Performance Improvement of Symmetric Linear System Solver in Shielding Current Analysis of HTS Thin Film: Application of $\mathcal{H}$-Matrix-Based Preconditioner[*)]

Ayumu SAITOH

Yamagata University, Yonezawa, Yamagata 992-8510, Japan

(Received 10 January 2022 / Accepted 20 May 2022)

The conjugate gradient (CG) method with the acceleration technique of using both $\mathcal{H}$-matrix arithmetic and $\mathcal{H}$-matrix-based preconditioning is applied to the linear system that appeared in the shielding current analysis of the uncracked high-temperature superconducting film and its performance is investigated numerically. The computational results show that the proposed acceleration technique is extremely effective for improving the speed of the CG method.

Keywords: finite element method, Newton method, conjugate gradient method, preconditioning, $\mathcal{H}$-matrix method, superconducting film

DOI: 10.1585/pfr.17.2401089

1. Introduction

For the development of superconducting applications, it is important to analyze the shielding current density in a high-temperature superconductor (HTS). To this end, an initial-boundary-value problem of the shielding current density must be solved numerically. Some numerical methods based on the current-vector-potential method have been proposed [1–3].

By discretizing with respect to both time and space and further applying Newton method, the initial-boundary-value problem of the shielding current density can be reduced to the problem in which the linear system is solved at each time step and iteration cycle of the Newton method. In this study, the iteration cycle of the Newton method is called the Newton cycle. A direct method has been adopted as the linear-system solver since the resulting linear system has a symmetric dense coefficient matrix [1, 2].

On the other hand, when the Krylov subspace method, such as the conjugate gradient (CG) method, is applied to a linear system with the symmetric dense matrix, a matrix-vector product must be calculated at each iteration for the linear-system solver. The operation count required for the matrix-vector product becomes $O(N^2)$, where N indicates the number of unknowns in the linear system. Therefore, the large operation count of the matrix-vector product is a disadvantage when the Krylov subspace method is used.

In order to resolve the above difficulty, the $\mathcal{H}$-matrix method [4–6] was proposed and has yielded excellent results [7, 8]. By using the $\mathcal{H}$-matrix method, the matrix-vector product can be calculated at a high speed. A high-speed shielding current analysis might be executed if an acceleration technique based on the $\mathcal{H}$-matrix method were incorporated into the linear-system solver.

The purpose of the present study is to apply the $\mathcal{H}$-matrix-based acceleration technique to the linear-system solver in the shielding current analysis of an uncracked HTS film and to investigate its performance numerically.

2. Shielding Current Analysis

2.1 Governing equation

In this study, we assume that an uncracked HTS film with a rectangle cross section Ω over the thickness is exposed to a time-varying magnetic field. In addition, we adopt the Cartesian coordinate system $\langle O : \boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z \rangle$ in which the z-axis is taken as the thickness direction. Furthermore, the origin is the centroid of the HTS film and $\boldsymbol{x}$ is a position vector of two points in the xy plate.

Under the thin-plate approximation, the time evolution of the shielding current density is governed by the following integro-differential equation:

$$\mu_0 \frac{\partial}{\partial t}(\hat{W}T) + (\nabla \times \boldsymbol{E}) \cdot \boldsymbol{e}_z = -\frac{\partial}{\partial t}\langle \boldsymbol{B} \cdot \boldsymbol{e}_z \rangle, \qquad (1)$$

where $\boldsymbol{E}$, $\boldsymbol{B}$, μ_0 and b are the electric field, the applied magnetic flux density, the permeability of vacuum and the HTS film thickness, respectively. In addition, a scalar function $T(\boldsymbol{x}, t)$ is the z-component of the current vector potential and $\langle\ \rangle$ is an average operator over the HTS film thickness. Moreover, $\hat{W}$ is the operator corresponding to the magnetic flux density that is generated by the shielding current density $\boldsymbol{j}$ and its explicit form is expressed in Ref. [1].

