OpenATLib and Xabclib

User's Manual
for Version 1.03

Information Technology Center, The University of Tokyo and
Central Research Laboratory, Hitachi Ltd.

June 26, 2015

DISCLAIMER

Contents

1. Overview

In this manual, functions for numerical library developers in OpenATLib and Xabclib are explained. Fig. 1-1 and Fig. 1-2 show the components of function on Xabclib and Xabclib.

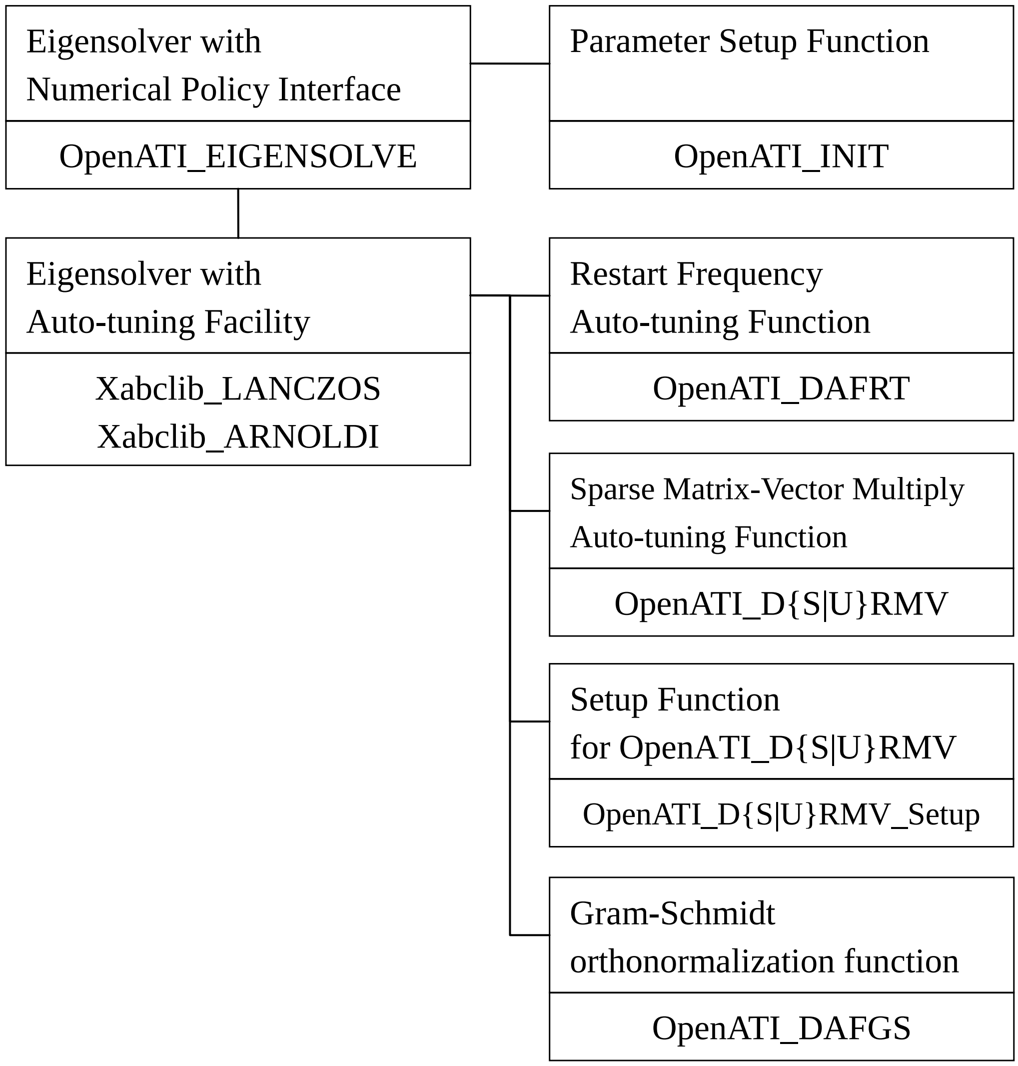

Fig. 1-1 Components of Function on Eigensolver.

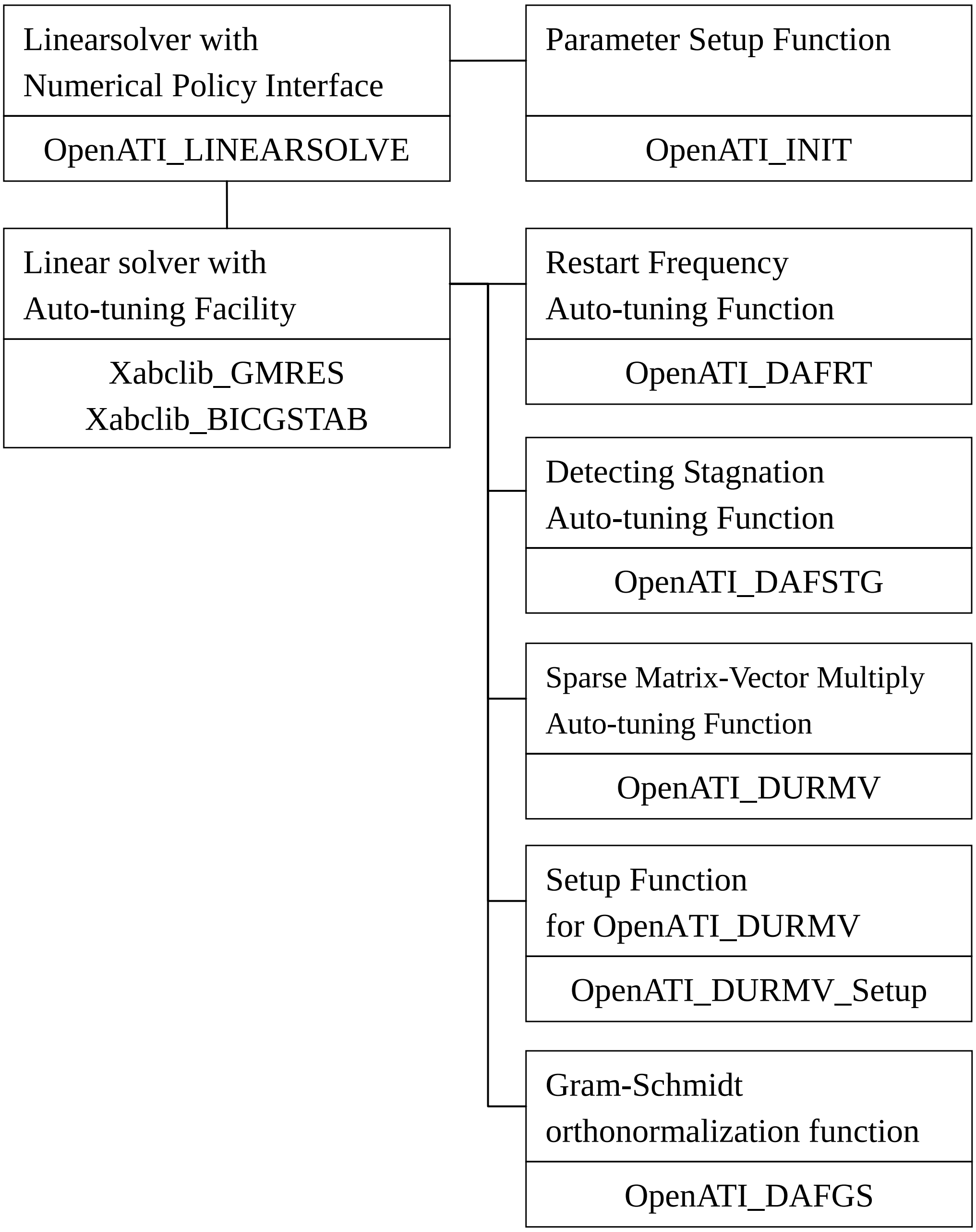

Fig. 1-2 Components of Function on Linearsolver.

2. Specification

2.1 Functions and Auguments of OpenATLib and Xabclib

In this section, library for functions and specification on a common auto-tuning interface, named OpenATLib, is explained. OpenATLib is an Application Programming Interface (API) to supply auto-tuning facility on arbitrary matrix computation libraries. For example, estimation function for the best values on algorithmic parameters, and best implementation for sparse matrix-vector multiplication (SpMxV).

(1) The function

Table 2-1 shows auto-tuning functions providing OpenATLib.

Table 2-1 Auto-tuning Function Providing OpenATLib

Function Name	Description
OpenATI_INIT	Set default parameter for OpenATLib and Xabclib.
OpenATI_DAFRT	Judge increment for restart frequency on Krylov subspace.
OpenATI_DAFSTG	Detect stagnation of relative residual for iterative method.
OpenATI_DSRMV	Judge the best implementation for double precision symmetric SpMxV on CRS format.
OpenATI_DURMV	Judge the best implementation for double precision non-symmetric SpMxV on CRS format.
OpenATI_DSRMV_Setup	Setup function for OpenATI_DSRMV.
OpenATI_DURMV_Setup	Setup function for OpenATI_DURMV.
OpenATI_DAFGS	Gram-Schmidt orthonormalization function with 4 implementations.
OpenATI_DAFMC_CCS2CRS	Convert matrix storage format from CCS into CRS.
OpenATI_LINEARSOLVE	Over-LinearSolver with numerical policy interface.
OpenATI_EIGENSOLVE	Over-EigenSolver with numerical policy interface.

The functions provided OpenATLib are classified for the following four categories:

a) Computation Function (Ex. OpenATI_D{S|U}RMV,)

b) Auxiliary Function (Ex. OpenATI_DAFRT, OpenATI_DAFSTG)

c) Setup Function (Ex. OpenATI_INIT, OpenATI_D{S|U}RMV_Setup)

d) Meta-interface (Ex. OpenATI_LINEARSOLVE)

For a) and b) functions, the function names are named by the manner on Table 2-1, following "OpenATI_" .

Table 2-2 Nomenclature of OpenATLib functions

First Character	The character shows data type. S : Single Precision D : Double Precision
Second and Third Characters	If the function is auxiliary, it comes "AF". If the function is computation, it comes matrix kinds in the second character, and matrix storage format in the third character. ● The second character: S : Symmetric. U : Non-symmetric. D : Diagonal. T : Tridiagonal. ● The third character: R : CRS Format. C : CCS Format.
Fourth and Fifth Characters	Process Kinds. MV: Matrix-vector multiplication. RT: Restart frequency.
Sixth and Above Characters	Property of Process kinds.

(2) Common Parametr List for OpenATLib and Xabclib

OpenATLib and Xabclib use common parameter lists named IATPARAM, RATPARAM. IATPARAM is integer parameter list, and RATPARAM is double precision parameter list. If you call OpenATI_INIT, this function sets these lists as default value.

Table 2-3 and 2-4 show description and default value of IATPARAM, RATPARAM.

Table 2-3 OpenATLib & Xabclib integer parameter list
(<L>: for Linear solver, <E>: for Eigen value solver)

	IATPARAM(50)		
index	**default**	**description**	**type**
1	mandatory		M
2	mandatory		M
		([3:20] OpenATLib's Information)	
3	(*1)	# of THREADS (SMP's) (*1): OMP_NUM_THREADS	I
4	0	Flag of Krylov subspace expand by MM-ratio (0:AT-off, 1:AT-on)	I
5	5	Incremental value for Krylov subspace when MM-ratio is less than threshold(RATPARAM(4))	I
6	10	A certain threshold value for judging stagnation.	I
7	3	OpenATI_DSRMV auto-tuned On/Off 0:AT-off 1:reserved 2:AT-on (select fastest type in '11' or '12') **3:AT-on (select fastest type in '11' , '12' or '13')**	I
8	12	Fastest OpenATI_DSRMV impl. Method (11 : block row decomp., **12 : nonzero decomp.**, 13 : parallel vector reduction)	I/O
9	3	OpenATI_DURMV auto-tuned On/Off 0:AT-off 1:AT-off and Auto-configure IATPARAM(11) 2:AT-on (select fastest type in '11' or '12') **3:AT-on (select fastest type in '11' , '12' or '13')** 4:AT-on (select fastest type in '11' , '12' or '13') And Auto-configure IATPARAM(11)	I
10	12	Fastest OpenATI_DURMV impl. Method (11 : block row decomp., **12 : nonzero decomp.**, 13 : BSS, 21 : original SS)	I/O
11	128	Columns of Segmented Scan's algorithms. If IATPARAM(9) is set as 1 or 4, IATPARAM(11) is set as (IATPARAM(11)) - Mod(IATPARAM(11),IATPARAM(3)) on OpenATI_DURMV and OpenATI_DURMV_Setup	I

12	2	Type of Gram-Schmidt procedure (0 : CGS, 1 : DGKS, **2 : MGS**, 3 : Blocked CGS)	I
13	-	DGKS refinement done or not (done : 1 , not : 0)	O
14	0	Access to meminfo(EIGENSOLVE/LINEARSOLVE) (done : 1 , **not : 0**)	I
15	-	Number of retried solver(EIGENSOLVE/LINEARSOLVE)	O
16	-	Total restart of solver(EIGENSOLVE/LINEARSOLVE)	O
17	-	Total Matrix-Vector times(EIGENSOLVE/LINEARSOLVE)	O
18	-	Last performed preconditioner type 1: None , 2 : Jacobi , 3 : SOR , 4 : ILU(0)_Diagonal, 5:ILU(0), 6:ILUT	O
19	-	Maximum number of fill-in's in each row(for ILUT preconditioner)	O
20	-	Last performed solver type 1:Xabclib_GMRES, 2:Xabclib_BICGSTAB	O
		([21:50] Xabclib's Information)	
21	-	# of OMP_NUM_THREADS	O
22	-1 (init)	Max. Iterations (if Solver recognize '-1' then set 'N')	I/O
23		# of Iterations	O
24	1	<L>preconditioner operations flag **1: not generated yet** , 2 : already generated	I
25	4	<L>preconditioner type in unsymmetric solver (Xabclib_GMRES, Xabclib_BICGSTAB) 1: None , 2 : Jacobi , 3 : SOR , **4 : ILU(0)_Diagonal**, 5:ILU(0), 6:ILUT in symmetric solver (Xabclib_CG) 1: None , 2 : Jacobi , 3 : Jacobi-Iterative , **4 : IC(0)_Diagonal**,	I
26	5	Integer parameter for preconditioner, for ILUT: Maximum number of fill-in's in each row for Jacobi-iterative: num of Jacobi iteration in a iteration of CG	I
27	20	Input size of Krylov subspace (in GMRES / Arnoldi) (caution) in Xabclib_ARNOLDI, must to be IATPARAM(27) >= NEV	I

28	2	Start size of Krylov subspace at subspace expand AT-on (in GMRES / Arnoldi). See IATPARAM(4) in Xabclib_ARNOLDI , if IATPARAM(28) less than NEV ,then start subspace size 'NEV' (overwritten).	I/(O)
29	-	Final size of Krylov subspace (in GMRES / Arnoldi)	O
30	1	<E> eigenvalue order option in Xabclib_LANCZOS **1: largest eigenvalue** 2: largest magnitude in Xabclib_ARNOLDI **1: largest real part eigenvalue** 2 : largest magnitude 3 : largest imaginary part	I
31	-	Total Matrix-Vector times	O
32		Krylov iteration times	O
33	0	When stagnation of relative residual occurs, solver is stopped. (**0: Off**, 1:On)	I
34	0	Minimum running iteration. (When IATPARAM(32)=1)	I
35-49	-	(reserved)	R
50	0	debug info (**0: Off**, 1:On)	I

Table 2-4 OpenATLib & Xabclib double precision parameter list
(<L>: for Linear solver, <E>: for Eigen value solver)

	RATPARAM(50)		
index	**default**	**description**	**type**
1	mandatory		M
2	mandatory		M
		([3:20] OpenATLib's Information)	
3		(reserved)	R
4	100.0	threshold of MM-ratio	I
5	-	Value of MM-ratio	O
6	0.01	“Exponent” for the Exponential Moving Average	I
7-13	-	(reserved)	R
14	-	Residual norm(EIGENSOLVE/LINEARSOLVE)	O
15	-	Set-up time(EIGENSOLVE/LINEARSOLVE)	O
16	-	Preconditioner time(EIGENSOLVE/LINEARSOLVE)	O
17	-	Solver time(EIGENSOLVE/LINEARSOLVE)	O
18	-	Total time(EIGENSOLVE/LINEARSOLVE)	O
19	-	Last Performed preconditioner parameter	O
20	-	(reserved)	R
		([21:50] Xabclib's Information)	
21	-	(reserved)	R
22	-1(∞)	Max. elapsed time (limit time)	I
23	1.0E-8	Convergence criterion	I
24		(reserved)	R
25	1.0E-8	<L>preconditioner parameter SOR(type=3): relaxation omega (1<= omega < 2) **ILU(0)(type=4) : Break down threshold (default 1.0E-8)** ILUT(type=6) : Dropping criterion	I
26-27		(reserved)	R
28	-	<L> 2-norm of RHS	O
29	-	2-norm of max. residual	O
30	-	Floating operations (× 10^9 operations)	O
31	-	<L> preconditioner time	O
32	-	Total solve time(elapsed)	O
33		(reserved)	R

34	0.0	Minimum running time. (When IATPARAM(32)=1)	I
35-50		(reserved)	R

(3) How to use the OpenATLib.

