Document Type: Tutorial
NI Supported: Yes
Publish Date: Jan 12, 2011

What is the NI LabVIEW MathScript RT Module?

Overview

LabVIEW MathScript RT is an add-on module for the LabVIEW Full and Professional Development Systems. It is designed to natively add text-based signal processing, analysis, and math into the graphical development environment of LabVIEW. With more than 800 built-in functions, LabVIEW MathScript RT gives you the ability to either run your existing custom .m files or create them from scratch. Using this native solution for text-based math, you can combine graphical and textual programming within LabVIEW because the text-based engine is part of the LabVIEW environment. With LabVIEW MathScript RT, you can choose whether graphical or textual programming is most appropriate for each aspect of your application.

Table of Contents

Key Terminology

MathScript RT Module – The LabVIEW MathScript RT Module is the add-on product for the LabVIEW development system and contains the technologies listed below.

MathScript – MathScript is the engine that accepts general .m file syntax and translates that into the G language of LabVIEW. The MathScript Engine does a lot of the “behind-the-scenes” work discussed later in this article.

MathScript Interactive Window – The MathScript Interactive Window is one of two methods for interacting with the MathScript Engine. It is a floating window accessed from the LabVIEW toolbar and is intended for developing your .m files.

MathScript Node – The MathScript Node is the other method for interacting with the MathScript Engine. The MathScript Node is a structure on the LabVIEW block diagram and is accessed from the functions palette. Although sufficiently useful for developing your .m files, the primary function of the MathScript Node is to execute your .m files inline with LabVIEW G code.

Why Should You Use the MathScript RT Module?

The question you ask with every product you encounter is, “Why should I use this product?” The following sections outline several of the primary benefits of using the MathScript RT Module.

MathScript Provides an Alternative Approach for Developing Mathematical Algorithms

G programming is performed by wiring together graphical icons on a diagram, which is then compiled directly to machine code so the computer processors can execute it. This approach aligns with the way most scientists and engineers mentally approach their problems (as in the sense of laying out a solution on a white board). Although intuitive and graphical, this approach can complicate the development of mathematical algorithms because of the graphical nature. Consider Figure 1:

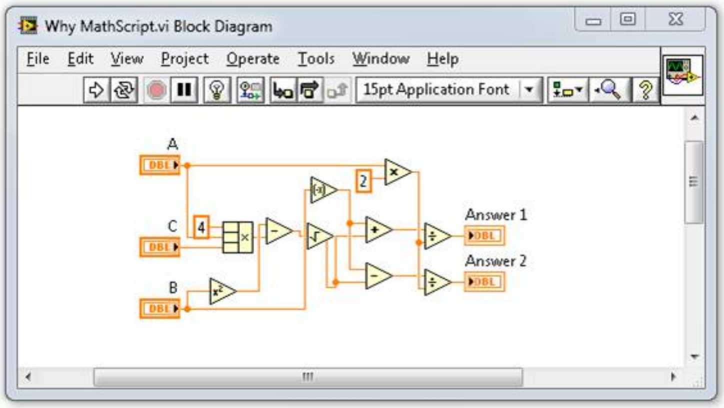
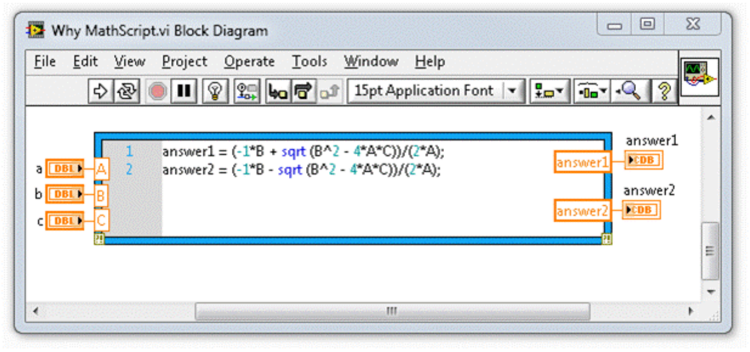

Why MathScript.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
A
C
B
4
2
Answer 1
Answer 2
DBL

Figure 1. G code is performing what appears to be a complex equation.

Textual math is an alternative approach to programming in the graphical development environment of LabVIEW. Even without knowing what syntax the code is using, it is much more intuitive to see Figure 2:


Why MathScript.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
a
b
c
DBL
A
B
C
answer1
answer2

```
answer1 = (-1*B + sqrt (B^2 - 4*A*C))/(2*A);
answer2 = (-1*B - sqrt (B^2 - 4*A*C))/(2*A);
```

Figure 2. MathScript code is calculating the quadratic equation.

