OD5080-23-A3MOS024

AMSR3 プロダクトアトリビュート
読み込みツール
取扱説明書

2023 年 10 月 25 日

目次

1　はじめに

1.1　目的

AMSR3 プロダクトアトリビュート読み込みツール（以下、NFTOOL とします。）は、AMSR3 プロダクトのグローバルアトリビュート、データセットアトリビュートを Fortran77、Fortran90 で読み込むために使用します。NFTOOL の概要を図 1-1 に示します。

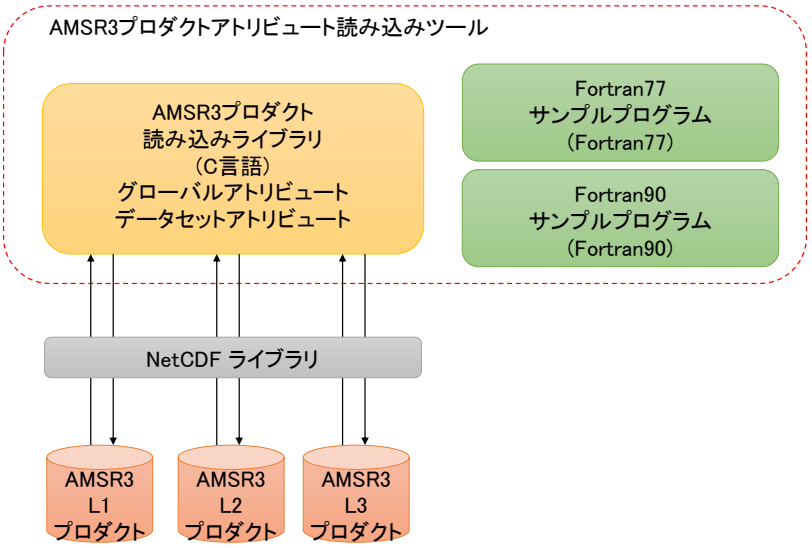


図 1-1 NFTOOL 概要

2　インストール

2.1　動作環境

NFTOOL の動作環境を表 2-1 に示します。なお、記載している環境は推奨環境です。表 2-1 以外の環境を使用する場合は以下の点をご確認ください。

- gcc または clang で C 言語ソースのコンパイルができること。
- gfortran で Fortran 言語ソースのビルドができること。
- NetCDF-C、NetCDF-Fortran が使用できること。

表 2-1 動作環境

		Linux	Windows	Mac
OS		Red Hat Enterprise Linux 8	Windows10	macOS Ventura 13.5.1
コンパイラ	C 言語	cc(4.4.7)	gcc(10.3.0)	clang(14.0.3)
	Fortran77	gfortran(4.8.4)	gfortran(10.3.0)	gfortran(13.1.0)
	Fortran90	gfortran(4.8.4)	gfortran(10.3.0)	gfortran(13.1.0)
使用メモリサイズ		40MB	35MB	35MB
NetCDF-C		4.8.0	4.9.2	4.9.2
NetCDF-Fortran		4.6.1	4.6.1	4.6.1

※使用メモリサイズは L1A 標準プロダクト読み込み時の値

2.2　外部ライブラリのインストール

NFTOOL では以下のライブラリが別途必要です。インストール方法を以下に示します。以下の説明では、外部ライブラリインストール先のディレクトリパスを$LIBS_DIR と表記します。

- NetCDF-C
- NetCDF-Fortran

また、上記のライブラリのインストール時に以下のライブラリが必要になります。必要に応じてインストールします。

- zlib
- szlib
- curl
- HDF5

2.2.1 Linux

- zlib

  https://zlib.net

  上記アドレスにアクセスして、zlib-1.2.11.tar.gz を入手し、任意のディレクトリにて以下のコマンドでインストールします。

```
$ tar xvf zlib-1.2.11.tar.gz
```

```
$ cd zlib-1.2.11
$ ./configure --prefix=$LIBS_DIR
$ make
$ make install
```

- szlib

  https://support.hdfgroup.org/ftp/lib-external/szip/2.1.1/src/

  上記アドレスにアクセスして、szip-2.1.tar.gz を入手し、任意のディレクトリにて以下のコマンドでインストールします。

```
$ tar xvf szip-2.1.tar.gz
$ cd szip-2.1/
$ ./configure --prefix=$LIBS_DIR
$ make
$ make install
```

- curl

  https://curl.se/download/

  上記アドレスにアクセスして、curl-7.79.1.tar.gz を入手し、任意のディレクトリにて以下のコマンドでインストールします。

```
$ tar xvf curl-7.79.1.tar.gz
$ cd curl-7.79.1/
$ ./configure --prefix=$LIBS_DIR --without-ssl
$ make
$ make install
```

