AMSR3 Product Attribute Reading Tool User Manual

October 25, 2023

Table of Contents

1 Introduction

1.1 Purpose of the Manual

The AMSR3 Product Attribute Reading Tool (referred to as NFTOOL, which stands for NetCDF Fortran Tool) is used to access global attributes and dataset attributes of AMSR3 products from Fortran77 and Fortran90. Figure 1-1 shows an overview of NFTOOL.

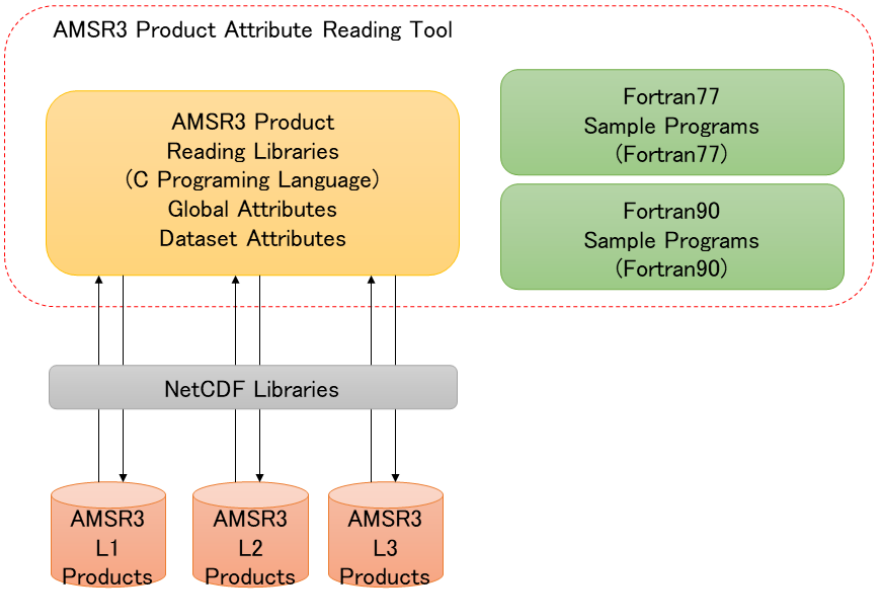


Figure 1-1 Overview of NFTOOL

2 Installation

2.1 Operating Environment

The operating environment for NFTOOL is shown in Table 2-1. This is the recommended environment. If using a different environment, you should be able to:

- Compile C language source code with gcc or clang.
- Build Fortran language source code with gfortran.
- Use NetCDF-C and NetCDF-Fortran Libraries.

Table 2-1 Operating Environment

		Linux	Windows	Mac
OS		Red Hat Enterprise Linux 8	Windows10	macOS Ventura 13.5.1
Compiler	C Language	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)
Used Memory Size (*1)		40 MB	35 MB	35 MB
NetCDF-C		4.8.0	4.9.2	4.9.2
NetCDF-Fortran		4.6.1	4.6.1	4.6.1

*1: The memory size used to load L1A standard product.

2.2 Installation of External Libraries

The following libraries are required for NFTOOL. The installation method is explained below. In the following explanation, the directory path for installing external libraries is represented as $LIBS_DIR.

- NetCDF-C
- NetCDF-Fortran

Installation of the following libraries is also required during the installation of the NetCDF libraries.

- zlib
- szlib
- curl
- HDF5

2.2.1 Linux

- zlib

  https://zlib.net

  Follow the link provided above. Then, get zlib-1.2.11.tar.gz and install it using the following command in any directory.

```
$ 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/

  Follow the link provided above. Then, get szip-2.1.tar.gz and install it using the following command in any directory.

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

- curl

  https://curl.se/download/

  Follow the link provided above. Then, get curl-7.79.1.tar.gz and install it using the following command in any directory.

```
$ 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

  Follow the link provided above. Then, get hdf5-1.12.0.tar.gz and install it using the following command in any directory.

```
$ 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

  Follow the link provided above. Then, get netcdf-c-4.8.0.tar.gz and install it using the following

command in any directory.

