

## SlopeIntercept Build

This document is intended to guide the reader through building a Virtual Instrument, VI, using LabVIEW. The VI displays the graph and equation of a line with controls for the slope and y-intercept.

This document and the VI are the property of Education Service Center, Region XIII. Please direct any comments and questions about the use or distribution of this document to:

Joules Webb  
Education Service Center, Region 20  
[julianne.webb@esc20.net](mailto:julianne.webb@esc20.net)  
(210) 370-5497

Thank you for reviewing this document and providing any feedback.

Warm regards,

Eric Mann

## Building SlopeIntercept VI

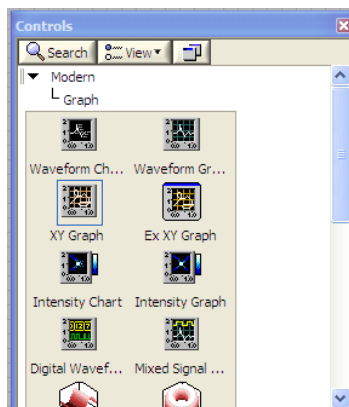
This guide walks the reader through building a Virtual Instrument, VI, using LabVIEW. This VI captures the relationship between algebraic and graphic representations of the linear function,

$$y = mx + b,$$

where  $m$  is the slope and  $b$  is the  $y$ -intercept of the line. We will include controls for each of these parameters as well as indicators for the graph and the equation. This VI can be run continuously for dynamic interaction and an immediate connection between the changing parameters, graph, and equation.

- 1) Create a new VI by selecting **File»New VI** or pressing **<Ctrl-N>**
- 2) Add an **XY Graph** to the front panel
  - a. Select **View»Controls Palette** or right-click on the front panel to open the **Controls** palette
  - b. Tack the **Controls** palette down by clicking the thumb tack in the upper left corner of the palette
  - c. Click on the double arrows at the bottom of the **Controls** palette to expand the menu
  - d. Select **XY Graph** from **Modern»Graph**
  - e. Click on the front panel to place the graph

There is also an **Express XY Graph**. Be sure to use the **XY Graph** found from **Modern»Graph**.



**Figure 1.** Controls palette

By default, the  $x$  and  $y$ -values on the axes will automatically adjust depending on what is graphed. This is called autoscaling. Autoscaling can be turned off. Then, the graph's window can be controlled using the values displayed on the X and Y Scales. Let's configure the graph for our purposes.

3) Configure the **XY Graph**

- a. Turn off autoscaling
  - i. Right-click on the graph and select **X Scale»AutoScale X** to deselect autoscaling
  - ii. Right-click on the graph and select **Y Scale»AutoScale Y** to deselect autoscaling
- b. Adjust the limits on the X axis by clicking on the left and right limits on the X axis and setting them to  $-10$  and  $10$  respectively
- c. Adjust the limits on the Y axis by clicking on the left and right limits on the Y axis and setting them to  $-10$  and  $10$  respectively
- d. Change the labels of the axes from "Time" and "Amplitude" to "x" and "y"
  - i. Double-click on each word to select it and then type the desired text

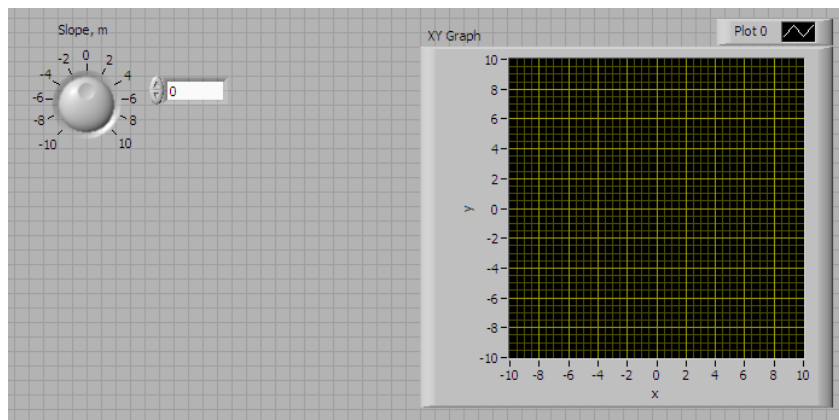
With the graph in place, start adding controls for the slope and  $y$ -intercept.