- HDF5

  https://portal.hdfgroup.org/display/support/HDF5+1.12.0

  上記アドレスにアクセスして、hdf5-1.12.0.tar.gz を入手し、任意のディレクトリにて以下のコマンドでインストールします。

```
$ tar xvf hdf5-1.12.0.tar.gz
$ cd hdf5-1.12.0/
$ ./configure --prefix=$LIBS_DIR --with-zlib=$LIBS_DIR --with-szlib=$LIBS_DIR --with-default-api-version=v18
$ make
$ make install
```

- NetCDF-C

https://github.com/Unidata/netcdf-c/releases/v4.8.0

上記アドレスにアクセスして、netcdf-c-4.8.0.tar.gz を入手し、任意のディレクトリにて以下のコマンドでインストールします。

```
$ tar xvf netcdf-c-4.8.0.tar.gz
$ cd netcdf-c-4.8.0
$ export LDFLAGS='-L$LIBS_DIR/lib'
$ export CPPFLAGS='-I$LIBS_DIR/include'
$ ./configure --prefix=$LIBS_DIR
$ make
$ make install
```

- NetCDF-Fortran

  https://github.com/Unidata/netcdf-fortran/releases

  上記アドレスにアクセスして、netcdf-fortran-4.6.1.tar.gz を入手し、任意のディレクトリにて以下のコマンドでインストールします。

```
$ tar xvf netcdf-fortran-4.6.1.tar.gz
$ cd netcdf-fortran-4.6.1
$ export LDFLAGS='-L$LIBS_DIR/lib'
$ export CPPFLAGS='-I$LIBS_DIR/include'
$ ./configure --prefix=$LIBS_DIR
$ make
$ make install
```

2.2.2 Windows

cmake でのインストール方法を説明します。

- TDM-GCC-64

  https://jmeubank.github.io/tdm-gcc/download/

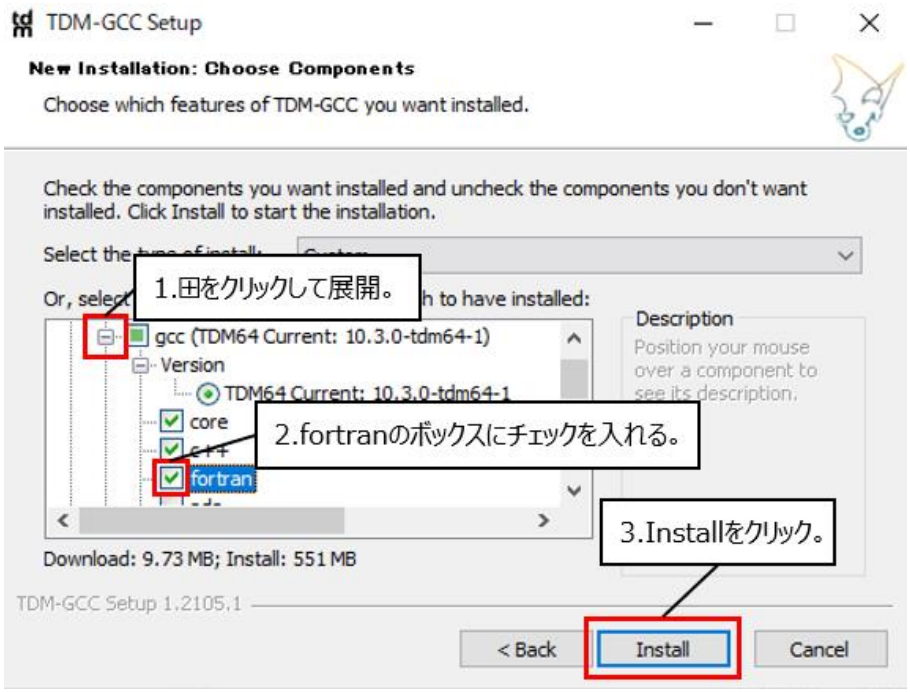
  上記アドレスにアクセスして、tdm64-gcc-10.3.0-2.exe を入手し、以下の手順でインストールします。

