

# FrontISTR インストールマニュアル

FrontISTR Commons

2019 年 7 月 30 日

## 目次

|        |                           |    |
|--------|---------------------------|----|
| 1      | FrontISTR インストールマニュアル     | 2  |
| 1.1    | マニュアルリスト                  | 3  |
| 1.2    | 本マニュアルの記載内容               | 3  |
| 1.3    | 動作環境                      | 3  |
| 1.3.1  | 必要なソフトウェア                 | 3  |
| 1.3.2  | 動作確認環境                    | 6  |
| 1.4    | アーカイブファイルの解凍・展開           | 7  |
| 1.5    | インストール                    | 7  |
| 1.5.1  | サポートされているインストール方法         | 7  |
| 1.6    | cmake でのインストール            | 9  |
| 1.6.1  | 準備                        | 9  |
| 1.6.2  | 構築                        | 9  |
| 1.6.3  | make install の実行          | 9  |
| 1.6.4  | cmake のオプション              | 10 |
| 1.7    | 簡易テスト機能について               | 11 |
| 1.8    | ソースコードのドキュメンテーションについて     | 12 |
| 1.9    | デバッグを有効にする                | 12 |
| 1.10   | Makefile.conf でのインストール    | 12 |
| 1.10.1 | Makefile.conf の編集         | 12 |
| 1.10.2 | setup.sh の実行              | 13 |
| 1.10.3 | make の実行                  | 14 |
| 1.10.4 | make install の実行          | 15 |
| 1.10.5 | Windows 環境へのインストール        | 15 |
| 1.11   | 付録                        | 15 |
| 1.11.1 | Makefile.conf の変数一覧       | 15 |
| 1.11.2 | Makefile.conf の設定例        | 19 |
| 1.11.3 | 京コンピュータおよび富士通 FX10 における注意 | 20 |
| 1.12   | 付録                        | 21 |
| 1.12.1 | Makefile.conf の変数一覧       | 21 |
| 1.12.2 | Makefile.conf の設定例        | 24 |
| 1.12.3 | 京コンピュータおよび富士通 FX10 における注意 | 26 |

|   |    |
|---|----|
| 1.13 参考 CentOS7.6 へのインストール手順例 (cmake) . . . . .           | 26 |
| 1.13.1 準備 . . . . .                                       | 26 |
| 1.13.2 ライブラリのインストール . . . . .                             | 27 |
| 1.13.3 FrontISTR のコンパイル . . . . .                         | 30 |
| 1.14 参考 CentOS7.6 へのインストール手順例 (Makefile.conf) . . . . .   | 32 |
| 1.14.1 準備 . . . . .                                       | 33 |
| 1.14.2 ライブラリのインストール . . . . .                             | 33 |
| 1.14.3 FrontISTR のコンパイル . . . . .                         | 37 |
| 1.15 参考 Ubuntu18.04 へのインストール手順例 (cmake) . . . . .         | 41 |
| 1.15.1 準備 . . . . .                                       | 41 |
| 1.15.2 ライブラリのインストール . . . . .                             | 41 |
| 1.15.3 FrontISTR のコンパイル . . . . .                         | 45 |
| 1.16 参考 Ubuntu18.04 へのインストール手順例 (Makefile.conf) . . . . . | 47 |
| 1.16.1 準備 . . . . .                                       | 47 |
| 1.16.2 ライブラリのインストール . . . . .                             | 47 |
| 1.16.3 FrontISTR のコンパイル . . . . .                         | 51 |
| 1.17 参考 Windows10 へのインストール手順例 (Makefile.conf) . . . . .   | 55 |
| 1.17.1 準備 . . . . .                                       | 55 |
| 1.17.2 ライブラリのインストール . . . . .                             | 56 |
| 1.17.3 FrontISTR のコンパイル . . . . .                         | 61 |

## 1 FrontISTR インストールマニュアル

本ソフトウェアは文部科学省次世代 IT 基盤構築のための研究開発「イノベーション基盤シミュレーションソフトウェアの研究開発」プロジェクトによる成果をシーズとして、継続的に開発されている並列有限要素解析プログラムです。本ソフトウェアを無償または営利目的でご使用になる場合、「MIT ライセンス」をご了承頂くことが前提となります。



| 項目       | 説明  |
|----------|---|
| ソフトウェア名称 | FrontISTR   |
| バージョン    | 5.0   |
| ライセンス形態  | MIT License   |
| 問い合わせ先   | 一般社団法人 FrontISTR Commons 東京都文京区弥生二丁目 11 番 16 号 (東京大学大学院工学系研究科総合研究機構内)E-mail : support@frontistr.com |

## 1.1 マニュアルリスト

- イン트로ダクション
- インストールマニュアル
- 理論マニュアル
- 解析マニュアル
- チュートリアル

本マニュアルでは、大規模並列 FEM 非線形構造解析プログラム FrontISTR のインストール方法を説明します。

## 1.2 本マニュアルの記載内容

- PDF
- 動作環境
- アーカイブの解凍・展開
- インストールの概要
- cmake でのインストール
  - CentOS7.6 へのインストール手順例 (cmake)
  - Ubuntu18.04 へのインストール手順例 (cmake)
- Makefile.conf でのインストール
  - Makefile.conf での設定項目
  - CentOS7.6 へのインストール手順例 (Makefile.conf)
  - Ubuntu18.04 へのインストール手順例 (Makefile.conf)
  - Windows10 へのインストール手順例 (Makefile.conf)

## 1.3 動作環境

### 1.3.1 必要なソフトウェア

本ソフトウェアのインストールに際して、インストールする環境に以下のソフトウェアがインストールされている必要があります。なお、これらのソフトウェアのインストールについては、各ソフトウェアのインストールマニュアルをご参照ください。

#### 1.3.1.1 C、C++、Fortran90 コンパイラー

本ソフトウェアのインストールには、C、C++ および Fortran90 コンパイラーが必要です。

#### 1.3.1.2 MPI

本ソフトウェアは MPI により並列化されているため、MPI-1 規格に準拠した MPI ライブラリが必要となります。MPI を実装したフリーで利用できるライブラリの代表的なものには、MPICH や OpenMPI などがあります。

OpenMPI は下記の WEB サイトから

<https://www.open-mpi.org/>

MPICH は下記の WEB サイトからダウンロードすることができます。

<http://www.mpich.org/>

#### 1.3.1.3 METIS

本ソフトウェアの領域分割ユーティリティは、METIS のライブラリを使用することで pMETIS、kMETIS による領域分割が可能です。これらの領域分割機能を利用するには METIS が必要となります。なお、METIS のバージョンは、最新の Ver.5 系列と Ver.4 系列が利用可能です。

また、METIS がインストールされていない環境でも、RCB アルゴリズムによる領域分割は可能です。

METIS は下記の WEB サイトからダウンロードすることができます。

<http://glaros.dtc.umn.edu/gkhome/metis/metis/overview>

#### 1.3.1.4 ParMETIS

本ソフトウェアの並列領域分割ユーティリティは、ParMETIS ライブラリを使用する予定です。

現時点では ParMETIS は不要です。

ParMETIS は下記の WEB サイトからダウンロードすることができます。

<http://glaros.dtc.umn.edu/gkhome/metis/parmetis/overview>

#### 1.3.1.5 HEC-MW

本ソフトウェアは、「革新的シミュレーションソフトウェアの研究開発」プロジェクトおよび「イノベーション基盤シミュレーションソフトウェアの研究開発」プロジェクトで開発された HEC-MW ライブラリを利用しています。

この HEC-MW は FrontISTR のアーカイブに同梱されており、本ソフトウェアのインストール時に自動的にコンパイルされるため、別途インストールする必要はありません。

#### 1.3.1.6 REVOCAP\_Refiner

本ソフトウェアは、「イノベーション基盤シミュレーションソフトウェアの研究開発」プロジェクトで開発されたメッシュ細分化ツール REVOCAP\_Refiner に対応しています。

メッシュ細分化機能を利用する場合には REVOCAP\_Refiner が必要となります。REVOCAP\_Refiner の最新版は下記の WEB サイトからダウンロードすることができます。

<http://www.multi.k.u-tokyo.ac.jp/FrontISTR/>

#### 1.3.1.7 REVOCAP\_Coupler

本ソフトウェアは、「イノベーション基盤シミュレーションソフトウェアの研究開発」プロジェクトで開発された連成解析ツール REVOCAP\_Coupler に対応しています。連成解析機能を利用する場合には REVOCAP\_Coupler が必要となります。REVOCAP\_Coupler は下記の WEB サイトからダウンロードすることができます。

<http://www.ciiss.iis.u-tokyo.ac.jp/dl/index.php>

#### 1.3.1.8 LAPACK/BLAS

本ソフトウェアは、CG 法および GMRES 法を用いた前処理適用後行列の条件数推定機能が実装されています。本機能を利用する場合には LAPACK が必要になります。また、LAPACK を利用するには BLAS が必要となります。

LAPACK のリファレンス実装は下記 WEB サイトからダウンロードすることができます。

<http://www.netlib.org/lapack/>

BLAS のリファレンス実装は下記 WEB サイトからダウンロードすることができます。

<http://www.netlib.org/blas/>

高速なオープンソースの実装としては OpenBLAS などが利用できます。OpenBLAS は下記 WEB サイトからダウンロードすることができます。

<http://www.openblas.net/>

なお、後述する Intel MKL がインストールされている場合、改めてインストールする必要はありません。

#### 1.3.1.9 MUMPS

本ソフトウェアは、パブリックドメインの並列直接法ソルバー MUMPS (a MULTifrontal Massively Parallel sparse direct Solver) に対応しています。MUMPS は、Esprit IV European project PARASOL (1996-1999) で開発されたソフトウェアをベースとし、CERFACS, CNRS, ENS Lyon, INPT(ENSEEIH)-IRIT, INRIA および University of

Bordeaux の各機関により研究開発されたものです。MUMPS は下記の WEB サイトからダウンロードすることができます。

<http://mumps.enseiht.fr/>

#### 1.3.1.10 ScaLAPACK

本ソフトウェアで直接利用していませんが、上述の MUMPS は ScaLAPACK を利用します。ScaLAPACK は下記の WEB サイトからダウンロードすることができます。

<http://www.netlib.org/scalapack/>

なお、後述する Intel MKL がインストールされ ScaLAPACK ライブラリがインストールされている場合、改めてインストールする必要はありません。

#### 1.3.1.11 ML

本ソフトウェアは、代数マルチグリッド法に基づく前処理ライブラリ ML (Multi-Level Preconditioner) に対応しています。ML は、Sandia National Laboratories で進められている Trilinos プロジェクトで開発されているパッケージのひとつです。ML は下記の WEB サイトからダウンロードすることができます。

<https://trilinos.org/>

#### 1.3.1.12 Intel MKL (Math Kernel Library)

本ソフトウェアの接触解析モジュールでは、Intel MKL を利用しています。

インストールする環境に Intel MKL がインストールされていない場合、接触解析の一部の機能が利用できません。

### 1.3.2 動作確認環境

本ソフトウェアは、下記の環境において動作確認を行っています。ただし、これ以外の環境においても、前述のインストールに必要なソフトウェアが導入されている場合、正常に動作すると思われます。

#### 1.3.2.1 動作確認環境

| システム                      | オペレーティングシステム | CPU                    | コンパイラ            | 並列化環境       |
|---------------------------|--------------|------------------------|------------------|-------------|
| K computer                | Linux        | SPARC64 VIIIfx         | Fujitsu compiler | Fujitsu MPI |
| PRIMEHPC                  | Linux        | SPARC V9 +<br>HPC-ACE2 | Fujitsu compiler | Fujitsu MPI |
| FX100                     |              |                        |                  |             |
| EARTH SIMU-<br>LATOR(ES3) | SUPER UX     | SX-ACE                 | NEC compiler     | NEC MPI     |

| システム              | オペレーティングシステム                      | CPU            | コンパイラ                | 並列化環境         |
|-------------------|-----------------------------------|----------------|----------------------|---------------|
| UV2000            | Linux (SUSE Linux Enterprise 10)  | Intel Xeon     | Intel compiler       | SGI MPT       |
| PC cluster        | Linux (CentOS-7)                  | Intel Xeon     | Intel compiler       | Intel MPI     |
| PC cluster        | Linux (RedHat Enterprise Linux 7) | Intel Xeon     | Intel compiler       | OpenMPI       |
| Desktop PC        | Linux (ubuntu 16.04, 18.04)       | AMD Ryzen      | GNU compiler         | OpenMPI       |
| Desktop PC        | Linux (ubuntu 16.04, 18.04)       | AMD Ryzen      | PGI compiler         | OpenMPI       |
| Desktop PC        | Linux (ubuntu 16.04, 18.04)       | Intel Core-i7  | GNU compiler         | OpenMPI       |
| Desktop PC        | Windows (7, 10)                   | Intel Core-i7  | GNU compiler (mingw) | Microsoft MPI |
| Raspberry PI 3 B+ | Linux (raspbian 32bit)            | ARM Cortex-A53 | GNU compiler         | OpenMPI       |
| Notebook PC       | macOS Mojave                      | Intel Core i7  | GNU Compiler         | OpenMPI       |

## 1.4 アーカイブファイルの解凍・展開

アーカイブファイルは、tar によりアーカイブ化され、gzip により圧縮されています。このアーカイブファイルを、以下のコマンドで解凍・展開します。

```
$ tar xzf FrontISTR_V50.tar.gz
```

本ソフトウェアをインストールする環境の tar コマンドが z オプションをサポートしていない場合は、以下のコマンドで解凍・展開します。

```
$ gzip -dc FrontISTR_V50.tar.gz | tar xf -
```

アーカイブファイルを解凍・展開すると、アーカイブを展開したディレクトリに FrontISTR というディレクトリが作成されます。以下、このディレクトリを  $\${FSTRBUILDDIR}$  と記します。

## 1.5 インストール

### 1.5.1 サポートされているインストール方法

本ソフトウェアでは、2つのインストール方法がサポートされています。

#### 1.5.1.1 cmake でのインストール

本ソフトウェアは、cmake を用いたインストールをサポートしています。

cmake を予めインストールしておく必要があります。cmake は下記 WEB サイトからダウンロードすることができます。

<https://cmake.org/>

```
$ cd ${FSTRBUILDDIR}
$ mkdir build
$ cd build
$ cmake ..
$ make -j2
$ make install
```

インストールされているライブラリを自動で探索し、FrontISTR の機能を有効にします。また、複数コアを持ったコンピュータでは、並列コンパイルを有効にすることで、コンパイル時間の短縮が期待できます。

cmake でのインストールつづき

#### 1.5.1.1.1 参考

- 参考 CentOS7.6 へのインストール手順例 (cmake)
- 参考 Ubuntu18.04 へのインストール手順例 (cmake)