If you want to develop own library using OpenATLib, you should take the following processes.

1. Put the static library of "libOpenAT.a" to current directory.
2. Call "OpenATI_INIT" in program on own library source code for setting default parameters, like Fig. 2-1.
3. Call target functions of OpenATLib on own library source code.
4. Describe makefile to link "libOpenAT.a".

```
INTEGER IATPARAM(50)
DOUBLE PRECISION RATPARAM(50)
CALL OpenATI_INIT(IATPARAM,RATPARAM,INFO)
CALL OpenATI_LINEARSOLVE(N,NZ,IRP,ICOL,VAL,B,X,
     $                              IATPARAM,RATPARAM,INFO)
```

Fig. 2-1 An Example of using the OpenATLib .

2.2 Linking and Running OpenATLib and Xabclib

2.2.1 Directory structure

Directory structure of this software is described as following Fig. 2-2.

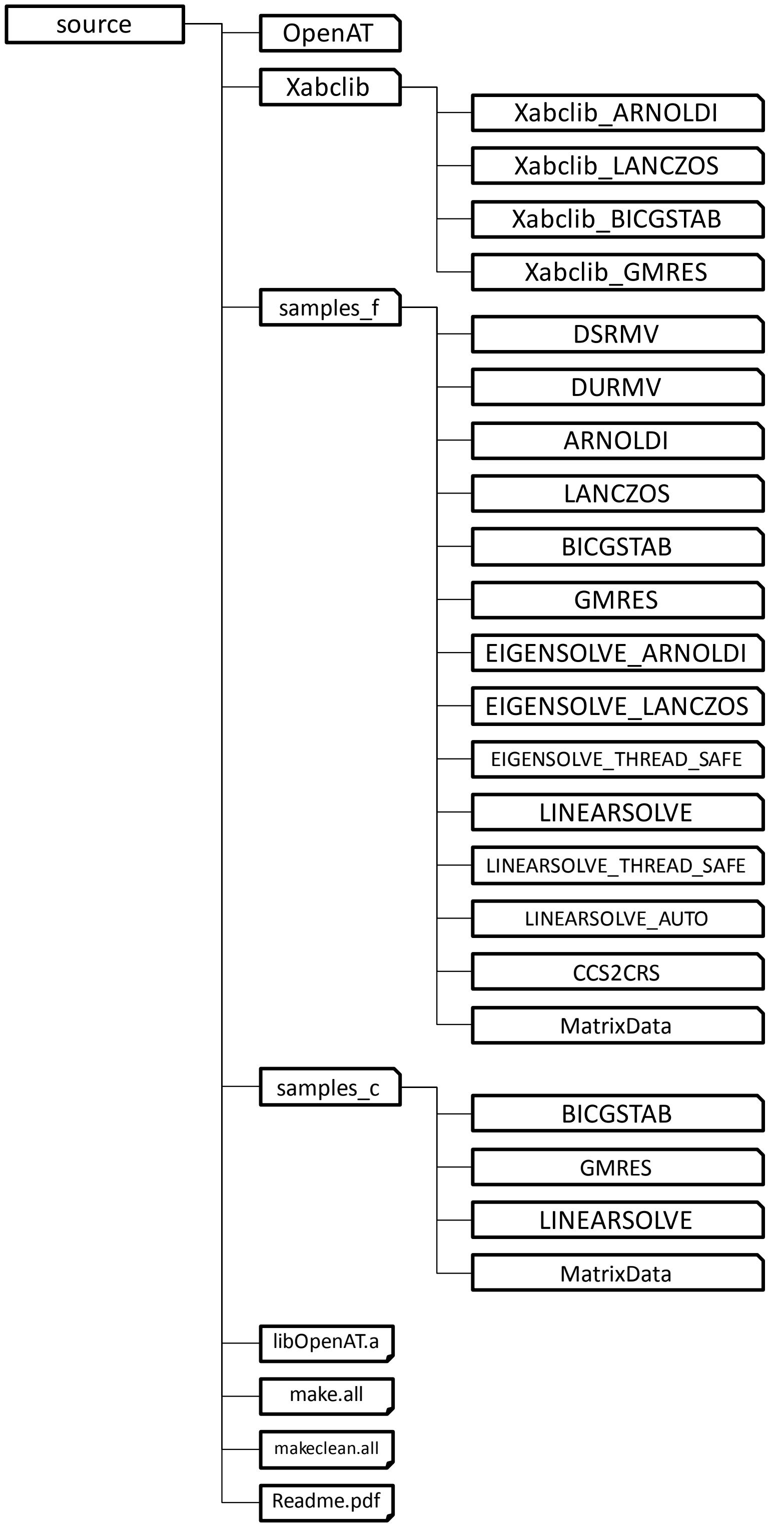

Fig 2-2. Directory structure of OpenATLib and Xabclib

2.2.2 Compiling

Compiling the current version OpenATLib and Xabclib requires the following installed version on your system.

a) Intel® Fortran Compiler version 11.0 or higher.
b) HITACHI Optimized Fortran (the environment variables "OPENATI_COMP" must be set to "HITACHI")

For compiling OpenATLib/Xabclib and making archive file "libOpenAT.a", you run shell script "make.all" on "source" directory.

2.2.3 Running sample programs

Sample programs are compiled by running the "make" command using the makefile on each sample directory. And, you try to run executable file by shell script "test.sh".

3 OpenATLib : A Common Auto-tuning Interface Library

3.1 OpenATI_INIT

3.1.1 Overview of the function

OpenATI_INIT sets default parameters for OpenATLib and Xabclib. This function must be called before using all functions of OpenATLib and Xabclib.

3.1.2 Argument Details and Error Code

(1) Argument Details

Argument	Type	IO	Description
IATPARAM (50)	Integer	OUTPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	OUTPUT	Array of double precision parameters for OpenATLib and Xabclib.
INFO	Integer	OUTPUT	Error code.

(2) Error Code

Value	Description
0	Normal return.

3.2 OpenATI_DAFSTG

3.2.1 Overview of the function

Recently, many iterative solvers and preconditioner methods are proposed. However, the history of relative residual shows the various movements by solvers, precondiotiners and matrices. Hence, we need to predict the solver will satisfy user's request or not from the history of relative residual so far.

OpenATI_DAFSTG enables us to detect the stagnation of relative residual from the history of them.

3.2.2 Overview of the auto-tuning method

OpenATI_DAFSTG uses gradient of the history as of then for detection. For example, at the fiftieth iteration, there are three histories like Fig.3-1.Like them, OpenATI_DAFSTG calculates gradient of them. Next, from the latest point of history, OpenATI_DAFSTG draws a prediction line with calculated gradient to the line of hundredth iterations as the time limit. If the point at the intersection of the prediction line with the line of time limit is less than the convergence criterion, OpenATI_DAFSTG estimates the iterative solver will converge. On the other hand, when the intersection point is greater than the criterion, OpenATI_DAFSTG estimates the solver will not converge.

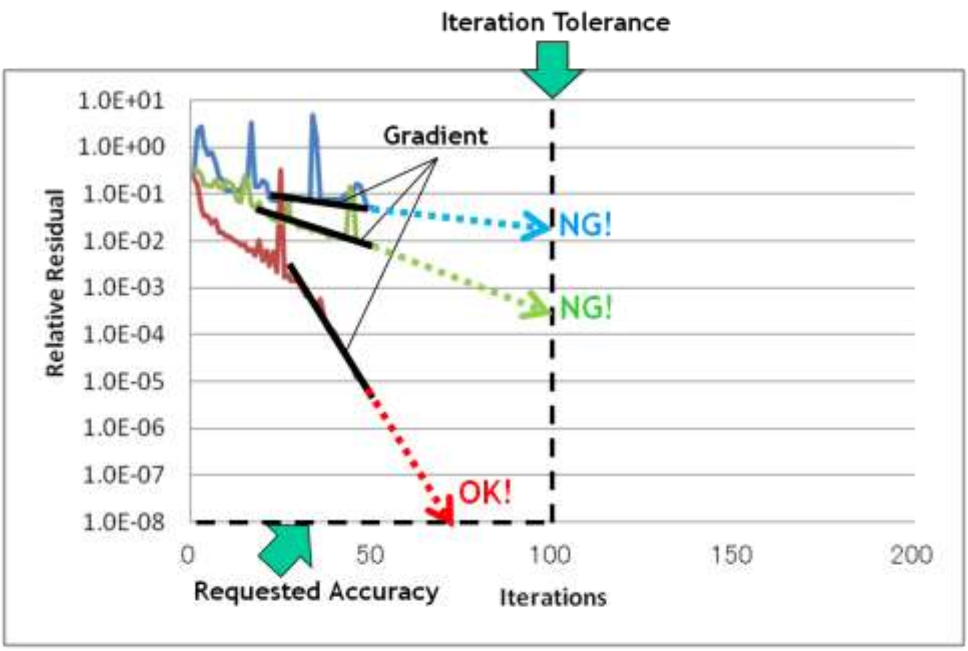

Fig. 3-1 The idea of this auto-tuning method

Next, the formulas of detection are explained. By time series analysis method, OpenATI_DAFSTG calculates the gradient of relative residual and predicts the value at time limit. OpenATI_DAFSTG uses Exponential Moving Average as time series analysis method for calculating the gradient. Because, this analysis method is easily calculated. And, it is not necessary to record the previous relative residual. Formulas of prediction as follow.

(1): $p = 0, e_0 = 0, G_0 = 0$

(2): Run 1 iterarion

(3): If $r_k < \varepsilon$ then output "convergence"

Else goto (4)

(4): $e_k = \log(r_k)$

(5): $G_k = \alpha(e_k - e_{k-1}) + (1-\alpha)G_{k-1}$

(6): $R = \min((T_{tol} - T_k)/t,\ I_{tol} - k)$

(T_{tol}: Time tolerant, t: Computation time for 1 iteration

I_{tol}: Iteration tolerant)

(7): $e_{tol} = e_k + G_k \times R$

(8): If $e_{tol} < \log(\varepsilon)$ then $p = 0$

Else $p = p + 1$

(9): If $p > p_{th}$ then output "stagnation"

Else goto (2)

Fig. 3-2 The formulas of detection

3.2.3 Argument Details and Error Code

(1) Argument Details

Argument	Type	IO	Description
ISTGCNT	Integer	INPUT/ OUTPUT	The counter for detecting stagnation of relative residual.
EMA	Double	INPUT/ OUTPUT	The exponential moving average of relative residual.
RERR	Double	INPUT	The error of the approximate solution vector.
PERR	Double	INPUT	The last error of the approximate solution vector.
STOP_TOL	Double	INPUT	Convergence criterion
ITER	Integer	INPUT	The number of iterations.
MAX_ITER	Integer	INPUT	Max. Iterations
ETIME	Double	INPUT	The elapsed time.
EITRTIME	Double	INPUT	The elapsed time per iteration.
MAX_ETIME	Double	INPUT	Max. elapsed time.
IATPARAM (50)	Integer	INPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT	Array of double precision parameters for OpenATLib and Xabclib.
INFO	Integer	OUTPUT	Error code.

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(6)	Integer	10	INPUT	A certain threshold value for judging stagnation. (In Fig.3-2, p_{th})

(3) Using parameters on RATPARAM

Number	Type	Initial Value	IO	Description
RATPARAM(6)	Double	0.01	INPUT	”Exponent” for the Exponential Moving Average (In Fig.3-2, α)

(4) Error Code

Value	Description
0	Normal return.

3.2.4 Usage Example

You can write the code like Fig. 3-3.

```
//Parameter Definition
Pth=10                      // Threshold for judging stagnation
PERR=1.0D0
ISTGCNT=0
EMA=0.0D0
ETIME1= OMP_GET_WTIME()
ETIME2= ETIME1
STOP_TOL=RATPARAM(23)
MAX_ITER=IATPARAM(22)
MAX_ETIME=RATPARAM(22)

                              - omission -

IF RERR < STOP_TOL   RETURN      // Convergence Test

ETIME3=ETIME2
ETIME2=OMP_GET_WTIME()
ETIME=ETIME2-ETIME1
EITRTIME=ETIME2-ETIME3

   CALL   OpenATI_ DAFSTG (ISTGCNT,EMA,RERR,PERR,STOP_TOL,
                           ITER,MAX_ITER,
                           ETIME,EITRTIME,MAX_ETIME,
                           IATPARAM,RATPARAM,INFO)

IF ISTGCNT >= Pth RETURN      // Stagnation

PERR=RERR
                              - omission -
```

Fig. 3-3　An Example of OpenATI_DAFSTG description.

3.3 OpenATI_DAFRT

3.3.1 Overview of the function

To perform Krylov subspace method, for example, Lanczos method for eigensolvers computation and GMRES method for linear equation solvers, they need to specify the dimension of the inner Krylov subspace to fix available memory space. If the iteration number is over for the fixed dimension, new computation is done with the current calculated approximation as initial vector to make new Krylov subspace. This process is called "restart", and the number of iterations is called "restart frequency". If the restart frequency is too small, it causes stagnation of reduction for residual vector, which is calculated by real solution and approximation vectors, then the number of iterations is increased. On the other hand, if the restart frequency is too big, it causes heave computation to make big Krylov subspaces, hence the execution time is very increased. The best frequency depends on input sparse matrix numerical condition, and it is very tough to estimate the best frequency without execution. Hence in the library point of view, we need on the fly, namely run-time, auto-tuning facility.

OpenATI_DAFRT enables us to judge the incensement of frequency based on the current information of Krylov subspace.

3.3.2 Overview of the auto-tuning method

The previous estimation for the best restart frequency is difficult; it can detect stagnation based on the run-time history of residuals. The method is proposed in [1].

The norm of the stagnation is defined by the value that maximum value divided by minimal vale from t-th time to s-th time. The values called "Ratio of Max-Min in residual". Hereafter, we describe the ratio "**MM ratio**" for simplification.

The MM ratio to past t-th time, namely $Ri\,(s,t)$, can be described with i-th residual r_i as follows:

$$R_i\left(s,t\right)=\frac{\max_z\left\{r_i\left(z\right);z=s-t+1,\cdots,s\right\}}{\min_z\left\{r_i\left(z\right);z=s-t+1,\cdots,s\right\}}.$$

If restart frequency is big enough, the residual tends to reduce bigly, hence MM ratio is going to be big. If restart frequency is small, it tends to cause stagnation, hence MM ratio is going to be small. Hence, we can control restart frequency at run-time monitor for the MM ratio. If the MM ratio is going to be small to a fixed value at run-time, the frequency should be increased.

3.3.3 Argument Details and Error Code

(1) Argument Details

Argument	Type	IO	Description
NSAMP	Integer	INPUT	The number of sampling points.
SAMP (NSAMP)	Double	INPUT	The values of sampling points.
IRT	Integer	OUTPUT	0 : Do not need to increase restart frequency. 1 : Need to increase restart frequency.
IATPARAM (50)	Integer	INPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT/ OUTPUT	Array of double precision parameters for OpenATLib and Xabclib.
INFO	Integer	OUTPUT	Error code.

