha A B + \beta C$$

$$X := A^{-1} B \qquad C := A B^T + B A^T + C \qquad X := L^{-1} M L^{-T} \qquad QR = A$$

LINPACK **BLAS** **LAPACK** **...**

$$x := A(B^TB + A^TR^T\Lambda RA)^{-1}B^TBA^{-1}y$$

$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T(HC_\dagger H^T)^{-1} \end{cases}$$

$$E := Q^{-1}U(I + U^TQ^{-1}U)^{-1}U^T$$

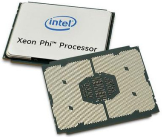
...

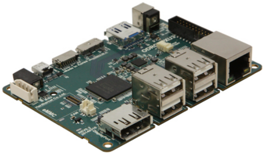
?

$y := \alpha x + y$ | $LU = A$ | $\cdots$ | $C := \alpha AB + \beta C$

$X := A^{-1}B$ | $C := AB^T + BA^T + C$ | $X := L^{-1}ML^{-T}$ | $QR = A$

LINPACK **BLAS** **LAPACK** **...**



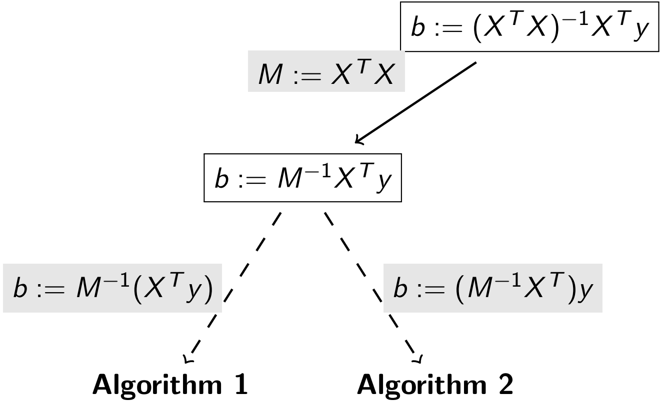


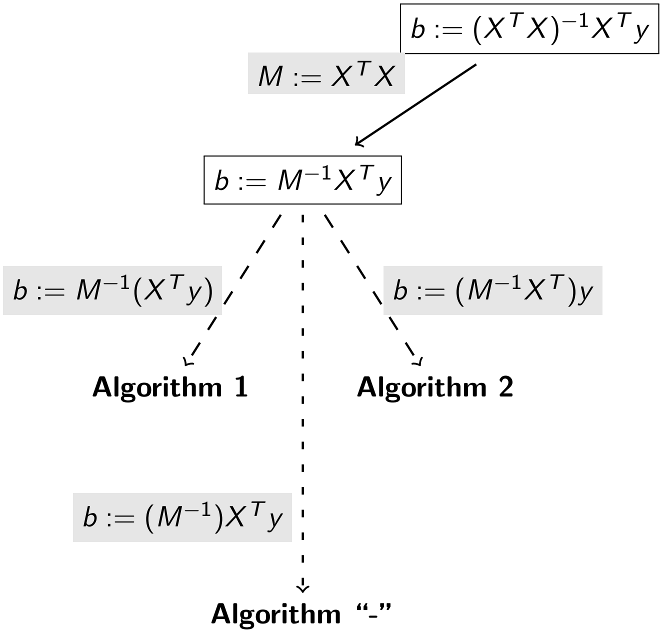
$$\boxed{b := (X^T X)^{-1} X^T y}$$

