The Linear Algebra Mapping Problem (LAMP)

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

Householder Symposium '17
June 19, 2017
Blacksburg, Virginia

Deutsche Forschungsgemeinschaft DFG

HPAC High Performance and Automatic Computing

RWTH AACHEN UNIVERSITY

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

exponential transient excision

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

reduced basis methodology for parametric PDEs

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

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

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

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

how to EFFICIENTLY compute these expressions?

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

L1-norm minimization on manifolds

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

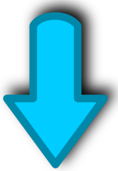
Kalman filter

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

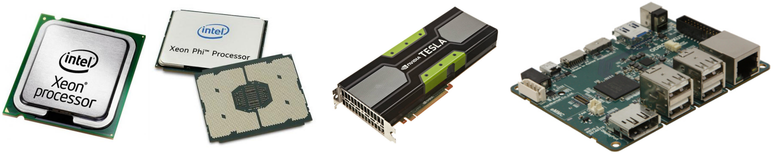
$$\begin{cases} C_\dagger := P C P^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

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

...



?



MUL ADD

MOV

MOVAPD

VFMADDPD ...

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

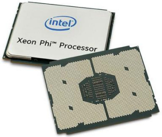
$$\begin{cases} C_\dagger := P C P^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

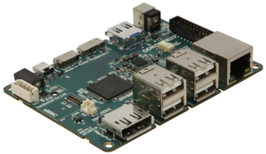
$$E := Q^{-1} U (I + U^T Q^{-1} U)^{-1} U^T \quad \ldots$$

$$y := \alpha x + y \qquad \{L, U\} := LU(A) \qquad C := \alpha A B + \beta C$$

$$L := L^{-1} \qquad C := A B^T + B A^T + C \quad \ldots$$

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





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

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

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

...

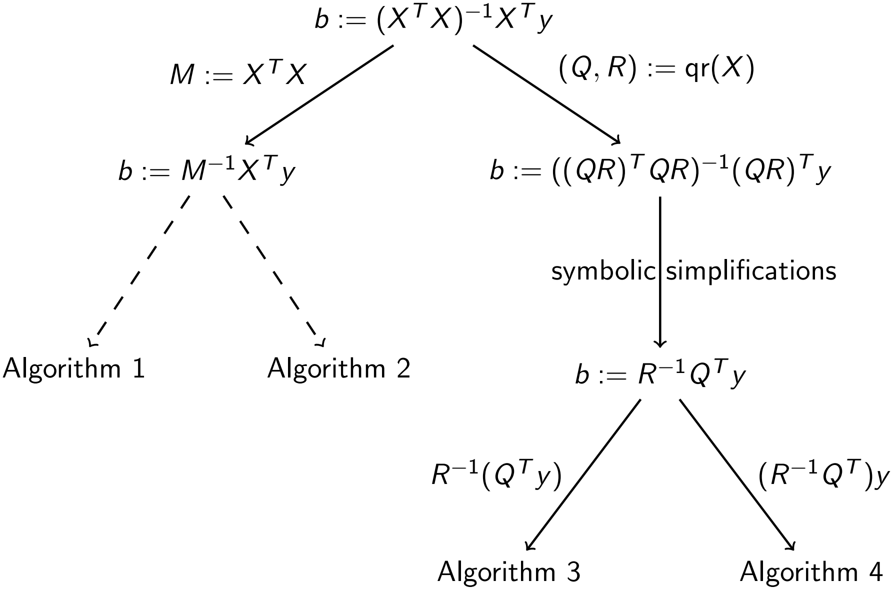
?

$y := \alpha x + y$ | $\{L, U\} := LU(A)$ | $C := \alpha AB + \beta C$

$L := L^{-1}$ | $C := AB^T + BA^T + C$ | ...

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

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


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

Linear Algebra Mapping Problem (LAMP)

Linear Algebra Mapping Problem (LAMP)

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

Linear Algebra Mapping Problem (LAMP)

- $\mathcal{E}$: a list of assignments $var_i := LAexp_i$
- $\mathcal{K}$: a list of available computational kernels

Linear Algebra Mapping Problem (LAMP)

- $\mathcal{E}$: a list of assignments $var_i := LAexp_i$
- $\mathcal{K}$: a list of available computational kernels
- $\mathcal{M}$: a metric

Linear Algebra Mapping Problem (LAMP)

- $\mathcal{E}$: a list of assignments $var_i := LAexp_i$
- $\mathcal{K}$: a list of available computational kernels
- $\mathcal{M}$: a metric

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 $var_i := LAexp_i$
- $\mathcal{K}$: a list of available computational kernels
- $\mathcal{M}$: a metric

