Inverse Of Scattering Matrix By Using Recursive Sherman-Morrison Formula

Hafiz Muhammad Fahad

Wednesday 14th February, 2018

1 Introduction

With the usual notation we write the one-group transport equation as

$$(\Omega.\nabla + \sigma)\psi = H\psi + \mathbb{S}$$

Where $H\psi$ is the scattering term.
Discretization of the transport equation in the angular variable based on the use of an angular quadrature formula.

$$\frac{1}{4\Pi}\int d\Omega.f(\Omega) = \sum_d \omega f_d$$

Where: f_d and $f(\Omega_d)$ is the number of directions in the quadrature.
The Discrete ordinates approximation is derived by considering the transport equation for the discrete set of directions $S_N[N_d] = set\{\Omega_d\}$ in the quadrature formula,

$$(\Omega_d.\nabla + \sigma)\psi_d = M\Sigma_d\Phi + \mathbb{S}_d, \Omega_d \in \mathbb{S}_d$$

Where
σ =total cross section
$\mathbb{S}(\Omega)$ =external source
Φ =angular flux moments
$M[N_d, N_k] = mat\{M_{dm} = A_m(\Omega_d)\}$ is called moment-to-discrete matrix.
The final form for the DOA is obtained by using the quadrature formula Q_N to approximate the angular flux moments $\Phi = (A, \psi) = \sum_d A(\omega_d)\psi = D\psi$
where

$$D[N_k, N_d] = mat\{M_{md} = \omega_d A_m(\Omega_d)\} \tag{1}$$

is called discrete-to-moment matrix.
A is the truncated kernel containing all the real spherical harmonics $A_{kl}(\Omega)$

1.1 Sherman-Morrison formula

Suppose A matrix is an invertible square matrix and u, v are column vectors. Suppose furthermore that $1+v^TA^{-1}u \neq 0$, then the Sherman-Morrison formula states that

$$(A+uv^T)^{-1} = A^{-1} + \frac{A^{-1}uv^TA^{-1}}{1+v^TA^{-1}u},$$

where uv^T is the outer product of two vectors u and v .

1.2 Recursive Formula

Let us have two matrices A and B. where A is an invertible square matrix probably sparse matrix and B is a square, possibly singular matrix. We can find $(A+B)^{-1}$ by using the Sherman-Morrison formula recursively.

The formula applies if we have a matrix A and u,v are column vectors, but in our case we have two matrices A and B so we can follow these steps:

1. Decompose B. Applying SVD we get

   $$B = U\Sigma V^*$$

   where

   U is a $m \times m$, unitary matrix (unitary matrices are orthogonal matrices)

   Σ is a diagonal $m \times n$ matrix with non-negative real numbers on the diagonal

   V is a $n \times n$, unitary matrix.

   V^* is the conjugate transpose of the $n \times n$ unitary matrix V, thus also unitary.

   The diagonal entries σ_i of Σ are known as the singular values of B. A common convention is to list the singular values in descending order. In this case, the diagonal matrix, Σ, is uniquely determined by B (though not the matrices U and V).

2. After that we use a separable method. The SVD can be thought of as decomposing a matrix into a weighted, ordered sum of separable matrices. By separable, we mean that a matrix C can also be written as an outer product of two vectors $C = u \otimes v$, or, in coordinates, $C_{ij} = u_i v_j$. Specifically, the matrix B can be decomposed as:

   $$B = \sum_i C_i = \sum_i \sigma_i u_i \otimes v_i^\dagger$$

   Here u_i and v_i are the i-th columns of the corresponding SVD matrices, σ_i are the ordered singular values, and each C_i is separable.

3. Finally we use recursive relation of the formula: let $T_0^{-1} = A^{-1}$ and let $T_n^{-1} = (A+B)^{-1}$ we perform the recursive iteration:

$$T_i^{-1} = T_{i-1}^{-1} - \frac{\sigma_i T_{i-1}^{-1} u_i v_i^T T_{i-1}^{-1}}{1 + \sigma_i v_i^T T_{i-1}^{-1} u_i} \qquad \text{for } i = 1 \text{ to } n$$

Here u_i and v_i are the i-th columns and σ_i are the ordered singular values of the corresponding SVD matrices of B. And n is the rank of the matrix B.

