

Matrix / Array Manipulation with LabVIEW

ECE 480

Thursday November 12, 2009

Mark VanCamp

Executive Summary

It is often useful to display data in graphs and charts. This application note will show how complex array operations can be done easily using the visual interfaces of LabVIEW. The application note will show how one can take a data file (Excel file) read it into LabVIEW, manipulate its contents and turn them into graphs.

Key words

Matrix, Array, LabVIEW, Graphs, Indicators

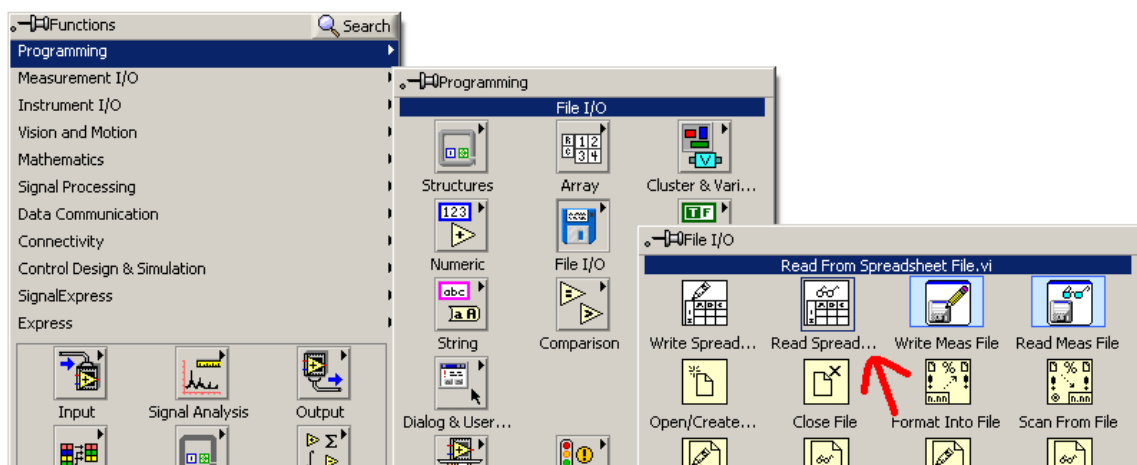
Overview and Objective

There are many different ways to manipulate arrays and create graphs from them; however this generally requires one to have a good understanding of programming and knowledge of tools such as MATLAB. For many users this programming knowledge is lacking and there is a need for a simpler method, this is where LabVIEW comes in. LabVIEW is a graphical programming environment that utilizes icons and wires that resemble flowcharts. This program is ideal for the programming illiterate because it does not require that actual code be written, one simply needs a basic understanding of flow diagrams. LabVIEW's simplicity is accomplished through the use of toolboxes that allow a user to simply select desired operations, drag them from the toolbox into a workspace, and connect them together thus creating their desired outcome. The objective of this application is to show how LabVIEW can be used to manipulate matrices and arrays and then create graphs.

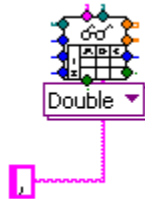
Procedure

Before LabVIEW can work with data it must first be saved in a proper format. A common format for data to be read into LabVIEW is .csv, which stands for comma separated values. This is a format that comes standard in Microsoft Excel so one can simply create a data matrix there and save their work as a .csv file. Once this initial step is complete LabVIEW is ready to be opened. To better understand the process of using LabVIEW this application note will go through a full example and explain the steps along the way. The sample .csv file that will be used throughout the application note will be a one column matrix with 805 rows.

Step 1: Open LabVIEW and select "New VI". This will open two windows, the first window is the Front Panel and the second window is the Block Diagram. The Block Diagram is more "behind the scenes" this is where most of the work is done; it is the portion that represents the flowchart that was mentioned earlier. The Front Panel is the user interface that displays graphs and gives control over the VI.



Step 3: The Read from Spreadsheet File.vi is used to bring the Excel file into LabVIEW. As with all of the functions hovering the mouse over them highlights their ports and describes their functions. The Read from Spreadsheet File VI has many connections but only a few are necessary for this example.



The top left teal port sets the file path, the bottom pink port sets the delimiter, and the top right orange port calls all rows in the file specified and makes them array type. For now nothing needs to be done to the file path port, when the VI is run it will simply ask that the path be specified then. Next place the mouse over the bottom pink port and right click. Select Create > Constant. This will make the pink box displayed above. By double clicking on the box data can be placed into it. Since this example is working with comma separated values (.csv) place a comma in this box to let LabVIEW know this is the delimiter that is used to separate values. Next right click on the top right orange port and select Create > Indicator. What this will do is place an indicator on the Front Panel that lets the user see what the data looks like at this point. Indicators are great tools for debugging because they can be placed almost anywhere and give insight to the manipulation of the data.

