

Windows10上での FrontISTR v5.0aの構築法

帝京大学 戦略的イノベーション研究センター

小川 道夫

2018年8月7日(火)

目次

1. 開発環境 git for windows SDK のインストール
2. OpenBLASのコンパイル
3. MUMPSのコンパイル
4. Trilinos MLのコンパイル
5. REVOCAP_Refinerのコンパイル
6. gitの簡単な説明
7. FrontISTRのコンパイル

構築するFrontISTR

- ▶ windows10上でコンパイル
- ▶ MPI無し
- ▶ OpenMP並列
- ▶ BLAS/LAPACKにはOpenBLASを利用
- ▶ 直接法ソルバーMUMPSの組み込み
- ▶ REVOCAP_Refinerの組み込み
- ▶ MLの組み込み。MPI無し
- ▶ tools作成(領域分割にはMETIS無し)

git for windows SDK(1)

- ▶ <https://github.com/git-for-windows/build-extra/releases>
から64ビット版のインストーラをダウンロード

The screenshot shows the GitHub release page for the repository `git-for-windows/build-extra`. The release is titled `git-sdk-1.0.7` and was released by `dscho` on April 11. The page displays the release details, including the commit hash `1227299` and the number of commits since the release (`157`). The `Assets` section lists the following files:

| Asset Name | Size |
|---|---------|
| git-sdk-installer-1.0.7-32.7z.exe | 3.41 MB |
| git-sdk-installer-1.0.7-64.7z.exe | 3.6 MB |
| Source code (zip) | |
| Source code (tar.gz) | |

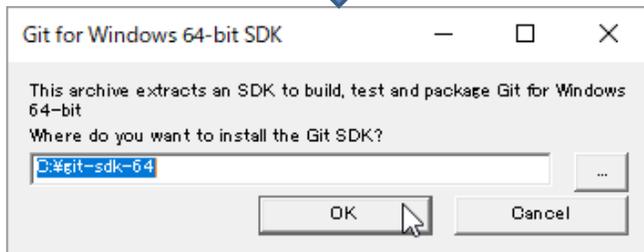
The `git-sdk-installer-1.0.7-64.7z.exe asset is highlighted with a mouse cursor. The page also shows the repository name, star count (198), and fork count (258). The URL at the bottom of the browser window is https://github.com/git-for-windows/build-extra/releases/download/git-sdk-1.0.7/git-sdk-installer-1.0.7-64.7z.exe.`

git for windows SDK(2)