```
$ 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

  Follow the link provided above. Then, get netcdf-fortran-4.6.1.tar.gz and install it using the following command in any directory.

```
$ 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

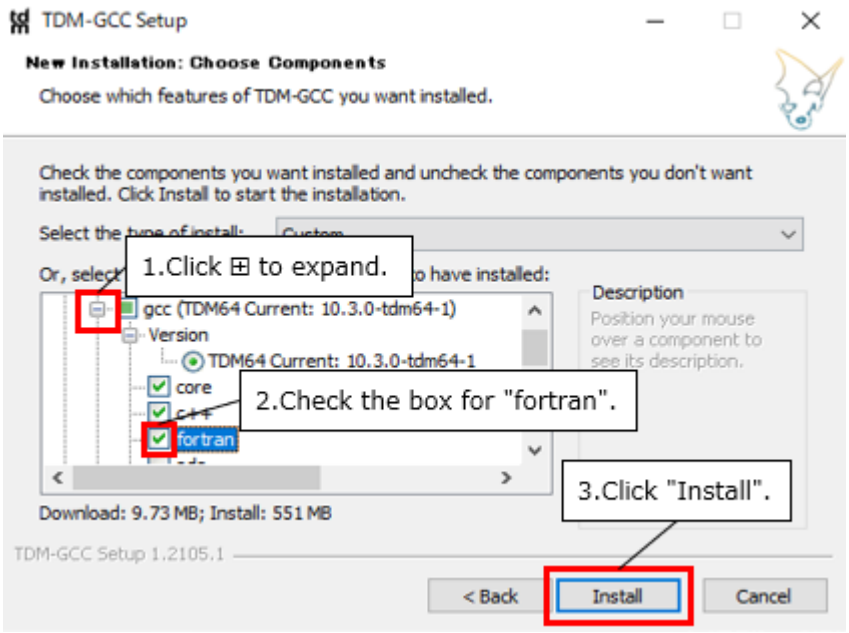
Here is an explanation of the installation method using cmake.

- TDM-GCC-64

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

  Follow the link provided above. Then, get netcdf-fortran-4.6.1.tar.gz and install it using the following steps in any directory.

① Execute tdm64-gcc-10.3.0-2.exe and click "Create".
② Select "MinGW-w64/TDM64 (32bit and 64-bit)" and click "Next".
③ Enter the destination folder and click "Next".
④ Check the box for "fortran" and click "Install".



⑤ After the installation starts, click "Next" when "Completed Successfully" is displayed.

⑥ If the screen shows "Completing TDM-GCC Setup", the installation is complete.

- Make

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

  Follow the link provided above. Then, get make-3.81-bin.zip and extract it to any directory.

- cmake

  https://cmake.org/download/

  Follow the link provided above. Then, get cmake-3.28.0-rc1-windows-x86_64.zip and extract it to any directory.

- NetCDF-C

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

  Follow the link provided above. Then, get netcdf-fortran-4.6.1.tar.gz and install it using the following steps in any directory.

① Execute netCDF4.9.2-NC4-64.exe and click "Next".

② Check the license agreement and click "Agree" if you agree to all the terms.

③ Select "Do not add netCDF to the system PATH" and click "Next".

④ Enter the destination folder and click "Next".

⑤ Click "Next".

⑥ Make sure that all components are checked and click "Install".

⑦ If the screen shows " NetCDF 4.9.2 Setup Wizard has completed", the installation is complete.

- Setting the Environment Variables

  Add the following paths to the "Path" environment variable.

  - TDM-GCC-64 binary path
  - make binary path
  - cmake binary path
  - NetCDF-C binary path

- NetCDF-Fortran

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

  Follow the link provided above. Then, get netcdf-fortran-4.6.1.zip and after extracting it to any directory, install it with the following command.

```
$ cd netcdf-fortran-4.6.1
$ mkdir build
$ cd build
$ cmake ..¥ -G “MinGW Makefiles” -DnetCDF_LIBRARIES=[Path to "netcdf.lib"(“/” as a delimiter).] -DnetCDF_INCLUDE_DIR=[NetCDF-CLibrary include path(“/” as a delimiter).]
$ cmake --build . --config RELEASE
$ cmake --build . --config RELEASE --target install （Administrator privileges are required）
```

2.2.3 Mac

Install the library via Homebrew by executing the following command.

- Homebrew

  https://brew.sh/

  Execute the commands written on the above website to install Homebrew.

- NetCDF-C

  Execute the following command to install.

```
$ brew install netcdf
```

- NetCDF-Fortran

  Execute the following command to install.

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

2.3 Installation of NFTOOL

The installation method for NFTOOL is explained below. In the following instructions, the directory path for the installation destination will be denoted as $INSTALL_DIR.

2.3.1 Linux

① Extract the File.

Copy AMSR3_Fortran_Ver[Version Number].tar.gz file to any directory and extract it.

