Project Proposal: Inversion of structured matrices with standard admissibility conditions

Chao Chen
ICME, Stanford University

February 23, 2015

1 Problem Description

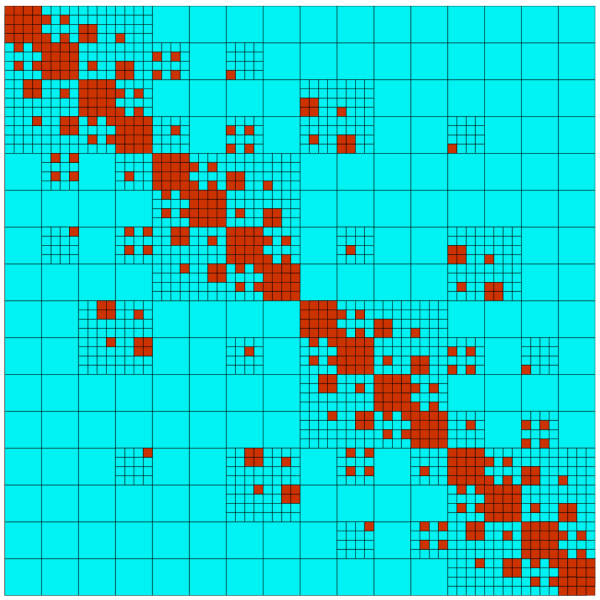

Figure 1: Standard admissibility leads to H2-Matrix

The problem is to invert structured matrices with standard admissibility conditions. Geometrically, for two sub-domains $k1$ and $k2$, the standard admissibility condition is

$$\min(\text{diam}(k1), \text{diam}(k2)) \leq \rho \, \text{dist}(k1, k2)$$

The corresponding matrix is shown in Fig. 1 as above. One choice would be to factorize the original matrix, which requires only $O(Nlog^2N)$ work. However the factorization achieves at most $O(log^2N)$ speedup if it is done in parallel, suffering from a long critical path. An alternative way is to use Newton-Schulz iteration. If we apply Newton's method to

$$f(X) = X^{-1} - A$$

we have the iteration

$$X_{k+1} = (2I - X_k A)X_k$$

which converges to A^{-1}. Ideally $O(log(\kappa(A)))$ approximate H-matrix compositions are required, where each composition involves $O(Nlog^2N)$ operations. Since H-matrix composition is highly parallelizable, and so, if the H-matrix approximations are all valid, the problem is solved.

Although for large classes of equations, A and A^{-1} are known to be representable as H-matrices with low ranks (see, e.g., [Hackbusch/Bebendorf-2003]), the intermediate iterates $X_{k+1} = (2IX_kA)X_k$ are typically not representable ([Hackbusch/Khoromskij/Tyrtyshnikov-2007]). Therefore, we need a sufficiently good initial guess for Newton-Schulz convergence, and this project aims at constructing the inverse of weekly-admissible H-matrices to start the iteration.

2 Inverse of H-matrix

We want to compute the explicit inverse of a weekly-admissible matrix A, i.e.

$$A = \begin{pmatrix} D_0 & U_0V_1^T \\ U_1V_0^T & D_1 \end{pmatrix}$$

where $U_0, U_1, V_0, V_1 \in \mathbb{R}^{n\times r}$ and D_0, D_1 have the same structure as A. Observe that A can be factorized as below.