#### 1.5.1.2 Makefile.conf でのインストール

本ソフトウェアでは、手動でライブラリやコンパイラ、有効にする機能を指定する方法がサポートされています。

```
$ cd ${FSTRBUILDDIR}
$ cp Makefile.conf.org Makefile.conf
$ vi Makefile.conf
    ファイルを編集しコンパイラやライブラリの場所を指定
$ ./setup.sh [有効にしたい機能を指定]
$ make
$ make install
```

cmake での自動設定が困難な環境では、こちらの方法での構築を推奨します。なお、こちらの方法は並列コンパイルがサポートされていません。

Makefile.conf でのインストールつづき

#### 1.5.1.2.1 参考

- 参考 CentOS7.6 へのインストール手順例 (Makefile.conf)
- 参考 Ubuntu18.04 へのインストール手順例 (Makefile.conf)
- 参考 Windows10 へのインストール手順例 (Makefile.conf)



## 1.6 cmake でのインストール

cmake には、ライブラリの自動探索機能が備わっています。それらを手動で明示することもできます。

cmake コマンドの詳細は、<https://cmake.org/> をご覧ください。

### 1.6.1 準備

本ソフトウェアの構築に必要なライブラリを予めインストールします。

インストールするライブラリのディレクトリ構成は

```
$HOME
|-- local
    |-- bin
    |-- include
    |-- lib
```

の様な構成を推奨します。

その際、上記の場合 \$PATH 環境変数に \$HOME/local/bin を追加してください。

cmake がインストールされているかを確認します。cmake はバージョン 2.8.11 以上が必要になります。

```
$ cmake --version
cmake version 2.8.12.2
```

### 1.6.2 構築

次に FrontISTR を構築します。

```
$ cd '${FSTRBUILDDIR}'
$ mkdir build
$ cd build
$ cmake ..
$ make -j2
```

make のオプション `-j2` は、並列コンパイルの数を示しています。構築するコンピュータのコア数に併せて数を増やすことで、コンパイル時間の短縮が期待できます。

### 1.6.3 make install の実行

make の実行が正常に終了したあと、本ソフトウェアをインストールするため、以下のコマンドを実行します。

```
$ make install
```

以上で/usr/local/binもしくは、-DCMAKE\_INSTALL\_PREFIXで指定したディレクトリに、本ソフトウェアがインストールされます。

インストールする場所を変えるには、cmake コマンドにオプションを追加します。

```
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/local ..
```

などとオプションを追加してください。

コンパイルされた FrontISTR( fistr1 ) が、どの機能を有効になっているかは

```
./fistr1 -v
FrontISTR version 5.0.0 (eb7fb1c1a3d210b0c1f70b41c92995bfc050e82)
MPI: Enabled
OpenMP: Enabled
HECMW_METIS_VER: 5
Compile Option: -p --with-tools --with-metis --with-mumps --with-lapack --with-ml
```

で確認することができます。

#### 1.6.4 cmake のオプション

cmake コマンドを実行する際、オプションを指定することで挙動を明示的に指定することができます。

| オプション (デフォルト)        | 説明                             | 備考                     |
|----------------------|--------------------------------|------------------------|
| -DWITH_TOOLS=ON      | パーティショナなどのツールもコンパイル            | hecmw_part1 などツール      |
| -DWITH_MPI=ON        | MPI を有効                        | ライブラリが必要               |
| -DWITH_OPENMP=ON     | OpenMP を有効                     | コンパイラの対応が必要            |
| -DWITH_REFINER=ON    | REVOCAP_Refiner の機能を有効         | ライブラリが必要               |
| -DWITH_REVOCAP=ON    | REVOCAP_Coupler の機能を有効         | ライブラリが必要               |
| -DWITH_METIS=ON      | METIS の機能を有効                   | 4.0.3 と 5.1.0 に対応      |
| -DMETIS_VER_4=OFF    | metis-4.0.3 を使う場合に設定           | metis-5.1.0 の場合指定不要    |
| -DWITH_PARMETIS=ON   | ParMETIS の機能を有効                | 3.2.0 と 4.0.3 に対応      |
| -DMETIS_VER_3=OFF    | ParMetis-3.2.0 を使う場合に設定        | parmetis-4.0.3 の場合指定不要 |
| -DWITH_MKL=ON        | MKL PARDISO の機能を有効             | ライブラリが必要               |
| -DWITH_MUMPS=ON      | MUMPS の機能を有効                   | ライブラリが必要               |
| -DWITH_LAPACK=ON     | LAPACK の機能を有効                  | ライブラリが必要               |
| -DWITH_ML=ON         | Trilinos ML の機能を有効             | ライブラリが必要               |
| -DWITH_DOC=OFF       | FrontISTR のソースコードをドキュメント化      | doxygen と graphviz が必要 |
| -DOLD_RES_FORMAT=OFF | ON で result ファイルの旧フォーマット出力を有効化 |                        |

cmake で設定されている変数の一覧は

```
$ cmake -L
```

で確認できます。

その他、使用するコンパイラの指定やライブラリの指定をするオプションは以下の通りです。

| オプション                         | 説明   | 備考  |
|-------------------------------|--|---|
| -<br>DBLA_VENDOR=             | 利用する BLAS のベンダーを指定                             | FindBLAS.cmake を参照  |
| -<br>DBLAS_LIBRARIES=         | BLAS ライブラリを直接指定                                | ライブラリを絶対パスで直接指定   |
| -<br>DLAPACK_LIBRARIES=       | LAPACK ライブラリを直接指定                              | ライブラリを絶対パスで直接指定   |
| -<br>DCMAKE_INSTALL_PREFIX=   | インストールするパスを設定。デフォルトは <code>\$HOME/local</code> | -DCMAKE_INSTALL_PREFIX=\$HOME/local で \$HOME/local/bin などにプログラムがインストールされる |
| -<br>DCMAKE_C_COMPILER=       | C コンパイラを指定                                     | -DCMAKE_C_COMPILER=icc (Intel C コンパイラ)                                    |
| -<br>DCMAKE_CXX_COMPILER=     | C++ コンパイラを指定                                   | -DCMAKE_CXX_COMPILER=icpc (Intel C++ コンパイラ)                               |
| -<br>DCMAKE_Fortran_COMPILER= | Fortran コンパイラを指定                               | -DCMAKE_Fortran_COMPILER=ifort (Intel Fortran コンパイラ)                      |
| -<br>DCMAKE_PREFIX_PATH=      | ライブラリ等の格納場所を指定                                 | -DCMAKE_PREFIX_PATH=\$HOME/tools (ライブラリなどを探索するパス)                         |

## 1.7 簡易テスト機能について

本ソフトウェアには、コンパイルしたオブジェクトが正しく動くことを確認するための簡易テストスクリプトが同梱されています。

テストを行うにはrubyを予めインストールします。rubyがインストールされていれば、cmake時にテストが自動的に有効になります。

cmakeで本ソフトウェアをコンパイル後、以下のようにしてテストを実行します。

```
$ make test
```

テストは以下のように実行されます。

```
/home/fistr/Work/FrontISTR/build$ make test
```

```
Running tests...
```

```
Test project /home/fistr/Work/FrontISTR/build
```

```
Start 1: Static_exA_Test
```

```
1/23 Test #1: Static_exA_Test ..... Passed 6.85 sec
```

```
Start 2: Static_exB_Test
```

```
2/23 Test #2: Static_exB_Test ..... Passed 6.48 sec
```

```
Start 3: Static_exC_Test
```

```
...
```

更に詳細なメッセージを出力する場合

```
$ make test ARGS="--VV -O test_log.txt"
```

とすると、test\_log.txtファイルの中に結果が出力されます。オプションの詳細は

```
$ ctest --help
```

を参照してください。

## 1.8 ソースコードのドキュメンテーションについて

本ソフトウェアのソースコードを学習に用いる際、各サブルーチンの相関やソースコードに埋め込まれているコメントを、ブラウザで参照することができます。

ソースコードのドキュメントを HTML で構築するには、予めdoxygenとgraphvizをインストールします。

以下の手順で HTML を構築します。

```
$ cmake -DWITH_DOC=ON ..  
$ make doc
```

作成された HTML を以下のようにして参照します。

```
$ firefox doc/html/index.html
```

## 1.9 デバッグを有効にする

デバッグを有効にするには、

```
$ cmake -DCMAKE_BUILD_TYPE="DEBUG" ..
```

としてからmakeをします。更に高度なデバッグオプションを有効にするには

```
$ cmake -DCMAKE_BUILD_TYPE="DEBUG" -DDEBUG_EXTRA=ON ..
```

とすると、メモリリークなどの検出に役立ちます。

## 1.10 Makefile.conf でのインストール

以下の手順で、本ソフトウェアをインストールします。

### 1.10.1 Makefile.conf の編集

`${FSTRBUILDDIR}`にある `Makefile.conf.org` を、本ソフトウェアをインストールする計算機環境に合わせて編集し、`Makefile.conf` を作成します。定義できる変数は数多くありますが、ほとんどの変数については既定値をそのまま利用できます。多くの環境では、下記の変数以外を変更する必要はないと思われます。

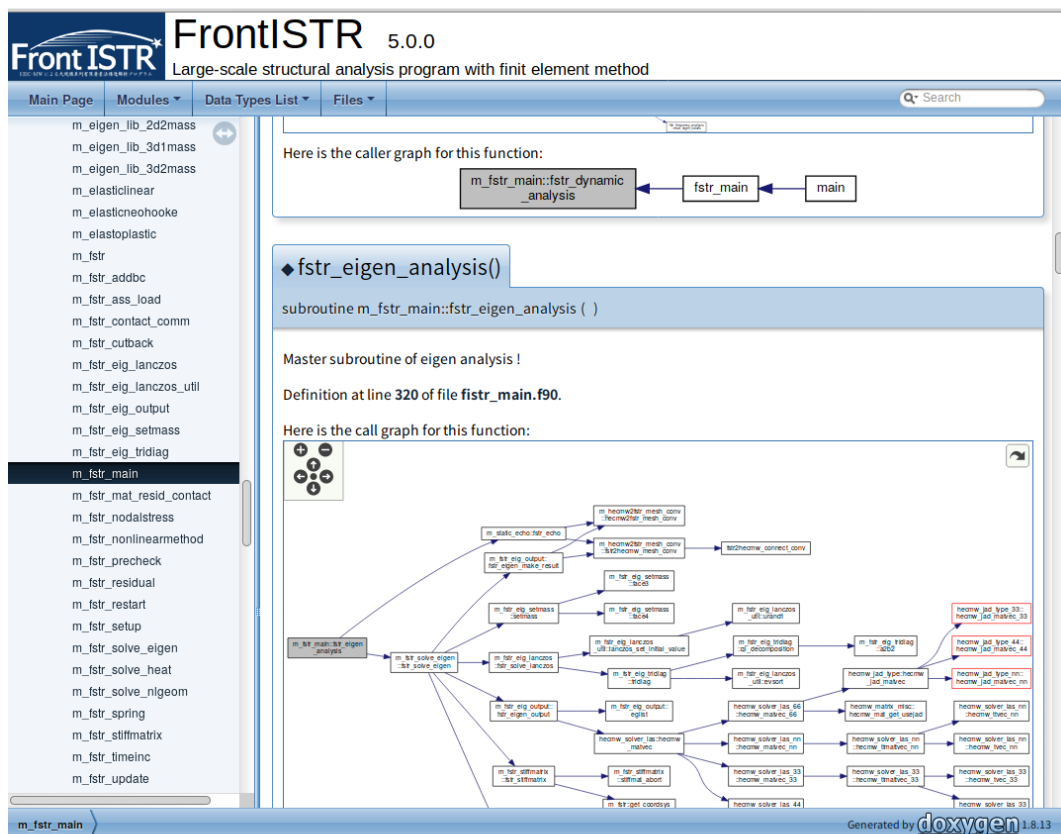


図1 ドキュメンテーション

| 変数名         | 説明                                 |
|-------------|------------------------------------|
| MPIDIR      | MPI がインストールされているディレクトリ             |
| PREFIX      | 本ソフトウェアの実行モジュールをインストールするディレクトリ     |
| METISDIR    | METIS がインストールされているディレクトリ           |
| PARMETISDIR | ParMETIS がインストールされているディレクトリ        |
| REFINERDIR  | REVOCAP_Refiner がインストールされているディレクトリ |
| REVOCAPDIR  | REVOCAP_Coupler がインストールされているディレクトリ |
| MUMPSDIR    | MUMPS がインストールされているディレクトリ           |
| CC          | C コンパイラ起動コマンド                      |
| CPP         | C++ コンパイラ起動コマンド                    |
| F90         | Fortran90 コンパイラ起動コマンド              |

すべての変数の詳細については、「付録 1 Makefile.conf の変数一覧」をご参照ください。また、「付録 2 Makefile.conf の設定例」に Makefile.conf の一例を記載します。

### 1.10.2 setup.sh の実行

`${FSTRBUILDDIR}` にて、シェルスクリプト `setup.sh` を以下のように実行し、Makefile を作成します。

```
$ ./setup.sh
```

並列計算用のライブラリを生成する場合などは、下記のオプションを指定して `setup.sh` を実行して下さい。

#### 1.10.2.2 並列処理用にコンパイルする場合

MPI がインストールされている並列実行環境で本ソフトウェアを使用する場合、以下のように **\*\* -p または -parallel \*\*** オプションを付けて `setup.sh` を起動します。

```
$ ./setup.sh -p
```

#### 1.10.2.3 パーティショナーなどのツールを生成する場合

パーティショナー (RCB) やビジュアライザーなどのプリ・ポスト処理用ツールが必要な場合、以下のように **\*\* -with-tools \*\*** オプションを付けて `setup.sh` を実行すると、各種ツールが生成されます。

```
$ ./setup.sh -p --with-tools
```

#### 1.10.2.4 METIS を使用する場合

METIS がインストールされている環境では、さらに以下のように **\*\* -with-metis \*\*** オプションを付けて `setup.sh` を実行すると、パーティショナーにおいて METIS の使用が可能となります。

```
$ ./setup.sh -p --with-tools --with-metis
```

#### 1.10.2.5 接触解析用にコンパイルする場合

接触解析用にコンパイルする場合、並列なしの場合と並列ありの場合の 2 通りの方法があります。並列なしの場合は、Intel MKL または MUMPS の利用が必要となります。

```
$ ./setup.sh --with-mkl
```

