SIMULATION PACKAGE FOR D1IGITAL CONTROL EDUCATION

Jenő Kovács[1], Imre Benyó[1] and György Lipovszki[2]

University of Oulu
POB 4e300, Linnanmaa, 90014 Oulun yliopisto, Finland
Jeno.Kovacs@oulu.fi, Benyo@rit.bme.hu

Budapest University of Technology and Economics
Goldmann Gy. tér 3, V2 ép. 1111 Budapest, Hungary
Lipovszki@rit.bme.hu

Abstract: The paper introduces a simulation package facilitating the education of discrete-time control theory. The package consists of three main parts: sampling phenomenon-aliasing-filtering, control design in polynomial (input-output) form and control design in state-space form. Beyond classroom demonstration, the simulator can be an effective tool to intensify self-study and distance education. It was created in the LabVIEW programming environment, which alloys the simplicity of graphical programming with the traceability of the analogue devices. Several examples are shown to demonstrate the features of the new tool. *Copyright © 2005 IFAC*

Keywords: simulation, control, mathematical models, computer-aided, education.

1. INTRODUCTION

The quality of education may be increased – besides the organisational possibilities – by introducing new results from research activities, industrial developments and theoretical achievements or by improving the quality of teaching. However, in basic studies, like in discrete-time control design, the first option is not feasible due to the completeness of theories and algorithms. The effectiveness can be rather improved by increasing the quality of teaching introducing new tools and teaching methods.

During the last decade, the development of educational and industrial software and simulation tools has been accelerated. Industrial applications focus on the replacement of expensive equipment by software tools (virtual equipment) and parallel, the new technologies, *e.g.* fieldbuses, are strongly supported by high-tech software solutions. Numerous flexible software solutions for industrial human machine interface (HMI) and supervisory control and data acquisition (SCADA) have appeared in the market. The university education increasingly integrates such industry-standard programming-environment tools mainly in laboratory processes but more and more frequently also in research and education. In classrooms, the demonstration is the most common utilisation. However, it is only the first level of education-purpose utilisation. At the second level, a simulation package works as an interactive tool for intensifying home study. The third stage is the long-distance education, which cannot be effective without such virtual instrument.

Considering systems and control engineering education, demonstration involves process modelling and simulation, imitated data acquisition and process control. It requires high-level graphical user interface for efficient communication and visualisation.

One of the most widespread industrial software also used in education is the LabVIEW, a National Instrument product (National Instrument, 2003). The LabVIEW is a block-oriented graphical programming environment developed at first place for data acquisition and monitoring, but process control and modelling are also fully supported. Due to the additional toolboxes – *e.g.* the Control engineering toolbox, TUBSIM (Lipovszki and Aradi, 1995) –, the application area is continuously expanding. Continuous-time control engineering education at the Budapest University of Technology and Economics utilising the TUBSIM has been earlier reported in Aradi, 1996.

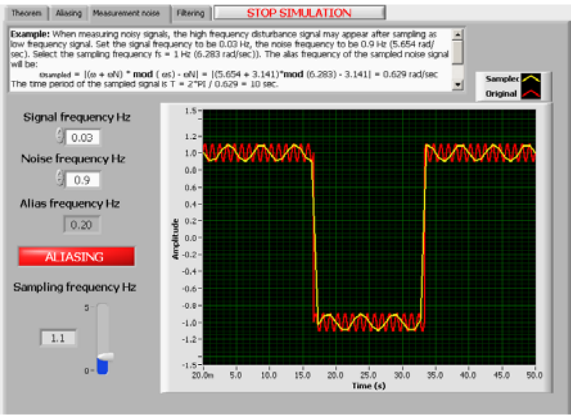
Simulation-demonstration tools for discrete-time control engineering are presented in this paper. The tools are developed for the "Discrete-time control theory" course at the University of Oulu.
The paper first introduces the simulation package and then several examples are given to demonstrate the effectiveness of the new tools.

2. CONTROL SIMULATION PACKAGE

The training purpose simulation package facilitates three topics: a) the sampling phenomenon-aliasing-filtering, b) the control design in polynomial (input-output) form and c) the control design in state-space form. The simulators completely satisfy the necessary demonstration and exercise requirements of the course "Discrete-time control theory".