In both cases, the code is calculating the quadratic equation. It is much clearer in the textual syntax. In most purely mathematical algorithms, or equation-type calculations, it is much cheaper in the way of time, complication, and block diagram space to use textual math.

MathScript Allows You to Reuse Your Existing .m Files Without Having to Rewrite Them

Simplifying IP reuse is quickly becoming a must-have in any modern-day software application. Every software environment has strengths and weaknesses relative to others, and today's casual user is much more adept in using multiple applications within the same application. Most .m file environments, such as The MathWorks Inc. MATLAB® software and Digiteo Scilab, are great tools for algorithm development. The .m file has become a general syntax used by many different environments.

As with many companies, you probably have a library of IP that you (or someone else at your company) have spent years developing and perfecting. There is no reason to reimplement that IP in a different language. The LabVIEW MathScript RT Module lets you simply import your existing .m files and run them as part of your LabVIEW program.

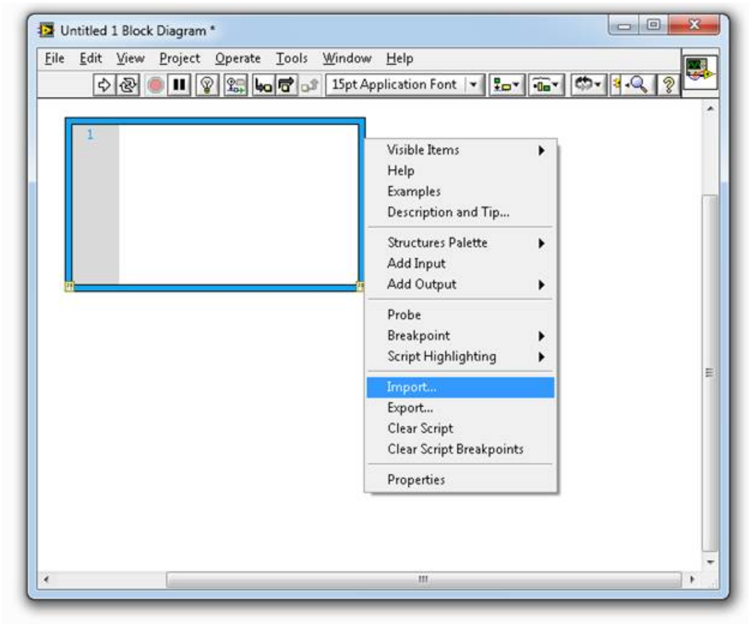


Figure 3. Use the MathScript Node to import your existing .m files to use them with LabVIEW.

Because MathScript is native to LabVIEW, you don't need to have the third-party software on the computer that is running your application. This is a great advantage when you are trying to deploy your IP to a machine dedicated to the deployed application, a compact solution, or embedded hardware.

MathScript Allows You to Perform Your Analysis While You Are Acquiring Your Data

Raw data from the real world does not always immediately convey useful information. Usually, you must transform the signal, remove noise disturbances, correct for data corrupted by faulty equipment, or compensate for environmental effects, such as temperature and humidity. For that reason, signal processing, which is the analysis, interpretation, and manipulation of signals, is a fundamental need in virtually all engineering applications.

Most vendors of data acquisition hardware provide some sort of interface to give you the ability to acquire and save your data to a file. Whether that interface is a proprietary software product or a DLL with function calls from ANSI C or C++, the process is generally trivial to an experienced programmer. Likewise, most math packages provide the necessary built-in functions to fully analyze your data, whether that requires some filtering, transforms, or noise reduction. However, the problem generally lies in the movement of data between these applications. This is because you can't actually perform the analysis of the signal **while** you are acquiring the signal.

This might seem trivial, but it is necessary when you need to perform actions based on the results of that analysis or correlate anomalies in the data with happenings in the real world. The LabVIEW MathScript RT Module gives you the power to combine your .m files inline with the acquisition of data, meaning your analysis happens as you are acquiring the data, providing results in real time. Consider Figure 4:

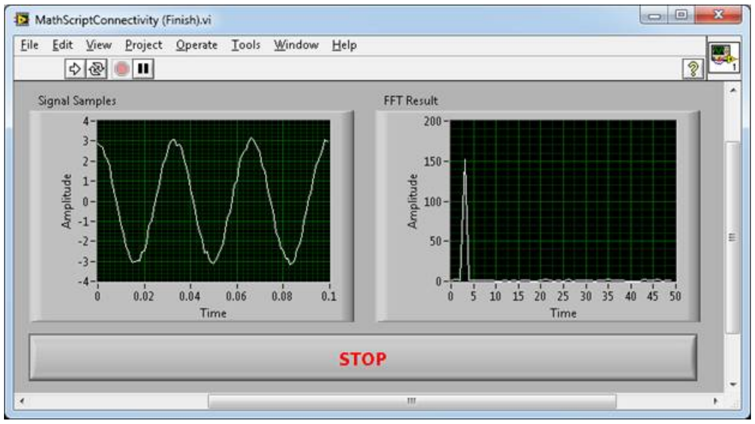
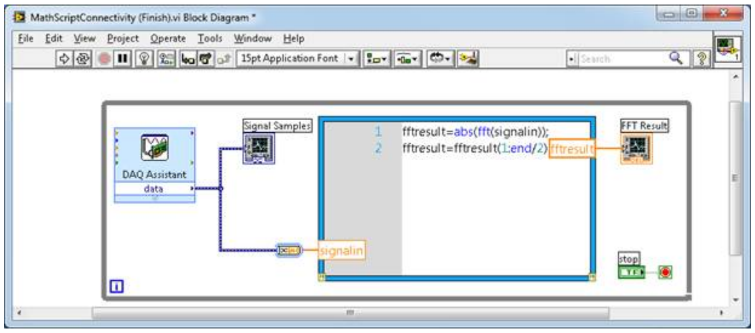

MathScriptConnectivity (Finish).vi
File Edit View Project Operate Tools Window Help
Signal Samples
Amplitude
Time
FFT Result
STOP

Figure 4. Inline analysis provides the results of your analysis as you are acquiring your data.

In this case, the application is performing a simple fast Fourier transform (FFT) measurement on an acquired sinusoid. If this were the vibration signal from rotating machinery, the source of the vibration signal could be determined based simply off of the integer order of the FFT peak. Performing the analysis as the data is acquired eliminates the need to move data between incompatible tools. Because the analysis IP already existed in an .m file, it is incorporated into LabVIEW with the MathScript Node. Examine Figure 5:


MathScriptConnectivity (Finish).vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
DAQ Assistant
data
Signal Samples
FFT Result
signalin
fftresult
stop

```
fftresult=abs(fft(signalin));
fftresult=fftresult(1:end/2);
```

Figure 5. Using MathScript, you can import your existing IP to perform inline analysis as you acquire the data.

By placing the MathScript Node on the block diagram and wiring your acquired data into it, the analysis occurs as the data is acquired, saving you precious time and resources.

LabVIEW Provides Native Hardware Connectivity

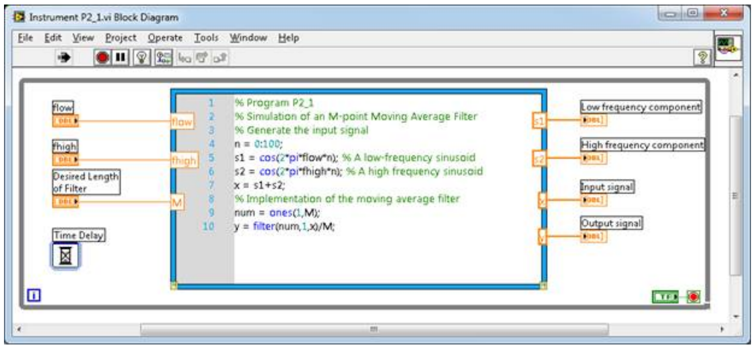
As an add-on for the LabVIEW development environment, the MathScript RT Module takes many of the benefits that the LabVIEW graphical development environment provides and extends them to .m file development. For more than 20 years, engineers and scientists have used LabVIEW to interface with measurement and control devices. LabVIEW integrates seamlessly with thousands of different hardware devices and helps save development time with convenient features and a consistent programming framework across all hardware. The MathScript RT Module extends this simplified hardware interface to you while developing your .m files.

LabVIEW Provides a Built-In Graphical User Interface for Your .m Files

A challenge that users of traditional .m file environments face is the development of graphical user interfaces (GUI). A GUI provides added interaction to algorithm development, giving you the ability to add a simple knob or slider to see how your algorithm responds to varying input variables.

LabVIEW contains a comprehensive collection of drag-and-drop controls and indicators so you can quickly and easily create user interfaces for your application and effectively visualize results without integrating third-party components or building views from scratch. The quick drag-and-drop approach does not come at the expense of flexibility. Power users can customize the built-in controls via the Control Editor and programmatically control UI elements to create highly customized user experiences.

Examine Figure 6:



```
% Program P2_1
% Simulation of an M-point Moving Average Filter
% Generate the input signal
n = 0:100;
s1 = cos(2*pi*flow*n); % A low-frequency sinusoid
s2 = cos(2*pi*fhigh*n); % A high frequency sinusoid
x = s1+s2;
% Implementation of the moving average filter
num = ones(1,M);
y = filter(num,1,x)/M;
```

Figure 6. This .m file performs a moving-average filter on two input sinusoids.

