

Digital Control-A Setting for a Mixture of Computing Strategies: Numerics, Symbolics and Simulation

Raymond G. Jacquot, Jerry C. Hamann and John E. McInroy
College of Engineering, University of Wyoming

Abstract

The authors report on a course that is a blend of classical and modern digital control, and digital filtering. Exercises in this course reflect a variety of computing resources and strategies and as a result low order system problems are no longer assigned. The authors have incorporated symbolic computing in several appropriate places where it can increase understanding and yield error-free algebraic manipulations. In most exercises, the computing is of the numerical type using algorithms developed in the course. Simulation, employing a graphical-oriented simulation language, lends reality to implementation of control laws and estimation algorithms.

Introduction

At the University of Wyoming a lower level graduate course is offered to introduce graduate students to discrete-time control and digital filtering. The assumed prerequisite material is that contained in first courses in control and discrete signals and systems. The students are typically from the controls, power systems and signal processing areas with the presence of an occasional mechanical or chemical engineer. The topics are a blend of classical discrete-data control topics, modern digital control topics and digital filter synthesis techniques. A topical outline is given in Table 1.

Table 1. Course Outline

- Review of z-transforms
- The pulse transfer function
- Single-loop controller design
- PID and lead-lag controllers
- Ragazzini controller design
- Digital filter synthesis
- Discrete-time state variable representation
- State feedback and associated algorithms
- Prediction, current and reduced order observers
- Nonzero setpoints versus regulators
- L-Q optimal control
- Reciprocal root locus

The work reported here represents an update of that reported previously based on a second class offering¹. In the past, essentially the identical course material has been covered, but due to computational limitations the problem assignments have been rather simple, usually of the second-order variety. With the convenience offered by MATLAB for Windows² and VisSim³, the authors have made the assigned problems more realistic and can concentrate class time on algorithms for the design process rather than with detailed classroom calculations. The Symbolic Toolbox of MATLAB allows some rather tedious, error-prone algebraic manipulations to be accomplished without error. The authors and others have reported previously the use of symbolic computation in a digital filter design course⁴. VisSim is a simulation environment that handles mixed continuous- and discrete-time elements with a graphical user interface. The learning curve for VisSim is essentially negligible, and students progress rapidly in their ability to simulate complex systems. Simulink⁵ is an alternative to VisSim and runs seamlessly with MATLAB.

The third order “theme problem” employed for most of the exercises is one that is interesting from both classical and a state space design points-of-view and one not amenable to hand calculations, thus motivating the students to develop an interest in the use of modern software tools.

Theme Problem Plant

The plant chosen for the student exercises is the thermal environmental chamber illustrated in Fig. 1. The students are given a handout that explains the physical situation and explores the small perturbation analysis for the system. The outer chamber is heated by a flow of steam that is controlled by a valve the dynamics of which may not be neglected. Fig. 1 is labeled with the perturbed temperature variables ($x_1(t)$, $x_2(t)$, $w(t)$) and the perturbed valve gate motion variable ($x_3(t)$). The current used to produce that perturbation is denoted as $u(t)$. The perturbation in the environmental temperature is denoted as $w(t)$.

