Code Generation in Linnea

Henrik Barthels, Paolo Bientinesi
Introduction

• How to compute the following expressions?

\[
b := (X^T M^{-1} X)^{-1} X^T M^{-1} y
\]
\[
x := W (A^T (AWA^T)^{-1} b - c)
\]
\[
x := (A^{-T} B^T B A^{-1} + R^T [\Lambda (R z)] R)^{-1} A^{-T} B^T B A^{-1} y
\]
\[
X_{k+1} := S (S^T A S)^{-1} S^T + (I_n - S (S^T A S)^{-1} S^T A) X_k (I_n - A S (S^T A S)^{-1} S^T)
\]

• High-level languages are easy to use, but performance is usually suboptimal.
• BLAS and LAPACK can be fast, but require a lot of expertise.
• Goal: Simplicity and performance.
How to compute...

\[ y' := H^\dagger y + (I_n - H^\dagger H)x \]  \hspace{2cm} \text{[TG17]}

...with these operations?

\[ x := Ab \quad 2n^2 \]
\[ X := AB \quad 2n^3 \]
\[ x := a \pm b \quad n \]
\[ X := A \pm B \quad n^2 \]
How to compute...
\[ y' := H^\dagger y + (I_n - H^\dagger H)x \]  
\[ \text{[TG17]} \]

...with these operations?

\[ x := Ab \quad 2n^2 \]

\[ X := AB \quad 2n^3 \]

\[ x := a \pm b \quad n \]

\[ X := A \pm B \quad n^2 \]

\[
\begin{align*}
M_1 &:= H^\dagger H \\
M_2 &:= I_n - M_1 \\
m_3 &:= M_2 x \\
m_4 &:= H^\dagger y \\
y' &:= m_3 + m_4 \\
\Rightarrow & \quad 2n^3 + 5n^2 + n \text{ FLOPs}
\end{align*}
\]
Introduction

How to compute...

\[ y' := H^\dagger y + (I_n - H^\dagger H)x \quad \text{[TG17]} \]

\[ \Leftrightarrow y' := H^\dagger y + x - H^\dagger Hx \]

\[ \Leftrightarrow y' := H^\dagger (y - Hx) + x \]

...with these operations?

\[ x := Ab \quad 2n^2 \]

\[ X := AB \quad 2n^3 \]

\[ x := a \pm b \quad n \]

\[ X := A \pm B \quad n^2 \]

\[ M_1 := H^\dagger H \]

\[ M_2 := I_n - M_1 \]

\[ m_3 := M_2 x \]

\[ m_4 := H^\dagger y \]

\[ y' := m_3 + m_4 \]

\[ \Rightarrow 2n^3 + 5n^2 + n \text{ FLOPs} \]
## Introduction

### How to compute...

<table>
<thead>
<tr>
<th>Equation</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y' := H^\dagger y + (I_n - H^\dagger H)x$</td>
<td>[TG17]</td>
</tr>
<tr>
<td>$\Leftrightarrow y' := H^\dagger y + x - H^\dagger Hx$</td>
<td></td>
</tr>
<tr>
<td>$\Leftrightarrow y' := H^\dagger (y - Hx) + x$</td>
<td></td>
</tr>
</tbody>
</table>

### ...with these operations?

<table>
<thead>
<tr>
<th>Operation</th>
<th>Equation</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x := A b$</td>
<td>$2n^2$</td>
<td></td>
</tr>
<tr>
<td>$X := A B$</td>
<td>$2n^3$</td>
<td></td>
</tr>
<tr>
<td>$x := a \pm b$</td>
<td>$n$</td>
<td></td>
</tr>
<tr>
<td>$X := A \pm B$</td>
<td>$n^2$</td>
<td></td>
</tr>
</tbody>
</table>

### Code Generation

<table>
<thead>
<tr>
<th>Equation</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1 := H^\dagger H$</td>
<td></td>
</tr>
<tr>
<td>$M_2 := I_n - M_1$</td>
<td></td>
</tr>
<tr>
<td>$m_3 := M_2 x$</td>
<td></td>
</tr>
<tr>
<td>$m_4 := H^\dagger y$</td>
<td></td>
</tr>
<tr>
<td>$y' := m_3 + m_4$</td>
<td></td>
</tr>
</tbody>
</table>

$\Rightarrow 2n^3 + 5n^2 + n$ FLOPs
Instruction Set

**BLAS** [DDC$^+90$]

- $y \leftarrow Ax + y$
- $C \leftarrow AB + C$
- $B \leftarrow A^{-1}B$
- ...