2.1 Sampling, aliasing and filtering

Understanding the sampling phenomenon is essential for the above-mentioned course. The first simulator attempts to cover the main features of sampling periodic signals. It has four tables: *Theorem*, *Aliasing*, *Measurement noise* and *Filtering*; see Figure 1.



Example: When measuring noisy signals, the high frequency disturbance signal may appear after sampling as low frequency signal. Set the signal frequency to be 0.03 Hz, the noise frequency to be 0.9 Hz (5.654 rad/sec). Select the sampling frequency fs = 1 Hz (6.283 rad/sec)). The alias frequency of the sampled noise signal will be:
ωsampled = |(ω + ωN) * **mod** (ωs) - ωN| = |(5.654 + 3.141)***mod** (6.283) - 3.141| = 0.629 rad/sec
The time period of the sampled signal is T = 2*PI / 0.629 = 10 sec.

Signal frequency Hz 0.03

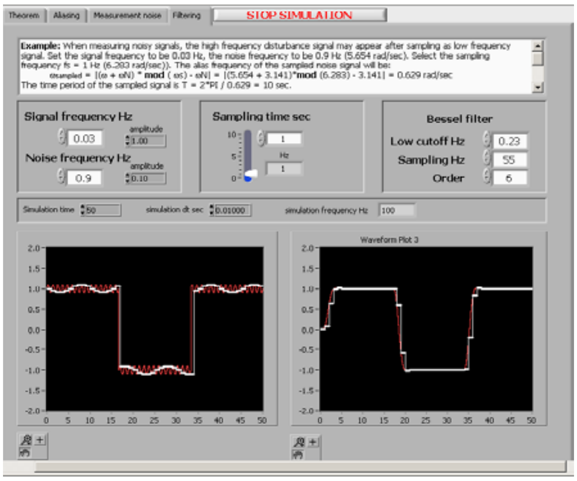
Noise frequency Hz 0.9

Alias frequency Hz 0.20

Sampling frequency Hz 1.1

Fig.1. Simulator for sampling phenomena: table *Measurement noise* demonstrating aliasing.

In the first *"Theorem"* table, the sampling theorem is explained in text format. The second *"Aliasing"* table offers to select the correct sampling time for periodic signal. At the same table, the aliasing phenomenon is also demonstrated by sampling periodic signals. The user may change the sampling frequency and observe the aliasing on the display or by comparing the frequencies of the original and sampled signals. The third table, *Measurement noise,* demonstrates the practical problem of aliasing: a high frequency (0.9 Hz) noise may appear as low frequency (0.03 Hz) signal when applying wrong sampling frequency (1.1 Hz); see Figure 1. The fourth table, *Filtering*, provides alternative solutions to avoid aliasing: a) choosing high sampling frequency (if the high frequency signal is not a noise but rather it contains the information carried on a low frequency signal) and b) filtering the noise signal before sampling, see Figure 2.



Example: When measuring noisy signals, the high frequency disturbance signal may appear after sampling as low frequency signal. Set the signal frequency to be 0.03 Hz, the noise frequency to be 0.9 Hz (5.654 rad/sec). Select the sampling frequency fs = 1 Hz (6.283 rad/sec)). The alias frequency of the sampled noise signal will be:
ωsampled = |(ω + ωN) * **mod** (ωs) - ωN| = |(5.654 + 3.141)***mod** (6.283) - 3.141| = 0.629 rad/sec
The time period of the sampled signal is T = 2*PI / 0.629 = 10 sec.

Signal frequency Hz 0.03 amplitude 1.00

Noise frequency Hz 0.9 amplitude 0.10

Sampling time sec 1 Hz 1

Bessel filter: Low cutoff Hz 0.23, Sampling Hz 55, Order 6

Simulation time 50, simulation dt sec 0.01000, simulation frequency Hz 100

Fig.2. Avoiding aliasing: high frequency sampling (on the left) or filtering the measured signal (on the right).

2.2 RST control design

The other element of the simulator package is the RST simulator. First, the applied theoretical background is summarised. The discrete-time controllers in input-output approach can be described by the general two-degree-of-freedom control structure, presented in Figure 3.

