PHYC 500: Introduction to LabView

M.P. Hasselbeck, University of New Mexico

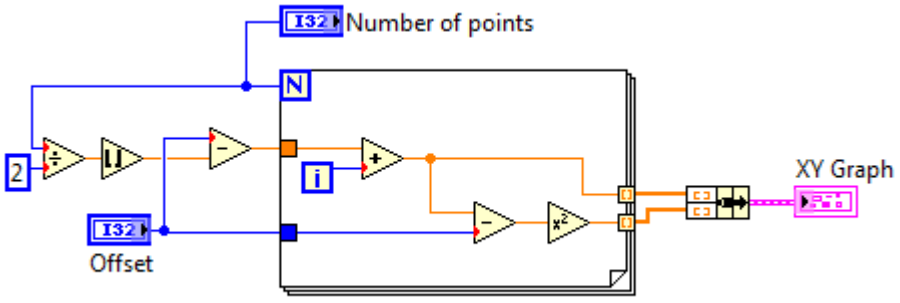
Exercise 8 (v 1.3)

Arrays, XY Graphs, Disk I/O

Place two numeric controls (label them "Number of points" and "Offset") on the Front Panel along with an XY Graph found on the Graph palette. The numeric controls should have I32 representations. Place a For Loop on the Block Diagram. The For Loop will generate two 1-dimensional input and output data arrays and then graph them. The output array is a shifted parabola: $f(x) = (x - x_0)^2$, where x_0 denotes the shift offset and x is read from the iteration terminal **[i]**. The user specifies: i) the number of points in the plot and ii) the offset x_0. When x is read directly from the iteration terminal, the first x-value will be 0. The parabola will appear asymmetric if an offset is present.

The VI should be modified to center the plot close to x_0. To do this, have it calculate each point in the (input) x-array to account for the offset. Simple logic for this is: divide the number of points by 2, round down to the nearest integer, and subtract this from x_0. The rounding function can be found on the Numeric palette. Both the input and output data can be whole numbers or integers, although it is not necessary that all the data be integer format. Calculations using a mixture of floating-point and integer data can be done with coercion (indicated by red dots on input terminals).

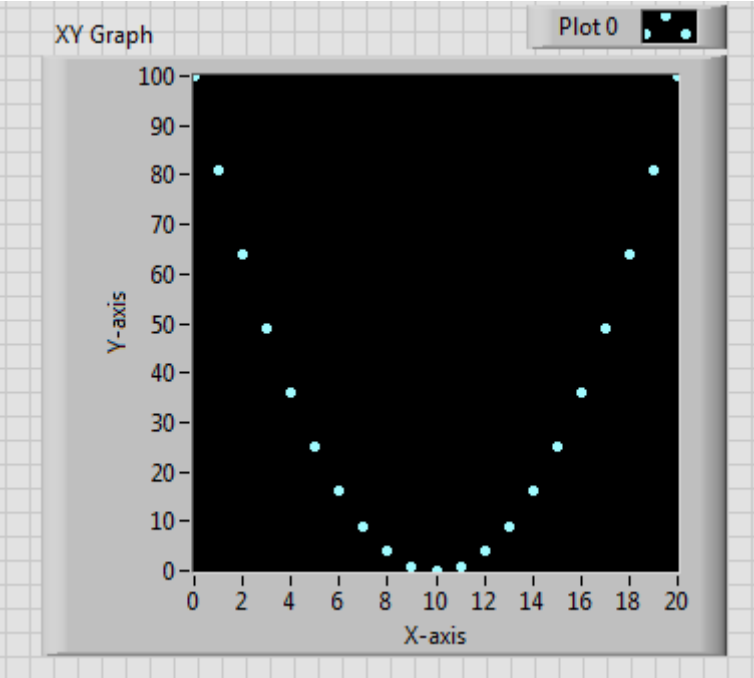
Develop code that uses the iteration terminal **[i]** of the For Loop to produce the x-data. The y-data is then obtained with a simple square operation (x^2). The data representing x and $f(x)$ can be wired to the right-edge of the For Loop as shown in the diagram below.



