Efficient Solvers for (sparse symm. pos. def.) Linear Systems

Mario Botsch
ETH Zurich
Problems in Geometry Processing

• Generic formulation as a PDE
  – Based on partial derivatives

• Discretization for triangle meshes
  – Finite elements / differences
  – Lead to linear systems (typically $10^4$ to $10^6$ DoFs)

• Partial derivatives are local operators
  – Sparse linear systems
Problems in Geometry Processing

- Most often the PDE can be considered as the Euler-Lagrange equation of an energy minimization problem

- or \( A^T A x = A^T b \) emerges as the normal equation for a least squares problem

⇒ Systems are usually symmetric and pos. definite
Problems in Geometry Processing

• Linear problems:
  – Solve $Ax = b$

• Non-linear problems:
  – Solve sequence of linear systems $A_k x_k = b_k$

• Matrix $A$ typically is
  – large
  – sparse
  – symmetric positive definite

Non-spd systems: See course notes
Overview

- Application scenarios
- Linear system solvers
- Benchmarks
Implicit Fairing

\[(I \pm \Delta^k_S) \mathbf{x}_{k+1} = \mathbf{x}_k\]
Variational Energy Minimization

\[ \Delta x \equiv 0 \]
\[ \Delta^2 x \equiv 0 \]
\[ \Delta^3 x \equiv 0 \]
Explicit Hole Filling

\[ \Delta^2 x = 0 \]
Conformal Parameterization

\[ \Delta s u = 0 \]
Variational Mesh Editing

\[ \Delta^k_S d = 0 \]
Gradient-Based Editing

\[ \Delta_{SP} = \text{div} \left( g \right) \]
\[ \Delta_S f (v) := \frac{2}{A(v)} \sum_{v_i \in N_1(v)} (\cot \alpha_i + \cot \beta_i) (f(v_i) - f(v)) \]
Laplace Matrix

\[
\begin{pmatrix}
\vdots \\
\Delta^k_S f_i \\
\vdots 
\end{pmatrix}
= (DM)^k
\begin{pmatrix}
\vdots \\
f_i \\
\vdots 
\end{pmatrix}
\]

\[
M_{ij} = \begin{cases} 
\cot\alpha_{ij} + \cot\beta_{ij}, & i \neq j, j \in N_1(v_i) \\
0 & i \neq j, j \notin N_1(v_i) \\
- \sum_{v_j \in N_1(v_i)} (\cot\alpha_{ij} + \cot\beta_{ij}) & i = j
\end{cases}
\]

\[
D = \text{diag} \left( \ldots, \frac{2}{A(v_i)}, \ldots \right)
\]
Laplace Matrix

\[
\begin{pmatrix}
\vdots \\
\Delta^k_S f_i \\
\vdots 
\end{pmatrix}
= (DM)^k
\begin{pmatrix}
\vdots \\
f_i \\
\vdots 
\end{pmatrix}
\]

- Degree of sparsity: \(1 + 3 (k^2 + k)\)
  - \(k=1\) ... 7
  - \(k=2\) ... 19
  - \(k=3\) ... 37
Laplace Matrix

\[
\begin{pmatrix}
\vdots \\
\Delta^k_S f_i \\
\vdots
\end{pmatrix}
= (DM)^k
\begin{pmatrix}
\vdots \\
f_i \\
\vdots
\end{pmatrix}
\]

- \((DM)^k\) is not symmetric, but \(M(DM)^{k-1}\) is symmetric.

\[\Rightarrow\text{ Instead of } (DM)^k x = b\]
\[\text{solve } M(DM)^{k-1} x = D^{-1}b\]
Laplace Matrix

\[
\begin{pmatrix}
\ddots \\
\Delta^k_S f_i \\
\ddots \\
\end{pmatrix}
= (DM)^k 
\begin{pmatrix}
\ddots \\
fi \\
\ddots \\
\end{pmatrix}
\]

- Positive definiteness
  - Can be derived by variational calculus
  - Energy minimization subject to constraints
Least Squares Conformal Maps

\[ A^T A x = A^T b \]
Non-Linear Problems

Non-linear minimization (Newton)

\[ H(x) h = -\nabla f(x) \]

Non-linear least squares (Gauss-Newton)

\[ J(x)^T J(x) h = -J(x)^T f(x) \]
Overview

- Application scenarios
- Linear system solvers
- Benchmarks
Dense Direct Solvers

• Symmetric positive definite \((\text{spd})\)
  – Cholesky factorization \((A=LL^T)\)
  – Solve systems by back-substitution
  – Numerically stable

• Complexity
  – Factorization \(O(n^3)\)
  – Back-substitution \(O(n^2)\)
Iterative Solvers

• Symmetric, positive definite, sparse
  – Conjugate gradients
  – Krylov spaces \( K_i = \{ b, Ab, ..., A^{i-1}b \} \)
  – Robust, monotone convergence
  – Exact solution after \( n \) iterations

• Complexity
  – Each iteration is \( O(n) \) (sparse!)
  – Total complexity \( O(n^2) \)
Iterative Solvers

• Numerical convergence rate
  – Depends on matrix condition
  – Preconditioning is mandatory \((A' = PAP^T)\)
  – Problematic for large systems ...

