Automating the generation of algorithms for Generalized Least-Squares problems

Diego Fabregat-Traver and Prof. Paolo Bientinesi

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

ECCOMAS 2012
Vienna, September 12th, 2012

... classic problems

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

... classic problems

- $b := \left(X^T X\right)^{-1} X^T y$
  - ➥ GELS

How to efficiently solve... ?

... classic problems

- $b := \left(X^T X\right)^{-1} X^T y$
  - ➥ GELS
- $b := \left(X^T M^{-1} X\right)^{-1} X^T M^{-1} y$
  - ➥ ?

... classic problems

- $b := \left(X^T X\right)^{-1} X^T y$
  - ➥ GELS
- $b := \left(X^T M^{-1} X\right)^{-1} X^T M^{-1} y$
  - ➥ ? $\rightarrow$ Reduce to above

How to efficiently solve... ?

... classic problems

- $b := \left(X^T X\right)^{-1} X^T y$
  - ➥ GELS
- $b := \left(X^T M^{-1} X\right)^{-1} X^T M^{-1} y$
  - ➥ ? $\rightarrow$ Reduce to above

... sequences of such problems

- $\begin{cases} b_{ij} := (X_i^T M_j^{-1} X_i)^{-1} X_i^T M_j^{-1} y_j \\ M_j := h_j \Phi + (1 - h_j) I \end{cases}$

How to efficiently solve... ?

... classic problems

- $b := \left(X^T X\right)^{-1} X^T y$
  - ➥ GELS
- $b := \left(X^T M^{-1} X\right)^{-1} X^T M^{-1} y$
  - ➥ ? $\rightarrow$ Reduce to above

... sequences of such problems

- $\begin{cases} b_{ij} := (X_i^T M_j^{-1} X_i)^{-1} X_i^T M_j^{-1} y_j \\ M_j := h_j \Phi + (1 - h_j) I \end{cases}$
  - ➥ Smart mapping onto BLAS/LAPACK

How to efficiently solve... ?

... classic problems

- $b := \left(X^T X\right)^{-1} X^T y$
  - ➥ GELS
- $b := \left(X^T M^{-1} X\right)^{-1} X^T M^{-1} y$
  - ➥ ? $\rightarrow$ Reduce to above

... sequences of such problems

- $\begin{cases} b_{ij} := (X_i^T M_j^{-1} X_i)^{-1} X_i^T M_j^{-1} y_j \\ M_j := h_j \Phi + (1 - h_j) I \end{cases}$
  - ➥ Smart mapping onto BLAS/LAPACK
  - ➥ The decomposition is not unique: many algorithms

RWTH AACHEN UNIVERSITY

Linear Algebra Compiler	
Input	Matrix equation + App-specific Knowledge

AI ces

Linear Algebra Compiler	
Input	Matrix equation + App-specific Knowledge
Output	Family of algorithms

Linear Algebra Compiler	
Input	Matrix equation + App-specific Knowledge
Output	Family of algorithms
Approach	Map onto high-performance kernels

Linear Algebra Compiler	
Input	Matrix equation + App-specific Knowledge
Output	Family of algorithms
Approach	Map onto high-performance kernels

Search: Not exhaustive. Guidelines. Led by knowledge.

How to explore the search space

- Inverse operator:
  - A^{-1}: factorization
    - ➥ $LL^T = A, \quad QR = A, \quad ZWZ^T = A, ...$

RWTH AACHEN UNIVERSITY

How to explore the search space

- Inverse operator:
  - A^{-1}: factorization
    - ➥ $LL^T = A$, $QR = A$, $ZWZ^T = A$, ...
  - $(X^T X)^{-1}$: factorization or mapping onto kernels
    - ➥ $S := X^T X$
    - ➥ $QR = X$

