MMM7.1 Transport Model

Prepared by

Lixiang Luo (lixiang.luo@lehigh.edu)
Tariq Rafiq (rafiq@lehigh.edu)
Arnold Kritz (kritz@lehigh.edu)

REVISION HISTORY

Date	Description
16-OCT-2012	Release - Version 7.1.1 (Minor bug fixed in theDRIBM Model)
11-May-2011	Documentation updated
11-Feb-2010	Release - Version 7.1.0

INTRODUCTION

This software package contains the multi-mode anomalous transport module version 7.1 (MMM7.1). MMM7.1 is a theory-based transport model which has been used to predict temperature, density and toroidal rotation profiles for tokamak plasmas. The model includes an improved Weiland model for the ITG, TEM, and MHD modes, the Horton model for short wavelength ETG model and a new model for the drift resistive inertial ballooning modes (DRIBM). The ETG transport threshold in the Horton model is refined by using threshold obtained from toroidal gyrokinetic ETG turbulence. These components of transport models provide contributions to transport in the different regions of plasma discharge. The complete documentation (doc/mmm_ref.pdf) of MMM7.1 package can be found in the "doc" subdirectory.

NOTE: MMM7.1 module has been tested and used only with double precision.
NOTE: In a number cases examined, the bug fix in 7.1.1 resulted in only negligible changes compared to results obtained MMM7.1.0.

PREREQUISITES

MMM7.1 is intended to be built on an x86 POSIX-complaint OS (UNIX, Linux, Cygwin, etc.) with a Fortran 90 compiler. The rest of README will assume that you are running an x86 Linux system with a BASH shell. "make" and "ar" must be installed in the OS. Building on Cygwin is supported, although the compiler options are much more limited. Building on other architectures may be possible but not yet tested. Users and developers are welcome to send in their experience on building MMM7.1 on different architectures.

CONTENTS

First, unpack the source using this command on any POSIX-compliant operating system:

```
$ tar xvzf mmm-7.1.tar.gz
```

Future MMM releases will have version numbers in the X.x form, where X is the major release number and lowercase x is the minor release number. The release numbers are used as a suffix, in the form of X_x, for naming some source files and directories.

Upon unpacking, you will be able to see the following files, organized in a cascade structure:

`mmm7_1/`

```
        | Makefile
        | README and README.html
        |
        +- doc/
        |   | mmm_ref.pdf
        |
        +- libmmm7_1/
        |   | Makefile
        |   | modmmm7_1.f90
        |   | w20_README.txt
        |   | w20mod.f90
        |
        +- testmmm/
            | Makefile
            | testmmm.f90
            |
            +- case-1mode/
            |   | sample_input
            |   | sample_output
            |
            +- case-hmode/
            |   | sample_input
            |   | sample_output
            |
            +- case-2ndkind/
                | sample_input
                | sample_output
```

These files are as follows:

Main directory

`Makefile`	The master makefile, intended for building the whole MMM7.1 package. This makefile is designed to accept any compiler command. See "COMPILATION INSTRUCTIONS" section for more details.

README.html README	This file. Brief descriptions of the other files, and instructions for compiling the module containing the MMM7.1 transport model and a driver program.

doc sub-directory

mmm_ref.pdf	Documentation for MMM7.1 in PDF format.

libmmm7_1 sub-directory

Makefile	Secondary makefile for libmmm7_1, intended for building the module only.
modmmm7_1.f90	F90 source file containing the mmm7_1 subroutine and documentation. Plasma transport coefficients obtained using the Multi-Mode transport model, are computed. Contributions to transport from the Weiland, DRIBM and ETG models are computed in this routine.
w20mod.f90	Original source of the Weiland20 module, packaged as one single Fortran 90 module.
w20_README.txt	Original documentation of the Weiland20 module.