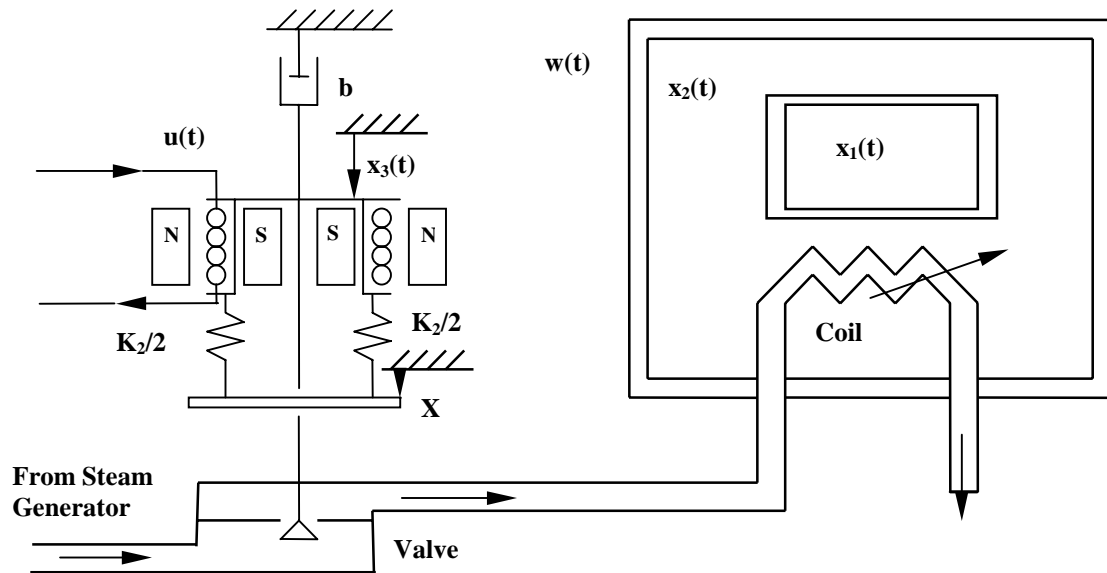


Figure 1. Thermal Chamber System to be Controlled

The governing differential equations for the temperature perturbations are

$$\begin{aligned}\frac{dx_1}{dt} &= -x_1 + x_2 \\ \frac{dx_2}{dt} &= x_1 - 2x_2 + 2x_3 + w(t) \\ \frac{dx_3}{dt} &= -4x_3 + u(t)\end{aligned}\quad (1)$$

If the temperature of the inner chamber $x_1(t)$ is taken to be the system output, the transfer function for the open-loop plant is

$$G(s) = \frac{X_1(s)}{U(s)} = \frac{2}{s^3 + 7s^2 + 13s + 4} \quad (2)$$

From relations (1) the continuous-time state description matrices are

$$F = \begin{bmatrix} -1 & 1 & 0 \\ 1 & -2 & 2 \\ 0 & 0 & -4 \end{bmatrix}, \quad g = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (3)$$

and

$$c^T = [1 \quad 0 \quad 0] \quad (4)$$

It is in this setting that the problem exercises are constructed. Of course, the problems could be of higher order, but in the authors' opinions this system is of sufficient complexity to be instructive while illustrating the power of contemporary computing tools.

The Pulse Transfer Function

After a brief review of the z -transform it is appropriate to examine how a discrete-time representation of the plant and the associated data converters can be obtained. The A/D and D/A converters are assumed to operate synchronously at a rate specified by the sampling interval T and to have infinitely precise representations of the digital data. The so-called pulse transfer function is then given by

$$G(z) = \frac{z-1}{z} Z \left\{ L^{-1} \left[\frac{G(s)}{s} \right] \right\}_{t=kT} \quad (5)$$

where $Z\{\cdot\}$ represents the taking of the z -transform and $L^{-1}[\cdot]$ represents the taking of the inverse Laplace transform. MATLAB is particularly useful utilizing the '*residue*' function for evaluation of the partial fraction coefficients for $G(s)/s$. The '*deconv*' function is particularly attractive for the reassembly of the z -domain partial fractions into a single polynomial ratio for $G(z)$. The pulse transfer function for this plant for a sampling interval of $T = 0.2$ s is

$$G(z) = \frac{0.001899(z + 2.6629)(z + 0.1865)}{(z - 0.9625)(z - 0.5924)(z - 0.4493)} \quad (6)$$

Cascade Controller Design

This content focuses on the synthesis of discrete-time controllers with transfer function $D(z)$ which will reshape either the root locus diagram or the open-loop frequency response to yield performance which is improved. Here the polynomial and complex number handling facilities of MATLAB are invaluable particularly in the synthesis of finite settling time and ripple-free controllers. Easy calculation and graphical display are highly desirable for these trial-and-error

designs. VisSim's ability to handle mixed discrete-time and continuous-time system simulation make it attractive for design evaluation and for understanding of control law implementation.