AI ces

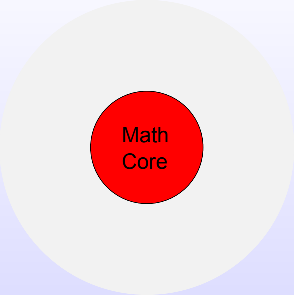
How to explore the search space

- Inverse operator:
  - A^{-1}: factorization
    - ➥ $LL^T = A, \quad QR = A, \quad ZWZ^T = A, ...$
  - $(X^TX)^{-1}$: factorization or mapping onto kernels
    - ➥ $S := X^TX$
    - ➥ $QR = X$
- Mapping onto kernels
  - Reuse computations: $S = X^TL^{-T}L^{-1}XC$
    - (1) $K := L^{-1}X$
    - (2) $S = K^TKC$

How to explore the search space

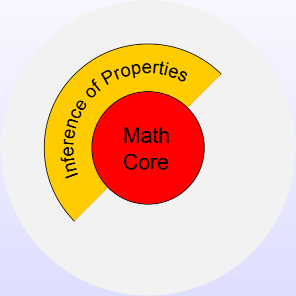
- Inverse operator:
  - A^{-1}: factorization
    - ➥ $LL^T = A, \quad QR = A, \quad ZWZ^T = A$, ...
  - $(X^TX)^{-1}$: factorization or mapping onto kernels
    - ➥ $S := X^TX$
    - ➥ $QR = X$
- Mapping onto kernels
  - Reuse computations: $S = X^TL^{-T}L^{-1}XC$
    (1) $K := L^{-1}X$
    (2) $S = K^TKC$
  - Reducing flops: $S = R^{-1}Q^TLy$

Compiler's engine



- Math core:
  - Matrix, Vector, Scalar, Size/Shape, ...
  - Diagonal, L/U triangular, Symm, ...
  - Operators: +, -, *, $^{-1}$, ^{T}. Properties.

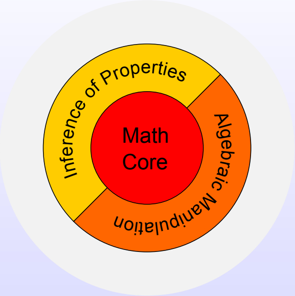
- X: {Matrix, FullRank, ColumnPanel}
- L: {Matrix, Square, Lower Triangular}
- $(LL^T)^{-1} \rightarrow L^{-T}L^{-1}$
- $(X^TX)^{-1} \rightarrow (X^TX)^{-1}$



- Math core:
  - Matrix, Vector, Scalar, Size/Shape, ...
  - Diagonal, L/U triangular, Symm, ...
  - Operators: +, -, *, $^{-1}$, ^{T}. Properties.
- Inference of properties / Propagation

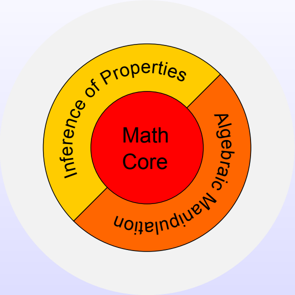
- $A := X^T X \rightarrow A$ is SPD
- $QR = X \rightarrow Q$ is Orthonormal, R is Triangular

Compiler's engine



- Math core:
  - Matrix, Vector, Scalar, Size/Shape, ...
  - Diagonal, L/U triangular, Symm, ...
  - Operators: +, -, *, $^{-1}$, ^{T}. Properties.
- Inference of properties / Propagation
- Arithmetic, simplifications

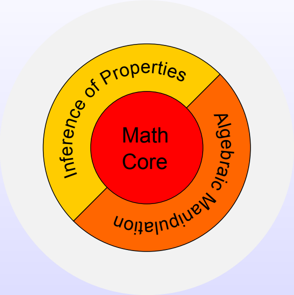
- $(R^TQ^TQR)^{-1}R^TQ^T$



- Math core:
  - Matrix, Vector, Scalar, Size/Shape, ...
  - Diagonal, L/U triangular, Symm, ...
  - Operators: +, -, *, $^{-1}$, ^{T}. Properties.
- Inference of properties / Propagation
- Arithmetic, simplifications

