Performance Evaluation of Parallel Gram-Schmidt Re-Orthogonalization Methods

Takahiro Katagiri[1]

PRESTO, Japan Science and Technology Corporation(JST)
JST Katagiri Laboratory,
Computer Centre Division, Information Technology Center,
The University of Tokyo
2-11-16 Yayoi, Bunkyo-ku, Tokyo 113-8658, JAPAN
Phone: +81-3-5841-2758, FAX: +81-3-3814-7231
katagiri@pi.cc.u-tokyo.ac.jp

Abstract. In this paper, the performance of the five kinds of parallel re-orthogonalization methods by using the Gram-Schmidt (G-S) method is reported. Parallelizm of the re-orthogonalization process depends on the implementation of G-S orthogonalization process, i.e. Classical G-S (CG-S) and Modified G-S (MG-S). To relax the parallelizm problem, we propose a new hybrid method by using both the CG-S and MG-S. The HITACHI SR8000 of 128 PEs, which is a distributed memory supercomputer, is used in this performance evaluation.

1 Introduction

The orthogonalization process is one of the most important processes to perform several linear algebra computations, such as eigendecomposition and QR decomposition [3, 10, 11]. Many researchers have paid more attention to QR decomposition [2, 5, 6, 12, 13]. Notwithstanding, we focus on the re-orthogonalization process in this paper. This is because a lot of iterative methods for solving linear equations and eigenvector computations need the re-orthogonalization process to maintain accuracy of results. For instance, the GMRES method, which is one of the latest iterative algorithms, requires the re-orthogonalization process to obtain base-vectors according to the number of its re-start frequency [4]. The approach of parallel re-orthogonalization is the main difference with respect to conventional parallel approaches based on QR decomposition [6, 12, 13].

Recently, there are a number of parallel machines, because parallel processing technologies have been established. For this reason, many researchers try to parallelize the orthogonalization process. It is known that, however, the parallelism of the process depends on the number of orthogonalized vectors we needed [7, 8]. The orthogonalized process, therefore, is classified as the following two kinds of processes in this paper:

- QR decomposition
  The orthogonalization process to obtain the normalized and orthogonalized vectors q_1, q_2, $\cdots$, q_n by using the normalized vectors a_1, a_2, $\cdots$, a_n.

– Re-orthogonalization
  The orthogonalization process to obtain the normalized and orthogonalized vector q_i by using the normalized and orthogonalized vectors $q_1, q_2, \cdots, q_{i-1}$.

This paper discusses how to parallelize the re-orthogonalization process by using the Gram-Schmidt (G-S) orthogonalization method. The focus is especially the parallelization of Classical G-S (CG-S) method, since the CG-S method has high parallelism. The CG-S method, however, has a trade-off problem between accuracy and execution speed. For example, T.Katagiri reported that more than 90% execution time is wasted by using MG-S method in a parallel eigensolver [8]. For this reason, the main aim of this paper is a proposition of a new parallel G-S algorithm to relax the trade-off problem.

This paper is organized as follows. Chapter 2 is the explanation of sequential G-S algorithms in the viewpoint of data dependency. Chapter 3 describes parallel G-S algorithms based on the sequential G-S algorithms. Chapter 4 proposes the new algorithm. Chapter 5 is the evaluation of the new algorithm by using the HITACHI's parallel super-computer. Finally, Chapter 6 summarizes the findings of this paper.

2 Sequential algorithms of G-S re-orthogonalization method

It is widely known that there are the following two methods to perform re-orthogonalization by using the G-S method. They are known as Classical G-S (CG-S) method and Modified G-S (MG-S) method. The CG-S method is a simply implemented method in the formula of the G-S orthogonalization, and the MG-S method is a modified one in order to obtain high accuracy. In this chapter, we will explain the difference between them from the viewpoint of data dependency.

