sional parameter indicating the degree of usefulness of a fundamental quantity in the design and analysis of many engineering disciplines, namely, the power. This Note demonstrates that a focus on the system control power efficiency does not curtail the designer's ability in monitoring other important quantities of the overall design; on the contrary, it brings in an added, but necessary, dimension to the SCS that is a time-tested proven concept in engineering design. The improvement of efficiency, in the least, simply makes better use of available control power since it results in reduced power interaction with the unmodeled dynamics. Furthermore, this reduction is not merely qualitative but it is quantified via efficiency. More importantly, all efficiency computations only involve the reduced-order control design model while extracting information about the behavior of the truth-model of the system.

References

[1]Öz, H., Farag, K., and Venkayya, V. B., "Efficiency of Structure-Control Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 12, No. 3, 1990, pp. 545–554.

[2]Öz, H., "Efficiency Modes Analysis of Structure-Control Systems," AIAA Paper 90-1210, April 1990; also *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 5, 1994, pp. 1028–1036.

[3]Öz, H., and Khot, N. S., "Structure-Control System Optimization with Fundamental Efficiency Constraint," *Proceedings of the Eighth Virginia Polytechnic Inst. and State Univ. Symposium on Dynamics and Control of Large Structure*, edited by L. Meriovitch, 1991, Blacksburg, VA, pp. 159–169.

[4]Öz, H., and Khot, N. S., "Optimization for Efficient Structure-Control Systems," *Proceedings of the Fifth NASA/DoD Conference on Control Structure* (Lake Tahoe, NV), Part I, 1992, pp. 425–454.

Polynomial Interpolation Between Input Samples for Continuous-Time Simulation

Sandeep Gupta*
ViGYAN, Inc., Hampton, Virginia 23666

Introduction

EFFICIENT techniques for computing the continuous-time response of linear time-invariant (LTI) systems using polynomial interpolation between input samples are presented in this Note. Usually the response of an LTI system is computed by discretizing the continuous-time equations with a zero-order hold approximation and performing discrete-time simulation with uniformly spaced input samples. This approach gives fairly accurate results when the time step T is small; in particular, for $T < \pi/|\lambda_{\max}|$, where $\lambda_{\max}$ is the eigenvalue of A with largest absolute value.[1] Linear interpolation between control samples can be achieved by appending integrators at each input channel of the given system, discretizing this combined system with zero-order hold, and applying the derivative of the input to this system,[2] though this approach increases the order of the system. This Note develops techniques to evaluate the response of an LTI system in state-space form by interpolating between the input samples with polynomials of arbitrary order. Further, efficient algorithms are presented for computation of certain matrix integrals required for implementation of these techniques. A numerical example is presented to demonstrate the benefits of these techniques.

Received Nov. 17, 1993; revision received March 15, 1994; accepted for publication March 20, 1994.

*Research Engineer; working under contract with the Spacecraft Controls Branch, Guidance and Control Division, NASA Langley Research Center, Hampton, VA 23681-0001. Member AIAA.

Response with Polynomial Interpolation

Consider a linear time-invariant system $\dot{x} = Ax + Bu$, $y = Cx + Du$, where A, B, C, D is a realization of the system. Assume that the input is specified at uniformly spaced time steps T, that is, $u[kT]$ for $k = 0, 1, 2, \ldots$ is specified. From the state of the system, $x[kT]$, at the instant kT, the state at the next instant, $x[(k+1)T]$, is obtained from

$$x[(k+1)T] = e^{AT}x[kT] + \int_0^T e^{A(T-\tau)}Bu(\tau + kT)\,d\tau \qquad (1)$$

and the output becomes $y[kT] = Cx[kT] + Du[kT]$. The expressions for evaluating the state response with polynomial approximations for $u(t)$ in the intersample interval, $kT \le t \le (k+1)T$, are developed in this section.