  ① tdm64-gcc-10.3.0-2.exe を実行し、「Create」をクリック。

  ② 「MinGW-w64/TDM64 (32bit and 64-bit)」を選択し、「Next」をクリック。

  ③ インストール先を入力し、「Next」をクリック。

  ④ fortran のボックスにチェックを入れ、「Install」をクリック。



⑤　インストールが開始され、「Completed Successfully」が表示されたら、「Next」をクリック。

⑥　「Completing TDM-GCC Setup」と画面に表示されていれば、インストール完了です。

- make

  https://gnuwin32.sourceforge.net/packages/make.htm

  上記アドレスにアクセスして、make-3.81-bin.zip を入手し、任意のディレクトリに展開してください。

- cmake

  https://cmake.org/download/

  上記アドレスにアクセスして、cmake-3.28.0-rc1-windows-x86_64.zip を入手し、任意のディレクトリに展開してください。

- NetCDF-C

  https://downloads.unidata.ucar.edu/netcdf/

  上記アドレスにアクセスして、netCDF4.9.2-NC4-64.exe を入手し、以下の手順でインストールします。

  ①　netCDF4.9.2-NC4-64.exe を実行し、「次へ」をクリック。

  ②　ライセンス契約書を確認し、全ての条件に同意するならば「同意する」をクリック。

  ③　「Do not add netCDF to the system PATH」を選択し、「次へ」をクリック。

  ④　任意のインストール先を入力して、「次へ」をクリック。

  ⑤　「次へ」をクリック。

  ⑥　全てのコンポーネントにチェックが入っていることを確認し、「インストール」をクリック。

  ⑦　インストールが開始され、「NetCDF 4.9.2 セットアップウィザードは完了しました。」と表示される画面に遷移すれば、インストールは完了です。

- 環境変数の設定

  以下のパスを環境変数”Path”へ追加してください。

  - TDM-GCC-64 バイナリパス
  - make バイナリパス
  - cmake バイナリパス
  - NetCDF-C バイナリパス

- NetCDF-Fortran

  https://github.com/Unidata/netcdf-fortran/releases

  上記アドレスにアクセスして、netcdf-fortran-4.6.1.zip を入手し、任意のディレクトリに展開後、以下のコマンドでインストールします。

```
$ cd netcdf-fortran-4.6.1
$ mkdir build
$ cd build
$ cmake ..¥ -G “MinGW Makefiles” -DnetCDF_LIBRARIES=[“netcdf.lib”のパス(″/″区切り)] -DnetCDF_INCLUDE_DIR=[NetCDF-C ライブラリインクルードパス(″/″区切り)]
$ cmake -build . --config RELEASE
$ cmake --build . --config RELEASE --target install （要管理者権限）
```

2.2.3 Mac

Homebrew 経由でライブラリをインストールします。

- Homebrew

  https://brew.sh/

  上記 HP に書かれているコマンドを実行し、homebrew をインストールします。

- NetCDF-C

  以下のコマンドを実行してインストールします。

  ```
  $ brew install netcdf
  ```

- NetCDF-Fortran

  以下のコマンドを実行してインストールします。

  ```
  $ brew install netcdf-fortran
  ```

2.3 NFTOOL のインストール

NFTOOL のインストール方法を以下に示します。以下の説明では、インストール先のディレクトリパスを $INSTALL_DIR と表記します。

2.3.1 Linux

① ファイルの展開

任意のディレクトリに AMSR3_Fortran_Ver[バージョン番号].tar.gz ファイルをコピーし、展開します。

```
$ tar xzf AMSR3_Fortran_Ver[バージョン番号].tar.gz
```

上記コマンドを実行すると、4.1 項に示すディレクトリ・ファイルが展開されます。

② Configure

ライブラリの作成に必要な Makefile を作成します。“$INSTALL_DIR/AMSR3_Fortran/NFTOOL”に移動し、以下のコマンドを実行します。

```
$ ./configure [オプション]
```

指定するオプションは以下のとおりです。

--prefix：

NFTOOL ライブラリ”libAMSR3.a”を配置するディレクトリのパス。4.1 項のとおりに配置する場合、以下を指定する。

```
--prefix=$INSTALL_DIR/AMSR3_Fortran/NFTOOL/lib
```

また、必要に応じて以下を指定してください。

--with-nc-lib：

NetCDF-C ライブラリ“libnetcdf.a”のあるディレクトリパス。”/*/*/lib”または“/*/*/*/lib”に配置されている場合、指定不要。

--with-nc-include：

NetCDF-C ヘッダファイル“netcdf.h”のあるディレクトリパス。”/*/*/include”または“/*/*/*/include”に配置されている場合、指定不要。

configure が成功すると、カレントディレクトリに Makefile が作成されます。

③ C 言語ライブラリのコンパイル

続いて、NFTOOL ライブラリ”libAMSR3.a”の作成を行います。

“$INSTALL_DIR/AMSR3_Fortran/NFTOOL”で以下のコマンドを実行します。

```
$ make
```

--prefix にて指定したパスに”libAMSR3.a”が作成されていれば、NFTOOL ライブラリの作成は終了で

す。以下のコマンドで確認できます。

```
$ ls -l $INSTALL_DIR/AMSR3_Fortran/NFTOOL/lib (--prefix にて指定したパス)
```