または、

```
$ ./setup.sh --with-mumps
```

並列ありで接触解析を行う場合は、**\*\* -p 、 -with-metis \*\*** オプションも必要となります。また並列ありの場合は Intel MKL は使えません。

```
$ ./setup.sh -p --with-metis --with-mumps
```

#### 1.10.3 make の実行

`${FSTRBUILDDIR}` にて、以下のように `make` を実行します。

```
$ make 2 > & 1 | tee make.log
```

make の実行には、計算機環境によっては数十分かかる場合があります。実行中にエラーが生じた場合は、Makefile.conf の設定の見直し等を行なって下さい。

#### 1.10.4 make install の実行

make の実行が正常に終了した後、Makefile.conf で指定したディレクトリに本ソフトウェアをインストールするために、以下のように make install を実行します。

```
$ make install
```

#### 1.10.5 Windows 環境へのインストール

Windows 環境では、以下の UNIX ライク環境を用いることにより、上記の手順でインストールが可能です。

- 逐次処理版 : MinGW, Cygwin
- 並列処理版 : MinGW + Microsoft MPI, Cygwin + OpenMPI

### 1.11 付録

#### 1.11.1 Makefile.conf の変数一覧

##### 1.11.1.1 MPI に関する設定

MPI 対応コンパイラーが自動参照している場合は、MPI に関する設定は不要である。

| 変数名       | 説明  | 既定値 |
|-----------|---|-----|
| MPIDIR    | MPI がインストールされているディレクトリのパスを指定する                    | なし  |
| MPIBINDIR | MPI の実行ファイル群がインストールされているディレクトリのパスを指定する            | なし  |
| MPIINCDir | MPI のヘッダーファイル群がインストールされているディレクトリのパスを指定する          | .   |
| MPILIBDIR | MPI のライブラリ群がインストールされているディレクトリのパスを指定する             | .   |
| MPILIBS   | C および Fortran90 のオブジェクトファイルにリンクさせる MPI ライブラリを指定する | なし  |

##### 1.11.1.2 インストールディレクトリに関する設定

| 変数名        | 説明                                       | 既定値                |
|------------|--|--------------------|
| PREFIX     | 本ソフトウェアをインストールするディレクトリのパスを指定する           | \$(HOME)/FrontIST  |
| BINDIR     | 本ソフトウェアの実行ファイル群をインストールするディレクトリのパスを指定する   | \$(PREFIX)/bin     |
| INCLUDEDIR | 本ソフトウェアのヘッダーファイル群をインストールするディレクトリのパスを指定する | \$(PREFIX)/include |
| LIBDIR     | 本ソフトウェアのライブラリ群をインストールするディレクトリのパスを指定する    | \$(PREFIX)/lib     |

#### 1.11.1.3 METIS に関する設定

| 変数名         | 説明   | 既定値                  |
|-------------|--|----------------------|
| METISDIR    | METIS がインストールされているディレクトリのパスを指定する                         | \$(HOME)/metis       |
| METISINCDIR | METIS のヘッダーファイル群 (metis.h など) がインストールされているディレクトリのパスを指定する | \$(METISDIR)/include |
| METISLIBDIR | METIS のライブラリ (libmetis.a) がインストールされているディレクトリのパスを指定する     | \$(METISDIR)/lib     |

#### 1.11.1.4 ParMETIS に関する設定

| 変数名            | 説明   | 既定値                     |
|----------------|--|-------------------------|
| PARMETISDIR    | ParMETIS がインストールされているディレクトリのパスを指定する。                           | \$(HOME)/ParMetis       |
| PAEMETISINCDIR | ParMETIS のヘッダーファイル群 (parmetis.h など) がインストールされているディレクトリのパスを指定する | \$(PARMETISDIR)/include |
| PARMETISLIBDIR | ParMETIS のライブラリ (libparmetis.a) がインストールされているディレクトリのパスを指定する     | \$(PARMETISDIR)/lib     |

#### 1.11.1.5 REVOCAP\_Refiner に関する設定

| 変数名           | 説明   | 既定値                      |
|---------------|--|--------------------------|
| REFINERDIR    | REVOCAP_Refiner がインストールされているディレクトリのパスを指定する           | \$(HOME)/REVOCAP_Refiner |
| REFINERINCDIR | REVOCAP_Refiner のヘッダーファイル群がインストールされているディレクトリのパスを指定する | \$(REFINERDIR)/include   |
| REFINERLIBDIR | REVOCAP_Refiner のライブラリ群がインストールされているディレクトリのパスを指定する    | \$(REFINERDIR)/lib       |

#### 1.11.1.6 REVOCAP\_Coupler に関する設定

| 変数名        | 説明   | 既定値                      |
|------------|--|--------------------------|
| REVOCAPDIR | REVOCAP_Coupler がインストールされているディレクトリのパスを指定する | \$(HOME)/REVOCAP_Coupler |



| 変数名           | 説明   | 既定値                    |
|---------------|--|------------------------|
| REVOCAPINC    | REVOCAP_Coupler のヘッダーファイル群がインストールされているディレクトリのパスを指定する | \$(REVOCAPDIR)/include |
| REVOCAPLIBDIR | REVOCAP_Coupler のライブラリ群がインストールされているディレクトリのパスを指定する    | \$(REVOCAPDIR)/lib     |

#### 1.11.1.7 MUMPS に関する設定

| 変数名         | 説明   | 既定値                  |
|-------------|--|----------------------|
| MUMPSDIR    | MUMPS がインストールされているディレクトリのパスを指定する           | \$(HOME)/MUMPS       |
| MUMPSINCDIR | MUMPS のヘッダーファイル群がインストールされているディレクトリのパスを指定する | \$(MUMPSDIR)/include |
| MUMPSLIBDIR | MUMPS のライブラリ群がインストールされているディレクトリのパスを指定する    | \$(MUMPSDIR)/lib     |

#### 1.11.1.8 ML に関する設定

| 変数名      | 説明                                      | 既定値               |
|----------|---|-------------------|
| MLDIR    | ML がインストールされているディレクトリのパスを指定する           | \$(HOME)/trilinos |
| MLINCDIR | ML のヘッダーファイル群がインストールされているディレクトリのパスを指定する | \$(MLDIR)/include |
| MLLIBDIR | ML のライブラリ群がインストールされているディレクトリのパスを指定する    | \$(MLDIR)/lib     |

#### 1.11.1.9 C コンパイラに関する設定

| 変数名      | 説明   | 既定値    |
|----------|--|--------|
| CC       | C コンパイラの起動コマンドを指定する  | mpicc  |
| CFLAGS   | C コンパイラに付与するオプションを指定する   | なし     |
| LDFLAGS  | リンカーに付与するオプションを指定する。REVOCAP_Refiner を使用する場合で、C プログラムのリンクに C コンパイラを用いる場合には、C++ の標準ライブラリ (-lstdc++ など) を指定する必要がある。 | -lm    |
| OPTFLAGS | C コンパイラに付与する最適化オプションなどを指定する  | -O3    |
| CLINKER  | C プログラムのリンク時に用いるコマンドを指定する。REVOCAP_Refiner を使用する場合で、C プログラムのリンクに C++ コンパイラを用いる必要がある場合などに指定する。                     | \$(CC) |

#### 1.11.1.10 C++ コンパイラに関する設定

| 変数名      | 説明   | 既定値                |
|----------|--|--------------------|
| CXX      | C++ コンパイラの起動コマンドを指定する  | mpic++             |
| CXXFLAGS | C++ コンパイラに付与するオプションを指定する。Boost ライブラリが C++ コンパイラから自動参照されない場合、-I オプションにより、インクルードファイルが格納されているディレクトリを指定する。 | -DMPICH_IGNORE_CXX |
| CXXLD    | C++ リンカーに付与するオプションを指定する  | なし                 |
| CXXOPTS  | C++ コンパイラに付与する最適化オプションなどを指定する  | -O3                |

#### 1.11.1.11 Fortran90 コンパイラに関する設定

| 変数名        | 説明   | 既定値                |
|------------|--|--------------------|
| F90        | Fortran90 コンパイラの起動コマンドを指定する  | mpif90             |
| F90FLAGS   | Fortran90 コンパイラに付与するオプションを指定する   | -DMPICH_IGNORE_F90 |
| F90LDFLAGS | Fortran90 リンカーに付与するオプションを指定する。Intel MKL を利用する場合には、そのリンクオプションを指定する。また、REVOCAP_Refiner を使用する場合で、Fortran90 プログラムのリンクに Fortran90 コンパイラを用いる場合には、C++ の標準ライブラリ (-lstl++ など) を指定する必要がある。 | なし                 |
| F90OPTS    | Fortran90 コンパイラに付与する最適化オプションなどを指定する  | -O2                |
| F90LINKER  | Fortran90 プログラムのリンク時に用いるコマンドを指定する。REVOCAP_Refiner を使用する場合で、Fortran90 プログラムのリンクに C++ コンパイラを用いる必要がある場合などに指定する（京コンピュータでは“mpiFCCpx -linkfortran”を指定する）。                              | \$(F90)            |

#### 1.11.1.12 UNIX コマンドに関する設定

| 変数名   | 説明  | 既定値      |
|-------|---|----------|
| MAKE  | make の起動コマンドを指定する。オプションが必要な場合は同時に指定する。          | make     |
| AR    | アーカイブの作成、変更などを行うコマンドを指定する。オプションが必要な場合は同時に指定する。  | ar ruv   |
| CP    | ファイルやディレクトリをコピーするコマンドを指定する。オプションが必要な場合は同時に指定する。 | cp -f    |
| RM    | ファイルやディレクトリを削除するコマンドを指定する。オプションが必要な場合は同時に指定する。  | rm -f    |
| MKDIR | ディレクトリを作成するコマンドを指定する。オプションが必要な場合は同時に指定する。       | mkdir -p |
| MV    | ファイルを移動するコマンドを指定する。オプションが必要な場合は同時に指定する。         | mv       |

### 1.11.2 Makefile.conf の設定例

```
# MPI
MPIDIR      =
MPIBINDIR   =
MPILIBDIR   =
MPIINCDir   =
MPILIBS     =

# for install option only
PREFIX      = $(HOME)/FrontISTR
BINDIR      = $(PREFIX)/bin
LIBDIR      = $(PREFIX)/lib
INCLUDEDIR  = $(PREFIX)/include

# Metis
METISDIR    = $(HOME)/Metis-4.0
METISLIBDIR = $(METISDIR)
METISINCDIR = $(METISDIR)/Lib

# ParMetis
PARMETISDIR    = $(HOME)/ParMetis-3.1
PARMETISLIBDIR = $(PARMETISDIR)
PARMETISINCDIR = $(PARMETISDIR)/ParMETISLib

# Refiner
REFINERDIR    = $(HOME)/REVOCAP_Refiner-1.1.0
REFINERINCDIR = $(REFINERDIR)/Refiner
REFINERLIBDIR = $(REFINERDIR)/lib/x86_64-linux

# Coupler
REVOCAPDIR    = $(HOME)/REVOCAP_Coupler-1.6.2
REVOCAPINCDIR = $(REVOCAPDIR)/librcap
REVOCAPLIBDIR = $(REVOCAPDIR)/librcap

# MUMPS
MUMPSDIR      = $(HOME)/MUMPS_4.10.0
MUMPSINCDIR   = $(MUMPSDIR)/include
MUMPSLIBDIR   = $(MUMPSDIR)/lib
```

```

# ML
MLDIR      = $(HOME)/trilinos/11.8.1/ml
MLINC_DIR  = $(MLDIR)/include
MLLIBDIR   = $(MLDIR)/lib

# C compiler settings
CC          = mpiicc
CFLAGS      =
LDFLAGS     = -lm
OPTFLAGS    = -O3
CLINKER     = mpiicc

# C++ compiler settings
CPP         = mpiicpc
CPPFLAGS    = -DMPICH_IGNORE_CXX_SEEK -I$(HOME)/include
CPPLDFLAGS  =
CPOPTFLAGS  = -O3

# Fortran compiler settings
F90         = mpiifort
F90FLAGS    =
F90LDFLAGS  = -lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core -liomp5
F90OPTFLAGS = -O2
F90LINKER   = mpiifort

```

### 1.11.3 京コンピュータおよび富士通 FX10 における注意

本バージョンでは、京コンピュータおよび富士通 FX10 向けのチューニングが行われていますが、これに伴い、利用する環境に応じてソースコードの一部を変更する必要があります。

変更するファイル：

hecmw1/src/solver/solver\_33/hecmw\_tuning\_fx.f90

変更内容：

ファイル内で定義されているパラメータ変数 **TotalSectorCacheSize** を

- 京コンピュータでは **12**
- FX10 では **24**

に設定する。

なお、初期状態では京コンピュータ向けの設定となっています。

## 1.12 付録

### 1.12.1 Makefile.conf の変数一覧

#### 1.12.1.1 MPI に関する設定

MPI 対応コンパイラーが自動参照している場合は、MPI に関する設定は不要である。

| 変数名       | 説明  | 既定値 |
|-----------|---|-----|
| MPIDIR    | MPI がインストールされているディレクトリのパスを指定する                    | なし  |
| MPIBINDIR | MPI の実行ファイル群がインストールされているディレクトリのパスを指定する            | なし  |
| MPIINCDIR | MPI のヘッダーファイル群がインストールされているディレクトリのパスを指定する          | .   |
| MPILIBDIR | MPI のライブラリ群がインストールされているディレクトリのパスを指定する             | .   |
| MPILIBS   | C および Fortran90 のオブジェクトファイルにリンクさせる MPI ライブラリを指定する | なし  |

#### 1.12.1.2 インストールディレクトリに関する設定

| 変数名        | 説明                                       | 既定値                |
|------------|--|--------------------|
| PREFIX     | 本ソフトウェアをインストールするディレクトリのパスを指定する           | \$(HOME)/FrontIST  |
| BINDIR     | 本ソフトウェアの実行ファイル群をインストールするディレクトリのパスを指定する   | \$(PREFIX)/bin     |
| INCLUDEDIR | 本ソフトウェアのヘッダーファイル群をインストールするディレクトリのパスを指定する | \$(PREFIX)/include |
| LIBDIR     | 本ソフトウェアのライブラリ群をインストールするディレクトリのパスを指定する    | \$(PREFIX)/lib     |

#### 1.12.1.3 METIS に関する設定

| 変数名         | 説明   | 既定値                  |
|-------------|--|----------------------|
| METISDIR    | METIS がインストールされているディレクトリのパスを指定する                         | \$(HOME)/metis       |
| METISINCDIR | METIS のヘッダーファイル群 (metis.h など) がインストールされているディレクトリのパスを指定する | \$(METISDIR)/include |
| METISLIBDIR | METIS のライブラリ (libmetis.a) がインストールされているディレクトリのパスを指定する     | \$(METISDIR)/lib     |