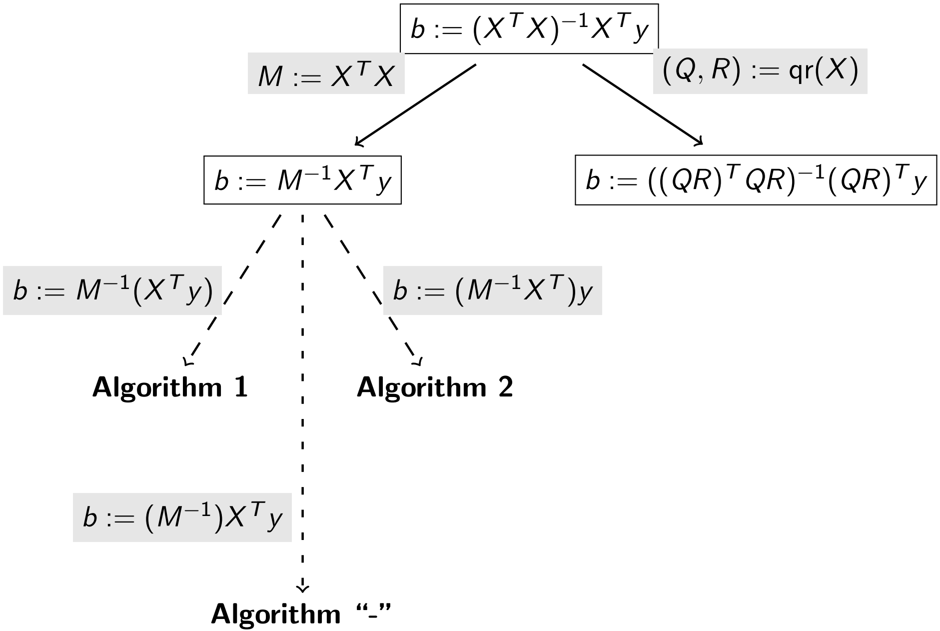
$$b := (X^T X)^{-1} X^T y$$

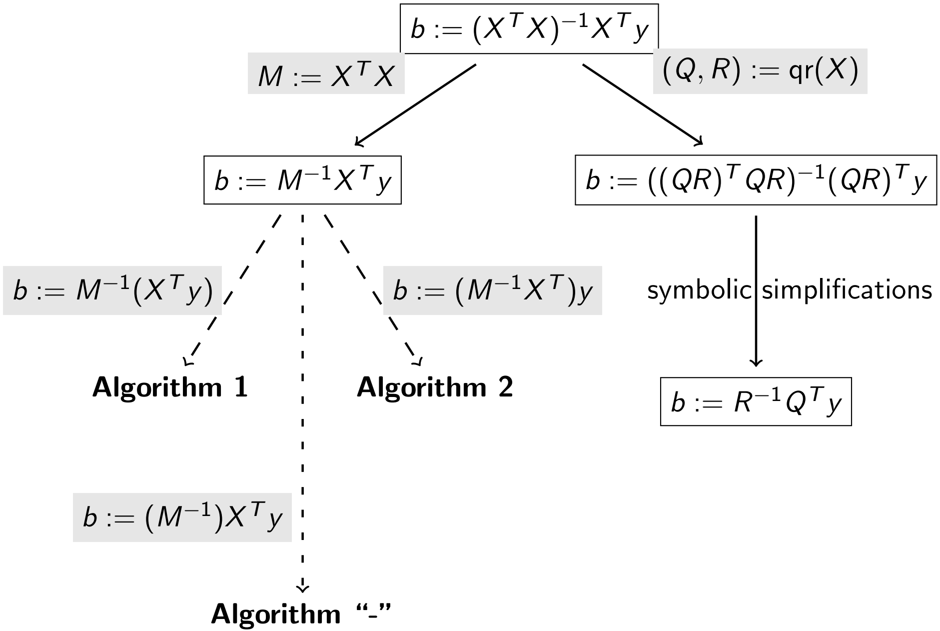
$$M := X^T X$$

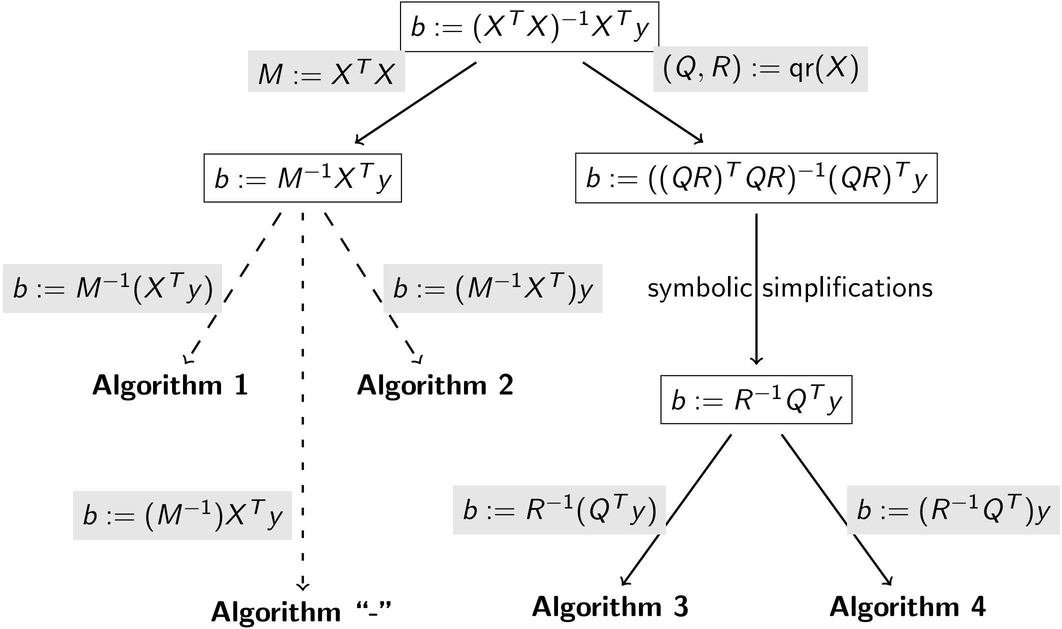
$$b := M^{-1} X^T y$$

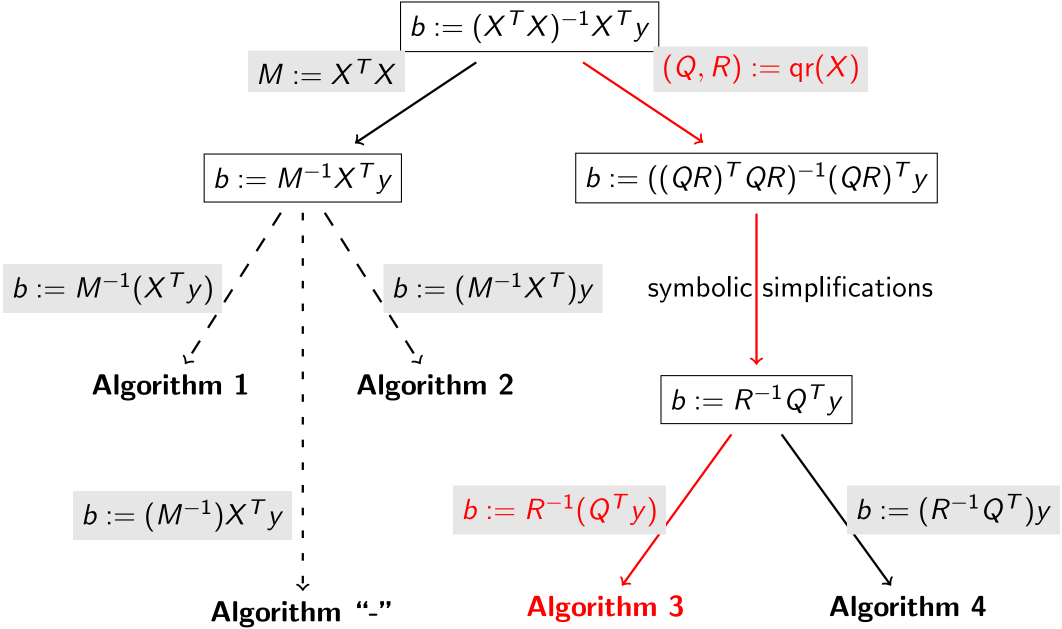

$b := (X^T X)^{-1} X^T y$
$M := X^T X$
$b := M^{-1} X^T y$
$b := M^{-1}(X^T y)$
$b := (M^{-1} X^T) y$
Algorithm 1
Algorithm 2