2.1 Classical Gram-Schmidt (CG-S) Method

The CG-S method to perform the re-orthogonalization is shown in Figure 1. The notation of $(\cdot, \cdot)$ in Figure 1 means an inner product.

```
⟨1⟩  q_i = a_i
⟨2⟩  do j = 1, i − 1
⟨3⟩    q_i = q_i − (q_j^T, a_i) q_j
⟨4⟩  enddo
⟨5⟩  Normalization of q_i.
```

Fig. 1. The CG-S method in re-orthogonalization to the vector q_i.

Figure 1 shows that the most inner kernel ⟨3⟩ has a parallelism. The nature of this can be explained as the parallelizm of the inner product (q_j^T, a_i), since the

inner product can be performed parallelly for j-loop when we obtain the initial vector of a_i.

2.2 Modified Gram-Schmidt (MG-S) Method

The MG-S method in re-orthogonalization is shown in Figure 2.

$\langle 1 \rangle$	$a_i^{(0)} = a_i$
$\langle 2 \rangle$	**do** $j = 1,\ i-1$
$\langle 3 \rangle$	$a_i^{(j)} = a_i^{(j-1)} - (q_j^T, a_i^{(j-1)}) q_j$
$\langle 4 \rangle$	**enddo**
$\langle 5 \rangle$	Normalization of $q_i = a_i^{(i-1)}$.

Fig. 2. The MG-S method in re-orthogonalization to the vector q_i.

Figure 2 shows that there is no parallelizm to the inner product of $\langle 3 \rangle$, since the inner product depends on the defined vector $a_i^{(j-1)}$ for j-loop. This is the basic difference with the CG-S method. For this reason, many researchers believe that the re-orthogonalization by the MG-S method is an unsuitable method for parallel processing.

3 Parallelization of G-S orthogonalization method

With parallel processing, the re-orthogonalization by the G-S method behaves differently according to the data distribution method for the orthogonalized vectors $q_1, q_2, ..., q_{i-1}$ [8]. We will explain this in this chapter.

3.1 Column-wise distribution

First of all, we explain a simple distribution method, named column-wise distribution (CWD). The CWD is a distribution that the whole elements of the normalized vector a_i and the normalized and orthogonalized vectors $q_1, q_2, ..., q_{i-1}$ are distributed to each PE (Processing Element).

Next is a discussion on how to implement parallel algorithms based on the CWD.

CG-S method Figure 3 shows a re-orthogonalization algorithm in CWD. The notation of "Local" in Figure 3 shows that the following formula should be performed by using local data (distributed data) only.

According to Figure 3, there is a parallelizm for computing the kernel of $\langle 9 \rangle$.

```
⟨1⟩  q_i = a_i
⟨2⟩  if (I have a_i) then
⟨3⟩    Broadcast (a_i)
⟨4⟩  else
⟨5⟩    receive (a_i)
⟨6⟩    q_i = 0
⟨7⟩  endif
⟨8⟩  do j = 1, i − 1
⟨9⟩    Local q_i = q_i − (q_j^T, a_i) q_j enddo
⟨10⟩ if (I have a_i) then
⟨11⟩   do j = 1, i − 1
⟨12⟩     receive (q_j) from PE that holds q_j
⟨13⟩     Local q_i = q_i + q_j enddo
⟨14⟩ else
⟨15⟩   send (q_i)
⟨16⟩ endif
⟨17⟩ Normalization of q_i.
```

Fig. 3. The CG-S method in column-wise distribution.

```
⟨1⟩  a_i^(0) = a_i
⟨2⟩  do j = 1, i − 1
⟨3⟩    receive (a_i^(j−1)) from PE that holds a_i^(j−1)
⟨4⟩    Local a_i^(j) = a_i^(j−1) − (q_j^T, a_i^(j−1)) q_j
⟨5⟩    send (a_i^(j)) to PE that holds a_i^(j+1)
⟨6⟩  enddo
⟨7⟩  Normalization of q_i = a_i^(i−1).
```

Fig. 4. The MG-S method in column-wise distribution.

MG-S method Figure 4 shows a re-orthogonalization algorithm in CWD.