testmmm subdirectory

Makefile	Secondary makefile for the "testmmm" driver program.
testmmm.f90	F90 source file of the driver program. The program reads the input file "input" and writes the output file, "output". Contributions of each of the modes to the thermal and particle diffusivities are written to the file "output". Two kinds of input data can be accepted. The first kind contains values for all profiles, and the second kind contains shape function parameters for constructing simple polynomial profiles.
case-lmode/sample_input	Sample input file for a D3D-like L-mode case.
case-lmode/sample_output	Sample output file for a D3D-like L-mode case.
case-hmode/sample_input	Sample input file for a D3D-like H-mode case.
case-hmode/sample_output	Sample output file for a D3D-like H-mode case.
case-2ndkind/sample_input	Sample input file for an L-mode case generated using second kind of input
case-2ndkind/sample_output	Sample output file for an L-mode case generated using second kind of input

COMPILATION INSTRUCTIONS

Follow the two-step procedure to build the binaries:

1. Set the environmental variable "MMMFC" to the compiler command on your system. Here are some common examples:

If you are using ...	MMMFC should be set to ...
G95	g95
GNU Fortran	gfortran
Intel Fortran	ifort
PathScale Fortran	pathf95

Use "export" on Bourne-style shells and "setenv" on C-style shells to set environment variables. For example, on BASH:

```
$ export MMMFC=pathf95
```

or on TCSH:

```
% setenv MMMFC pathf95
```

Note that if you do not set any value for MMMFC, it will default to gfortran.

2. To compile the whole package, including the Fortran module modmmm7_1 and the stand-alone driver program "testmmm" simply run:

```
$ make
```

inside the main directory to invoke the master makefile, which will then invoke secondary makefiles to build the module and the driver program. The binary files (*.o, *.a and testmmm) are placed alongside with their source codes. To delete all binary files, use:

```
$ make clean
```

You can generate debugging-ready binary files using these commands:

```
$ make clean



$ make debug
```

It is possible to build the module and the driver program separately, using only the secondary makefiles. In this case, follow these steps:

1. Set the environmental variable "MMMFC" to the compiler command on your system.
2. Change the current directory to the subdirectory where you want to build binaries and issue the "make" command.

Note that the driver program depends on the module. Any change to the module source codes requires a re-compilation of the module before the driver program should be build. This procedure is automatic when the master makefile is used. However, when using secondary makefiles this has to be done manually.

TESTING INSTRUCTIONS

The driver program "testmmm" requires only one input file called "input". Since the input file is a Fortran namelist file, the variables can be arranged in any order.

To produce the test cases:

1. Change the current directory to the subdirectory of one of the test cases, such as `case-lmode`
2. Copy the sample input file, "sample_input", to "input":

```
$ cp sample_input input
```

3. Run the driver program by

```
$ ../testmmm
```

Note that the driver program is located in the parent directory.

As it runs, "testmmm" will generate an output file "output". The output file contains a listing of all possible input arguments for subroutine mmm7_1 and it is followed by a listing of all possible output arguments of the subroutine. You can then do a simple comparison between the output you obtained and the given sample output, using the "diff" command:

```
$ diff output sample_output
```

which should give little to no output, if the driver program is correctly built. Further information on `testmmm` can be found in the documentation (doc/mmm_ref.pdf).

Output is given as a simple text spreadsheet, which can be interpreted easily by plotting tools such as gnuplot.

For example, the output of case-lmode case be visualized using gnuplot by these commands run in the `case-lmode` subdirectory:

```
$ gnuplot



gnuplot> plot '< tail -n 51 output' u 1:4 w l ti 'theig'
```

Note that `case-lmode` has 51 zones (radial points). This gnuplot command extracts the last 51 lines of the output file and generates an X-Y plot with the minor radius (column 1) on X axis and the electron thermal diffusivity `theig` (column 4) on the Y axis.