- ▶ インストーラを起動し、git for Windows SDK をインストール



git-sdk-installer-1.0.7-64.7z



インストールする場所を指定(デフォルトでOK)

```
C:\WINDOWS\system32\cmd.exe
Cloning the Git for Windows SDK...
warning: templates not found C:\git-sdk-64\mini\mingw64\share\git-core/templates
Initialized empty Git repository in C:/git-sdk-64/.git/
remote: Counting objects: 49047, done.
remote: Compressing objects: 100% (35884/35884), done.
remote: Total 49047 (delta 18819), reused 31240 (delta 11892), pack-reused 0
Receiving objects: 100% (49047/49047), 588.83 MiB | 4.09 MiB/s, done.

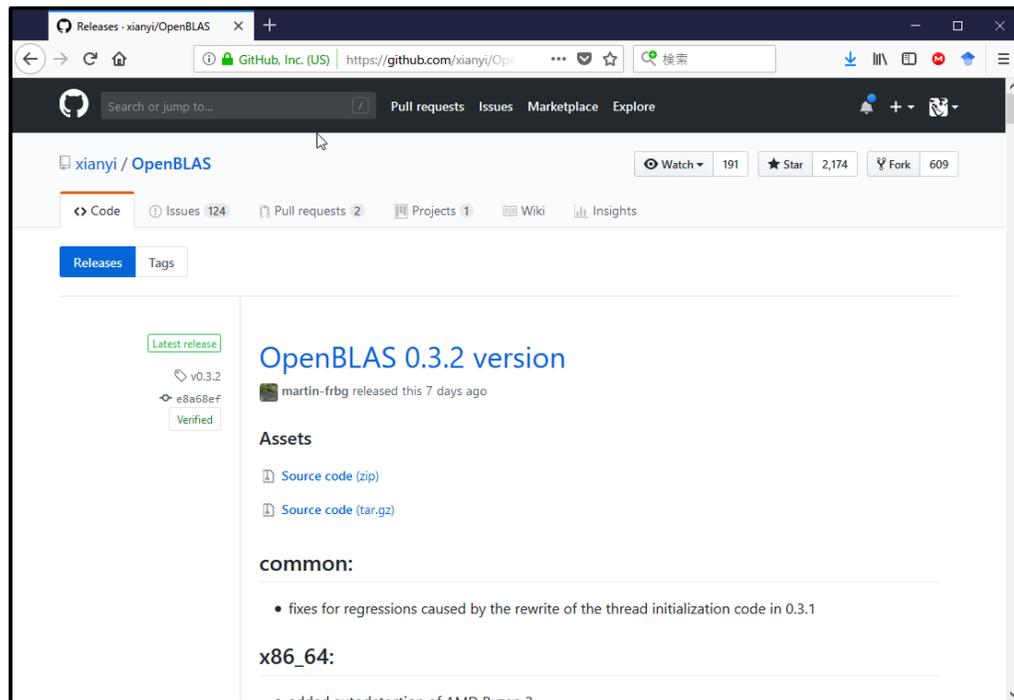
Resolving deltas: 100% (18819/18819), done.
From https://github.com/git-for-windows/git-sdk-64
* [new branch]      master -> origin/master
Checking out files: 2% (2248/99750)
```

インストール中。環境によっては10分程かかる



完了。デスクトップにアイコンが表示される。

OpenBLASのコンパイル(1)



▶ ダウンロードサイト

<https://github.com/xianyi/OpenBLAS/releases>

OpenBLAS-0.3.2.tar.gz をダウンロード

以降の手順は Git SDK 64-bit (コマンドライン)の中で行います。



OpenBLASのコンパイル(2)

- ▶ ダウンロードしたアーカイブを展開します。
- ▶ ディレクトリ内のMakefile.ruleを右の様に編集します。

```
$ tar xvf OpenBLAS-0.3.2.tar.gz
$ cd OpenBLAS-0.3.2
$ vi Makefile.rule
$ make
$ make PREFIX=$HOME/local install
```

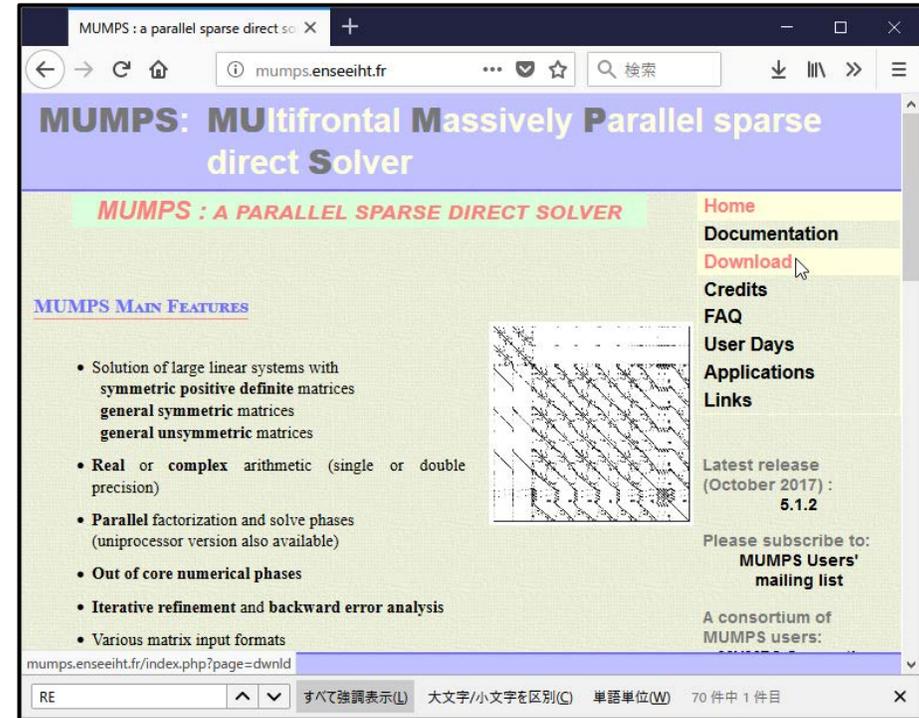
- ▶ DYNAMIC_ARCH=1
 - ▶ 実行するマシンが同じ場合、この指定は必要ありません(コメントを外さない)。
 - ▶ コンパイル時間が長くなります。
- ▶ CC=gcc
- ▶ FC=gfortran
- ▶ BINARY=64
- ▶ USE_OPENMP=1
- ▶ NO_AVX2=1
 - ▶ 現在のバージョンではSkylake XのAVX2のコードに問題があるようです。

