CS 109 – C/C ++ Programming for Engineers w. MatLab– Fall 2009
Homework Assignment 4 – FIRST DRAFT
Finding the Maximum of a 2-D Function

Due: **Sunday November 1st at 11:59 P.M.** **The hard copy to be handed in at the beginning of class on Monday.**

Overall Assignment

For this assignment, you are to evaluate the function XYsin(X)sin(Y) in the range of X from 0 to 20.0 and Y from 0 to 10.0, using single and two-dimensional arrays. You are to use a series of functions to build the 2-D function, and then to find maximum values of two vectors and a matrix.

Necessary Functions:

The following functions will need to be written for this assignment:

- double sine(double x, double tolerance, int & limit);

  Uses a Taylor Series expansion about X = 0 to estimate the value of sine(x). (See below for implementation details.) On input, tolerance will be the convergence criteria to terminate calculations, and limit will be an upper limit on the number of terms to include in the series. On output, limit will contain the actual number of terms calculated, and the return value will be the estimated value of sine(x).

- int fillVector(double first, double last, double increment, int size, double data[ ]);

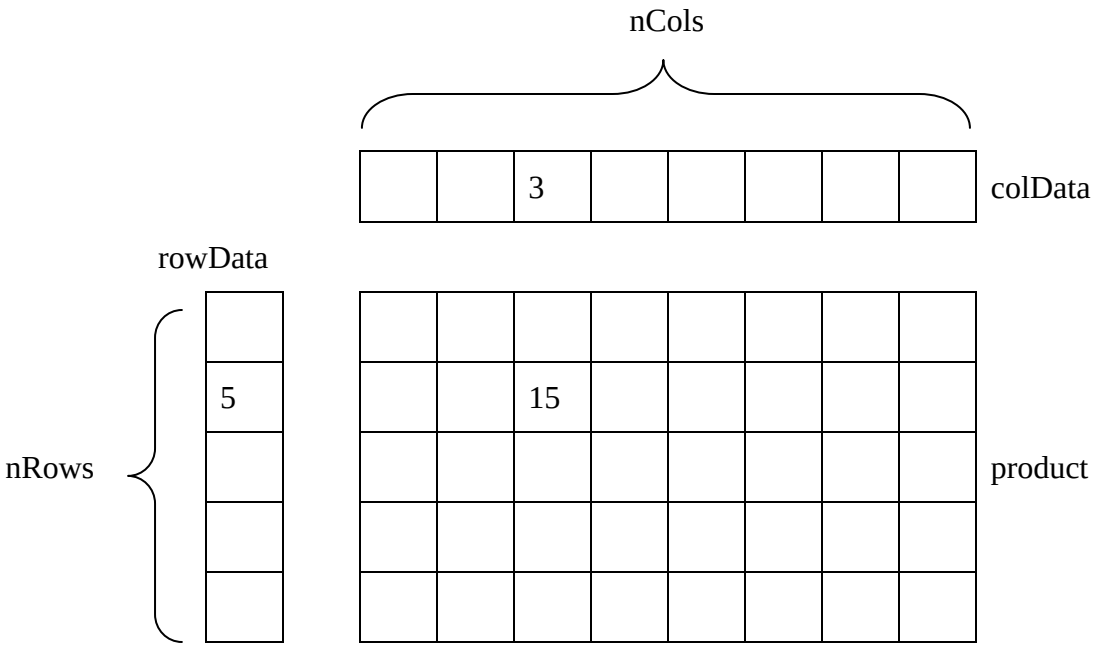
  This function will fill the vector "data" with doubles ranging from first to last in steps of increment, up to a maximum number of values equal to the size of the array. The return value is the number of values actually stored.

- int xSineX(double x[ ], double result[ ], int size, double tolerance, int limit);

  Calculates X * sin(X) for each value in x[ ], storing the results in result[ ], where size is the length of both x[ ] and result[ ]. This function should call upon the sine function listed above to do the actual calculations, passing along tolerance and limit.

- int multiply(double rowData[ ], int nRows, double colData[ ], int nCols, double product[ ][MAXCOLS]);

  Calculates the matrix multiplication of an nRows x 1 column vector (rowData) times a 1 x nCols row vector (columnData) to yield an nRows x nCols matrix, where product[r][c] = rowData[r] * colData[c], as shown in the diagram below. (Note that the rowData vector contains data for each row but is actually a column vector, and vice versa.)



- double maxVector(double data[ ], int size, int & errorCode);

  Finds and returns the largest value in the vector, and sets errorCode to zero, if no errors occur. Otherwise sets errorCode to -1 and returns 0.0.