2.3.2 Windows

① ファイルの展開

任意のディレクトリに AMSR3_Fortran_Ver[バージョン番号].zip ファイルをコピーし、展開します。4.1 項に示すディレクトリ・ファイルが展開されます。

② 環境変数の設定

NFTOOL ライブラリ”libAMSR3.a”の作成に必要な環境変数を設定します。

“$INSTALL_DIR/AMSR3_Fortran/_setEnv.bat”をテキストエディタ等で開き、以下の環境変数の値を書き換えて保存してください。

```
set NC_LIB_PATH=[NetCDF-C ライブラリパス]
set NCINC=%NC_LIB_PATH%¥include (“netcdf.h”のあるディレクトリパス)
set NCLIB=%NC_LIB_PATH%¥bin (“netcdf.dll”のあるディレクトリパス)
```

_setEnv.bat を実行し、環境変数を有効化します。

さらに、%NCINC%、%NCLIB%で設定したディレクトリパスを%PATH%に追加します。

③ C 言語ライブラリのコンパイル

NFTOOL ライブラリ”libAMSR3.a”を作成します。“$INSTALL_DIR/AMSR3_Fortran/NFTOOL”で以下のコマンドを実行します。

```
$ make
```

以下のコマンドで”libAMSR3.a”が作成されていることが確認できれば、NFTOOL ライブラリの作成は終了です。

```
$ dir $INSTALL_DIR¥AMSR3_Fortran¥NFTOOL¥lib
```

2.3.3 Mac

インストール手順は 2.3.1 項と同様です。

3　使用方法

3.1　ライブラリを使用したプログラミングの流れ

NFTOOL ライブラリを使用したプログラミングの流れについて説明します。サンプルコードを実行する場合は 3.3 項を参照してください。

3.1.1 Fortran ソースコードの作成

① ヘッダファイルインクルード

NFTOOL ライブラリで使用するヘッダファイルをインクルードします。

Fortran77 の場合は“AM3TK_f77.h”を、Fortran90 の場合は“AM3TK_f90.h”を指定します。

② 変数定義

読み込んだアトリビュートの名称や値を保持する変数を定義します。

③ NetCDF ファイルオープン

NetCDF ファイルを開き、NetCDF ID を取得します。

④ グローバル/データセットアトリビュート読み込み

4.2 項を参考に、アトリビュート名やデータ型、データ長を取得します。NFTOOL ライブラリのメソッドはステータスを戻り値とする function、またはステータスを引数とする subroutine として使用できます。アトリビュート取得対象のデータセットは変数 ID を使って指定します。グローバルアトリビュートは変数 ID に「NF_GLOBAL」を、データセットアトリビュートは該当する変数 ID を指定してください。

⑤ NetCDF ファイルクローズ

NetCDF ID を指定し、NetCDF ファイルを閉じます。

3.1.2 ライブラリのリンク

Fortran ソースコードをコンパイルするときは、NetCDF ライブラリに加えて、NFTOOL ライブラリ(libAMSR3.a)をオプションで指定する必要があります。

gfortran の場合の例を以下に示します。

```
gfortran [ソース] -I[NerCDF-C_INC] -I[NerCDF-Fortran_INC] -I[NFTOOL_INC] -L[NerCDF-C_LIB] -L[NerCDF-Fortran_LIB] -L[NFTOOL_LIB] -lnetcdf -lnetcdff -lAMSR3 -o [ターゲット]
```

3.1.3 プログラム実行

プログラムの実行時には、4.5 項の環境変数を設定してください。

3.2 Fortran サンプルプログラムの説明

NFTOOL で用意しているサンプルプログラムを表 3-1、表 3-2 に示します。「使用する関数」欄にある番号は 4.2 項にある関数一覧の番号に対応しています。nf90_open、nf90_close、nf_open, nf_close は NetCDF-Fortran の関数に対応します。

表 3-1 Fortran77 サンプルコード一覧

ファイル名	実行形式名	読み込みプロダクト	使用する関数(オープン、クローズ)	使用する関数(読み込み部分)
sample1_read_L1product.f	sample1	L1 プロダクト	F01, F02	F03, F08
sample2_read_L1product.f	sample2	L1 プロダクト	nf_open, nf_close	F03～F07
sample3_read_L2product.f	sample3	L2 プロダクト	nf_open, nf_close	S03～S07
sample4_read_L2product.f	sample4	L2 プロダクト	nf_open, nf_close	S03, S08
sample5_read_L3product.f	sample5	L3 プロダクト	S01, S02	S03～S07