#### 1.12.1.4 ParMETIS に関する設定

| 変数名            | 説明   | 既定値                     |
|----------------|--|-------------------------|
| PARMETISDIR    | ParMETIS がインストールされているディレクトリのパスを指定する。                           | \$(HOME)/ParMetis       |
| PAEMETISINCDIR | ParMETIS のヘッダーファイル群 (parmetis.h など) がインストールされているディレクトリのパスを指定する | \$(PARMETISDIR)/include |
| PARMETISLIBDIR | ParMETIS のライブラリ (libparmetis.a) がインストールされているディレクトリのパスを指定する     | \$(PARMETISDIR)/lib     |

#### 1.12.1.5 REVOCAP\_Refiner に関する設定

| 変数名           | 説明   | 既定値                      |
|---------------|--|--------------------------|
| REFINERDIR    | REVOCAP_Refiner がインストールされているディレクトリのパスを指定する           | \$(HOME)/REVOCAP_Refiner |
| REFINERINCDIR | REVOCAP_Refiner のヘッダーファイル群がインストールされているディレクトリのパスを指定する | \$(REFINERDIR)/include   |
| REFINERLIBDIR | REVOCAP_Refiner のライブラリ群がインストールされているディレクトリのパスを指定する    | \$(REFINERDIR)/lib       |

#### 1.12.1.6 REVOCAP\_Coupler に関する設定

| 変数名           | 説明   | 既定値                      |
|---------------|--|--------------------------|
| REVOCAPDIR    | REVOCAP_Coupler がインストールされているディレクトリのパスを指定する           | \$(HOME)/REVOCAP_Coupler |
| REVOCAPINCDIR | REVOCAP_Coupler のヘッダーファイル群がインストールされているディレクトリのパスを指定する | \$(REVOCAPDIR)/include   |
| REVOCAPLIBDIR | REVOCAP_Coupler のライブラリ群がインストールされているディレクトリのパスを指定する    | \$(REVOCAPDIR)/lib       |

#### 1.12.1.7 MUMPS に関する設定

| 変数名         | 説明   | 既定値                  |
|-------------|--|----------------------|
| MUMPSDIR    | MUMPS がインストールされているディレクトリのパスを指定する           | \$(HOME)/MUMPS       |
| MUMPSINCDIR | MUMPS のヘッダーファイル群がインストールされているディレクトリのパスを指定する | \$(MUMPSDIR)/include |
| MUMPSLIBDIR | MUMPS のライブラリ群がインストールされているディレクトリのパスを指定する    | \$(MUMPSDIR)/lib     |

### 1.12.1.8 ML に関する設定

| 変数名      | 説明                                      | 既定値               |
|----------|---|-------------------|
| MLDIR    | ML がインストールされているディレクトリのパスを指定する           | \$(HOME)/trilinos |
| MLINCDIR | ML のヘッダーファイル群がインストールされているディレクトリのパスを指定する | \$(MLDIR)/include |
| MLLIBDIR | ML のライブラリ群がインストールされているディレクトリのパスを指定する    | \$(MLDIR)/lib     |

### 1.12.1.9 C コンパイラーに関する設定

| 変数名      | 説明  | 既定値    |
|----------|---|--------|
| CC       | C コンパイラーの起動コマンドを指定する  | mpicc  |
| CFLAGS   | C コンパイラーに付与するオプションを指定する   | なし     |
| LDFLAGS  | リンカーに付与するオプションを指定する。REVOCAP_Refiner を使用する場合で、C プログラムのリンクに C コンパイラーを用いる場合には、C++ の標準ライブラリ (-lstdc++ など) を指定する必要がある。 | -lm    |
| OPTFLAGS | C コンパイラーに付与する最適化オプションなどを指定する  | -O3    |
| CLINKER  | C プログラムのリンク時に用いるコマンドを指定する。REVOCAP_Refiner を使用する場合で、C プログラムのリンクに C++ コンパイラーを用いる必要がある場合などに指定する。                     | \$(CC) |

### 1.12.1.10 C++ コンパイラーに関する設定

| 変数名         | 説明   | 既定値                |
|-------------|--|--------------------|
| CPP         | C++ コンパイラーの起動コマンドを指定する   | mpic++             |
| CPPFLAGS    | C++ コンパイラーに付与するオプションを指定する。Boost ライブラリが C++ コンパイラーから自動参照されない場合、-I オプションにより、インクルードファイルが格納されているディレクトリを指定する。 | -DMPICH_IGNORE_CXX |
| CPPLDFLAGS  | リンカーに付与するオプションを指定する  | なし                 |
| CPPOPTFLAGS | C++ コンパイラーに付与する最適化オプションなどを指定する   | -O3                |

### 1.12.1.11 Fortran90 コンパイラーに関する設定

| 変数名         | 説明  | 既定値                     |
|-------------|---|-------------------------|
| F90         | Fortran90 コンパイラーの起動コマンドを指定する  | mpif90                  |
| F90FLAGS    | Fortran90 コンパイラーに付与するオプションを指定する   | -DMPICH_IGNORE_CXX_SEMI |
| F90LDFLAGS  | Fortran90 リンカーに付与するオプションを指定する。Intel MKL を利用する場合には、そのリンクオプションを指定する。また、REVOCAP_Refiner を使用する場合で、Fortran90 プログラムのリンクに Fortran90 コンパイラーを用いる場合には、C++ の標準ライブラリ (-lstlstdc++ など) を指定する必要がある。 | なし                      |
| F90OPTFLAGS | Fortran90 コンパイラーに付与する最適化オプションなどを指定する  | -O2                     |
| F90LINKER   | Fortran90 プログラムのリンク時に用いるコマンドを指定する。REVOCAP_Refiner を使用する場合で、Fortran90 プログラムのリンクに C++ コンパイラーを用いる必要がある場合などに指定する（京コンピュータでは“mpiFCCpx -linkfortran”を指定する）。                                  | \$(F90)                 |

#### 1.12.1.12 UNIX コマンドに関する設定

| 変数名   | 説明  | 既定値      |
|-------|---|----------|
| MAKE  | make の起動コマンドを指定する。オプションが必要な場合は同時に指定する。          | make     |
| AR    | アーカイブの作成、変更などを行うコマンドを指定する。オプションが必要な場合は同時に指定する。  | ar ruv   |
| CP    | ファイルやディレクトリをコピーするコマンドを指定する。オプションが必要な場合は同時に指定する。 | cp -f    |
| RM    | ファイルやディレクトリを削除するコマンドを指定する。オプションが必要な場合は同時に指定する。  | rm -f    |
| MKDIR | ディレクトリを作成するコマンドを指定する。オプションが必要な場合は同時に指定する。       | mkdir -p |
| MV    | ファイルを移動するコマンドを指定する。オプションが必要な場合は同時に指定する。         | mv       |

#### 1.12.2 Makefile.conf の設定例

```
# MPI
MPIDIR      =
MPIBINDIR   =
MPILIBDIR   =
MPIINCDIR   =
MPILIBS     =

# for install option only
PREFIX      = $(HOME)/FrontISTR
BINDIR      = $(PREFIX)/bin
LIBDIR      = $(PREFIX)/lib
INCLUDEDIR  = $(PREFIX)/include

# Metis
```



```

METISDIR      = $(HOME)/Metis-4.0
METISLIBDIR   = $(METISDIR)
METISINCDIR   = $(METISDIR)/Lib

# ParMetis
PARMETISDIR   = $(HOME)/ParMetis-3.1
PARMETISLIBDIR = $(PARMETISDIR)
PARMETISINCDIR = $(PARMETISDIR)/ParMETISLib

# Refiner
REFINERDIR    = $(HOME)/REVOCAP_Refiner-1.1.0
REFINERINCDIR = $(REFINERDIR)/Refiner
REFINERLIBDIR = $(REFINERDIR)/lib/x86_64-linux

# Coupler
REVOCAPDIR    = $(HOME)/REVOCAP_Coupler-1.6.2
REVOCAPINCDIR = $(REVOCAPDIR)/librcap
REVOCAPLIBDIR = $(REVOCAPDIR)/librcap

# MUMPS
MUMPSDIR      = $(HOME)/MUMPS_4.10.0
MUMPSINCDIR   = $(MUMPSDIR)/include
MUMPSLIBDIR   = $(MUMPSDIR)/lib

# ML
MLDIR         = $(HOME)/trilinos/11.8.1/ml
MLINCDIR      = $(MLDIR)/include
MLLIBDIR      = $(MLDIR)/lib

# C compiler settings
CC            = mpiicc
CFLAGS       =
LDFLAGS      = -lm
OPTFLAGS     = -O3
CLINKER      = mpiicc

# C++ compiler settings
CPP          = mpiicpc
CPPFLAGS     = -DMPICH_IGNORE_CXX_SEEK -I$(HOME)/include
CPPLDFLAGS   =
CPPOPTFLAGS  = -O3

```

```
# Fortran compiler settings
F90          = mpiifort
F90FLAGS     =
F90LDFLAGS   = -lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core -liomp5
F90OPTFLAGS  = -O2
F90LINKER    = mpiifort
```

### 1.12.3 京コンピュータおよび富士通 FX10 における注意

本バージョンでは、京コンピュータおよび富士通 FX10 向けのチューニングが行われていますが、これに伴い、利用する環境に応じてソースコードの一部を変更する必要があります。

変更するファイル：

```
hecmw1/src/solver/solver_33/hecmw_tuning_fx.f90
```

変更内容：

ファイル内で定義されているパラメータ変数 **TotalSectorCacheSize** を

- 京コンピュータでは **12**
- FX10 では **24**

に設定する。

なお、初期状態では京コンピュータ向けの設定となっています。

## 1.13 参考 CentOS7.6 へのインストール手順例 (cmake)

CentOS7.6 上へ本ソフトウェアと、それに必要な外部ライブラリの構築手順の例を示します。他の環境へのインストールの参考にしてください。

また、各ライブラリの詳細な構築方法は、それぞれのドキュメントを参考にしてください。

### 1.13.1 準備

最初に本ソフトウェアをコンパイルするのに必要なツールやパッケージをインストールしてください。

```
$ su
# yum group mark install "Development Tools"
# yum update
# yum install openmpi-devel cmake
# exit
```

次に MPI の環境設定を行います。コマンドライン上で

```
$ module purge
$ module load mpi/openmpi-x86_64
```

\$HOME/.bash\_profileに記述しておけば、次回ログイン時も設定が反映されます。

gcc/g++/gfortran および MPI のラッパーが正しくインストールされているか確認してください。

```
$ which gcc g++ gfortran mpicc mpic++ mpifort
/usr/bin/gcc
/usr/bin/g++
/usr/bin/gfortran
/usr/lib64/openmpi/bin/mpicc
/usr/lib64/openmpi/bin/mpic++
/usr/lib64/openmpi/bin/mpifort
```

### 1.13.2 ライブラリのインストール

本ソフトウェアに必要なライブラリをインストールします。作業ディレクトリは\$HOME/work、インストール先のディレクトリは\$HOME/localとします。

各ディレクトリを作成し、\$HOME/local/binを PATH 環境変数に追加します。

```
$ cd $HOME
$ mkdir work
$ mkdir -p local/bin local/lib local/include
$ export PATH=$HOME/local/bin:$PATH
```

#### 1.13.2.1 ダウンロード

以下のソフトウェアをダウンロードし、作業ディレクトリ\$HOME/workへ保存します。

| ソフトウェア名                         | ダウンロード先   |
|---------------------------------|---|
| REVOCAP_Refiner-1.1.04.tar.gz   | <a href="https://www.frontistr.com/">https://www.frontistr.com/</a>   |
| FrontISTR_V50.tar.gz            | <a href="https://www.frontistr.com/">https://www.frontistr.com/</a>   |
| OpenBLAS-0.2.20.tar.gz          | <a href="http://www.openblas.net/">http://www.openblas.net/</a>   |
| metis-5.1.0.tar.gz              | <a href="http://glaros.dtc.umn.edu/gkhome/metis/metis/download">http://glaros.dtc.umn.edu/gkhome/metis/metis/download</a> |
| scalapack-2.0.2.tgz             | <a href="http://www.netlib.org/scalapack/">http://www.netlib.org/scalapack/</a>   |
| MUMPS_5.1.2.tar.gz              | <a href="http://mumps.enseiht.fr/">http://mumps.enseiht.fr/</a>   |
| trilinos-12.14.1-Source.tar.bz2 | <a href="https://trilinos.org/download/">https://trilinos.org/download/</a>   |

#### 1.13.2.2 REVOCAP\_Refiner のコンパイル

```
$ cd $HOME/work
$ tar xvf REVOCAP_Refiner-1.1.04.tar.gz
$ cd REVOCAP_Refiner-1.1.04
$ make
$ cp lib/x86_64-linux/libRcapRefiner.a ~/local/lib
$ cp Refiner/rcapRefiner.h ~/local/include
```

### 1.13.2.3 OpenBLAS のコンパイル

```
$ cd $HOME/work
$ tar xvf OpenBLAS-0.2.20.tar.gz
$ make BINARY=64 NO_SHARED=1 USE_OPENMP=1
$ make PREFIX=~/local install
```

### 1.13.2.4 METIS のコンパイル

```
$ cd $HOME/work
$ tar xvf metis-5.1.0.tar.gz
$ cd metis-5.1.0
$ make config prefix=~/local cc=gcc openmp=1
$ make
$ make install
```

### 1.13.2.5 ScaLAPACK のコンパイル

```
$ cd $HOME/work
$ tar xvf scalapack-2.0.2.tgz
$ cd scalapack-2.0.2
$ mkdir build
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/local \
        -DCMAKE_EXE_LINKER_FLAGS="-fopenmp" \
        -DCMAKE_BUILD_TYPE="Release" \
        -DBLAS_LIBRARIES=$HOME/local/lib/libopenblas.a \
        -DLAPACK_LIBRARIES=$HOME/local/lib/libopenblas.a \
        ..
$ make
$ make install
```

### 1.13.2.6 MUMPS のコンパイル

```
$ cd $HOME/work
$ tar xvf MUMPS_5.1.2.tar.gz
$ cd MUMPS_5.1.2
$ cp Make.inc/Makefile.inc.generic Makefile.inc
```