```
$ tar xzf AMSR3_Fortran_Ver[Version Number].tar.gz
```

When you execute the above command, the directories and files mentioned in section 4.1 will be extracted.

② Configure

Create a Makefile necessary for creating a library.

Change directory to "$INSTALL_DIR/AMSR3_Fortran/NFTOOL" and execute the following command.

```
$ ./configure [Option]
```

The specified options are as follows.

--prefix :

Specify the path to the directory where NFTOOL library "libAMSR3.a" will be placed. If you follow the instructions in section 4.1, specify the following.

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

Also, specify the following as needed.

--with-nc-lib :

Specify the directory path where the NetCDF-C library "libnetcdf.a" is located. If it is located in "/*/*/lib" or "/*/*/*/lib", no need to specify.

--with-nc-include :

Specify the directory path where the NetCDF-C header file "netcdf.h" is located. If it is located in "/*/*/include" or "/*/*/*/include", no need to specify.

When the configure process is successful, a Makefile will be created in the current directory.

③ Compiling C libraries.

Execute the following command at "$INSTALL_DIR/AMSR3_Fortran/NFTOOL" to create the NFTOOL library "libAMSR3.a".

```
$ make
```

If "libAMSR3.a" has been created at the path specified by "--prefix", then the creation of the NFTOOL

library is complete. You can verify this using the following command.

```
$ ls -l $INSTALL_DIR/AMSR3_Fortran/NFTOOL/lib (the path specified with --prefix)
```

2.3.2 Windows

① File Extraction

Copy AMSR3_Fortran_Ver[Version Number].zip file to any directory and extract it. The directory and files mentioned in section 4.1 will be extracted.

② Setting environment variables

Edit "$INSTALL_DIR/AMSR3_Fortran/_setEnv.bat" to modify the following environment variables needed to create "libAMSR3.a".

```
set NC_LIB_PATH=[ NetCDF-C library path]
set NCINC=%NC_LIB_PATH%¥include (Directory path where "netcdf.h" is located)
set NCLIB=%NC_LIB_PATH%¥bin (Directory path where “netcdf.dll” is located)
```

Run "_setEnv.bat" to activate the environment variables., and add the directory paths set with %NCINC% and %NCLIB% to the %PATH%.

③ Compile C language library.

Execute the following command at "$INSTALL_DIR/AMSR3_Fortran/NFTOOL" to create "libAMSR3.a".

```
$ make
```

If "libAMSR3.a" has been created at "$INSTALL_DIR/AMSR3_Fortran/NFTOOL", the creation of the NFTOOL library is complete.

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

2.3.3 Mac

The installation procedure is the same as described in section 2.3.1.

3 How To Use The Tool

3.1 Programming steps using NFTOOL

Here is an explanation of the programming steps using the NFTOOL library. Please refer to section 3.3 if you want to execute the sample code.

3.1.1 Creating Fortran Source Code

① Including Header Files

To include the header file for the NFTOOL library, specify "AM3TK_f77.h" for Fortran 77 and "AM3TK_f90.h" for Fortran 90.

② Variable definition

Define variables to store the names and values of the loaded attributes.

③ Open NetCDF file

Open the NetCDF file and get the NetCDF ID.

④ Read global/dataset attributes.

Get attribute names, data types, and data lengths. Section 4.2 will serve as a reference. The methods in the NFTOOL library can be used as functions that return the status or as subroutines that take the status as an argument. You can specify the dataset for attribute retrieval using the variable ID. For global attributes, use "NF_GLOBAL" as the variable ID, and for dataset attributes, specify the corresponding variable ID.

⑤ Close the NetCDF file

Specify the NetCDF ID and close the NetCDF file.

3.1.2 Linking Libraries

When compiling Fortran source code, it is necessary to specify the NFTOOL library (libAMSR3.a) "-lAMSR3" as well as the NetCDF libraries.

Here is an example for gfortran.