- double maxMatrix(double data[ ] [MAXCOLS], int nRows, int nCols, int & errorCode);

  Finds and returns the largest value in the matrix, and sets errorCode to zero, if no errors occur. Otherwise sets errorCode to -1 and returns 0.0.

- double outputMatrix(double data[ ] [MAXCOLS], int nRows, int nCols, ostream & out);

  Writes the data in the matrix out to the output stream out, which may be passed as either cout or as any open output file stream.

- int main(void);

  main declares and initializes all of the needed arrays, and calls the functions listed above, and reports the results.

  Initially main( ) is a test driver, used to test each function one by one as you develop them.

  For the final version, main( ) should ask the user for minimum, maximum, and increment values for X and Y, and the name of an output filename for storing the results. main( ) should then call upon the functions listed above to calculate Xsin(X), YSin(Y), and XYsin(X)sin(Y), storing the matrix in a file for later use and reporting the maximum values in each of the vectors and the matrix.

Taylors Series Expansion

- Taylors Series Expansion can be used to approximate the value of sin(x) near zero according to the infinite series:

$$\sin(x) \approx x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} + \quad ...$$

- The more terms that are included, the better the approximation.

- The obvious way of calculating each term of the series is inefficient, and leads to severe numerical overflow problems. A better approach is to calculate each term of the series from the previous term, as:

$$term_0 = x$$

$$term_i = \frac{-\ term_{i-1}\ *\ x^2}{2i * (\ 2i + 1\ )} \quad \forall\ i > 0$$

- The algorithm of the function is to start adding up terms (in a for loop), and continue until either the limit on the number of terms is reached, or until the (absolute value of the) term added is less than the convergence tolerance.

-

Program Implementation

- For testing purposes vary X from 0.0 to 10.0 and Y from 0.0 to 20.0, in steps of 0.1 each.

- MAXROWS and MAXCOLS should be global const ints with values of AT LEAST 101 and 201 respectively. (Each step in the arrays will correspond to a delta X or delta Y of 0.1 in each direction. The delta values should not be set as constants, but should be calculated as needed.)

- Individual functions should not be dependent on particular values of limits and sizes, etc, but should work for any given values.

Incremental Development

For any large programming task, it is best to develop and test the code in small incremental steps, rather than trying to do it all at once. For this assignment, it is recommended that you develop your program in the order of the functions listed above, and test each function as it is developed.

Program Input

The user should input the range and increment to use for X and Y, the limit and tolerance desired, and the name of the output file.

Program Output

Your program should first print out your name and ID, and explain to the user what the program does. Then it should report problem results to the screen in a clear human-readable form, as appropriate for the current problem.

What to Hand In:

1. Your code, **including a readme file and a project file if requested by the TA,** should be handed in electronically using Blackboard.
2. The purpose of the readme file is to make it as easy as possible for the grader to understand your program. If the readme file is too terse, then (s)he can't understand your code; If it is overly verbose, then it is extra work to read the readme file. It is up to you to provide the most effective level of documentation.
3. If there are problems that you know your program cannot handle, it is best to document them in the readme file, rather than have the TA wonder what is wrong with your program. In particular, if your program does not complete all of the steps outlined above, then you should document just exactly which portions of the project your program does accomplish.
4. A printed copy of your program, along with your readme file and any supporting documents you wish to provide, (such as hand-drawn sketches or diagrams) should be handed in **in class**.
5. Make sure that your **name and your CS account name** appear at the beginning of each of your files. Your program should also print this information when it runs.

Optional Enhancements:

It is course policy that students may go above and beyond what is called for in the base assignment if they wish. These optional enhancements will not raise any student's score above 100 for any given assignment, but they may make up for points lost due to other reasons.

- Write a function to find and report all of the ***local maxima*** in the matrix:

  Finds and prints out cells in the array that correspond to local maxima, defined as cells that are **no lower than** all neighboring cells. For most cells, there will be four neighbors, but for cells on the edges or in the corners, there will be 3 or 2 neighbors respectively. The function should also print out the overall maximum, which is the highest of all the local maxima found. The return value is the number of local maxima discovered, or a negative number in the event of an error. xMin, xMax, yMin, and yMax are necessary to report the x and y values at which the maximum are found.

  You may find it helpful to write a helper function to calculate a value of X or Y corresponding to a given index, given the minimum and maximum values and the total number of values. I.e. calculate X from xMin, xMax, c, and nCols or calculate Y from yMin, yMax, r, and nRows. This method is not required but may be helpful.

- Other enhancements that you think of – Check with TA for acceptability.