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(4)	Integer	1	INPUT	1 : Judge incensement of restart frequency based on MM ratio.
IATPARAM(5)	Integer	5	INPUT	Incremental value for Krylov subspace when MM-ratio is less than threshold(RATPARAM(4))

(3) Using parameters on RATPARAM

Number	Type	Initial Value	IO	Description
RATPARAM(4)	Double	100.0	INPUT	Threshold value for MM ratio.
RATPARAM(5)	Double	-	OUTPUT	Value of MM ratio.

(4) Error Code

Value	Description
0	Normal return.

3.3.4 Usage Example

Judge incensement of restart frequency per 5 iterations. If it is needed to increase, the frequency is increased by stridden 1. In this case, you can write the code like Fig. 3-4.

```
//Parameter Definition
MSIZE=1                    // Initial restart frequency.
I=5                        // Judgment frequency.

                              - omission -

IF RSDID < TOL   RETURN     // Convergence Test

SAMP (K)=RSDID      //Set residual to SAMP(K).

IF (mod (K, I) .eq. 0)   THEN     //Call DAFRT per I times.
        IRT=0
        CALL   OpenATI_ DAFRT (I, SAMP,IRT,
                                    IATRARAM,RATPARAM,INFO)

        IF IRT= 1   MSIZE=MSIZE+1     //Increase restart frequency.
        K=0
END IF

K=K+1

                              - omission -
```

Fig. 3-4 An Example of OpenATI_DAFRT description.

3.4 OpenATI_DSRMV and OpenATI_DURMV, OpenATI_DSRMV_Setup, OpenATI_DURMV_Setup

3.4.1 Overview of the function

Sparse matrix-vector multiplication (SpMxV) is crucial function and widely-used in many iterative methods. Its execution time directly affects total execution time in many cases. There are many implementations to perform SpMxV. The best implementation depends on computer environment and numerical characteristics of input sparse matrix. It is hence difficult to fix the best method. We need auto-tuning method at run-time to adapt user's computer environment and matrices.

OpenATI_DSRMV is designed for double symmetric SpMxV, and OpenATI_DURMV is designed for double non-symmetric SpMxV auto-tuning APIs for their implementations at run-time.

3.4.2 Overview of auto-tuning method

In this function, the API surveys all candidates of SpMxV implementations in the first iteration time, then select the best implementation after that. This method was proposed by [2].

The following several implementations are supplied for OpenATI_DSRMV(3 kinds) and OpenATI_DURMV(4 kinds) in version beta.

- OpenATI_DSRMV
  S1) Row Decomposition Method.
  S2) Normalized NZ Method.
  S3) Normalized NZ Method, with vector reduction parallelization.

- OpenATI_DURMV
  U1) Row Decomposition Method.
  U2) Normalized NZ Method (for scalar multi-core processors).
  U3) Branchless Segmented Scan (for scalar multi-core processors).
  U4) Original Segmented Scan (for vector processors).

[Row Decomposition Method and Normalized NZ Method]

- Row Decomposition Method

  Input Matrix is divided into the number of threads blocks for balancing the number of row processed by each thread.

- Normalized NZ Method

  Input Matrix is divided into the number of threads blocks for normalizing the number of non-zero element processed by each thread.

Figure 3-5 shows an example of Row Decomposition Method and Normalized NZ Method in case of 6 dimension matrix processed by 4 threads.

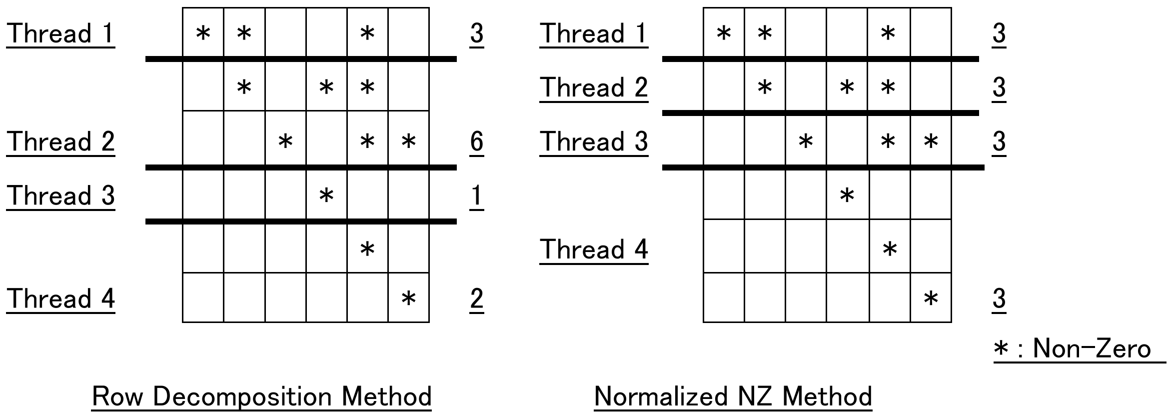

Fig 3-5　An example of Row Decomposition Method and Normalized NZ Method

[Original Segmented Scan method, Branchless Segmented Scan method]

Original Segmented Scan[5] is designed for sparse matrix multiplication on vector multiprocessors. In this method, input matrix is divided into fixed length of Non-Zero element group. These Non-Zero element group are named segment-vector, In a code of Original Segmented Scan, innermost loop has fixed length of loop and mask process with FLAG representing the beginning of row. (Fig 3-6 shows an example of segment-vector of length 6 processed by 5 threads).

Branchless Segmented Scan is the method modified for scalar multi-core system by removing IF operator for mask process in innermost loop. In this method, row pointer array in CSR format is extended for segment-vector (In Fig3-6, IRP is expanded MFLAG) .

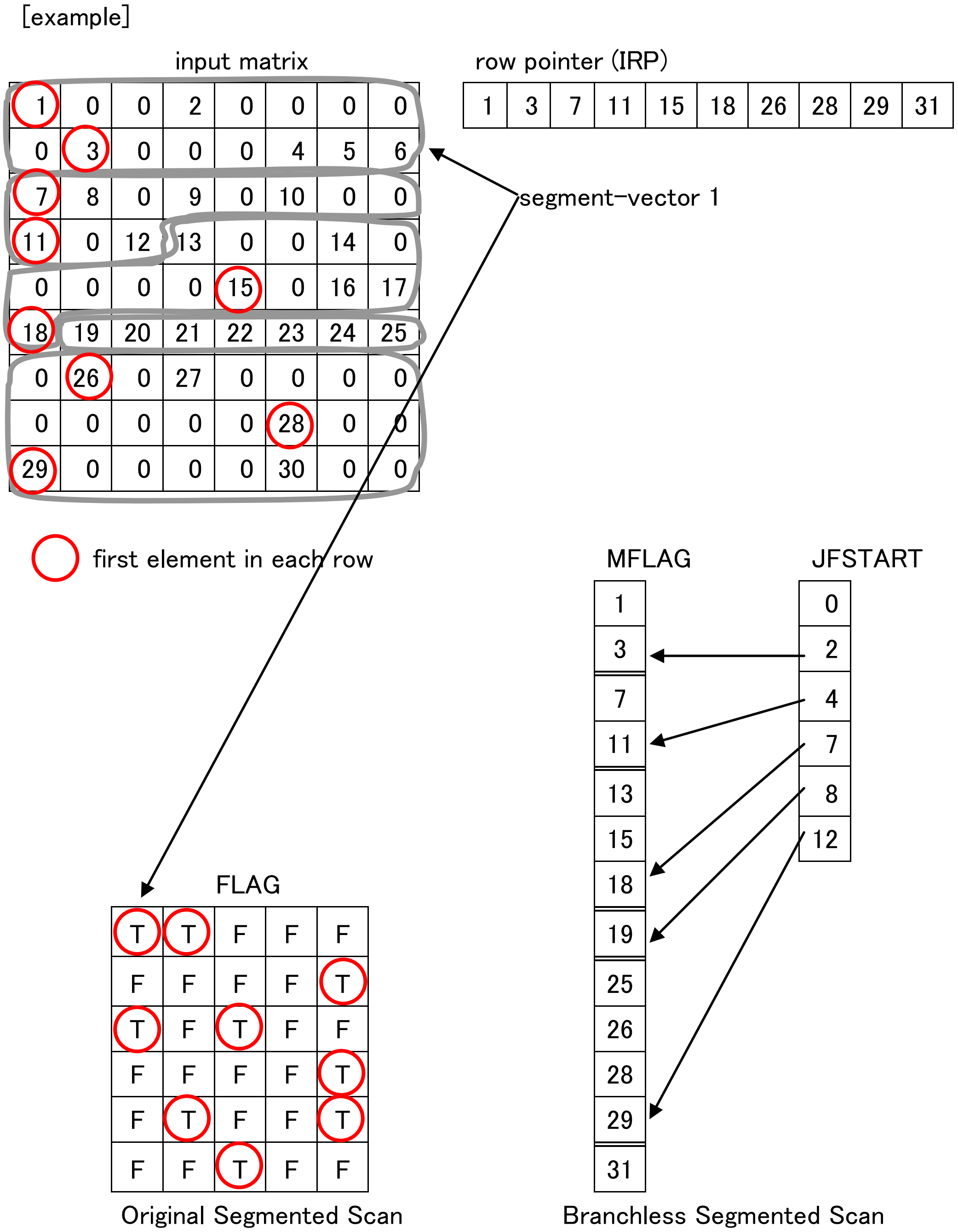

Fig 3-6 An example of Original Segmented Scan and Branchless Segmented Scan.

If you want to specify SpMxV implementation of OpenATI_DSRMV or OpenATI_DURMV, you need to run setup function before call OpenATI_DSRMV or OpenATI_DURMV.

OpenATI_DSRMV_Setup

(S1) No necessary to run setup function.

(S2) Fix the groups of rows processed by each thread for normalized non-zero elements.

(S3) Fix the groups of rows processed by each thread for normalized non-zero elements, and the start and end point of reduction part of each thread.

OpenATI_DURMV_Setup

(U1) No necessary to run setup function.

(U2) Fix the groups of rows processed by each thread for normalize non-zero elements.

(U3) Set array of MFLAG and JFSTART for Branchless Segmented Scan.

(U4) Set array of FALG for Original Segmented Scan

3.4.3 Argument Details and Error Code of OpenATI_DSRMV_Setup

(1) Argument Details

Argument	Type	IO	Description
N	Integer	INPUT	The number of dimension for the matrix. （N>=1）
NNZ	Integer	INPUT	The number of non-zero elements for the matrix.
IRP(N+1)	Integer	INPUT	Pointers to first elements on each row for the matrix.
ICOL(NNZ)	Integer	INPUT	The non-zero row indexes for the matrix.
IATPARAM (50)	Integer	INPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT	Array of double precision parameters for OpenATLib and Xabclib.
SINF (LSINF)	Double	OUTPUT	If IATPARAM(8)=11 No returns. If IATPARAM(8)=12,13 Returns the groups of rows processed each thread for OpenATI_DSRMV.
LSINF	Integer	INPUT	The size of SINF IATPARAM(8)=11: LSINF >= 0 IATPARAM(8)=12: LSINF >= int(0.5*NUM_SMP)+1 IATPARAM(8)=13: LSINF >= N+NUM_SMP+3 (NUM_SMP=IATPARAM(3))
INFO	Integer	OUTPUT	Error Code

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(8)	Integer	12	INPUT	Set the number corresponding implementation of SpMxV in OpenATI_DSRMV. 11: No necessary to run this function. 12: Create information for

				Normalized NZ Method. 13: Create information for Normalized NZ Method with vector reduction parallelization

(3) Using parameters on RATPARAM

OpenATI_DSRMV_Setup doesn't use RATPARAM.

(4)Error Code

Value	Description
0	Successful exit.
100	Invalid IATPARAM(8) value is inputted.
200	Invalid LSINF value is inputted. (IATPARAM(8)=12 or 13)

3.4.4 Argument Details and Error Code of OpenATI_DURMV_Setup

(1) Argument Details

Argument	Type	IO	Description
N	Integer	INPUT	The number of dimension for the matrix. (N>=1)
NNZ	Integer	INPUT	The number of non-zero elements for the matrix.
IRP(N+1)	Integer	INPUT	Pointers to first elements on each row for the matrix.
IATPARAM (50)	Integer	INPUT/ OUTPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT	Array of double precision parameters for OpenATLib and Xabclib.
UINF (LUINF)	Double	OUTPUT	IATPARAM(10)=11: No returns. IATPARAM(10)=12,13,21: Returns the groups of rows processed each thread or information array for segmented scan.
LUINF	Integer	INPUT	The size of UINF IATPARAM(10)=11: LUINF >= 0 IATPARAM(10)=12: LUINF >= int(0.5*NUM_SMP)+1 IATPARAM(10)=13: LUINF >= int(1.5*N)+ int(4.25*JL)+10 (JL= IATPARAM(11)) IATPARAM(10)=21: LUINF >= int(1.125*NNZ)+ int(2.125*JL)+10 (NUM_SMP=IATPARAM(3), JL= IATPARAM(11))
INFO	Integer	OUTPUT	Error Code

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(9)	Integer	3	INPUT	OpenATI_DURMV auto-tuned On/Off 0: Perform SpMxV specified by IATPARAM(10). 1: Perform SpMxV specified by IATPARAM(10), and auto-configure IATPARAM(11). 2: Perform SpMxV to judge the best methods between three methods, except for Original Segment Scan. 3: Perform SpMxV to judge the best method among four implementations. 4 : Perform SpMxV to judge the best method among four implementations, and auto-configure IATPARAM(11).
IATPARAM(10)	Integer	12	INPUT /OUTPUT	If IATPARAM(9)=0 or 1, then set the number of implementations. If IATPARAM(9)=2,3 or 4, the best number of implementations returns. 11: Row Decomposition Method. 12: Normalized NZ Method. 13: Branchless Segmented Scan. 21: Original Segmented Scan.
IATPARAM(11)	Integer	128	INPUT	Columns of Segmented Scan's algorithms. If IATPARAM(9) is set as 1 or 4, IATPARAM(11) is set as (IATPARAM(11)) – Mod(IATPARAM(11),IATPARAM(3))

(3) Using parameters on RATPARAM

OpenATI_DURMV_Setup doesn't use RATPARAM.

(4)Error Code

Value	Description
0	Successful exit.
100	Invalid IATPARAM(10) value.
200	LUINF value exceeds upper limit of Integer.
300	Invalid LUINF value (IATPARAM(10)=12,13,21).

3.4.5 Argument Details and Error Code for OpenATI_DSRMV

(1) Argument Details

Argument	Type	IO	Description
N	Integer	INPUT	The number of dimension for the matrix. (N>=1)
NNZ	Integer	INPUT	The number of non-zero elements for the matrix.
IRP(N+1)	Integer	INPUT	Pointers to diagonal elements on each row for the matrix.
ICOL(NNZ)	Integer	INPUT	The non-zero row indexes for the matrix.
VAL(NNZ)	Double	INPUT	The non-zero elements for the matrix.
X(N)	Double	INPUT	Right hand side vector elements.
Y(N)	Double	OUTPUT	Solution vector elements for SpMxV.
IATPARAM (50)	Integer	INPUT/ OUTPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT	Array of double precision parameters for OpenATLib and Xabclib.
WK(N, IATPARAM (3))	Double	WORK	If IATPARAM(7)=0 and IATPARAM(8)=13, or IATPARAM(7)=3, then set workspace to the argument.
SINF (LSINF)	Double	INPUT/ OUTPUT	If IATPARAM(7)=0 (INPUT) IATPARAM(8)=11 : Not necessary to set. IATPARAM(8)=12,13 : Set SINF retuned by OpenATI_DSRMV_Setup. If IATPARAM(7)=2,3 (INPUT) Not necessary to set. (OUTPUT) Returns setup information for best implementation.
LSINF	Integer	INPUT	The size of SINF If IATPARAM(7)=0 IATPARAM(8)=11: LSINF >= 0 IATPARAM(8)=12: LSINF >= int(0.5*NUM_SMP)+1

			IATPARAM(8)=13: LSINF >= N+NUM_SMP+3 If IATPARAM(7)=2 LSINF >= int(0.5*NUM_SMP)+1 If IATPARAM(7)=3 LSINF >= N+NUM_SMP+3
INFO	Integer	OUTPUT	Error code.

