

## Assignment 6

(Due Monday 10 October)

### Assignment 6:

This assignment demonstrates resource management in LabView. The goal is to: i) read a data file on the Internet, ii) convert the data file to a 1D array of floating point numbers, iii) write this 1D array to a file on the local disk, iv) open the file and use its data in a summing For Loop, and v) write the new data to disk. This is implemented in two independent VIs.

WARNING: Data Socket operations are known to sometimes cause problems when LabView is run on the OSX operating system. Apple users may have to test their VIs on a Windows PC.

The file of interest is located at this URL:

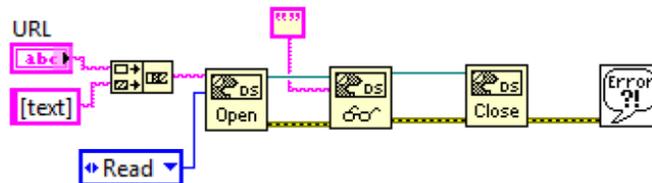
<http://www.unm.edu/~mph/306/assignment6>

The file can be accessed in LabView with a Data Socket, which should be open, read, and then closed.

Create a string control on the front panel and copy the entire URL above into it. The characters [text] must be appended to the URL string. This is performed with the Concatenate Strings function, which splices two strings together to make a single string of characters. The resulting string will be:

[http://www.unm.edu/~mph/306/assignment6\[text\]](http://www.unm.edu/~mph/306/assignment6[text])

Wire this string to the URL input of Data Socket Open and configure the mode terminal for Read. Pass the connection id to Data Socket Read. This function must be instructed as to what type of data it is dealing with; wiring any string to the type (Variant) terminal defines the data type. In the diagram shown below, a blank string is used, but any string will work as the content is ignored. Close the resource when the read completes. With a working Internet connect, test the VI.

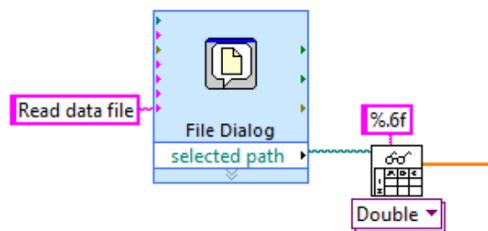


The data string is next converted to a 1D array using the Spreadsheet String to Array function. In addition to the spreadsheet string from the Data Socket Read, this function needs formatting and configuration for a 1D array (2D is default). LabView has a powerful numeric formatting capabilities. Use the format syntax `%.6f` on the format string terminal. This specifies a floating point number with 6 digits of precision after the decimal point. Next, place an Array Constant on the Block Diagram. Create a DBL Numeric Constant and drop it into the Array Constant to

define a 1D array. Connect it to the array type terminal. This forces the function to produce a 1D array of DBL floating point numbers.

A spreadsheet file can be created and written in a single, highlevel operation using Write to Spreadsheet File.vi found on the File I/O palette. Wire the 1D array data to the appropriate input and format as %.6f as before. When the VI is run, the user will be prompted to save a file to disk. Also save the working VI.

Open a blank VI and place Read from Spreadsheet File.vi on the Block Diagram. This highlevel VI will be used to access the 1D array file written above. It will deliver the data on the output terminal labeled first row. Set the format syntax to %.6f and wire the output into a For Loop. To setup a user prompt for opening the file, the Express VI File Dialog can be used:

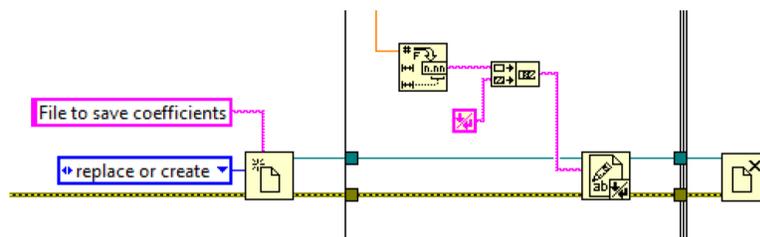


Following the procedure from Assignment 4, divide each term in the data file by  $i!$  where  $i$  is the iteration count of the For Loop. Sum the result of each iteration using a shiftregister. Mathematically, the following finite series is being constructed:

$$\sum_{i=0} x_i / i!$$

If performed correctly, the series should sum to a value very close to 0.05.

Each term in the series should be written to a separate text file. This can be done when the For Loop exits, but it is often desirable to record data as it is being generated. From the File I/O palette, select the Open/Create/Replace File VI and place it outside the For Loop. Configure it to replace or create and add a custom user prompt. The refnum is wired to a Write to Text VI inside the For Loop. Each series term is a floating point number that must be converted to a text string using Number to Fractional String VI. The Concatenate Strings functions adds a Line Feed character to the end of each data point so the data is displayed in a single column; otherwise, a single continuous line of text would be written. When the For Loop exits, the file is closed as shown:



This VI is dealing with two files: one is being read and a second is written. To force the read operation followed by write, connect the error out cluster from the File Dialog Express VI to the error in terminal of the Open/Replace/Create File VI. Terminate the error cluster in both VIs with a Simple Error Handler.

Demonstrate and explain both VIs for the instructor.