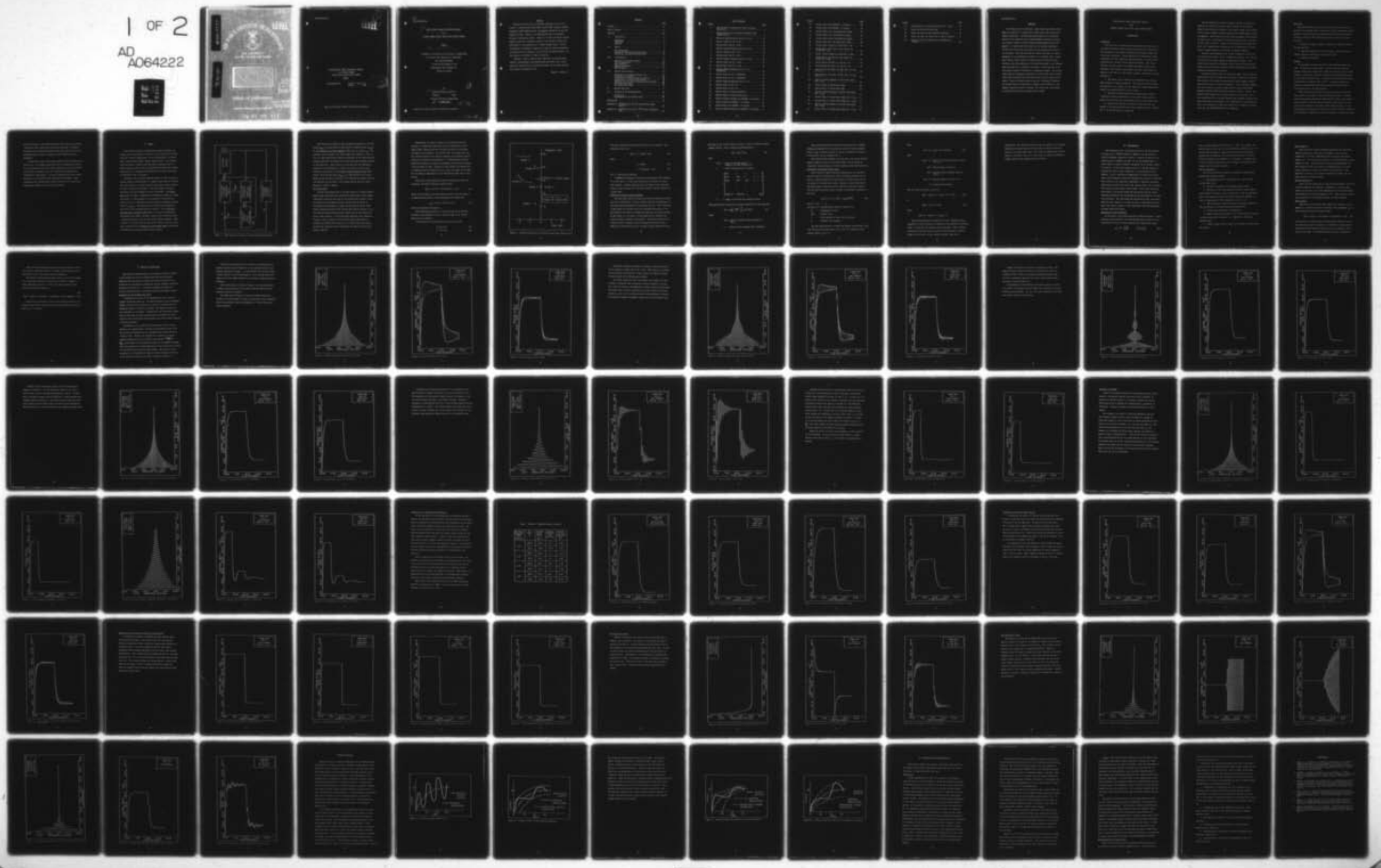
AD-A064 222 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 1/3
APPLICATION OF MODEL ALGORITHMIC CONTROL TO A LIGHTLY DAMPED SI--ETC(U)
DEC 78 H J COLSON
UNCLASSIFIED AFIT/GE/EE/78-21 NL

1 OF 2
AD A064222



AFIT/GE/EE/78-21

LEVEL II

APPLICATION OF MODEL ALGORITHMIC CONTROL TO A LIGHTLY DAMPED SINGLE INPUT SINGLE OUTPUT SYSTEM

THESIS

AFIT/GE/EE/78-21 Howard J. Colson, Jr.
Captain USAF

Approved for public release; distribution unlimited

(14) AFIT/GE/EE/78-21

(6) APPLICATION OF MODEL ALGORITHMIC CONTROL

TO A

LIGHTLY DAMPED SINGLE INPUT SINGLE OUTPUT SYSTEM

THESIS

(9) Master's thesis,

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air Training Command

in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

by

(10) Howard J. Colson, Jr., B.S.

Captain USAF

Graduate Electrical Engineering

(11) December 1978

(12) 111 p.

Approved for public release; distribution unlimited.

012225

Preface

When this thesis topic was initially proposed, it was to be a study of a complete Model Algorithmic Control (MAC) computer program, designated IDCOM (Identification and Command) applied to an aircraft dynamics problem. However, the IDCOM program was not available because of proprietary rights. Captain J. G. Reid, my thesis advisor, helped me develop a simplified version of the concept to study the performance of the algorithm for a lightly damped system. He was instrumental in guiding my research into this new control philosophy and provided me with insights I could not have attained unaided. For his efforts in my behalf, I wish to thank him.

Finally, I wish to thank my wife, Mary Ann, who besides being patient, understanding, and devoted during this AFIT tour, contributed her typing skills to produce both drafts and the final product. This thesis is dedicated to her.

Howard J. Colson, Jr.