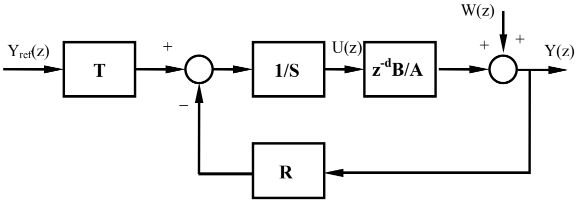


Fig.3.General two-degree-of-freedom control system.

The process to be controlled is here defined by the pulse transfer function $z^{-d}\frac{B}{A}$. The controller is constructed from the three control polynomials, R, S and T; therefore it is called the RST control structure

(Landau, 1990). The structure is very attractive, since most of the discrete-time controllers can be described in or easily transformed into the RST structure. The control design aims to define the R, S, and T polynomials to achieve certain required performance for regulation and tracking, defined respectively by the pulse transfer functions

$$H_D\left(z^{-1}\right)=\frac{Y(z)}{W(z)}=\frac{A\left(z^{-1}\right)S\left(z^{-1}\right)}{A\left(z^{-1}\right)S\left(z^{-1}\right)+z^{-d}B\left(z^{-1}\right)R\left(z^{-1}\right)} \qquad (1)$$

$$H_{CL}\left(z^{-1}\right)=\frac{Y(z)}{Y_{ref}(z)}=\frac{B\left(z^{-1}\right)T\left(z^{-1}\right)}{A\left(z^{-1}\right)S\left(z^{-1}\right)+z^{-d}B\left(z^{-1}\right)R\left(z^{-1}\right)} \qquad (2)$$

The pole- and pole-zero placement control design algorithms are here applied. The S and R polynomials are obtained by solving the following Diophantine equation:

$$A\left(z^{-1}\right)\cdot S\left(z^{-1}\right)+z^{-d}B\left(z^{-1}\right)\cdot R\left(z^{-1}\right)=P_D\left(z^{-1}\right), \qquad (3)$$

where $P_D\left(z^{-1}\right)$ is the desired closed-loop characteristic polynomial. Integral action is ensured if the polynomial S has a factor ($1-z^{-1}$). Zero cancellation, which has a role in tracking, can be achieved by applying P_DB^+ on the right hand side; B^+ denotes the zeros to be cancelled as a factor of B, $B = B^+\cdot B^-$. Further tracking requirement can be satisfied by the proper choice of polynomial T; *e.g.* choosing $T(1) = P_D(1) / B^-(1)$, guarantees zero-steady state error in response to reference signal. It is further possible to make the best of the two-degree-of-freedom feature of the RST structure allowing the user to define different tracking dynamics by introducing an external reference model, B_m/A_m, (see also in Figure .4) in series to the polynomial T, which is in this case

$$T\left(z^{-1}\right)=\frac{P_D\left(z^{-1}\right)}{B^-(1)}, \qquad (4)$$

thus resulting:

$$\frac{Y(z)}{Y_{ref}(z)}=z^{-d}\frac{B_m\left(z^{-1}\right)B^-\left(z^{-1}\right)}{A_m\left(z^{-1}\right)B^-(1)}. \qquad (5)$$

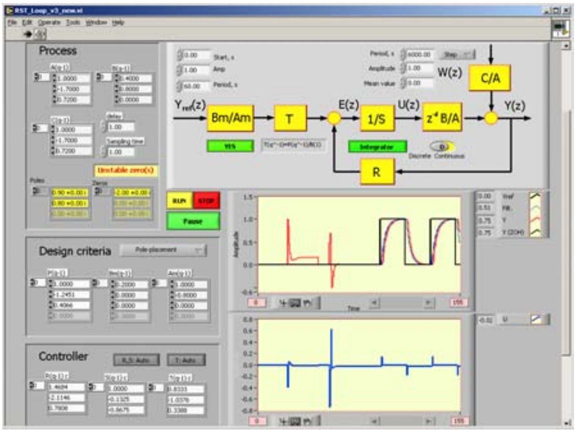
Pole-placement with implicit reference model is another alternative in (Åström and Wittenmark, 1997). In that case, the R, S and T polynomials are directly designed to satisfy the requirement

$$H_{CL}\left(z^{-1}\right)=\frac{Y(z)}{Y_{ref}(z)}=z^{-d}\frac{B_m\left(z^{-1}\right)B^-\left(z^{-1}\right)}{A_m\left(z^{-1}\right)B^-(1)}. \qquad (6)$$

2.3. RST simulator

The RST simulator was developed in LabVIEW graphical programming environment. Utilising special continuous- and discrete-time blocks which are available from the TUBSIM toolbox (Lipovszki and Aradi, 1995), while others are described in (Benyó *et al.* 2003).