表 3-2 Fortran90 サンプルコード一覧

ファイル名	実行形式名	読み込みプロダクト	使用する関数(オープン、クローズ)	使用する関数(読み込み部分)
sample1_read_L1product.f90	sample1	L1 プロダクト	F01, F02	F03, F08
sample2_read_L1product.f90	sample2	L1 プロダクト	nf90_open, nf90_close	F03～F07
sample3_read_L2product.f90	sample3	L2 プロダクト	nf90_open, nf90_close	S03～S07
sample4_read_L2product.f90	sample4	L2 プロダクト	nf90_open, nf90_close	S03, S08
sample5_read_L3product.f90	sample5	L3 プロダクト	S01, S02	S03～S07

3.3　サンプルプログラムの実行

NFTOOL に格納されている Fortran77、Fortran90 のサンプルコードの実行方法を示します。

3.3.1 Linux

① Configure

サンプルコード実行形式の作成に必要な Makefile を作成します。

“$INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/Fortran77(Fortran)”に移動し、以下のコマンドを実行します。

```
$ ./configure [オプション]
```

指定するオプションは以下のとおりです。

--prefix：

サンプルコード実行形式(sample[N])を配置するディレクトリのパス。4.1 項のとおりに配置する場合、以下を指定する。

```
--prefix=$INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/Fortran77(Fortran)
```

また、必要に応じて以下を指定してください。

--with-nc-lib：

NetCDF-C ライブラリ“libnetcdf.a”のあるディレクトリパス。”/*/*/lib”または“/*/*/*/lib”に配置されている場合、指定不要。

--with-nc-include：

NetCDF-C ヘッダファイル“netcdf.h”のあるディレクトリパス。”/*/*/include”または“/*/*/*/include”に配置されている場合、指定不要。

--with-nf-lib：

NetCDF-Fortran ライブラリ“libnetcdff.a”のあるディレクトリパス。”/*/*/lib”または“/*/*/*/lib”に配置されている場合、指定不要。

--with-nf-include：

NetCDF-Fortran ヘッダファイル“netcdf.inc”のあるディレクトリパス。”/*/*/include”または“/*/*/*/include”に配置されている場合、指定不要。

configure が成功すると、カレントディレクトリに Makefile が作成されます。

② 環境変数の設定

サンプルコードの実行時に使用する環境変数を設定します。

“$INSTALL_DIR/AMSR3_Fortran/_setEnv.bash”をテキストエディタ等で開き、以下の環境変数の値を書き換えて保存してください。

```
export　LD_LIBRARY_PATH=″$LD_LIBRARY_PATH:/usr/local/lib:/usr/local/lib64:[NetCDF-C 、NetCDF-Fortran ライブラリ格納ディレクトリ]″
```

“$INSTALL_DIR/AMSR3_Fortran/”で以下のコマンドを実行し、環境変数を有効化します。

```
$ source _setEnv.bash
```

起動時に環境変数を設定したい場合は、~/.bashrc に_setEnv.bash の内容を追記してください。

③ サンプルコードの編集

指定したプロダクトを読み込むようにサンプルコードを編集します。

まず、任意のディレクトリにプロダクトファイルを格納します。4.1 項に示すように、NFTOOL では“$INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/data”をプロダクトファイル格納用のディレクトリとして用意しています。

テキストエディタ等でサンプルコードを開き、文字列型の変数”fname”の値に読み込み対象のプロダクトファイルのパスを記載します。さらに、記載したファイルパスの文字数を character の配列サイズに指定します。(下の例では 59)

<sample1_read_L1product.f>

```
～～～
      character*59 fname
      data fname
     *     /'../data/GGWAM3_202107010843A033_S1ADNA00Z01A23024.nc'
     *     /
～～～
```

<sample1_read_L1product.f90>

```
～～～
      character*59 fname
      data fname &
     &/'../data/GGWAM3_202107010843A033_S1ADNA00Z01A23024.nc'&
     &     /
～～～
```

ファイルを保存し、編集は完了です。

④ サンプルコードのコンパイル

サンプルコードの実行形式の作成を行います。

“$INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/Fortran77(Fortran)”で以下のコマンドを実行します。

```
$ make
```

--prefix にて指定したパスに”sample[N]”が作成されていれば、サンプルコードの実行形式の作成は終了です。以下のコマンドで確認できます。

```
$ ls -l $INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/Fortran77(Fortran) (--prefix にて指定したパス)
```

make に失敗した場合、3.3.4 項を参照してください。

3.3.2 Windows

① 環境変数の設定

サンプルコードのコンパイル時や実行時に使用する環境変数を設定します。

“$INSTALL_DIR/AMSR3_Fortran/_setEnv.bat”をテキストエディタ等で開き、以下の環境変数の値を書き換えて保存してください。

```
set NC_LIB_PATH=[NetCDF-C ライブラリパス]
set NF_LIB_PATH=[NetCDF-Fortran ライブラリパス]
set NCINC=%NC_LIB_PATH%¥include (“netcdf.h”のあるディレクトリパス)
set NFINC=%NF_LIB_PATH%¥include (“netcdf.inc”のあるディレクトリパス)
set NCLIB=%NC_LIB_PATH%¥bin (“netcdf.dll”のあるディレクトリパス)
set NFLIB=%NF_LIB_PATH%¥bin (“libnetcdff.dll”のあるディレクトリパス)
```