• Iterative solvers are “smoothers”
  – Rapid elimination of high frequency errors
  – Impractically slow convergence for low frequencies
Multigrid Solvers

• Build a hierarchy of meshes
  – Mesh decimation
  – $O(\log n)$ levels
Multigrid Solvers

• Apply some pre-smoothing steps on finest level
  – Removes highest error frequencies

• Remaining low frequency error \((r=b-Ax)\)
  – Corresponds to high frequencies on coarser levels
  – Iterate / solve residual system \((Ae=r)\) on coarse level

• Propagate solution to finer level
  – Followed by post-smoothing steps

• Total \(O(n)\) complexity!
Multigrid Solvers

- V-cycle
- Full Multigrid
- Cascading MG
Multigrid Solvers

- MG can be quite tricky:
  - How to build an irregular hierarchy?
  - How many levels?
  - Special MG pre-conditioners
  - Restriction of system
  - Prolongation of coarse solution

- [Aksoyulu et al. 2003], [Shi et al. 2006]
Direct Sparse Solvers

• Dense solvers do not exploit sparsity
  – Matrix factors are dense

\[ A = LL^T \]
Direct Sparse Solvers

• Dense solvers do not exploit sparsity
  – Matrix factors are dense

• Band-limitation can be exploited
  – Bandwidth of factors is that of $\mathbf{A}$
  – More precisely: envelope is preserved
Direct Sparse Solvers

- Dense solvers do not exploit sparsity
  - Matrix factors are dense

- Band-limitation can be exploited
  - Bandwidth of factors is that of $A$
  - More precisely: envelope is preserved

- Complexity
  - Factorization $O(nb^2)$
  - Back-substitution $O(nb)$
Matrix Re-Ordering

Natural

$$LL^T$$

36k NZ

$$L$$
Matrix Re-Ordering

• Find symmetric permutation $A' = P^TAP$

• ... which minimizes the band-width:
  ➡ Cuthill-McKee algorithm
Matrix Re-Ordering

Natural  RCMK

$LL^T$

$L$

36k NZ  14k NZ
Matrix Re-Ordering

- Find symmetric permutation $A' = P^T A P$

- ... which minimizes the band-width:
  - Cuthill-McKee algorithm

- ... which minimizes the envelope fill-in of $L$:
  - Minimum Degree algorithm
Matrix Re-Ordering

\[ LL^T \]

\[ L \]

Natural | RCMK | MD

36k NZ | 14k NZ | 6.2k NZ
Matrix Re-Ordering

- Find symmetric permutation \( A' = P^T A P \)

- ... which minimizes the band-width:
  ➪ Cuthill-McKee algorithm

- ... which minimizes the envelope fill-in of \( L \):
  ➪ Minimum Degree algorithm

- ... based on recursive graph partitioning:
  ➪ METIS algorithm
Matrix Re-Ordering

\[ LL^T \]

Natural  
RCMK  
MD  
Metis  

\[ L \]

36k NZ  
14k NZ  
6.2k NZ  
7.1k NZ  

Mario Botsch, ETH Zurich
Sparse Cholesky Factorization

• Non-zero structure of $L$ can be predicted from the non-zero structure of $A$
  – Build a static data structure in advance
  – Symbolic factorization

• Compute numerical entries of $L$ based on this data structure
  – Better memory coherence
  – Numerical factorization
Sparse Cholesky Solver

1. Matrix re-ordering \[ \tilde{A} = P^T A P \]
2. Symbolic factorization \[ L \]
3. Numerical factorization \[ \tilde{A} = LL^T \]
4. Solve system \[ y = L^{-1} P^T b, \quad x = PL^{-T} y \]
Sparse Cholesky Solver

Only right hand side changes

1. Matrix re-ordering \( \tilde{A} = P^T A P \)
2. Symbolic factorization \( L \)
3. Numerical factorization \( \tilde{A} = LL^T \)
4. Solve system
   \[ y = L^{-1} P^T b, \quad x = PL^{-T} y \]
Sparse Cholesky Solver

Matrix values change

1. Matrix re-ordering \( \tilde{A} = P^T A P \)
2. Symbolic factorization \( L \)
3. Numerical factorization \( \tilde{A} = LL^T \)
4. Solve system \( y = L^{-1} P^T b, \quad x = PL^{-T} y \)
Overview

- Application scenarios
- Linear system solvers
- Benchmarks
Small Laplace Systems

Setup + Precomp. + 3 Solutions

Iterative  Multigrid  Sparse Cholesky
Small Laplace Systems

3 Solutions (per frame costs)

- Iterative
- Multigrid
- Sparse Cholesky
Large Laplace Systems

Setup + Precomp. + 3 Solutions

- Iterative
- Multigrid
- Sparse Cholesky
Large Laplace Systems

3 Solutions (per frame costs)

- Iterative
- Multigrid
- Sparse Cholesky
Small Bi-Laplace Systems

Setup + Precomp. + 3 Solutions

- Iterative
- Multigrid
- Sparse Cholesky

Mario Botsch, ETH Zurich
Small Bi-Laplace Systems

3 Solutions (per frame costs)

- Iterative
- Multigrid
- Sparse Cholesky

Mario Botsch, ETH Zurich
Large Bi-Laplace Systems

Setup + Precomp. + 3 Solutions

- Multigrid
- Sparse Cholesky
Large Bi-Laplace Systems

3 Solutions (per frame costs)

- Multigrid
- Sparse Cholesky

Mario Botsch, ETH Zurich
Shi et al., Fast MG Algo

Setup + Precomp. + 3 Solutions

Sparse Cholesky

Multigrid
Shi et al., Fast MG Algo

3 Solutions (per frame costs)

Multigrid

Sparse Cholesky
Conclusion

• Typical geometry processing problems are
  – large but sparse
  – symmetric positive definite

• Multigrid solvers
  – Require careful implementation
  – Use it if mesh / matrix changes frequently

• Direct sparse solvers
  – Easy to use (black-box)
  – Well suited for multiple rhs, or if only matrix values change