Hierarchical inversion of Toeplitz and two-level Toeplitz matrices

Christopher K. Turnes

Thursday 24th April, 2014

1 Overview

This work is concerned with implementing hierarchical matrix inversion methods to solve scalar and multi-level Toeplitz systems. Hierarchical methods exploit the structure of matrices that have low-rank submatrices away from the main diagonal. While this is not the case for Toeplitz systems, simple Fourier transformations may be used to obtain so-called "Loewner forms" of Toeplitz systems that are indeed hierarchical. In the case of multi-level systems, a permutation is also required.

2 Transformations of Toeplitz systems

Hierarchical methods are not directly useful for scalar and multi-level Toeplitz matrices, as they do not have low-rank off-diagonal blocks. However, using simple Fourier transformations, it is possible to show that these systems can be efficiently translated into alternate forms that lend themselves well to hierarchical structure. Since these transformations involve only Fourier operators, they providing convenient and effective alternatives to directly exploiting Toeplitz structure.

2.1 Scalar Toeplitz transformation

While most Toeplitz inversion algorithms use shift operators to exploit the apparent structure of the system, it is possible to make more subtle use of structure. Specifically, one may employ Fourier-like operators to transform a Toeplitz system $Tx = b$ into a new system $Uy = c$ that has other advantageous properties. Notably, the transformed matrices U have a highly compressible structure that allows the proxy system $Uy = c$ to be solved efficiently and stably *in lieu* of the original problem through the use of hierarchical methods. With the transformed system solved, the original solution x can be recovered from y in only $\mathcal{O}(n \log n)$ operations by inverting the transformation.

Two prominent examples of this approach exist. The first, developed by Martinsson and Rokhlin [8], transforms the Toeplitz matrix T using the 2-D Fourier transform. The off-diagonal blocks of the transformed matrix have low rank, a property shown in Rokhlin's previous work on non-uniform Fourier transforms [2]. This rank structure is used to solve the system in a fashion inspired by the Fast Multipole Method (FMM) [6, 1], yielding an inversion complexity of $\mathcal{O}(n \log n)$.

The second existing compression algorithm transforms the Toeplitz matrix into a generalized Cauchy matrix with coefficients of the form

$$L_{ij} = \frac{u_i^T v_j}{c_i - d_j} \tag{1}$$

for some generators $u_i, v_j \in \mathbb{C}^2$, and denominator nodes c_i and d_j, with $c_i \neq d_j$ for all i and j. The transformed matrix is referred to as the "Loewner form" of the Toeplitz matrix. Similar to Rokhlin's transformed matrix, the Loewner form has off-diagonal blocks with low rank. However, it is more advantageous because its coefficients have been derived in closed-form.

The specific Fourier transformation to derive the Loewner form was first given by Fiedler, who showed how Hankel matrices (which are column-reversed Toeplitz matrices) can be transformed into their Loewner forms [3]. The analogous result for Toeplitz structure surfaced in later works [9, 5, 4, 7], and is given as follows.

Theorem 1. *Let $T = [a_{i-j}]$ be an $n \times n$ scalar Toeplitz matrix. Define the Fourier-like matrices*

$$\left(\mathbf{F}_{\pm 1}^{(n)}\right)_{j,k} = \frac{1}{\sqrt{n}} (\omega_{j-1}^{\pm})^{k-1}$$

where $\omega = \mathbf{e}^{\mathbf{j}2\pi/n}$, $\eta = \mathbf{e}^{\mathbf{j}\pi/n}$, $\omega_k^+ = \omega^k$, and $\omega_k^- = \omega_k^+ \eta$. Similarly, define two Fourier-like transformations

$$\check{a}_i = \frac{1}{n} \sum_{k=1-n}^{n-1} a_k (\omega_{i-1}^-)^k, \tag{2}$$

$$\hat{a}_i = \frac{1}{n} \sum_{k=1-n}^{n-1} a_k (\omega_{i-1}^+)^k, \tag{3}$$

The matrix $\Phi = (\mathbf{F}_{-1}^{(n)})^H T \mathbf{F}_{+1}^{(n)}$ can be expressed as $\Phi = L\Delta$, where $\Delta = \operatorname{diag}(\omega_0^+, \ldots, \omega_{n-1}^+)$ and L is a **Loewner matrix** *with coefficients*

$$L = \left[\frac{\check{a}_i - \hat{a}_j}{\omega_{i-1}^- - \omega_{j-1}^+} \right]_{i,j=1}^{n}. \tag{4}$$

2.2 Two-level Toeplitz transformation

As described in Section 2, one method of inverting scalar Toeplitz matrices is to use Fourier-like operators to transform them into corresponding Loewner forms. The same approach can be taken with two-level Toeplitz matrices, replacing the 1-D Fourier operators with 2-D analogs. This type of transformation allows one to work with two-level Loewner matrices in the place of two-level Toeplitz systems.