$b := (X^T X)^{-1} X^T y$
$M := X^T X$
$b := M^{-1} X^T y$
$b := M^{-1}(X^T y)$
$b := (M^{-1} X^T) y$
Algorithm 1
Algorithm 2
$b := (M^{-1}) X^T y$
Algorithm "-"


$b := (X^T X)^{-1} X^T y$
$M := X^T X$
$(Q, R) := \text{qr}(X)$
$b := M^{-1} X^T y$
$b := ((QR)^T QR)^{-1} (QR)^T y$
$b := M^{-1}(X^T y)$
$b := (M^{-1} X^T) y$
Algorithm 1
Algorithm 2
$b := (M^{-1}) X^T y$
Algorithm "-"


$b := (X^T X)^{-1} X^T y$
$M := X^T X$
$(Q, R) := \mathrm{qr}(X)$
$b := M^{-1} X^T y$
$b := ((QR)^T QR)^{-1} (QR)^T y$
$b := M^{-1}(X^T y)$
$b := (M^{-1} X^T) y$
symbolic simplifications
Algorithm 1
Algorithm 2
$b := R^{-1} Q^T y$
$b := (M^{-1}) X^T y$
Algorithm "-"


$b := (X^T X)^{-1} X^T y$
$M := X^T X$
$(Q, R) := \mathrm{qr}(X)$
$b := M^{-1} X^T y$
$b := ((QR)^T QR)^{-1} (QR)^T y$
$b := M^{-1}(X^T y)$
$b := (M^{-1} X^T) y$
symbolic simplifications
Algorithm 1
Algorithm 2
$b := R^{-1} Q^T y$
$b := (M^{-1}) X^T y$
$b := R^{-1}(Q^T y)$
$b := (R^{-1} Q^T) y$
Algorithm “-”
Algorithm 3
Algorithm 4


$b := (X^TX)^{-1}X^Ty$
$M := X^TX$
$(Q, R) := \mathrm{qr}(X)$
$b := M^{-1}X^Ty$
$b := ((QR)^TQR)^{-1}(QR)^Ty$
$b := M^{-1}(X^Ty)$
$b := (M^{-1}X^T)y$
symbolic simplifications
Algorithm 1
Algorithm 2
$b := R^{-1}Q^Ty$
$b := (M^{-1})X^Ty$
$b := R^{-1}(Q^Ty)$
$b := (R^{-1}Q^T)y$
Algorithm "-"
Algorithm 3
Algorithm 4

Linear Algebra Mapping Problem (LAMP)

Linear Algebra Mapping Problem (LAMP)

- $\mathcal{E}$: a list of assignments $var_i := EXP_i$

Linear Algebra Mapping Problem (LAMP)

- $\mathcal{E}$: a list of assignments $var_i := EXP_i$
- $\mathcal{K}$: a list of available computational kernels (BLAS, LAPACK, ...)

Linear Algebra Mapping Problem (LAMP)

- $\mathcal{E}$: a list of assignments $var_i := EXP_i$
- $\mathcal{K}$: a list of available computational kernels (BLAS, LAPACK, ...)
- $\mathcal{M}$: a metric (FLOPs, data movement, stability, time)

Linear Algebra Mapping Problem (LAMP)

- $\mathcal{E}$: a list of assignments $var_i := EXP_i$
- $\mathcal{K}$: a list of available computational kernels (BLAS, LAPACK, ...)
- $\mathcal{M}$: a metric (FLOPs, data movement, stability, time)

LAMP:

Find a decomposition of the expressions $\mathcal{E}$ in terms of the kernels $\mathcal{K}$, optimal according to the metric $\mathcal{M}$.

Linear Algebra Mapping Problem (LAMP)

- $\mathcal{E}$: a list of assignments $var_i := EXP_i$
- $\mathcal{K}$: a list of available computational kernels (BLAS, LAPACK, ...)
- $\mathcal{M}$: a metric (FLOPs, data movement, stability, time)

LAMP:

Find a decomposition of the expressions $\mathcal{E}$ in terms of the kernels $\mathcal{K}$, optimal according to the metric $\mathcal{M}$.