1.3 Block Matrix

Previous recursive equation is for normal matrices.but in our case we have a block matrices.As we know that A is a block diagonal matrix we can easily find inverse of A by inverting all the blocks in diagonal saperatly. so the recursive equation become :

for i=1 to n

$$T_i^{-1} = T_{i-1}^{-1} - \frac{\sigma_i T_{(i-1)}^{-1} u_i v_i^T T_{(i-1)}^{-1}}{I + \sigma_i v_i^T T_{(i-1)}^{-1} u_i}$$

n = num of block colum in matrices

u_i v_i are the i-th block columns and σ_i are the block ordered singular values of the corresponding SVD matrices of B.
Since in the denominator term $(I + \sigma_i v_i^T T_{(i-1)}^{-1} u_i)$ the answer will be a block size matrix.

So,

$$\text{denominator} = (I + \sigma_i v_i^T T_{(i-1)}^{-1} u_i)^{-1} = \frac{1}{(I + \sigma_i v_i^T T_{(i-1)}^{-1} u_i)}$$

So the final recursive formula is :

for i=1 to n

$$T_i^{-1} = T_{i-1}^{-1} - \sigma_i T_{(i-1)}^{-1} u_i v_i^T T_{(i-1)}^{-1} * denominator$$

EXAMPLE 1

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}, B = \begin{bmatrix} 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 \\ 3 & 0 & 4 & 0 \\ 0 & 3 & 0 & 4 \end{bmatrix}$$

Now we apply SVD to the matrix B we get

$$\sigma = \begin{bmatrix} 5.465 & 0 & 0 & 0 \\ 0 & 5.000 & 0 & 0 \\ 0 & 0 & 0.3660 & 0 \\ 0 & 0 & 0 & 0.000 \end{bmatrix},$$

$$u = \begin{bmatrix} -0.4046 & 0 & -0.9145 & 0 \\ 0 & -0.4472 & 0 & -0.8944 \\ -0.9145 & 0 & 0.4046 & 0 \\ 0 & -0.8944 & 0 & 0.4472 \end{bmatrix}$$

$$vt = \begin{bmatrix} -0.5760 & 0 & 0.8174 & 0 \\ 0 & -0.4472 & 0 & -0.8944 \\ -0.8174 & 0 & -0.5760 & 0 \\ 0 & -0.8944 & 0 & 0.4472 \end{bmatrix}$$

finally apply Sherman-Morrison formula

$$(A+B)^{-1} = \begin{bmatrix} 1.0000 & 0 & -0.3333 & 0 \\ 0 & 0.7500 & 0 & -0.2500 \\ -0.5000 & 0 & 0.3333 & 0 \\ 0 & -0.2500 & 0 & 0.2500 \end{bmatrix}$$

in this case the answer is correct

ORIGNAL PROBLEM

Now come to our orignal problem for Scattering Matrix. Here u and vt are the moment-to-discrete matrix (M) and discrete-to-moment matrix (D) respectivly. The blocks of A is the discretization of the gradient term and the σ are $moments$.

$$A = \begin{bmatrix} A_1 & 0 & ..... & 0 & 0 \\ 0 & A_1 & 0 & 0 & 0 \\ . & . & . & 0 & . \\ . & . & . & . & . \\ 0 & 0 & 0 & 0 & A_1 \end{bmatrix}, B = \begin{bmatrix} 0.4375 & 0.3125 & 0.0625 & 0.1875 \\ 0.3125 & 0.4375 & 0.1875 & 0.0625 \\ 0.0625 & 0.1875 & 0.4375 & 0.3125 \\ 0.1875 & 0.0625 & 0.3125 & 0.4375 \end{bmatrix}$$

$$\sigma = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.250 \end{bmatrix}, u = \begin{bmatrix} 1 & 1.7321 & 1.7321 \\ 1 & 1.7321 & -1.7321 \\ 1 & -1.7321 & -1.7321 \\ 1 & -1.7321 & 1.7321 \end{bmatrix}$$