Adding a GUI to this program would provide the added benefit of data interaction. That is, you could easily explore how the algorithm responds to varied sinusoid frequencies or filter lengths. Consider the UI displayed in Figure 7:

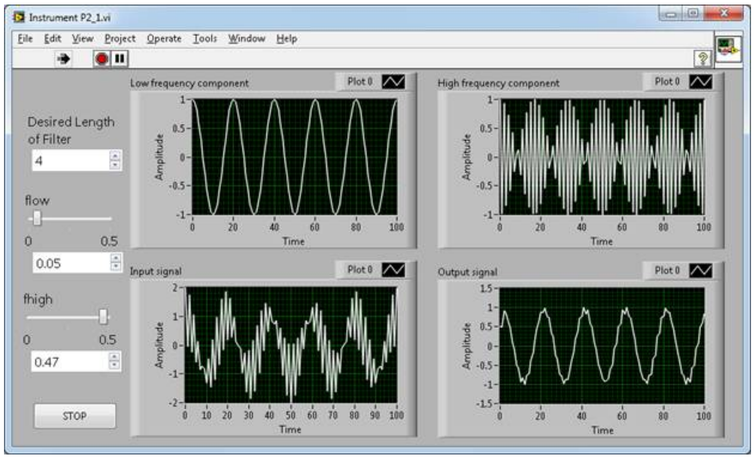


Figure 7. Adding a GUI to your IP adds invaluable data interaction and simplifies development.

With this GUI (which took only a matter of seconds to create), it is much easier to explore the merits of the moving-average filter algorithm. You can simply *slide* the low and high frequency sliders to see the result change on the lower-right graph.

Deploy Your Custom .m Files to Embedded Hardware

The LabVIEW MathScript RT Module delivers the ability to deploy .m files directly to real-time hardware.

Take a second to completely digest that.

The LabVIEW MathScript RT Module delivers the ability to deploy .m files directly to real-time hardware. No code rewrites. No translating to ANSI C. None of that. That is a big deal. This is important because right now there is no other direct methodology for doing this.

Many scientists and engineers developing mathematical algorithms do so in one of several .m file environments. A primary challenge of these highly abstract .m file languages is that they lack some key characteristics necessary for deployment to embedded hardware. These languages are loosely typed, which means that the data type of a variable can change at run time without explicit casting. Although this can be valuable in a desktop environment where memory is abundant, dynamically changing a variable's data type during an operation introduces jitter, which could violate the application's timing constraints in a real-time scenario. The lack of explicit resource management functions and timing constructs further complicates the deployment to embedded hardware.

Read this white paper to learn how the LabVIEW MathScript RT Module solves these problems and provides a direct path to embedded hardware for user's .m files, even if they were developed outside of MathScript. Developers can incorporate their .m files into a LabVIEW VI and then deploy that to embedded hardware like any other LabVIEW VI. The steps in this process are simplified compared to other environments and involve LabVIEW, the Real-Time Module, and of course, the MathScript RT Module.

How Do I Use the MathScript RT Module?

There are two methodologies for using MathScript. The first is the MathScript Interactive Window. This window, accessed from the Tools menu, provides an intuitive interface to MathScript. With a command-line interface and a window to build batch files, the MathScript Interactive Window is designed to help you develop your scripts.

See more detailed information on using the MathScript Interactive Window.

The second methodology is using MathScript inline with graphical LabVIEW code. The MathScript Node is a structure on the LabVIEW block diagram that gives you the ability to put text-based MathScript code inline with G. You can define inputs and outputs on the node borders to pass data back and forth between the two paradigms. The node even supports debugging with single steps, breakpoints, syntax highlighting, and a probe for intermittent values.

See more detailed information on using the MathScript Node.

The typical workflow for developing your own script from scratch is to use the MathScript Interactive Window for the development, and then, to run the script among G code using the MathScript Node.

Using the MathScript RT Module Combines the Benefits of Graphical and Textual Programming Into One Environment

LabVIEW MathScript RT is an add-on module for the LabVIEW Full and Professional Development Systems. This module is designed to natively add text-based signal processing, analysis, and math into the graphical development environment of LabVIEW. With more than 800 built-in functions, LabVIEW MathScript gives you the ability to either run your existing custom .m files or create them from scratch. Using this native solution for text-based math, you can combine graphical and textual programming within LabVIEW because the text-based engine is part of the LabVIEW environment. With LabVIEW MathScript RT, you can choose whether graphical or textual programming is most appropriate for each aspect of your application.