Please note that there is no parallelizm for computing the kernel of ⟨4⟩ according to Figure 3, because PE holds $a_i^{(j-1)}$ and PE holds $a_i^{(j)}$ are located in different places in CWD.

For this reason, the re-orthogonalization by MG-S method in CWD has basically poor parallelizm.

3.2 Row-wise distribution

Next we will explain another well-known distribution, namely row-wise distribution (RWD). The RWD is a distribution in which the elements of vectors (a_i, q_i) and orthogonalized vectors $q_1, q_2, ..., q_{i-1}$ are distributed to different PEs. Please note that the difference between CWD and RWD is that the CWD does not distribute the elements of vectors.

In RWD, to calculate the inner products of $(q_j^T, a_i^{(j-1)})$ and (q_j^T, a_i), we need a scalar reduction operation in both of CG-S and MG-S methods for the RWD, since each PE does not have whole elements to calculate the inner products.

CG-S method Figure 5 shows the parallel re-orthogonalization algorithm of the CG-S method. The notation of "Global sum" in Figure 5 is the collective communication operation, which sums up distributed data and then distributes the result to all PEs. The collective communication can be implemented by using the `MPI_ALLREDUCE` function on the MPI (Message Passing Interface).

```
⟨1⟩  q_i = a_i
⟨2⟩  do j = 1, i − 1
⟨3⟩    Local (q_j^T, a_i)   enddo
⟨4⟩  Global sum of η_j = (q_j^T, a_i).
                          (j = 1, ··· , i − 1)
⟨5⟩  do j = 1, i − 1
⟨6⟩    Local q_i = q_i − η_j q_j
⟨7⟩  enddo
⟨8⟩  Normalization of q_i.
```

Fig. 5. The CG-S method in row-wise distribution.

For Figure 5, we find that the Global sum operation is not needed in every step because the CG-S method does not require the inner producted value calculated in the most inner loop. This fact also can be found in the explanation of Chapter 2, since there is no dependency in the direction of the j-loop. The inner producted value is easily calculated by using the value before normalization. The vector length of the Global sum operation depends on the number of orthogonalized vectors $i-1$.

MG-S method Figure 6 shows the parallel re-orthogonalization algorithm for the MG-S method.

For Figure 6, we find that a scalar reduction for distributed data around PEs is essential in every steps, since the Global sum operation is located in the innermost loop. In $j+1$-th step, the MG-S method needs the inner producted data calculated j-th step, thus we have to implement the Global sum operation in the most inner loop. The explanation in Chapter 2 also shows the fact, since there is a flow dependency in the direction of the j-loop.

Comparison of the number of communications Now we compare the number of execution times for the Global sum operation in the MG-S and CG-S methods. Let the number of re-orthogonalization i be fixed as k. It is clear that

```
⟨1⟩  a_i^(0) = a_i
⟨2⟩  do j = 1, i − 1
⟨3⟩    Local (q_j^T, a_i^(j−1))
⟨4⟩    Global sum of η = (q_j^T, a_i^(j−1)).
⟨5⟩    Local a_i^(j) = a_i^(j−1) − η q_j
⟨6⟩  enddo
⟨7⟩  Normalization of q_i = a_i^(i−1).
```

Fig. 6. The MG-S method in row-wise distribution.

the MG-S method requires $k-1$ times of the Global sum operation while the CG-S method requires 1 times. Thus, from the view point of execution times, the CG-S method is superior to the MG-S method. The CG-S method, however, has lower accuracy than the accuracy of the MG-S method. Therefore, we can not determine the best parallel method in the RWD.

From the viewpoint of these variations, the RWD will be a good choice. The discussion on the RWD, however, is omitted in this paper. The reason is that the distributing the elements of vectors makes the implementation difficult in many iterative algorithms and poor utility for many users.

4 Proposal of the Hybrid Gram-Schmidt (HG-S) method

In this chapter, we propose a new parallel re-orthogonalization method, named Hybrid Gram-Schmidt (HG-S) method.