Specifically, as was shown in [10], there is the following result.

Theorem 2. *Let T be an (m,n) two-level Toeplitz matrix with blocks $A_{i-j} = [a_{i-j,k-\ell}]$. Define the Fourier-like matrices*

$$\begin{aligned}\left(\mathbf{F}_{\pm 1}^{(n)}\right)_{j,k} &= \frac{1}{\sqrt{n}}(\omega_{j-1}^{\pm})^{k-1} \quad \textit{and} \\ \left(\mathbf{F}_{\pm 1}^{(m)}\right)_{j,k} &= \frac{1}{\sqrt{m}}(\phi_{j-1}^{\pm})^{k-1},\end{aligned}$$

where $\omega = \mathbf{e}^{\mathbf{j}2\pi/n}$, $\eta = \mathbf{e}^{\mathbf{j}\pi/n}$, $\phi = \mathbf{e}^{\mathbf{j}2\pi/m}$, $\nu = \mathbf{e}^{\mathbf{j}\pi/m}$, $\omega_k^+ = \omega^k$, $\omega_k^- = \omega_k^+\eta$, $\phi_k^+ = \phi^k$, and $\phi_k^- = \phi_k^+\nu$. Define four Fourier-like transformations

$$[\Theta_{00}]_{i,j} = \frac{1}{mn}\sum_{k=1-n}^{n-1}\sum_{s=1-m}^{m-1} a_{k,s}(\omega_{i-1}^-)^k(\phi_{j-1}^-)^s, \tag{5}$$

$$[\Theta_{01}]_{i,j} = -\frac{1}{mn}\sum_{k=1-n}^{n-1}\sum_{s=1-m}^{m-1} a_{k,s}(\omega_{i-1}^-)^k(\phi_{j-1}^+)^s, \tag{6}$$

$$[\Theta_{10}]_{i,j} = -\frac{1}{mn}\sum_{k=1-n}^{n-1}\sum_{s=1-m}^{m-1} a_{k,s}(\omega_{i-1}^+)^k(\phi_{j-1}^-)^s, \tag{7}$$

$$[v_{11}]_{i,j} = \frac{1}{mn}\sum_{k=1-n}^{n-1}\sum_{s=1-m}^{m-1} a_{k,s}(\omega_{i-1}^+)^k(\phi_{j-1}^+)^s. \tag{8}$$

The matrix $\tilde{\Phi} = (\mathbf{F}_{-1}^{(n)} \otimes \mathbf{F}_{-1}^{(m)})^H T(\mathbf{F}_{+1}^{(n)} \otimes \mathbf{F}_{+1}^{(m)})$ can be expressed as $\tilde{\Phi} = L\Delta$, where Δ is an $mn \times mn$ diagonal matrix with main diagonal given by

$$\Delta = \begin{bmatrix}\omega_0^+ \\ \vdots \\ \omega_{n-1}^+\end{bmatrix} \otimes \begin{bmatrix}\phi_0^+ \\ \vdots \\ \phi_{m-1}^+\end{bmatrix}$$

and L is a **block Loewner matrix** *with blocks*

$$L = \left[\frac{J_i - K_j}{\omega_{i-1}^- - \omega_{j-1}^+}\right]_{i,j=1}^{n}, \tag{9}$$

where the matrices J_i and K_j are themselves Loewner matrices given by

$$J_i = \left[\frac{[\Theta_{00}]_{i,\ell} - [\Theta_{01}]_{i,r}}{\phi_{\ell-1}^- - \phi_{r-1}^+}\right]_{\ell,r=1}^{m}, \qquad K_j = \left[\frac{[\Theta_{10}]_{j,\ell} - [\Theta_{11}]_{j,r}}{\phi_{\ell-1}^- - \phi_{r-1}^+}\right]_{\ell,r=1}^{m}. \tag{10}$$

Matrices of this form are **two-level Loewner matrices**.