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(7)	Integer	3	INPUT	OpenATI_DSRMV auto-tuned On/Off 0: Perform SpMxV specified by IATPARAM(8). 1: reserved 2: Perform SpMxV to judge the best methods between two methods, except for reduction parallel implementation. 3: Perform SpMxV to judge the best method among three methods. Note that workspace according to the number of threads is needed.
IATPARAM(8)	Integer	12	INPUT /OUTPUT	If IATPARAM(7)=0, then set the number of implementations. If IATPARAM(7)=2 or 3, the best number of implementations returns. 11: Row Decomposition Method. 12: Normalized NZ Method. 13: Normalized NZ Method, with vector reduction parallelization.

(3) Using parameters on RATPARAM

OpenATI_DSRMV doesn’t use RATPARAM.

(4) Error Code

Value	Description
0	Successful exit.
100	The value of IATPARAM(8) is illegal. (If IATPARAM(7)=0.)
200	The value of IATPARAM(7) is illegal.

3.4.6 Argument Details and Error Code for OpenATI_DURMV

(1) Argument Details

Argument	Type	IO	Description
N	Integer	INPUT	The number of dimension for the matrix.（N>=1)
NNZ	Integer	INPUT	The number of non-zero elements for the matrix.
IRP(N+1)	Integer	INPUT	Pointers to first elements on each row for the matrix.
ICOL(NNZ)	Integer	INPUT	The non-zero row indexes for the matrix.
VAL(NNZ)	Double	INPUT	The non-zero elements for the matrix.
X(N)	Double	INPUT	Right hand side vector elements.
Y(N)	Double	OUTPUT	Results vector elements for SpMxV.
IATPARAM (50)	Integer	INPUT/ OUTPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT	Array of double precision parameters for OpenATLib and Xabclib.
UINF (LUINF)	Double	INPUT/ OUTPUT	If IATPARAM(9)=0 or 1 (INPUT) IATPARAM(10)=11 ：Not necessary to set IATPARAM(10)=12,13,21 ：Set UINF returned by OpenATI_DURMV_Setup. If IATPARAM(9)=2,3 or 4 (INPUT) Not necessary to set. (OUTPUT) Returns setup information for best implementation.
LUINF	Integer	INPUT	The size of UINF If IATPARAM(9)=0 or 1 IATPARAM(10)=11: LUINF >= 0 IATPARAM(10)=12: LUINF >= int(0.5*NUM_SMP)+1 IATPARAM(10)=13: LUINF >= int(1.5*N)+ int(4.25*JL)+10

			(JL= IATPARAM(11)) IATPARAM(10)=21: LUINF >= int(1.125*NNZ)+ int(2.125*JL)+10 If IATPARAM(9)=2. LUINF >= int(0.5*NUM_SMP)+1 If IATPARAM(9)=3 or 4, LUINF >= int(1.5*N)+ int(4.25*JL)+10 (NUM_SMP=IATPARAM(3), JL= IATPARAM(11))
INFO	Integer	OUTPUT	Error Code.

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(9)	Integer	3	INPUT	OpenATI_DURMV auto-tuned On/Off 0: Perform SpMxV specified by IATPARAM(10). 1 : Perform SpMxV specified by IATPARAM(10), and auto-configure IATPARAM(11). 2 : Perform SpMxV to judge the best methods between three methods, except for Original Segment Scan. 3 : Perform SpMxV to judge the best method among four implementations. 4 : Perform SpMxV to judge the best method among four implementations, and auto-configure IATPARAM(11).
IATPARAM(10)	Integer	12	INPUT /OUTPUT	If IATPARAM(9)=0 or 1, then set the number of implementations. If IATPARAM(9)=2,3 or 4, the best number of implementations returns.

				11: Row Decomposition Method. 12: Normalized NZ Method. 13: Branchless Segmented Scan. 21: Original Segmented Scan.
IATPARAM(11)	Integer	128	INPUT	Columns of Segmented Scan's algorithms. If IATPARAM(9) is set as 1 or 4, IATPARAM(11) is set as (IATPARAM(11)) – Mod(IATPARAM(11),IATPARAM(3)) on OpenATI_DURMV and OpenATI_DURMV_Setup.

(3) Using parameters on RATPARAM

OpenATI_DURMV doesn't use RATPARAM.

(4)Error Code

Value	Description
0	Successful exit.
100	The value of IATPARAM(10) is illegal. (If IATPARAM(9)=0.)
200	The value of IATPARAM(9) is illegal.

3.4.7 Usage Example

Search the best implementation of SpMxV in the first iteration time, then the best implementation is used after that based on the run-time searching. To implement this, see the code of Fig. 3-7.

```
//Parameter definition.
IATPARAM(7)=3                    //Initialize DSRMV parameter.
LSINF= N+NUM_SMP+3
ALLOCATE(SINF(LSINF))
                                 - omission -

//The first SpMxV.
CALL   OpenATI_DSRMV (N, NNZ, IRP, ICOL, VAL, X, Y,
                      IATRARAM, RATPARAM, WK, SINF, LSINF,
                      INFO)
IATPARAM(7)=0   //Hereafter, we select the best one.

                                 - omission -

// SpMxV after run-time searching.
// We can use the best implantation based on previous information.
CALL   OpenATI_DSRMV (N, NNZ, IRP, ICOL, VAL, X, Y, NUM_SMP,
                      IATRARAM, RATPARAM, WK, SINF, LSINF,
                      INFO)

                                 - omission -
```

Fig. 3-7 An Example of OpenATI_DSRMV Description.

If you want to specify SpMxV implementation in OpenATI_DSRMV, implement the code like Fig.3-8.

```
// Parameter definition.
IATPARAM(7)=0                      // Initialize DSRMV parameter.
IATPARAM(8)= 13                    // Initialize DSRMV parameter.

                      - omission -

// Call SpMxV.
LSINF=N+NUM_SMP+3         //Allocate memory for setup
ALLOCATE(SINF(LSINF))
CALL    OpenATI_DSRMV_Setup(N,NNZ,IRP,ICOL,
                            IATPARAM,RATPARAM,
                            SINF,LSINF,INFO)
CALL   OpenATI_DSRMV (N, NNZ, IRP, ICOL, VAL, X, Y,
                      IATRARAM, RATPARAM, WK, SINF, LSINF,
                      INFO)
                      - omission -
```

Fig.3-8　An example of OpenATI_DSRMV Description with specified SpMxV implementation.

3.5 OpenATI_ DAFGS

3.5.1 Overview of the function

Vector orthonormalization spends a lot of CPU time in many Krylov Subspace methods. Gram-Schmidt orthonormalization method[7] is typical orthonormalization method. There are many implementations to perform Gram-Schmidt method, and trade-offs must be made between computational complexity and accracy. Hence, It is difficult to fix the best implementation.

OpenATI_DAFGS is API that supplies selectable from 4 kinds Gram-Schmidt orthonormalization implementation.

3.5.2 Overview of Reorthonormalization method

In this function, the API has 4 kinds Gram-Schmidt orthonormalization method. Selected method is indicated by value of IATPARAM(12). By default , Modified Gram-Schmidt method is selected.

(1) Classical Gram-Schmidt method (CGS)

When Krylov Subspace size is large, accuracy of orthonormalization is lowering. Acceleration performance by parallelization is excellent.

(2) DGKS method

This method supplies improved accuracy by running CGS 2 times. DGKS method computational complexity needs twice as many as CGS' one.

(3) Modified Gram-Schmidt method (MGS)

MGS is most popular Gram-Schmidt method. This method is most effective performance and accuracy.

(4) Blocked Classical Gram-Schmidt method (BCGS)

BCGS method is orthonormalized by intra-block with CGS, by inter-block with MGS. Block length is 4.

3.5.3 Argument Details and Error Code

(1) Argument Details

Argument	Type	IO	Description
NORMALFLG	Integer	INPUT	Normalization of Output vector 0：not normalized 1：normalized
N	Integer	INPUT	Vector length（N>=1）
X(N)	Double	INPUT	Vector for normalization
Q(LQ,MM)	Double	INPUT	Orthonormalized vectors Q(1:N,MM)
LQ	Integer	INPUT	Leading Dimension of Q
MM	Integer	INPUT	The number of vector of Q
HR(MM)	Double	OUTPUT	Inner product X by Q(1:N,M)
IATPARAM(50)	Integer	INPUT/OUTPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT	Array of double precision parameters for OpenATLib and Xabclib.

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(12)	Integer	2	INPUT	0：Classical Gram-Schmidt 1：DGKS 2：Modified Gram-Schmidt 3：Blocked Gram-Schmidt
IATPARAM(13)	Integer	-	OUTPUT	Iterative refinement of DGKS 0：no Iterative refinement 1：Iterative refinement

(3) Using parameters on RATPARAM

OpenATI_ DAFGS doesn't use RATPARAM.

3.6 OpenATI_DAFMC_CCS2CRS

3.6.1 Overview of the function

OpenATI_DAFMC_CCS2CRS converts sparse matrix storage format from CCS(Compressed Column Storage) into CRS(Compressed Row Storage).

3.6.2 Argument Details and Error Code

(1) Argument Details

Argument	Type	IO	Description
IATPARAM (50)	Integer	INPUT	Array of integer parameters for OpenATLib and Xabclib.
N	Integer	INPUT	The order of the matrix. (N >= 1)
NNZ	Integer	INPUT	Non-Zero elements of the matrix. (NNZ>=N)
IPTR(N+1)	Integer	INPUT	Pointers of first element on each column of the matrix in CCS format.
INDEX(NNZ)	Integer	INPUT	Row indexes of elements in CCS format.
VALUE(NNZ)	Double	INPUT	Value of elements in CCS format.
IRP(N+1)	Integer	OUTPUT	Pointers of first element on each row of the matrix in CRS format.
ICOL(NNZ)	Integer	OUTPUT	The non-zero column indexes for the matrix in CRS format.
VAL(NNZ)	Double	OUTPUT	Value of elements in CRS format.

3.7 OpenATI_LINEARSOLVE and OpenATI_EIGENSOLVE : Sparse iterative solvers with Numerical policy

3.7.1 Overview of the function

Numerical policy is requirement and priority of memory, CPU time, accuracy and others specified by library user. OpenATI supplies OpenATI_LINEARSOLVE is designed for unsymmetric liner problem, and OpenATI_EIGENSOLVE is designed for symmetric/unsymmetric eigenvalue problem as sparse iterative solvers with numerical policy.

OpenATI_LINEARSOLVE and OpenATI_EIGENSOLVE are Meta-Solvers that call Xabclib and set optimized arguments automatically on user's numerical policy.

3.7.2 Overview of numerical policy

If you want to use Meta-Solvers, you make numerical policy file with following format, and input numerical policy file's name is "OPENATI_POLICY_INPUT.#"(#: Thread number).

Policy file's format is as follow.

```
<keyword> = <value>
```

There are `POLICY/CPU/RESIDUAL/MAXMEMORY/MAXTIME/PRECONDITIONER/SOLVER` as configurable keywords. Unregistered `<keyword>` in policy file is inputted the default value. The explanation of all `<keyword>` is as follow.

```
POLICY = <value>
  <value> : TIME / ACCURACY / MEMORY / STABLE
   "TIME" is selected by default.
  ① If POLICY = TIME, Meta-Solvers preference for execution time
     over accuracy and saving memory. Therefore, algorithms for
     high performance are positively selected.
  ② If POLICY = ACCURACY, Meta-Solvers recalculation solution of
     solvers. If false convergence occurs, Meta-Solvers continue
     to reexcute with more exact convergence test until true
     convergence.
  ③ If POLICY = MEMORY, Meta-Solvers set arguments with less
     memory usage.
```

```
    ④ If POLICY = STABLE, Meta-Solvers set arguments without AT.
       In this case, Meta-Solvers set IATPARAM as following value.
       IATPARAM(4),(7) and (9)=0
       IATPARAM(27) and (28)=30(LINEARSOLVE) or NEV*5(EIGENSOLVE)
       The others are set as default value.
```

```
  CPU = <value>
    <value> : entry OMP_NUM_THREADS at run-time.
   OMP_GET_NUM_THREADS is selected by default.
    Note) 1 <= <value> <= OMP_GET_MAX_THREADS()
```

```
RESIDUAL = <value>
    <value> : entry require accuracy by real value.
    The default value is 1.0D-8.
     In case of "POLICY = ACCURACY" is set and false convergence
     occur, solver continue to re-excute with more exact convergence
     test until true convergence.
```

```
MAXMEMORY = <value>
    <value>: entry require memory usage in [Gbyte].
    The default value is "memfree" in /proc/meminfo (Linux).
    If fails to get property in /proc/meminfo, search and allocate
    free memory dynamically.
    Note) The maximum limit of MAXMEMORY is 16Gbyte.
```

```
MAXTIME = <value>
    <value> : entry time tolerance in [sec].
    The default value is infinite.
     When execution time exceeds time tolerance, computation is
stopped.
```

```
PRECONDITIONER = <value>
    <value> : NO / JACOBI / SSOR / ILU0D / ILU0 / ILUT / AUTO
     ILU0 is selected by default. This keyword is used by only
OpenATI_LINEARSOLVE.
    ① PRECONDITIONER = NO  : No preconditioner
```

```
② PRECONDITIONER =  JACOBI  :JACOBI
③ PRECONDITIONER =  SSOR  :SSOR
④ PRECONDITIONER =  ILU0D :ILU(0)_Diagonal
⑤ PRECONDITIONER =  ILU0  :ILU(0)
⑥ PRECONDITIONER =  ILUT  :ILUT
⑦ PRECONDITIONER =  AUTO  :Automatic select (*1)
```

```
SOLVER = <value>
   <value> : XABCLIB_GMRES / XABCLIB_BICGSTAB / AUTO (*1)
             (OpenATI_LINEARSOLVE)
             XABCLIB_LANCZOS/ XABCLIB_ARNOLDI (OpenATI_EIGENSOLVE)
   The default value is XABCLIB_GMRES(OpenATI_LINEARSOLVE).
```

(*1)Detail of this policy is explained in 3.7.3.

3.7.3 Automatic selection of preconditioner and solver

OPENATI_LINEARSOLVE has the function of performing preconditioned iterative solvers under the given order.
This function can call two or more iterative solvers and preconditioners and performs these solvers and preconditioners in order for satisfying time tolerant and required accuracy. Algorithm of automatic selection of preconditioner and solver policy as follow.

1. $r_{\min} = 1.0\text{D}0,\ S_{retry} = 0$
   Set strategy $S_1, \cdots, S_m$ (S_i involves type of solver and preconditioner)
2. For i=1,m
3. Call solver according to S_i with a function of detecting stagnation.
4. If stagnation occured then go to 5
   Else go to 8
5. If rerative residual $r_i < r_{\min}$ then
   $r_{\min} = r_i,\ S_{retry} = S_i$
6. End For
7. If $S_{retry} \neq 0$ then
   Call solver according to S_{retry} without a function of detecting stagnation.
8. Output solution and report

In the following, the order of strategy is listed.