4.1 Basic idea

We have explained that there is a trade-off problem between accuracy and parallel execution speed in G-S method. The following is a summary of the problem for CWD.

- **CG-S Method** Accuracy: Low, Parallelism: Yes
- **MG-S Method** Accuracy: High, Parallelism: None

To relax the problem, using both CG-S and MG-S methods for re-orthogonalization is one of the reasonable ways. We call this "Hybrid"-ed G-S method as HG-S method in this paper.

4.2 The HG-S re-orthogonalization method

Figure 7 shows a sketch of the algorithm for the HG-S method. Please note that the code of Figure 7 shows the nature of re-orthogonalization process, since many iterative algorithms, for example the inverse iteration method for eigenvector computation, needs the re-orthogonalization like Figure 7.

Please note that we select MG-S method after CG-S method to obtain high accuracy, since the HG-S method in Figure 7 can use MG-S orthogonalized vectors in the process of CG-S method.

```
do i = 1,...,max_num_vectors
....
   do iter = 1,...,max_iter
      ....
      Re-orthogonalization (q_i) by using the CG-S Method.
      ....
      if (q_i is converged) break the iter-loop.
   enddo
   Re-orthogonalization (q_i) by using the MG-S Method.
....
enddo
```

Fig. 7. The HG-S method in re-orthogonalization. The notations of *max_num_vectors* and *max_iter* mean the number of vectors to orthogonalize, and the maximal number of iterations in the iterative method, respectively.

5 Experimental Results

The HG-S method using a distributed memory parallel machine is evaluated in this chapter. To evaluate the HG-S method, we implemented the HG-S method to Inverse Iteration Method (IIM) for computation of eigenvectors.

5.1 Experimental environment

The HITACHI SR8000/MPP is used in this experiment. The HITACHI SR8000 system is a distributed memory, message-passing parallel machine of the MIMD class. It is composed of 144 nodes, each having 8 Instruction Processors (IPs), 14.4GFLOPS of theoretical peak performance and 16 Gigabytes of main memory, interconnected via a communication network with the topology of a three-dimensional hyper-crossbar. The peak interprocessor communications bandwidths are 1.6 Gbytes/s in one-way, and 3.2 Gbytes/s in both-way.

The SR8000 system has two types of parallel environments. One is intra-node parallel processing, and the other is inter-node parallel processing. The intra-node parallel processing is so-called parallel processing as a shared memory parallel machine. On the other hand, the inter-node parallel processing is similar to parallel processing as a distributed memory parallel machine, and it should perform interprocessor communications. The HITACHI Optimized Fortran90 V01-04 compiler is used. In this experiment, *-opt=ss -parallel=0* is used as a compile option in the inter-node parallel processing.

As communication library, optimized MPI (Message Passing Interface) by HITACHI is used in this experiment.

5.2 Details of implemented re-orthogonalization methods

We implemented parallel re-orthogonalization methods in IIM for eigenvector computation [10]. The details of the IIM algorithm are summarized as follows.

- Object Matrix : Tridiagonal real symmetric matrix T
- Method : The Rayleigh quotient IIM
- Max Iterative Numbers : 40
- Requirement residual : $||Tx_i - \lambda_i x_i||_2 \leq ||T||_1 \times \epsilon$, $(i = 1, 2, ..., n)$, where ϵ is a machine epsilon, x_i is the i-th eigenvector, and λ_i is the i-th eigenvalue in an eigensystem.

We have developed an efficient parallel eigensolver by using a reduced communication method [9]. The IIM method is also available for the eigensolver, therefore, we measure the execution time for the routine of IIM.

The test matrices are chosen as the following.