The user interface of the simulator, shown in Figure .4, has three main areas: the *RST structure* in the upper right-hand corner, the *graphical windows* below it, and the *parameter windows* on the left-hand side.



A(q-1): 1.0000, -1.7000, 0.7200
B(q-1): 0.4000, -0.8000, 0.0000
C(q-1): 1.0000, -1.7000, 0.7200
delay: 1.00
Sampling time: 1.00
Poles: 0.90 +0.00i, 0.80 +0.00i
Zeros: 2.00 +0.00i
Start, s: 0.00
Amp: 1.00
Period, s: 60.00
Period, s: 6000.00
Amplitude: 1.00
Mean value: 0.00
P(q-1): 1.0000, -1.2451, 0.4066
Bm(q-1): 0.2000, 0.0000, 0.0000
Am(q-1): 1.0000, -0.8000, 0.0000
R(q-1): 1.4604, -2.1146, 0.7808
S(q-1): 1.0000, -0.1325, -0.8675
T(q-1): 0.8333, -1.0376, 0.3388

Fig.4. The user interface of the RST simulator.

The *RST structure* illustrates the control structure and provides several option for:

- choosing between continuous- and discrete-time process model,
- setting the parameters of the reference signal, Y_{ref}, (amplitude, period, start time) and the disturbance signal, W, (period and amplitude of a deterministic stepwise signal, amplitude and mean value of a stochastic noise),
- applying (or not) integral action,
- choosing the type of T (polynomial or scalar),
- utilising (or not) a reference model, B_m/A_m.

The *graphical windows* plot the reference signal, the output signal and the sampled output (in case of continuous-time process) in the upper window, and the control input is shown in the lower one.

In the uppermost *parameter window,* the user can define the process (discrete- or continuous-time process with output disturbance model). The zeros and poles of the process are automatically calculated and warning is given when any of those is unstable. It has an importance when considering zero cancellation.

The control design criteria, the P_D polynomial and the B_m/A_m reference model (if required) are set in the window below. When reference model is used, the T polynomial has a fixed polynomial form; otherwise the user may select between a constant and a polynomial form for T in the *RST structure* window. The user may further choose from the list of offered ready-made controllers:

- pole placement controller,
- pole-zero placement controller,

- dead-beat controller, and
- minimum-variance controller.

Notice, that for dead-beat and minimum variance controllers, the simulator automatically selects the P_D as $P_D(z^{-1}) = 1$ and $P_D(z^{-1}) = C(z^{-1})$, respectively.

The control polynomials, R, S and T, are then automatically calculated and displayed at the bottom window. The user can accept those control polynomials but one can also apply other control polynomials using the manual mode for R, S and T. By this option, the main feature of the RST control structure – namely, that all the discrete-time controller can be described or converted into the RST form. For example, the simple PI controller can be interpreted as $S = 1 - z^{-1}$, and $T = R = r_o + r_1 z^{-1}$. This way, any other control design can be tested in this simulator, even open-loop controller, by selecting R = 0. It is especially advantageous, when continuous-time process is chosen, since it requires manual setting of the control polynomials.

The simulator has a user-friend interface, displaying only the necessary elements; *e.g.* the polynomials of the reference model appear only when the user selects to apply one; or any modification in process parameters or design criteria is immediately updates the R, S and T polynomials. The simulation can be paused at any time for changing signal parameters.

2.3.1 Demonstration example

The RST simulator offers numerous possibilities for demonstrating and designing control structures. During an introductory course to control theory, simple example helps to understand the performance degradation caused by a stepwise output disturbance signal and how to eliminate it by applying an integrator in the forward loop. The difference between the regulation and tracking trajectories can be easily visualised by selecting the proper controller parameters in a fast and simple manner. Later, advanced students may design their own controllers and easily test those using the simulator.

A simple pole-placement design has been already shown in Figure .4. The design steps of the pole-placement controller can be summarised the following way: a) design of the output disturbance elimination based on the required closed-loop characteristic polynomial, b) introducing additional integration action to avoid non-zero steady-state error in response to output disturbance, c) ensuring zero-steady state error in reference signal following and finally d) selecting a tracking dynamics to emphasise the two-degree-of-freedom feature of the RST control structure.

