# M4D.2015.2 - What's Here - Setup - Run

This folder m4d.2015.2 contains:

**a.m4dmac10.6** M4D compiled on a 2010 Mac Mini with a 2.4 GHz Intel Core 2 Duo running Mac OS X Version 10.6.4.

**a.m4dmac10.2** M4D compiled on an iMac with a 700 MHz PowerPC G4 running Mac OS X Version 10.2.8 with no optimization

**code** directory containing the C source code for M4D.

**documentation** documentation directory. Documentation is in both .pdf and (so you can add to it) .doc, Microsoft Word files.

> **m4d.commands** - an introduction to the command format, then an alphabetical reference list of all commands, followed by descriptions, sample input, and arrays needed, created and/or modified for each command.
> **m4d.commands.short** - a briefer description of the commands by purpose. Browse through this to see if there is a command to do what you want. Then look the command up in m4d.commands for further details.
> **m4d.variables** - reference list and descriptions of the named variables including their size and type, how they are created and what commands need them.
> **m4d.subroutines** - for the programmer. M4D is designed to be expandable. This gives a description of how to add commands, how to create and access named variables (at the coding level) and the input/output subroutines. There is also a complete list of the subroutines which are not also commands with some descriptions. (The subroutines which are commands are described in m4d.commands.)
> **m4d.fileformats** - descriptions (or where to find them) of various input file formats.

**examples** directory containing 6 examples. Three examples demonstrating adapted control volumes, then three examples using the MARVS Reynolds stress model.

| | Run on my Mac: | CPU | Max Memory |
|---|---|---|---|
| **box.cavity.example** | | 3 sec | 11 Mbytes |
| **kh2d.example** | | 31 min | 64 Mbytes |
| **chansq.DNS.Rtau300.example** | (total) | 6.3 hours | 116 Mbytes |
| **chan2d.fdf.example** | (KA example) | 11 sec | 5 Mbytes |
| **backstep.example** | | 4 min | 27 Mbytes |
| **turbine2d.example** | | 13 min | 62 Mbytes |

See m4d.README for a brief description of the examples including pictures. Each example given here has been tested and run. Look at the doc.rtf file in each example directory for the UNIX description to run the example. This will need modifying if you move or rename any files. If you move the directory jgm relative to the example directory, the command line in the input 'c: constant defaultdir ../../jgm' will also need to be changed to obtain plot results.

All examples are setup with the M4D command input files prefixed with 'in.', for the file to start the run, or 'inn.', essentially subroutine files. All examples are designed to put the output (files and plots) in directory 'out'. Sample results are included, with some files omitted to reduce the overall package size. The 'out' directories for the test runs have all had there names modified by the addition of '.jgm'.

**jgm** directory containing files needed for the plot package.

**m4d.README** an introduction to M4D, including it's scope and the examples.

**m4d.whatshere.setup.run** this file with "suggestions for getting started", and "error troubleshooting" below.

## Suggestions for Getting Started

1. Recompile M4D on your computer for fastest run times. If you are running on a Mac with OS X 10.6 or later, you may instead copy a.m4dmac10.6 to a.m4d. If using an earlier Mac with OS X you may instead copy a.m4dmac2.8 to a.m4d. (a.m4dmac2.8 also runs on my Mac with Mac OS X 10.6.4 but 6 times slower than a.m4dmac10.6.)
UNIX instructions:
Bring up a UNIX terminal window. (On a Mac, Applications/Utilities/Terminal.app brings up a UNIX terminal window. You will also need to have loaded the Developer's bundle so that the C compiler and other needed files are available.)
In the directory containing m4d.2015.2.tar.gz, unzip, then untar with

```
gunzip m4d.2015.2.tar.gz
tar -xf m4d.2015.2.tar
```

to give directory m4d.2015.2. (Simply double clicking on the file m4d.2015.2.tar.gz may or may not work correctly.)
cd to directory code.
modify the file 'makefile' if/as needed then in directory code type

```
make
```

to create the compiled version, a.m4d (in the main directory).

2. Look at the Command Format section of documentation/m4d.commands, then run M4D in the terminal window to make sure it runs. (Suggested input shown in green, code response in magenta.) In directory m4d.2015.2, type

```
./a.m4d
```

```
M4D - a research CFD code by Joan G. Moore
  for steady or time accurate calculations
  for inviscid, laminar or turbulent flow
              Features:
    Convection adapted control volumes
The transitional MARVS Reynolds stress model
            Version 2015.2

c: comment  *** testing

----- c: comment -----     time 0   change 0  maxsize 376832
 *** testing
c: constant dum d 5 1. 3.5 5. -1 -2

----- c: constant -----     time 0   change 0  maxsize 389120
dum, type: d, length 5, values 1 3.5 5 -1 -2
c: algebra dumm "" dum "" 2. 0. 1.

----- c: algebra -----     time 0   change 0  maxsize 405504
dumm =  *(2 * dum + 0)^1 +
c: print dumm 10

----- c: print -----     time 0   change 0  maxsize 413696

  dumm  size 5  type d values: 2 7 10 -2 -4
c: end

----- c: end -----     time 0   change 0  maxsize 421888
```

The time, change and maxsize printed at the start of each command, is the CPU time is secs (0.01 is the smallest it prints), the change in CPU since the last command start, and the maximum size used by the code in bytes. (This may not work on some computers. Hopefully it will then give zeros.)

3. Go into one of the 6 example directories and follow the instructions in doc.rtf to run it. Familiarize yourself with the output files and compare the results with the test results (in the directory out.jgm).

4. Create your own example. Start by copying the example closest to what you want to do. Then modify it.

If you wish to use a different geometry or grid, start with the inn.grid… file of the example and read about command gridfrommefp in documentation/m4d.commands. Note that the end result of this command is to create the arrays *idim4d, abcd, xyz, clt*. Depending on the software you have available, it may be easier to create this information externally then use command arrayread to set the arrays (see documentation/m4d.fileformats). If you make any changes at all to a grid which has collapsed c-grid corners, use command gridcorner so that the hidden points match correctly.

I strongly recommend:

plotting the grid and initial velocity estimate before trying a complete calculation;

running one full iteration or time-step with full print on to let the program check for input errors;

and checking that print for more subtle errors and warnings.

## Error Troubleshooting

M4D prints a message and stops when it detects an error. This can be a typo in the input, missing input, calling a command without the needed arrays available, trying to read from a file which isn't there, or writing to a directory which doesn't exist.

The hardest input errors to track down result (on my machine) in "segmentation faults". This is when the operating system terminates the program because it is trying to address space out of the program's allotted range. If this occurs when running one of the sample test cases, it may be because you have compiled with a high level of optimization, and the compiler has done it incorrectly. If you have changed the input, it is probably due to an array being smaller than the program expects, which can also result in other arrays being overwritten with inappropriate information. Since redirected print output is buffered, the print file may end well before the error has occurred. Rerunning the program in a terminal window, without redirecting the print (let it come to the window) will show you exactly where the program stopped. Inserting commands arraylist, and print or prints before where the program stopped, can assist in finding the error.