A Block Approximate Inverse With Cutoff Preconditioner For Semi-Sparse Linear Systems

Ikuro Yamazaki
Graduate School of Systems and Information Engineering, University of Tsukuba, Japan
yamazaki@mma.cs.tsukuba.ac.jp
Masayuki Okada
Department of Middleware & Platform Software, Hitachi, Ltd., Japan
okada@mma.cs.tsukuba.ac.jp
Hiroto Tadano, Tetsuya Sakurai
Department of Computer Science, University of Tsukuba, Japan
tadano@cs.tsukuba.ac.jp, sakurai@cs.tsukuba.ac.jp
Keita Teranishi
Department of Cray, Inc., USA
keita@cray.com

1 Introduction

In the recent nano-science simulations, the calculation of the energy often involves a solution of large sparse linear systems $A\boldsymbol{x} = \boldsymbol{b}$, where $A \in \mathbb{C}^{n\times n}$ is poorly-conditioned and relatively dense (a few hundred nonzero entries per row). For the solution of such linear systems on distributed memory multiprocessors, Krylov subspace methods preconditioned with sparse approximate inverse based on Frobenius norm minimization (SAI) appears very attractive because of the good parallel efficiency in both preconditioner construction and application. However, SAI is typically less effective at accelerating the convergence than the conventional preconditioning methods such as incomplete factors. In addition, SAI requires a huge computational cost in its construction when a large number of nonzero entries are kept in the approximate inverse matrix. We attempt to overcome these performance bottlenecks using a blocked version of Frobenius norm minimization to mitigate the side effect of the minimization process applied to individual columns of the approximate inverse. We also apply different drop-threshold schemes to achieve a huge reduction the computational costs of preconditioner construction and application at the cost of a small increase in iteration counts.

In the presentation, we are going to demonstrate how our approach improves the performance of SAI preconditioning.

2 SAI preconditioner and its block variant

In SAI preconditioner, the preconditioning matrix $M = [\boldsymbol{m}_1, \boldsymbol{m}_2, \cdots, \boldsymbol{m}_n] \approx A^{-1}$ is constructed by solving n independent least square problems:

$$\min_{\boldsymbol{m}_k} \|A\boldsymbol{m}_k - \boldsymbol{e}_k\|_2^2, \quad k = 1, 2, \cdots, n$$

where $\boldsymbol{e}_k \in \mathbb{R}^n$ is a k-th unit vector.

Block SAI preconditioner is proposed by Barnard and Grote [3] in order to improve the accuracy of the preconditioning matrix. In Block SAI preconditioner, the preconditioning

matrix $M = [M_1, M_2, \cdots, M_{\lceil n/l \rceil}]$ is constructed by solving $\lceil n/l \rceil$ independent least square problems:

$$\min_{M_k} \|AM_k - E_k\|_2^2, \quad k = 1, 2, \cdots, \lceil n/l \rceil$$

where l is a block size, E_k is a submatrix of the identity matrix I such that $I = [E_1, E_2, \cdots, E_{\lceil n/l \rceil}]$. The sparsity pattern of the preconditioning matrix M is decided by the following strategy:

$$A' = [a'_{ij}] \quad a'_{ij} = \begin{cases} 1, & (|a_{ij}| > \varepsilon \text{ or } \ i = j) \\ 0, & \text{otherwise} \end{cases}$$
$$\text{spy}(A') = \text{spy}(M)$$

where ε is a real value and "spy" denotes the sparsity pattern. This method is similar to ParaSails [2].

3 Block SAI with Cutoff (BSAIC) preconditioner

In our preconditioning, the Cutoff strategy is applied to the coefficient matrix A in order to reduce the computation time of least square problems which appear in Block SAI. Firstly, the approximate coefficient matrix A_c is generated by the following Cutoff strategy:

$$A_c = [\tilde{a}_{ij}] \qquad \tilde{a}_{ij} = \begin{cases} a_{ij}, & (|a_{ij}| > \theta \text{ or } \ i = j) \\ 0, & \text{otherwise} \end{cases}$$

where θ is a real value. Then, least square problems with the approximate matrix A_c

$$\min_{M_k} \|A_c M_k - E_k\|_2^2, \quad k = 1, 2, \cdots, \lceil n/l \rceil$$

are solved. The matrix $M = [M_1, M_2, \cdots, M_{\lceil n/l \rceil}]$ is employed as the preconditioning matrix. We call this preconditioner BSAIC preconditioner.

4 Numerical experiments

In this section, BSAIC preconditioner is compared with SAI and Block SAI by numerical experiments. All experiments were carried out by MATLAB 7.4 on MacBook (CPU: Intel Core 2 Duo 2.0GHz, Memory: 2.0Gbytes, OS: Mac OS 10.5.6). The test problem was solved by the preconditioned BiCGSTAB method. The maximum iteration counts were set to 1,000, and the stopping criterion for the relative residual was 10^{-10}. The initial guess $\boldsymbol{x}_0$ was set to $\mathbf{0}$ and all elements of $\boldsymbol{b}$ were set to 1. In this experiment, the value of ε and the block size l were set to 10^{-4} and 20, respectively.

The test matrix was derived from the computation of the molecular orbitals of DNA. The matrix size was 1,980 and the number of nonzero elements was 3,490,478. The pattern of nonzero elements of the coefficient matrix and molecular illustration of DNA are presented in Figures 1 and 2, respectively.

Table 1 shows the results for DNA. SAI and Block SAI took about 1,200 seconds and 315 seconds, respectively. By contrast, BSAIC took 24 seconds when $\theta = 1.0 \times 10^{-2}$.

References

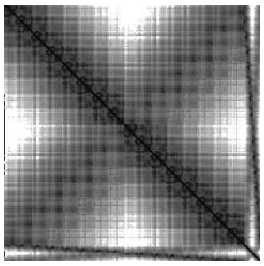

Figure 1: The pattern of nonzero elements in A.

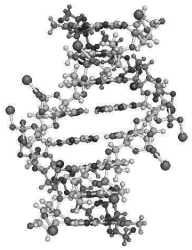

Figure 2: Molecular illustration of DNA.

Table 1: Results for DNA.

Preconditioner		The number of iterations	Wall clock time [sec]		
			Preconditioning	Iteration	Total
SAI		70	1192.46	13.09	1205.55
Block SAI		30	308.27	6.09	314.36
BSAIC	$\theta = 1.0 \times 10^{-4}$	30	48.96	6.03	54.99
	$\theta = 1.0 \times 10^{-3}$	53	18.97	10.88	29.85
	$\theta = 1.0 \times 10^{-2}$	95	4.79	19.27	24.06
	$\theta = 1.1 \times 10^{-2}$	162	4.34	30.81	35.15
	$\theta = 1.2 \times 10^{-2}$	166	4.01	31.77	35.78
	$\theta = 1.5 \times 10^{-2}$	301	3.42	58.62	62.04

[1] M. Benzi, and M. Tuma: A sparse approximate inverse preconditioner for nonsymmetric linear systems. *SIAM J. Sci. Comput*, 19:968–994, 1998.

[2] E. Chow: Parallel implementation and practical use of sparse approximate inverse preconditioners with a priori sparsity patterns. *Int. J. High Perf. Comput. Appl.*, 15:56–74, 2001.

[3] S. Barnard and M. Grote: A block version of the SPAI preconditioner. *in Proc. of the 9th SIAM conf. on Parallel Process. for Sci. Comput*, San Antonio, TX, 1999.