Let the process to be controlled, using h=1 sec sampling time:

$$A(z^{-1}) = 1 - 1.7z^{-1} + 0.72z^{-2}, \tag{7}$$

$$B(z^{-1}) = 0.4 + 0.8z^{-1},\ d = 1. \tag{8}$$

The desired regulation dynamics is

$$P_D\,(z^{-1}) = 1 - 1.2451z^{-1} + 0.4066z^{-2}. \tag{9}$$

First, the simulation is run without integral action, therefore non-zero static error remains when a step output disturbance occurs. Applying the integration, the error can be eliminated. As the last step, the response to change in reference signal is tested with an explicit reference model. The responses in these three steps can be shown in the same graphical window, as seen in Figure .4, to support the full understanding.

It must be mentioned that the determination/ calculation of the desired characteristic polynomial, P_D, is not the concern of the simulator, the user must have the skill to determine it alone; it is one of the tasks in the “Discrete-time control theory” course.

An important and often neglected topic in discrete-time control design is the *intersampling ripple* phenomenon. It is a specific discrete-time control phenomenon. Deriving the controller based on the discrete-time process model may lead to such a performance where the controlled output is stable at the sampling instants but oscillating between. Although this phenomenon has a remarkable importance in real-time application, it can be easily overlooked during control design and the control education. Avoiding this mistake, the RST simulator allows to define the process in continuous-time and to apply discrete-time controller.

Consider the following double integrator process to be controlled, $G(s) = 1/s^2$. The control task is to achieve an open-loop behaviour defined by

$$H_{OL}(s) = \frac{0.2}{s(1+s)}. \tag{10}$$

The discrete-time transfer functions assuming ZOH and sampling-time h = 1 sec are

$$G(z^{-1}) = 0.5\frac{z^{-1} + z^{-2}}{(1 - z^{-1})^2} \tag{11}$$

$$H_{OL}(z^{-1}) = 0.074\frac{z^{-1} + 0.718z^{-2}}{(1 - z^{-1})(1 - 0.368z^{-1})}. \tag{12}$$

The simplest open-loop controller is

$$H(z^{-1}) = \frac{H_{OL}(z^{-1})}{G(z^{-1})} = 0.148\frac{(1 + 0.718z^{-1})(1 - z^{-1})}{(1 + z^{-1})(1 - 0.368z^{-1})}. \tag{13}$$

In RST structure it can be applied as:

$$T\left(z^{-1}\right)=0.148\left(1+0.718z^{-1}\right)\left(1-z^{-1}\right) \tag{14}$$

$$S\left(z^{-1}\right)=\left(1+z^{-1}\right)\left(1-0.368z^{-1}\right) \tag{15}$$

$$R\left(z^{-1}\right)=0\,. \tag{16}$$

The control inputs response to an impulse reference signal is

$$U(z)=H\left(z^{-1}\right)Y_{ref}(z)=0.148\frac{\left(1+0.718z^{-1}\right)\left(1-z^{-1}\right)}{\left(1+z^{-1}\right)\left(1-0.368z^{-1}\right)} \tag{17}$$

The critically stable pole of the control input causes the oscillation in the control signal as therefore in the controlled output, as shown in Figure 5.

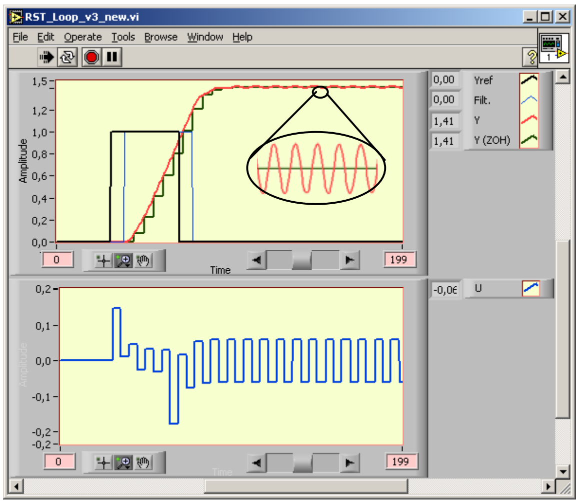


Fig.5. Intersampling ripple phenomenon. The upper graphic window also shows how the continuous-time signal oscillates while the sampled signal is constant.