- $(R^T Q^T Q R)^{-1} R^T Q^T$

  $\Rightarrow (R^T R)^{-1} R^T Q^T$

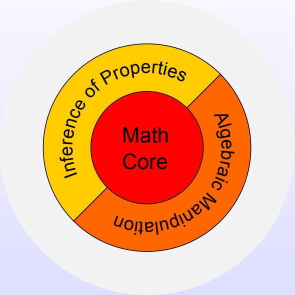


- Math core:
  - Matrix, Vector, Scalar, Size/Shape, ...
  - Diagonal, L/U triangular, Symm, ...
  - Operators: +, -, *, $^{-1}$, ^{T}. Properties.
- Inference of properties / Propagation
- Arithmetic, simplifications

- $(R^TQ^TQR)^{-1}R^TQ^T$

  $\Rightarrow (R^TR)^{-1}R^TQ^T$

  $\Rightarrow R^{-1}R^{-T}R^TQ^T$



- Math core:
  - Matrix, Vector, Scalar, Size/Shape, ...
  - Diagonal, L/U triangular, Symm, ...
  - Operators: +, -, *, $^{-1}$, ^{T}. Properties.
- Inference of properties / Propagation
- Arithmetic, simplifications

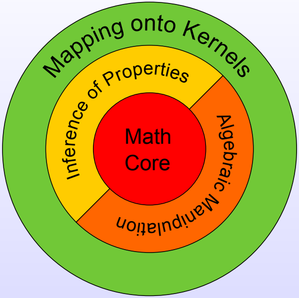
- $(R^T Q^T Q R)^{-1} R^T Q^T$

  $\Rightarrow (R^T R)^{-1} R^T Q^T$

  $\Rightarrow R^{-1} R^{-T} R^T Q^T$

  $\Rightarrow R^{-1} Q^T$

Compiler's engine



- Math core:
  - Matrix, Vector, Scalar, Size/Shape, ...
  - Diagonal, L/U triangular, Symm, ...
  - Operators: +, -, *, $^{-1}$, ^{T}. Properties.
- Inference of properties / Propagation
- Arithmetic, simplifications
- Kernels

- Factorizations: QR, LU, Cholesky, Eigen, ...
- BLAS: GEMM, TRSM, GEMV, DOT, ...
- LAPACK: inverse of a triangular matrix, ...
- Extensible

Example: Input

$$\begin{cases} b := (X^T M^{-1} X)^{-1} X^T M^{-1} y \\ M := h\Phi + (1-h)I \end{cases}$$

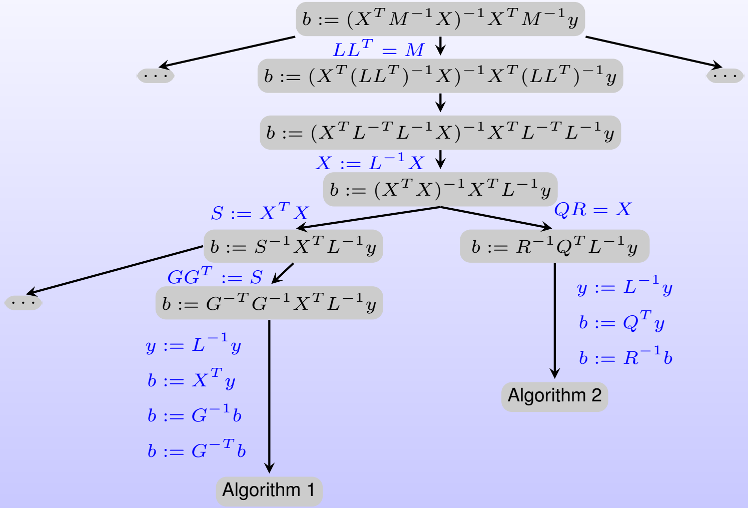
```
equation = {
  equal[b,
    times[ inv[ times[ trans[X], inv[M], X ] ],
      ...
      y ]
  ] };
```

```
properties = {
  {X, {"Input", "Matrix", "ColPanel", "FullRank"}}
  {y, {"Input", "Vector" }}
  ...
  {b, {"Output", "Vector" }}
};
```