Contents

List of Figures

Abstract

A new digital control technique, called Model Algorithmic Control (MAC), was applied to a single-input single-output model containing complex eigenvalues. The MAC algorithm developed was a simplified version of the algorithm used by ADERSA/GERBIOS Corporation (France) in a complete computer program designated IDCOM (Identification and Command). A second-order model based on the dominant eigenvalues of the B-52E Flutter Mode was selected as the hypothetical system to be controlled. Data were obtained for various selections of control parameters in the implementing program; i.e., sampling times, control input limiter values, number of discrete impulse response storage locations, and time-constants for the first-order model to be followed. Once an optimum set of parameters was found, a study was made of the controlling program's "robustness" to eigenvalue changes. The second-order model was subsequently expanded to a third-order, then a fourth-order while maintaining the previously found "optimum" control parameters. Results obtained indicated that the "robustness" exhibited in the MAC concept as implemented in IDCOM is not attributed to the impulse response prediction technique, but rather must be attributed to the particular control algorithm used in IDCOM.

APPLICATION OF MODEL ALGORITHMIC CONTROL TO A LIGHTLY DAMPED SINGLE INPUT SINGLE OUTPUT SYSTEM

I Introduction

Background

Over the last two decades several approaches have been used in the design of digital, multivariable control systems. Essentially, these methods may be broadly categorized into Frequency Domain and State Space techniques. Besides differing in computational design techniques, the above approaches generally differ in terms of model representation and basic control design philosophy. (Ref 1:1-2) The basis for this thesis is the Model Algorithmic Control (MAC) technique, which differs significantly from each of the above two methods in that MAC is a time domain technique. MAC relies on three principles:

1. The system to be controlled is represented by an internal model composed of impulse responses. This model is used to predict the behavior of the system, with its inputs and outputs being determined by the observed behavior of the system.

2. The desired system response is characterized by means of a reference trajectory (generally, a first-order model), which is updated on-line using the actual system output at the initial point.

3. Controls are computed iteratively so that when they are applied to the internal model they produce outputs as close as possible to the desired reference trajectory.

The MAC technique has been successfully applied to design and operation of industrial processes (Ref 2), design of an aircraft engine control system, and design of an aircraft flight control system (Ref 3) using a complete computer program called IDCOM (Identification and Command). When research for this thesis was begun, there had been no flight control applications of the MAC concept; in fact, only Ref 1 and Ref 2 were available. However, as the final draft was being prepared, Ref 3 emerged with a description of the MAC concept, as implemented in IDCOM, applied to the Mirage III (French fighter).

The MAC concept appeared to be a good candidate for flight control applications because of its "robustness" property; i.e., its lack of sensitivity to system changes in eigenvalues and gains. This characteristic would be a desirable trait in aircraft control system design because the aircraft dynamics change dramatically with flight conditions (altitude and airspeed).

As indicated above, when this thesis was begun, there had been no applications of MAC to flight control problems; therefore, no data had been collected for systems characterized by lightly damped, complex eigenvalues (typical of aircraft dynamics). It was initially proposed that this thesis be a study of IDCOM applied to the B-52E Control Configured Vehicle (CCV) Flutter Mode Control Problem. However, the complete IDCOM program was not made available to the writer; so, a very simplified form of the MAC algorithm was developed to analyze the basic properties of the concept. As a consequence of this simplification, the conclusions of this study should not be construed as reflections upon the ADERSA/GERBIOS (Ref 2) algorithm.

Objectives

The prime motivation for this thesis was to provide a basis for possible future flight control applications of MAC (specifically, MAC applied to the B-52E Control Configured Vehicle Flutter Mode Control Problem); therefore, the following objectives were directed toward that goal:

1. Develop a computer program to implement a simplified form of the MAC algorithm.

2. Apply the algorithm to a reduced-order, complex eigenvalue (lightly damped), single-input single-output model.

3. Evaluate the results.

Approach

To accomplish the above objectives, the following approach was used: An appropriate reduced-order, complex eigenvalue (lightly damped), single-input single-output model was selected based on B-52E dynamics; specifically, the model simulated the dominant eigenvalues of the flutter mode (Ref 4:20). The model was then discretized using a deconvolution technique (Ref 5:72-79). A computer program was created to accomplish the deconvolution/discretization and subsequent application to the MAC algorithm.

After the computer program was created, parameters within MAC were varied in an attempt to arrive at "optimum" values. The parameters investigated included the sampling time, control input limiter values, number of discrete impulse response values stored, and trajectory time constants. Once these values were determined, a mismatch between the theoretical system model and the "actual" simulation was introduced to observe the system's robustness to eigenvalue change.

After the results of the second-order model MAC study were obtained, a third and then a fourth-order model were developed. Parameters determined in the second-order model study were used in both of the enlarged models in order to determine if the results were still applicable.

At this point, due to the results obtained with the higher-order models (i.e., an extremely high sensitivity to parameter variation), it was hypothesized that more "robust" performance could be obtained with further refinement to the very simplified MAC algorithm as implemented in this thesis. In fact, an approach which was actually much closer to the IDCOM control algorithm was attempted. These refinements were incorporated as major revisions to the initial MAC implementing program and are still being tested.

II Theory

This section describes the hypothetical model following controller used in this thesis. As such, it is the basis for this MAC study for a special application: that of MAC applied to a Single-Input, Single-Output (SISO), lightly damped system. It is assumed that the reader is familiar with the theory associated with the impulse response function and the superposition/convolution integral. Additionally, two techniques ("deconvolution" and model following), as described in Ref 4, are used.

Basically, the idea behind the MAC concept is that of using an impulse response representation of a system, along with the recorded past input history, to predict and apply the future inputs required to follow a specified trajectory. The block diagram in Figure 1 illustrates the basic elements of such a controller. The System Model block of Figure 1 represents the mathematical model of a physical system. In this diagram, it is realized by a discrete-time solution y(p) of a differential equation based on piecewise constant inputs (u_{future}). The solution of this differential equation is then used in two succeeding blocks as follows: In the Calculate Discrete Impulse Response (CDIR) block it is used in conjunction with applied step inputs, through a process called "deconvolution", to generate specific values of the system's impulse response. Additionally, the solution y(p) is used as the initial point of a first-order trajectory in the Calculate Future Desired Output (CFDO) block to produce the vector y_d (future desired outputs).

