LabVIEW Day 1 Basics

Vern Lindberg

LabVIEW first shipped in 1986, with very basic objects in place. As it has grown (currently to Version 10.0) higher level objects such as “Express VIs” have entered, additional programming constructs have been added, and the user interface has been vastly improved (thank heavens for polymorphism!)

Let’s start by looking at LabVIEW and building a “simple” program. This will introduce you to the parts of LabVIEW (panels, palettes, tools), some of the data types, and one of the graphing objects.

Along the way many of the features of LabVIEW will be introduced, and ways of finding getting help.

1  The Look of LabVIEW

1. Open National Instruments, LabVIEW Click on “Blank VI”. LabVIEW is written in G, a graphical programming language. Programs in LabVIEW are called virtual instruments or “vi”s and have that extension.

Most computer languages (FORTRAN, C++, ...) use control flow, that is steps in the program occur with explicit ordering. LabVIEW uses dataflow. Each icon on the Block Diagram will execute once all its inputs are provided. This allows parallel execution of tasks. There are special structures that allow you to force things to execute in a specific order, should that be needed. This discussion will be deferred until another day.

Most of the time you will using LabVIEW in the developer mode. The code is quickly compiled when you run the program. When you have a complete application it can be saved in a stand alone mode that does not need the whole LabVIEW program.

2. There are several windows in LabVIEW. The following should always be open for your use.

• Front Panel—this is like the front of a piece of equipment
Fig. 1: Front Panel and Connector pane with Numeric Control for frequency, Boolean Control for loop, and Graphical Indicator for output

- Block Diagram—this is where the “coding” is done Ctrl-E or Cmd-E switches between Front panel and Block diagram

Fig. 2: Block Diagram and Connector pane with While Loop, Simulate Signal, Numeric Control for Frequency and Graphical Indicator for output
• Icon/Connector Pane—it is the upper right part of both the Front panel and the Block Diagram, and used for subprograms, “Subvi”s. More on this later.

• Tools Panel

![Tools Panel](image)

Fig. 3: The tools palette

• Contextual Help: This is a brief help function that displays help on the object over which you place your mouse. Click on the ? on the menu bar for front panel or block diagram to toggle it on or off.

![Contextual Help](image)

Fig. 4: Contextual Help for Waveform Graph

3. Palettes

When you are in the Front Panel, a right-click will pop up the Controls Palette that contains items that go on the front panels: these are called Controls (get input from a user), Indicators (display results), and Decorations (make a pretty front panes). The various items are different sub-menus (Numeric, Boolean, String & Path, etc.) Use only the Modern menu, as the system menu and classic menu are outdated.
In addition, double clicking on either panel will open a text box where you can write comments and user instructions.

![Controls palette]

**Fig. 5:** The Controls palette, accessible only from Front Panel

When you are in the Block Diagram, a right-click will bring up a *Functions Palette.* This is where the programming is done.

![Functions palette]

**Fig. 6:** Functions palette accessible only from block diagram

If you want the palettes to remain in view (often helpful if there is space), use the “push pin” in the upper left corner to pin any palette down.
To move items onto the front panel or block diagram, click on a palette item, then click (or click and drag) in the window.

4. **Useful Shortcut Keys**  
   \[\text{Cmd} = \text{Apple key (Mac)} \text{ or } \text{Ctrl key (PC)}\]  
   | **Cmd-E** | Switch Between Front Panel and Block Diagram |
   | **Cmd-B** | Delete broken wires |
   | **Cmd-I** | VI Properties |
   | Right Click | Contextual items |
   | **Ctrl-drag (PC)** | make duplicate copy of an object or collection |
   | **Alt-drag (Mac)** | |

5. Icons for Run, Run Continuous, Abort (i.e. emergency stop), Pause/Resume, Font Settings, Align, Distribute, Reorder, and Context Help are at the top of both windows. Resize Objects is at the top of the front panel, Five Debugging Icons are at the top of the block diagram.

6. Contextual Help is very useful in learning Labview. Activate Contextual help from the Help menu, or by using the ? icon on the window. In addition there is a much more detailed Labview help available on the help menu that includes links to sample programs.

## 2 Data types

Programming a Labview application consists of placing icons on the Block Diagram, placing controls and indicators on the Front Panel (we will see ways to do this from the Block Diagram), changing the Properties of the elements, and wiring the Block Diagram.