_setEnv.bat を実行し、環境変数を有効化します。

さらに、%NCINC%、%NCLIB%、%NFINC%、%NFLIB%で設定したディレクトリパスを PATH に追加します。

② サンプルコードの編集

指定したプロダクトを読み込むようにサンプルコードを編集します。

まず、任意のディレクトリにプロダクトファイルを格納します。4.1 項に示すように、NFTOOL では“$INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/data”をプロダクトファイル格納用のディレクトリとして用意しています。

テキストエディタ等でサンプルコードを開き、文字列型の変数”fname”の値に読み込み対象のプロダクトファイルのパスを記載します。さらに、記載したファイルパスの文字数を character の配列サイズに指定します。(下の例では 59)

<sample1_read_L1product.f>

```
～～～
      character*59 fname
      data fname
     *     /'../data/GGWAM3_202107010843A033_S1ADNA00Z01A23024.nc'
     *     /
～～～
```

<sample1_read_L1product.f90>

```
～～～
        character*59 fname
        data fname &
       &/'../data/GGWAM3_202107010843A033_S1ADNA00Z01A23024.nc'&
       &      /
～～～
```

ファイルを保存し、編集は完了です。

③ サンプルコードのコンパイル

サンプルコードの実行形式の作成を行います。

“$INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/Fortran77(Fortran)”で以下のコマンドを実行します。

```
$ make
```

以下のコマンドで”sample[N].exe” が作成されていることが確認できれば、サンプルコードの実行形式の作成は終了です。

```
$ dir $INSTALL_DIR¥AMSR3_Fortran¥NFTOOL¥sample¥Fortran77(Fortran)
```

make に失敗した場合、3.3.4 項を参照してください。

3.3.3 Mac

サンプルプログラム作成手順は 3.3.1 項と同様です。

3.3.4 コンパイルエラーとなる場合

gcc のバージョンによって、サンプルコードの make にてエラーが発生する場合があります。対処方法は以下を参考にしてください。

- Error: BOZ literal constant at (1) cannot appear …
  次のコンパイルオプションを追加してください。
  -fallow-invalid-boz

- Error: Type mismatch between actual argument at…and actual argument at…
  次のコンパイルオプションを追加してください。
  -fallow-argument-mismatch

4　参考

4.1　ディレクトリ構成

AMSR3_Fortran
- ├ _setenv.bash(_setenv.bat)： 環境変数ファイル
- └ NFTOOL/
  - ├ Makefile：NFTOOL ライブラリを作成するための Makefile　※1
  - ├ Makefile.in：Makefile のひな型
  - ├ autom4te.cache/：automake 関連ファイル格納ディレクトリ
  - ├ autoscan.log：automake 関連ファイル
  - ├ config.guess：automake 関連ファイル
  - ├ config.log：configure 関連ファイル
  - ├ config.status：configure 関連ファイル
  - ├ config.sub：automake 関連ファイル
  - ├ configure：configure ファイル
  - ├ configure.ac：automake 関連ファイル
  - ├ configure.scan：automake 関連ファイル
  - ├ install-sh：automake 関連ファイル
  - ├ include/
  - │　├ AM3TK_f77.h：Fortran77 サンプルコード用ヘッダファイル
  - │　└ AM3TK_f90.h：Fortran90 サンプルコード用ヘッダファイル
  - ├ lib/
  - │　└ libAMSR3.a：NFTOOL ライブラリ
  - ├ sample/
  - │　├ Fortran77/：Fortran77 用サンプルコード格納ディレクトリ
  - │　│　├ Makefile： サンプルコード実行形式を作成するための Makefile　※1
  - │　│　├ Makefile.in
  - │　│　├ config.log
  - │　│　├ config.status
  - │　│　├ configure
  - │　│　├ sample1(sample1.exe)
  - │　│　├ sample1_read_L1product.f
  - │　│　├ sample2(sample2.exe)
  - │　│　├ sample2_read_L1product.f
  - │　│　├ sample3(sample3.exe)
  - │　│　├ sample3_read_L2product.f
  - │　│　├ sample4(sample4.exe)
  - │　│　├ sample4_read_L2product.f

```
|   |   ├ sample5(sample5.exe)
|   |   └ sample5_read_L3product.f
|   ├ Fortran/：Fortran90 用のサンプルコード格納ディレクトリ
|   |   ├ Makefile： サンプルコード実行形式を作成するための Makefile  ※1
|   |   ├ Makefile.in
|   |   ├ config.log
|   |   ├ config.status
|   |   ├ configure
|   |   ├ sample1(sample1.exe)
|   |   ├ sample1_read_L1product.f90
|   |   ├ sample2(sample2.exe)
|   |   ├ sample2_read_L1product.f90
|   |   ├ sample3(sample3.exe)
|   |   ├ sample3_read_L2product.f90
|   |   ├ sample4(sample4.exe)
|   |   ├ sample4_read_L2product.f90
|   |   ├ sample5(sample5.exe)
|   |   └ sample5_read_L3product.f90
|   └ data/ ： サンプルコード用のプロダクトファイル格納ディレクトリ
└ src/：NFTOOL ライブラリ作成用のソースコード、ヘッダファイル格納ディレクトリ
```