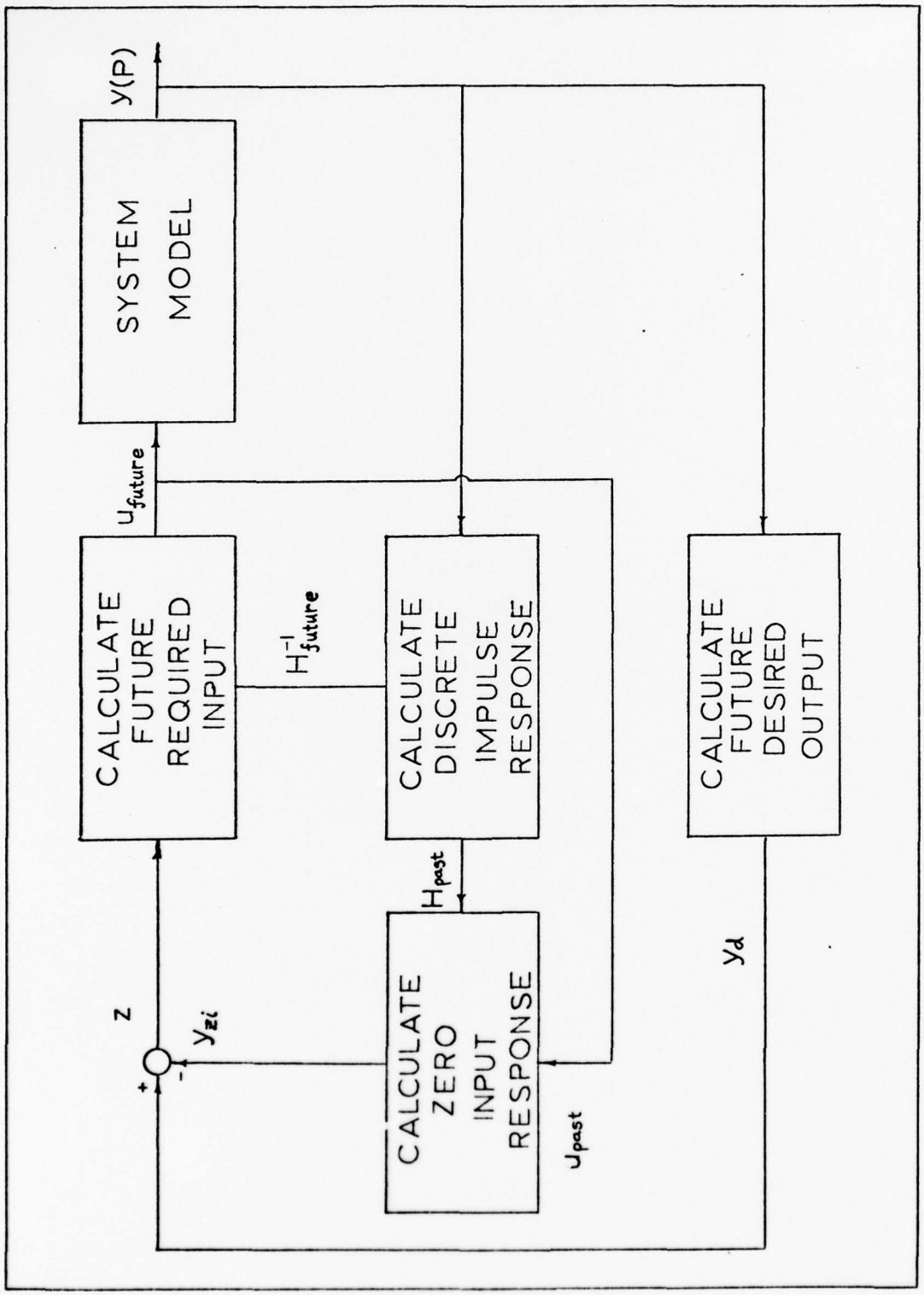


Figure 1. Block Diagram of Hypothetical Model Following Controller

The values of the discrete impulse response generated in the CDIR block (H_{past}) are used with the past history of applied inputs (u_{past}) in the Calculate Zero Input Response (CZIR) block to produce $\underline{y}_{zi}$, the system's future response if no future inputs were applied. A new vector $\underline{z}$ is then generated by taking the difference of the future desired outputs obtained in the CFDO block and the zero input response obtained in the CZIR block. This vector $\underline{z}$ is then used with the information generated in the CDIR block to produce the vector of future inputs required, as indicated in the Calculate Future Required Input (CFRI) block. The resulting value (u_{future}) is then applied to the system model and the process is repeated. The following sub-sections describe the mechanics of each block in the diagram and how they were implemented on a digital computer.

The System Model

To show the applicability of the MAC concept to a lightly damped SISO system, practically any second-order system with a small damping coefficient could have been used as the system model; however, since the impetus of this thesis initially evolved from the desire to investigate the possibility of using the MAC approach on the B-52E Control Configured Vehicle (CCV) Flutter Mode Control Problem, it was felt that the model selected should exhibit some of the characteristics of that aircraft. In line with this reasoning, this subsection contains a short synopsis of the B-52E CCV Flutter Mode Control Problem, the method used to reduce it to a SISO second-order model, and then the technique used to discretize the model and provide the model's solution.

Essentially, the primary purpose of a flutter mode control system is to lessen the aeroelastic flutter instabilities on the wings without adding structural stiffness and mass or restricting the speed of the airplane. For the B-52 CCV, the flutter mode control system consists of an outboard flaperon and an outboard aileron driven by two vertical accelerometers. To determine which elastic modes were sensitive to changes in airspeed, the root locus of Figure 2 was used (Ref 4:2-20). According to this figure, the flutter mode is identified as the fifth mode and, as such, was chosen as the basis for the dominant eigenvalues for the second-order model used in this thesis.

Since only a SISO hypothetical model following controller was developed, the model's defining equation became

$$\ddot{y}(t) + 0.5\ \dot{y}(t) + 237.2225\ y(t) = u(t) \tag{1}$$

where y(t) was taken to be some arbitrary output, u(t) was taken to be some arbitrary input, and the eigenvalues of interest were

$$\lambda_{1,2} = \sigma \pm j\omega = -0.25 \pm j\ 15.4 \tag{2}$$

which were obtained from Figure 2.

After developing equation (1), the next step was to develop a solution to the equation so that it could be used in the "System Model" block of Figure 1.