- Find a decomposition $\rightarrow$ easy
- Achieve optimality $\rightarrow$ NP complete

LAMP is everywhere

High-level languages

- Matlab
- R
- Julia
- Mathematica
- . . .

LAMP is everywhere

High-level languages

- Matlab
- R
- Julia
- Mathematica
- ...

Libraries

- Armadillo
- Blaze
- Blitz
- Eigen
- ...
- NumPy

LAMP is everywhere

High-level languages

- Matlab
- R
- Julia
- Mathematica
- . . .

Libraries

- Armadillo
- Blaze
- Blitz
- Eigen
- . . .
- NumPy

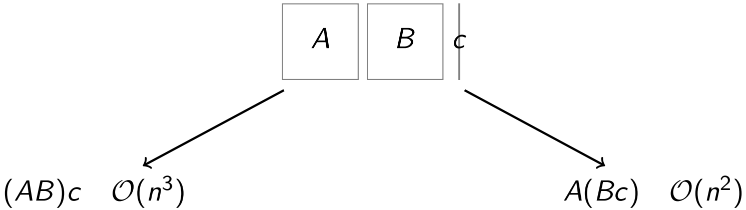
human productivity vs. machine efficiency

Challenges and State of the Art

- **Parenthesisation**

Challenges and State of the Art

- **Parenthesisation**



$\Rightarrow$ Matrix Chain Algorithm

Challenges and State of the Art

- **Parenthesisation**

  In practice:

  - Unary operators: transposition, inversion $(X := AB^T C^{-T} D + \dots)$
  - Overlapping kernels (e.g., $L \leftarrow L^{-1}$, $X = A^{-1}B$)
  - Decompositions (e.g., $A \rightarrow Q^T DQ$, $A \rightarrow LU$)
  - Properties & specialized kernels (`GEMM`, `TRMM`, `SYMM`, ...)

$\Rightarrow$ **Generalized** Matrix Chain Algorithm

Challenges and State of the Art

- **Metric:** FLOPs vs. execution time

$$\underset{\mathcal{A}}{\operatorname{argmin}} \left(\text{ FLOPs}(\mathcal{A}) \right) \neq \underset{\mathcal{A}}{\operatorname{argmin}} \left(\text{ time}(\mathcal{A}) \right)$$

Challenges and State of the Art

- **Metric:** FLOPs vs. execution time

$$\underset{\mathcal{A}}{\text{argmin}}\,(\ \text{FLOPs}(\mathcal{A})\ ) \neq \underset{\mathcal{A}}{\text{argmin}}\,(\ \text{time}(\mathcal{A})\ )$$

Locality in space and time $\equiv$ memory accesses, caching, prefetching, ...

$\Rightarrow$ **Performance prediction** (efficiency)

Challenges and State of the Art

- **Parallelism?** Libraries, explicit multi-threading, runtime, hybrid?

$$X := A((B^T C^{-T})D) \quad \text{vs.} \quad X := (AB^T)(C^{-T}D) \quad \text{vs.} \quad \dots$$

Challenges and State of the Art

- **Parallelism?** Libraries, explicit multi-threading, runtime, hybrid?

$$X := A((B^T C^{-T})D) \quad \text{vs.} \quad X := (AB^T)(C^{-T} D) \quad \text{vs.} \quad \dots$$

$\Rightarrow$ **Performance prediction** (efficiency, scalability)

Challenges and State of the Art

- **Linear algebra knowledge**: operators, identities, theorems

- Distributivity, commutativity, partitionings, ...
- $((QR)^T QR)^{-1}(QR)^T y \;\rightarrow\; (R^T Q^T QR)^{-1} R^T Q^T y \;\rightarrow\; R^{-1} R^{-T} R^T Q^T y \;\rightarrow\; R^{-1} Q^T y$
- $\text{SPD}(A) \rightarrow \text{SPD}(A_{BR} - A_{BL} A_{TL}^{-1} A_{BL}^T)$ Schur complement
- $\cdots$