図 4-1 ディレクトリ構成

※1 configure コマンドを実行すると作成・上書きされます。

4.2　関数一覧

本項では NFTOOL のライブラリ関数の詳細を示します。関数の番号が”F”で始まる場合は Fortran の function として、”S”で始まる場合は Fortran の subroutine として使用することができます。

F01			
status = nftool_open_read_func(fname, nc_id)			
NetCDF ファイルを読み込み専用で開き、NetCDF ID を取得する。			
項目	型	変数名	説明
入力	character*N	fname	プロダクトファイルパス
出力	integer	nc_id	NetCDF ID
戻り値	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)

S01			
call nftool_open_read_sub (fname, nc_id, status)			
NetCDF ファイルを読み込み専用で開き、NetCDF ID を取得する。			
項目	型	変数名	説明
入力	character*N	fname	プロダクトファイルパス
出力	integer	nc_id	NetCDF ID
	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)
戻り値	–	–	–

F02			
status = nftool_close_func(nc_id)			
指定した NerCDF ID のファイルを閉じる。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
出力	–	–	–
戻り値	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)

S02			
call nftool_close_sub(nc_id, status)			
指定した NerCDF ID のファイルを閉じる。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
出力	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)
戻り値	–	–	–

F03			
status = nftool_getAttr_func(nc_id, varid, attr_name, attr_val)			
指定したアトリビュート名の値を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
	character*N	attr_name	アトリビュート名
出力	※1	attr_val	アトリビュート値
戻り値	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)

※1 取得するアトリビュート値に合わせたデータ型を指定する

S03			
call nftool_getAttr_sub(nc_id, varid, attr_name, attr_val, status)			
指定したアトリビュート名の値を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
	character*N	attr_name	アトリビュート名
出力	※1	attr_val	アトリビュート値
	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)
戻り値	-	-	-

※1 取得するアトリビュート値に合わせたデータ型を指定する

F04			
status = nftool_getAttrType_func(nc_id, varid, attr_name, attr_type)			
指定したアトリビュート名のアトリビュートのデータ型を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
	character*N	attr_name	アトリビュート名
出力	integer	attr_type	アトリビュートのデータ型に対応した整数値(4.4 項参照)
戻り値	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)

S04			
call nftool_getAttrType_sub(nc_id, varid, attr_name, attr_type, status)			
指定したアトリビュート名のアトリビュートのデータ型を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
	character*N	attr_name	アトリビュート名
出力	integer	attr_type	アトリビュートのデータ型に対応した整数値(4.4 項参照)
	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)
戻り値	–	–	–

F05			
status = nftool_getAttrLen_func(nc_id, varid, attr_name, attr_len)			
指定したアトリビュート名のアトリビュートの配列長を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
	character*N	attr_name	アトリビュート名
出力	integer	attr_len	アトリビュートの配列長
戻り値	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)

S05			
call nftool_getAttrLen_sub(nc_id, varid, attr_name, attr_len, status)			
指定したアトリビュート名のアトリビュートの配列長を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
	character*N	attr_name	アトリビュート名
出力	integer	attr_len	アトリビュートの配列長
	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)
戻り値	-	-	-

F06			
status = nftool_getNumAttr_func(nc_id, varid, num_attr)			
指定した変数 ID のアトリビュートの総数を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
出力	integer	num_attr	アトリビュートの総数
戻り値	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)

S06			
call nftool_getNumAttr_sub(nc_id, varid, num_attr, status)			
指定した変数 ID のアトリビュートの総数を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
出力	integer	num_attr	アトリビュートの総数
	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)
戻り値	–	–	–