LINKING INSTRUCTIONS

To use MMM7.1 in your own program, the following issues need to be taken care of:

1. The "USE" statement in your Fortran program
2. A proper "CALL" statement of the mmm7_1 subroutine
3. Linking of libmmm7_1.a against other binary object files

Note that the current version of MMM7.1 uses optional arguments extensively, thus requiring the use of explicit interface. If any optional argument is omitted, argument association by keywords must be used. Even in the case where no optional argument is omitted, the use of argument keywords is still strongly recommended, considering the large amount of arguments involved. An example of this style of subroutine call is given in the source file of the driver program, "testmmm.f90". One clear advantage of argument keywords is that the compiler can always determine the correct argument association, regardless of the order and the choices of actual arguments.

All array dummy arguments are defined as assumed-shape arrays, in contrast to earlier versions of MMM, which used deferred-shape arrays. Several arguments are optional. Please refer to the module source (modmmm7_1.f90) for more details on the selection of optional arguments.

A successful build of the MMM module will generate a number of binary files, among them are the two most important, "libmmm7_1.a" and "modmmm7_1.mod" ("modmmm7_1.MOD" if PathScale compilers are used) in the "libmmm7_1" subdirectory. "libmmm7_1.a" is the static-link library and "modmmm7_1.mod" is the FORTRAN module file.

The compilation of any source file that use the modmmm module requires the compiler to correctly locate the module file ("modmmm7_1.mod"). Most compiler search "*.mod" files in directories listed after the -I option. Module files are generally incompatible among different compilers. You will not be able to compile the driver program with compiler B if the MMM7.1 module is compiled using compiler A.

Linking of "libmmm7_1.a" against other binary object files should only involve putting libmmm7_1.a in the object file list, as long as the file can be located by the compiler. Note that only static linking is supported in this version. Please follow the instructions of your Fortran compiler to set appropriate options.

The manual compilation commands

If the make utility does not work, or if the module is being compiled on an unsupported platform then the compilations may need to be done by hand. What follows are step-by-step instructions for creating the test executable "testmmm".

1. Compile and produce the static-link library from the module source files. Suppose the Fortran 90 compiler on your system is called "f90". An object code is produced when the compiler is invoked with a "-c" option. Starting from main directory:
   ```
   $ cd libmmm7_1
   ```

```
$ f90 -c w20mod.f90 -o w20mod.o


$ f90 -c modmmm7_1.f90 -o modmmm7_1.o


$ ar rcs libmmm7_1.a w20mod.o modmmm7_1.o


$ cd ..
```

Note that "w20mod.f90" must be compiled first because it is the source file of a Fortran 90 module used by other MMM source files.

2. Compile the driver program and link the object codes together, producing the executable "testmmm":

```
$ cd testmmm


$ f90 -c -I../libmmm7_1 testmmm.f90 -o testmmm.o


$ f90 testmmm.o ../libmmm7_1/libmmm7_1.a -o testmmm
```

Please note that how the module file ("modmmm7_1.mod") is located by the -I option and how the static-link library ("libmmm7_1.a") is linked with other object files. To generate debugging-ready binary files, replace all "-c" with "-g -c".

COMPILER-WISE NOTES

g95

Earlier versions of g95 may have problems with data types. Please use the latest stable version (>0.93).

PathScale (as for v3.2)

The driver program generated by this compiler cannot process namelist input files with Windows/DOS-style newline. A typical error message when running "testmmm" would look like this:

```
lib-4324 : UNRECOVERABLE library error




' is unrecognized in namelist input.


Encountered during a namelist READ from unit ...
```

A tool called "dos2unix" (included in Cygwin and most Linux distributions) can be used to convert these input files.

CONTACTS

If you have any problems, please contact

Lixiang Luo
Department of Physics, Lehigh University
lixiang.luo@lehigh.edu

Tariq Rafiq
Department of Physics, Lehigh University
rafiq@lehigh.edu

Arnold H. Kritz
Department of Physics, Lehigh University
kritz@lehigh.edu