Solving Finite Element Linear Systems

Kaushik Kulkarni
University of Illinois at Urbana-Champaign

Finite Element Linear Systems

- Finite Element Linear Systems are classified as massive and sparse linear systems.
- These matrices have dense block diagonals and sparse off-diagonal pattern.
- Direct Solvers along with pivoting prove to be very useful in cases when the system is not well-conditioned.

Overview of direct solvers for such systems

- **1970**: A frontal solver for SPD matrices due to **Irons B.M.**
- **1983**:The Multifrontal Solution of Indefinite Sparse Symmetric Linear Systems due to **I. S. Duff et al.**
- **1995**: A Supernodal approach to Sparse Linear Systems due to **J. W. Demmel et al.**

Current Solvers

All the Finite Element solvers re-route their linear solves to a different library. Some of those solvers are:

- **UMFPACK**: Implements the Multifrontal method for general sparse matrices.
- **MUMPS**: Implements the Multifrontal method for symmetric positive definite matrices.
- **SuperLU**: Implements ths supernodal method for general matrices.

Objectives

- Study the state of the art methods that are currently used for such systems
- Implement and observe the results obtained
- To develop a deep understanding of how permutation matrices are obtained for ordering

Multifrontal Cholesky

The Cholesky decomposition helps us to factorize an **SPD** matrix $\boldsymbol{A}$ into a lower triangular matrix $\boldsymbol{L}$, such that:

$$\boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{A} \tag{1}$$

Multifrontal Methods help us to transform the factorization steps for a Sparse matrix to a combination of dense Cholesky factorizations. Multifrontal methods involve the following important 3 steps:

1. **Ordering:** Permuting the rows and columns such that we factorize $\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^T$ instead of $\boldsymbol{A}$
2. **Symbolic Factorization:** In order to know the extra fill generated in the graph we would need to generate the filled matrix
3. **Numerical Factorization:** Traversing along the elimination tree and running a recursive algorithm that deals with only Dense matrices.

Secret Ingredient

The last step accounts for all the concurrency in the algorithm. But the concurrency depends on the Elimination tree which is controlled in the first step. Hence, our objective would be to look at the ordering of the matrix so that we have the least fill introduced in order to approach to a *Fat elimination tree.*

Previous Studies

It has been reported that the problem of finding the optimal ordering is **NP-Complete**, making it a hard problem. But over the last few decades there have been some heuristic based algorithms that try to get to the solution.

- Minimum Degree Ordering
- *Approximate* Minimum Degree Ordering
- Graph Partitioning based Ordering

For finite elements linear systems, the second algorithm has been reported to perform better empirically. Hence, we try to analyze and implement it.

Results

The Approximate Minimum Degree ordering was implemented and the results were found to decrease the fill in. The following plots describe the change in the structure of the matrix after the ordering. The fill was observed to decrease by 25% relative to the natural ordering.

(a) Natural ordering (b) AMD ordering

Figure: Structure of the matrix affected by the *AMD*

Minor Detour

Some time had also been spent in trying to figure out whether there exists simple pre-conditioner/transformation for the matrix which would reduce the fill in the matrix. The Block-Jacobi preconditioner was seen to perform well by reducing the fill, but conditioning of the system would get affected, hence this was discontinued. The following plots would explain it better.

(a) Before PC (b) After PC

Figure: Effect of Block Jacobi Pre-conditioner

References

[1] Patrick R Amestoy, Timothy A Davis, and Iain S Duff.
An approximate minimum degree ordering algorithm.
SIAM Journal on Matrix Analysis and Applications, 17(4):886–905, 1996.

[2] Iain S Duff and John K Reid.
The multifrontal solution of indefinite sparse symmetric linear.
ACM Transactions on Mathematical Software (TOMS), 9(3):302–325, 1983.

Acknowledgements

I would like to thank Prof. Edgar Solomonik for his constructive feedback during the project.