In the HTS film, the shielding current density and the electric field are closely related to each other. As the J-

author's e-mail: *saitoh@yz.yamagata-u.ac.jp*

*) This article is based on the presentation at the 30th International Toki Conference on Plasma and Fusion Research (ITC30).

E constitutive relation, the power law is employed [1–3]. The initial and boundary conditions are assumed as $T = 0$ at $t = 0$ and $\boldsymbol{j} \cdot \boldsymbol{n} = 0$ on $\partial\Omega$, respectively. Here, $\boldsymbol{n}$ denotes an outward normal unit vector on the boundary $\partial\Omega$ of Ω.

2.2 Discretization

In this section, let n be a value at the nth time step $t = n\Delta t$, where Δt is the time-step size. First, by using the backward Euler method, the initial-boundary-value problem of Eq. (1) is reduced to the problem in which the solution $T(\boldsymbol{x}, n\Delta t)$ of the nonlinear boundary-value problem at each time step is obtained. Next, the linear boundary-value problem at each Newton cycle is obtained when the Newton method is applied to its nonlinear boundary-value problem. Finally, the above linear boundary-value problem can be discretized as the following linear system using the finite element method:

$$A\,\delta\boldsymbol{T} = \boldsymbol{b}, \tag{2}$$

where $\delta\boldsymbol{T}$ and $\boldsymbol{b}$ are N-dimensional unknown and given vectors, respectively.

On the other hand, the coefficient matrix A is given by $A = U^T(W+\delta E_{\mathrm{N}})U + F$, where $U \equiv I - F$. Moreover, I and F denote an identity matrix and the matrix determined by the boundary condition, respectively. In addition, $W \in \mathbb{R}^{N\times N}$ denotes the symmetric dense matrix obtained by discretizing with respect to space, whereas $\delta E_{\mathrm{N}} \in \mathbb{R}^{N\times N}$ indicates the symmetric sparse matrix that changes depending on the time step and the Newton cycle.

By solving Eq. (2) at every time step and Newton cycle, we can analyze the time evolution of the shielding current density in the uncracked HTS film. Throughout the present study, the shielding current analysis is applied to the scanning permanent magnet method [3]. The values of physical and geometric parameters are described in Ref. [3].

3. Linear-System Solver

As mentioned above, W becomes the symmetric dense matrix. Therefore, Eq. (2) has so far been solved by using the direct method. Kamitani *et al.* applied the Bunch-Kaufman (B-K) factorization method to Eq. (2) because it has half the operation count as the Gaussian elimination [1].

In the present study, we apply an iterative method to Eq. (2). When the standard CG method, *i.e.*, the CG method without any acceleration technique, is adopted as the iterative method, its operation count becomes $O(K\,N^2)$, where K denotes the iteration number required for the convergence. In the following, the iteration number required for the convergence is called the convergent iteration number. If the inequality, $K < N/6$, is satisfied, the speed of the standard CG method is faster than that of the B-K factorization method.

As is well known, a preconditioning is an acceleration technique of the standard CG method. In particular, the preconditioning is to equivalently convert Eq. (2) into the following linear system:

$$\left(C^{-1}\,A\,C^{-T}\right)\left(C^{T}\,\delta\boldsymbol{T}\right) = C^{-1}\,\boldsymbol{b}, \tag{3}$$

where the preconditioner C is the $N \times N$ regular matrix.

In order to rapidly solve Eq. (3), it is crucial to choose the preconditioner appropriately. The suitable conditions for the preconditioner C are as follows:

1. The regular matrix M which can be transformed into $C\,C^T$ is easily generated.
2. $CC^T \approx A$.
3. C is the sparse matrix.

Several methods have been proposed to determine the preconditioner. The renowned method is to compose the preconditioner C by applying an incomplete Cholesky (IC) factorization to the coefficient matrix. Because A in Eq. (2) is a symmetric dense matrix, the operation count required to generate the preconditioner becomes $O(N^3/3)$ by using the IC factorization. This implies that the speed of the preconditioning almost matches with that of the Gaussian elimination. For quickly solving Eq. (3), an appropriate preconditioner must be selected.