コピーしたMakefile.incの以下の部分を書き換えます。

```
$ cp Make.inc/Makefile.inc.generic Makefile.inc
$ vi Makefile.inc
```

```
LMETISDIR = $(HOME)/local
IMETIS     = -I$(LMETISDIR)/include
LMETIS     = -L$(LMETISDIR)/lib -lmetis
```

```
ORDERINGSF = -Dmetis -Dpord
```

```
CC          = mpicc
FC          = mpifort
FL          = mpifort
```

```
LAPACK = -L$(HOME)/local/lib -lopenblas
```

```
SCALAP = -L$(HOME)/local/lib -lscalapack
```

```
INCPAR = -I/usr/include/openmpi-x86_64
```

```
LIBPAR = $(SCALAP) -L/usr/lib64/openmpi/lib -lmpi
```

```
LIBBLAS = -L$(HOME)/local/lib -lopenblas
```

```
OPTF     = -O -DBLR_MT -fopenmp
OPTC     = -O -I. -fopenmp
OPTL     = -O -fopenmp
```

書き換えが完了したら保存し make します。

```
$ make
$ cp lib/*.a $HOME/local/lib
$ cp include/*.h $HOME/local/include
```

### 1.13.2.7 Trilinos ML のコンパイル

```
$ cd $HOME/work
$ tar xvf trilinos-12.14.1-Source.tar.gz
$ cd trilinos-12.14.1-Source
$ mkdir build
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/local \
        -DCMAKE_C_COMPILER=mpicc \
        -DCMAKE_CXX_COMPILER=mpic++ \
        -DCMAKE_Fortran_COMPILER=mpifort \
        -DTPL_ENABLE_MPI=ON \
        -DTPL_ENABLE_LAPACK=ON \
        -DTPL_ENABLE_SCALAPACK=ON \
        -DTPL_ENABLE_METIS=ON \
        -DTPL_ENABLE_MUMPS=ON \
        -DTPL_MUMPS_INCLUDE_DIRS=$HOME/local/include \
        -DTrilinos_ENABLE_ML=ON \
        -DTrilinos_ENABLE_Zoltan=ON \
        -DTrilinos_ENABLE_OpenMP=ON \
        -DTrilinos_ENABLE_Amesos=ON \
        -DTrilinos_ENABLE_ALL_OPTIONAL_PACKAGES=OFF \
        -DBLAS_LIBRARY_DIRS=$HOME/local/lib \
        -DLAPACK_LIBRARY_DIRS=$HOME/local/lib" \
        -DSCALAPACK_LIBRARY_DIRS=$HOME/local/lib" \
        -DBLAS_LIBRARY_NAMES="openblas" \
        -DLAPACK_LIBRARY_NAMES="openblas" \
        -DSCALAPACK_LIBRARY_NAMES="scalapack" \
        ..
$ make
$ make install
```

### 1.13.3 FrontISTR のコンパイル

上記ライブラリのコンパイルが済んだら FrontISTR をコンパイルします。

```
$ cd $HOME/work
$ tar xvf FrontISTR_V50.tar.gz
$ cd FrontISTR
$ mkdir build
$ cd build
```

```
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/FrontISTR \
        -DWITH_ML=ON \
        -DBLAS_LIBRARIES=$HOME/local/lib/libopenblas.a \
        -DLAPACK_LIBRARIES=$HOME/local/lib/libopenblas.a \
        ..
```

#### 1.13.3.1 make の実行

make を実行します。

```
$ make
```

4 並列コンパイルをする場合、

```
$ make -j4
```

とします。並列コンパイルにより、コンパイル時間が短縮されます。

#### 1.13.3.2 make install の実行

make が完了したら、make install を実行し指定したディレクトリへインストールします。この例では \$(HOME)/FrontISTR/bin になります。

```
$ make install
```

#### 1.13.3.3 動作確認

本ソフトウェアに同梱されているチュートリアルを実行して、動作を確認します。

```
$ cd $HOME/work/FrontISTR/tutorial
$ cd 01_elastic_hinge
$ $HOME/FrontISTR/bin/fistr1
Step control not defined! Using default step=1
fstr_setup: OK
Start visualize PSF 1 at timestep 0

loading step=    1
sub_step= 1,    current_time=  0.0000E+00, time_inc=  0.1000E+01
loading_factor=    0.0000000    1.0000000
### 3x3 BLOCK CG, SSOR, 1
    1    1.903375E+00
    2    1.974378E+00
```

```

3      2.534627E+00
4      3.004045E+00
5      3.202633E+00
6      3.203864E+00

```

```
...
```

```
...
```

解析が終了すると以下の様に画面上に表示されます。

```
...
```

```
...
```

```

2966      1.143085E-08
2967      1.078272E-08
2968      1.004759E-08
2969      9.372882E-09

```

```
#### Relative residual = 9.39169E-09
```

```
### summary of linear solver
```

```

2969 iterations      9.391687E-09
set-up time         : 4.108060E-01
solver time         : 6.506822E+01
solver/comm time    : 4.342469E-01
solver/matvec       : 1.923199E+01
solver/precond      : 2.688405E+01
solver/1 iter       : 2.191587E-02
work ratio (%)      : 9.933263E+01

```

```
Start visualize PSF 1 at timestep 1
```

```
### FSTR_SOLVE_NLGEOM FINISHED!
```

---



---

```

TOTAL TIME (sec) : 74.93
pre (sec) : 1.86
solve (sec) : 73.07

```

---



---

```
FrontISTR Completed !!
```

## 1.14 参考 CentOS7.6 へのインストール手順例 (Makefile.conf)

CentOS7.6 上へ本ソフトウェアと、それに必要な外部ライブラリの構築手順の例を示します。他の環境へのインストールの参考にしてください。



また、各ライブラリの詳細な構築方法は、それぞれのドキュメントを参考にしてください。

#### 1.14.1 準備

最初に本ソフトウェアをコンパイルするのに必要なツールやパッケージをインストールしてください。

```
$ su
# yum group mark install "Development Tools"
# yum update
# yum install openmpi-devel cmake
# exit
```

次に MPI の環境設定を行います。コマンドライン上で

```
$ module purge
$ module local mpi/openmpi-x86_64
```

\$HOME/.bash\_profileに記述しておけば、次回ログイン時も設定が反映されます。

gcc/g++/gfortran および MPI のラッパーが正しくインストールされているか確認してください。

```
$ which gcc g++ gfortran mpicc mpic++ mpifort
/usr/bin/gcc
/usr/bin/g++
/usr/bin/gfortran
/usr/lib64/openmpi/bin/mpicc
/usr/lib64/openmpi/bin/mpic++
/usr/lib64/openmpi/bin/mpifort
```

#### 1.14.2 ライブラリのインストール

本ソフトウェアに必要なライブラリをインストールします。作業ディレクトリは\$HOME/work、インストール先のディレクトリは\$HOME/localとします。

各ディレクトリを作成し、\$HOME/local/binを PATH 環境変数に追加します。

```
$ cd $HOME
$ mkdir work
$ mkdir -p local/bin local/lib local/include
$ export PATH=$HOME/local/bin:$PATH
```

##### 1.14.2.1 ダウンロード

以下のソフトウェアをダウンロードし、作業ディレクトリ\$HOME/workへ保存します。

| ソフトウェア名                         | ダウンロード先   |
|---------------------------------|---|
| REVOCAP_Refiner-1.1.04.tar.gz   | <a href="https://www.frontistr.com/">https://www.frontistr.com/</a>   |
| FrontISTR_V50.tar.gz            | <a href="https://www.frontistr.com/">https://www.frontistr.com/</a>   |
| OpenBLAS-0.2.20.tar.gz          | <a href="http://www.openblas.net/">http://www.openblas.net/</a>   |
| metis-5.1.0.tar.gz              | <a href="http://glaros.dtc.umn.edu/gkhome/metis/metis/download">http://glaros.dtc.umn.edu/gkhome/metis/metis/download</a> |
| scalapack-2.0.2.tgz             | <a href="http://www.netlib.org/scalapack/">http://www.netlib.org/scalapack/</a>   |
| MUMPS_5.1.2.tar.gz              | <a href="http://mumps.enseiht.fr/">http://mumps.enseiht.fr/</a>   |
| trilinos-12.14.1-Source.tar.bz2 | <a href="https://trilinos.org/download/">https://trilinos.org/download/</a>   |

#### 1.14.2.2 REVOCAP\_Refiner のコンパイル

```
$ cd $HOME/work
$ tar xvf REVOCAP_Refiner-1.1.04.tar.gz
$ cd REVOCAP_Refiner-1.1.04
$ make
$ cp lib/x86_64-linux/libRcapRefiner.a ~/local/lib
$ cp Refiner/rcapRefiner.h ~/local/include
```

#### 1.14.2.3 OpenBLAS のコンパイル

```
$ cd $HOME/work
$ tar xvf OpenBLAS-0.2.20.tar.gz
$ make BINARY=64 NO_SHARED=1 USE_OPENMP=1
$ make PREFIX=~/local install
```

#### 1.14.2.4 METIS のコンパイル

```
$ cd $HOME/work
$ tar xvf metis-5.1.0.tar.gz
$ cd metis-5.1.0
$ make config prefix=~/local cc=gcc openmp=1
$ make
$ make install
```

#### 1.14.2.5 ScaLAPACK のコンパイル

```

$ cd $HOME/work
$ tar xvf scalapack-2.0.2.tgz
$ cd scalapack-2.0.2
$ mkdir build
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/local \
        -DCMAKE_EXE_LINKER_FLAGS="-fopenmp" \
        -DBLAS_LIBRARIES=$HOME/local/lib/libopenblas.a \
        -DLAPACK_LIBRARIES=$HOME/local/lib/libopenblas.a \
        ..
$ make
$ make install

```

#### 1.14.2.6 MUMPS のコンパイル

```

$ cd $HOME/work
$ tar xvf MUMPS_5.1.2.tar.gz
$ cd MUMPS_5.1.2
$ cp Make.inc/Makefile.inc.generic Makefile.inc

```

コピーしたMakefile.incの以下の部分を書き換えます。

```

$ vi Makefile.inc
$ cp Make.inc/Makefile.inc.generic Makefile.inc
$ vi Makefile.inc
LMETISDIR = $(HOME)/local
IMETIS     = -I$(LMETISDIR)/include
LMETIS     = -L$(LMETISDIR)/lib -lmetis

```

```

ORDERINGSF = -Dmetis -Dpardiso

```

```

CC          = mpicc
FC          = mpifort
FL          = mpifort

```

```

LAPACK = -L$(HOME)/local/lib -lopenblas

```

```

SCALAP = -L$(HOME)/local/lib -lscalapack

```

```

INCPAR = -I/usr/include/openmpi-x86_64

```

```

LIBPAR = $(SCALAP) -L/usr/lib64/openmpi/lib -lmpi

```

```
LIBBLAS = -L$(HOME)/local/lib -lopenblas
```

```
OPTF    = -O -DBLR_MT -fopenmp
```

```
OPTC    = -O -I. -fopenmp
```

```
OPTL    = -O -fopenmp
```

書き換えが完了したら保存し make します。

```
$ make
```

```
$ cp lib/*.a $HOME/local/lib
```

```
$ cp include/*.h $HOME/local/include
```

#### 1.14.2.7 Trilinos ML のコンパイル

```
$ cd $HOME/work
```

```
$ tar xvf trilinos-12.14.1-Source.tar.gz
```

```
$ cd trilinos-12.14.1-Source
```

```
$ mkdir build
```

```
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/local \
  -DCMAKE_C_COMPILER=mpicc \
  -DCMAKE_CXX_COMPILER=mpic++ \
  -DCMAKE_Fortran_COMPILER=mpifort \
  -DTPL_ENABLE_MPI=ON \
  -DTPL_ENABLE_LAPACK=ON \
  -DTPL_ENABLE_SCALAPACK=ON \
  -DTPL_ENABLE_METIS=ON \
  -DTPL_ENABLE_MUMPS=ON \
  -DTrilinos_ENABLE_ML=ON \
  -DTrilinos_ENABLE_Zoltan=ON \
  -DTrilinos_ENABLE_OpenMP=ON \
  -DTrilinos_ENABLE_Amesos=ON \
  -DTrilinos_ENABLE_ALL_OPTIONAL_PACKAGES=OFF \
  -DBLAS_LIBRARY_DIRS=$HOME/local/lib \
  -DLAPACK_LIBRARY_DIRS=$HOME/local/lib" \
  -DSCALAPACK_LIBRARY_DIRS=$HOME/local/lib" \
  -DBLAS_LIBRARY_NAMES="openblas" \
  -DLAPACK_LIBRARY_NAMES="openblas" \
  -DSCALAPACK_LIBRARY_NAMES="scalapack" \
```

```
..
```

```
$ make
```

```
$ make install
```

### 1.14.3 FrontISTR のコンパイル

上記ライブラリのコンパイルが済んだら FrontISTR をコンパイルします。

```
$ cd $HOME/work
$ tar xvf FrontISTR_V50.tar.gz
$ cd FrontISTR
```

#### 1.14.3.1 Makefile.conf の編集

雛形をコピーして、環境に合わせた内容に編集します。この例では、以下の様に編集します。

```
$ cp Makefile.conf.org Makefile.conf
$ vi Makefile.conf

#####
#                                                                    #
#      Setup Configuration File for FrontISTR                      #
#                                                                    #
#####

# MPI
MPIDIR      = /usr/lib64/openmpi
MPIBINDIR   = $(MPIDIR)/bin
MPILIBDIR   = $(MPIDIR)/lib
MPIINCDir   = /usr/include/openmpi-x86_64
MPILIBS     = -lmpi -lmpi_cxx -lmpi_mpifh

# for install option only
PREFIX      = $(HOME)/FrontISTR
BINDIR      = $(PREFIX)/bin
LIBDIR      = $(PREFIX)/lib
INCLUDEDIR  = $(PREFIX)/include

# Metis
METISDIR    = $(HOME)/local
METISLIBDIR  = $(METISDIR)/lib
METISINCDir  = $(METISDIR)/include
HECMW_METIS_VER= 5
```

```

# ParMetis
PARMETISDIR      = $(HOME)/local
PARMETISLIBDIR   = $(PARMETISDIR)/lib
PARMETISINCDIR   = $(PARMETISDIR)/include

# Refiner
REFINERDIR       = $(HOME)/local
REFINERINCDIR    = $(REFINERDIR)/include
REFINERLIBDIR    = $(REFINERDIR)/lib

# Coupler
REVOCAPDIR       = $(HOME)/local
REVOCAPINCDIR    = $(REVOCAPDIR)/include
REVOCAPLIBDIR    = $(REVOCAPDIR)/lib

# MUMPS
MUMPSDIR         = $(HOME)/local
MUMPSINCDIR      = $(MUMPSDIR)/include
MUMPSLIBDIR      = $(MUMPSDIR)/lib
MUMPSLIBS        = -ldmumps -lmumps_common -lpord -L$(HOME)/local/lib -lscalapack

# MKL PARDISO
MKLDIR           = $(HOME)/
MKLINCDIR        = $(MKLDIR)/include
MKLLIBDIR        = $(MKLDIR)/lib

# ML
MLDIR            = $(HOME)/local
MLINCDIR         = $(MLDIR)/include
MLLIBDIR         = $(MLDIR)/lib
MLLIBS           = -lml -lamesos -ltrilinosss -lzoltan -lepetra -lteuchosremainder -lteuchosnu

# C compiler settings
CC               = mpicc -fopenmp
CFLAGS           =
LDFLAGS          = -lstdc++ -lm
OPTFLAGS         = -O3

# C++ compiler settings
CPP              = mpic++ -fopenmp
CPPFLAGS         =