$$A = \begin{pmatrix} D_0 & U_0V_1^T \\ U_1V_0^T & D_1 \end{pmatrix} = \begin{pmatrix} D_0 & \\ & D_1 \end{pmatrix} \begin{pmatrix} I & \bar{U}_0V_1^T \\ \bar{U}_1V_0^T & I \end{pmatrix}$$

where $D_0\bar{U}_0 = U_0, D_1\bar{U}_1 = U_1$. With Woodbury matrix identity, we have

$$\begin{aligned}
\begin{pmatrix} I & \bar{U}_0V_1^T \\ \bar{U}_1V_0^T & I \end{pmatrix}^{-1} &= \left(I + \begin{pmatrix} \bar{U}_0 & \\ & \bar{U}_1 \end{pmatrix} \begin{pmatrix} & V_1^T \\ V_0^T & \end{pmatrix}\right)^{-1} \\
&= I - \begin{pmatrix} \bar{U}_0 & \\ & \bar{U}_1 \end{pmatrix} \left(I + \begin{pmatrix} & V_1^T \\ V_0^T & \end{pmatrix} \begin{pmatrix} \bar{U}_0 & \\ & \bar{U}_1 \end{pmatrix}\right)^{-1} \begin{pmatrix} & V_1^T \\ V_0^T & \end{pmatrix} \\
&= I - \begin{pmatrix} \bar{U}_0 & \\ & \bar{U}_1 \end{pmatrix} \begin{pmatrix} I & V_1^T\bar{U}_1 \\ V_0^T\bar{U}_0 & I \end{pmatrix}^{-1} \begin{pmatrix} & V_1^T \\ V_0^T & \end{pmatrix}
\end{aligned}$$

Therefore, we know A^{-1} explicitly as follows

$$\begin{aligned}
A^{-1} &= \begin{pmatrix} I & \bar{U}_0V_1^T \\ \bar{U}_1V_0^T & I \end{pmatrix}^{-1} \begin{pmatrix} D_0^{-1} & \\ & D_1^{-1} \end{pmatrix} \\
&= \begin{pmatrix} D_0^{-1} & \\ & D_1^{-1} \end{pmatrix} - \begin{pmatrix} \bar{U}_0 & \\ & \bar{U}_1 \end{pmatrix} \begin{pmatrix} I & V_1^T\bar{U}_1 \\ V_0^T\bar{U}_0 & I \end{pmatrix}^{-1} \begin{pmatrix} & V_1^TD_1^{-1} \\ V_0^TD_0^{-1} & \end{pmatrix} \\
&= \begin{pmatrix} D_0^{-1} & \\ & D_1^{-1} \end{pmatrix} - \begin{pmatrix} \bar{U}_0 & \\ & \bar{U}_1 \end{pmatrix} \begin{pmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{pmatrix} \begin{pmatrix} & V_1^TD_1^{-1} \\ V_0^TD_0^{-1} & \end{pmatrix} \\
&= \begin{pmatrix} D_0^{-1} - \bar{U}_0B_{00}V_1^TD_1^{-1} & -\bar{U}_0B_{01}V_0^TD_0^{-1} \\ -\bar{U}_1B_{10}V_1^TD_1^{-1} & D_1^{-1} - \bar{U}_1B_{11}V_0^TD_0^{-1} \end{pmatrix}
\end{aligned}$$

Algorithm 1 Inverse of H-matrix

(1) Recursively compute D_0^{-1} and D_1^{-1}
(2) Compute matrix-matrix product $\bar{U}_0 = D_0^{-1}U_0$, $\bar{U}_1 = D_1^{-1}U_1$, $V_0^T D_0^{-1}$ and $V_1^T D_1^{-1}$
(3) Compute the inverse of the small matrix, i.e. $B_{00}, B_{01}, B_{10}, B_{11}$
(4) Update D_0^{-1} and D_1^{-1} by $-\bar{U}_0 B_{00} V_1^T D_1^{-1}$ and $-\bar{U}_1 B_{11} V_0^T D_0^{-1}$ respectively

2.1 Algorithm and Computational Cost

Given D_0^{-1} and D_1^{-1}, each matrix-matrix product in step (2) costs $O(r^2\, n \log_2 n)$. Also to compute the inverse of the small matrix of size $2r \times 2r$ is very cheap. The last step is to update an H-matrix with a low-rank matrix as in step (4), which will be further investigated.