MUMPSのコンパイル(1)

ダウンロード先

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

- ▶ ユーザ登録が必要



MUMPSのコンパイル(2)

- ▶ 展開したアーカイブの中の Make.inc/
ディレクトリにある
Makefile.inc.generic.SEQ をひな形にし
て右の様に編集します

```
$ tar xvf MUMPS_5.1.2.tar.gz
$ cd MUMPS_5.1.2
$ cp Make.inc/Makefile.inc.generic.SEQ
Makefile.inc
$ vi Makefile.inc
$ make
$ cp lib/*.a $HOME/local/lib
$ cp libseq/libmpiseq.a $HOME/local/lib
$ cp include/*.h $HOME/local/include
$ cp libseq/*.h $HOME/local/include
```

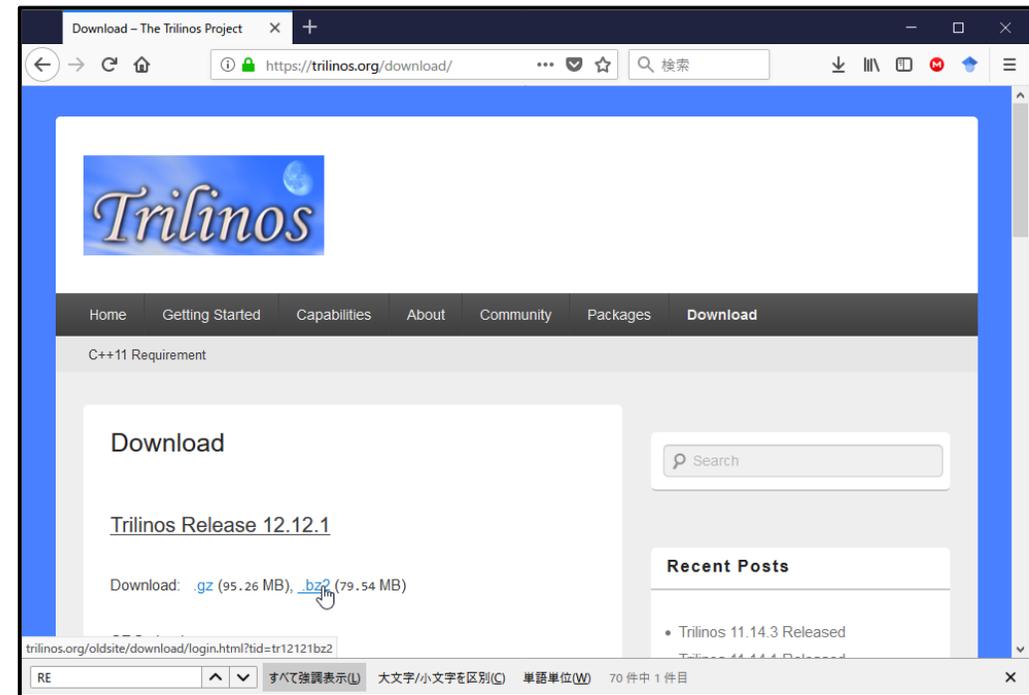
```
CC=gcc
FC=gfortran
FL=gfortran
LAPACK=-L$(HOME)/local/lib -lopenblas
LIBBLAS=$(LAPACK)
OPTF = -O -fopenmp -DBLR_MT
OPTC = -O -l. -fopenmp
OPTL = -O -fopenmp
```