STRATEGY	PRECONDITIONER	SOLVER
1	SSOR	BiCGStab
2	SSOR	GMRES(m)
3	ILU0-Diagonal	BiCGStab
4	ILU0-Diagonal	GMRES(m)
5	ILU0	BiCGStab
6	ILU0	GMRES(m)
7	ILUT(10,1.0E-08)	BiCGStab
8	ILUT(10,1.0E-08)	GMRES(m)

3.7.4 Argument Details and Error Code of OpenATI_LINEARSOLVE

CALL OpenATI_LINEARSOLVE (N,NNZ,IRP,ICOL,VAL,B,X,
IATPARAM,RATPARAM,INFO)

(1) Argument Details

Argument	Type	IO	Description
N	Integer	INPUT	The number of dimension for the matrix. (N≧1)
NNZ	Integer	INPUT	The number of non-zero elements for the matrix.
IRP(N+1)	Integer	INPUT	Pointes to first position on each row for the matrix. Note: Satisfy IRP(1)=1, IRP(N+1)=NNZ+1.
ICOL(NNZ)	Integer	INPUT	The row indexes for non-zero elements for the matrix.
VAL(NNZ)	Double	INPUT	The non-zero elements for the matrix.
B(N)	Double	INPUT	The elements for right hand size vector *b*.
X(N)	Double	INPUT / OUTPUT	INPUT: Set the elements of initial guess for solution vector *x_0*. OUTPUT: Return the elements of solution vector *x*.
IATPARAM (50)	Integer	INPUT/ OUTPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT/ OUTPUT	Array of double precision parameters for OpenATLib and Xabclib.
INFO	Integer	OUTPUT	Error Code

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(14)	Integer	0	INPUT	Access to meminfo (for Linux system) (1:done, 0:not)
IATPARAM(15)	Integer	-	OUTPUT	Number of retried solver
IATPARAM(16)	Integer	-	OUTPUT	Total restart of solver
IATPARAM(17)	Integer	-	OUTPUT	Total Matrix-Vector times

IATPARAM(18)	Integer	-	OUTPUT	Last performed preconditioner type 1: None , 2 : Jacobi , 3 : SOR , 4 : ILU(0)_Diagonal, 5:ILU(0), 6:ILUT
IATPARAM(19)	Integer	-	OUTPUT	Maximum number of fill-in's in each row(for ILUT preconditioner)
IATPARAM(20)	Integer	-	OUTPUT	Last performed solver type 1:Xabclib_GMRES, 2:Xabclib_BICGSTAB

(3) Using parameters on RATPARAM

Number	Type	Initial Value	IO	Description
RATPARAM(14)	Double	-	OUTPUT	Residual norm
RATPARAM(15)	Double	-	OUTPUT	Set-up time
RATPARAM(16)	Double	-	OUTPUT	Preconditioner time
RATPARAM(17)	Double	-	OUTPUT	Solver time
RATPARAM(18)	Double	-	OUTPUT	Total time
RATPARAM(19)	Double	-	OUTPUT	Last Performed preconditioner parameter

(4) Error Code

Value	Description
0	Normal return.
-100	“=” in POLICY FILE is illegal.
-200	The value of IATPARAM(9) is illegal
-300	“POLICY” in POLICY FILE is illegal
-310	“PRECONDITIONER” in POLICY FILE is illegal
-320	“SOLVER” in POLICY FILE is illegal
-400	The value of “MAXMEMORY” in POLICY FILE is greater than free size of memory
-500	Failing to allocate work area
>0	Error code from Xabclib_GMRES/ Xabclib_BICGSTAB. For more detail, refer 3.3.4 and 3.4.4.

3.7.5 Argument Details and Error Code of OpenATI_EIGENSOLVE

CALL OpenATI_EIGENSOLVE(N,NNZ,IRP,ICOL,VAL,IORDER, NEV,EV,EVEC, IATPARAM,RATPARAM,INFO)

(1) Argument Details

Argument	Type	IO	Description
N	Integer	INPUT	The number of dimension for the matrix. (N≧1)
NNZ	Integer	INPUT	The number of non-zero elements for the upper triangle part.
IRP(N+1)	Integer	INPUT	Pointes to diagonal elements on each row. Note: Satisfy IRP(1)=1, IRP(N+1)=NNZ+1.
ICOL(NNZ)	Integer	INPUT	The row indexes for non-zero elements on the upper triangle part.
VAL(NNZ)	Double	INPUT	The values for non-zero elements on the upper triangle part.
NEV	Integer	INPUT	The number of eigenvalues you need.
EV(NEV)	Double	OUTPUT	The eigenvalues. The k-th eigenvalue is set to EV(k).
EVEC (N,NEV)	Double	OUTPUT	The eigenvectors. The k-the eigenvector corresponding to the eigenvalue EV(k) is set to the k-th column.
IATPARAM (50)	Integer	INPUT/ OUTPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT/ OUTPUT	Array of double precision parameters for OpenATLib and Xabclib.
INFO	Integer	OUTPUT	Error Code

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(14)	Integer	0	INPUT	Access to meminfo (for Linux system) (1:done, 0:not)
IATPARAM(15)	Integer	-	OUTPUT	Number of retried solver
IATPARAM(16)	Integer	-	OUTPUT	Total restart of solver

IATPARAM(17)	Integer	-	OUTPUT	Total Matrix-Vector times

(3) Using parameters on RATPARAM

Number	Type	Initial Value	IO	Description
RATPARAM(15)	Double	-	OUTPUT	Residual norm
RATPARAM(15)	Double	-	OUTPUT	Set-up time
RATPARAM(17)	Double	-	OUTPUT	Solver time
RATPARAM(18)	Double	-	OUTPUT	Total time

(4) Error Code

Value	Description
0	Normal return.
-100	“=” in POLICY FILE is illegal.
-200	The value of IATPARAM(7) or IATPARAM(9) is illegal
-300	“POLICY” in POLICY FILE is illegal
-310	“PRECONDITIONER” in POLICY FILE is illegal
-320	“SOLVER” in POLICY FILE is illegal
-400	The value of “MAXMEMORY” in POLICY FILE is greater than free size of memory
-500	Failing to allocate work area
>0	Error code from Xabclib_LANCZOS/Xabclib_Arnoldi. For more detail, refer 3.1.4 and 3.2.4.

3.7.6 Usage Example

(1)OPENATI_LINEARSOLVE

An example of policy file

```
POLICY          =      ACCURACY
RESIDUAL        =      1.0D-10
CPU             =      16
PRECONDITIONER =      ILU0
SOLVER =          XABCLIB_GMRES
MAXMEMORY =      1.0
MAXTIME   =      500.0
```

Before running, put policy input file named “OPENATI_POLICY_INPUT.#” (#: thread number).

When OpenATI_LINEARSOLVE running is complete, computation result and input parameters are reported in “OPENATI_POLICY_REPORT.#” (#: thread number).

An example of “OPENATI_POLICY_REPORT.#” as follow.

```
****************************************************
*****  OpenATI  LINEAR SOLVER POLICY REPORT    *****
*****                   2010.0114   11:30     *****      ←report date / time
****************************************************

 [Environment variables]                                  ↓input parameters
    OPENATI_DEBUG  =
    OPENATI_POLICY = ./input_policy.dat
 [Policy Definitions]
   POLICY         = ACCURACY
   SMPs           =           16
   SOLVER         = XABCLIB_GMRES
   PRECONDITIONER = ILU0
   REQUIREMENT WORKING MEMORY =    1.0000000000000
     <<< Upper Bound 16GBYTE >>>
   REQUIREMENT RESIDUAL      =   1.000000000000000E-008
   REQUIREMENT MAX. TIME     =    500.000000000000

   MAX. SUBSPACE SIZE   =       14214
   RUNTIME MEMORY USE   =      3.24 [GBYTE]

   KRYLOV SUBSPACE EXPAND AT =  1   ,MATVEC AT =  1
   Initial Gram-Schmidt Strategy = BCGS

 ====== OPENATI_LINEARSOLVE SUCCESSFULY ENDED ======       ↓successfully exit

 [OPENATI_LINEARSOLVE RESULT]                              ↓result report
   MATRIX DATA : N=        14214  NNZ=       259688
   FASTEST MATVEC NO.  =          11                       ←fastest OpenATI_DURMV case
   FINAL KRYLOV SUBSPACE SIZE =         42                 ←Msize for convergence
   FINAL Gram-Schmidt Strategy = DGKS
   2-Norm of RHS =   25.2388589282479                      ←initial norm of RHS
   NUMBER OF RETRYED GMRES =          6                    ←retried iterations
   TOTAL RESTARTS of GMRES =        197
   RESIDUAL NORM           =  3.005885687924543E-010
   SET-UP TIME             =  1.126790046691895E-002  [SEC]
   SOLVER TIME             =   1.32032704353333       [SEC]

   TOTAL  TIME             =   1.33159494400024       [SEC]
```

(2)OPENATI_EIGENSOLVE

An example of policy file

```
POLICY          =      TIME
RESIDUAL        =      1.0D-8
CPU             =      16
SOLVER =        XABCLIB_LANCZOS
MAXMEMORY =      16.0
MAXTIME   =      600.0
```

Before running, put policy input file named “OPENATI_POLICY_INPUT.#” (#: thread number).

When OpenATI_EIGENSOLVE running is complete, computation result and input parameters are reported in “OPENATI_POLICY_REPORT.#” (#: thread number).

An example of “OPENATI_POLICY_REPORT.#” as follow.

```
****************************************************
*****  OpenATI  EIGEN  SOLVER POLICY REPORT    *****
*****                    2011.1129   14:53    *****
****************************************************

 [Environment variables]
    OPENATI_DEBUG  =           0
    OPENATI_POLICY = OPENATI_POLICY_INPUT.0
 [Policy Definitions]
   POLICY         = TIME
   SMPs           =          16
   SOLVER         = XABCLIB_LANCZOS
   REQUIREMENT WORKING MEMORY =   16.0000000000000
     <<< Upper Bound 16GBYTE >>>
   REQUIREMENT RESIDUAL       =   1.000000000000000E-008
   REQUIREMENT MAX. TIME      =    600.000000000000

   MAX. SUBSPACE SIZE   =       12326
   RUNTIME MEMORY USE   =      3.65 [GBYTE]

   KRYLOV SUBSPACE EXPAND AT =  1   ,MATVEC AT =  3
   Initial Gram-Schmidt Strategy = BCGS

 ====== OPENATI_EIGENSOLVE SUCCESSFULY ENDED ======

 [OPENATI_EIGENSOLVE RESULT]
   MATRIX DATA : N=        12328  NNZ=       177578
   FASTEST MATVEC NO. =           13
   FINAL KRYLOV SUBSPACE SIZE =          30
   FINAL Gram-Schmidt Strategy = BCGS
   NUMBER OF RETRYED LANCZOS=           1
   TOTAL RESTARTS of LANCZOS=          21
   SET-UP TIME             =  5.362033843994141E-004  [SEC]
   SOLVER TIME             =  0.654937982559204       [SEC]

  TOTAL  TIME             =  0.655474185943604       [SEC]
```

If you want to use these Meta-Solvers for thread-safe, refer to sample code in Appendix.A

4. Xabclib : A Numerical Library with Auto-tuning Facility on OpenATLib

4.1 Xabclib_LANCZOS

4.1.1 Overview of the function

Xabclib_LANCZOS can compute several eigenvalues from the absolutely largest value for large-scale symmetric matrices in the standard eigenproblem.

4.1.2 Target problem formularization and data format

(1) Target problem

The target problem is the standard eigenproblem $A\ v = \lambda\ v$ for computing eigenvalues and eigenvectors on large-scale sparse matrices, where A is a large-scale sparse matrix, λ is an eigenvalue, and v is an eigenvector.

(2) Input data format

The data format for input symmetric sparse matrix A is Compressed Row Storage (CRS) shown in Fig.4-1. Please note that the format is dedicated for symmetric matrices, hence we do not need lower elements.

$$\begin{pmatrix} 1 & 2 & 0 & 3 & 0 \\ & 4 & 0 & 0 & 5 \\ & & 6 & 0 & 0 \\ & & & 7 & 8 \\ & & & & 9 \end{pmatrix} =$$

Pointers to diagonal elements.	1	4	6	7	9	10			
Row indexes for non-zero elements.	1	2	4	2	5	3	4	5	5
Values for non-zero elements.	1	2	3	4	5	6	7	8	9

Fig. 4-1 Compressed Row Storage (CRS) for Symmetric Matrices.

4.1.3 The Lanczos Method

The Lanczos method using this library is shown in Fig. 4-2. The algorithm is based on the algorithm referred by [3].

1. $Start\ with\ v_0 \equiv r, \beta_0 := \|r\|_2, lock = 0$
2. $For\ IR = 1,2,\cdots, maxrestart\ Do:$
3. $\quad For\ j = lock+1,\cdots, m\ Do:$
4. $\quad\quad Compute\ v_j := r/\beta_0$
5. $\quad\quad r := Av_j$
6. $\quad\quad \alpha_{\mathrm{j}} := (r, v_j)$
7. $\quad\quad if\ \ (j=1)\ then\quad r := r - \alpha_{\mathrm{j}} v_j$
8. $\quad\quad if\ \ (j \neq 1)\ then\quad r := r - \alpha_{\mathrm{j}} v_j - \beta_{\mathrm{j\text{-}1}} v_{j-1}$
9. $\quad\quad r \perp V_{j-1}\ \ by\ modified\ Gram\text{-}Schmidt$
10. $\quad\quad \beta_j := \|r\|_2$
11. $\quad EndDo$
12. $\quad Eigen\ solve\ T = S\Theta S^T\ \ ,\ T = \begin{bmatrix} \alpha_{\mathrm{lock+1}} & & & & \\ \beta_{lock+1} & \alpha_{\mathrm{lock+2}} & & & \\ & \cdots & \cdots & & \\ & & \cdots & \cdots & \\ & & & \beta_{m-1} & \alpha_m \end{bmatrix}$
13. $\quad k\text{-}th\ residual\ estimate\ with\ |\beta_m S_{m,k}| / |\Theta_k|\ for\ k = lock+1, NEV$
14. $\quad creat\ Ritz\ vectors\ \mathrm{Q_k} = V_m S_k$
15. $\quad count-up\ \ 'new\ locked'\ Ritz\ pair$
16. $\quad if\ \ (lock + 'new\ lock'\ \ \geq NEV)\ \ goto\ exit$
17. $\quad create\ new\ starting\ Shur\ vector\ r = V_m S_{'new\ locked'+1}$
18. $\quad deflation\ V_{lock+L} = Q_{lock+L}\ \ for\ L = 1,'new\ lock', then\ lock = 'new\ lock'$
19. $EndDo$

Fig. 4-2 The Lanczos Method.

4.1.4 Argument Details and Error Code

(1) Argument Details

Argument	Type	IO	Description
N	Integer	INPUT	The number of dimension for the matrix. (N>=1)
NNZ	Integer	INPUT	The number of non-zero elements for the upper triangle part.
IRP(N+1)	Integer	INPUT	Pointes to diagonal elements on each row. Note: Satisfy IRP(1)=1, IRP(N+1)=NNZ+1.
ICOL(NNZ)	Integer	INPUT	The row indexes for non-zero elements on the upper triangle part.
VAL(NNZ)	Double	INPUT	The values for non-zero elements on the upper triangle part.
NEV	Integer	INPUT	The number of eigenvalues you need. The execution time increases according to the NEV. If NEV>100, the execution time will be enormous, hence it may not solve in practical time.
EV(NEV)	Double	OUTPUT	The eigenvalues. The k-th eigenvalue is set to EV(k).
EVEC (LDE,NEV)	Double	OUTPUT	The eigenvectors. The k-the eigenvector corresponding to the eigenvalue EV(k) is set to the k-th column.
LDE	Integer	INPUT	The leading dimension of EVEC array (LDE>=N)
IATPARAM (50)	Integer	INPUT/ OUTPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARA M(50)	Double	INPUT/ OUTPUT	Array of double precision parameters for OpenATLib and Xabclib.
WK (LWK)	Double	WORK	Workspace.
LWK	Integer	INPUT	The size of the double precision workspace WK. Satisfy LWK >= (1+MSIZE)*N + 2*MSIZE*MSIZE + 7*MSIZE + 5*NEV +2. (MSIZE= IATPARAM(27))
IWK (LIWK)	Integer	WORK	Workspace.

LIWK	Integer	INPUT	The size of the integer workspace IWK. Satisfy LIWK >= 5*MSIZE + 3. (MSIZE= IATPARAM(27))
INFO	Integer	OUTPUT	Error code.