Challenges and State of the Art

- **Linear algebra knowledge**: operators, identities, theorems

- Distributivity, commutativity, partitionings, . . .
- $((QR)^T QR)^{-1}(QR)^T y \;\rightarrow\; (R^T Q^T QR)^{-1} R^T Q^T y \;\rightarrow\; R^{-1} R^{-T} R^T Q^T y \;\rightarrow\; R^{-1} Q^T y$
- $\text{SPD}(A) \rightarrow \text{SPD}(A_{BR} - A_{BL} A_{TL}^{-1} A_{BL}^T)$ Schur complement
- $\cdots$

$\Rightarrow$ "**Knowledge base**" – expert system – pattern matching

Challenges and State of the Art

- **Inference of properties**

$$E := L_1 * U^T * L_2 \qquad \text{triangular}(E)\ ?$$

$$E := Q^{-1}U(I + U^T Q^{-1} U)^{-1} U^T \qquad \text{properties}(I + U^T Q^{-1} U)\ ?$$

$$\lambda(A, B) \wedge \begin{cases} \text{symm}(A) \\ \text{SPD}(B) \end{cases} \rightarrow \lambda(L^{-T} A L^{-1}) \qquad \text{symmetric}(L^{-T} A L^{-1})\ ?$$

Challenges and State of the Art

- **Inference of properties**

$E := L_1 * U^T * L_2$	triangular(E) ?
$E := Q^{-1}U(I + U^T Q^{-1} U)^{-1} U^T$	properties($I + U^T Q^{-1} U$) ?
$\lambda(A, B) \wedge \begin{cases} \text{symm}(A) \\ \text{SPD}(B) \end{cases} \rightarrow \lambda(L^{-T} A L^{-1})$	symmetric($L^{-T} A L^{-1}$) ?

$\Rightarrow$ **Symbolic analysis** – pattern matching

Challenges and State of the Art

- **Common subexpressions**

$$\begin{cases} X := AB^{-T}C \\ Y := B^{-1}A^{T}D \end{cases} \quad \rightarrow \quad \begin{cases} Z := AB^{-T} \\ X := ZC \\ Y := Z^{T}D \end{cases}$$

Challenges and State of the Art

- **Common subexpressions**

$$\begin{cases} X := AB^{-T}C \\ Y := B^{-1}A^{T}D \end{cases} \quad \rightarrow \quad \begin{cases} Z := AB^{-T} \\ X := ZC \\ Y := Z^{T}D \end{cases}$$

$\Rightarrow$ **Pattern matching**

Linnea – Linear algebra compiler

Example: $w := AB^{-1}c, \quad \mathsf{SPD}(B)$

Linnea – Linear algebra compiler

Example: $w := AB^{-1}c, \quad \mathsf{SPD}(B)$

Naive

```
w = A*inv(B)*c
```

Recommended

```
w = A*(B\c)
```

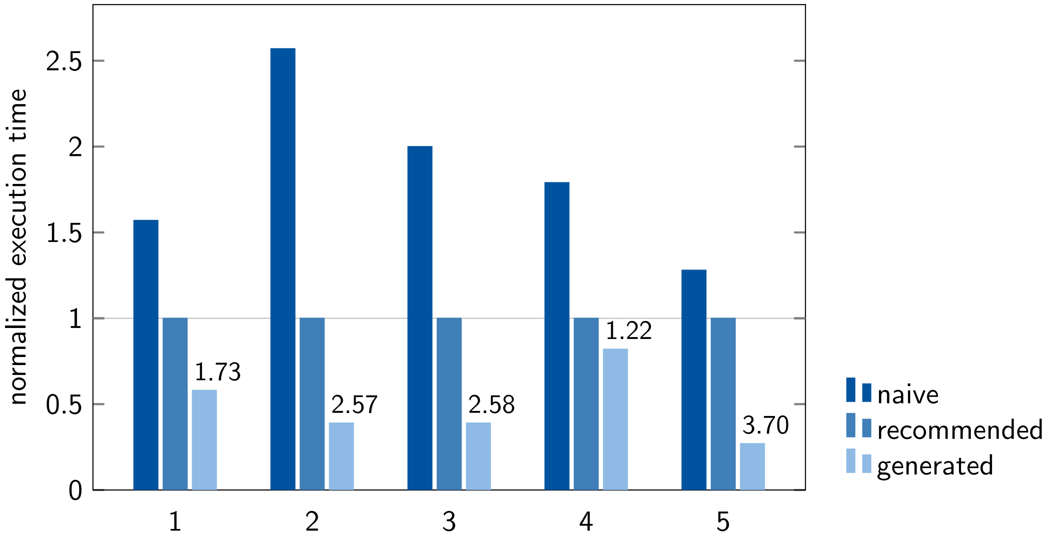
Generated

```
ml0 = A; ml1 = B; ml2 = c;
potrf!('L', ml1)
trsv!('L', 'N', 'N', ml1, ml2)
trsv!('L', 'T', 'N', ml1, ml2)
ml3 = Array{Float64}(10)
gemv!('N', 1.0, ml0, ml2, 0.0, ml3)
w = ml3
```