Trilinos MLのコンパイル(1)

ダウンロード先

<https://trilinos.org/download/>

- ▶ trilinos-12.12.1-Source.tar.bz2 をダウンロード
- ▶ ユーザ登録が必要



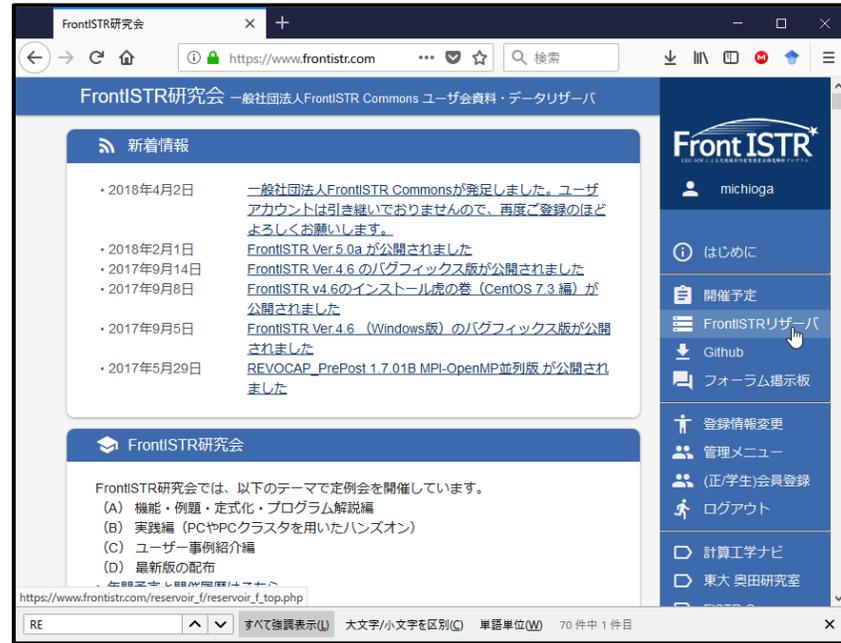
Trilinos MLのコンパイル

- ▶ cmakeでMakefileを作成
- ▶ cmakeを起動するときに渡す引数は右の通り

```
$ pacman -S mingw-w64-x86_64-cmake  
(インストールされていない場合)  
$ tar xvf trilinos-12.12.1-Source.tar.bz2  
$ cd trilinos-12.12.1-Source  
$ mkdir build  
$ cd build  
$ cmake ..  
$ make  
$ make install
```

```
cmake -G "MSYS Makefiles" ¥  
-DCMAKE_INSTALL_PREFIX=$HOME/local ¥  
-DTPL_BLAS_LIBRARIES=$HOME/local/lib/libopenblas.a ¥  
-DTPL_LAPACK_LIBRARIES=$HOME/local/lib/libopenblas.a ¥  
-DTrilinos_ENABLE_OpenMP=ON ¥  
-DTrilinos_ENABLE_ML=ON ¥  
-DTrilinos_ENABLE_ALL_OPTIONAL_PACKAGES=OFF ¥  
..
```