With zero-order polynomial approximation, the input is constant between two samples, that is, $u(t) = a_0[kT]$ for $kT \le t \le (k+1)T$. The constant coefficients $a_0[kT]$ for $k = 0, 1, 2, \ldots$ are determined by the interpolation scheme used, usually $a_0[kT] = u[kT]$. Defining

$$H_0(t) = \int_0^t e^{A(t-\tau)}\,d\tau$$

the state at instant $(k+1)T$ for zero-order hold becomes

$$x[(k+1)T] = e^{AT}x[kT] + H_0(T)Ba_0[kT] \qquad (2)$$

These equations correspond to the well-known zero-order hold discretization of continuous-time system equations.

First-order polynomial approximation of the intersample behavior of the input is expressed as $u(t) = a_0[kT] + a_1[kT](t - kT)$ for $kT \le t \le (k+1)T$. For linear interpolation between $u[kT]$ and $u[(k+1)T]$, $a_0[kT] = u[kT]$ and $a_1[kT] = \{u[(k+1)T] - u[kT]\}/T$. Defining

$$H_1(t) = \int_0^t \tau e^{A(t-\tau)}\,d\tau$$

the evolution of the state vector is expressed as

$$x[(k+1)T] = e^{AT}x[kT] + H_0(T)Ba_0[kT] + H_1(T)Ba_1[kT] \qquad (3)$$

Approximation of the intersample behavior of the input by nth-order polynomials ($n \ge 2$) can be expressed as

$$u(t) = \sum_{i=0}^{n} a_i[kT](t - kT)^i, \qquad kT \le t \le (k+1)T \qquad (4)$$

The coefficients $a_i[kT]$, $i = 0, 1, \ldots, n$, are selected to prescribe the desired intersample behavior between input samples. Using Eq. (1), the state $x[(k+1)T]$ becomes

$$x[(k+1)T] = e^{AT}x(kT) + \sum_{i=0}^{n} H_i(T)Ba_i[kT] \qquad (5)$$

where

$$H_i(t) = \int_0^t \tau^i e^{A(t-\tau)}d\tau$$

With these formulas, polynomials of arbitrary order can be used to interpolate between the input samples for computation of the continuous-time response of linear time-invariant systems.

Evaluating the Matrix Integrals

Efficient evaluation of the matrix integrals $H_i(t)$ is the critical issue in implementation of this approach for computing the response of a linear time-invariant system. This section presents an efficient algorithm for robust computation of these matrix integrals, based on Padé approximations for matrix exponentials. For simplicity, the following discussion is limited to computation of $H_i(t)$ for $i = 0, 1, 2$.

A key result in the development is that with a matrix

$$S = \begin{bmatrix} A & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{6}$$

the matrix integrals $H_i(t), i = 0, 1, 2$ can be obtained from the matrix exponential

$$e^{St} = \begin{bmatrix} e^{At} & H_0(t) & H_1(t) & \frac{1}{2}H_2(t) \\ 0 & I & It & \frac{1}{2}It^2 \\ 0 & 0 & I & It \\ 0 & 0 & 0 & I \end{bmatrix} \tag{7}$$

This result can be derived using the approach in Ref. 3. Equating appropriate submatrices in the equations

$$\frac{\mathrm{d}}{\mathrm{d}t}[e^{St}] = Se^{St}, \qquad e^{St}|_{t=0} = I$$

leads to the equations

$$\dot{H}_i(t) = AH_i(t) + It^i, \qquad H_i(0) = 0 \tag{8}$$

for $i = 0, 1, 2$. Solving these equations shows that $H_i(t), i = 0, 1, 2$, are of the desired form.