Experiments

#	Example	
1	$b := (X^T X)^{-1} X^T y$	FullRank(X)
2	$b := (X^T M^{-1} X)^{-1} X^T M^{-1} y$	SPD(M), FullRank(X)
3	$W := A^{-1} BCD^{-T} EF$	LowTri(A), UppTri(D, E)
4	$\begin{cases} X := AB^{-1}C \\ Y := DB^{-1}A^T \end{cases}$	SPD(B)
5	$x := W(A^T(AWA^T)^{-1}b - c)$	FullRank(A, W) Diag(W), Pos(W)

Performance results



Future Work

- **Linnea** as a compiler (off line) vs. **Linnea** as an interpreter (real time)
- Integration into languages and libraries
- Aforementioned challenges, and then some:
  sequences of operations, memory usage, tensors, ...
- **YOU**: What instances of LAMP do you encounter?
  How do you solve them? Please let me know.

(Initial) References

- **A Domain-specific Compiler for Linear Algebra Operations**,
  Diego Fabregat-Traver and Paolo Bientinesi
  Lecture Notes in Computer Science, Vol.7851, 2013.

- **Application-tailored Linear Algebra Algorithms: A Search-based Approach**,
  Diego Fabregat-Traver and Paolo Bientinesi,
  International Journal of High Performance Computing Applications, Vol.27(4), 2013.

- **The Matrix Chain Algorithm to Compile Linear Algebra Expressions**,
  Barthels and Paolo Bientinesi,
  DSLDI 2016, `https://arxiv.org/pdf/1611.05660`.

(Initial) References

- **A Domain-specific Compiler for Linear Algebra Operations**,
  Diego Fabregat-Traver and Paolo Bientinesi
  Lecture Notes in Computer Science, Vol.7851, 2013.

- **Application-tailored Linear Algebra Algorithms: A Search-based Approach**,
  Diego Fabregat-Traver and Paolo Bientinesi,
  International Journal of High Performance Computing Applications, Vol.27(4), 2013.

- **The Matrix Chain Algorithm to Compile Linear Algebra Expressions**,
  Barthels and Paolo Bientinesi,
  DSLDI 2016, `https://arxiv.org/pdf/1611.05660`.

Thank You!

Term

Function Symbols: f, g

Term

Function Symbols: f, g

Constant Symbols: a, b, c

Term

Function Symbols: f, g

Constant Symbols: a, b, c

Variables: $\boldsymbol{x}, \boldsymbol{y}$

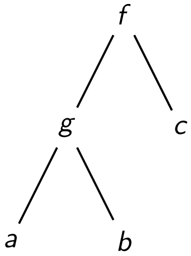
Term

Function Symbols: f, g

Constant Symbols: a, b, c

Variables: $\boldsymbol{x}, \boldsymbol{y}$

Examples: a, $f(\boldsymbol{x}, b)$,
$f(g(a, b), c)$



Pattern Matching

Pattern Matching

Definition: Find substitution $\sigma : \mathcal{X} \nrightarrow \mathcal{T}$ such that $\hat{\sigma}(\textit{pattern}) = \textit{subject}$

Pattern Matching

Definition: Find substitution $\sigma : \mathcal{X} \nrightarrow \mathcal{T}$ such that $\hat{\sigma}(\mathit{pattern}) = \mathit{subject}$