Since equation (1) could readily be put into the form

$$\dot{\underline{x}} = A\underline{x} + \underline{b}u \tag{3}$$

$$y = [1 \quad 0]\ \underline{x} \tag{4}$$

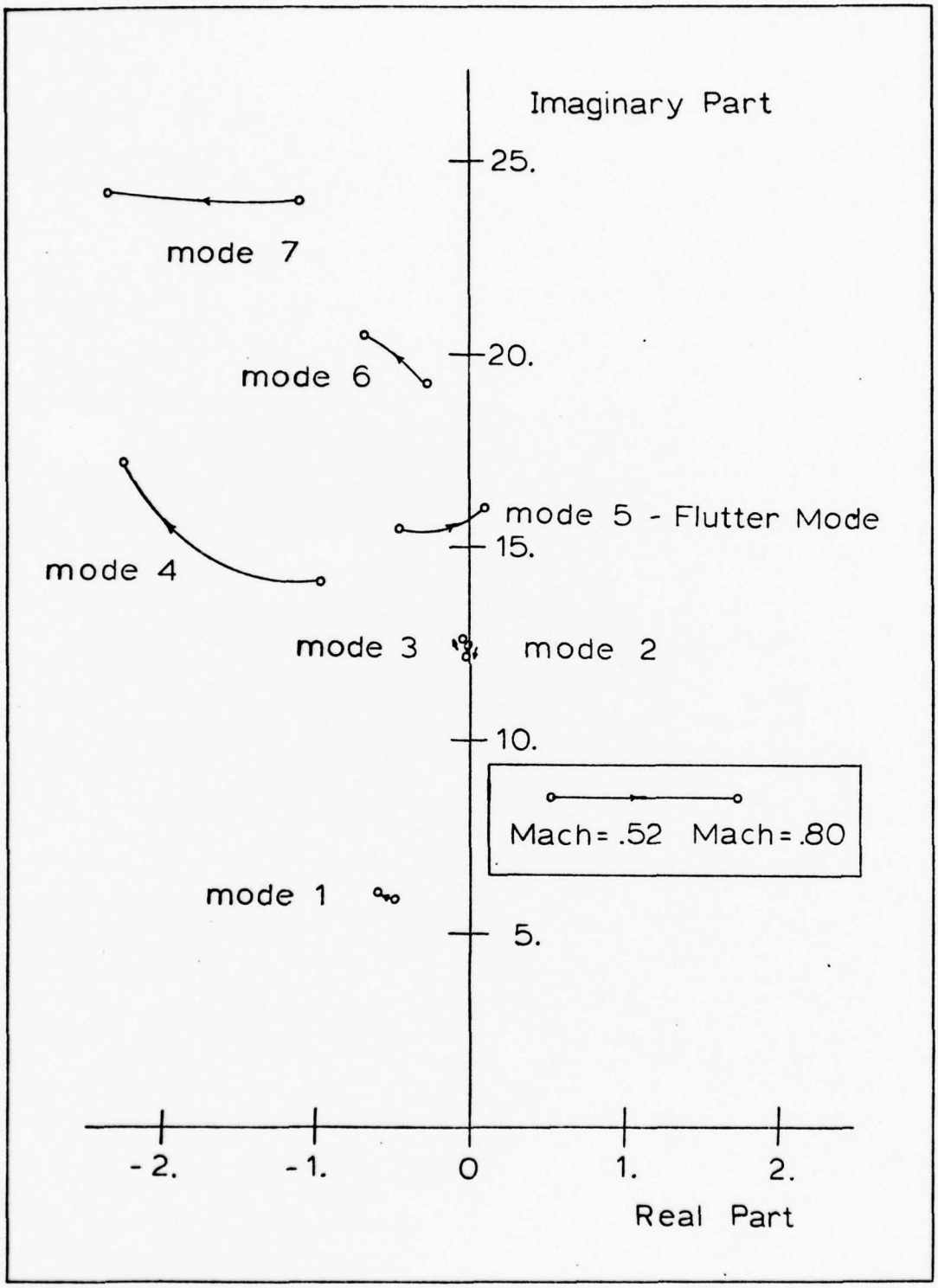


Figure 2. Airspeed Root Locus for B-52E CCV Flutter Mode Identification

the state transition matrix method was used for the solution. The solution has the form

$$\underline{x}(k + 1) = \Phi\ \underline{x}(k) + \Gamma u(k) \tag{5}$$

where

$$\Phi = \exp(A\Delta) \tag{6}$$

$$\Gamma = A^{-1}[\exp(A\Delta) - I]\ B \tag{7}$$

and Δ = discrete-time interval.

A FORTRAN subprogram was coded using this approach (See Appendix A) so that when Δ, σ, and ω were specified, the values for Φ and Γ were computed. Another subprogram was coded that in turn used these values to solve equation (5), and thus provided a specific output for a given input.

Forming the Discrete Impulse Response

The next major step in setting up the model following controller was that of arriving at a discrete impulse response representation of the system model described in the preceding section. Since ultimately a semi-adaptive controller could be an extended design goal of the MAC approach, the means for adding this capability later was provided by the choice, at the outset, of the method used to determine the discrete impulse response of the system. The method chosen was the "deconvolution" technique, which is developed/described in Ref. 4. Basically, this method just uses a system's output combined with the

knowledge of past system inputs to create a vector of discrete impulse response values. This is expressed mathematically as

$$\underline{h}(N) = \frac{1}{\Delta} U^{-1} y(N) \qquad (8)$$

where

$\mathbf{h}(N)$ = vector of discrete impulse response at N time intervals (Δ)

$\mathbf{U}$ = lower triangular matrix of inputs

$$= \begin{bmatrix} u(0) & 0 & 0 & 0 & \cdots & 0 \\ u(\Delta) & u(0) & 0 & 0 & \cdots & 0 \\ u(2\Delta) & u(\Delta) & u(0) & 0 & \cdots & 0 \\ \cdot & \cdot & \cdot & & & \\ \cdot & \cdot & \cdot & & & \\ \cdot & \cdot & \cdot & & & \\ u((N-1)\Delta) & u((N-2)\Delta) & \cdots & & & u(0) \end{bmatrix}$$

$\mathbf{N}$ = number of discrete time intervals stored

This matrix/vector problem may be solved recursively via the algorithm

$$h(n) = \frac{1}{u(0)} \left[\frac{y(n)}{\Delta} - \sum_{i=1}^{n-1} u(n-i)h(i)\right] \qquad (9)$$

where

$h(n)$ = value of discrete impulse response at time n.

n = specific time increment being considered

These relationships were coded and incorporated into a FORTRAN subprogram (Appendix A) which produced a vector of discrete impulse values, given a system's outputs and inputs, the number of values to be stored (N), and the sampling rate Δ.