The two 1-D arrays must then be bundled to plot an XY Graph. Right-click on the diagram and select Cluster, Class, and Variant: Bundle. Wire the x-array to the top terminal and the y-array to the lower terminal of the Bundle icon. Note the thicker orange lines indicating the presence of array data. Wire the bundled cluster to the XY

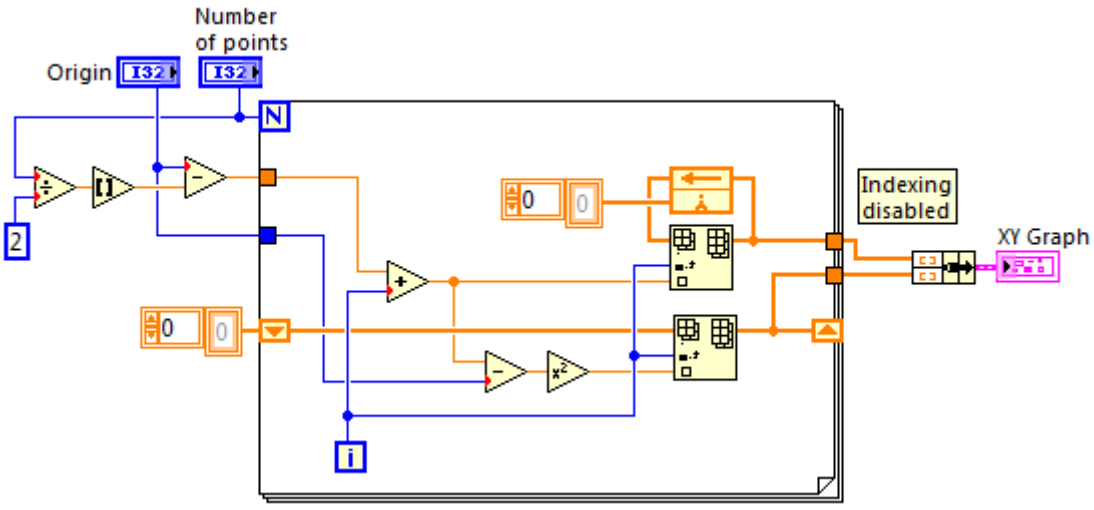
Graph indicator. Test your VI and confirm that the plot is closely centered on the specified value of x_o.

The XY Graph can be customized in the same way as a Waveform Graph. Right-click on the Plot 0 palette to access the controls. Change to a point style plot with solid circular points. Change the point color from default white. Also remove the major and minor grid lines. To do this, right-click on the graph and select Properties. Under the Scales tab, go to Grid Style and Colors. Click on Major and Minor grids and select T (transparent). Do this for both axes and re-name them as well. An XY graph of a 21-point parabola centered at x_o = 10 is shown below.



Develop a second VI that draws the same output plot, except the input data is produced with a Shift Register that increments +1 on each iteration. If you are uncertain about the Shift Register concept, review Exercise 4. Show both VIs to the instructor.

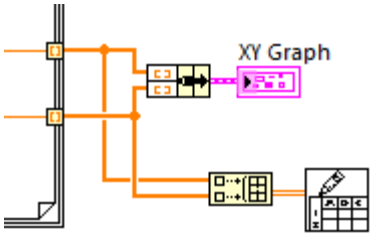
The following Block Diagram illustrates how the parabola in the above VIs can be realized with the x- and y-arrays built entirely *inside* the For Loop. The "Insert into Array" function in the Array palette accepts (from top to bottom) an input array, the index of the new data element (i), and the data element (x_i or y_i). The updated array is passed to the next iteration using a Feedback Node (x-data) and a Shift Register (y-data). Both operations are functionally identical. Note that array indexing has been disabled at the For Loop output as shown by the solid orange tunnels. If indexing is not disabled, the For Loop will generate two 2-D arrays.



The Block Diagram also reveals an important concept in the use of Shift Registers and Feedback Nodes. Both these operations are able to retain their most recent values in memory. These values will persist at the input when the VI is run again *if* they are not initialized. Persistent memory has great utility in advanced LabView programming, but it can cause problems in simpler VIs. It is wise to initialize Shift Registers and Feedback Nodes to zero as shown in the Block Diagram above to prevent bugs in the code. A simple way to do this is to right-click on the initialization terminal and create a constant. The default value is zero, shown as a 1-D array in the diagram.