- Matrix(1) : Tridiagonal matrix reduced from the Frank matrix
  - Dimension : 10,000
  - The length to decide the clustered eigenvalues : 58471.532
  - The number of groups for the clustered eigenvalues : 8
  - The maximum number of clustered eigenvalues : 9,993

- Matrix(2) : Glued Wilkinson W_{21}^+ Matrix, δ =1.0
  - Dimension : 10,000
  - The length to decide the clustered eigenvalues : 64.998
  - The number of groups for the clustered eigenvalues : 1
  - The maximum number of clustered eigenvalues : 10,000

- Matrix(3) : Glued Wilkinson W_{21}^+ Matrix, $\delta = 10^{-14}$
  - Dimension : 10,000
  - The length to decide the clustered eigenvalues (for a block diagonal matrix) : 0.129
  - The number of groups for the clustered eigenvalues (for a block diagonal matrix) : 13
  - The maximum number of clustered eigenvalues (for a block diagonal matrix) : 2
  - The number of block diagonal matrices : 477

Please note that the re-orthogonalization is performed to maintain accuracy of eigenvectors. The number of re-orthogonalized eigenvectors is the same as the number of clustered eigenvalues.

Next, we summarize the implemented parallel re-orthogonalization methods as follows.

- CG-S (1) : Applying MG-S as local orthogonalizations
- CG-S (2) : Applying CG-S as local orthogonalizations
- MG-S
- HG-S
- IRCG-S : Iterative Refinement CG-S Method [1, 10]
- NoOrt : Not Re-orthogonalized

There are two kinds of methods in the parallel re-orthogonalization of CG-S method. The CG-S(1) is a method in which CG-S is performed inter PEs, and MG-S is performed to intra PE data. The CG-S(2) is a method in which CG-S is performed inter PEs, and CG-S is also performed to intra PE data.

The IRCG-S method shown above is called Iterative Refinement CG-S Method, and this method performs CG-S method multiple times. In this experiment, the number of CG-S method is fixed as 2 times.

5.3 The results of experiment

Table 1 shows the execution time for the five parallel re-orthogonalization methods in IIM. For the restriction of super-computer environment, over $\lfloor 64/\lceil p/8 \rceil \rfloor$ hours jobs are automatically killed, where p is the number of PEs.

Table 2 shows the orthogonalization accuracy of the calculated eigenvectors by IIM, which is measured by the Frobenius norm. The Frobenius norm of $n \times n$ matrix $A = (a_{ij})$, $i, j = 1, ..., n$ is defined as:

$$||A||_F \equiv \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}^2}. \tag{1}$$

5.4 Discussion

Test matrix (1) Table 1 (a) shows there was poor parallelizm for MG-S. The reason is explained as no parallelizm in MG-S process in CWD. On the other hand, by using CG-S Method, we obtained speed-up factors increasing the number of PEs.

For HG-S method, the HG-S was faster than CG-S. This is a surprising result in a sense, because HG-S performs multi-times CG-S and one-time MG-S. We think the main reason is that by using the HG-S method, total number of iterations in IIM is reduced by using better accuracy eigenvalues with comparison to eigenvectors orthogonalized by CG-S only.

In the case of the Frank matrix, we could not find the difference of accuracy for eigenvectors except for not-orthogonalized cases. For this reason, HG-S is the best method in this case.

Table 1. The execution time of each method in the IIM. The unit is in seconds.

(a) Test Matrix(1): Reduced Frank Matrix

#PEs(IPs)	CG-S(1)	CG-S(2)	MG-S	HG-S	IRCG-S	NoOrt
8	6604	6527	38854	6604	12883	23.857
16	3646	3526	24398	3574	6987	12.065
32	2061	2059	28050	2082	3906	7.044
64	1633	1559	27960	1614	3059	3.025
128	2091	2204	>14400	2027	3978	1.808

(b) Test Matrix(2): Glued Wilkinson Matrix δ =1.0

#PEs(IPs)	CG-S(1)	CG-S(2)	MG-S	HG-S	IRCG-S	NoOrt
8	7779	7898	77851	10825	14673	30.88
16	4358	4264	61242	4348	7595	15.31
32	2533	2347	32131	2480	4429	8.99
64	2152	2110	>28800	2189	3241	4.44
128	2696	2390	>14400	2450	4699	2.45