The two-level Loewner forms of multi-level Toeplitz matrices have structure that resembles hierarchical matrices, but the off-diagonal blocks are not low rank. However, by permuting the columns and rows of the two-level Loewner form with a Morton/Z-order curve reordering, the matrix much more closely resembles its scalar counterpart. This reordering costs no flops, and allows hierarchical methods to be used in solving

two-level Toeplitz systems.

3 Numerical experiments

A series of numerical experiments were conducted to evaluate the use of hierarchical methods for Toeplitz systems. First, details of the implementation and code structure are provided. Next, the results of experiments detailing the performance for scalar Toeplitz inversion problems are given. Finally, the results of preliminary experiments involving two-level Toeplitz inversion problems are given.

3.1 Implementation

The bulk of the code is implemented in `C++` with `MEX` interfaces to `MATLAB`, with the remainder implemented as `m`-files. Specifically, three `MEX` gateway functions are implemented:

- `dense2hm`: Converts a supplied dense matrix to a hierarchical structure;
- `hmtimes`: Multiplies an input matrix by a hierarchical matrix; and
- `hminv`: Computes the inverse of a hierarchical matrix.

For each gateway function, an `m`-file is provided that supplies information to the user when invoked by the `help` command.

The hierarchical matrix data is stored in a structure containing two fields. The first is `.data`, which contains a flat data array supplying all of the necessary data to reconstruct the hierarchical matrix. The second is `.meta`, an $11 \times p$ integer matrix that supplies metadata used to reconstruct the hierarchical matrix. This data format allows the code to avoid computing the hierarchical structure from a dense matrix during each function call.

Several `m`-files are also provided:

- `compile_kernel.m`: Assuming `mwblas` and `mwlapack` have been added to the `mexopts` configuration script, running this file will compile the `MEX` gateway functions;
- `lrsvd.m`: Computes the SVD of a low-rank matrix;
- `morton2.m`: Returns an array of indices corresponding to a Morton re-ordering;
- `tltoeptrans.m`: Returns the (dense) two-level Loewner form of a two-level Toeplitz matrix;
- `tmult.m`: Multiplies an input matrix by a scalar Toeplitz matrix;

Table 1: *Results for the scalar Toeplitz experiments. "Avg. compression" refers to the average rate of information compression; i.e., the ratio of the amount of data storage in the hierarchical structure to the amount of data storage necessary for a dense representation of the matrix. Note that this ratio can become greater than one if the supposed low-rank blocks have rank larger than half of its maximum value.*

n	Time to compute inverse	Time to apply inverse	Avg. compression	Avg. residual
2^7	13.0 ms	0.55 ms	1.11	2.22E-13
2^8	56.6 ms	1.06 ms	0.83	2.31E-12
2^9	233 ms	4.79 ms	0.56	7.05E-12
2^{10}	862 ms	12.1 ms	0.36	2.97E-11
2^{11}	3.22 s	30.2 ms	0.22	6.15E-11
2^{12}	12.8 s	71.2 ms	0.013	3.59E-10
2^{13}	60.7 s	174 ms	0.0792	3.64E-10

- `toeptrans2hmat.m`: Converts a scalar Toeplitz matrix to a hierarchical matrix; and
- `tsolve_hier.m`: Solves a scalar Toeplitz system of equations using hierarchical matrix inversion.

3.2 Scalar Toeplitz inversion

Two `m`-files are provided to perform experiments for scalar Toeplitz inversion. The first is `example.m`, which establishes an example problem whose system matrix is the (Toeplitz) Gramian of a non-uniform Fourier matrix. This experiment records the time it takes to compute the hierarchical structure of the inverse matrix as well as the time required to apply it to future inputs. The average residual over several trials of the inverse computation, $\epsilon = \|x - (T^{-1}(Tx))\|$ is also returned.

The second `m`-file, `batch_scalar.m`, calls `example.m` over a range of matrix sizes ($n = 2^7 = 128$ to $n = 2^{13} = 8192$) to view how these quantities evolve with the problem size. A sample run of this script gives the results in Table 1.

3.3 Two-level Toeplitz inversion

Several important changes must be made to the code before it can be useful for two-level Toeplitz problems. Namely, while the off-diagonal blocks indeed have low rank, a considerable savings could be realized if they were also partitioned with a quad-tree structure. Such a strategy would be beneficial; if the low-rank blocks are partitioned, the off-diagonal blocks of the partitioned tend to be considerably lower rank than the entire submatrix. Therefore, the maximum amount of compression for these matrices is not realized with a standard hierarchical structure.