```
gfortran [Source] -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 [Target]
```

3.1.3 Executing Programs

Set the environment variables mentioned in section 4.5 before running the program.

3.2 Fortran Sample Program

The sample programs provided by NFTOOL are shown in Table 3-1 and Table 3-2. The symbols such as F01, F02, etc. in the "Functions Used" column correspond to the symbols in the function list in section 4.2. Function nf90_open, nf90_close, nf_open, and nf_close correspond to the functions in NetCDF-Fortran.

Table 3-1 List of sample codes for Fortran77

File Name	Executable file name.	Read Products	Functions Used (Open, Close)	Functions Used (Reading Part)
sample1_read_L1product.f	sample1	L1 Product	F01, F02	F03, F08
sample2_read_L1product.f	sample2	L1 Product	nf_open, nf_close	F03～F07
sample3_read_L2product.f	sample3	L2 Product	nf_open, nf_close	S03～S07
sample4_read_L2product.f	sample4	L2 Product	nf_open, nf_close	S03, S08
sample5_read_L3product.f	sample5	L3 Product	S01, S02	S03～S07

Table 3-2 List of sample codes for Fortran90

File Name	Executable file name.	Read Products	Functions Used (Open, Close)	Functions Used (Reading Part)
sample1_read_L1product.f90	sample1	L1 Product	F01, F02	F03, F08
sample2_read_L1product.f90	sample2	L1 Product	nf90_open, nf90_close	F03～F07
sample3_read_L2product.f90	sample3	L2 Product	nf90_open, nf90_close	S03～S07
sample4_read_L2product.f90	sample4	L2 Product	nf90_open, nf90_close	S03, S08
sample5_read_L3product.f90	sample5	L3 Product	S01, S02	S03～S07

3.3 Executing Sample Programs

Here is how to execute the sample codes stored in NFTOOL for Fortran77 and Fortran90.

3.3.1 Linux

① Configure

To create the Makefile necessary for creating the executable format of the sample code, change directiory to "$INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/Fortran77(Fortran)" and execute the following command.

```
$ ./configure [Option]
```

The options to specify are as follows.

--prefix :

Specify the path to the directory where the executable samples (sample[N]) are placed. If you follow the instructions in section 4.1, specify the following.

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

Also specify the following if necessary.

--with-nc-lib :

Specify the directory path where the NetCDF-C library "libnetcdf.a" is located.If it is located in "/*/*/lib" or "/*/*/*/lib", no need to specify.

--with-nc-include :

Specify the directory path where the NetCDF-C header file "netcdf.h" is located.If it is located in "/*/*/include" or "/*/*/*/include", no need to specify.

--with-nf-lib :

Specify the directory path where the NetCDF-Fortran library "libnetcdff.a" is located.If it is located in "/*/*/lib" or "/*/*/*/lib", no need to specify.

--with-nf-include :

Specify the directory path where the NetCDF-Fortran header file " netcdf.inc " is located.If it is located in "/*/*/include" or "/*/*/*/include", no need to specify.

When the configuration process is successful, a Makefile is created in the current directory.

② Setting Environment Variables

Set the environment variables to be used when executing the sample code. Open "$INSTALL_DIR/AMSR3_Fortran/_setEnv.bash" and modify “LD_LIBRARY_PATH“.

```
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:/usr/local/lib:/usr/local/lib64:[NetCDF-C,NetCDF-Fortran Library storage directory]"
```

Execute the following command in “$INSTALL_DIR/AMSR3_Fortran/” to activate the environment variables.

```
$ source _setEnv.bash
```

If you want to enable the environment variables at startup, please append the contents of "_setEnv.bash" to your "~/.bashrc" file.

③ Edit the Sample Code

Edit the sample code to load a specified product. First, store the product file in any directory. As mentioned in section 4.1, NFTOOL provides "$INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/data" as the directory for storing product files. Open the sample code with a text editor and enter the path of the product file to be read into the string variable "fname". Specify the length of the character array as the number of characters of the file path. (in the example below, it is 59 characters).

<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'&
     &     /
~~~
```

Save the source code and the editing is complete.

④ Compiling the Sample Code

To create the executable file for the sample code, execute the following command in "$INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/Fortran77(Fortran)".

```
$ make
```

If the executable file "sample[N]" is created in the path specified by "--prefix", then the creation of the executable file for the sample code is complete. You can confirm it using the following command.

```
$ ls -l $INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/Fortran77(Fortran) (the path specified by "--prefix")
```

If the make process fails, please refer to section 3.3.4 for troubleshooting steps.

3.3.2 Windows

① Setting Environment Variables

Set the environment variables to be used during the compilation and execution of the sample code.

Open "$INSTALL_DIR/AMSR3_Fortran/_setEnv.bat" and modify the following environment variables and save the file.