F07			
status = nftool_getAttrName_func(nc_id, varid, attid, attr_name)			
指定したアトリビュート番号のアトリビュート名を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
	integer	attid	アトリビュート番号(1 以上)
出力	character*N	attr_name	アトリビュート名
戻り値	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)

S07			
call nftool_getAttrName_sub(nc_id, varid, attid, attr_name, status)			
指定したアトリビュート番号のアトリビュート名を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
	integer	attid	アトリビュート番号(1 以上)
出力	character*N	attr_name	アトリビュート名
	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)
戻り値	-	-	-

F08			
status = nftool_getAttrName_all_func(nc_id, varid, attr_name_list)			
指定した変数 ID に含まれるすべてのアトリビュート名の配列を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
出力	character*N(M)	attr_name_list	アトリビュート名の一覧
戻り値	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)

S08			
call nftool_getAttrName_all_sub(nc_id, varid, attr_name_list, status)			
指定した変数 ID に含まれるすべてのアトリビュート名の配列を取得する。varid に”NF_GLOBAL”を指定すると、グローバルアトリビュートを対象とする。			
項目	型	変数名	説明
入力	integer	nc_id	NetCDF ID
	integer	varid	変数 ID(0 以上)
出力	character*N(M)	attr_name_list	アトリビュート名の一覧
	integer	status	正常時:0 異常時:エラーコード(4.3 項参照)
戻り値	–	–	–

4.3　戻り値

NFTOOL ライブラリで使用するエラー番号には以下のものがあります。

番号	説明	備考
-300	格納先メモリサイズ不足	配列型のアトリビュートを取得する変数の配列数が足りていない場合に発生する。
-301	格納先不定	ファイル名やアトリビュート名などを格納する変数のメモリ数が足りていない場合に発生する。
-400	メモリ確保異常	MALLOC に失敗した場合に発生する。

上記以外の番号は NetCDF ライブラリ(C 言語用)が定義するエラーです。詳細は以下を参照してください。
NetCDF: netcdf.h File Reference (ucar.edu)

NFTOOL ライブラリでは、NetCDF ライブラリ(C 言語用)が送出するエラーはそのまま戻り値として使用されます。

4.4　ライブラリ変数

ヘッダファイル”AM3TK_f77.h”、”AM3TK_f90.h”で定義している変数、構造体を以下に示します。使用例はサンプルコードを参考にしてください。

〈データ型〉

変数名	型	値	説明	備考
AM3_DTYPE_CHAR	integer	2	文字列型	
AM3_DTYPE_SHORT	integer	3	符号あり 2byte 整数	
AM3_DTYPE_INT	integer	4	符号あり 4byte 整数	
AM3_DTYPE_FLOAT	integer	5	32bit 実数	
AM3_DTYPE_DOUBLE	integer	6	64bit 実数	
AM3_DTYPE_UBYTE	integer	7	符号なし 1byte 整数	
AM3_DTYPE_USHORT	integer	8	符号なし 2byte 整数	

〈データセットアトリビュート(変数)〉

変数名	型	値	説明	備考
DEF_NUM_ATT	integer	16	データセットアトリビュート名称の種類の数	

〈データセットアトリビュート(構造体)〉 ※"AM3TK_f90.h"のみ

構造体定義	メンバ	型	説明	備考
global_attr (グローバルアトリビュート)	attname	character*50	名称	ex) processing_level
	attlen	integer	配列長	ex) 7
	atttype	integer	データ型	ex) 2
	val_str	character*3080	文字列型値	ex) Level1A
	val_ubyte	integer*2	ubyte 型値	
	val_short	integer*2	short 型値	
	val_ushort	integer*4	ushort 型値	
	val_f	real*4	float 型値	
	val_d	real*8	double 型値	
	val_int	integer*4	int 型値	
dataset_attr (データセットアトリビュート)	attname	character*20	名称	ex) flag_masks
	attlen	integer	配列長	ex) 2
	atttype	integer	データ型	ex) 4
	val_str	character*1024	文字列型値	
	val_ubyte	Integer*2	ubyte 型値	
	val_short	Integer*2	short 型値	
	val_ushort	integer*4	ushort 型値	
	val_f	real*4	float 型値	
	val_d	real*8	double 型値	
	val_int(10)	integer*4	int 型値	ex) 4, 128
dataset (データセット)	ds_name	character*50	データセット名	ex) ObsCount_Ch06H_Quality
	ds_attr	dataset_attr	データセットアトリビュート	

4.5 環境変数

プログラムの実行時には以下の環境変数を参照します。必要となるライブラリの検索パスを追記して使用します。

OS	変数名	内容	備考
Linux, Mac	LD_LIBRARY_PATH	NetCDF ライブラリ(Fortran, C 言語)パス	
Windows	PATH	NetCDF ライブラリ(Fortran, C 言語)パス	