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

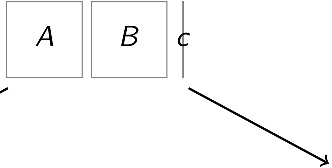
productivity vs. efficiency

Challenges and State of the Art

- Parenthesisation

Challenges and State of the Art

- Parenthesisation



$(AB)c \quad \mathcal{O}(n^3)$ $\qquad$ $A(Bc) \quad \mathcal{O}(n^2)$

$\Rightarrow$ Matrix Chain Algorithm

Challenges and State of the Art

- Parenthesisation

In practice: $\begin{cases} X := AB^T C^{-T} D + \dots \\ \text{LowerTriangular}(B) \\ \text{Symmetric}(C) \end{cases}$

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

Challenges and State of the Art

- Metric: FLOPs vs. execution time;

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

Challenges and State of the Art

- Metric: FLOPs vs. execution time;

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

$\Rightarrow$ Performance prediction

Challenges and State of the Art

- Metric: FLOPs vs. execution time; data moved, constraint on memory usage

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

$\Rightarrow$ Performance prediction

Challenges and State of the Art

- Multi-level metric: efficiency + stability

$X := AB^T C^{-T} D$

Never explicit inversions

$\rightarrow$ linear systems

$\Rightarrow$ Tradeoff between speed and stability;

Challenges and State of the Art

- Multi-level metric: efficiency + stability

$X := AB^T C^{-T} D$

Never explicit inversions

$\rightarrow$ linear systems

however…

$Y := A^{-1} B^{-1}$?

$\Rightarrow$ Tradeoff between speed and stability; conditioning?

Challenges and State of the Art

- Linear algebra knowledge: identities, implications, theorems

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

$\Rightarrow$ "Knowledge base" – expert system

Challenges and State of the Art

- Inference of properties

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

$$\lambda(A, B) \wedge \mathrm{SPD}(B) \rightarrow \lambda(L^{-T}AL^{-1}) \qquad \text{symmetric}(L^{-T}AL^{-1})\ ?$$

$\Rightarrow$ Static analysis

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

- LAMP is NP complete

Linnea – Linear algebra compiler

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

Linnea – Linear algebra compiler

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

Naive

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

Recommended

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

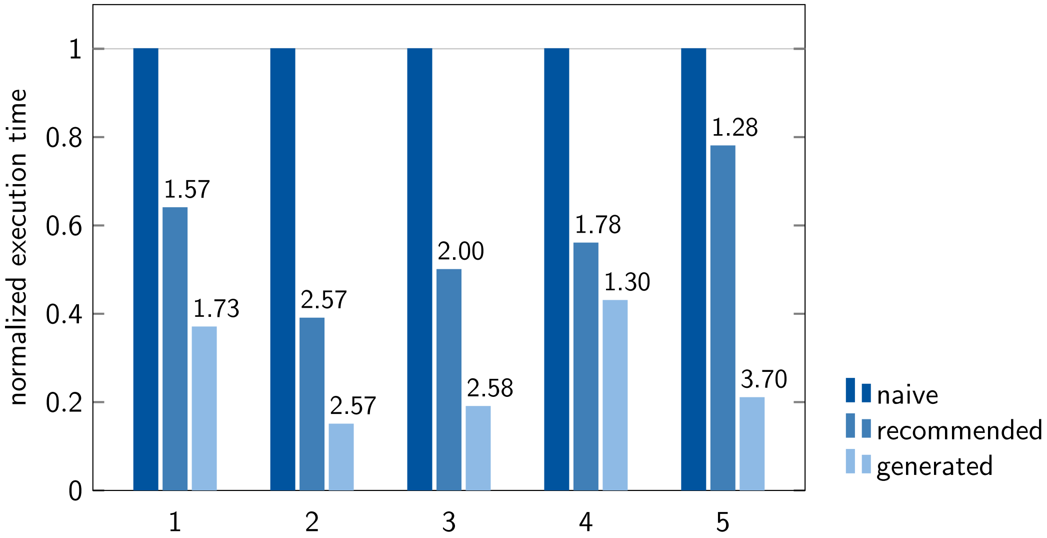
Generated

```
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	$b := (X^T X)^{-1} X^T y$
2	$b := (X^T M^{-1} X)^{-1} X^T M^{-1} y$
3	$W := A^{-1} B C D^{-T} E F$
4	$\begin{cases} X := A B^{-1} C \\ Y := D B^{-1} A^T \end{cases}$
5	$x := W(A^T (A W A^T)^{-1} b - c)$

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-sased 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-sased 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