```
set NC_LIB_PATH=[NetCDF-C Library Path]
set NF_LIB_PATH=[NetCDF-Fortran Library Path]
set NCINC=%NC_LIB_PATH%¥include (Directory Path of “netcdf.h”)
set NFINC=%NF_LIB_PATH%¥include (Directory Path of “netcdf.inc”)
set NCLIB=%NC_LIB_PATH%¥bin (Directory Path of “netcdf.dll”)
set NFLIB=%NF_LIB_PATH%¥bin (Directory Path of “libnetcdff.dll”)
```

Execute _setEnv.bat to activate the environment variables and add the directory paths %NCINC%, %NCLIB%, %NFINC%, and %NFLIB% to the “PATH”.

② Editing the Sample Code

Editing the sample code to load a specified product. First, store the product file in any directory. As mentioned in section 4.1, NFTOOL provides "$INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/data" as the directory for storing product files. Open the sample code in a text editor and enter the path of the product file to be read into the string variable "fname". Specify the length of the character array as the number of characters in the file path. (in the example below, it is 59 characters).

<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'&
      &      /
～～～
```

Save the source code and the editing is complete.

③ Compiling the Sample Code

To create the executable file for the sample code, execute the following command in "$INSTALL_DIR/AMSR3_Fortran/NFTOOL/sample/Fortran77(Fortran)".

```
$ make
```

If you can confirm that "sample[N].exe" has been created with the following command, the creation of the executable for the sample code is complete.

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

If the make process fails, please refer to section 3.3.4.

3.3.3 Mac

The procedure for creating the sample program is the same as described in section 3.3.1.

3.3.4 Troubleshooting Compilation Error

If you encounter errors during the make process of the sample code due to different versions of gcc, please refer to the following methods.

- Error: BOZ literal constant at (1) cannot appear …
  Add the following compilation options.
  -fallow-invalid-boz

- Error: Type mismatch between actual argument at…and actual argument at…
  Add the following compilation options.
  -fallow-argument-mismatch

4 Reference

4.1 Directory Structure

```
AMSR3_Fortran
├ _setenv.bash(_setenv.bat) : Environment Variable File
└ NFTOOL/
    ├ Makefile : Makefile for Creating NFTOOL Library *1
    ├ Makefile.in : Makefile Template
    ├ autom4te.cache/ : Directory for Storing Automake-related Files
    ├ autoscan.log : automake-related Files
    ├ config.guess : automake-related Files
    ├ config.log : configure-related Files
    ├ config.status : configure-related Files
    ├ config.sub : automake-related Files
    ├ configure : configure-related Files
    ├ configure.ac : automake-related Files
    ├ configure.scan : automake-related Files
    ├ install-sh : automake-related Files
    ├ include/
    │    ├ AM3TK_f77.h : Header File for Fortran 77 Sample Code
    │    └ AM3TK_f90.h : Header File for Fortran 90 Sample Code
    ├ lib/
    │    └ libAMSR3.a : NFTOOL Library
    ├ sample/
    │    ├ Fortran77/ : Directory for Fortran 77 Sample Codes
    │    │    ├ Makefile : Makefile for creating an executable of sample code  *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/ : Directory for Storing Fortran 90 Sample Code
│   │   ├ Makefile : Makefile for creating an executable of sample code   *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/ ： Directory for Storing Product Files for Sample Code
└ src/ : the source code for creating the NFTOOL library, The directory for storing the header file
```

Figure 4-1 Directory Structure

*1: When you run the configure command, Makefile will be created or overwritten.

4.2 Function List

This section provides details of the functions in NFTOOL. If a function number starts with "F", it can be used as a Fortran function. If it starts with "S", it can be used as a Fortran subroutine.