4) Add a knob to control the slope of the line

- a. From the **Controls** palette, select **Express»Numeric Controls»Knob**
- b. Click to the left of the **XY Graph** to place the knob on the front panel
- c. Name the knob "Slope, m"
  - i. If necessary, double-click where it says "Knob" to select the text
- d. Change the lower limit of the knob to  $-10$  (Note: These values can be changed even while the VI is running.)

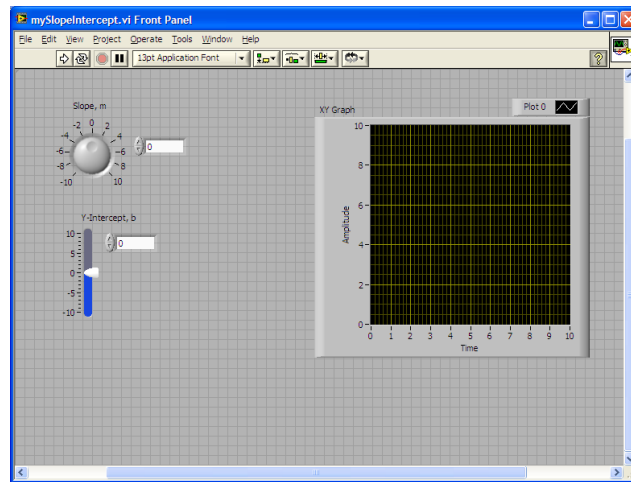
Many numeric controls have the option of having a **Digital Display**. The **Digital Display** not only shows more precisely what value is being selected, it can be used to enter a specific value.

- e. Right-click on the knob and select **Visible Items»Digital Display**



**Figure 2.** Front panel displaying XY Graph and Slope knob

- 5) Add a slider to control vertical translation, i.e. the y-intercept, of the graph
  - a. From the **Controls** palette, select **Express»Numeric Controls»Vertical Pointer Slide**
  - b. Name the slider “Y-intercept, b”
  - c. Change the lower limit of the slider to  $-10$
  - d. Right-click on the slider and select **Visible Items»Digital Display**
- 6) Save your VI
  - a. Select **File»Save** or press **<Ctrl-S>**
  - b. Call the VI **mySlopeIntercept**



**Figure 3.** Front panel displaying graph and controls

The block diagram of a VI displays the code that executes when the VI is run. You will add VIs to the block diagram to use the  $m$  and  $b$  to generate the graph of a line.

- 7) Select **Window»Show Block Diagram** or press **<Ctrl-E>** to view the block diagram

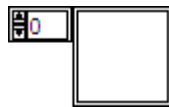
The block diagram will have three icons on it, one for each of the controls and one for the XY Graph. You will connect these icons using wires in such a way that a line is displayed in the graph.



**Figure 4.** Icons on block diagram

For your  $x$ -values, you will create an array with two  $x$ -values,  $-100$  and  $100$ . It will take a little bit of work to set up the array. However, it will be worth it. Working with an array of values will be much simpler than working with each  $x$ -value individually.

- 8) Add an array to store the  $x$ -values
  - a. Right-click on the block diagram to bring up the **Functions** palette
  - b. Tack the **Functions** palette down by clicking the thumb tack in the upper left corner of the palette
  - c. Click on the double arrows at the bottom of the **Functions** palette to expand the menu
  - d. Select **Programming»Array»Array Constant**
  - e. Click on the block diagram to place the constant



Creating an array with constant values is a two-step process. By default, the array constant is an empty shell and is black in color. To set the type of array you want, you add a numeric or string constant to the array. You want the array you have just added to be a numeric array.

- f. To have the array be a numeric array, select **Programming»Numeric»Numeric Constant** and click inside of the array



By default, the numeric constant is a 32-bit integer, signified by the blue color of the array. You want the array to hold a double precision number (rational number) as opposed to an integer.