Writing and reading from disk

The next step is to write the data to disk. There are several methods to accomplish this, but a simple way is with File I/O: Write to Spreadsheet File.vi. The two 1-D output arrays should be combined into a single 2-D array and wired to the write VI. Use the Build Array function (resize it for two inputs) as shown in the following diagram:



Note that the wire appearance has changed to indicate the presence of a 2-D array. When the VI is run, a dialog will open asking where the output should be saved. Choose a name (it's a good idea to use a .txt or .dat suffix) and write the data file to the desktop. Save

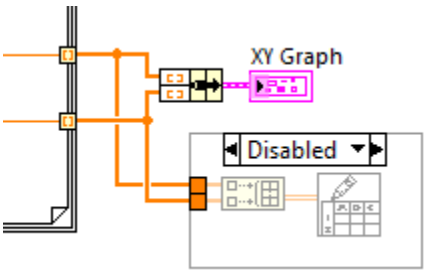
this VI.

The data will be retrieved using a second VI. Open a blank VI and go to the Block Diagram. Select File I/O: Read from Spreadsheet File.vi and place it on the blank diagram. With Context Help enabled (CTRL-H), you can see the large number of input and output terminals available. If the "file path" terminal is left unwired, a dialog box will automatically open asking the user to specify a location. Alternatively, a constant or control could be wired to this terminal with the path information, but for now leave it unwired.

The *x*- and *y*-data that define the parabola are accessed on the output terminal labeled "all rows". To separate the 2-D data into two 1-D *x*- and *y*-arrays, select Array: Index Array and place to the right of the Read from Spreadsheet File.vi. Place a second Index Array below it (press CTRL and drag). Connect the "all rows" output terminal to the **n-dimensional array** input of both Index Array operations. Right-click on the **index 0** input terminal and create a constant. Set the first at 0 and the second to 1, corresponding to the *x*- and *y*-arrays, respectively. The selected 1-D arrays can now be wired to the Bundle operation needed to create an XY Graph. Run the VI and confirm that the file containing the parabola data can be read and plotted. Save this second VI.

Disable Diagram

Go back to the first VI. Adjust one or both of the numeric controls and run the VI. Note that the VI will ask you to save the waveform data to a spreadsheet file each time it is run, but this is not necessary until the desired waveform has been attained. One could delete the File I/O code from the Block Diagram, but it's more convenient to selectively disable it. Right-click and select Structures: Diagram Disable Structure. Use the cursor to draw a box around the file write operation as shown below:



When the desired parabola curve is ready to be saved, simply remove the Disable structure by right-clicking on its edge and highlighting "Remove Disable Diagram Structure". This structure can be very handy for troubleshooting problems in the Block Diagram.

Global Variables

It would be convenient if the reading VI automatically knew the location of the data file generated in the writing VI. This can be implemented by using a Global Variable. Go to the Block Diagram of the writing VI, right-click, select Structures: Global Variable, and place it to the right of Write to Spreadsheet File.vi. Alternatively, find it with Quick Drop (CTRL + spacebar). A Global Variable can be thought of as a SubVI without a Block Diagram. It can hold as many data controls and indicators as needed.

Double-click the Global Variable icon to open it. Right-click on its Front Panel and select String & Path: File Path Indicator. Save this VI to disk (it can optionally remain open). Go back to the Block Diagram and right-click on the Global Variable icon and Select Item: Path. Connect the "new file path" output terminal to the Global Variable icon. On the Front Panel, enter 50 data points and set $x_0 = 0$. Run the VI again and save the spreadsheet file when prompted. The file path is now retained in the Global Variable.

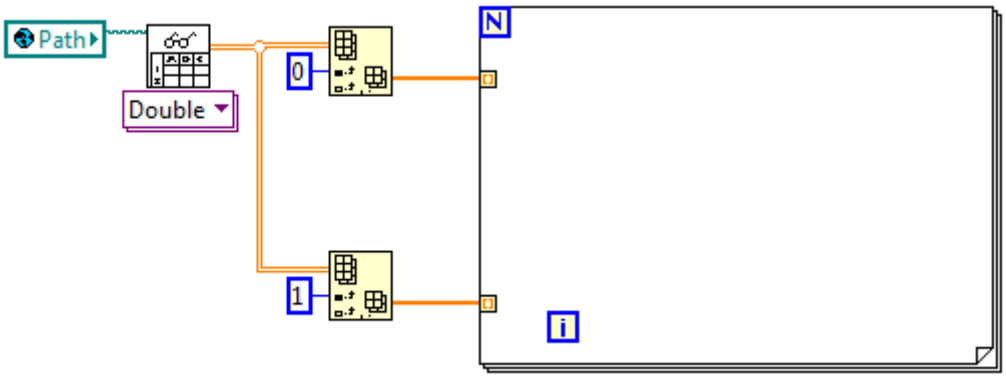
Go to the Block Diagram of the reading VI, right-click, and "Select a VI...". Navigate to the location of the saved Global Variable VI and place it to the left of the Read from Spreadsheet File.vi. Right-click the icon and "Change to Read". The icon can now be wired to the **file path** input terminal. Confirm that the new parabola data is read and displayed.