Step 4: Now that the data is in LabVIEW and is an array type it can now be manipulated. The first thing that is useful to do is pulling data points out of the file. This can be done several ways. The first method lets the user select a single element and remove it from the data, where as the second will allow for multiple points of the data to be removed at once. In order to do this first method of getting a single data point a new function is needed. Open the functions pallet and select Programming > Array > Index Array.



The Index Array will take in the existing array from the Read from Spreadsheet File function and create a new-sub array. As seen above the Index Array function has several ports on it. The top left orange port (in array) will connect to the top right orange port of the read from spreadsheet file (out array).

Note: To connect ports simply click on one of the ports then move the mouse to the second port and click again. A dotted line will appear between the ports showing the wire path after the first port is clicked on. Once the second port is clicked on the dotted line will turn a solid color and this means the connection is valid. LabVIEW uses colored ports to show valid connections types eg. orange ports (array type) can connect to orange ports as shown above. If a connection is not valid the line will simply stay dotted. By hovering the mouse over this invalid connection LabVIEW will explain the problem.

Next there are two blue ports on the Index Array. The top one will affect the rows of the existing array and the bottom one will affect its columns. By creating a constant and setting its value to zero on the upper blue port, (affects rows) it will effectively make the output (sub-array) be the first row of the original array, which in this case will only be one element because the sample data only had one column. Inversely, if a constant of zero was placed on the bottom blue port (affects columns) the sub array would be the entire array for this example because once again there is only the one column. Lastly the port on the right is the output array or sub-array. Sub-array. This is a good location for another indicator to ensure your sub-array is working correctly.

The second Method for data removal is removing multiple elements at once. In order to accomplish this return to the functions pallet and select Programming > Array > Delete from Array.



The top left orange port is the array in port. The top blue port defines length and the bottom blue port defines index. Create two constants here to specify these two values. The length input says how many cells to delete from the array and the index says where to start in the array for the deletion.

Note: If the input array has multiple rows and columns the blue ports will change for the Delete from Array function. The first will remain length, but the second will control rows, and a third will appear that controls columns. This modification will allow a user to pinpoint a portion of data they would like to delete.

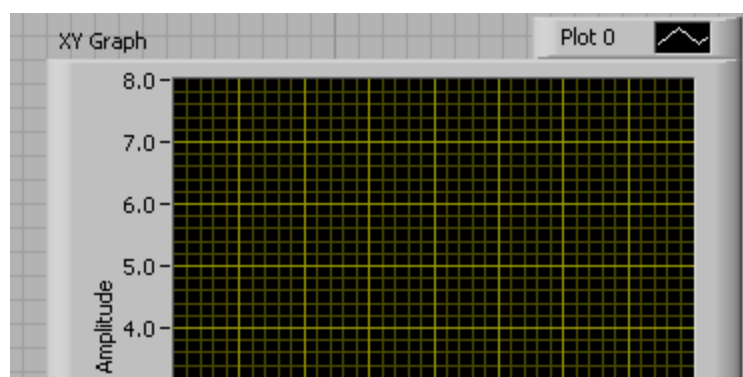
Finally there are two ports on the right side of the function. The upper port outputs the new array with the deleted elements removed, and the bottom port outputs only the elements that were removed.

Step 5: Another useful function is being able to decimate arrays. Open the functions pallet and select Programming > Array > Decimate 1D Array.



The 1D Array decimation is being used because the input file only has one column. There are also 2D Array Decimators in the Array Pallet for larger input files. The orange port on the left is the input port for the array, and the two ports on the right are the output ports which create two new arrays. This function essentially sorts an array. The sorting that is taking place is that every other element is placed on the opposite output. Essentially the odd numbered elements of the original array leave through the top port and the even numbered elements leave through the bottom, thus giving two new sub arrays. Once again indicators would be a good choice here to observe the effects of the Decimator.

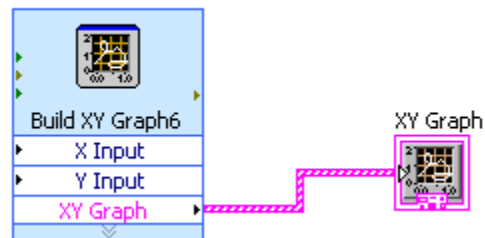
Step 6: In order to graph the arrays that have been previously manipulated switch over to the Front Panel. Similarly to the method used in the Block Diagram right click in the open space and a new function pallet will open. Select Graph Indicators > Express XY Graphs. A blank graph will appear with handles on it that will allow the user to stretch the graph to a desired size.



Note: all of the indicators that were placed on the Block Diagram will be displayed on the Front Panel. They can be positioned anywhere by simply clicking on them and dragging them around; the same is true for the XY Graph.

Next go back to the Block Diagram; a new box will have appeared that corresponds to the graph.

From here simply connect the X and Y inputs of the XY Graph respectively



the array to the inputs of the XY Graph respectively. The array that will convert the array to a Dynamic Data Type which is necessary for plotting. Finally press the play button in the upper left corner of the screen. The program will ask you where the .csv file is located and once specified the data will be manipulated and displayed on the graph.