(c) Test Matrix(3): Glued Wilkinson Matrix $\delta = 10^{-14}$

#PEs(IPs)	CG-S(1)	CG-S(2)	MG-S	HG-S	IRCG-S	NoOrt
8	0.558	0.561	0.510	0.593	2.494	0.59
16	0.554	0.696	1.922	1.697	0.619	0.33
32	0.364	0.828	0.970	0.293	0.858	0.20
64	0.530	0.451	0.521	0.431	0.378	0.38
128	0.277	0.221	0.315	0.269	0.259	0.07

Test matrix (2) First of all, we have to mention that the accuracy of eigenvectors was very poor when we use over 32 PEs except for MG-S method. For this reason, we should select the MG-S method in this case. As a result, if we use 16 PEs and 8 PEs, the methods of CG-S(1), CG-S(2) and HG-S will be considerable methods in this case.

For the execution time, we also had a surprising result for MG-S, because although the MG-S method has no parallelizm in this case, there are speed-up factors of MG-S. We think that the result is caused by reducing data amount by parallel processing, however, detailed analysis for this result is a future work.

Test matrix (3) In this case, the execution time of each method is quite small compared to the cases of the matrix (1) and (2). This reason is explained as the implementation of parallel IIM routine. In our IIM routine, if an element of sub-diagonals of the tridiagonal matrix T is of quite small value such as machine ϵ, the diagonal matrices separated by the element are treated as independent matrices of the eigensystem. This separation can dramatically reduce computational complexity and increase parallelizm in this matrix. In the case of matrix

Table 2. The eigenvector accuracy of each method in the IIM. The norm of Frobenius is used.

(a) Test Matrix(1): Reduced Frank matrix
(MG-S : Residual of $max_i||Ax_i - \lambda_i x_i||_2$ = 0.1616E-006)

#PEs(IPs)	CG-S(1)	CG-S(2)	MG-S	HG-S	IRCG-S	NoOrt
8	0.6485E-012	0.6505E-012	0.6670E-012	0.6485E-012	0.6455E-012	1.414
16	0.6621E-012	0.6627E-012	0.6666E-012	0.6620E-012	0.6613E-012	1.414
32	0.6892E-012	0.6895E-012	0.6664E-012	0.6893E-012	0.6899E-012	1.414
64	0.9422E-012	0.9413E-012	0.6665E-012	0.9412E-012	0.9419E-012	1.414
128	0.1546E-011	0.1547E-011	Time Out	0.1540E-011	0.1549E-011	1.414

(b) Test Matrix(2): Glued Wilkinson Matrix δ =1.0
(MG-S : Residual of $max_i||Ax_i - \lambda_i x_i||_2$ = 0.4476E-007)

#PEs(IPs)	CG-S(1)	CG-S(2)	MG-S	HG-S	IRCG-S	NoOrt
8	0.1261E-011	0.2679E-011	0.1432E-011	0.3087E-008	0.2137E-012	283.7
16	0.1255E-011	0.6093E-011	0.1971E-011	0.1349E-011	0.4658E-012	282.3
32	2.0191	1.4260	0.5255E-012	1.967	1.9455	283.4
64	3.7387	3.5735	Time Out	3.492	3.7619	284.9
128	5.2028	4.7178	Time Out	4.9206	5.0163	284.9

(c) Test Matrix(3): Glued Wilkinson Matrix $\delta = 10^{-14}$
(MG-S : Residual of $max_i||Ax_i - \lambda_i x_i||_2$ = 0.1625E-010)

#PEs(IPs)	CG-S(1)	CG-S(2)	MG-S	HG-S	IRCG-S	NoOrt
8	0.3966E-007	31.50	0.3966E-007	0.3966E-007	0.3966E-007	31.50
16	0.3966E-007	31.50	0.3966E-007	0.3966E-007	0.3966E-007	31.50
32	0.3967E-007	31.38	0.3966E-007	0.3967E-007	0.3967E-007	31.50
64	0.3966E-007	31.31	0.3966E-007	0.3966E-007	0.3966E-007	31.50
128	0.3967E-007	31.06	0.3966E-007	0.3967E-007	0.3967E-007	31.50

(3), the test matrix is separated by several small diagonal matrices, each has a dimension of 21. For this reason, the number of re-orthogonalizations is reduced, naturally and the execution time is also very shortened.