Example: Generation

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

Example: Generation



Sequences of correlated problems

$$\begin{cases} b_{ij} = (X_i^T M_j^{-1} X_i)^{-1} X_i^T M_j^{-1} y_j & \text{with } 1 \leq i \leq m \\ M_j = h_j \Phi + (1 - h_j) I & \text{and } 1 \leq j \leq t. \end{cases}$$

- We have to solve not one but a sequence of **correlated** problems

Sequences of correlated problems

$$\begin{cases} b_{ij} = (X_i^T M_j^{-1} X_i)^{-1} X_i^T M_j^{-1} y_j & \text{with } 1 \le i \le m \\ M_j = h_j \Phi + (1 - h_j) I & \text{and } 1 \le j \le t. \end{cases}$$

- We have to solve not one but a sequence of **correlated** problems
- Goal: **reuse of computation**

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

$$
\begin{array}{l}
\texttt{for}\ \ i = 1 : m \\
\quad \texttt{for}\ \ j = 1 : t \\
\quad\quad LL^T = M_j \\
\quad\quad X^T \leftarrow X_i^T L^{-T} \\
\quad\quad QR = X \\
\quad\quad y \leftarrow L^{-1} y_j \\
\quad\quad b \leftarrow Q^T y \\
\quad\quad b_{ij} \leftarrow R^{-1} b
\end{array}
$$

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

$$\begin{aligned}
&\texttt{for}\ \ i = 1 : m \\
&\quad \texttt{for}\ \ j = 1 : t \\
&\qquad L_j L_j^T = M_j \\
&\qquad X_{ij}^T \leftarrow X_i^T L_j^{-T} \\
&\qquad Q_{ij} R_{ij} = X_{ij} \\
&\qquad y_j \leftarrow L_j^{-1} y_j \\
&\qquad b_{ij} \leftarrow Q_{ij}^T y_j \\
&\qquad b_{ij} \leftarrow R_{ij}^{-1} b_{ij}
\end{aligned}$$

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

$\texttt{for}\ \ i = 1 : m$
$\quad\texttt{for}\ \ j = 1 : t$
$\qquad L_j L_j^T = M_j$
$\qquad X_{ij}^T \leftarrow X_i^T L_j^{-T}$
$\qquad Q_{ij} R_{ij} = X_{ij}$
$\qquad y_j \leftarrow L_j^{-1} y_j$
$\qquad b_{ij} \leftarrow Q_{ij}^T y_j$
$\qquad b_{ij} \leftarrow R_{ij}^{-1} b_{ij}$

$\texttt{for}\ \ j = 1 : t$
$\quad\texttt{for}\ \ i = 1 : m$
$\qquad L_j L_j^T = M_j$
$\qquad X_{ij}^T \leftarrow X_i^T L_j^{-T}$
$\qquad Q_{ij} R_{ij} = X_{ij}$
$\qquad y_j \leftarrow L_j^{-1} y_j$
$\qquad b_{ij} \leftarrow Q_{ij}^T y_j$
$\qquad b_{ij} \leftarrow R_{ij}^{-1} b_{ij}$

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

$$\begin{array}{l}
\texttt{for}\ \ i = 1:m \\
\quad \texttt{for}\ \ j = 1:t \\
\quad\quad L_j L_j^T = M_j \\
\quad\quad X_{ij}^T \leftarrow X_i^T L_j^{-T} \\
\quad\quad Q_{ij} R_{ij} = X_{ij} \\
\quad\quad y_j \leftarrow L_j^{-1} y_j \\
\quad\quad b_{ij} \leftarrow Q_{ij}^T y_j \\
\quad\quad b_{ij} \leftarrow R_{ij}^{-1} b_{ij}
\end{array}$$

$$\begin{array}{l}
\texttt{for}\ \ j = 1:t \\
\quad \texttt{for}\ \ i = 1:m \\
\quad\quad {\color{red} L_j L_j^T = M_j} \\
\quad\quad X_{ij}^T \leftarrow X_i^T L_j^{-T} \\
\quad\quad Q_{ij} R_{ij} = X_{ij} \\
\quad\quad {\color{red} y_j \leftarrow L_j^{-1} y_j} \\
\quad\quad b_{ij} \leftarrow Q_{ij}^T y_j \\
\quad\quad b_{ij} \leftarrow R_{ij}^{-1} b_{ij}
\end{array}$$