There are many data types in Labview, each with a unique color or color combination. Some of these are

<table>
<thead>
<tr>
<th>Color</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>blue</td>
<td>Integer</td>
<td>signed or unsigned, 1, 2, 3, or 4 bytes</td>
</tr>
<tr>
<td>orange</td>
<td>Floating point</td>
<td>Fixed point, single, double, extended precision</td>
</tr>
<tr>
<td>green</td>
<td>Boolean</td>
<td>True False</td>
</tr>
<tr>
<td>pink</td>
<td>String</td>
<td>Hello 1999 World!</td>
</tr>
<tr>
<td>yellow/black</td>
<td>Error</td>
<td>to be discussed later</td>
</tr>
<tr>
<td>brown/pink</td>
<td>Cluster</td>
<td>see below</td>
</tr>
<tr>
<td>blue/white</td>
<td>Dynamic Data Type (DDT)</td>
<td>on Express submenu</td>
</tr>
</tbody>
</table>

Consider a floating point number like 3.5 that could be a single measurement of a temperature. Often we need to measure temperature at several times. This would be stored as a **one dimensional floating point array of numbers**. Each element of the array is the same type of thing, here a floating point number. We can have arrays of almost anything,
Booleans, strings, waveforms, . . . . Arrays can be one dimensional (vector) two dimensional (table), or more. While a single number will be shown as a thin line, a 1D array will be a thicker line, and a 2D array a double line, etc.

Clusters of data can contain information of different types, such as a single number, an array, and a string.

We can also have arrays of clusters and clusters of arrays. You’ll see these later in the course.

Polymorphism is a term meaning that a single icon can operate on data of different types. A simple example is an add icon. If we wire two integers to the inputs, the output is an integer. If we add two floating point numbers, the output is a floating point number. If we add one integer and one floating point number, the icon will produce a floating point output and a coercion dot will be placed at the integer input. We can add two arrays, we can add a two waveforms, or even an array and a waveform.

Coercion dots will serve to convert data from one data type to another but there will be some loss in speed. Conversion icons serve to do the conversion quickly and with minimal impact on required memory. Numeric data types are shown in Table 1

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>I8</td>
<td>Signed 8 bit integer</td>
<td>U8</td>
<td>Unsigned 8 bit integer</td>
</tr>
<tr>
<td>I16</td>
<td>Signed 16 bit integer</td>
<td>U16</td>
<td>Unsigned 16 bit integer</td>
</tr>
<tr>
<td>I32</td>
<td>Signed 16 bit integer</td>
<td>U32</td>
<td>Unsigned 16 bit integer</td>
</tr>
<tr>
<td>I64</td>
<td>Signed 16 bit integer</td>
<td>U64</td>
<td>Unsigned 16 bit integer</td>
</tr>
<tr>
<td>SGL</td>
<td>Single Precision Floating Point</td>
<td>CSG</td>
<td>Complex Single Precision Floating Point</td>
</tr>
<tr>
<td>DBL</td>
<td>Double Precision Floating Point</td>
<td>CDB</td>
<td>Complex Double Precision Floating Point</td>
</tr>
<tr>
<td>EXT</td>
<td>Extended Precision Floating Point</td>
<td>CXT</td>
<td>Complex Extended Precision Floating Point</td>
</tr>
<tr>
<td>FXP</td>
<td>Fixed Point</td>
<td>icon</td>
<td>128 bit time stamp</td>
</tr>
</tbody>
</table>

3 A simple program

3.1 Showing a sine wave

Open a new blank vi. An empty front panel and an empty block diagram should appear. Go to the block diagram. From the Express sub menu pick Input then Simulate Signal. Henceforth this will be written Express→Input→Simulate Signal.
Another window will open, and on it choose a sine wave of frequency 10 Hz, with amplitude 1 and Add Noise of amplitude 0.3. Click OK and the express vi will build.

We want to see the results, which emerge from the right of the icon at the Sine With Uniform Noise— terminal. Hover the terminal output and right click to bring up a menu. Create → Graphical Indicator and a Waveform Graph icon should appear. If you go to the front panel you will see the graph. Run the program and you will see one complete period of a signal.

### 3.2 Controlling the frequency and running continuously

It would be nice to have a control for the frequency. Place your cursor over the frequency input of the Simulate Signal, right click and Create → Control which should result in a control. Go to the front panel and you will see a box in which you can type a number for another frequency. Alternately you can use the small Increment/Decrement arrows to change values.

Change the value from 10 to 20, and run. Does it do what you expect?

So far the program runs once, producing a sample of a sine wave with 100 data points spaced by 1 ms for a total time of 0.1 s, displays the signal, and stops. Often we want to have the data continually generated until we say “stop.” This is the role of a While Loop.