As for the accuracy of the results, we could find that the accuracy of CG-S(2) method was poor in this matrix with comparison to CG-S(1) method. For this result, therefore, we can say that data which holds intra PE should be re-orthogonalized by using MG-S method, even CG-S method is used for inter PEs.

Acculacy of all matrices The accuracy in this experiment varies according to the number of PEs. This is also a surprising result, because the computations of each G-S method are same. We think that the result is caused by changing the order for the computation of G-S method, however, detail analysis is a part of future work.

6 Conclusion

In this paper, we have proposed a hybrid Gram-Schmidt re-orthogonalized method, and evaluated the five kinds of parallel re-orthogonalization methods including the new hybrid method by using IIM for eigenvector computation.

For the experimental result, the new hybrid method also has an accuracy problem for computed eigenvectors, but it has a benefit for parallel execution time compared to CG-S method. The main reason of this is using higher accuracy eigenvectors than that of CG-S orthogonalized, however, detailed analysis is needed.

The analysis and additional experiments for the test matrices are important future work.

Acknowledgments

I would like to express my sincere thanks to staff at Computer Centre Division, Information Technology Center, the University of Tokyo, for supporting my super-computer environments. I would also like to express my sincere thanks to all members at Kanada Laboratory, Information Technology Center, for giving me useful discussions for the study.

This study is supported by PRESTO, Japan Science and Technology Corporation (JST).

References

1. S. Balay, W. Gropp, L. C. McInnes, and B. Smith. Petsc 2.0 users manual, 1995. ANL-95/11 - Revision 2.0.24, http://www-fp.mcs.anl.gov/petsc/.
2. C. Bischof and C. van Loan. The WY representation for products of Householder matrices. *SIAM J. Sci. Stat. Comput.*, 8(1):s2–s13, 1987.
3. J. W. Demmel. *Applied Numerical Linear Algebra*. SIAM, 1997.
4. J. J. Dongarra, I. S. Duff, D. C. Sorensen, and H. A. van der Vorst. *Numerical Linear Algebra for High-Performance Computers*. SIAM, 1998.
5. J. J. Dongarra and R. A. van de Geijn. Reduction to condensed form for the eigenvalue problem on distributed memory architectures. *Parallel Computing*, 18:973–982, 1992.
6. B. A. Hendrickson and D. E. Womble. The tours-wrap mapping for dense matrix calculation on massively parallel computers. *SIAM Sci. Comput.*, 15(5):1201–1226, 1994.
7. T. Katagiri. A study on parallel implementation of large scale eigenproblem solver for distributed memory architecture parallel machines. *Master's Degree Thesis, the Department of Information Science, the University of Tokyo*, 1998.
8. T. Katagiri. A study on large scale eigensolvers for distributed memory parallel machines. *Ph.D Thesis, the Department of Information Science, the University of Tokyo*, 2000.
9. T. Katagiri and Y. Kanada. An efficient implementation of parallel eigenvalue computation for massively parallel processing. *Parallel Computing*, 27:1831–1845, 2001.

10. B. N. Parlett. *The Symmetric Eigenvalue Problem*. SIAM, 1997.
11. G. W. Stewart. *Matrix Algorithms Volume II:Eigensystems*. SIAM, 2001.
12. D. Vanderstraeten. A parallel block Gram-Schmidt algorithm with controlled loss of orthogonality. *Proceedings of the Ninth SIAM Conference on Parallel Processing for Scientific Computing*, 1999.
13. Y. Yamamoto, M. Igai, and K. Naono. A new algorithm for accurate computation of eigenvectors on shared-memory parallel processors. *Proceedings of Joint Symposium on Parallel Processing (JSPP)'2000*, pages 19–26, 2000. in Japanese.