Next, efficient computation of the matrix exponential e^{St} using Padé approximations[4] is presented and the sparse structure of S is exploited to compute $H_i(t), i = 0, 1, 2$. Padé approximations for the matrix exponential e^{St} are given by $D^{-1}N$, where

$$N = \sum_{k=0}^{q} p_k(St)^k, \qquad D = \sum_{k=0}^{q} p_k(-St)^k$$

with

$$p_k = \frac{(2q-k)!q!}{(2q)!k!(q-k)!}$$

It follows that (for $q > 2$), N and D are given by

$$N = \begin{bmatrix} N_e & N_0 & N_1 & N_2 \\ 0 & I & \frac{1}{2}It & p_2It^2 \\ 0 & 0 & I & \frac{1}{2}It \\ 0 & 0 & 0 & I \end{bmatrix}$$

$$D = \begin{bmatrix} D_e & D_0 & D_1 & D_2 \\ 0 & I & -\frac{1}{2}It & p_2It^2 \\ 0 & 0 & I & -\frac{1}{2}It \\ 0 & 0 & 0 & I \end{bmatrix} \tag{9}$$

where $p_2 = (q-1)/4(2q-1)$ and

$$N_e = \sum_{k=0}^{q} p_k A^k t^k, \qquad D_e = \sum_{k=0}^{q} p_k A^k (-t)^k$$

$$N_i = \sum_{k=i+1}^{q} p_k A^{k-(i+1)} t^k \tag{10}$$

$$D_i = \sum_{k=i+1}^{q} p_k A^{k-(i+1)} (-t)^k, \qquad i = 0, 1, 2$$

Using Eq. (7) with this Padé approximation for e^{St}, the matrix integrals, $H_i(t), i = 0, 1, 2$, are approximated by

$$\begin{aligned} e^{At} &= D_e^{-1}N_e \\ H_0(t) &= D_e^{-1}(N_0 - D_0) \\ H_1(t) &= D_e^{-1}(N_1 - D_1 - D_0t) \\ H_2(t) &= 2D_e^{-1}\left(N_2 - D_2 - D_1t - \tfrac{1}{2}D_0t^2\right) \end{aligned} \tag{11}$$

It was shown in Ref. 5 that the computation of the matrix exponential e^{St} by the Padé approximation method has good numerical accuracy when $\|St\|_\infty < \frac{1}{2}$. If this condition is not satisfied, the matrix St is scaled by a factor 2^j for some integer j such that $\|St/2^j\|_\infty < \frac{1}{2}$. Using the algorithm described above with this scaled matrix, $e^{At/2^j}$ and $H_i(t/2^j), i = 0, 1, 2$, can be computed accurately. Then, e^{At} is computed by repeated squaring, and $H_i(t), i = 0, 1, 2$, are obtained by recursively using the following "doubling formulas"[6]:

$$\begin{aligned} H_0(2t) &= e^{At}H_0(t) + H_0(t) \\ H_1(2t) &= e^{At}H_1(t) + tH_0(t) + H_1(t) \\ H_2(2t) &= e^{At}H_2(t) + t^2H_0(t) + 2tH_1(t) + H_2(t) \end{aligned} \tag{12}$$

These formulas result from manipulations of the integrals involved.

The discussion above is for computing $H_i(t)$, for $i = 0, 1, 2$, which are required for simulations with second-order polynomial approximation of the intersample behavior of the input. For the approximation with polynomials of an arbitrary order, a straightforward extension of this technique may be used to compute the required matrix integrals.

Numerical Example

Numerical simulations of the phase II control-structure interaction (CSI) evolutionary model (CEM) structure,[7] a testbed at NASA Langley for experiments in control of flexible structures, were performed with polynomial interpolation between the input samples. The simulation model had two force inputs (in pound-force) and two angular deflection outputs (in arc-seconds), with 50 states corresponding to flexible modes of the structure with natural frequencies up to 13.2 Hz. The input for these simulations is a 20-Hz bandwidth disturbance, obtained by the superposition of random sinusoids. The "true" system response is computed using Runge-Kutta integration of the continuous-time state equations and cubic-spline interpolations (which required 90.481 Mflops). Simulations were performed with four time step sizes, namely, 0.020, 0.015, 0.010, and 0.005 s, using zero-, first-, and second-order polynomial interpolation between the input samples. Computational error is determined as the difference between the true solution and the simulation results. Figure 1 illustrates the significant reduction in computational error in the first output for first-order simulation (solid line) as opposed to zero-order simulations (dashed line) for a time step of 0.020 s. For a comparison of the accuracy of various simulations, the computational error is quantified as $\sqrt{\sum_k T(e_k^T e_k)}$, where e_k is the error at time step k. The flop count (in megaflops) provides a measure of the computational effort involved in these simulations. Table 1 summarizes the simulation results. Reduction in computational error is observed with increasing order of polynomial interpolation between input samples for all time steps, as