F01			
status = nftool_open_read_func(fname, nc_id)			
Open a NetCDF file in read-only mode and get the NetCDF ID.			
Item	Type	Variable name	Description
Input	character*N	fname	Product File Path
Output	integer	nc_id	NetCDF ID
Return Value	integer	status	Normal：0 Error：Error code (refer to section 4.3)

S01			
call nftool_open_read_sub (fname, nc_id, status)			
Open a NetCDF file in read-only mode and get the NetCDF ID.			
Item	Type	Variable name	Description
Input	character*N	fname	Product File Path
Output	integer	nc_id	NetCDF ID
	integer	status	Normal：0 Error：Error code (refer to section 4.3)
Return Value	-	-	-

F02			
status = nftool_close_func(nc_id)			
Close the specified NetCDF ID file.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
Output	-	-	-
Return Value	integer	status	Normal：0 Error：Error code (refer to section 4.3)

S02			
call nftool_close_sub(nc_id, status)			
Close the specified NetCDF ID file.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
Output	integer	status	Normal：0 Error：Error code (refer to section 4.3)
Return Value	-	-	-

F03			
status = nftool_getAttr_func(nc_id, varid, attr_name, attr_val)			
Get the value of the specified attribute name. If you specify "NF_GLOBAL" for varid, it targets the global attributes.			
Item	Type	Variable name	description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
	character*N	attr_name	Attribute Name
Output	*1	attr_val	Attribute Value
Return Value	integer	status	Normal：0 Error：Error code (refer to section 4.3)

*1: Specify the data type that corresponds to the attribute value you want to retrieve.

S03			
call nftool_getAttr_sub(nc_id, varid, attr_name, attr_val, status)			
Get the value of the specified attribute name. if you specify "NF_GLOBAL" for varid, it targets the global attributes.			
Item	Type	Variable name	description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
	character*N	attr_name	Attribute Name
Output	*1	attr_val	Attribute Value
	integer	status	Normal：0 Error：Error code (refer to section 4.3)
Return Value	-	-	-

*1: Specify the data type that corresponds to the attribute value you want to retrieve.

F04			
status = nftool_getAttrType_func(nc_id, varid, attr_name, attr_type)			
Get the data type of the specified attribute name, if you specify "NF_GLOBAL" for varid. it targets the global attributes.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
	character*N	attr_name	Attribute Name
Output	integer	attr_type	Integer value corresponding to the data type of an attribute (refer to section 4.4).
Return Value	integer	status	Normal：0 Error：Error code (refer to section 4.3)

S04			
call nftool_getAttrType_sub(nc_id, varid, attr_name, attr_type, status)			
Get the data type of the specified attribute name, if you specify "NF_GLOBAL" for varid. it targets the global attributes.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
	character*N	attr_name	Attribute Name
Output	integer	attr_type	Integer value corresponding to the data type of an attribute (refer to section 4.4).
	integer	status	Normal：0 Error：Error code (refer to section 4.3)
Return Value	-	-	-

F05			
status = nftool_getAttrLen_func(nc_id, varid, attr_name, attr_len)			
Get the length of the array of attributes for the specified attribute name. If "NF_GLOBAL" is specified for varid, it targets global attributes.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
	character*N	attr_name	Attribute Name
Output	integer	attr_len	Length of an attribute array.
Return Value	integer	status	Normal：0 Error：Error code (refer to section 4.3)

S05			
call nftool_getAttrLen_sub(nc_id, varid, attr_name, attr_len, status)			
Get the length of the array of attributes for the specified attribute name. If "NF_GLOBAL" is specified for varid, it targets global attributes.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
	character*N	attr_name	Attribute Name
Output	integer	attr_len	Length of an attribute array.
	integer	status	Normal：0 Error：Error code (refer to section 4.3)
Return Value	-	-	-

F06			
status = nftool_getNumAttr_func(nc_id, varid, num_attr)			
Get the total number of attributes for the specified variable ID. If "NF_GLOBAL" is specified for varid, it targets global attributes.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
Output	integer	num_attr	Total number of attributes
Return Value	integer	status	Normal：0 Error：Error code (refer to section 4.3)

S06			
call nftool_getNumAttr_sub(nc_id, varid, num_attr, status)			
Get the total number of attributes for the specified variable ID. If "NF_GLOBAL" is specified for varid, it targets global attributes.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
Output	integer	num_attr	Total number of attributes
	integer	status	Normal：0 Error：Error code (refer to section 4.3)
Return Value	-	-	-

F07			
status = nftool_getAttrName_func(nc_id, varid, attid, attr_name)			
Get the attribute name of the specified attribute number. If "NF_GLOBAL" is specified for varid, it targets global attributes.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
	integer	attid	Attribute Number (≧1)
Output	character*N	attr_name	Attribute Name
Return Value	integer	status	Normal：0 Error：Error code (refer to section 4.3)