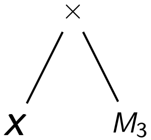
Example: Pattern: $f(\boldsymbol{x}, \boldsymbol{y})$

$$\Big\downarrow \begin{array}{l} \boldsymbol{x} \mapsto a \\ \boldsymbol{y} \mapsto g(b) \end{array}$$

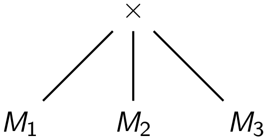
Subject: $f(a, g(b))$

Associativity

$\boldsymbol{X} \times M_3$

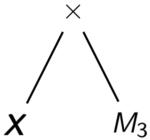


$M_1 \times M_2 \times M_3$

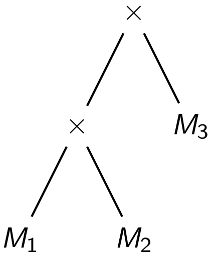


Associativity

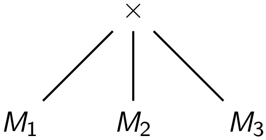
$\boldsymbol{X} \times M_3$



$(M_1 \times M_2) \times M_3$

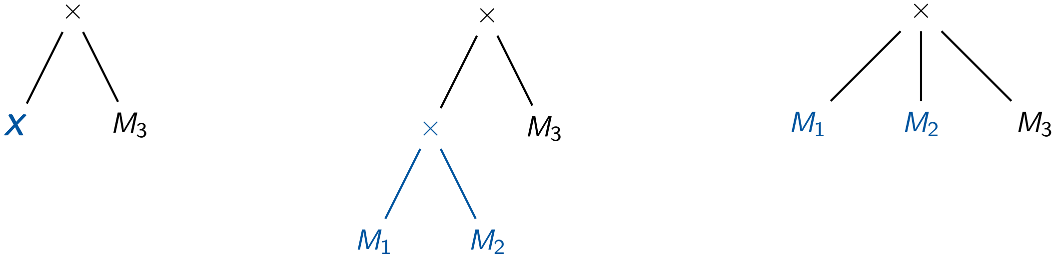


$M_1 \times M_2 \times M_3$

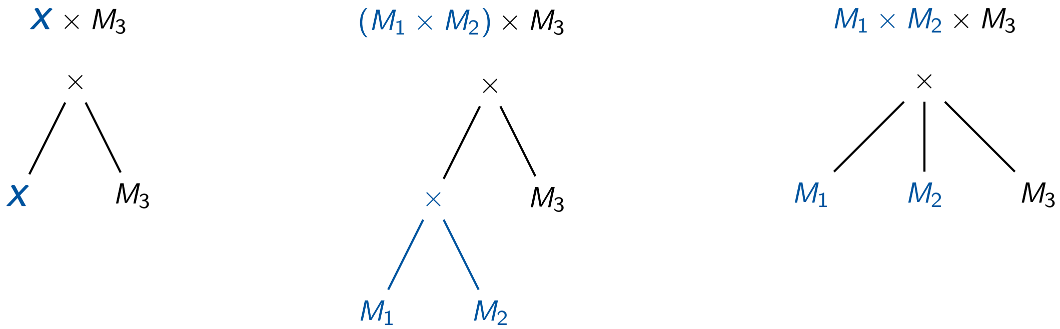


Associativity

$\boldsymbol{X} \times M_3$ $\quad$ $(M_1 \times M_2) \times M_3$ $\quad$ $M_1 \times M_2 \times M_3$



Associativity



$$\sigma = \{\boldsymbol{X} \mapsto (M_1 \times M_2)\}$$

Many-to-one Matching

- Many patterns, one subject

Many-to-one Matching

- Many patterns, one subject
- Speedup by simultaneous matching

Many-to-one Matching

- Many patterns, one subject
- Speedup by simultaneous matching
- Use similarity between patterns

Many-to-one Matching

- Many patterns, one subject
- Speedup by simultaneous matching
- Use similarity between patterns

- MatchPy
  https://github.com/hpac/matchpy