```

```

CPPLDFLAGS      =
CPPOPTFLAGS     = -O3

# Fortran compiler settings
F90              = mpif90 -fopenmp
F90FLAGS        =
F90LDLDFLAGS     = -lstdc++ -L$(HOME)/local/lib -lopenblas
F90OPTFLAGS      = -O2
F90FPP          = -cpp
F90LINKER       = mpif90 -fopenmp

MAKE            = make
AR              = ar ruv
MV              = mv -f
CP              = cp -f
RM              = rm -f
MKDIR           = mkdir -p

```

#### 1.14.3.2 setup.sh の実行

編集が完了したら、setup.sh を実行します。

```

$ ./setup.sh -p --with-tools --with-refiner \
               --with-metis --with-mumps --with-lapack --with-ml

```

#### 1.14.3.3 make の実行

make を実行します。

```

$ make

```

#### 1.14.3.4 make install の実行

make が完了したら、make install を実行し Makefile.conf で指定したディレクトリへインストールします。この例では \$(HOME)/FrontISTR/bin にインストールされます。

```

$ make install

```

#### 1.14.3.5 動作確認

本ソフトウェアに同梱されているチュートリアルを実行して、動作を確認します。

```
$ cd $HOME/work/FrontISTR/tutorial
$ cd 01_elastic_hinge
$ $HOME/FrontISTR/bin/fistr1
Step control not defined! Using default step=1
fstr_setup: OK
Start visualize PSF 1 at timestep 0

loading step=      1
sub_step= 1,      current_time=  0.0000E+00, time_inc=  0.1000E+01
loading_factor=      0.0000000    1.0000000
### 3x3 BLOCK CG, SSOR, 1
      1      1.903375E+00
      2      1.974378E+00
      3      2.534627E+00
      4      3.004045E+00
      5      3.202633E+00
      6      3.203864E+00
...
...
```

解析が終了すると以下の様に画面上に表示されます。

```
...
...
2966      1.143085E-08
2967      1.078272E-08
2968      1.004759E-08
2969      9.372882E-09
### Relative residual = 9.39169E-09

### summary of linear solver
      2969 iterations      9.391687E-09
set-up time      :      4.108060E-01
solver time      :      6.506822E+01
solver/comm time :      4.342469E-01
solver/matvec     :      1.923199E+01
solver/precond    :      2.688405E+01
```



```
solver/1 iter      :      2.191587E-02
work ratio (%)     :      9.933263E+01
```

```
Start visualize PSF 1 at timestep 1
### FSTR Solve NLGEOM FINISHED!
```

---

---

```
TOTAL TIME (sec) :      74.93
      pre (sec) :       1.86
      solve (sec) :      73.07
```

---

---

```
FrontISTR Completed !!
```

## 1.15 参考 Ubuntu18.04 へのインストール手順例 (cmake)

Ubuntu18.04 上へ本ソフトウェアと、それに必要な外部ライブラリの構築手順の例を示します。他の環境へのインストールの参考にしてください。

また、各ライブラリの詳細な構築方法は、それぞれのドキュメントを参考にしてください。

### 1.15.1 準備

最初に本ソフトウェアをコンパイルするのに必要なツールやパッケージをインストールしてください。

```
$ sudo apt install build-essential gfortran cmake openmpi-bin libopenmpi-dev
```

gcc/g++/gfortran および MPI のラッパーが正しくインストールされているか確認してください。

```
$ which gcc g++ gfortran mpicc mpic++ mpifort
/usr/bin/gcc
/usr/bin/g++
/usr/bin/gfortran
/usr/bin/mpicc
/usr/bin/mpic++
/usr/bin/mpifort
```

### 1.15.2 ライブラリのインストール

本ソフトウェアに必要なライブラリをインストールします。作業ディレクトリは\$HOME/work、インストール先のディレクトリは\$HOME/localとします。

各ディレクトリを作成し、\$HOME/local/binを PATH 環境変数に追加します。

```
$ cd $HOME
```

```
$ mkdir work
$ mkdir -p local/bin local/lib local/include
$ export PATH=$HOME/local/bin:$PATH
```

### 1.15.2.1 ダウンロード

以下のソフトウェアをダウンロードし、作業ディレクトリ\$HOME/workへ保存します。

| ソフトウェア名                         | ダウンロード先   |
|---------------------------------|---|
| REVOCAP_Refiner-1.1.04.tar.gz   | <a href="http://www.frontistr.com/">http://www.frontistr.com/</a>   |
| FrontISTR_V50.tar.gz            | <a href="https://www.frontistr.com/">https://www.frontistr.com/</a>   |
| OpenBLAS-0.2.20.tar.gz          | <a href="http://www.openblas.net/">http://www.openblas.net/</a>   |
| metis-5.1.0.tar.gz              | <a href="http://glaros.dtc.umn.edu/gkhome/metis/metis/download">http://glaros.dtc.umn.edu/gkhome/metis/metis/download</a> |
| scalapack-2.0.2.tgz             | <a href="http://www.netlib.org/scalapack/">http://www.netlib.org/scalapack/</a>   |
| MUMPS_5.1.2.tar.gz              | <a href="http://mumps.enseiht.fr/">http://mumps.enseiht.fr/</a>   |
| trilinos-12.14.1-Source.tar.bz2 | <a href="https://trilinos.org/download/">https://trilinos.org/download/</a>   |

### 1.15.2.2 REVOCAP\_Refiner のコンパイル

```
$ cd $HOME/work
$ tar xvf REVOCAP_Refiner-1.1.04.tar.gz
$ cd REVOCAP_Refiner-1.1.04
$ make
$ cp lib/x86_64-linux/libRcapRefiner.a $HOME/local/lib
$ cp Refiner/rcapRefiner.h $HOME/local/include
```

### 1.15.2.3 OpenBLAS のコンパイル

```
$ cd $HOME/work
$ tar xvf OpenBLAS-0.2.20.tar.gz
$ make BINARY=64 NO_SHARED=1 USE_OPENMP=1
$ make PREFIX=$HOME/local install
```

### 1.15.2.4 METIS のコンパイル

```
$ cd $HOME/work
$ tar xvf metis-5.1.0.tar.gz
```

```
$ cd metis-5.1.0
$ make config prefix=$HOME/local cc=gcc openmp=1
$ make
$ make install
```

#### 1.15.2.5 ScaLAPACK のコンパイル

```
$ cd $HOME/work
$ tar xvf scalapack-2.0.2.tgz
$ cd scalapack-2.0.2
$ mkdir build
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/local \
        -DCMAKE_EXE_LINKER_FLAGS="-fopenmp" \
        -DBLAS_LIBRARIES=$HOME/local/lib/libopenblas.a \
        -DLAPACK_LIBRARIES=$HOME/local/lib/libopenblas.a \
        ..
$ make
$ make install
```

#### 1.15.2.6 MUMPS のコンパイル

```
$ cd $HOME/work
$ tar xvf MUMPS_5.1.2.tar.gz
$ cd MUMPS_5.1.2
$ cp Make.inc/Makefile.inc.generic Makefile.inc
```

コピーしたMakefile.incの以下の部分を書き換えます。

```
$ vi Makefile.inc
$ cp Make.inc/Makefile.inc.generic Makefile.inc
$ vi Makefile.inc
LMETISDIR = $(HOME)/local
IMETIS     = -I$(LMETISDIR)/include
LMETIS     = -L$(LMETISDIR)/lib -lmetis
```

```
ORDERINGSF = -Dmetis -Dpardiso
```

```
CC          = mpicc
FC          = mpifort
FL          = mpifort
```

```
LAPACK = -L$(HOME)/local/lib -lopenblas
```

```
SCALAP = -L$(HOME)/local/lib -lscalapack
```

```
INCPAR =
```

```
LIBPAR = $(SCALAP)
```

```
LIBBLAS = -L$(HOME)/local/lib -lopenblas
```

```
OPTF = -O -DBLR_MT -fopenmp
```

```
OPTC = -O -I. -fopenmp
```

```
OPTL = -O -fopenmp
```

書き換えが完了したら保存し make します。

```
$ make
```

```
$ cp lib/*.a $HOME/local/lib
```

```
$ cp include/*.h $HOME/local/include
```

#### 1.15.2.7 Trilinos ML のコンパイル

```
$ cd $HOME/work
```

```
$ tar xvf trilinos-12.14.1-Source.tar.gz
```

```
$ cd trilinos-12.14.1-Source
```

```
$ mkdir build
```

```
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/local \
  -DCMAKE_C_COMPILER=mpicc \
  -DCMAKE_CXX_COMPILER=mpic++ \
  -DCMAKE_Fortran_COMPILER=mpifort \
  -DTPL_ENABLE_MPI=ON \
  -DTPL_ENABLE_LAPACK=ON \
  -DTPL_ENABLE_SCALAPACK=ON \
  -DTPL_ENABLE_METIS=ON \
  -DTPL_ENABLE_MUMPS=ON \
  -DTrilinos_ENABLE_ML=ON \
  -DTrilinos_ENABLE_Zoltan=ON \
  -DTrilinos_ENABLE_OpenMP=ON \
  -DTrilinos_ENABLE_Amesos=ON \
  -DTrilinos_ENABLE_ALL_OPTIONAL_PACKAGES=OFF \
```

```

-DBLAS_LIBRARY_DIRS=$HOME/local/lib \
-DLAPACK_LIBRARY_DIRS=$HOME/local/lib" \
-DSCALAPACK_LIBRARY_DIRS=$HOME/local/lib" \
-DBLAS_LIBRARY_NAMES="openblas" \
-DLAPACK_LIBRARY_NAMES="openblas" \
-DSCALAPACK_LIBRARY_NAMES="scalapack" \
..
$ make
$ make install

```

### 1.15.3 FrontISTR のコンパイル

上記ライブラリのコンパイルが済んだら FrontISTR をコンパイルします。

```

$ cd $HOME/work/FrontISTR
$ mkdir build
$ cd build
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/FrontISTR \
-DWITH_ML=ON \
-DBLAS_LIBRARIES=$HOME/local/lib/libopenblas.a \
-DLAPACK_LIBRARIES=$HOME/local/lib/libopenblas.a \
..

```

#### 1.15.3.1 make の実行

make を実行します。

```
$ make
```

4 並列コンパイルをする場合、

```
$ make -j4
```

とします。並列コンパイルにより、コンパイル時間が短縮されます。

#### 1.15.3.2 make install の実行

make が完了したら、make install を実行し Makefile.conf で指定したディレクトリへインストールします。この例では \$(HOME)/FrontISTR/bin になります。

```
$ make install
```

### 1.15.3.3 動作確認

本ソフトウェアに同梱されているチュートリアルを実行して、動作を確認します。

```
$ cd $HOME/work/FrontISTR/tutorial
$ cd 01_elastic_hinge
$ $HOME/FrontISTR/bin/fistr1
Step control not defined! Using default step=1
fstr_setup: OK
Start visualize PSF 1 at timestep 0

loading step=      1
sub_step= 1,      current_time=  0.0000E+00, time_inc=  0.1000E+01
loading_factor=      0.0000000    1.0000000
### 3x3 BLOCK CG, SSOR, 1
      1      1.903375E+00
      2      1.974378E+00
      3      2.534627E+00
      4      3.004045E+00
      5      3.202633E+00
      6      3.203864E+00
...
...
```

解析が終了すると以下の様に画面上に表示されます。

```
...
...
2966      1.143085E-08
2967      1.078272E-08
2968      1.004759E-08
2969      9.372882E-09
### Relative residual = 9.39169E-09

### summary of linear solver
      2969 iterations      9.391687E-09
set-up time      :      4.108060E-01
solver time      :      6.506822E+01
solver/comm time :      4.342469E-01
solver/matvec     :      1.923199E+01
solver/precond    :      2.688405E+01
```

```
solver/1 iter      :      2.191587E-02
work ratio (%)     :      9.933263E+01
```

```
Start visualize PSF 1 at timestep 1
### FSTR Solve NLGEOM FINISHED!
```

---

---

```
TOTAL TIME (sec) :      74.93
      pre (sec) :       1.86
      solve (sec) :      73.07
```

---

---

```
FrontISTR Completed !!
```

## 1.16 参考 Ubuntu18.04 へのインストール手順例 (Makefile.conf)

Ubuntu18.04 上へ本ソフトウェアと、それに必要な外部ライブラリの構築手順の例を示します。他の環境へのインストールの参考にしてください。

また、各ライブラリの詳細な構築方法は、それぞれのドキュメントを参考にしてください。

### 1.16.1 準備

最初に本ソフトウェアをコンパイルするのに必要なツールやパッケージをインストールしてください。

```
$ sudo apt install build-essential gfortran cmake openmpi-bin libopenmpi-dev
```

gcc/g++/gfortran および MPI のラッパーが正しくインストールされているか確認してください。

```
$ which gcc g++ gfortran mpicc mpic++ mpifort
/usr/bin/gcc
/usr/bin/g++
/usr/bin/gfortran
/usr/bin/mpicc
/usr/bin/mpic++
/usr/bin/mpifort
```

### 1.16.2 ライブラリのインストール

本ソフトウェアに必要なライブラリをインストールします。作業ディレクトリは\$HOME/work、インストール先のディレクトリは\$HOME/localとします。

各ディレクトリを作成し、\$HOME/local/binを PATH 環境変数に追加します。

```
$ cd $HOME
```

```
$ mkdir work
$ mkdir -p local/bin local/lib local/include
$ export PATH=$HOME/local/bin:$PATH
```