This deconvolution technique was used only to set up the discrete impulse response values for a specified model, not to produce an "adaptive" impulse response for slowly changing system characteristics.

Calculating the Required Future Input

Once a discrete impulse response representation was available, the next step in constructing the model following controller was to select a suitable model for the controller to follow and then to set up a means of calculating the inputs required to follow that model. The method used in this thesis was to choose a first-order model so that the desired discrete output followed a first-order trajectory of the form

$$y_d(p + n) = y_s + [y(p) - y_s]\exp(\frac{-\Delta \cdot n}{\tau}) \qquad (10)$$

where $n = 1,2, \ldots Q$

and $y_d(p + n)$ = desired system output at time $(p + n)$.

y_s = designated set point

$y(p)$ = system output

τ = time constant of first order trajectory

p = present time increment

The next objective was to compute the present control input, $u(p)$, such that the next actual output value, $y(p + 1)$, equals the next desired output, $y_d(p + 1)$.

Since

$$y(p + 1) = y_{zi}(p + 1) + h(1)u(p) \qquad (11)$$

where

$y_{zi}(p + 1)$ = system's zero-input response at time $(p + 1)\Delta$

$u(p)$ = required input at time $p \cdot \Delta$

$h(1)$ = discrete impulse response value for time $1 \cdot \Delta$

$y(p + 1)$ = systems output at time $(p + 1)\Delta$

p = present time increment

then the control required is given by

$$u(p) = [y_d(p + 1) - y_{zi}(p + 1)] \; / \; h(1) \qquad (12)$$

or

$$u(p) = z(p + 1) \; / \; h(1) \qquad (13)$$

where

$$z(p + 1) = y_d(p + 1) - y_{zi}(p + 1)$$

These relationships were coded and put into a FORTRAN program (Appendix A), which combined the elements of the subsections discussed above, to arrive at the complete control algorithm. Other features incorporated into the control algorithm were provisions to specify changes in set points and the maximum allowable input value.

Additionally, the simulated system's output was sampled at $\Delta/5$ seconds, although the discrete impulse response and the control inputs were based on a Δ second sample time. The reason for sampling the system's output at the faster rate was to insure that the system's performance between control input applications was observed.

III Investigation

The investigation into the characteristics of the MAC approach was begun after a FORTRAN program to implement the algorithm was coded and debugged (Appendices A and B). A search was made for the maximum value of sampling time (DT = Δ) and the minimum number of stored values of impulse response information (N) to provide acceptable results. An arbitrary maximum value of N = 300 was assigned to simulate the finite storage capability of a hypothetical on-line computer. After a satisfactory combination of N and DT were determined, other parameters within the algorithm were varied to illustrate their effects on the algorithm's performance. Among the parameters varied were the first-order model time constant (TOW), the set points (YS), and the control limiter value (ULIM). After these variations were studied, a mis-match between the system model and the system simulation eigenvalues was introduced and the performance of the algorithm evaluated. Upon concluding the second-order study, the system model was next expanded to a third, and then a fourth-order model to see if the results obtained in the second-order case were applicable to higher order SISO systems.

Hypothetical Problem Statement

At the outset, the following specific problem was posed: A hypothetical system governed by the differential equation (12) was to be controlled such that the system achieved the following set points

$$y_s = YS = \begin{cases} 100 & 0 < t \leq 8 \\ 20 & 8 < t \leq 40 \end{cases} \qquad (14)$$

with a system response time constant of τ = TOW = 0.5 seconds. The discrete time interval used in the deconvolution technique was Δ = DT seconds and the number of values stored for the discrete impulse response was N; therefore, the effective time interval of impulse response information was CONTIME = N · DT seconds. A nominal value of 5 was initially selected for IQ, the number of future intervals predicted by the control algorithm.

Initial Assumptions

The following initial assumptions were made to establish the hypothetical problem statement.

1. The maximum value of N allowed was 300 - due to memory storage limitations.

2. No limit was posed for the computed control inputs.

3. Influence of the system dynamics would not extend significantly beyond 24 seconds, since at that value of time the expoential part of the impulse response time function would have decayed to a value of approximately .0025 (compared to 1.0 at t = 0).

4. The differential equation defining the impulse response and the one defining the system simulation matched exactly.

5. No noise was introduced or accounted for.

6. Output sampling at intervals of D = DT/5 would be sufficient to detect undesirable responses; i.e., comply with the Shannon Sampling rate.

7. Control inputs would be applied at intervals no smaller than DT seconds.

Data Collection

Once the initial problem statement/assumptions were specified, numerous program runs were made in an attempt to find the best combination of N and DT commensurate with the storage limitations imposed on N. To facilitate the data runs, a plotting preview routine, CCPREV, was used. After the plots of each run were previewed, the most significant ones were routed to the Calcomp Plotter for hard copies (appearing in Section IV, Results and Discussion). Also appearing in Section IV are figures indicating the sample runs and their respective parameter values.

Revised Assumptions

After the best combination of N and DT was determined, two of the initial assumptions were modified. Specifically, a control input limiter was imposed on the system (Assumption #2) and a model/simulation mismatch was introduced (Assumption #4). Again data was collected, as described in the preceding section, for each situation.

Model Changes

Following the second-order SISO system study, equation (1) was changed to a third-order equation by adding a real eigenvalue $\lambda_3 = -1$. The resulting differential equation became

$$\dddot{y}(t) + 1.5\ddot{y}(t) + 237.7225\dot{y}(t) + 237.2225y(t) = u(t) \quad (15)$$

To facilitate the discretization and differential equation solution, the controller program of Appendix A was modified to accept higher-order differential equations through the use of a computer library routine called ODE. This modified program is shown in Appendix B.

After the above-mentioned program modification was made, third-order data was collected using the "optimum" values obtained in the second-order study. The results were then compared.