**LAPACK** [AB$^+99$]

- Matrix factorizations.
- Eigensolvers.
- Solvers for linear systems with specific properties.
Storage Formats

An Algebra of Banded Matrices
**Storage Formats**

**Code Generation in Linnea**

$$w := AB^{-1}c$$

\[
\begin{align*}
L &:= \text{cholesky}(B) \\
v_1 &:= L^{-1}c \\
v_2 &:= L^{-T}v_1 \\
w &:= A v_2
\end{align*}
\]

ml0 = A; ml1 = B; ml2 = c; 
\text{potrf}(!_L', ml1) \\
\text{trsv}(!_L', 'N', 'N', ml1, ml2) \\
\text{trsv}(!_L', 'T', 'N', ml1, ml2) \\
ml3 = \text{Array}[\text{Float64}](1000) \\
\text{gemv}(!_N', 1.0, ml0, ml2, 0.0, ml3) \\
w = ml3
Storage Formats

Example: LU Factorization \((\text{getrf})\)

\[ A \rightarrow PLU \]
Storage Formats

Example: Triangular solve \((\text{trsm})\)

\[ C \leftarrow A^{-1}B \]

- \(A\) can be upper/lower triangular.
- \(A\) can have unit diagonal.
Storage Formats

The Problem

• We need to know how operands are stored.
• We need to know how kernels read operands.
• We need to be able to change storage formats.
• Goal: Only change storage formats when necessary.
Storage Formats

- (Unit-)Triangular matrices.
- Permutation.
- Symmetric matrices.

- Banded matrices.

- Packed (triangular and symmetric).

- Diagonal.
Storage Formats

Compatibility Relation

\[
\text{lower\_triangular} \preceq \text{full} \\
\text{full} \not\preceq \text{lower\_triangular}
\]

\(\alpha \preceq \beta\) if

- all explicitly represented elements in \(\alpha\) are explicitly represented in \(\beta\), and
- all elements in \(\alpha\) have the same positions in \(\beta\).
Storage Formats

The Algorithm

• Given kernel \( K \) with input operands \( M_1, M_2, \ldots \)
• Kernels are annotated with required input formats \( r_1, r_2, \ldots \)
• Operands are annotated with current format \( f_1, f_2, \ldots \)
• Database of storage format conversions.

\textbf{for all} \( M_i \):
  \textbf{if} \( r_i \not< f_i \):
    \text{find conversion} \( f_i \rightarrow f \) with \( r_i \not< f \)
  \textbf{if} conversion is in place:
    \text{check if something gets overwritten}
Storage Formats

An Algebra of Banded Matrices
An Algebra of Banded Matrices

Upper and Lower Bandwidth

Definition [GVL13]:

$A \in \mathbb{R}^{n \times n}$ has

- lower bandwidth $l$ if $i > j + l$ implies $a_{ij} = 0$, and
- upper bandwidth $u$ if $j > i + u$ implies $a_{ij} = 0$. 
An Algebra of Banded Matrices

Examples

\[(0,0)\]  \[(0,4)\]  \[(1,4)\]
An Algebra of Banded Matrices

What about operations on banded matrices?

Bandwidth of $A + B$ is $(\max(l_A, l_B), \max(u_A, u_B))$. 
What about operations on banded matrices?

Bandwidth of $A \cdot B$ is $(l_A + l_B, u_A + u_B)$. 
An Algebra of Banded Matrices

Generalization to Non-Square Matrices
Generalization to Negative Bandwidth

\[ l + u + 1 \leq 0 \] implies that \( A \) is zero.
An Algebra of Banded Matrices

Problem: Overapproximation

\[
\begin{pmatrix} 3, -2 \end{pmatrix} \quad \begin{pmatrix} -1, 4 \end{pmatrix} \quad \begin{pmatrix} 2, 2 \end{pmatrix}
\]

\[
\begin{array}{cccc}
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\end{array}
\quad \times
\quad \begin{array}{cccc}
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\end{array}
\quad =
\quad \begin{array}{cccc}
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\end{array}
\]
An Algebra of Banded Matrices

Partitioning Matrices
An Algebra of Banded Matrices

• Simple propagation of matrix properties.
• Type system for banded matrices (github.com/JuliaLang/julia, issue #8240).
• Better support in libraries.
Results
Example: \( w := AB^{-1}c \)

Naive
\( w = A*\text{inv}(B)*c \)

Recommended
\( w = A*(B\backslash c) \)

Generated
\[
\begin{align*}
ml0 &= A; \ ml1 = B; \ ml2 = c; \\
potrf!('L', ml1) \\
trsv!('L', 'N', 'N', ml1, ml2) \\
trsv!('L', 'T', 'N', ml1, ml2) \\
ml3 &= \text{Array}\{\text{Float64}\}(1000) \\
gemv!('N', 1.0, ml0, ml2, 0.0, ml3) \\
w &= ml3
\end{align*}
\]
Results

1 Thread

![Diagram showing speedup of Linnea for different test problems with various symbols representing different types of problems.]
Results

24 Threads

![Graph showing speedup of Linnea for 24 threads across various test problems. The graph includes symbols for different types of problems such as JL n, JL r, Arma n, Arma r, Eig n, Eig r, Mat n, and Mat r.]
References


Linnea is available online: https://github.com/HPAC/linnea