$$, vt = \begin{bmatrix} 0.2500 & 0.2500 & 0.2500 & 0.2500 \\ 0.1443 & 0.1443 & -0.1443 & -0.1443 \\ 0.1443 & -0.1443 & -0.1443 & 0.1443 \end{bmatrix}$$

after apply block recursive farmula in this problem we get

$$(A+B)^{-1} = \begin{bmatrix} 0.4194 & -0.0528 & -0.0028 & -0.0306 \\ -0.0528 & 0.4194 & -0.0306 & -0.0028 \\ -0.0028 & -0.0306 & 0.4194 & -0.0528 \\ -0.0306 & -0.0028 & -0.0528 & 0.4194 \end{bmatrix}$$

keep in mind that in our problem we have block matrices. So, in all these matrices every single entry expand into a diagonal block matrix 3x3 order,like

$$0.4194 = \begin{bmatrix} 0.4194 & 0 & 0 \\ 0 & 0.4194 & 0 \\ 0 & 0 & 0.4194 \end{bmatrix}$$

1.4 Implimentation in C++

We have implemented our code in C++ by using Deal.ii library.

First, since A has a same diagonal blocks matrix so,we can find easily the A^{-1} by inverting only block of A. Then made diffrent operator functions like $+, -, *, /$. In operator $*$ there are two types of multiplications e.g; Matrix-Matrix multiplication and matrix to block multiplication (e.g: when we multiply inverse of denominator term (block size matrix(3x3)) to Numinator whole matrix). The inverse of denominator block will multiply with each block of numinator matrix.

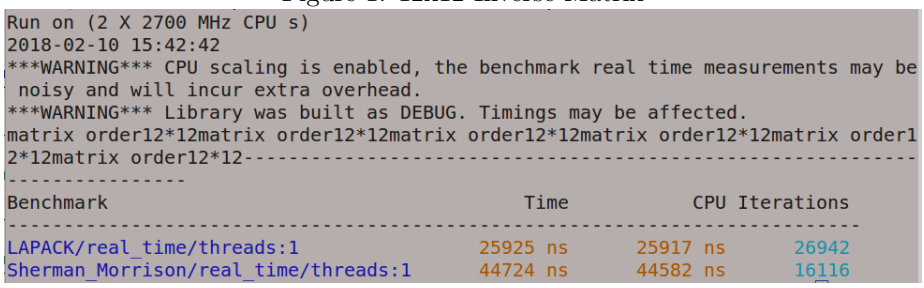
We made another function that convert every element of u and vt into diagonal blocks. Then Impliment a Sherman-Morrison Recursive formula to find the inverse. Then also find the inverse of matrix by using Lapack.

After that we implement a benchmark to check a performance of algorithm Shown below

.

