CS 404: Problem Set 2

Directions for Submission

E-mail your answers to me at `ajp9@cornell.edu`. The subject of your message should be "CS404 PS2," and the body of the message should contain your answers. Some mailers can format messages using HTML or RTF. Please turn this feature off and send your message using plain text. If for some reason you cannot send your message as text, you may attach your answers as a text file.

Essential Knowledge–Please give a brief answer (1-2 sentences) for each

1. Ocean currents are driven by very small differences in the height of the sea surface. Thus, one way to compute the velocity of a current is to measure the sea-surface height at two locations and take the difference. One ocean circulation model does exactly this: at each time step, it computes the change in sea-surface height due to a variety of processes and then computes the velocity field by taking the height difference between adjacent points. Values for mean sea-surface height (cheifly, the depth of the seafloor) vary from 100m on the continental shelf >4,000m in the deep sea, while the height difference between adjacent points is usually less than 0.1cm (0.001m). Based on your knowledge of floating point arithmetic, would the currents predicted by this model be more accurate in deep or shallow water. Please explain.

2. Why is it easier to link a C program which calls FORTRAN routines using a FORTRAN compiler? If you wanted to do the linking with the C compiler, what would you need to do?

3. Computer memory is one dimensional. How would the 2D array:

1	2	3	4
10	20	30	40
100	200	300	400

   be stored in the memory of a C (or C++ or Java) program. How would Matlab or FORTRAN store the array?

Calling C from FORTRAN (and maybe even some MATLAB...)

Download the TAR file FpcaC.tar from the course website (the link is in the Problem Set section). Untarring this file will create the director FpcaC with a Makefile, data file (data.txt), several FORTRAN files (.f), and one C file (savemat.c). The FORTRAN code is identical to the code used for PS I, including a correct call to the BLAS routine SGEMV, except for a call to the C subroutine SaveToMat. Presently, the call looks like:

```
CALL SaveToMat(PC,TRIM(name2)//CHAR(0),TRIM(name3)//CHAR(0),m)
```

Eventually, this call will save the prinicpal component vector PC in a MATLAB .mat file called PC.mat (name2). The FORTRAN function TRIM(string) removes trailing spaces from string. The code "//CHAR(0)" appends the ASCII character 0 (the null character) to the string. This is needed when calling C since C strings are terminated by the null character. The second string (name3='PC') says that upon loading PC.mat, a Matlab array PC will appear in your workspace containing the data. Presently, it is not possible to call the MATLAB C Library in ACCEL–hopefully, this will be possible on Wednesday. Our strategy will be to get the subroutine call to work, then we will worry about MATLAB.

Your task can be divided into two pieces: 1. Get the call to SaveToMat to work, and 2. Get SaveToMat to call MATLAB correctly. To make the first step easier, I've placed two versions of SaveToMat in savemat.c, one that calls MATLAB, the other that just saves the data as a text file. Which version gets compiled is controlled using C-preprocessor directives (these are the commands preceeded by "#"). Specifically, if you want MATLAB calls to work, you must define a variable caled CALLMATLAB by placing the command "#define CALLMATLAB" at the top of the file. If you don't want MATLAB, you should un-define CALLMATLAB by commenting out the line So, the first thing you should do is to undefine CALLMATLAB.

Now that MATLAB is turned off, you need to get the call to SaveToMat to work. Here are some things you might need to fix:

a. When the C-compiler creates savemat.o, it will give the SaveToMat routine a name (probably, SaveToMat). Similarly, the FORTRAN compiler will replace "CALL SaveToMat" with a reference to some subroutine. The FORTRAN function will assume that SaveToMat will

be named like a FORTRAN routine. You need to make sure that the C routine is named in the way FORTRAN expects. To find the correct name, compile the objects and try to link them. If your names are wrong, the linker will give you an error message telling you the name the calling program is using. This message should help you figure out how to rename SaveToMat (be sure to rename the prototype at the top of savemat.c).

b. FORTRAN uses call-by-reference exclusively. Thus, to call a C routine from FORTRAN, we need to change any variables that are called by value (that are NOT pointers or arrays) to call by reference. This is actually an easy thing to do. If a variable X is called by value, you can replace it with *X in the subroutine declaration, prototype, and all places in the code.

To make the build easier, I've provided a Makefile. You should change the F77, CC, and LIBPATH macros to the correct versions for your system (they are set for ACCEL).

4. Get the call to SaveToMat to work without calling MATLAB. Describe in detail the changes you made to main.f and savemat.c. If you are working on a system other than ACCEL, please describe the system and any differences between it and ACCEL. For example, "I built Fpca on my Mac using an Absoft FORTRAN compiler and GNU C compiler. To get the call to work, I had to compile the FORTRAN code with the -f and -N15 flags."

5. If ACCEL gets its act together, I will provide directions on how to get the MATLAB calls to work.