The original defining differential equation (1) was then changed to a fourth-order equation using the roots of Mode 1 of Figure 2. These eigenvalues were $\lambda 3,4 = -.55 \pm j6.0$, which resulted in the following differential equation.

$$\ddddot{y}(t) + 1.6\dddot{y}(t) + 274.075\ddot{y}(t) + 279.096\dot{y}(t) + 8611.769806y(t) = u(t) \quad (16)$$

Again data was collected, first for the optimum DT based on the previous second-order results and then for a DT based on the new frequency $\omega_2 = 6.0$ rad/sec.

IV Results and Discussion

This section contains samples of the computer products obtained in the simulations run on the hypothetical MAC control problem, beginning with the results obtained in the initial search for the optimum DT and continuing on through the various parameter variations mentioned in Section III. A narrative throughout describes the salient features noted in the Calcomp plots and line printer output.

Searching for the Optimum Value of DT

In attempting to arrive at the optimum value of DT, over 150 computer simulations were run. In these simulations, the relationship CONTIME = N·DT was held constant at a value of 24 seconds (based on assumption number 3) while DT was varied. The results of each run were displayed in two formats: Calcomp plots and line printer output, both of which were evaluated qualitatively for satisfactory output (based on first-order model characteristics and control inputs required to follow the model).

Intuitively, it was felt that the best place to start in the search for an optimum value of DT was with the smallest value of DT that would be consistent with the constraints that N was limited to a value of 300. Since it was assumed that 24 seconds of impulse response information was to be stored, this gave $DT = \frac{CONTIME}{N} = \frac{24}{300} = 0.08$ seconds as the minimum DT allowed in the problem statement. Thus 0.08 seconds was the beginning point in the trial and error search for the best DT for this particular problem. The results of this selection and a representative sample of other selections leading up to the "best" DT selection are shown in Figures 3 through 17.

From the first selection of DT = 0.08, it can be seen that the impulse response function (Figure 3) is not contained within a pure decaying exponential envelope. In fact, between two and six seconds a slight "bulge" of the envelope appears. At 24 seconds there still appears to be some slight influence of the impulse response function remaining.

The control inputs, as shown in Figure 4, are oscillatory with a frequency approaching two and one-half times the natural system frequency indicated in Figure 3.

The system output (Figure 5) tracks the desired first-order trajectory "on-the-average", but has an oscillatory nature corresponding to the oscillatory inputs (approximately 2.7 times the natural system frequency).

