PARALLEL MATRIX MULTIPLICATION BY GRAMIAN OF TOEPLITZ-BLOCK MATRIX

Gabriel Okša, Martin Bečka and Marián Vajteršic
Mathematical Institute, Department of Informatics,
Slovak Academy of Sciences, 840 00 Bratislava, Slovakia,
e-mail: (gabo, mato, marian)@ifi.savba.sk

Keywords: Toeplitz-block matrix, Gramian, displacement structure of matrix, generator, fast Fourier transform

Abstract

Let $C = (T_{ij}) \in \mathbb{C}^{Mm \times Nn}$ be a matrix of block order $M \times N$ where each block T_{ij} is a Toeplitz matrix of order $m \times n$, $m \geq n$. Such matrix arises, for example, in the total least squares formulation of the forward-backward linear prediction modelling of multidimensional signals.

Let us compute the d-dimensional subspace of right singular vectors of matrix C. When using some iterative method without the shift-and-invert operator (e.g., direct Chebyshev, Lanczos, etc.) for this partial singular value decomposition of C, the computationally most intensive task in each iteration step consists of matrix multiplication $Y = C^H C X$ where $X \in \mathbb{C}^{Nn \times d}$ is the iteration matrix, the columns of which approximate the right singular vectors.

The standard algorithm for the computation of Y is based on the associative law (i.e., $Y = C^H (CX)$) and on the embedding of each Toeplitz block T_{ij}, T_{ij}^H into a circulant. We present another algorithm based, first, on the computation of the generator of Toeplitz-like Gramian $C^H C$ from its displacement structure, and, second, on the use of generalized Gohberg-Semencul formula for the matrix multiplication $Y = C^H C X$. Both algorithms have the preparatory phase (circulants vs. generator), and the computational phase (matrix multiplication). For the fixed matrix C, the preparatory phase is computed only once, whereas the matrix multiplication is needed in each iteration step. The time complexity of the computational phase of standard algorithm is $O(m \log m)$, and that of the new one $O(n \log n)$, so that a considerable gain in speed can be expected whenever $m \gg n$ (i.e., when the Toeplitz blocks are "tall and thin").

We show how to parallelize both algorithms. Their parallel versions were implemented on SGI-Cray T3E parallel computer for various lengths of data segments (from 512 to 64000) and various numbers of processors (from 1 to 32). We report, for both algorithms, the profiling results, which confirm the expected gain in speed for the computational phase of new algorithm.

Acknowledgement: This research was supported by the NATO Collaborative Linkage Grant no. 977203 (Slovak Academy of Sciences, Bratislava, Slovakia, and University of Bergen, Bergen, Norway), and by the Project no. 2/1097/21 of the VEGA Grant Agency, Slovakia.