REVOCAP_Refinerのコンパイル



ダウンロード先

<https://www.frontistr.com/>

の FrontISTRリザーバ よりダウンロード
展開し、そのまま make で構築完了

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

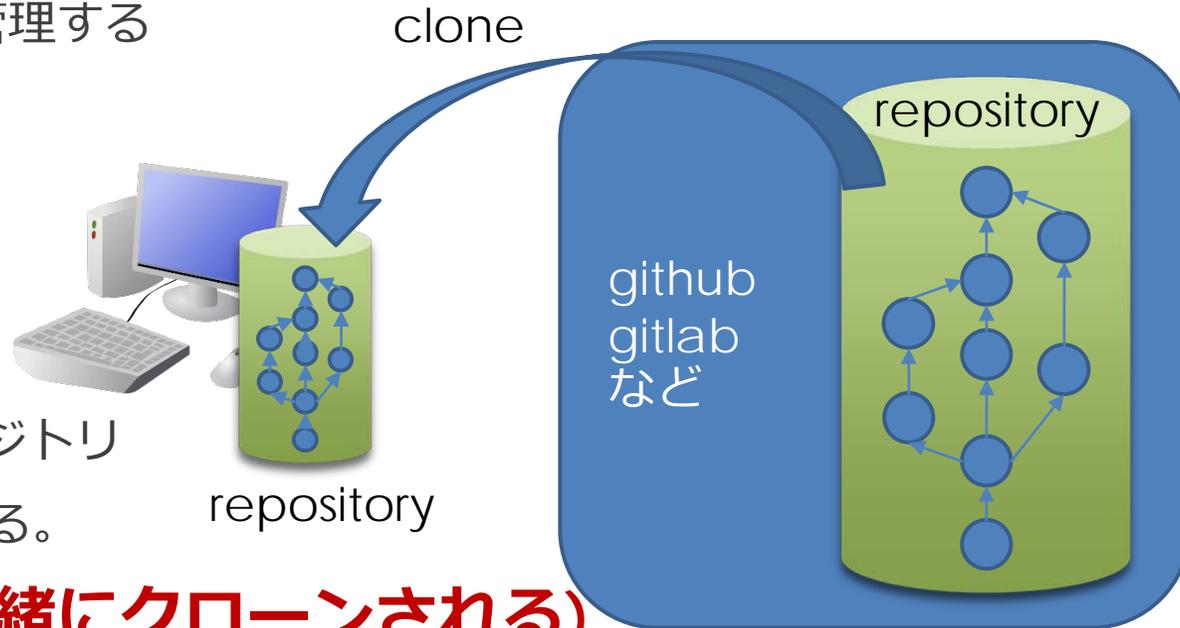
gitの簡単な説明(1)

- ▶ ソースコードなどを分散して管理する



githubやgitlabのリモートのレポジトリをローカルにクローンして作業する。

(これまでの変更履歴も一緒にクローンされる)



gitの簡単な説明(2)

- ▶ FrontISTRは git でプログラムを管理しています。
- ▶ git に便利なユーザインターフェースを付けてホスティングしているサービスがあります。
 - ▶ <https://gitlab.com> (Gitlab Inc.が運営するサービス)
 - ▶ <https://github.com> (Microsoftが先日買収したサービス)
- ▶ 従来通り、frontistr.com からアーカイブを取得することが出来ますが、開発版を利用したい場合 git の使い方を知っておく必要があります。
- ▶ Windowsの場合、先にインストールした git for windows SDKに同梱されているので改めてインストールする必要はありません。
- ▶ 他のプラットフォームでも、バイナリパッケージが提供されています。