4. $\mathcal{H}$-Matrix-Based Acceleration Technique

This section explains two techniques for accelerating the speed of the linear-system solver. One technique is the $\mathcal{H}$-matrix arithmetic, whereas the other is the $\mathcal{H}$-matrix-based preconditioning.

4.1 $\mathcal{H}$-matrix arithmetic

When Eq. (2) is solved by using the standard CG method, a matrix-vector product must be calculated at each iteration for the linear-system solver. For the case with the matrix W, the operation count of the matrix-vector product $W\boldsymbol{v}$ is $O(N^2)$, where $\boldsymbol{v}$ denotes an arbitrary N-dimensional vector.

In order to resolve the above difficulty, the $\mathcal{H}$-matrix method [4] with the adaptive cross approximation [5, 6] was proposed. In this method, a cluster tree based on node information is first created. Next, the target matrix is converted into some submatrices by using the cluster tree. Finally, the low-rank approximation is executed if and only if the submatrix fulfills the admissibility condition. Throughout the present study, the submatrix to which the low-rank approximation is applied is called the low-rank block. In contrast, the submatrix that remains as the original matrix is called the full-rank block.

By executing the above procedure, a hierarchical matrix $H\,(\equiv H_F + H_L)$ is obtained from the target matrix. Here, H_{F} and H_{L} denote the matrix extracted full-rank and low-rank blocks, respectively. Note that the number of elements in H_F is much lower than that in H_L. Therefore, the matrix-vector product can be executed at a high speed.

When the $\mathcal{H}$-matrix method is applied to the matrix W, Eq. (2) is transformed into the following linear system:

$$A_H \, \delta \boldsymbol{T} = \boldsymbol{b}. \tag{4}$$

Here, $A_H \equiv U^T (H + \delta E_N) U + F$.

4.2 $\mathcal{H}$-matrix based preconditioning

To further accelerate the speed of the linear-system solver, the preconditioning is applied to Eq. (4). By using the preconditioner C, Eq. (4) is equivalently transformed into the following linear system:

$$\left(C^{-1} A_H C^{-T}\right)\left(C^T \delta \boldsymbol{T}\right) = C^{-1} \boldsymbol{b}. \tag{5}$$

In this section, C is determined by using the following three steps:

Step 1 M is given by $M = U^T (H_F + \delta E_N) U + F$.
Step 2 M is approximated as LDL^T using the IC factorization, where L and D denote the lower triangular and diagonal matrices, respectively.
Step 3 $LD^{1/2}$ is adopted as C, where $D^{1/2}$ denotes the square root of the elements of D.

Throughout the present study, the above preconditioning is called the $\mathcal{H}$-matrix-based preconditioning. By solving Eq. (5) instead of Eq. (4), a numerical solution may be obtained at a high speed.

4.3 Performance evaluation

Let us investigate the influence of the $\mathcal{H}$-matrix-based acceleration technique on the solver speed of the linear system. As the experimental parameters, the judgement of convergence ϵ_{CG} and the initial guess $\delta \boldsymbol{T}_0$ are given by $\epsilon_{CG} = 10^{-9}$ and $\delta \boldsymbol{T}_0 = \boldsymbol{0}$, respectively. In addition, the value of N is fixed as $N = 25056$. In this study, the computer environment is shown as follows: (OS: Linux version 3.10.0-1160.el7.x86_64, CPU: Xeon Gold 6252 Processor × 2, memory: 512GB, compiler: gfortran -O3).

First, we examine the influence of the $\mathcal{H}$-matrix arithmetic on the performance of the matrix-vector product. As measures of the accuracy and the speed, the relative error $e_H = \|W\boldsymbol{v} - H\boldsymbol{v}\|/\|W\boldsymbol{v}\|$ and the speedup ratio $S_H = \tau_W/\tau_H$, respectively. Here, τ_W and τ_H are the CPU times of matrix-vector products for W and H, respectively. In addition, the ratio of the number of elements in H_F to that in H is called the full-rank block ratio.

