Notes on version 1.4 of the FM package

The changes in version 1.4 were made to enable a thread-safe special version of FM to be created. See file `FM_parallel.f95` for the thread-safe version.

The memory model for multi-precision variables has been changed from having one global database kept in module `FMVALS` that holds all the numbers to making the multi-precision variables local to the routines using them.

The way the user declares and uses type(fm), etc., variables is the same in this version as before.

Some improvements from the user's point of view are:

No longer needing to insert calls into the user's routines to `FM_ENTER_FUNCTION`, etc.

No need to call `FM_DEALLOCATE` before deallocating a multi-precision variable.

I have tested the new version using three compilers: gfortran, nagfor, and ifort. There are some issues concerning the first two of these compilers that could impact FM users.

gfortran

There is a bug in all the versions of gfortran I have tested (up to 10.2, running on a Mac). It seems to be similar to the bug I reported in 2017 in that it affects FM functions that return multi-precision arrays as function values.

For the first release of FM 1.4 in August, 2021, this bug causes the original version of program `SampleFM.f95` to fail in example 10 while computing eigenvalues of a matrix.

Starting with the September, 2021 version, I have modified FM to work around the bug, so the various Test and Sample programs of the package run correctly on all three compilers.

See the Troubleshooting section of the `FM_User_manual.txt` file for some discussion of how to work around this bug if your program has its own function subprograms that return multi-precision arrays as function values.

nagfor

Using nagfor with `-O3` optimization can cause some FM programs to crash with this error message:

```
  Runtime Error:  *** Arithmetic exception:  Floating invalid operation - aborting
```

The NAG developers said this problem with `-O3` seemed to be a bug in the clang (back-end C compiler) optimizer, and only showed up on Intel-based machines. They advised using optimization `-O2 -Wc,-O1` instead.

Later (2022) versions of nagfor and clang have also shown bugs that cause a "Floating invalid" message to be displayed after the program runs, when compiled with optimization level `-O1` or higher. In all cases I have seen, the program results are correct and the message can be ignored.