S07			
call nftool_getAttrName_sub(nc_id, varid, attid, attr_name, status)			
Get the attribute name of the specified attribute number. If "NF_GLOBAL" is specified for varid, it targets global attributes.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
	integer	attid	Attribute Number (≧1)
Output	character*N	attr_name	Attribute Name
	integer	status	Normal：0 Error：Error code (refer to section 4.3)
Return Value	-	-	-

F08			
status = nftool_getAttrName_all_func(nc_id, varid, attr_name_list)			
Get an array of all attribute names included in the specified variable ID. If "NF_GLOBAL" is specified for varid, it targets global attributes.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
Output	character*N(M)	attr_name_list	List of attribute names
Return Value	integer	status	Normal：0 Error：Error code (refer to section 4.3)

S08			
call nftool_getAttrName_all_sub(nc_id, varid, attr_name_list, status)			
Get an array of all attribute names included in the specified variable ID. If "NF_GLOBAL" is specified for varid, it targets global attributes.			
Item	Type	Variable name	Description
Input	integer	nc_id	NetCDF ID
	integer	varid	Variable ID (≧0)
Output	character*N(M)	attr_name_list	List of attribute names
	integer	status	Normal：0 Error：Error code (refer to section 4.3)
Return Value	-	-	-

4.3 Return Value

Some of the error codes used in the NFTOOL library are explained below.

Number	Description	Note
-300	Insufficient memory size for storage destination.	This occurs when the number of variables in the array is insufficient to obtain the attribute of the array type.
-301	Undefined storage destination	This occurs when the memory allocated for variables storing file names, attribute names, or other data is insufficient.
-400	Failure of memory allocation	This occurs when MALLOC fails.

Other error codes not listed above are defined by the NetCDF library (for C language). Refer to the following for more details. NetCDF: netcdf.h File Reference (ucar.edu)

In the NFTOOL library, errors thrown by the NetCDF library (for C language) are used directly as return values.

4.4 Library Variable

Below are the variables and structures defined in the header files "AM3TK_f77.h" and "AM3TK_f90.h". Please refer to the sample codes for usage examples.

<data type >

Variable Name	Type	Value	Description	Note
AM3_DTYPE_CHAR	integer	2	String type	
AM3_DTYPE_SHORT	integer	3	Signed 2-byte integer	
AM3_DTYPE_INT	integer	4	Signed 4-byte integer	
AM3_DTYPE_FLOAT	integer	5	32-bit floating-point number.	
AM3_DTYPE_DOUBLE	integer	6	64-bit floating-point number	
AM3_DTYPE_UBYTE	integer	7	Unsigned 1-byte integer	
AM3_DTYPE_USHORT	integer	8	Unsigned 2-byte integer	

<Dataset attribute (variable)>

Variable Name	Type	Value	Description	Note
DEF_NUM_ATT	integer	16	The number of different types of dataset attribute names	

<Dataset attribute (structure)> Only ”AM3TK_f90.h”

Structure definition	Member	Type	Description	Note
global_attr (Global Attribute)	attname	character*50	Name	ex) processing_level
	attlen	integer	Array length	ex) 7
	atttype	integer	Data type	ex) 2
	val_str	character*3080	String type value	ex) Level1A
	val_ubyte	integer*2	Ubyte type value	
	val_short	integer*2	Short type value	
	val_ushort	integer*4	Ushort type value	
	val_f	real*4	Float type value	
	val_d	real*8	Double type value	
	val_int	integer*4	Int type value	
dataset_attr (Dataset Attribute)	attname	character*20	Name	ex) flag_masks
	attlen	integer	Array length	ex) 2
	atttype	integer	Data type	ex) 4
	val_str	character*1024	String type value	
	val_ubyte	Integer*2	Ubyte type value	
	val_short	Integer*2	Short type value	
	val_ushort	integer*4	Ushort type value	
	val_f	real*4	Float type value	
	val_d	real*8	Double type value	
	val_int(10)	integer*4	Int type value	ex) 4, 128
dataset (Dataset)	ds_name	character*50	Dataset name	ex) ObsCount_Ch06H_Quality
	ds_attr	dataset_attr	Dataset attribute	

4.5 Environment Variable

The following environment variables are referenced during program execution. They are used to append the search path for the required libraries.

OS	Variable name	Contents	Note
Linux, Mac	LD_LIBRARY_PATH	NetCDF library (Fortran and C) path	
Windows	PATH	NetCDF library (Fortran and C) path	