- 9) Change the data-type of the array
  - a. Right-click on the array and select **Representation»Double Precision** from the context menu
  - b. Note the array changes color from blue to orange to denote it is now an array of double precision numbers



- 10) Resize the array to hold two values
  - a. Click on the right-hand portion of the array and then click and drag downwards to display another element of the array
- 11) Set the values of the array elements
  - a. Click on the first element of the array and then click and drag to the right to increase the size of the element
  - b. Double-click on the first element and set its value to  $-100$
  - c. Double-click on the second element and set its value to  $100$

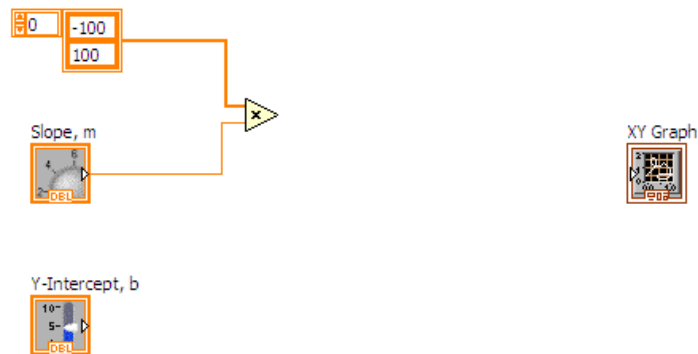


**Figure 5.** Block diagram with array constant, controls, and XY Graph

Next, you will add the code needed to operate on those  $x$ -values to get  $y$ -values. Follow the basic logic of the equation,  $y = mx + b$ . Multiply each  $x$ -value by  $m$  and then add  $b$ . This can be done simply.

- 12) Multiply the array of  $x$ -values by  $m$ 
  - a. Select **Programming»Numeric»Multiply** and click on the block diagram to place the VI
  - b. Wire the array of  $x$ -values to the  $x$ -terminal of the **Multiply** VI
  - c. Wire the **Slope,  $m$**  to the  $y$ -terminal of the **Multiply** VI
- 13) Save your VI, <Ctrl-S>

To wire the components together, hover the mouse over the terminal of the array until the wiring tool appears. Then, hold down the left mouse button and drag the cursor to the destination terminal. When the terminal appears, release the mouse button to connect the wire.



**Figure 6.** Block diagram with Multiply VI

The multiplication of the scalar and the array happens element by element. That is, each element of the array gets multiplied by the scalar value,  $m$ . This is also the case for addition.

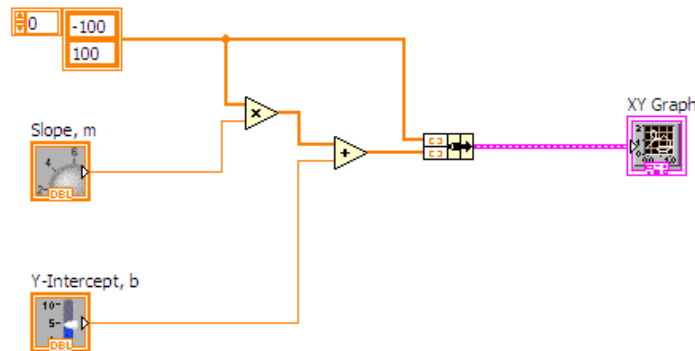
- 14) Add the  $y$ -intercept to the result of the multiplication
  - a. Select **Programming»Numeric»Add** and click on the block diagram

- b. Wire the output of the **Multiply** VI to the x-terminal of the **Add** VI
- c. Wire the **Y-intercept, b** to the y-terminal of the **Add** VI

Now that you have multiplied the array of  $x$ -values by  $m$  and then added  $b$ , you have an array of  $y$ -values corresponding to the array of  $x$ -values. The **XY Graph** uses both of these arrays of values to graph the line. To connect the two arrays to the **XY Graph**, you will bundle the arrays together as a cluster.

A cluster is like an array. It can hold multiple elements. The two types differ in that a cluster can hold multiple data types, while an array holds multiple elements of a single data type.