The Global Variable allows data to be shared between VIs, but it is not persistent. If LabView is closed and re-started, the Global Variables have to be re-written. This also applies to Shift Registers and Feedback Nodes. Demonstrate operation to the instructor.

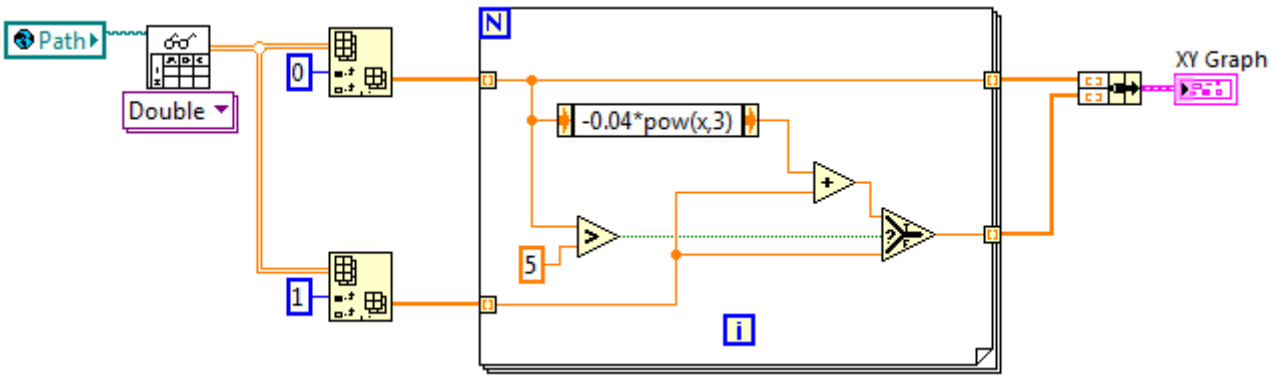
Array Operations

The saved data will be selectively edited to introduce a higher order nonlinearity for the condition $x > 5$. Specifically, the data will be converted to $f(x) = x^2 - 0.04x^3$ for $x > 5$. For $x \leq 5$, the function is unaffected. Open the read data VI and go to the Block Diagram. Disconnect the graph and move it temporarily out of the way. Create a For Loop. The number of iterations will be exactly the number of data points that define the waveform. You may recall that there are 50 points, but LabView can determine this value directly using auto indexing.

When an array is connected to an input tunnel of a For Loop as shown below, the number of iterations is automatically set to the array size. This means the count terminal **[N]** can be left unwired. If multiple arrays with different sizes are connected to the For Loop, the number of iterations is clamped at the smallest array size.



Wire the x- and y-data arrays to the left border of the For Loop; a partially filled tunnel should appear indicating that indexing is enabled. Indexing automatically selects a single x- and y-array element on each iteration; notice that the thicker 1-D array wire changes to a single value wire inside the For Loop. Each element in the x-data array is compared to the reference constant of 5. If $x_i \leq 5$ the data remains unaltered; if $x_i > 5$, then the cubic nonlinearity is added to the y-data. This logic suggests a comparison together with a Case Structure. Because the logic results in a simple numerical selection, it is easier to use the Select function found in the Boolean palette. A value is selected depending on the state of the T/F Boolean input. At the output tunnel a new y-array is built with indexing. The x-array passes through the For Loop unchanged. The complete Block Diagram is shown below:



The cubic term is easily written with an Expression Node. A list of available functions recognized by the Expression Node is found in its Help menu. Operate the VI and confirm the cubic distortion for x > 5.