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(3)	Integer	OMP_GET_MAX_THREADS()	INPUT	Number of THREADS.
IATPARAM(4)	Integer	1	INPUT	Flag of Krylov subspace expand by MM-ratio.
IATPARAM(5)	Integer	5	INPUT	incremental value for Krylov subspace when MM-ratio is less than threshold(RATPARAM(4))
IATPARAM(7)	Integer	3	INPUT	OpenATI_DSRMV auto-tuned On/Off 0: Perform SpMxV specified by IATPARAM(8). 1: reserved 2: Perform SpMxV to judge the best methods between two methods, except for reduction parallel implementation. 3: Perform SpMxV to judge the best method among three methods. Note that workspace according to the number of threads is needed.
IATPARAM(8)	Integer	12	INPUT /OUTPUT	If IATPARAM(7)=0, then set the number of implementations. If IATPARAM(7)=2 or 3, the best number of implementations returns. 11: Row Decomposition Method.

				12: Normalized NZ Method. 13: Normalized NZ Method, with vector reduction parallelization.
IATPARAM(12)	Integer	2	INPUT	0 : Classical Gram-Schmidt 1 : DGKS 2 : Modified Gram-Schmidt 3 : Blocked Gram-Schmidt
IATPARAM(13)	Integer	-	OUTPUT	Iterative refinement of DGKS 0 : no Iterative refinement 1 : Iterative refinement
IATPARAM(22)	Integer	-1	INPUT	Maximum number of restart iterations.
IATPARAM(23)	Integer	-	OUTPUT	Final number of restart iterations.
IATPARAM(27)	Integer	20	INPUT	Max size of Krylov subspace.
IATPARAM(28)	Integer	2	INPUT	Start size of Krylov subspace at subspace expand AT-on. See IATPARAM(4) If IATPARAM(28) less than NEV ,then start subspace size 'NEV' (overwritten).
IATPARAM(29)	Integer	-	OUTPUT	Final size of Krylov subspace.
IATPARAM(30)	Integer	1	INPUT	Eigenvalue order option. 1: largest eigenvalue 2: largest magnitude
IATPARAM(31)	Integer	-	OUTPUT	Total Matrix-Vector times
IATPARAM(32)	Integer	0	INPUT	When stagnation of relative residual occurs, solver is stopped. (0: Off, 1:On)

(3) Using parameters on RATPARAM

Number	Type	Initial	IO	Description

		Value		
RATPARAM(4)	Double	100.0	INPUT	Threshold value for MM ratio.
RATPARAM(22)	Double	-1	INPUT	Max. elapsed time.
RATPARAM(23)	Double	1.0E-08	INPUT	Convergence criterion.
RATPARAM(29)	Double	-	OUTPUT	2-norm of max. residual.
RATPARAM(30)	Double	-	OUTPUT	floating operations (x10^9 operations).
RATPARAM(32)	Double	-	OUTPUT	total solve time.

(4) Error Code

Value	Description
0	Normal return.
Less than 0	If -i returns, the value of i-th argument is illegal.
100	Computation was stopped by breakdown for zero vector division.
200	Computation was stopped by abnormal computation of eigenvalues in part of tridiagonal matrix computation.
300	Computation was stopped by exceeding the maximum number of restart.
400	Computation was stopped by exceeding the execution time tolerance.
500	Computation was stopped by failing to allocate memory in case of IATPARAM(8)=12,13.

4.2 Xabclib_ARNOLDI

4.2.1 Overview of the function

Xabclib_ARNOLDI can compute several eigenvalues for large-scale unsymmetric matrices in the standard eigenproblem.

4.2.2 Target problem formularization and data format

(1) Target problem

The target problem is the standard eigenproblem $A\ v = \lambda\ v$ for computing eigenvalues and eigenvectors on large-scale sparse matrices, where A is a large-scale sparse matrix, λ is an eigenvalue, and v is an eigenvector.

(2) Input data format

The data format for input symmetric sparse matrix A is Compressed Row Storage (CRS) shown in Fig.4-3. Please note that the format is dedicated for symmetric matrices, hence we do not need lower elements.

$$\begin{pmatrix} 1 & 2 & 0 & 3 & 0 \\ 4 & 5 & 0 & 0 & 6 \\ 0 & 0 & 7 & 0 & 0 \\ 8 & 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 0 & 10 \end{pmatrix} =$$

Pointers to first row elements.

1	4	7	8	10	11

Row indexes for non-zero elements.

1	2	4	1	2	5	3	1	4	5

Values for non-zero elements.

1	2	3	4	5	6	7	8	9	10

Fig. 4-3 Compressed Row Storage (CRS) for Unsymmetric Matrices.

4.2.3 The Arnoldi Method

The Arnoldi method using this library is shown in Fig. 4-4. The algorithm is based on the algorithm referred by [9].

$\textbf{\textit{Explicitly re - start Arnoldi method with deflated Schur - vector}}$

(step 1) random vector u_0

(step 2) $l = 0$

(step 3) $Q_l = 0,\ v_l = u_0$

(step 4) Arnoldi decompose

$$AQ_m = Q_m H_m + \beta_{m+1} q_{m+1} \mathbf{e}_m^T$$

(step 5) solve Hessenberg system $H_m S^{(m)} = \theta^{(m)} S^{(m)}$

(step 6) check convergence $\left|\beta_{m+1} S_{m,i}{}^{(m)} \theta_i^{(m)}\right| \le eps$

(step 7) (deflation)

if $(\theta_i^{(m)}, S_i^{(m)})$ is converged, then

$y_k = Q_m S_i^{(m)}$

$v_k \leftarrow y_k \perp Q_l$

$H_{i,k} = v_i^* A v_k \quad for\ k = 1,2,\ldots,l$

$l = l + 1$

$Q_l = v_k$

end if

(step 8) if one more eigenpair desired, then

$u_0 = Q_{m-l} S_j{}^{(m)}$, a sampling '$j$'

goto (step 4)

end if

Fig. 4-4 The Arnoldi Method.

4.2.4 Argument Details and Error Code

(1) Argument Details

Argument	Type	IO	Description
N	Integer	INPUT	The number of dimension for the matrix. (N≧1)
NNZ	Integer	INPUT	The number of non-zero elements for the matrix.
IRP(N+1)	Integer	INPUT	Pointes to first position on each row for the matrix. Note: Satisfy IRP(1)=1, IRP(N+1)=NNZ+1.
ICOL(NNZ)	Integer	INPUT	The row indexes for non-zero elements for the matrix.
VAL(NNZ)	Double	INPUT	The non-zero elements for the matrix.
NEV	Integer	INPUT	The number of eigenvalues you need. The execution time increases according to the NEV. If NEV>100, the execution time will be enormous, hence it may not solve in practical time.
EV(NEV)	COMPLEX*16	OUTPUT	The eigenvalues. The k-th eigenvalue is set to EV(k).
EVEC (LDE,NEV)	COMPLEX*16	OUTPUT	The eigenvectors. The k-th eigenvector corresponding to the eigenvalue EV(k) is set to the k-th column.
LDE	Integer	INPUT	The leading dimension of EVEC array (LDE>=N)
IATPARAM (50)	Integer	INPUT/ OUTPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT/ OUTPUT	Array of double precision parameters for OpenATLib and Xabclib.
WORK (LWORK)	Double	WORK	Workspace.
LWORK	Integer	INPUT	The size of the double precision workspace WORK. Satisfy LWORK >= (5+MSIZE)*N + 5*MSIZE*MSIZE + 9*MSIZE + 6*NEV. (MSIZE= IATPARAM(27))
IWORK (LIWORK)	Integer	WORK	Workspace.

LIWORK	Integer	INPUT	The size of the integer workspace IWORK. Satisfy LIWORK >= MSIZE. (MSIZE= IATPARAM(27))
INFO	Integer	OUTPUT	Error code.

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(3)	Integer	OMP_GET_MAX_THREADS()	INPUT	Number of THREADS.
IATPARAM(4)	Integer	1	INPUT	Flag of Krylov subspace expand by MM-ratio.
IATPARAM(5)	Integer	5	INPUT	incremental value for Krylov subspace when MM-ratio is less than threshold(RATPARAM(4))
IATPARAM(9)	Integer	0	INPUT	OpenATI_DURMV auto-tuned On/Off 0 : Perform SpMxV specified by IATPARAM(10). 2 and 3 : Perform SpMxV to judge the best method among three implementations.
IATPARAM(10)	Integer	12	INPUT	If IATPARAM(9)=0, then set the number of implementations. If IATPARAM(9)=2 or 3, the best number of implementations returns. 11: Row Decomposition Method. 12: Normalized NZ Method. 13: Branchless Segmented Scan. 21: Original Segmented Scan.
IATPARAM(11)	Integer	128	INPUT	Columns of Segmented Scan's algorithms.

IATPARAM(12)	Integer	2	INPUT	0：Classical Gram-Schmidt 1：DGKS 2：Modified Gram-Schmidt 3：Blocked Gram-Schmidt
IATPARAM(13)	Integer	-	OUTPUT	Iterative refinement of DGKS 0：no Iterative refinement 1：Iterative refinement
IATPARAM(22)	Integer	-1	INPUT	Maximum number of restart iterations.
IATPARAM(23)	Integer	-	OUTPUT	Final number of restart iterations.
IATPARAM(27)	Integer	20	INPUT	Max size of Krylov subspace.
IATPARAM(28)	Integer	2	INPUT/ OUTPUT	Start size of Krylov subspace at subspace expand AT-on. See IATPARAM(4). If IATPARAM(28) less than NEV ,then start subspace size 'NEV' (overwritten).
IATPARAM(29)	Integer		OUTPUT	Final size of Krylov subspace.
IATPARAM(30)	Integer	1	INPUT	Eigenvalue order option. 1: largest real part eigenvalue 2：largest magnitude 3：largest imaginary part
IATPARAM(31)	Integer	0	OUTPUT	Total Matrix-Vector times.
IATPARAM(32)	Integer	0	INPUT	When stagnation of relative residual occurs, solver is stopped. (0: Off, 1:On)

(3) Using parameters on RATPARAM

Number	Type	Initial Value	IO	Description
RATPARAM(4)	Double	100.0	INPUT	Threshold value for MM ratio.
RATPARAM(22)	Double	-1	INPUT	Max. elapsed time.
RATPARAM(23)	Double	1.0E-08	INPUT	Convergence criterion.
RATPARAM(29)	Double	-	OUTPUT	2-norm of max. residual.

RATPARAM(30)	Double	-	OUTPUT	floating operations (x10^9 operations).
RATPARAM(32)	Double	-	OUTPUT	total solve time.

(4) Error Code

Value	Description
0	Normal return.
Less than 0	If -i returns, the value of i-th argument is illegal.
100	Computation was stopped by breakdown for zero vector division.
200	Computation was stopped by abnormal computation of eigenvalues in part of tridiagonal matrix computation.
300	Computation was stopped by exceeding the maximum number of restart.
400	Computation was stopped by exceeding the execution time tolerance.
500	Eigenvalue and eigenvector are illegal.
600	Computation was stopped by failing to allocate memory in case of IATPARAM(10)=12,13,21.

4.3 Xabclib_GMRES

4.3.1 Overview of the function

Xabclib_GMRES can solve large-scale unsymmetric sparse matrices in the linear equations problem.

4.3.2 Target problem and data format

(1) Target problem

The problem to be solved in the library is the linear equations problem $A\ x = b$, where A is a large-scale sparse matrix, x is a solution vector, and b is a right hand side vector.

(2) Input data format

The unsymmetric sparse matrix format is Compressed Row Storage (CRS) for unsymmetric matrices shown in Fig. 4-3.

4.3.3 Overview of the algorithm

The algorithm used in this solver is the GMRES method, which is shown in Fig. 4-5. The algorithm was presented in [4].

1. $Compute\ r_0 = b - Ax_0, \beta := \|r_0\|_2, and\ v_1 := r_0 / \beta$
2. $Define\ the\ (m+1) \times m\ matrix\ \overline{H}_m = \{h_{ij}\}_{1 \le i \le m+1, 1 \le j \le m}, Set\ \overline{H}_m = 0$
3. $For\ j = 1, 2, \cdots, m\ Do:$
4. $\quad Compute\ \omega_j := Av_j$
5. $\quad For\ i = 1, \cdots, j\ Do:$
6. $\quad\quad h_{ij} := (\omega_j, v_i)$
7. $\quad\quad \omega_j := \omega_j - h_{ij} v_i$
8. $\quad EndDo$
9. $\quad h_{j+1,j} = \|\omega_j\|_2. If\ h_{j+1,j} = 0\ Set\ m := j\ and\ go\ to\ 12$
10. $\quad v_{j+1} = \omega_j / h_{j+1,j}$
11. $EndDo$
12. $Compute\ y_m\ the\ minimizer\ of\ \|\beta e_1 - \overline{H}_m y\|_2\ and\ x_m = x_0 + V_m y_m.$

Fig. 4-5 The GMRES Method.

4.3.4 Argument Details and Error Code

(1) Argument Details

Argument	Type	IO	Description
N	Integer	INPUT	The number of dimension for the matrix. (N>=1)
NNZ	Integer	INPUT	The number of non-zero elements for the matrix.
IRP(N+1)	Integer	INPUT	Pointes to first position on each row for the matrix. Note: Satisfy IRP(1)=1, IRP(N+1)=NNZ+1.
ICOL(NNZ)	Integer	INPUT	The row indexes for non-zero elements for the matrix.
VAL(NNZ)	Double	INPUT	The non-zero elements for the matrix.
B(N)	Double	INPUT	The elements for right hand size vector *b*.
X(N)	Double	INPUT / OUTPUT	INPUT: Set the elements of initial guess for solution vector *x_0*. OUTPUT: Return the elements of solution vector *x*.
PRECOND (NPRE)	Double	INPUT / OUTPUT	INPUT: ● If IATPARAM(24)=1, then none to be set. ● If IATPARAM(24)=2, then set preconditioner kind of M already specified. OUTPUT: ● If IATPARAM(24)=1, then the preconditioner kind of M returns. ● If IATPARAM(24)=2, then no modification.

NPRE	Integer	INPUT	The size of PRECOND array.
			If IATPARAM(25) = 1, then NPRE>=0.
			If IATPARAM(25) = 2,3 or 4, then NPRE>=N.
			If IATPARAM(25) = 5, then
			NPRE>=3*NNZ/2+2*N+50
			If IATPARAM(25) = 6, then
			NPRE>=3*(2.0*IFILL+1)*N/2+3*N+50
			(IFILL=IATPARAM(26))

IATPARAM (50)	Integer	INPUT/ OUTPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT/ OUTPUT	Array of double precision parameters for OpenATLib and Xabclib.
WK (LWK)	Double	WORK	Workspace.
LWK	Integer	INPUT	The size of the workspace for double precision WK. Satisfy LWK >= (MSIZE+2)*N + (MSIZE+1)*(MSIZE+1) + (N-1)/2+1. (MSIZE= IATPARAM(27))
INFO	Integer	OUTPUT	Error code.

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(3)	Integer	OMP_GET_MAX_THREADS()	INPUT	Number of THREADS.
IATPARAM(4)	Integer	1	INPUT	Flag of Krylov subspace expand by MM-ratio.
IATPARAM(5)	Integer	5	INPUT	incremental value for Krylov subspace when MM-ratio is less than threshold(RATPARAM(4))
IATPARAM(9)	Integer	0	INPUT	OpenATI_DURMV auto-tuned On/Off 0 : Perform SpMxV specified by IATPARAM(10). 2 and 3 : Perform SpMxV to judge the best method among three implementations.
IATPARAM(10)	Integer	12	INPUT	If IATPARAM(9)=0, then set the number of implementations.