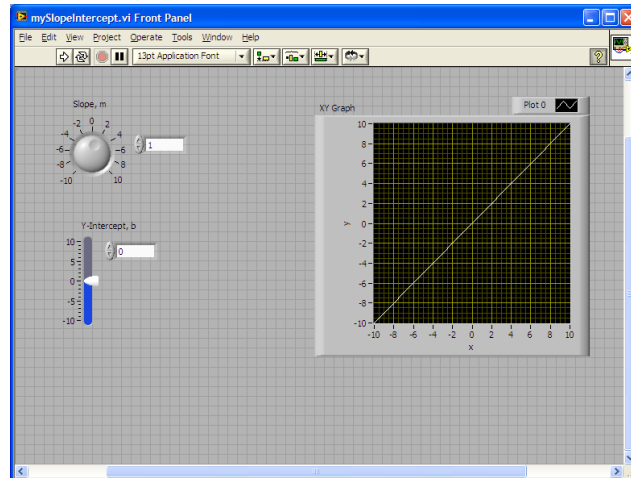
- 15) Bundle the  $x$  and  $y$  arrays into a cluster
  - a. Select **Programming»Cluster, Class, & Variant»Bundle** and click on the block diagram to place the VI
  - b. Continue the wire from the original  $x$ -array and connect it to the first element of the **Bundle** VI
  - c. Wire the output of the **Add** VI to the second element of the **Bundle** VI
- 16) Wire the output of the **Bundle** VI to the **XY Graph**
  - a. Note: the **XY Graph** will change from brown to pink to denote its data type as a cluster of numeric arrays
- 17) Save your VI, <Ctrl-S>



**Figure 7.** Block diagram with  $y = mx + b$  implemented

Your VI is now ready to run.

- 18) Return to the front panel by pressing <Ctrl-E>
- 19) Set  $m$  and  $b$  before you run the VI
  - a. Use the knob to set the slope to 1
  - b. Use the slider to set the  $y$ -intercept to 0
- 20) Run the VI by press the **Run** button or by pressing <Ctrl-R>



**Figure 8.** Graphing a line

Notice the origin is in the center of the graph. The scales on the left and bottom of the graph are not the axes. The axes run where  $x = 0$  and  $y = 0$ .

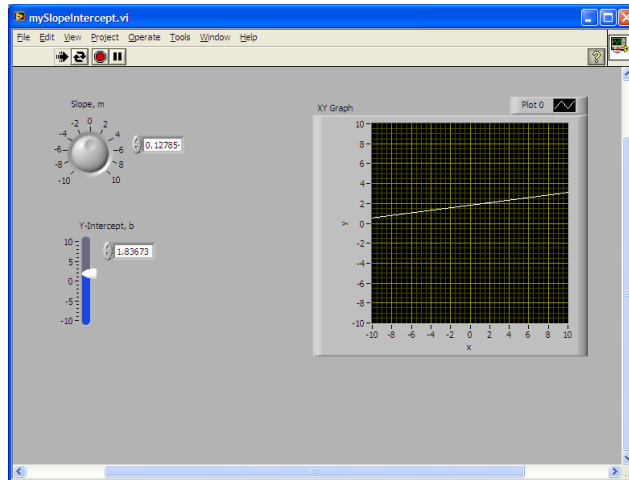
You can set these values for slope and y-intercept as the default values for the VI. These will be the values shown when the VI is first opened.

- 21) From the LabVIEW menu, select **Edit»Make Current Values Default**
- 22) Save the VI, **<Ctrl-S>**

You can run the VI using the **Run** button or the **Run Continuously** button. Running continuously will have the code in the block diagram executed over and over again. To stop the execution, use the **Abort Execution** button.

- 23) Run the VI continuously
- 24) Change the values of  $m$  and  $b$  and notice the graph adjusts accordingly

You should be seeing the graph of the line respond immediately to changes in the slope or y-intercept.



**Figure 9.** Continuous execution of VI with interactive controls

25) Stop the VI

You could stop here, and wouldn't it be great if you could see the equation for the line change as the graph is changing?

It is possible to capture the values of the controls and display them in an equation. LabVIEW has tools for turning numeric data into text. You will use one of these tools to display the equation of the line.

- 26) Return to the block diagram, select **Programming»String»Format Into String**
  - a. Click on the diagram to place the VI
  - b. Press **<Ctrl-H>** to open the **Context Help** and see what this VI does

The **<Ctrl-h>** keys in keyboard shortcuts correspond to the **(Mac OS)** **<Shift-Command-h>** keys.