### 1.16.2.1 ダウンロード

以下のソフトウェアをダウンロードし、作業ディレクトリ\$HOME/workへ保存します。

| ソフトウェア名                         | ダウンロード先   |
|---------------------------------|---|
| REVOCAP_Refiner-1.1.04.tar.gz   | <a href="https://www.frontistr.com/">https://www.frontistr.com/</a>   |
| FrontISTR_V50.tar.gz            | <a href="http://www.frontistr.com/">http://www.frontistr.com/</a>   |
| OpenBLAS-0.2.20.tar.gz          | <a href="http://www.openblas.net/">http://www.openblas.net/</a>   |
| metis-5.1.0.tar.gz              | <a href="http://glaros.dtc.umn.edu/gkhome/metis/metis/download">http://glaros.dtc.umn.edu/gkhome/metis/metis/download</a> |
| scalapack-2.0.2.tgz             | <a href="http://www.netlib.org/scalapack/">http://www.netlib.org/scalapack/</a>   |
| MUMPS_5.1.2.tar.gz              | <a href="http://mumps.enseiht.fr/">http://mumps.enseiht.fr/</a>   |
| trilinos-12.14.1-Source.tar.bz2 | <a href="https://trilinos.org/download/">https://trilinos.org/download/</a>   |

### 1.16.2.2 REVOCAP\_Refiner のコンパイル

```
$ cd $HOME/work
$ tar xvf REVOCAP_Refiner-1.1.04.tar.gz
$ cd REVOCAP_Refiner-1.1.04
$ make
$ cp lib/x86_64-linux/libRcapRefiner.a $HOME/local/lib
$ cp Refiner/rcapRefiner.h $HOME/local/include
```

### 1.16.2.3 OpenBLAS のコンパイル

```
$ cd $HOME/work
$ tar xvf OpenBLAS-0.2.20.tar.gz
$ make BINARY=64 NO_SHARED=1 USE_OPENMP=1
$ make PREFIX=$HOME/local install
```

### 1.16.2.4 METIS のコンパイル

```
$ cd $HOME/work
$ tar xvf metis-5.1.0.tar.gz
```



```
$ cd metis-5.1.0
$ make config prefix=$HOME/local cc=gcc openmp=1
$ make
$ make install
```

#### 1.16.2.5 ScaLAPACK のコンパイル

```
$ cd $HOME/work
$ tar xvf scalapack-2.0.2.tgz
$ cd scalapack-2.0.2
$ mkdir build
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/local \
        -DCMAKE_EXE_LINKER_FLAGS="-fopenmp" \
        -DBLAS_LIBRARIES=$HOME/local/lib/libopenblas.a \
        -DLAPACK_LIBRARIES=$HOME/local/lib/libopenblas.a \
        ..
$ make
$ make install
```

#### 1.16.2.6 MUMPS のコンパイル

```
$ cd $HOME/work
$ tar xvf MUMPS_5.1.2.tar.gz
$ cd MUMPS_5.1.2
$ cp Make.inc/Makefile.inc.generic Makefile.inc
```

コピーしたMakefile.incの以下の部分を書き換えます。

```
$ vi Makefile.inc
$ cp Make.inc/Makefile.inc.generic Makefile.inc
$ vi Makefile.inc
LMETISDIR = $(HOME)/local
IMETIS     = -I$(LMETISDIR)/include
LMETIS     = -L$(LMETISDIR)/lib -lmetis
```

```
ORDERINGSF = -Dmetis -Dpardiso
```

```
CC          = mpicc
FC          = mpifort
FL          = mpifort
```

```
LAPACK = -L$(HOME)/local/lib -lopenblas
```

```
SCALAP = -L$(HOME)/local/lib -lscalapack
```

```
INCPAR =
```

```
LIBPAR = $(SCALAP)
```

```
LIBBLAS = -L$(HOME)/local/lib -lopenblas
```

```
OPTF    = -O -DBLR_ML -fopenmp
```

```
OPTC    = -O -I. -fopenmp
```

```
OPTL    = -O -fopenmp
```

書き換えが完了したら保存し make します。

```
$ make
```

```
$ cp lib/*.a $HOME/local/lib
```

```
$ cp include/*.h $HOME/local/include
```

#### 1.16.2.7 Trilinos ML のコンパイル

```
$ cd $HOME/work
```

```
$ tar xvf trilinos-12.14.1-Source.tar.gz
```

```
$ cd trilinos-12.14.1-Source
```

```
$ mkdir build
```

```
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/local \
  -DCMAKE_C_COMPILER=mpicc \
  -DCMAKE_CXX_COMPILER=mpic++ \
  -DCMAKE_Fortran_COMPILER=mpifort \
  -DTPL_ENABLE_MPI=ON \
  -DTPL_ENABLE_LAPACK=ON \
  -DTPL_ENABLE_SCALAPACK=ON \
  -DTPL_ENABLE_METIS=ON \
  -DTPL_ENABLE_MUMPS=ON \
  -DTrilinos_ENABLE_ML=ON \
  -DTrilinos_ENABLE_Zoltan=ON \
  -DTrilinos_ENABLE_OpenMP=ON \
  -DTrilinos_ENABLE_Amesos=ON \
  -DTrilinos_ENABLE_ALL_OPTIONAL_PACKAGES=OFF \
```

```

-DBLAS_LIBRARY_DIRS=$HOME/local/lib \
-DLAPACK_LIBRARY_DIRS=$HOME/local/lib" \
-DSCALAPACK_LIBRARY_DIRS=$HOME/local/lib" \
-DBLAS_LIBRARY_NAMES="openblas" \
-DLAPACK_LIBRARY_NAMES="openblas" \
-DSCALAPACK_LIBRARY_NAMES="scalapack" \
..

```

```
$ make
```

```
$ make install
```

### 1.16.3 FrontISTR のコンパイル

上記ライブラリのコンパイルが済んだら FrontISTR をコンパイルします。

```
$ cd $HOME/work
```

```
$ tar xvf FrontISTR_V50.tar.gz
```

```
$ cd FrontISTR
```

#### 1.16.3.1 Makefile.conf の編集

雛形をコピーして、環境に合わせた内容に編集します。この例では、以下の様に編集します。

```
$ cp Makefile.conf.org Makefile.conf
```

```
$ vi Makefile.conf
```

```
#####
#                                                                    #
#      Setup Configuration File for FrontISTR      #
#                                                                    #
#####
```

```
# MPI
```

```
MPIDIR      = /usr/lib/x86_64-linux-gnu/openmpi
```

```
MPIBINDIR   = /usr/bin
```

```
MPILIBDIR   = $(MPIDIR)/lib
```

```
MPIINC_DIR  = $(MPIDIR)/include
```

```
MPILIBS     = -lmpi -lmpi_cxx -lmpi_mpifh
```

```
# for install option only
```

```
PREFIX      = $(HOME)/FrontISTR
```

```
BINDIR      = $(PREFIX)/bin
```

```
LIBDIR      = $(PREFIX)/lib
```

```
INCLUDEDIR      = $(PREFIX)/include
```

```
# Metis
```

```
METISDIR        = $(HOME)/local
```

```
METISLIBDIR     = $(METISDIR)/lib
```

```
METISINCDIR     = $(METISDIR)/include
```

```
HECMW_METIS_VER= 5
```

```
# ParMetis
```

```
PARMETISDIR     = $(HOME)/local
```

```
PARMETISLIBDIR  = $(PARMETISDIR)/lib
```

```
PARMETISINCDIR = $(PARMETISDIR)/include
```

```
# Refiner
```

```
REFINERDIR      = $(HOME)/local
```

```
REFINERINCDIR  = $(REFINERDIR)/include
```

```
REFINERLIBDIR  = $(REFINERDIR)/lib
```

```
# Coupler
```

```
REVOCAPDIR     = $(HOME)/local
```

```
REVOCAPINCDIR  = $(REVOCAPDIR)/include
```

```
REVOCAPLIBDIR  = $(REVOCAPDIR)/lib
```

```
# MUMPS
```

```
MUMPSDIR       = $(HOME)/local
```

```
MUMPSINCDIR    = $(MUMPSDIR)/include
```

```
MUMPSLIBDIR    = $(MUMPSDIR)/lib
```

```
MUMPSLIBS      = -ldmumps -lmumps_common -lpord -L$(HOME)/local/lib -lscalapack
```

```
# MKL PARDISO
```

```
MKLDIR         = $(HOME)/
```

```
MKLINCDIR      = $(MKLDIR)/include
```

```
MKLLIBDIR      = $(MKLDIR)/lib
```

```
# ML
```

```
MLDIR          = $(HOME)/local
```

```
MLINCDIR       = $(MLDIR)/include
```

```
MLLIBDIR       = $(MLDIR)/lib
```

```
MLLIBS         = -lml -lamesos -ltrilinosss -lzoltan -lepetra -lteuchosremainder -lteuchosnu
```

```
# C compiler settings
```

```

CC                = mpicc -fopenmp
CFLAGS            =
LDLDFLAGS         = -lstdc++ -lm
OPTFLAGS          = -O3

# C++ compiler settings
CPP              = mpic++ -fopenmp
CPPFLAGS         =
CPPLDFLAGS       =
CPPOPTFLAGS      = -O3

# Fortran compiler settings
F90              = mpif90 -fopenmp
F90FLAGS         =
F90LDFLAGS       = -lstdc++ -L$(HOME)/local/lib -lopenblas
F90OPTFLAGS      = -O2
F90FPP          = -cpp
F90LINKER        = mpif90 -fopenmp

MAKE             = make
AR               = ar ruv
MV              = mv -f
CP              = cp -f
RM              = rm -f
MKDIR            = mkdir -p

```

### 1.16.3.2 setup.sh の実行

編集が完了したら、setup.sh を実行します。

```
$ ./setup.sh -p --with-tools --with-refiner \
               --with-metis --with-mumps --with-lapack --with-ml
```

### 1.16.3.3 make の実行

make を実行します。

```
$ make
```

#### 1.16.3.4 make install の実行

make が完了したら、make install を実行し Makefile.conf で指定したディレクトリヘインストールします。この例では \$(HOME)/FrontISTR/bin になります。

```
$ make install
```

#### 1.16.3.5 動作確認

本ソフトウェアに同梱されているチュートリアルを実行して、動作を確認します。

```
$ cd $HOME/work/FrontISTR/tutorial
```

```
$ cd 01_elastic_hinge
```

```
$ $HOME/FrontISTR/bin/fistr1
```

```
Step control not defined! Using default step=1
```

```
fstr_setup: OK
```

```
Start visualize PSF 1 at timestep 0
```

```
loading step= 1
```

```
sub_step= 1, current_time= 0.0000E+00, time_inc= 0.1000E+01
```

```
loading_factor= 0.0000000 1.0000000
```

```
### 3x3 BLOCK CG, SSOR, 1
```

```
1 1.903375E+00
```

```
2 1.974378E+00
```

```
3 2.534627E+00
```

```
4 3.004045E+00
```

```
5 3.202633E+00
```

```
6 3.203864E+00
```

```
...
```

```
...
```

解析が終了すると以下の様に画面上に表示されます。

```
...
```

```
...
```

```
2966 1.143085E-08
```

```
2967 1.078272E-08
```

```
2968 1.004759E-08
```

```
2969 9.372882E-09
```

```
### Relative residual = 9.39169E-09
```

```
### summary of linear solver
```

```
2969 iterations 9.391687E-09
```

```

set-up time      :      4.108060E-01
solver time      :      6.506822E+01
solver/comm time :      4.342469E-01
solver/matvec    :      1.923199E+01
solver/precond   :      2.688405E+01
solver/1 iter    :      2.191587E-02
work ratio (%)   :      9.933263E+01

```

```

Start visualize PSF 1 at timestep 1
### FSTR_SOLVE_NLGEOM FINISHED!

```

---



---

```

TOTAL TIME (sec) :      74.93
      pre (sec) :      1.86
      solve (sec) :      73.07

```

---



---

FrontISTR Completed !!

## 1.17 参考 Windows10 へのインストール手順例 (Makefile.conf)

Windows10 上へ、本ソフトウェアとそれに必要な外部ライブラリの構築手順の例を示します。他の環境へのインストールの参考にしてください。

また、各ライブラリの詳細な構築方法は、それぞれのドキュメントを参考にしてください。

### 1.17.1 準備

最初に本ソフトウェアをコンパイルするのに必要なツールやパッケージをインストールしてください。

#### 1.17.1.1 開発環境の準備

はじめに開発環境をインストールします。使用する開発環境は MSYS2 です。

<https://www.msys2.org/>

下記 URL から 64 ビット版のインストーラ `msys2-x86_64-xxxxxxx.exe` (xxxxxxx はバージョン番号) をダウンロードしインストールします。

#### 1.17.1.2 パッケージのインストール

インストールが完了したらMSYS2 MinGW 64-bitと書かれたコマンドプロンプトを立ち上げ、コンパイルに必要なパッケージをインストールします。

```
(MINGW64) pacman -S base-devel mingw-w64-x86_64-toolchain \
    mingw-w64-x86_64-cmake \
    mingw-w64-x86_64-binutils \
    mingw-w64-x86_64-perl \
    git
```

gcc/g++/gfortran が正しくインストールされているか確認してください。

```
(MINGW64) which gcc g++ gfortran
/mingw64/bin/gcc
/mingw64/bin/g++
/mingw64/bin/gfortran
```

### 1.17.2 ライブラリのインストール

本ソフトウェアに必要なライブラリをインストールします。作業ディレクトリは\$HOME/work、インストール先のディレクトリは\$HOME/localとします。

各ディレクトリを作成し、\$HOME/local/binを PATH 環境変数に追加します。

```
(MINGW64) cd $HOME
(MINGW64) mkdir work
(MINGW64) mkdir -p local/bin local/lib local/include
(MINGW64) export PATH=$HOME/local/bin:$PATH
```

#### 1.17.2.1 MPI のインストール

この例では、MPI として Microsoft 社の MPI を利用します。

下記 URL からランタイム (msmpisetup.exe) と SDK(msmpisdk.msi) がダウンロードできます。

Download Microsoft MPI v10.0

##### 1.17.2.1.1 .a ライブラリの作成

インストールしたライブラリを MinGW-w64 の gcc や gfortran でリンクできるように変更を加えます。

インストールした.dll から.a を生成します。

```
(MINGW64) cd $HOME/local/lib
(MINGW64) gendef /c/Windows/System32/msmpi.dll
(MINGW64) dlltool -d msmpi.def -l libmsmpi.a -D /c/Windows/System32/msmpi.dll
```



```
(MINGW64) ls
libmsmpi.a msmpi.def
```