2.4. State-space design

State-feedback control is the other main area covered by the "Discrete-time control design" course. It is necessary to introduce the state-space approach because although it has less weight in the current course, it gives the basis for the next, advanced control course where the feature of this approach is exploited.

The structure of the state-space simulator is similar to the RST simulator and thus it is easy to get familiar with; see Figure 6.

A simple state-feedback control is here designed to follow the setpoint changes – which maybe filtered by the reference model B_m/A_m – by a given closed-loop characteristics. The new features of state-space approach, namely the process observability and controllability – are emphasized by displaying the *Process analysis window*. The *Control design window"* requires the desired closed-loop pools and the reference model if selected. The feedback matrix K, the achieved closed-loop state matrix and the achieved closed-loop poles are also displayed. Its importance rises when uncontrollable states exist highlighting immediately the inefficiency of the controller.

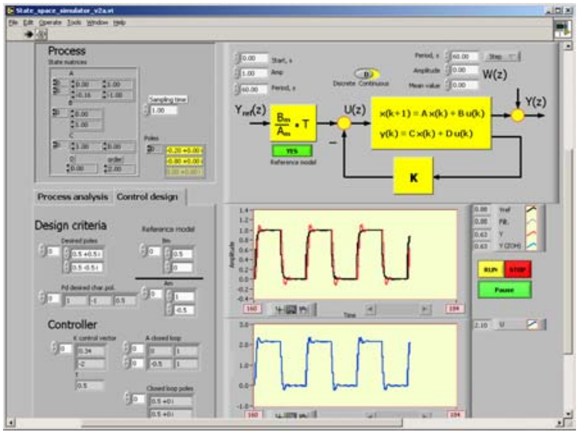


Fig.6. Demonstrating state-space design.

CONCLUSION

Computer-aided education may be an advantageous solution for increasing the efficiency of control education. The classroom education, similarly to laboratory exercises, may be further visualised by introducing target-oriented simulation/demonstration environment. Such a tool has been developed and introduced in the current study.

A package of advanced simulation tools is designed to serve particular courses on digital control theory and advanced control engineering. The developed simulation elements provide an easy-to-use and easy-to-understand manner to deepen the enhanced knowledge of the course.

The presented tools support the simulation of sampling, discrete-time modelling, discrete-time control in input-output approach and control design in state-space. The programming environment was chosen to be the LabVIEW due to its attractiveness.

The RST simulator facilitates the discrete-time control education, especially supporting the polynomial input-output approach for system description. The basic structure is the two-degree-of-freedom formulated by the control polynomials, R, S and T.

Further advantage of the LabVIEW based simulator is that it can be transformed into a self-executable file and run it in any Windows based operation system. It requires only a runtime engine, which is freely available from the web site of the National Instrument.

The proposed simulator has been evaluated in practice in a course “Digital control theory” at the University of Oulu and will be used in co-operation with the Budapest University of Technology and Economics.

The simulator has been developed in LabVIEW version 7.0. Since the application of the simulator in the education, the version 7.1 has been released; this already contains several features for control design and different control algorithms which were developed by the authors. The authors are however not yet completely convinced to apply these new features in the current application.

REFERENCES

Aradi, P. (1996). Using LabVIEW in Education of Systems and Control Engineering. In: Proceedings of NIWeek’96, Austin, TX, USA.

Åström, K.J. and Wittenmark, B. (1997). *Computer-Controlled Systems*. Prentice-Hall, Upper Saddle River USA.

Benyó, I.; Lipovszki, Gy. and Kovács, J. (2003). Advanced Control Simulation Tools In LabVIEW Environment, In: *Proc. 6th IFAC Symposium on Advances in Control Education*, Oulu, Finland, 275-279.

Ikonen, E. and Najim, K. (2001). *Advanced Process Identification And Control*, Dekker.

Landau, I.D. (1990). *System Identification and Control Design*. Prentice-Hall, Englewood Cliffs New Jersey.

Lipovszki, Gy. And Aradi, P. (1995). General Purpose Block Oriented Simulation System Using LabVIEW, In: Proceedings of NIWeek’95, Austin, TX, USA.

National Instruments, (2003). *LabVIEW User Manual* at http://www.ni.com/pdf/manuals/