- c. Click on the link to detailed help from the **Context Help** for **Format Into String**

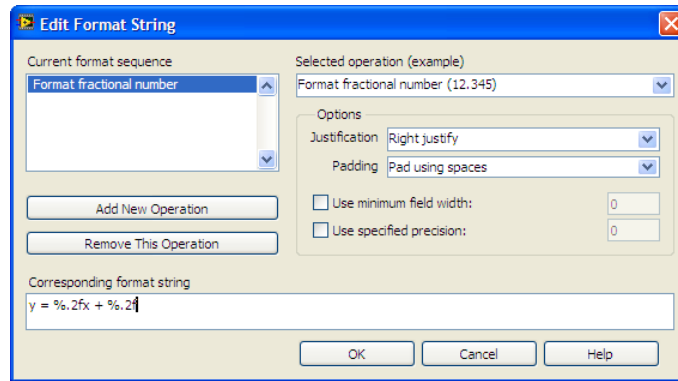
We will format our numeric data into string data using the **Format Into String** VI. To do this, we will need to create a template for how to display the data.

- 27) Right-click the **Format Into String** VI and select **Edit Format String** from the context menu to create and edit the template
  - a. Use the **Edit Format String** window to build the following expression:

$$y = \%.2fx + \%.2f$$

This expression takes each numeric input and displays 2 digits to the right of the decimal. Also, the numeric data is set into the familiar form,

$$y = mx + b.$$



**Figure 10.** Edit Format String window

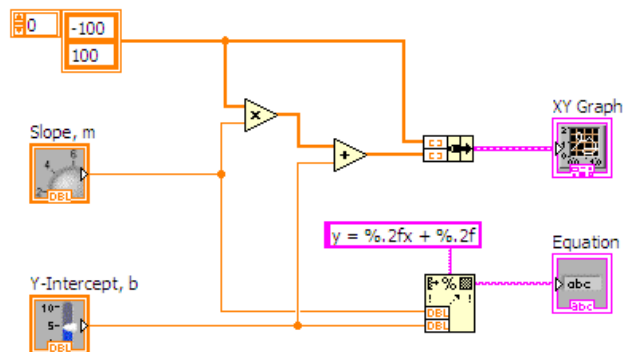
- b. Click OK to close the **Edit Format String** window

The **Format Into String** VI will now have terminals for two double precision (DBL) inputs. You will connect our controls to these terminals being careful to display them in the proper order. The first input, **input 1**, is for the Slope ( $m$ ). The second input is for the Y-intercept ( $b$ ).

- 28) Connect the controls to the **Format Into String** VI
  - a. Connect the wire from the **Slope,  $m$**  control to **input 1**
  - b. Connect the wire from the **Y-intercept,  $b$**  control to **input 2**

Now that the inputs for **Format Into String** are set, we need to add an indicator to display our equation.

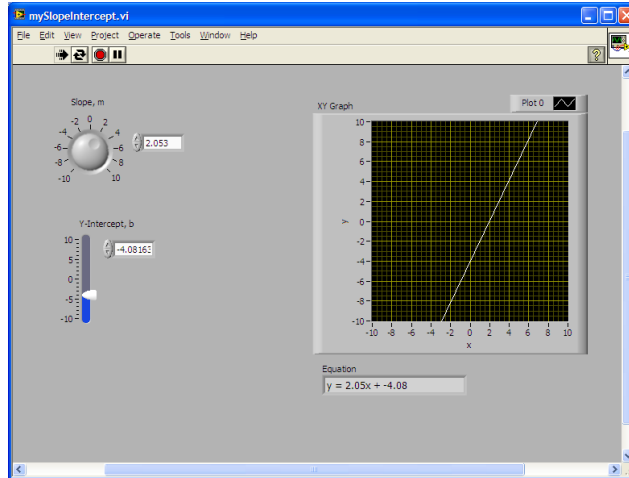
- 29) Right-click on the **resulting string** terminal of the **Format Into String** function and select **Create»Indicator**
  - a. Change the label of the indicator to **Equation**



**Figure 11.** Linear equation configured for display

- 30) Go the front panel and adjust the size of the **Equation** indicator
  - a. Click on the indicator to select it
  - b. Drag the right edge to the right to increase the size of the indicator

- c. Click inside of the indicator to display a cursor
  - d. Adjust the size of the text by pressing <Ctrl-+>
  - e. The **Text Settings** in the toolbar will show the font and font size
- 31) Save, then run the VI
- a. Use the controls to transform the line
  - b. Notice the equation and the graph respond



**Figure 12.** Line and equation displayed in executing VI

The SlopeIntercept VI is complete. Of course, more customization is possible. Right-click on any of the controls or indicators to see what else could be made visible or what properties could be changed. Many display options are configurable.