#### 1.17.2.1.2 ヘッダファイルの修正

次にヘッダファイルをコピーします。

```
(MINGW64) cd $HOME/local/include
(MINGW64) cp /c/Program\ Files\ \((x86\) /Microsoft\ SDKs/MPI/Include/*.h .
(MINGW64) cp /c/Program\ Files\ \((x86\) /Microsoft\ SDKs/MPI/Include/x64/*.h .
(MINGW64) ls
mpi.h mpif.h mpifptr.h mpio.h mspms.h pmidbg.h
```

#### 1.17.2.2 ダウンロード

その他のソフトウェアをダウンロードし、作業ディレクトリ\$HOME/workへ保存します。

| ソフトウェア名                         | ダウンロード先   |
|---------------------------------|---|
| REVOCAP_Refiner-1.1.04.tar.gz   | <a href="https://www.frontistr.com/">https://www.frontistr.com/</a>   |
| FrontISTR_V50.tar.gz            | <a href="https://www.frontistr.com/">https://www.frontistr.com/</a>   |
| OpenBLAS-0.2.20.tar.gz          | <a href="http://www.openblas.net/">http://www.openblas.net/</a>   |
| metis-5.1.0.tar.gz              | <a href="http://glaros.dtc.umn.edu/gkhome/metis/metis/download">http://glaros.dtc.umn.edu/gkhome/metis/metis/download</a> |
| scalapack-2.0.2.tgz             | <a href="http://www.netlib.org/scalapack/">http://www.netlib.org/scalapack/</a>   |
| MUMPS_5.1.2.tar.gz              | <a href="http://mumps.enseiht.fr/">http://mumps.enseiht.fr/</a>   |
| trilinos-12.14.1-Source.tar.bz2 | <a href="https://trilinos.org/download/">https://trilinos.org/download/</a>   |

#### 1.17.2.3 REVOCAP\_Refiner のコンパイル

```
(MINGW64) cd $HOME/work
(MINGW64) tar xvf REVOCAP_Refiner-1.1.04.tar.gz
(MINGW64) cd REVOCAP_Refiner-1.1.04
(MINGW64) make
(MINGW64) cp lib/x86_64-linux/libRcapRefiner.a $HOME/local/lib
(MINGW64) cp Refiner/rcapRefiner.h $HOME/local/include
```

#### 1.17.2.4 OpenBLAS のインストール

OpenBLAS は MSYS2 から提供されるバイナリパッケージを利用します。

```
(MINGW64) pacman -S mingw-w64-x86_64-openblas
```

### 1.17.2.5 METIS のコンパイル

```
(MINGW64) cd $HOME/work
```

```
(MINGW64) tar xvf metis-5.1.0.tar.gz
```

```
(MINGW64) cd metis-5.1.0
```

MinGW-w64 に合わせるため、以下のファイルを一部修正します。

- Makefile
- GKlib/gk\_arch.h
- GKlib/getopt.c

```
% vim Makefile
```

60 行目の

```
cd $(BUILDDIR) && cmake $(CURDIR) $(CONFIG_FLAGS)
```

を

```
cd $(BUILDDIR) && cmake -G "MSYS Makefiles" $(CURDIR) $(CONFIG_FLAGS)
```

に変更

```
(MINGW64) vim GKlib/gk_arch.h
```

44 行目の

```
#include <sys/resource.h>
```

を削除

```
(MINGW64) vim GKlib/gk_getopt.h
```

54 行目からの

```
/* Function prototypes */
```

```
extern int gk_getopt(int __argc, char **__argv, char *__shortopts);
```

```
extern int gk_getopt_long(int __argc, char **__argv, char *__shortopts,  
                          struct gk_option *__longopts, int *__longind);
```

```
extern int gk_getopt_long_only (int __argc, char **__argv,  
                                char *__shortopts, struct gk_option *__longopts, int *__longind);
```

を削除。

```
(MINGW64) make config prefix=$HOME/local cc=gcc openmp=1
```

```
(MINGW64) make
```

```
(MINGW64) make install
```

### 1.17.2.6 ScaLAPACK のコンパイル

```
(MINGW64) cd $HOME/work
(MINGW64) tar xvf scalapack-2.0.2.tgz
(MINGW64) cd scalapack-2.0.2
```

サンプルの SLmake.inc.example を SLmake.inc としてコピーし、環境に合わせて編集します。

```
(MINGW64) cp SLmake.inc.example SLmake.inc
(MINGW64) vi SLmake.inc
```

```
#
# The fortran and C compilers , loaders , and their flags
#

FC          = gfortran -fno-range-check
CC          = gcc
NOOPT       = -O0
FCFLAGS     = -O3 -I$(HOME)/local/include
CCFLAGS     = -O3 -I$(HOME)/local/include
FCLOADER    = $(FC)
CCLOADER    = $(CC)
FCLOADFLAGS = $(FCFLAGS) -L$(HOME)/local/lib -lmsmpi
CCLOADFLAGS = $(CCFLAGS) -L$(HOME)/local/lib -lmsmpi

#
# BLAS, LAPACK (and possibly other) libraries needed for linking test programs
#

BLASLIB      = -lopenblas
LAPACKLIB    = -lopenblas
LIBS         = $(LAPACKLIB) $(BLASLIB)
```

編集が完了したら make し、完成したライブラリをコピーします。

```
(MINGW64) make
(MINGW64) cp libscalapack.a $HOME/local/lib
```

コンパイル終了時にエラーが表示されますが無視して構いません。

#### 1.17.2.7 MUMPS のコンパイル

```
(MINGW64) cd $HOME/work
(MINGW64) tar xvf MUMPS_5.1.2.tar.gz
(MINGW64) cd MUMPS_5.1.2
```

```
(MINGW64) cp Make.inc/Makefile.inc.generic Makefile.inc
```

コピーしたMakefile.incの以下の部分を書き換えます。

```
(MINGW64) vi Makefile.inc
```

```
(MINGW64) cp Make.inc/Makefile.inc.generic Makefile.inc
```

```
(MINGW64) vi Makefile.inc
```

```
LMETISDIR = $(HOME)/local
```

```
IMETIS     = -I$(LMETISDIR)/include
```

```
LMETIS     = -L$(LMETISDIR)/lib -lmetis
```

```
ORDERINGSF = -Dmetis -Dpord
```

```
CC          = gcc
```

```
FC          = gfortran -fno-range-check
```

```
FL          = gfortran
```

```
LAPACK = -lopenblas
```

```
SCALAP     = -L$(HOME)/local/lib -lscalapack
```

```
INCPAR = -I$(HOME)/local/include
```

```
LIBPAR     = $(SCALAP) $(LAPACK) -L$(HOME)/local/lib -lmsmpi
```

```
LIBBLAS = -lopenblas
```

```
LIBOTHERS = -lpthread
```

```
OPTF       = -O -fopenmp
```

```
OPTC       = -O -I. -fopenmp
```

```
OPTL       = -O -fopenmp
```

書き換えが完了したら保存し make します。

```
(MINGW64) make
```

```
(MINGW64) cp lib/*.a $HOME/local/lib
```

```
(MINGW64) cp include/*.h $HOME/local/include
```

#### 1.17.2.8 Trilinos ML のコンパイル

```
(MINGW64) cd $HOME/work
```

```

(MINGW64) tar xvf trilinos-12.14.1-Source.tar.gz
(MINGW64) cd trilinos-12.14.1-Source
(MINGW64) mkdir build
(MINGW64) cmake -G "MSYS Makefiles" \
    -DCMAKE_INSTALL_PREFIX="$HOME/local" \
    -DCMAKE_CXX_FLAGS="-I$HOME/local/include" \
    -DCMAKE_C_FLAGS="-I$HOME/local/include" \
    -DBLAS_LIBRARY_NAMES="openblas" \
    -DLAPACK_LIBRARY_NAMES="openblas" \
    -DMPI_USE_COMPILER_WRAPPERS=OFF \
    -DMPI_C_HEADER_DIR="$HOME/local/include" \
    -DMPI_CXX_HEADER_DIR="$HOME/local/include" \
    -DTPL_ENABLE_MPI=ON \
    -DTrilinos_ENABLE_OpenMP=ON \
    -DTrilinos_ENABLE_ML=ON \
    -DTrilinos_ENABLE_Zoltan=ON \
    -DTrilinos_ENABLE_ALL_OPTIONAL_PACKAGES=OFF \
    ..
(MINGW64) make
(MINGW64) make install

```

### 1.17.3 FrontISTR のコンパイル

上記ライブラリのコンパイルが済んだら FrontISTR をコンパイルします。

```

(MINGW64) cd $HOME/work
(MINGW64) tar xvf FrontISTR_V50.tar.gz
(MINGW64) cd FrontISTR

```

#### 1.17.3.1 Makefile.conf の編集

雛形をコピーして、環境に合わせた内容に編集します。この例では、以下の様に編集します。

```

(MINGW64) cp Makefile.conf.org Makefile.conf
(MINGW64) vi Makefile.conf
#####
#                                                                    #
#      Setup Configuration File for FrontISTR                      #
#                                                                    #
#####

```

```

# MPI
MPIDIR          = $(HOME)/local
MPIBINDIR       = "/c/Program\ Files/Microsoft\ MPI/Bin/"
MPILIBDIR       = $(MPIDIR)/lib
MPIINCDir       = $(MPIDIR)/include
MPILIBS         = -lmsmpi

# for install option only
PREFIX          = $(HOME)/FrontISTR
BINDIR          = $(PREFIX)/bin
LIBDIR          = $(PREFIX)/lib
INCLUDEDIR      = $(PREFIX)/include

# Metis
METISDIR        = $(HOME)/local
METISLIBDIR     = $(METISDIR)/lib
METISINCDIR     = $(METISDIR)/include
HECMW_METIS_VER= 5

# ParMetis
PARMETISDIR     = $(HOME)/local
PARMETISLIBDIR  = $(PARMETISDIR)/lib
PARMETISINCDIR = $(PARMETISDIR)/include

# Refiner
REFINERDIR      = $(HOME)/local
REFINERINCDIR   = $(REFINERDIR)/include
REFINERLIBDIR   = $(REFINERDIR)/lib

# Coupler
REVOCAPDIR      = $(HOME)/local
REVOCAPINCDIR   = $(REVOCAPDIR)/include
REVOCAPLIBDIR   = $(REVOCAPDIR)/lib

# MUMPS
MUMPSDIR        = $(HOME)/local
MUMPSINCDIR     = $(MUMPSDIR)/include
MUMPSLIBDIR     = $(MUMPSDIR)/lib
MUMPSLIBS       = -ldmumps -lmumps_common -lpord -L$HOME/local/lib -lscalapack

# MKL PARDISO

```

```

MKLDIR      = $(HOME)/
MKLINCDIR   = $(MKLDIR)/include
MKLLIBDIR   = $(MKLDIR)/lib

# ML
MLDIR        = $(HOME)/local
MLINCDIR     = $(MLDIR)/include
MLLIBDIR     = $(MLDIR)/lib
MLLIBS       = -lm1 -lzoltan -lws2_32

# C compiler settings
CC           = gcc -fopenmp
CFLAGS       = -D_WINDOWS
LDFLAGS      = -lstdc++ -lm
OPTFLAGS     = -O3

# C++ compiler settings
CPP          = g++ -fopenmp
CPPFLAGS     = -D_WINDOWS
CPPLDFLAGS   =
CPPOPTFLAGS  = -O3

# Fortran compiler settings
F90          = gfortran -fopenmp -fno-range-check
F90FLAGS     =
F90LDFLAGS   = -lstdc++ -lopenblas
F90OPTFLAGS  = -O2
F90FPP       = -cpp
F90LINKER    = gfortran -fopenmp

MAKE         = make
AR           = ar ruv
MV           = mv -f
CP           = cp -f
RM           = rm -f
MKDIR        = mkdir -p

```

### 1.17.3.2 setup.sh の実行

編集が完了したら、setup.sh を実行します。

```
(MINGW64) ./setup.sh -p --with-tools --with-refiner \
--with-metis --with-mumps --with-lapack --with-mt
```

### 1.17.3.3 make の実行

make を実行します。

```
(MINGW64) make
```

### 1.17.3.4 make install の実行

make が完了したら、make install を実行し Makefile.conf で指定したディレクトリへインストールします。この例では \$(HOME)/FrontISTR/bin です。

```
(MINGW64) make install
```

### 1.17.3.5 動作確認

本ソフトウェアに同梱されているチュートリアルを実行して、動作を確認します。

```
(MINGW64) cd $HOME/work/FrontISTR/tutorial
(MINGW64) cd 01_elastic_hinge
(MINGW64$) $HOME/FrontISTR/bin/fistr1
Step control not defined! Using default step=1
fstr_setup: OK
Start visualize PSF 1 at timestep 0

loading step=    1
sub_step= 1,    current_time=  0.0000E+00, time_inc=  0.1000E+01
loading_factor=    0.0000000    1.0000000
### 3x3 BLOCK CG, SSOR, 1
    1    1.903375E+00
    2    1.974378E+00
    3    2.534627E+00
    4    3.004045E+00
    5    3.202633E+00
    6    3.203864E+00
...
...
```

解析が終了すると以下の様に画面上に表示されます。



```

...
...
2966      1.143085E-08
2967      1.078272E-08
2968      1.004759E-08
2969      9.372882E-09
### Relative residual = 9.39169E-09

### summary of linear solver
      2969 iterations      9.391687E-09
set-up time      :      4.108060E-01
solver time      :      6.506822E+01
solver/comm time :      4.342469E-01
solver/matvec    :      1.923199E+01
solver/precond   :      2.688405E+01
solver/1 iter    :      2.191587E-02
work ratio (%)   :      9.933263E+01

```

```

Start visualize PSF 1 at timestep 1
### FSTR_SOLVE_NLGEOM FINISHED!

```

|                    |       |
|--------------------|-------|
| <hr/>              |       |
| TOTAL TIME (sec) : | 74.93 |
| pre (sec) :        | 1.86  |
| solve (sec) :      | 73.07 |
| <hr/>              |       |

FrontISTR Completed !!

#### 1.17.3.6 補足

MinGW のインストールされていない環境で実行するには、FrontISTR `fistr1.exe` と同じディレクトリに以下のファイルをコピーします。

- `libwinpthread-1.dll`
- `libgfortran-3.dll`
- `libgcc_s_seh-1.dll`
- `libgomp-1.dll`
- `libstdc++-6.dll`
- `libquadmath-0.dll`

通常は、

C:\msys64\mingw64\bin

の下にありますので、バイナリを実行するコンピュータにコピーします。

また、Microsoft MPI のランタイム MSMpiSetup.exe も実行するコンピュータにインストールします。