Adam Carleton
October 27th, 2020
AMS 129
Ian May

Homework 2 Report

1)

The reason we can use "real (dp)" throughout the dftmod.f90 and dft.f90 is because it was already declared as an integer, parameter with "dp" being held constant to precision kind(0.d0) in the utility module. Whenever dp is called, kind(0.d0) is referenced. In both dftmod and dft, the command "use utility" at the start of each will set the variables within the utility module to be referenced for the remainder of the program.

2)

The separation of duties between dft.f90 and dftmod.f90 is to simplify the code and to make sure that the subroutines/functions made have the proper arguments. By placing the subroutines/functions it is much easier to check they work. If another program is created other than dft.f90, and it needs the same subroutines/functions, the same module can be used without the need to rewrite them.

3)

The keyword "elemental" refers to the idea that the procedure can be called with scalar or array arguments. It isn't declared as external at the top of the code block because it is a dummy argument, not an actual argument.

4)

Error Table for Different Values of N with dp Precision					
N =	20	40	60	80	100
Err	1.4502810478608463E-002	9.9622964344625586E-007	5.6431170847304202E-010	2.6645352591003757E-015	3.1086244689504383E-015

It can be seen that as the number N increases. The error decreases and begins converging around ~10^{-15}.

5)

Error Table for Different Values of N with kind=4 Precision					
N =	20	40	60	80	100
Err	1.44997835E-02	3.69548798E-06	2.14576721E-06	1.19209290E-06	2.38418579E-06

Changing dp = kind(0.e0) changes the precision of error to kind=4 and it is arguably less accurate than kind(0.d0) double precision. The values at higher orders of N converge to ~10^{-6}. This is 10^{9} order of magnitude difference.

6)

-fPIC:
Generate position-independent code if possible

-fmax-errors=<number> :
Maximum number of errors to report

-Wimplicit-interface:
Warn about calls with implicit interface

-Wextra:
Print extra (possibly unwanted) warnings

-Wall:
Enable most warning messages

-g:
Produces debugging information that GDB can use

-fcheck=all:
Enables run-time tests such as an array bounds check

-fbacktrace:
Specifies that when a run-time error or a deadly signal is emitted the fortran run-time library will output a backtrace of the error.

-Wno-suprising:
When -Wall is turned on -Wsurprising is also included. Using the Wno-surprising flag removes the comment of the bogus error read out by -Wsurprising

7)

This project demonstrated the use of Fortran for compiling code to produce an executable. In this project, the executable was used to find the error in a boundary value problem of a Fourier Transform. By creating a module to contain the various subroutines/functions, the main file dft.f90 was able call the ones needed when specified. Within the module a function was declared which took the output of a matrix multiplication of rank two array “A” and a rank one array “x” and stored it in the output vector y. A physical domain was set up for “x” which was then fed through the forcing function to produce “f”. Next, “k” and “T” arrays were filled using the subroutine “dft_TransMat(x,k,T)” and “Tinv” was filled using “dft_InvTransMat(x,k,Tinv)”. To produce “ft”, the previously filled arrays: “T,f” were fed into the function “matvecprod(T,f)”. “ft” was then scaled accordingly and “u” was generated by using the same “matvecprod” function but by using the inputs “Tinv” and “ft”. The whole process covered how to use loops to fill arrays as well using if/then/else logical statements to fill them with desired values. A make file was useful in compiling the multiple files without having to do them individually with the myriad of flags appended. Changing the value “N” to larger numbers gave a smaller error as there were more points specified in the array.