The relative error e_H and the speedup ratio S_H are calculated as a function of the full-rank block ratio α and are plotted in Fig. 1. This figure shows that both S_H and e_H monotonously diminish as the value of α increases. In other words, the speed of the matrix-vector product is enhanced with a decrease in the full rank block ratio, whereas the accuracy is degraded drastically. From a speed and accuracy perspective, the admissibility parameter in the $\mathcal{H}$-matrix method is determined so that $e_H \lesssim 10^{-6}$ is satisfied.

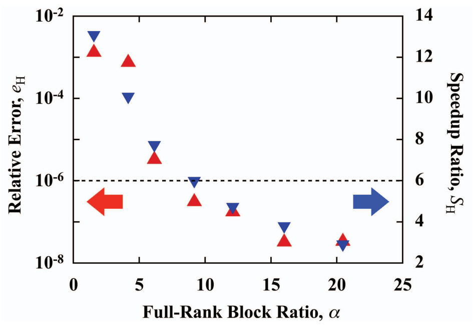

Fig. 1 Dependence of the relative error e_H and the speedup ratio S_H on the full-rank block ratio α. Here, the red and the blue symbols denote the values of e_H and S_H, respectively.

Next, we estimate the speed performance of the CG method with the $\mathcal{H}$-matrix-based acceleration technique. Throughout the present study, the CG method with the $\mathcal{H}$-matrix-based acceleration technique is called the ICCG with the $\mathcal{H}$-matrix method (ICCGH). In order to clarify the effect of the preconditioning, the proposed methods are compared with the CG method without any acceleration technique. As the measure of the speed of the linear-system solver, two speedup ratios, S_C and S_P, are defined by $S_C = \tau_S/\tau_C$ and $S_P = \tau_A/\tau_C$, respectively. Here, τ_C, τ_A and τ_S represent the CPU time of CG methods with two acceleration techniques, solely with $\mathcal{H}$-matrix arithmetic and without any acceleration technique, respectively. Note that τ_C, τ_A and τ_S include the CPU times for generating the $\mathcal{H}$-matrix, the creation of the preconditioned matrix and the execution of the solver. The result of the numerical experiment shows that the values of S_C and S_P are 3.83 and 1.27, respectively.

From the above results, the ICCGH is faster than the standard CG method. However, the acceleration effect of the $\mathcal{H}$-matrix-based preconditioning is lower than that of the $\mathcal{H}$-matrix arithmetic.

5. Speed Improvement of $\mathcal{H}$-Matrix Based Preconditioning

5.1 k time-step-skip approach

As is apparent from numerical results, the speedup performance of $\mathcal{H}$-matrix based preprocessing is lower than that of $\mathcal{H}$-matrix arithmetic. In this section, we aim to improve the $\mathcal{H}$-matrix based preconditioning.

Since δE_N must be determined at each time step and Newton cycle, the preconditioner is generated for each change in δE_N. To shorten the generation time, we propose a new preconditioner that adopts the approximation matrix δE_N^* instead of δE_N. Specifically, Step 1 in the $\mathcal{H}$-matrix based preconditioning is replaced with the following two procedures.

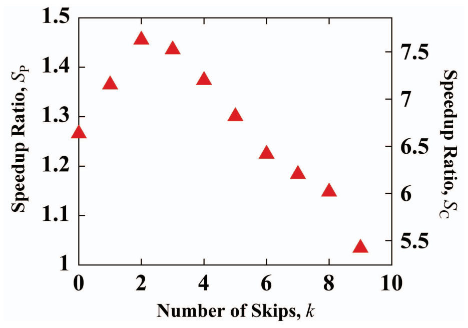

Fig. 2 Dependence of the speedup ratios S_P and S_C on the number k of skips.