ブラウザ上のインターフェース

github

gitlab

The image displays two overlapping browser windows. The background window shows the GitHub interface for the repository 'FrontISTR / FrontISTR'. The page title is 'FrontISTR / FrontISTR' and the description is 'Open-Source Large-Scale Parallel FEM Program for Nonlinear Structural Analysis (オープンソース大規模並列FEM非線形構造解析プログラム): FrontISTR https://www.frontistr.com/'. It lists various tags like 'fem', 'finite-element-analysis', and 'mechanical-engineering'. The commit history shows a recent commit by 'nqomorita' with the message 'Fixed to output 2nd order tetra element in vtk format'. The foreground window shows the GitLab interface for the repository 'FrontISTR-Commons / FrontISTR-devel'. The page title is 'FrontISTR-devel' and it is identified as a 'FrontISTR Developer's private repository'. It shows 0 stars and 0 forks. The commit history shows a recent commit by 'morita' with the message 'Fixed to output 2nd order tetra element in vtk format'. The file list includes 'cmake' and 'doc'.

gitでFrontISTRのアーカイブを取得

- ▶ ここではコマンドラインの git を用いてFrontISTRを取得してみます。
- ▶ Git SDK 64-bitを開き、以下のコマンドを打ち込んでください。
- ▶ 最新開発ブランチ master が取得できます

```
% git clone https://gitlab.com/FrontISTR-Commons/FrontISTR.git
```

gitのコマンド
リモートのレポジトリをそのまま
コピーという意味
これまでの変更履歴も含まれます

コピーする元を指定し
ます。

gitで v5.0aを取り出す

- ▶ 取得したリポジトリには、過去にリリースされたバージョンも入っています。
- ▶ 以降の手順では v5.0a を対象として説明しますので、以下のコマンドを実行してください。
 - ▶ % git checkout v5.0a
- ▶ これで、今取得したリポジトリは v5.0a と同じものになりました。

% git status でレポジトリの状態をチェック

```
% cd FrontISTR
% git tag
v4.0
v4.1
v4.2
v4.2c
v4.3
v4.3rc1
v4.3rc2
v4.3rc3
v4.4
v4.5
v4.5rc1
v4.6
v5.0a
```

FrontISTRのコンパイル

- ▶ Makefile.conf.org をひな形として設定をします。
- ▶ `cp Makefile.conf.org Makefile.conf; vi Makefile.conf`
- ▶ `sh ./setup.sh --with-tools --with-lapack --with-refiner --with-mumps --with-ml`
- ▶ `make`
- ▶ `make install`

で\$HOME/FrontISTR/binに **fistr1.exe (FrontISTR)** がインストールされます。
Makefile.conf は巻末の参考をご覧ください。

※注意点

- ▶ `-D_WINDOWS` の指定
- ▶ `-lws2_32` (winsock)の指定

実行の準備

- ▶ 出来上がった fistr1.exe は右の図の様なDLLに依存しています。
- ▶ /c/WINDOWS/System32に入っていないDLLをパスの通った場所にコピーして利用してください。

libwinpthread-1.dll
libgcc_s_seh-1.dll
libgfortran-4.dll
libstdc++-6.dll
libgomp-1.dll
libquadmath-0.dll

```
$ ldd fistr1.exe
ntdll.dll => /c/WINDOWS/SYSTEM32/ntdll.dll (0x7ffffbac10000)
KERNEL32.DLL => /c/WINDOWS/System32/KERNEL32.DLL (0x7ffffb9910000)
KERNELBASE.dll => /c/WINDOWS/System32/KERNELBASE.dll (0x7ffffb7080000)
msvcrt.dll => /c/WINDOWS/System32/msvcrt.dll (0x7ffffba030000)
USER32.dll => /c/WINDOWS/System32/USER32.dll (0x7ffffba2a0000)
win32u.dll => /c/WINDOWS/System32/win32u.dll (0x7ffffb7540000)
GDI32.dll => /c/WINDOWS/System32/GDI32.dll (0x7ffffb9b60000)
libwinpthread-1.dll => /mingw64/bin/libwinpthread-1.dll (0x64940000)
gdi32full.dll => /c/WINDOWS/System32/gdi32full.dll (0x7ffffb7560000)
msvc_p_win.dll => /c/WINDOWS/System32/msvc_p_win.dll (0x7ffffb6fe0000)
ucrtbase.dll => /c/WINDOWS/System32/ucrtbase.dll (0x7ffffb7890000)
WS2_32.dll => /c/WINDOWS/System32/WS2_32.dll (0x7ffffb9bd0000)
RPCRT4.dll => /c/WINDOWS/System32/RPCRT4.dll (0x7ffffb97e0000)
libgcc_s_seh-1.dll => /mingw64/bin/libgcc_s_seh-1.dll (0x61440000)
libgfortran-4.dll => /mingw64/bin/libgfortran-4.dll (0x6da80000)
libstdc++-6.dll => /mingw64/bin/libstdc++-6.dll (0x6fc40000)
libgomp-1.dll => /mingw64/bin/libgomp-1.dll (0x63600000)
ADVAPI32.dll => /c/WINDOWS/System32/ADVAPI32.dll (0x7ffffb94e0000)
sechost.dll => /c/WINDOWS/System32/sechost.dll (0x7ffffb9fd0000)
libquadmath-0.dll => /mingw64/bin/libquadmath-0.dll (0x6cf00000)

michioga@flow-p06 MINGW64 ~/Work/FrontISTR/fistr1/bin ((v5.0a))
$ |
```