However, given the hierarchical code in place, one can still establish a set of simulations to observe the amount of compression achieved for the Loewner forms of two-level Toeplitz matrices. These simulations

Table 2: *Compression ratios achieved with hierarchical representations of Loewner forms of (m, n) two-level Toeplitz matrices for various values of m and n. The compression ratio is computed as the ratio of the amount of data storage in the hierarchical structure to the amount of data storage necessary for a dense representation of the matrix. Entries with an asterisk are not computed due to memory limitations.*

m/n	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}	2^{11}	2^{12}
2^0	1	1	1	1	1	1	1	1	0.85	0.58	0.37	0.23	0.14
2^1	1	1	1	1	1	1	1	1	0.83	0.56	0.36	0.23	*
2^2	1	1	1	1	1	1	1	1	0.82	0.55	0.35	*	*
2^3	1	1	1	1	1	1	1	1	0.80	0.54	*	*	*
2^4	1	1	1	1	1	1	1	1	0.78	*	*	*	*
2^5	1	1	1	1	1	1	1	1	*	*	*	*	*
2^6	1	1	1	1	1	1	1	*	*	*	*	*	*
2^7	1	1	1	1	1	1	*	*	*	*	*	*	*
2^8	0.85	0.83	0.82	0.80	0.78	*	*	*	*	*	*	*	*
2^9	0.58	0.56	0.55	0.54	*	*	*	*	*	*	*	*	*
2^{10}	0.37	0.36	0.35	*	*	*	*	*	*	*	*	*	*
2^{11}	0.23	0.23	*	*	*	*	*	*	*	*	*	*	*
2^{12}	0.14	*	*	*	*	*	*	*	*	*	*	*	*

are conducted with the `m`-file `batch_two_level.m`, and produce a series of (m, n) two-level Toeplitz matrices were constructued for varying values of m and n such that the overall matrix side length mn satisfied $mn \leq 2^{12}$ to comply with memory requirements, For each pair of values (m, n), the ratio of the amount of storage necessary for the hierarchical form *versus* a dense representation was computed as in Section 3.2. The results are given in Table 2.

As is clear from the results in Table 2, the hierarchical compression schemes are not useful until one of the dimension – either m or n – is sufficiently large. Interestingly, the scaling of the compression of the hierarchical scheme seems to follow the exact same patterns as in the scalar case for its largest dimension. In other words, a seemingly tight upper-bound for the compression ratio of an (m, n) two-level Loewner form can be obtained by looking at the compression ratio for a $k \times k$ scalar Loewner form, where $k = \max(m, n)$. Since only a limited range of sizes can be explored, it is unclear whether extending the second dimension would even further reduce the compression ratio, though the trends in the table entries seem to indicate that this may indeed be the case.

4 Future work

The major future steps for the work can be summarized as follows:

- Re-work the code to improve performance in general;
- Modify the hierarchical structure so that the low-rank blocks can also split in a quad-tree fashion; and
- Construct a hierarchical representation of the two-level Loewner form without an explicit construction

of the matrix itself.

References

[1] J. Carrier, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm for particle simulations*, SIAM Journal on Scientific and Statistical Computing **9** (1988), 669–686.

[2] A. Dutt and V. Rokhlin, *Fast Fourier transforms for nonequispaced data, ii.*, Applied Computational Harmonic Analysis **2** (1995), 85–10.

[3] M. Fiedler, *Hankel and Loewner matrices*, Linear Algebra and Its Applications **58** (1984), 75–95.

[4] I. Gohberg, T. Kailath, and V. Olshevsky, *Fast Gaussian elimination with partial pivoting for matrices with displacement structure*, Mathematics of Computation **64** (1995), 39–59.

[5] I. Gohberg and V. Olshevsky, *Complexity of multiplication with vectors for structured matrices*, Linear Algebra and Its Applications **202** (1994), 163–192.

[6] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, Journal of Computational Physics **73** (1987), 325–348.

[7] G. Heinig, *Inversion of generalized Cauchy matrices and other classes of structured matrices*, Linear Algebra for Signal Processing, vol. 69, pp. 63–81, Springer, 1995.

[8] P.G. Martinsson, V. Rokhlin, and M. Tygert, *A fast algorithm for the inversion of general Toeplitz matrices*, Computers and Mathematics with Applications **50** (2005), 741–752.

[9] V.Y. Pan, *On computations with dense structured matrices*, Mathematics of Computation **55** (1990), 179–190.

[10] C.K. Turnes, *Efficient solutions to Toeplitz-structured linear systems for signal processing*, Mathematics theses, May 2014.