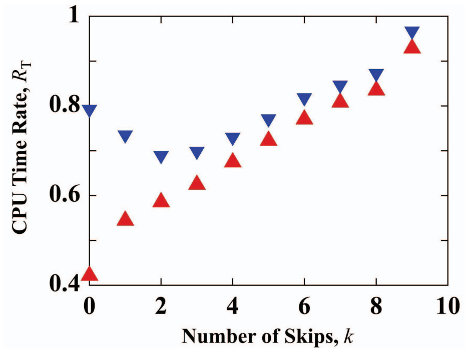

Fig. 3 Dependence of the CPU time rate R_T on the number k of skips. Here, the symbols, ▼ and ▲, denote the CPU time rate for the preconditioning and that for executing the solver, respectively.

1. The approximation matrix δE^*_N is determined from the shielding current density $\boldsymbol{j}$ obtained at every k time step.
2. M is determined by $M = U^T \left(H_F + \delta E^*_N\right) U + F$.

Throughout the present study, the ICCGH incorporated with the above two procedures is called the ICCGH with the skip approach.

Let us investigate the speed performance of the proposed preconditioning. S_P and S_C are calculated as a function of the number k of skips and are depicted in Fig. 2. S_P decreases monotonously with k after a slight increase until satisfying $k > 2$. In addition, the ICCGH with the skip approach becomes up to about 1.5 times faster than that solely with the $\mathcal{H}$-matrix arithmetic.

To investigate this tendency, the CPU time rate R_T is calculated as a function of k and is plotted in Fig. 3. Here, R_T denotes the ratio of the CPU times required for the preconditioning or the execution of the solver to τ_A. We see from this figure that the CPU time required for the preconditioning diminishes with k until the value of k satisfies $k \geq 3$. On the other hand, the CPU time required for executing the solver monotonously rises with an increase of k. These results show that the solver speed is improved if the CPU time required for the preconditioning decreases.

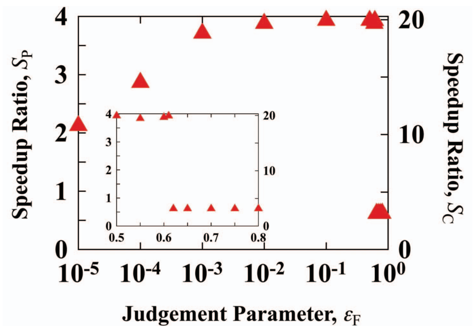

Fig. 4 Dependence of speedup ratios S_P and S_C on the judgement parameter ϵ_F. Here, the inset shows the enlarged figure of Fig. 4 from $\epsilon_F = 0.5$ to $\epsilon_F = 0.8$.

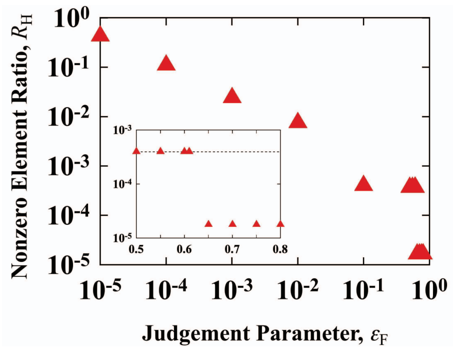

Fig. 5 Dependence of the nonzero element Ratio R_H on the judgement parameter ϵ_F. Here, the inset shows the enlarged figure of Fig. 5 from $\epsilon_F = 0.5$ to $\epsilon_F = 0.8$.

These results show that the acceleration effect of the proposed preconditioning exceeds that of the conventional preconditioning. In the following, the value of k is fixed as $k = 2$.

5.2 Compression of number of nonzero elements in H_F

Although the number of nonzero elements in H_F is much smaller than that in W, it is much larger than that in δE_N. Therefore, we enhance the generation approach of the preconditioner by reducing the number of nonzero elements in H_F.