				If IATPARAM(9)=2 or 3, the best number of implementations returns. 11: Row Decomposition Method. 12: Normalized NZ Method. 13: Branchless Segmented Scan. 21: Original Segmented Scan.
IATPARAM(11)	Integer	128	INPUT	Columns of Segmented Scan's algorithms.
IATPARAM(12)	Integer	2	INPUT	0 : Classical Gram-Schmidt 1 : DGKS 2 : Modified Gram-Schmidt 3 : Blocked Gram-Schmidt
IATPARAM(13)	Integer	-	OUTPUT	Iterative refinement of DGKS 0 : no Iterative refinement 1 : Iterative refinement
IATPARAM(22)	Integer	-1	INPUT	Maximum number of restart iterations.
IATPARAM(23)	Integer	-	OUTPUT	Final number of restart iterations.
IATPARAM(24)	Integer	1	INPUT	Preconditioner operations flag. 1: not generated yet 2 : already generated
IATPARAM(25)	Integer	4	INPUT	Set preconditioner kinds. 1:None. 2:Jacobi. 3:SSOR. 4:ILU(0)_Diagonal. 5:ILU(0) 6:ILUT
IATPARAM(26)	Integer	5	INPUT	Maximum number of fill-in's in each row (for ILUT).
IATPARAM(27)	Integer	20	INPUT	Max size of Krylov subspace.
IATPARAM(28)	Integer	2	INPUT	Start size of Krylov subspace at subspace expand AT-on. See IATPARAM(4)

IATPARAM(29)	Integer	-	OUTPUT	Final size of Krylov subspace.
IATPARAM(31)	Integer	-	OUTPUT	Total Matrix-Vector times.
IATPARAM(32)	Integer	0	INPUT	When stagnation of relative residual occurs, solver is stopped. (0: Off, 1:On)
IATPARAM(33)	Integer	0	INPUT	Minimum running iteration.

(3) Using parameters on RATPARAM

Number	Type	Initial Value	IO	Description
RATPARAM(4)	Double	100.0	INPUT	Threshold value for MM ratio.
RATPARAM(22)	Double	-1	INPUT	Max. elapsed time.
RATPARAM(23)	Double	1.0E-08	INPUT	Convergence criterion.

RATPARAM(25)	Double	1.0E-08	INPUT	If IATPARAM(25)=3, then
				Set parameter ω for SSOR preconditioner. (1<=ω<2)
				If IATPARAM(25)=4 or 5, then
				Set threathold value to judge breakdown when computing ILU(0) preconditioner.
				If IATPARAM(25)=6, then
				Set value of dropping criterion when computing ILU(0) preconditioner.

RATPARAM(28)	Double	-	OUTPUT	2-norm of RHS.
RATPARAM(29)	Double	-	OUTPUT	2-norm of max. residual.
RATPARAM(30)	Double	-	OUTPUT	Floating operations (x10^9 operations).
RATPARAM(31)	Double	-	OUTPUT	Preconditioner time.
RATPARAM(32)	Double	-	OUTPUT	Total solve time.
RATPARAM(33)	Double	0.0	INPUT	Minimum running time.

(4) Error Code

Value	Description
0	Normal return.

Less than 0	If -i returns, the value of i-th argument is illegal.
100	Computation was stopped by failing to make preconditioner.
200	Computation was stopped by breakdown.
300	Computation was stopped by that the value of OpenATI_DAFRT is illegal.
400	Computation was stopped by exceeding the execution time tolerance.
500	Computation was stopped by exceeding the maximum number of restart.
600	Computation was stopped by failing to allocate memory in case of IATPARAM(10)=12,13,21.
700	Computation was stopped by the value of LUINF exceeds Integer max in case of IATPARAM(10)=21.
1000	Computation was stopped by stagnation of relative residual. This error code is output only when IATPARAM(32)=1.

4.4 Xabclib_BICGSTAB

4.4.1 Overview of the function

Xabclib_BICGSTAB can solve large-scale unsymmetric sparse matrices in the linear equations problem.

4.4.2 Target problem and data format

(1) Target problem

The problem to be solved in the library is the linear equations problem $A\,x = b$, where A is a large-scale sparse matrix, x is a solution vector, and b is a right hand side vector.

(2) Input data format

The unsymmetric sparse matrix format is Compressed Row Storage (CRS) for unsymmetric matrices shown in Fig. 4-3.

4.4.3 Overview of the algorithm

The algorithm used in this solver is the BiCGStab method, which is shown in Fig. 4-6. The algorithm was presented in [10].

BiCGStab with right preconditioner by Dr. Itoh

(1) $x_0 = \text{initial guess}\ ,\ r = b - Ax_0, \underline{r_0^* = M^{-1}r, \text{solve } M\hat{r} = r, \rho_0 = \langle r_0^*, \hat{r} \rangle},$

$\beta = 0,\ p = v = 0$

(2) iter $k = 0,1,2,\ldots,\text{max\_iter}$

(3) $p = \hat{r} + \beta z$

(4) $\underline{p = Ap}$

(5) $\underline{\text{solve } Mv = p}$ $\Rightarrow v = M^{-1}Ap$

(6) $\gamma = \langle r_0^*, v \rangle$

(7) $\alpha = \rho_0 / \gamma$

(8) $\underline{s = r - \alpha p}$; and $\underline{\hat{s} = \hat{r} - \alpha v}$;

check conv.? if $\|\hat{s}\|$ *small enough* then $x = x + \alpha p; exit$

(9) $t = A\hat{s}$

(10) $\zeta = \langle t^*, s \rangle / \langle t^*, t \rangle$

(11) $x = x + \alpha p + \zeta \hat{s}$

(12) $r = s - \zeta t$

(13) check conv.? if $\|r\|$ *small enough exit*

(14) $\underline{\text{solve } M\hat{r} = r}$

(15) $z = p - \zeta v$

(16) $\rho_N = \langle r_0^*, \hat{r} \rangle$

(17) $\beta = \alpha / \zeta \cdot \rho_N / \rho_0$

(18) $\rho_0 = \rho_N$

(19) end iter

Fig. 4-6 The BiCGStab Method.

4.4.4 Argument Details and Error Code

(1) Argument Details

Argument	Type	IO	Description
N	Integer	INPUT	The number of dimension for the matrix. (N>=1)
NNZ	Integer	INPUT	The number of non-zero elements for the matrix.
IRP(N+1)	Integer	INPUT	Pointes to first position on each row for the matrix. Note: Satisfy IRP(1)=1, IRP(N+1)=NNZ+1.
ICOL(NNZ)	Integer	INPUT	The row indexes for non-zero elements for the matrix.
VAL(NNZ)	Double	INPUT	The non-zero elements for the matrix.
B(N)	Double	INPUT	The elements for right hand size vector *b*.
X(N)	Double	INPUT / OUTPUT	INPUT: Set the elements of initial guess for solution vector *x_0*. OUTPUT: Return the elements of solution vector *x*.
PRECOND (NPRE)	Double	INPUT / OUTPUT	INPUT: ● If IATPARAM(24)=1, then none to be set. ● If IATPARAM(24)=2, then set preconditioner kind of M already specified. OUTPUT: ● If IATPARAM(24)=1, then the preconditioner kind of M returns. ● If IATPARAM(24)=2, then no modification.

NPRE	Integer	INPUT	The size of PRECOND array.
			If IATPARAM(25) = 1, then NPRE>=0.
			If IATPARAM(25) = 2,3 or 4, then NPRE>=N.
			If IATPARAM(25) = 5, then
			NPRE>=3*NNZ/2+2*N+50
			If IATPARAM(25) = 6, then
			NPRE>=3*(2.0*IFILL+1)*N/2+3*N+50
			(IFILL=IATPARAM(26))

IATPARAM	Integer	INPUT/	Array of integer parameters for OpenATLib and

(50)		OUTPUT	Xabclib.
RATPARAM(50)	Double	INPUT/ OUTPUT	Array of double precision parameters for OpenATLib and Xabclib.
WORK (LWORK)	Double	WORK	Workspace.
LWORK	Integer	INPUT	The size of the workspace for double precision WORK. Satisfy LWORK >= 9*N + (N-1)/2+1.
INFO	Integer	OUTPUT	Error code.

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(3)	Integer	OMP_GET_MAX_THREADS()	INPUT	Number of THREADS.
IATPARAM(4)	Integer	1	INPUT	Flag of Krylov subspace expand by MM-ratio.
IATPARAM(9)	Integer	0	INPUT	OpenATI_DURMV auto-tuned On/Off 0 : Perform SpMxV specified by IATPARAM(10). 2 and 3 : Perform SpMxV to judge the best method among three implementations.
IATPARAM(10)	Integer	12	INPUT	If IATPARAM(9)=0, then set the number of implementations. If IATPARAM(9)=2 or 3, the best number of implementations returns. 11: Row Decomposition Method.

				12: Normalized NZ Method. 13: Branchless Segmented Scan. 21: Original Segmented Scan.
IATPARAM(11)	Integer	128	INPUT	Columns of Segmented Scan's algorithms.
IATPARAM(12)	Integer	2	INPUT	0 : Classical Gram-Schmidt 1 : DGKS 2 : Modified Gram-Schmidt 3 : Blocked Gram-Schmidt
IATPARAM(13)	Integer	-	OUTPUT	Iterative refinement of DGKS 0 : no Iterative refinement 1 : Iterative refinement
IATPARAM(22)	Integer	-1	INPUT	Maximum number of restart iterations..
IATPARAM(23)	Integer	-	OUTPUT	Final number of restart iterations.
IATPARAM(24)	Integer	1	INPUT	Preconditioner operations flag. 1: not generated yet 2 : already generated
IATPARAM(25)	Integer	4	INPUT	Set preconditioner kinds. 1:None. 2:Jacobi. 3:SSOR. 4:ILU(0)_Diagonal. 5:ILU(0) 6:ILUT
IATPARAM(26)	Integer	5	INPUT	Maximum number of fill-in's in each row (for ILUT).
IATPARAM(31)	Integer	-	OUTPUT	Total Matrix-Vector times.
IATPARAM(32)	Integer	0	INPUT	When stagnation of relative residual occurs, solver is stopped. (0: Off, 1:On)
IATPARAM(33)	Integer	0	INPUT	Minimum running iteration.

(3) Using parameters on RATPARAM

Number	Type	Initial Value	IO	Description
RATPARAM(4)	Double	100.0	INPUT	Threshold value for MM ratio.
RATPARAM(22)	Double	-1	INPUT	Max. elapsed time.
RATPARAM(23)	Double	1.0E-08	INPUT	Convergence criterion.

RATPARAM(25)	Double	1.0E-08	INPUT	If IATPARAM(25)=3, then
				Set parameter ω for SSOR preconditioner. (1<=ω<2)
				If IATPARAM(25)=4 or 5, then
				Set threathold value to judge breakdown when computing ILU(0) preconditioner.
				If IATPARAM(25)=6, then
				Set value of dropping criterion when computing ILU(0) preconditioner.

RATPARAM(28)	Double	-	OUTPUT	2-norm of RHS.
RATPARAM(29)	Double	-	OUTPUT	2-norm of max. residual.
RATPARAM(30)	Double	-	OUTPUT	Floating operations (x10^9 operations).
RATPARAM(31)	Double	-	OUTPUT	Preconditioner time.
RATPARAM(32)	Double	-	OUTPUT	Total solve time.
RATPARAM(33)	Double	0.0	INPUT	Minimum running time.

(4) Error Code

Value	Description
0	Normal return.
Less than 0	If -i returns, the value of i-th argument is illegal.
100	Computation was stopped by failing to make preconditioner.
200	Computation was stopped by breakdown.
400	Computation was stopped by exceeding the execution time tolerance.
500	Computation was stopped by exceeding the maximum number of restart.
600	Computation was stopped by failing to allocate memory in case of IATPARAM(10)=12,13,21.

700	Computation was stopped by the value of LUINF exceeds Integer max in case of IATPARAM(10)=21.
1000	Computation was stopped by stagnation of relative residual. This error code is output only when IATPARAM(32)=1.

4.5 Xabclib_CG

4.5.1 Overview of the function

Xabclib_CG can solve large-scale symmetric sparse matrices in the linear equations problem.

4.5.2 Target problem and data format

(1) Target problem

The problem to be solved in the library is the linear equations problem $A\ x = b$, where A is a large-scale sparse matrix, x is a solution vector, and b is a right hand side vector.

(2) Input data format

The symmetric sparse matrix format is Compressed Row Storage (CRS) for symmetric matrices shown in Fig. 4-2.

4.5.3 Overview of the algorithm

The algorithm used in this solver is the CG method.

4.5.4 Argument Details and Error Code

(1) Argument Details

Argument	Type	IO	Description
N	Integer	INPUT	The number of dimension for the matrix. (N>=1)
NNZ	Integer	INPUT	The number of non-zero elements for the matrix.
IRP(N+1)	Integer	INPUT	Pointes to first position on each row for the matrix. Note: Satisfy IRP(1)=1, IRP(N+1)=NNZ+1.
ICOL(NNZ)	Integer	INPUT	The row indexes for non-zero elements for the matrix.
VAL(NNZ)	Double	INPUT	The non-zero elements for the matrix.
B(N)	Double	INPUT	The elements for right hand size vector *b*.
X(N)	Double	INPUT / OUTPUT	INPUT: Set the elements of initial guess for solution vector *x_0*. OUTPUT: Return the elements of solution vector *x*.
PRECOND (NPRE)	Double	INPUT / OUTPUT	INPUT: ● If IATPARAM(24)=1, then none to be set. ● If IATPARAM(24)=2, then set preconditioner kind of M already specified. OUTPUT: ● If IATPARAM(24)=1, then the preconditioner kind of M returns. ● If IATPARAM(24)=2, then no modification.
NPRE	Integer	INPUT	The size of PRECOND array. If IATPARAM(25) = 1, then NPRE>=0. If IATPARAM(25) = 2,3 or 4, then NPRE>=N.
IATPARAM (50)	Integer	INPUT/ OUTPUT	Array of integer parameters for OpenATLib and Xabclib.
RATPARAM(50)	Double	INPUT/ OUTPUT	Array of double precision parameters for OpenATLib and Xabclib.

WORK (LWORK)	Double	WORK	Workspace.
LWORK	Integer	INPUT	The size of the workspace for double precision WORK. Satisfy LWORK >= 4*N.
INFO	Integer	OUTPUT	Error code.

(2) Using parameters on IATPARAM

Number	Type	Initial Value	IO	Description
IATPARAM(3)	Integer	OMP_GET_MAX_THREADS()	INPUT	Number of THREADS.
IATPARAM(4)	Integer	1	INPUT	Flag of Krylov subspace expand by MM-ratio.
IATPARAM(7)	Integer	3	INPUT	OpenATI_DSRMV auto-tuned On/Off 0: Perform SpMxV specified by IATPARAM(8). 1: reserved 2: Perform SpMxV to judge the best methods between two methods, except for reduction parallel implementation. 3: Perform SpMxV to judge the best method among three methods. Note that workspace according to the number of threads is needed.
IATPARAM(8)	Integer	12	INPUT /OUTP	If IATPARAM(7)=0, then set the number of implementations.

			UT	If IATPARAM(7)=2 or 3, the best number of implementations returns. 11: Row Decomposition Method. 12: Normalized NZ Method. 13: Normalized NZ Method, with vector reduction parallelization.
IATPARAM(12)	Integer	2	INPUT	0 : Classical Gram-Schmidt 1 : DGKS 2 : Modified Gram-Schmidt 3 : Blocked Gram-Schmidt
IATPARAM(13)	Integer	-	OUTPUT	Iterative refinement of DGKS 0 : no Iterative refinement 1 : Iterative refinement
IATPARAM(22)	Integer	-1	INPUT	Maximum number of restart iterations..
IATPARAM(23)	Integer	-	OUTPUT	Final number of restart iterations.
IATPARAM(24)	Integer	1	INPUT	Preconditioner operations flag. 1: not generated yet 2 : already generated
IATPARAM(25)	Integer	4	INPUT	Set preconditioner kinds. 1:None. 2:Jacobi. 3:Jacobi-iterative. 4:IC(0)_Diagonal.
IATPARAM(26)	Integer	5	INPUT	number of Jacobi-iteration in a CG iteration (for Jacobi-iterative).
IATPARAM(31)	Integer	-	OUTPUT	Total Matrix-Vector times.

(3) Using parameters on RATPARAM

Number	Type	Initial Value	IO	Description
RATPARAM(4)	Double	100.0	INPUT	Threshold value for MM ratio.