Digital Filter Synthesis

In this course there is in-depth discussion of several approaches to digital filter synthesis. The reason for examination of this topic is that often controllers are synthesized in the s-domain and are to then be implemented in discrete-time form. The two methods addressed are the bilinear transform method (Tustin's method) and the matched-pole-zero technique.

The authors have previously discussed the use of symbolic computing for the bilinear-transform synthesis of digital filters⁴. The approach taken is that the designer starts with a unit frequency lowpass prototype of Butterworth, Chebyshev, inverse Chebyshev or elliptic type. The first step is to perform a lowpass, bandpass, bandstop or highpass transformation to yield an s-domain filter of the correct type.

For bilinear transform synthesis, the next step is to prewarp the poles of this filter so they will have the correct critical frequencies after the bilinear transform to the z-domain has been performed. Depending upon the extent to which one desires to involve the student in the process, the MATLAB Symbolic Toolbox can be used to advantage for both the prewarping and bilinear transform processes by eliminating some error-prone algebra. This still demands a substantial degree of understanding of the process on the part of the student.

A second very reliable technique for the synthesis process is that of pole-zero matching wherein the poles and zeros of the s-domain filter are exponentially mapped to the z-domain and then the filter gain is matched at one critical frequency. MATLAB is useful for the pole-zero mapping and for the gain matching process. Students are asked to address a comparative design problem utilizing both processes.

State Variable Representation

The first step in the discrete-time state variable design process is to find a discrete-time state variable representation of the plant and the associated data converters. The plant and data converters can be represented by the matrix-vector difference equation

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{b}u(k) \quad (7)$$

The process of evaluation of the \mathbf{A} and \mathbf{b} matrices can be carried out either analytically or numerically. The analytical process involves the inversion of Laplace transforms for which the '*residue*' function is invaluable. The numeric process involves the calculation of the matrix exponential, for which MATLAB is particularly well-suited. The discrete-time state variable matrices computed for a sampling interval of $T = 0.2$ s are

$$\mathbf{A} = \begin{bmatrix} 0.8341 & 0.1494 & 0.0254 \\ 0.1494 & 0.6847 & 0.2227 \\ 0 & 0 & 0.4493 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 0.0019 \\ 0.0273 \\ 0.1377 \end{bmatrix} \quad (8)$$

At this point it is also advantageous to explore system controllability and observability employing the '*rank*' function in MATLAB.

State and Estimated State Feedback

This design process begins with the consideration of pole placement using full state feedback. This involves the expansion of a symbolic determinant and equating it to the desired closed-loop characteristic equation $\alpha_c(z)$

$$\alpha_c(z) = \det[zI - A + bf^T] \quad (9)$$

This algebraic determinant expansion involves the symbol z and the symbolic elements of the vector f^T and is a process that is quite prone to human error. This is an ideal application of the MATLAB Symbolic Toolbox that is also incorporated into the Student Edition of MATLAB⁶. A numerical way around the aforementioned determinant expansion is to employ the algorithm of Ackermann^{6,8}. The algorithm for the state feedback gains for a single-input-single-output system is

$$f^T = e_1^T [A^{n-1}b \ A^{n-2}b \ \dots \ Ab \ b]^{-1} [A^n + d_{n-1}A^{n-1} + \dots + d_1A + d_0I] \quad (10)$$

where the d_i are the desired closed loop characteristic equation coefficients and e_1^T is the first unit row vector.

In the absence of a complete set of state measurements, the appropriate control law is to feed back estimates of the states obtained from an asymptotic observer either of the prediction or current type. The observer design process is identical to state feedback design involving the symbolic expansion of a determinant. The separation principle indicates the design of the control law and the estimator may be accomplished independently and the two designs do not interact. Simulation allows students to explore the effect of closed-loop pole locations on transient responses, system rise time, settling time and bandwidth. In the case of estimated state feedback, VisSim is useful for simulation of the closed-loop structure and allows the students to investigate the convergence of the state estimates to the actual states when the plant and estimator are started with different initial conditions.