In the improved preconditioning, Step 1 in the $\mathcal{H}$-matrix-based preconditioning is further replaced with the following four procedures:

1. H_F is substituted into the approximation matrix H^*_F.
2. $\left(H^*_F\right)_{ij} = 0$ if and only if $\left|\left(H^*_F\right)_{ij}\right| / |H_{\max}| < \epsilon_F$ is satisfied.
3. The approximation matrix δE^*_N is determined from $\boldsymbol{j}$ at every k time step.
4. M is determined by $M = U^T \left(H^*_F + \delta E^*_N\right) U + F$.

Here, ϵ_F denotes the judgement parameter and $H_{\max}$ indicates the maximum value of H_F. In the following, the IC-

CGH incorporated with the above procedure is called the ICCGH with the modified skip approach.

Let us evaluate the speed performance of the modified preconditioning. S_P and S_C are calculated as a function of the judgement parameter ϵ_F and are plotted in Fig. 4. S_P rises monotonically with an increase in ϵ_F and its value becomes constant for the case with $0.01 \lesssim \epsilon_\mathrm{F} \lesssim 0.61$. However, the value of S_P is less than 1 for $\epsilon_\mathrm{F} \gtrsim 0.61$. In addition, the ICCGH with the modified skip approach becomes up to about 20 times faster than the standard CG method.

To explain the reason for the above result, the dependence of the nonzero element ratio R_H on ϵ_F is shown in Fig. 5. After the nonzero element ratio decreases monotonously until $\epsilon_\mathrm{F} \approx 10^{-1}$, it becomes almost constant. For $\epsilon_\mathrm{F} \gtrsim 0.61$, the value of R_H is less than 10^{-4}. In other words, the nonzero element of H_F^* almost vanishes. This means that H_F^* has almost no effect on the preconditioned matrix for the case with $\epsilon_\mathrm{F} \gtrsim 0.61$. Therefore, S_P indicates the behavior as shown in Fig. 5.

In this way, the modified $\mathcal{H}$-matrix-based preconditioning is effective to enhance the speed of the ICCGH.

6. Conclusion

In this study, the ICCGH method has been applied to the linear system in the shielding current analysis of the uncracked HTS film and its performance is investigated numerically. Conclusions are summarized as follows.

1. By improving the $\mathcal{H}$-matrix-based preconditioning, the speed performance of the $\mathcal{H}$-matrix-based acceleration technique is enhanced significantly.
2. The $\mathcal{H}$-matrix based acceleration technique can suitably speed up the linear-system solver.

From these results, the $\mathcal{H}$-matrix-based acceleration technique might be a useful tool for executing the shielding current analysis at high speed.

In this study, the hierarchical matrix H is generated from matrix W. Therefore, memory consumption is extremely high. In future works, the generation method of H will be improved for reducing memory consumption.

Acknowledgment

The author is deeply grateful to the editor and anonymous reviewers for the constructive comments. This work was partly supported by the NIFS Collaboration Research program (NIFS20KKGS027).

[1] A. Kamitani, T. Takayama and H. Nakamura, Plasma Fusion Res. **5**, S2112 (2010).
[2] T. Takayama, A. Kamitani and A. Tanaka, Physica C **470**, 1354 (2010).
[3] A. Kamitani, T. Takayama, A. Saitoh and H. Nakamura, Plasma Fusion Res. **16**, 2405005 (2021).
[4] M. Bebendorf, *Hierarchical Matrices* (Springer-Verlag, Berlin, 2008), p.49. **22** (1983).
[5] S. Kurz, O. Rain and S. Rjasanow, IEEE Trans. Magn. **38**, 421 (2002).
[6] V. Le-Van, B. Bannwarth, A. Carpentier, O. Chadebec, J.M. Guichon and G. Meunier, IEEE Trans. Magn. **50**, 7010904 (2014).
[7] Y. Takahashi, C. Matsumoto and S. Wakao, IEEE Trans. Magn. **43**, 1277 (2007).
[8] A. Ida, T. Iwashita, T. Mifune and Y. Takahashi, IEEE Trans. Magn. **52**, 7205104 (2016).