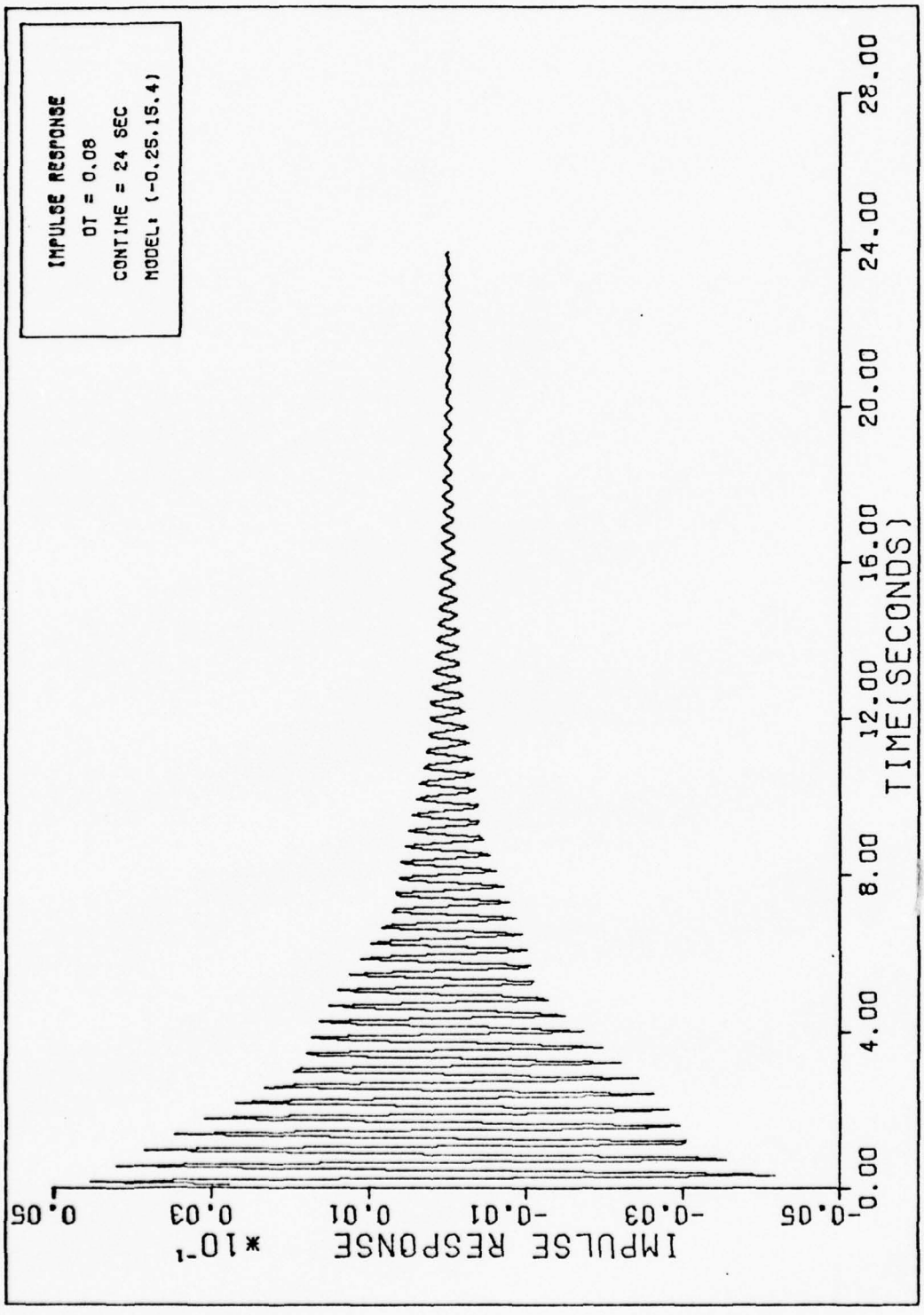


Figure 3. Discrete Impulse Response with DT = 0.08

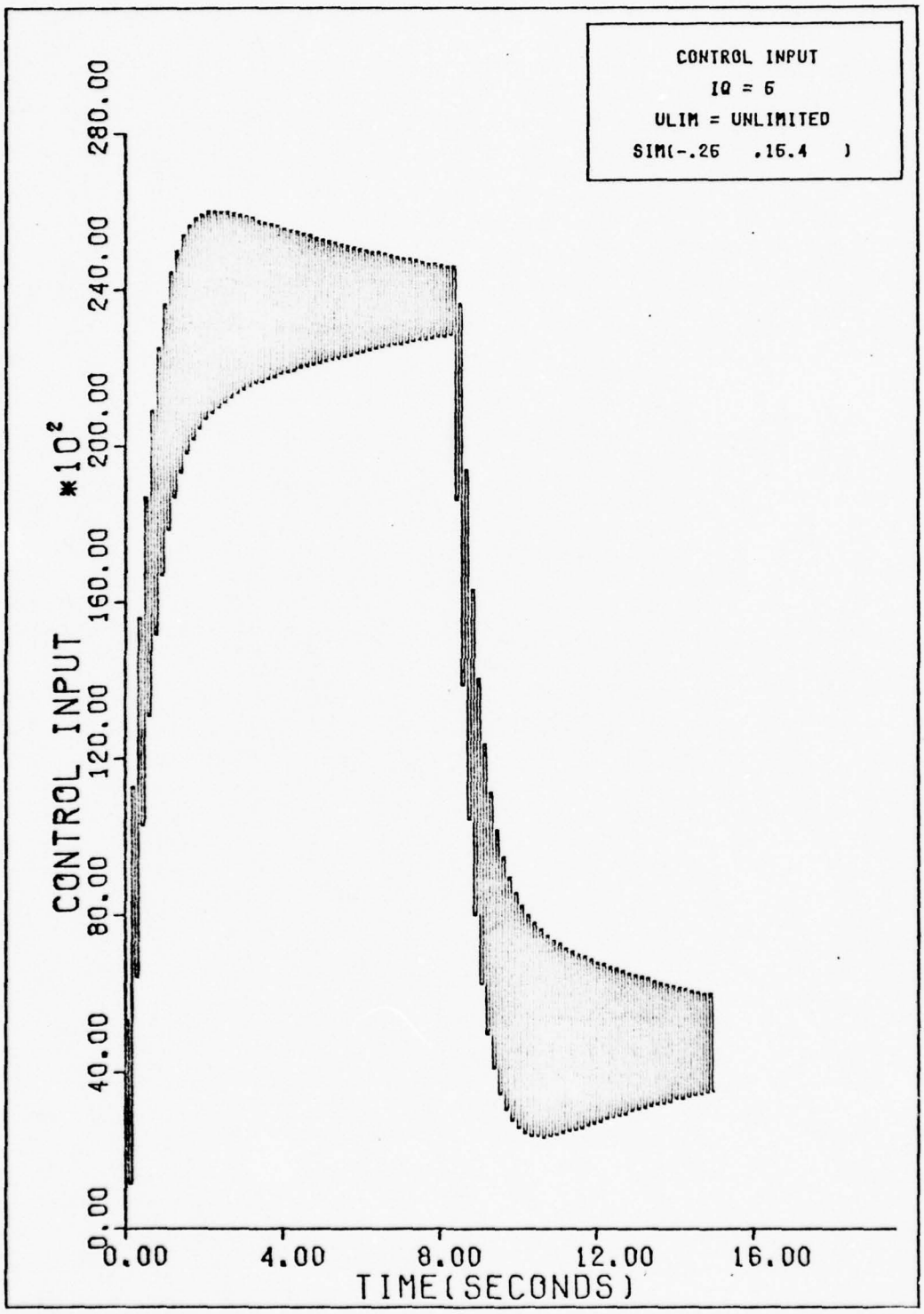


Figure 4. Control Inputs for DT = 0.08

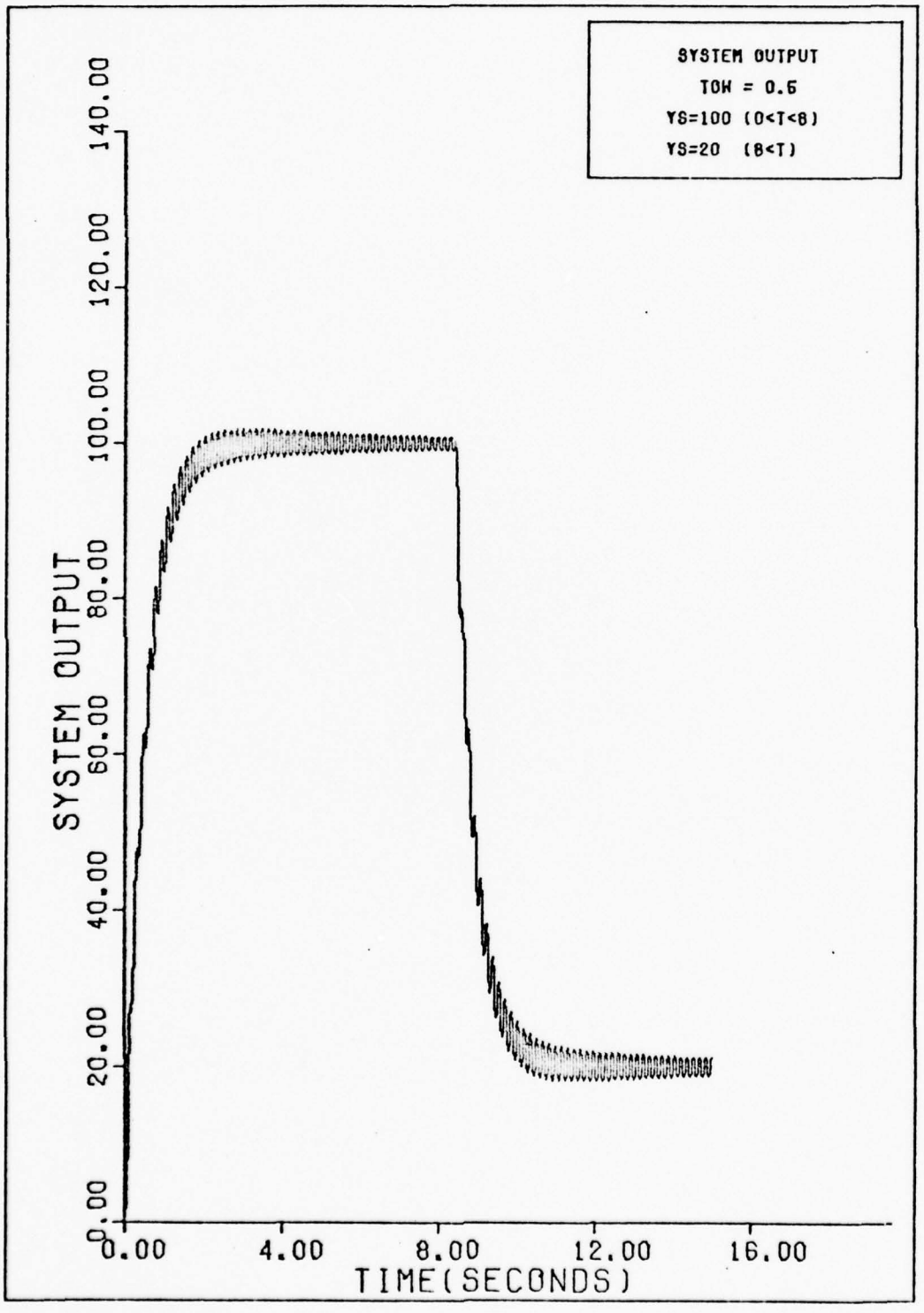


Figure 5. System Output with DT = 0.08

The results indicated in Figures 6 through 8 indicate the effect of an increase in sample time to DT = 0.10. These results are similar to the preceding case where DT = 0.08; however, the impulse response function (Figure 6) has become more deformed.