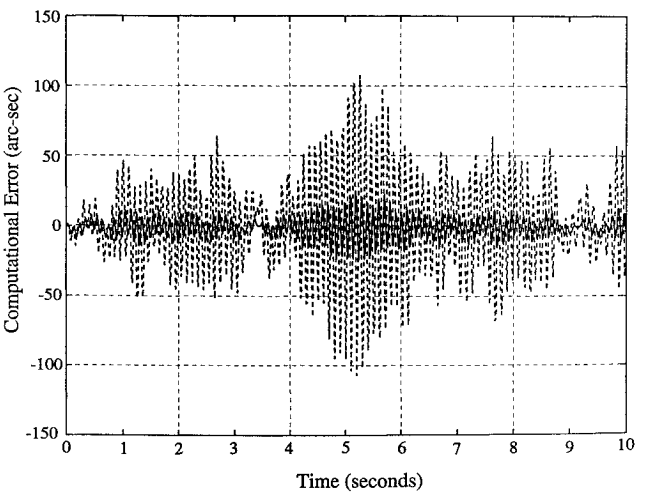

Fig. 1 Comparison of computation errors for simulation with zero-order (dashed) and first-order (solid) interpolations between input samples at 0.020 s time step.

Table 1 Comparison of accuracy and computational effort for simulation with different time steps and orders of polynomials interpolation

Time step	Zero-order simulation		First-order simulation		Second-order simulation	
T, s	Error	Mflops	Error	Mflops	Error	Mflops
0.020	116.445	2.735	25.643	2.872	15.669	3.013
0.015	88.972	3.641	15.721	3.820	8.743	4.005
0.010	60.151	5.465	8.140	5.729	4.823	6.001
0.005	30.352	10.925	3.027	11.443	2.337	11.977

expected. Note that first-order simulations result in better accuracy than zero-order simulations at smaller time steps. Although the reduction in computational error from zero-order simulations to first-order simulations is significant, the improvement in accuracy from first-order simulations to second-order simulations is smaller. Moreover, a comparison of the flop counts shows that there is a small computational overhead in using higher order simulations. However, the computational effort for a higher order simulation is less than that for simulation with the same order of polynomial interpolation with a smaller time step.

This study shows that significant increase in solution accuracy can be achieved by using first- or second-order interpolations between the input samples as opposed to zero-order hold discretizations. In general, the benefits achieved by polynomial interpolation depend on the time step size, the dynamic model being simulated, and the frequency content of the input. Further, the advantages seem to taper off for higher order polynomial interpolations; therefore, it is advisable to use smaller time steps rather than higher order polynomial interpolations beyond second-order interpolation for better solution accuracy.

Conclusions

This Note presents a computationally efficient approach for simulation of linear time-invariant systems, with polynomial interpolation between input samples. Explicit expressions have been developed for such simulations, and efficient techniques for computation of the matrix integrals required in these expressions are presented. Numerical simulations were presented for comparing accuracy and computational effort involved with various time step sizes and for different orders of polynomial interpolation. It is concluded that polynomial interpolation between input samples leads to significant improvement in solution accuracy over the usual zero-order hold discretization for LTI simulations.

Acknowledgment

This research was supported by the Spacecraft Controls Branch, Guidance and Control Division, NASA Langley Research Center, Hampton, VA 23681, under contract NAS1-19341.

References

[1]*MatrixX Users Manual*, Integrated Systems, Santa Clara, CA, 1990.