これまでFrontISTR研究会で行った インストールに関する資料

- ▶ 2018年5月11日 FrontISTR Commons設立記念シンポジウム [インストールの実際](#)
- ▶ 2017年12月22日 第40回FrontISTR Ver.5.0プログラムの公開に向けて [FrontISTRビルドスクリプトの紹介](#)
- ▶ 2017年11月10日 第39回FrontISTRのプログラム開発・実行環境に関する二、三の話題 [FrontISTRのビルド方法の紹介](#)
- ▶ 2016年10月17日 第31回さらに便利になったFrontISTR実行環境 [メタビルドシステムCMakeによるFrontISTRの構築 -FrontISTR v5.0に向けて-](#)
- ▶ 2015年12月21日 第24回便利になったFrontISTR実行環境、「Cistr」のハンズオン [説明](#) [虎の巻1](#) [虎の巻2](#) [虎の巻3](#) [虎の巻4](#) [虎の巻5](#)
- ▶ 2014年7月30日 第11回機能・例題・定式化・プログラム解説編 [社内linuxマシンでFrontISTR](#)

古いものも含まれていますが、参考にしてください。

外部リソース

また、Qiitaにもまとめてありますので参考にしてください。

- ▶ [FrontISTR v4.6のインストール \(ubuntu 16.04 LTS\)](#)
- ▶ [FrontISTR v4.6のインストール \(CentOS 7.3\)](#)
- ▶ [FrontISTR v4.6のインストール \(Windows10 - MinGW-w64\)](#)

- ▶ [FrontISTR v5.0aのインストール \(macOS High Sierra 10.13.4\)](#)
 - ▶ 東大生産研 森田様 寄稿

```

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

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

# Refiner
REFINERDIR  = $(HOME)/local
REFINERINCDIR = $(REFINERDIR)/include
REFINERLIBDIR = $(REFINERDIR)/lib
# Coupler
REVOCAPDIR  = $(HOME)/REVOCAP_Coupler
REVOCAPINCDIR = $(REVOCAPDIR)/librcap
REVOCAPLIBDIR = $(REVOCAPDIR)/librcap

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

```

```

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

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

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

# C++ compiler settings
CPP         = g++
CPPFLAGS    = -D_WINDOWS -fopenmp
CPPLDFFLAGS =
CPOPTFFLAGS = -O3

# Fortran compiler settings
F90         = gfortran
F90FLAGS    = -D_WINDOWS -fopenmp
F90LDFLAGS  = -fopenmp -lstdc++ $(HOME)/local/lib/libopenblas.a -lws2_32
F90OPTFFLAGS = -O2
F90FPP      = -cpp
F90LINKER   = gfortran

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

```