L-Q Optimal Control

The final portion of the course addresses the computation of state feedback gains by minimization of the quadratic performance index

$$J = \frac{1}{2} \left\{ x(N)^T H x(N) + \sum_{k=0}^{N-1} x(k)^T Q x(k) + u(k)^T R u(k) \right\} \quad (11)$$

subject to the constraints imposed by the matrix-vector difference equation (eqn.(7)) governing the plant. This minimization problem results in the discrete-time matrix Riccati equation which must be solved backwards in time to yield the time varying feedback gain matrix. If the planning horizon is sufficiently long (large N), then the feedback gains approach constant values and these are what are often implemented in engineering practice and are used in simulations using VisSim.

In the single-input/single-output case, the reciprocal root locus is used as a tool to aid in the selection of the Q/R ratio to give particular pole locations, and MATLAB is a tool that makes the plotting of such loci a reasonable task.

Conclusion

This paper reports a change of teaching strategy in a digital control course reflecting state-of-the-art computing tools including symbolic computing to aid the design process where appropriate. Students view these developments positively, are generally confident that they understand the design concepts involved, and well aware that the computing is a means to an end and not the end in itself.

The authors have developed many MATLAB m-files for accomplishing the more complicated computing tasks and these and the problem exercises are available from the authors on request.

References

1. Jacquot, R.G. and J.C. Hamann, "Teaching Digital Control and Filtering Using MATLAB and VisSim," *Proc. Frontiers in Education: 26th Annual Conference*, Salt Lake City, UT, Nov. 6-9, 1996, paper 7c1.
2. The Math Works, Inc., *MATLAB Reference Guide*, Natick, MA, 1992.
3. Visual Solutions, Inc., *VisSim User's Guide--Version 1.2*, Westford, MA, 1993.
4. Jacquot, R.G., J.C. Hamann, J.W. Pierre and R.F. Kubichek, "Teaching Digital Filtering Using MATLAB and Derive," *Proc. of the 1995 ASEE Annual Conference*, Anaheim, CA, June 25-28, 1995, pp. 618-621.
5. The Math Works, Inc., *SIMULINK User's Guide*, Natick, MA, 1992.
6. The Math Works, Inc., *The Student Edition of MATLAB Version 5 User's Guide*, Prentice-Hall, Englewood Cliffs, NJ, 1997.
7. Ackermann, Jurgen, "Der Entwurf Linearer Regelungssysteme im Zustandsraum," (not translated) *Regelungstechnik und Prozessdatenverarbeitung*, Vol. 7, pp. 297-300, 1972.
8. Jacquot, R.G., *Modern Digital Control Systems:2nd Ed.*, Marcel Dekker, Inc., New York, NY, 1995.

RAYMOND G. JACQUOT

Ray Jacquot received his BSME and MSME degrees at the University of Wyoming in 1960 and 1962 respectively. He was an NSF Science Faculty Fellow at Purdue University where he received his Ph.D. in 1969. He is a member of ASEE, IEEE and ASME and has been active in the Computers in Education Division of ASEE and has educational interests in the control of dynamic systems and is currently Professor of Electrical Engineering.

JERRY C. HAMANN

Jerry Hamann received the BS degree in Electrical Engineering/Bioengineering from the University of Wyoming in 1984. After employment with Hewlett-Packard he returned to the University of Wyoming as a NSF Graduate Fellow completing the MSEE degree in 1988. He completed the Ph.D. at the University of Wisconsin in 1993 with a focus in robust control. His interests are in systems, real-time signal processing and web-based data acquisition. He is currently Assistant Professor of Electrical Engineering.

JOHN E. McINROY

John McInroy received his BSEE degree from the University of Wyoming in 1986. He received the MS and Ph.D. degrees in Electrical Engineering from Rensselaer Polytechnic Institute in 1988 and 1991 respectively with a focus in robotics. His interests are in robotic systems, sensor data fusion and precision motion control. He is currently Associate Professor of Electrical Engineering.