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

$$\begin{array}{l}
\texttt{for } i = 1 : m \\
\quad \texttt{for } j = 1 : t \\
\quad\quad L_j L_j^T = M_j \\
\quad\quad X_{ij}^T \leftarrow X_i^T L_j^{-T} \\
\quad\quad Q_{ij} R_{ij} = X_{ij} \\
\quad\quad y_j \leftarrow L_j^{-1} y_j \\
\quad\quad b_{ij} \leftarrow Q_{ij}^T y_j \\
\quad\quad b_{ij} \leftarrow R_{ij}^{-1} b_{ij}
\end{array}$$

$$\begin{array}{l}
\texttt{for } j = 1 : t \\
\quad L_j L_j^T = M_j \\
\quad y_j \leftarrow L_j^{-1} y_j \\
\quad \texttt{for } i = 1 : m \\
\quad\quad X_{ij}^T \leftarrow X_i^T L_j^{-T} \\
\quad\quad Q_{ij} R_{ij} = X_{ij} \\
\quad\quad b_{ij} \leftarrow Q_{ij}^T y_j \\
\quad\quad b_{ij} \leftarrow R_{ij}^{-1} b_{ij}
\end{array}$$

Computational cost

How to pick a variant?

- Many algorithms for a single target equation. How do we pick one?

- Many algorithms for a single target equation. How do we pick one?
- Metric: flop count (often times not too descriptive)

Scenario	Alg. 1	Alg. 2	Alg. 3
One instance	$O(n^3)$	$O(n^3)$	$O(n^3)$
2D sequence	$O(tn^3 + mtn^2)$	$O(tn^3 + mtn^2)$	$O(n^3 + mtn)$

- Many algorithms for a single target equation. How do we pick one?
- Metric: flop count (often times not too descriptive)

Scenario	Alg. 1	Alg. 2	Alg. 3
One instance	$O(n^3)$	$O(n^3)$	$O(n^3)$
2D sequence	$O(tn^3 + mtn^2)$	$O(tn^3 + mtn^2)$	$O(n^3 + mtn)$

Elmar Peise, *Hierarchical Performance Modeling for Ranking Dense Linear Algebra Algorithms*, 2012. `http://arxiv.org/abs/1207.5217`

Conclusions and future work

So far...

- Domain-specific linear algebra compiler
- Equation + Knowledge $\rightarrow$ Families of algorithms

Conclusions and future work

So far...

- Domain-specific linear algebra compiler
- Equation + Knowledge $\rightarrow$ Families of algorithms
- Guidelines + Engine
- Extensions: Sequences, cost analysis

Conclusions and future work

So far...

- Domain-specific linear algebra compiler
- Equation + Knowledge $\rightarrow$ Families of algorithms
- Guidelines + Engine
- Extensions: Sequences, cost analysis
- Sequences of GLSs (GWAS): speedups > 100

Conclusions and future work

So far...

- Domain-specific linear algebra compiler
- Equation + Knowledge $\rightarrow$ Families of algorithms
- Guidelines + Engine
- Extensions: Sequences, cost analysis
- Sequences of GLSs (GWAS): speedups > 100

TO-DO

- Encode more available knowledge
- Rank algorithms to pick the "best"
- Matlab/Fortran code generator

Thanks to:

- Dr. Edoardo Di Napoli
- Matthias Petschow
- Roman Iakymchuk
- Elmar Peise

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

Deutsche Forschungsgemeinschaft
DFG

AI ces

More details in

D. Fabregat, P. Bientinesi, *A Domain-Specific Compiler for Linear Algebra Operations*, 2012. http://arxiv.org/abs/1205.5975

Further questions?

fabregat@aices.rwth-aachen.de