[2]Grace, A., Laub, A. J., Little, J. N., and Thompson, C. M., *Control System Toolbox, for Use with Matlab*, The MathWorks, Natick, MA, 1992.

[3]Van Loan, C. F., "Computing Integrals Involving the Matrix Exponential," *IEEE Transactions on Automatic Control*, Vol. AC-23, No. 3, 1978, pp. 395–404.

[4]Golub, G. H., and Van Loan, C. F., *Matrix Computations*, Johns Hopkins Univ. Press, Baltimore, MD, 1989.

[5]Moler, C., and Van Loan, C. F., "Nineteen Dubious Ways to Compute the Exponential of a Matrix," *SIAM Review*, Vol. 20, No. 4, 1978, pp. 801–836.

[6]Armstrong, E. S., and Caglayan, A. K., "An Algorithm for the Weighting Matrices in the Sampled-Data Optimal Regulator Problem," NASA TN D-8372, 1976.

[7]Horta, L. G., Reaves, M., Elliott, K. B., Belvin, W. K., Sulla, J. L., Gregory, J., Ugoletti, R., and Sandridge, C. A., "Langley's CSI Evolutionary Model: Phase II," NASA TM 109059, 1993.

Attainable Moments for the Constrained Control Allocation Problem

Wayne C. Durham*
Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061

Introduction

MODERN tactical aircraft are being designed with many more than the classical three sets of control effectors (ailerons, elevator, and rudder). The next generation of highly maneuverable airplanes are projected to have as many as 20 primary flight control effectors. These controls will all be constrained to certain limits, determined by the physical geometry of the control actuators or in some cases by aerodynamic considerations. The effective allocation, or blending, of these controls to achieve specific objectives is the control allocation problem.

The geometry of the constrained control allocation problem was developed in Ref. 1. In Ref. 2 we described a means of determining the subset of attainable moments, yielding a description of the boundary that contained the necessary information for the determination of controls in the allocation problem. The method presented in Ref. 2, although offering the advantage of generality, was admittedly complicated and difficult to implement. For the control allocation problems of particular interest, such generality is not required, and a simpler method of determining the attainable moment subset is available. That method is the subject of this Note.

The problem statement and nomenclature used in this note may be found in Refs. 1 and 2.

Method

Subset of Constrained Controls

First, consider the subset of constrained controls, Ω. Its m-dimensional bounding surface, $\partial(\Omega)$, is made up of rectangular two-dimensional surfaces, each of which corresponds to a given pair of controls varying within their constraints, whereas all other controls are at one or the other of their constraining values. Since there are $m-2$ other such controls, and each of these other controls may assume one of two values, there are thus 2^{m-2} surfaces, or facets, corresponding to a given pair of controls. These 2^{m-2} surfaces are all parallel in the m-dimensional control space.

For example, if there are three controls ($m=3$), say simple aileron, rudder, and horizontal tail, the constrained control subset is a simple box in three-dimensional space. There are two ($2^{m-2}=2$) facets of the box for each combination of the three controls taken two at a time.

If we use the horizontal tail differentially, then with the left and right horizontal tails considered as separate controls, there are four controls ($m=4$). In this case the subset of constrained controls is a four-dimensional hypercube that is bounded by eight three-dimensional cubes made up of 24 facets. With some imagination, one may sketch two three-dimensional boxes next to each other and connect all the corresponding vertices. This is the two-dimensional projection of the four-dimensional hypercube, in which one may visualize parallel facets occurring in groups of four.

This example could be continued by adding more controls, but our ability to visualize these higher dimensional spaces becomes strained. Suffice to say that, in considering the facets generated by considering the controls in pairs, there are 2^{m-2} such facets, all of which are parallel. Collectively, all of the facets in Ω comprise $\partial(\Omega)$.

Received July 8, 1993; revision received Dec. 14, 1993; accepted for publication March 7, 1994.

*Assistant Professor, Department of Aerospace and Ocean Engineering. Member AIAA.