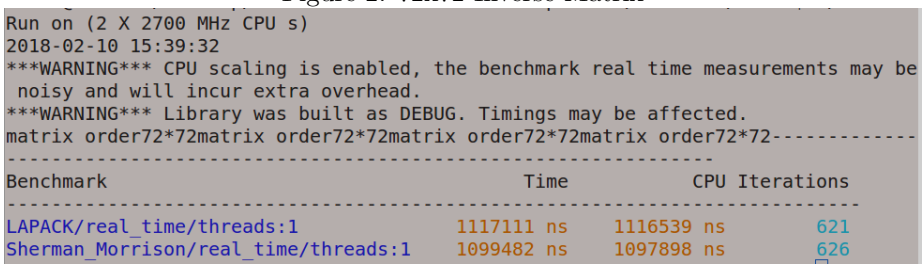
Results from C++

Figure 1: 12x12 Inverse Matrix



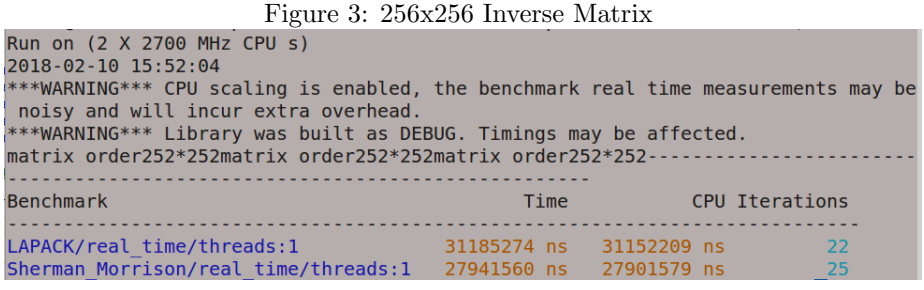
```
Run on (2 X 2700 MHz CPU s)
2018-02-10 15:42:42
***WARNING*** CPU scaling is enabled, the benchmark real time measurements may be
 noisy and will incur extra overhead.
***WARNING*** Library was built as DEBUG. Timings may be affected.
matrix order12*12matrix order12*12matrix order12*12matrix order12*12matrix order1
2*12matrix order12*12------------------------------------------------------------
---------------
Benchmark                                      Time           CPU Iterations
------------------------------------------------------------------------------
LAPACK/real_time/threads:1                 25925 ns      25917 ns      26942
Sherman_Morrison/real_time/threads:1       44724 ns      44582 ns      16116
```

Figure 2: 72x72 Inverse Matrix



```
Run on (2 X 2700 MHz CPU s)
2018-02-10 15:39:32
***WARNING*** CPU scaling is enabled, the benchmark real time measurements may be
 noisy and will incur extra overhead.
***WARNING*** Library was built as DEBUG. Timings may be affected.
matrix order72*72matrix order72*72matrix order72*72matrix order72*72-------------
--------------------------------------------------------------
Benchmark                                      Time           CPU Iterations
------------------------------------------------------------------------------
LAPACK/real_time/threads:1               1117111 ns    1116539 ns        621
Sherman_Morrison/real_time/threads:1     1099482 ns    1097898 ns        626
```

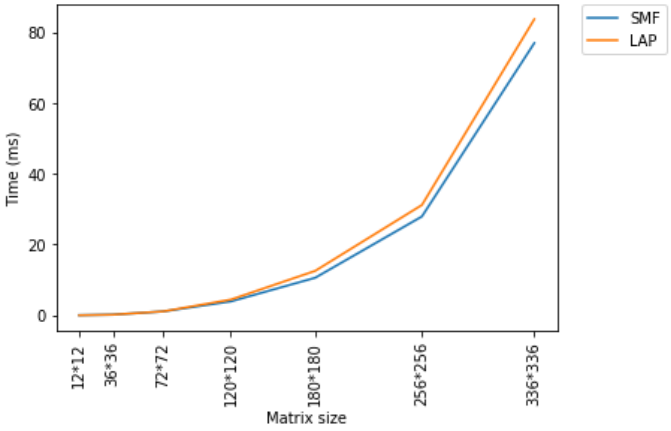
Figure 3: 256x256 Inverse Matrix



```
Run on (2 X 2700 MHz CPU s)
2018-02-10 15:52:04
***WARNING*** CPU scaling is enabled, the benchmark real time measurements may be
 noisy and will incur extra overhead.
***WARNING*** Library was built as DEBUG. Timings may be affected.
matrix order252*252matrix order252*252matrix order252*252------------------------
-------------------------------------------------
Benchmark                                      Time           CPU Iterations
------------------------------------------------------------------------------
LAPACK/real_time/threads:1              31185274 ns   31152209 ns         22
Sherman_Morrison/real_time/threads:1    27941560 ns   27901579 ns         25
```

Figure 4: Graph

```
Diffrences between Time in 'ms.
 [-0.018799 -0.01425   0.017629  0.531331  1.947825  3.243714  6.743211]
```



1.5 Conclusion

Lapack inverse is good for small matrix but Sherman-Morrison Recursive formula is good for a huge matrices ,we clearly see above that when we increase the matrix size the sherman-Morrison farmula become more efficient. Above graph shows a comperison between Sherman-Morrison and Lapack inverse and also time diffrence between two methods.