Go to Programming → Structures → While Loop and surround your existing code with a while loop. In addition to the gray border of the while loop, two other icons appear, an i, and a little stop sign. The index, i, or loop iteration counter, starts at 0 and counts the number of times the loop is evaluated. This can be useful, but you are not required to use it.

The stop sign is more critical. You may notice that the run button is broken. Click on it to see the error “While Loop: Conditional Terminal not wired.” There must be a way to stop the loop from repeating. Right click on the stop sign and Create Control and a Boolean button will appear labeled Stop. On the front panel you can see the control.

We will look further into while loops another day, but for now run the program and you will be able to change the frequency until you click the Stop button. The display may look like an untriggered 'scope at some frequencies.

---

1 There are 3 fundamental types of graphical indicator. Here we have Waveform Graph. The others are Waveform Chart and XY Graph
3.3 Adding some analysis

In a real application, instead of simulating a signal we would measure a real signal and do some analysis of it. We have simulated the data with the Simulate Signal and will use an analysis tool from the express menu.

Deposit an Express→Signal Analysis→Spectral Measurements icon inside your loop, and in the window that pops up select Power Spectrum, dB, then click ok. Power spectrum is related to a Fourier transform, specifically it is the Fourier Transform of the Auto-Correlation function. I'll leave details of that to other courses.

Wire the simulated signal to the Signals input of the Spectral Measurement and create a Graph Indicator. Run the program and you should be able to see a peak at the frequency you designated.

3.4 Changing the graph

Look at the front panel graphs. There are several parts to the display: the graph itself, a label (same as what appears on the block diagram) above and to the left of the graph, and a Plot Legend above and to the right of the graph that has words and a small icon of the graph.

The label, “Waveform Graph” just seems in the way. Right click in the graph area, find Visible Items→Label and uncheck the label. It should be gone on the front panel, but still be present on the block diagram.

You can right click on the axis and see options there, including Autoscale. You can right click on the small icon within the Plot Legend and change things like the Color of the line, or the line width, and also Common Plots that allows you to choose whether to show a line, the individual data points, both, as well as other options. For now show individual points with a line connecting them.

Some of these properties of the graph are also on the Properties. Control of the properties you have seen, and others, is an advanced topic that we will touch on later when we introduce Property Nodes.

3.5 Cleaning up the block diagram and front panel

Aligning icons on both panels can be easily done with the Align Objects on the tool strip of the panel. Icons and wires can be moved by small amounts using the arrow keys, or by larger amounts by using shift-arrow.
Labels can be moved separately from the icons they describe. A crooked wire can be neatened up by right clicking on it and *Clean Up Wire*. Finally, the appearance of the icons can be changed. Right click on a control and uncheck *View as Icon* and the control or indicator will reduce in size. Right click on *Simulate Signal* and check *View as Icon*. Right click on *Spectral Measurements* and *Icon Style → View as Subvi*.

The larger icons give more information, however as we build more complicated vis, we will appreciate having a cleaner look and more space, and will tend to use smaller icons.

### 3.6 Making a more complicated signal

Duplicate the signal generation by dragging over the frequency control and Simulate Signal icons, and holding *control* (PC) or *option* (Mac) then dragging the icons to a blank space. The previous names are used but a number is added making names like *Frequency 2*.

We will add the two signals together using an *Programming → Numeric → Add*. Place this on the block diagram and wire the two signals to the two inputs. Rewire the graph and the Spectral Measurements to the output of the add. Set the frequencies to 40 and 100 and run. The signal should look complicated, but on the Power Spectrum you should see two peaks at the two frequencies.

### 3.7 Adding a cursor to a graph

It would be nice to read data off the graph, and this is easily accomplished.

Right click on the front panel graph and check *Visible items → Cursor Legend* and a blank legend will appear to the right of the graph. Right click on the blank space and *Create Cursor → Single Plot* and a cursor will appear that is tied to the data on the plot.

You can change properties of the cursor (such as its color) by right clicking on the cursor legend and choosing *Properties*.

### 4 What you should now know

- Parts of LabVIEW including Block Diagram, Front Panel, Control Palette, Functions Palette, Tools Palette, Context Help, and how to switch between them.
- While Loop with Index and Stop, Numeric control, Boolean control, Waveform graphs
• Right clicking tricks, Create . . . , Properties, . . .
• Appearance of floating point, Boolean, and DDT lines.
• Keyboard shortcuts, and ways to neaten up a front panel or block diagram
• Changing look of a graph, adding a cursor