RATPARAM(22)	Double	-1	INPUT	Max. elapsed time.
RATPARAM(23)	Double	1.0E-08	INPUT	Convergence criterion.
RATPARAM(28)	Double	-	OUTPUT	2-norm of RHS.
RATPARAM(29)	Double	-	OUTPUT	2-norm of max. residual.
RATPARAM(30)	Double	-	OUTPUT	Floating operations (x10^9 operations).
RATPARAM(31)	Double	-	OUTPUT	Preconditioner time.
RATPARAM(32)	Double	-	OUTPUT	Total solve time.

(4) Error Code

Value	Description
0	Normal return.
Less than 0	If -i returns, the value of i-th argument is illegal.
100	Computation was stopped by failing to make preconditioner.
200	Computation was stopped by breakdown.
400	Computation was stopped by exceeding the execution time tolerance.
500	Computation was stopped by exceeding the maximum number of restart.
600	Computation was stopped by failing to allocate memory in case of IATPARAM(10)=12,13.

5. References

[1] T. Sakurai, K. Naono, M. Egi, M. Igai, and H. Kidachi: Proposal on Runtime Parameter Auto Tuning Approach for Restarted Lanczos Method, IPSJ SIG Notes, 2007-HPC-111, pp.173-178, (2007)(in Japanese).

[2] M. Kudo, H. Kuroda, T. Katagiri, and Y. Kanada: The Effect of Optimal Algorithm Selection of Parallel Sparse Matrix-Vector Multiplication, IPSJ SIG Notes, 2002-ARC-147, pp.151-156 (2002)(in Japanese).

[3]V. Hernandez, J. E. Roman, and A. Tomas: Evaluation of Several Variants of Explicitly Restarted Lanczos Eigensolvers and Their Parallel Implementations, High Performance Computing for Computational Science - VECPAR 2006, pp.403-416 (2007).

[4] Y. Saad: Iterative methods for sparse linear systems, SIAM, (1996).

[5]Guy E. Blelloch, Michael A. Heroux, and Marco Zagha: Segmented Operations for Sparse Matrix Computation on Vector Multiprocessors, Carnegie Mellon University, Pittsburgh, PA, (1993).

[6] K. Naono, M. Igai and H. Kidachi: Performance Evaluation of the Gram-Schmidt Orthogonalization Library with Numerical Policy Interface on Heterogeneous Platforms, IPSJ Tran. on Advanced computing systems, 46(SIG_12(ACS_11)) , pp.279-288 (2005)(in Japanese).

[7]Daniel, J., Gragg, W.B., Kaufman, L. And Stewart, G.W.: Reorthogonalization and stable algorithms for updating the Gram-Schmidt QR factorization, Math. of Computation, Vol.30, pp.772-795 (1976).

[8] K. Naono, M. Igai and H. Kidachi: Performance Evaluation of the Gram-Schmidt Orthogonalization Library with Numerical Policy Interface on Heterogeneous Platforms, Transaction on Advanced Computing Systems, Vol. 46 No. SIG12 (ACS11), pp. 279-288 (2005) (in Japanese).

[9] Lehoucq, Richard Bruno: Analysis and implementation of an implicitly restarted Arnoldi iteration, TR95-13 (Rice Univ.)(1995)

[10] S. Itoh, K. Katagiri, T. Sakurai, M. Igai, S. Ohshima, H. Kuroda, K. Naono and K. Nakajima: An improvement in preconditioned BiCGStab method, High Performance Computing Symposium 2011, (2011) .

Appendix.A Sample code of OpenATI_EIGENSOLVE for thread-safe,

```
      PROGRAM MAIN
      IMPLICIT NONE
C
      INTEGER NMAX, NZMAX
      parameter (NMAX=268100,NZMAX=9400000)

      INTEGER NTMP,NZTMP,NEVTMP
      INTEGER IRPTMP(NMAX+1), ICOLTMP(NZMAX)
      DOUBLE PRECISION ATMP(NZMAX)
      INTEGER N, NZ, NEV, INFO
      INTEGER IRP, ICOL, IATPARAM
      ALLOCATABLE :: IRP(:), ICOL(:), IATPARAM(:)
      DOUBLE PRECISION A, E, V, RATPARAM
      ALLOCATABLE :: A(:), E(:), V(:), RATPARAM(:)
      DOUBLE PRECISION WK, O
      ALLOCATABLE :: WK(:), O(:)
      INTEGER I,ITEST,MAXP,IP,OMP_GET_THREAD_NUM
C
      EXTERNAL OMP_GET_NUM_THREADS,OMP_GET_MAX_THREADS
      INTEGER  OMP_GET_NUM_THREADS,OMP_GET_MAX_THREADS
C
      open(31,file='Input.param',status='OLD')
      read(31,*) itest
      close(31)
      CALL MATGEN(ITEST,NTMP,NZTMP,IRPTMP,ICOLTMP,ATMP)
      MAXP=OMP_GET_MAX_THREADS()
C
      NEVTMP=10
      WRITE(6,*) '++++++++++++++ Input Parameter List +++++++++++++++++'
      write(6,*) '+ itest =',itest
      WRITE(6,*) '+ Matrix Info. N=',NTMP,' NZ=',NZTMP
      WRITE(6,*) '+ OpenMP Number of MAX. Threads=',MAXP
      WRITE(6,*) '+++++++++++++++++++++++++++++++++++++++++++++++++++++'
C
!$omp parallel default(none)
!$omp+ private(N, NZ, IRP, ICOL, A, NEV, E, V, INFO)
!$omp+ private(IP, IATPARAM, RATPARAM)
!$omp+ private(WK, O)
!$omp+ shared(NTMP, NZTMP, IRPTMP, ICOLTMP, ATMP, NEVTMP, ITEST)
      IP=OMP_GET_THREAD_NUM()
*
      N=NTMP
      NZ=NZTMP
      NEV=NEVTMP
      ALLOCATE(IRP(N+1),ICOL(NZ), A(NZ))
      ALLOCATE(E(2*NEV), V(2*N*NEV))
      ALLOCATE(IATPARAM(50), RATPARAM(50))
      ALLOCATE(WK(2*N),O(NEV*N))
      DO I=1,N+1
         IRP(I)=IRPTMP(I)
      ENDDO
      DO I=1,NZ
         ICOL(I)=ICOLTMP(I)
         A(I)=ATMP(I)
      END DO

      CALL OpenATI_INIT(IATPARAM, RATPARAM, INFO)
      write(6,*)'*** OpenATI_EIGENSOLVE THREAD-SAFE TEST ***',IP

C      IATPARAM(50)=1
      IATPARAM(30)=2
      CALL OpenATI_EIGENSOLVE(N,NZ,IRP,ICOL,A,NEV,E,V,
     $                        IATPARAM,RATPARAM,INFO)
!$omp barrier
      write(6,*) 'OpenATI_EIGENSOLVE INFO=',INFO
*
      if (info.lt.0) THEN
         write(6,*) '  !!!! Parameter Error !!! Info=',INFO
```

```
         GOTO 9000
      else if (info .ne.0) then
         write(6,*) '  !!!! Breakdown Error !!! Info=',INFO
         GOTO 9000
      end if
      IF(ITEST.GT.300 .AND. ITEST.LE.321) THEN
         call resid(n,irp,icol,nz,a,nev,e,v,n,wk)
         call ORTHO(N,nev,V,N,O)
      ELSE IF (ITEST.GT.200 .AND. ITEST.LE.222)then
         call residz(n,irp,icol,nz,a,nev,e,v,n,wk)
      END IF
 9000 CONTINUE
      DEALLOCATE(IRP,ICOL, A)
      DEALLOCATE(E, V)
      DEALLOCATE(IATPARAM, RATPARAM)
      DEALLOCATE(WK,O)
*
!$omp barrier
!$omp end parallel
      STOP
      END
*
      subroutine resid(n,irp,icol,nz,a,nev,e,v,nv1,r)
      implicit real*8 (a-h,o-z)
      integer*4 irp(n+1),icol(nz)
      real*8    a(nz),e(nev),v(nv1,nev),r(n)
C>>>>>>>>>>
      resmax=0.0D0
      do 100 ic=1,nev
C--------------mat*vec
        do 200 i=1,n
          r(i)=0.0d0
  200   continue
        do 210 i=1,n
          s=a(irp(i))*v(i,ic)
          do 220 jc=irp(i)+1,irp(i+1)-1
            jj=icol(jc)
            s=s+a(jc)*v(jj,ic)
            r(jj)=r(jj)+a(jc)*v(i,ic)
  220     continue
          r(i)=r(i)+s
  210   continue
C
        do 230 i=1,n
          r(i)=r(i)-e(ic)*v(i,ic)
  230   continue
C
        zansa=0.0d0
        do 240 i=1,n
          zansa=zansa+r(i)*r(i)
  240   continue
        write(6,*) 'IC=',IC,'E=',e(ic),'RES=',sqrt(zansa)/abs(e(ic))
        resmax=max(resmax,sqrt(zansa)/abs(e(ic)))
  100 continue
C
      WRITE(6,*) '================================================'
      WRITE(6,*) '=== MAX RESID    =',resmax
      WRITE(6,*) '================================================'
C
      return
      end
C*******************
      SUBROUTINE ORTHO(N,NV,V,NV1,O)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 V(NV1,NV),O(NV1,NV)
C
      ICHK=0
      DO 400 J=1,NV
        DO 500 I=1,J
          S=0.0D0
          DO 600 K=1,N
            S=S+V(K,I)*V(K,J)
```

```
  600     CONTINUE
          IF (I.EQ.J) THEN
            IF (DABS(DSQRT(S)-1.0D0) .GT.1.0D-12) THEN
              ICHK=1
              WRITE(6,*) '!!!NG!!! EIGENVECTOR=',J,' IS NOT NORMALIZED'
     &                   ,SQRT(S)
C             RETURN
            END IF
          END IF
          O(I,J)=S
  500   CONTINUE
  400 CONTINUE
      ERR=0.0D0
      DO 700 J=1,NV
        DO 800 I=1,J-1
          IF (I.NE.J) ERR=MAX(ERR,O(I,J))
  800   CONTINUE
  700 CONTINUE
      IF (ICHK.EQ.0) THEN
        WRITE(6,*) '!!! OK !!! EIGENVECTOR NORMALIZED'
      END IF
      WRITE(6,*) '==============================================='
      WRITE(6,*) '=== ORTHOGONALITY=',ERR
      WRITE(6,*) '==============================================='
C
      RETURN
      END
      subroutine matgen(itest,n,nz,irp,icol,a)
      implicit real*8 (a-h,o-z)
      integer*4 irp(*),icol(*)
      real*8    a(*)
      character filename*60
C
      IF      (itest .EQ. 207) THEN
         filename="ex19.dat"
      else if (itest.eq.301) then
         filename='vibrobox.rb'
      end if
C
      if(itest.gt.300 .and. itest.le.321) then
         call matread(itest,filename,n,irp,icol,nz,a)
      else if(itest.gt.200 .and. itest.le.222) then
         OPEN(5,FILE=filename)
         READ(5,*) N,NZ
         READ(5,*) (IRP (I),I=1,N+1)
         READ(5,*) (ICOL(I),I=1,NZ)
         READ(5,*) (A (I),I=1,NZ)
         CLOSE(5)
      endif
C
      return
      end
      subroutine matread(itest,filename,ncol,colptr,rowind,nnzero,
     *                   values)
      implicit real*8 (a-h,o-z)
c
C     =============================================================
C     ... SAMPLE CODE FOR READING A SPARSE MATRIX IN STANDARD FORMAT
C     =============================================================

      CHARACTER      TITLE*72 , KEY*8    , MXTYPE*3 ,
     1               PTRFMT*16, INDFMT*16, VALFMT*20, RHSFMT*20
      INTEGER        TOTCRD, PTRCRD, INDCRD, VALCRD, RHSCRD,
     1               NROW  , NCOL  , NNZERO, NELTVL
      INTEGER        COLPTR (*), ROWIND (*)
      REAL*8         VALUES (*)
      character     filename*60
C
      lunit=23
      open(lunit,file=filename)
      if (itest.eq.308) then
      READ ( LUNIT, 1100 ) TITLE , KEY   ,
```

```
     1                    TOTCRD, PTRCRD, INDCRD, VALCRD, RHSCRD,
     2                    MXTYPE, NROW  , NCOL  , NNZERO,
     3                    PTRFMT, INDFMT, VALFMT, RHSFMT
 1100   FORMAT ( A72, A8 / 5I14 / A3, 11X, 3I14 / 2A16, 2A20 )
      READ ( LUNIT, * )
      else
      READ ( LUNIT, 1000 ) TITLE , KEY   ,
     1                    TOTCRD, PTRCRD, INDCRD, VALCRD, RHSCRD,
     2                    MXTYPE, NROW  , NCOL  , NNZERO, NELTVL,
     3                    PTRFMT, INDFMT, VALFMT, RHSFMT
 1000   FORMAT ( A72, A8 / 5I14 / A3, 11X, 4I14 / 2A16, 2A20 )
      endif
      write(6,*) '===> INPUT FILE NAME IS ',filename
      write(6,*) TITLE
      write(6,*) KEY
C     ------------------------
C     ... READ MATRIX STRUCTURE
C     ------------------------
      READ ( LUNIT, PTRFMT ) ( COLPTR (I), I = 1, NCOL+1 )
      READ ( LUNIT, INDFMT ) ( ROWIND (I), I = 1, NNZERO )
      IF  ( VALCRD .GT. 0 )  THEN
C         ---------------------
C         ... READ MATRIX VALUES
C         ---------------------
          READ ( LUNIT, VALFMT ) ( VALUES (I), I = 1, NNZERO )
      ENDIF
      return
      end
*
      subroutine residz(n,irp,icol,nz,a,nev,e,v,nv1,r)
      implicit real*8 (a-h,o-z)
      integer*4 irp(n+1),icol(nz)
      real*8    a(nz)
      complex*16   e(nev),v(nv1,nev),r(n)
      complex*16   s
C
      resmax=0.0D0
      do 100 ic=1,nev
C---------------mat*vec
        do 210 i=1,n
          s=dcmplx(0.0d0,0.0d0)
          do 220 jc=irp(i),irp(i+1)-1
            jj=icol(jc)
            s=s+a(jc)*v(jj,ic)
  220     continue
          r(i)=s
  210   continue
        do 230 i=1,n
          r(i)=r(i)-e(ic)*v(i,ic)
  230   continue
        zansa=0.0d0
        do 240 i=1,n
          zansa=zansa+dreal(conjg(r(i))*r(i))
  240   continue
        write(6,*) 'IC=',IC,'E=',e(ic),'RES=',sqrt(zansa)/abs(e(ic))
        resmax=max(resmax,sqrt(zansa)/abs(e(ic)))
  100 continue
      WRITE(6,*) '================================================'
      WRITE(6,*) '=== MAX RESID    =',resmax
      WRITE(6,*) '================================================'
      return
      end
```

Appendix.B Revision History of OpenATLib and Xabclib

- Version 1.03 (June 26th, 2015)

Sorce code

i) Xabclib_CG is implemented.

Manual

a) Xabclib_CG is added.

- Version 1.02 (October 26th, 2012)

Sorce code

i) "make.all" file in Xabclib-vX.XX directory is updated for building sample programs after making archive file of OpenATlib and Xabclib.

ii) Preconditioner functions of linear solvers (Xabclib_GMRES and Xabclib_BICGSTAB) are updated for outputting error code 100 when creating Jacobi, SSOR, ILU0-Diagonal and ILU0 preconditioner of the matrices having empty diagonal elements.

iii) A bug of selection algorithm for the "better" combination of preconditioner and iterative solver in OpenATI_LINEARSOLVE is fixed.

Manual

a) Appendix.B is added.

b) Descriptions of IATPARAM(7) are updated. (IATPARAM(7)=1: reserved)

- Version 1.01 (May 15th, 2012)

Sorce code

i) Bugs caused by false OpenMP directives in Xabclib_GMRES and Xabclib_BICGSTAB are fixed.