The control inputs are still oscillatory, with roughly the same increase in frequency over the system's natural frequency as in the DT = 0.08 case; however, the magnitude of control inputs has decreased. The system output is still oscillatory with the frequency decreasing slightly to just over 2.2 times the natural system frequency; however, the magnitude changes are slightly larger than in the preceding case.

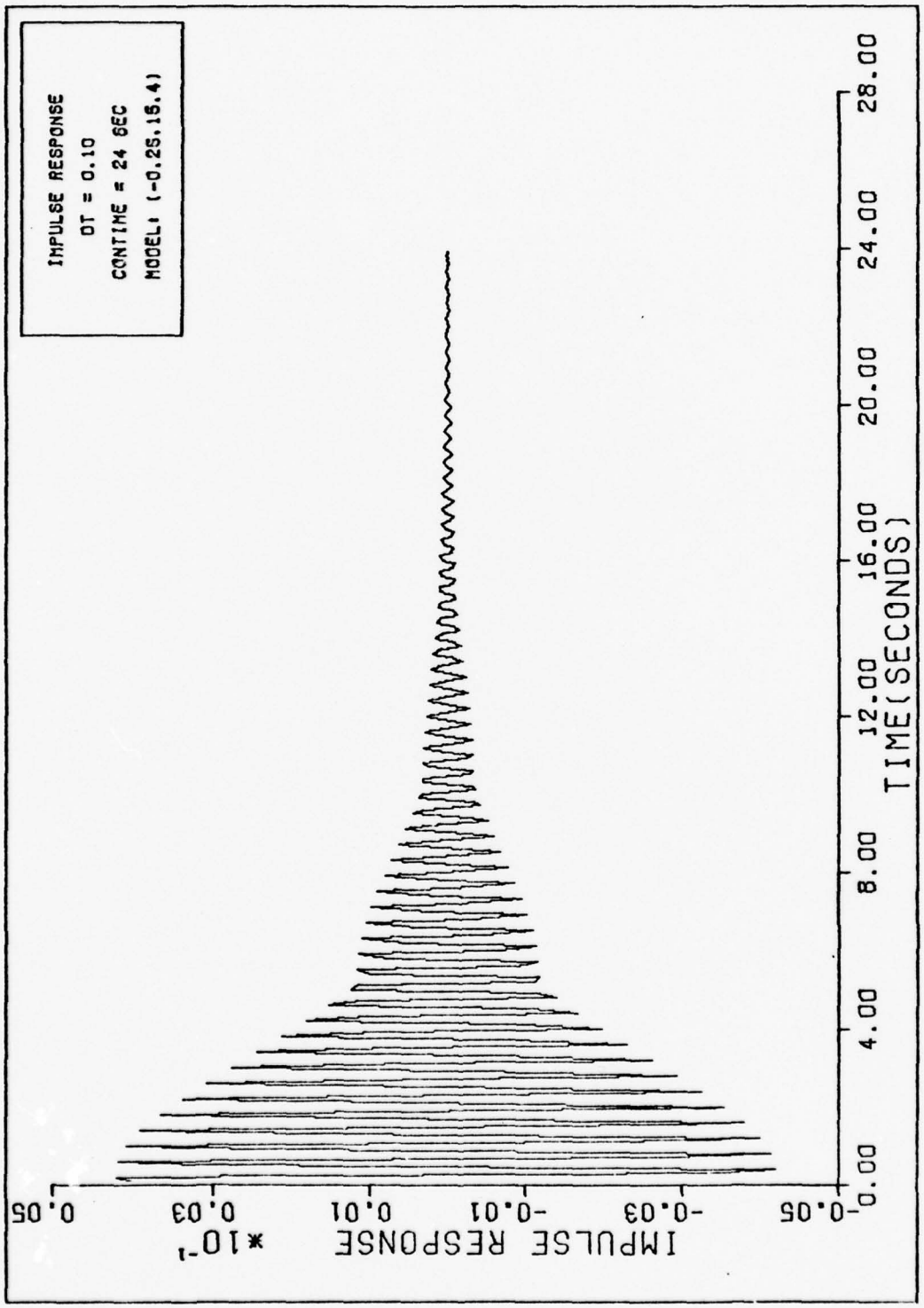


Figure 6. Discrete Impulse Response with DT = 0.10

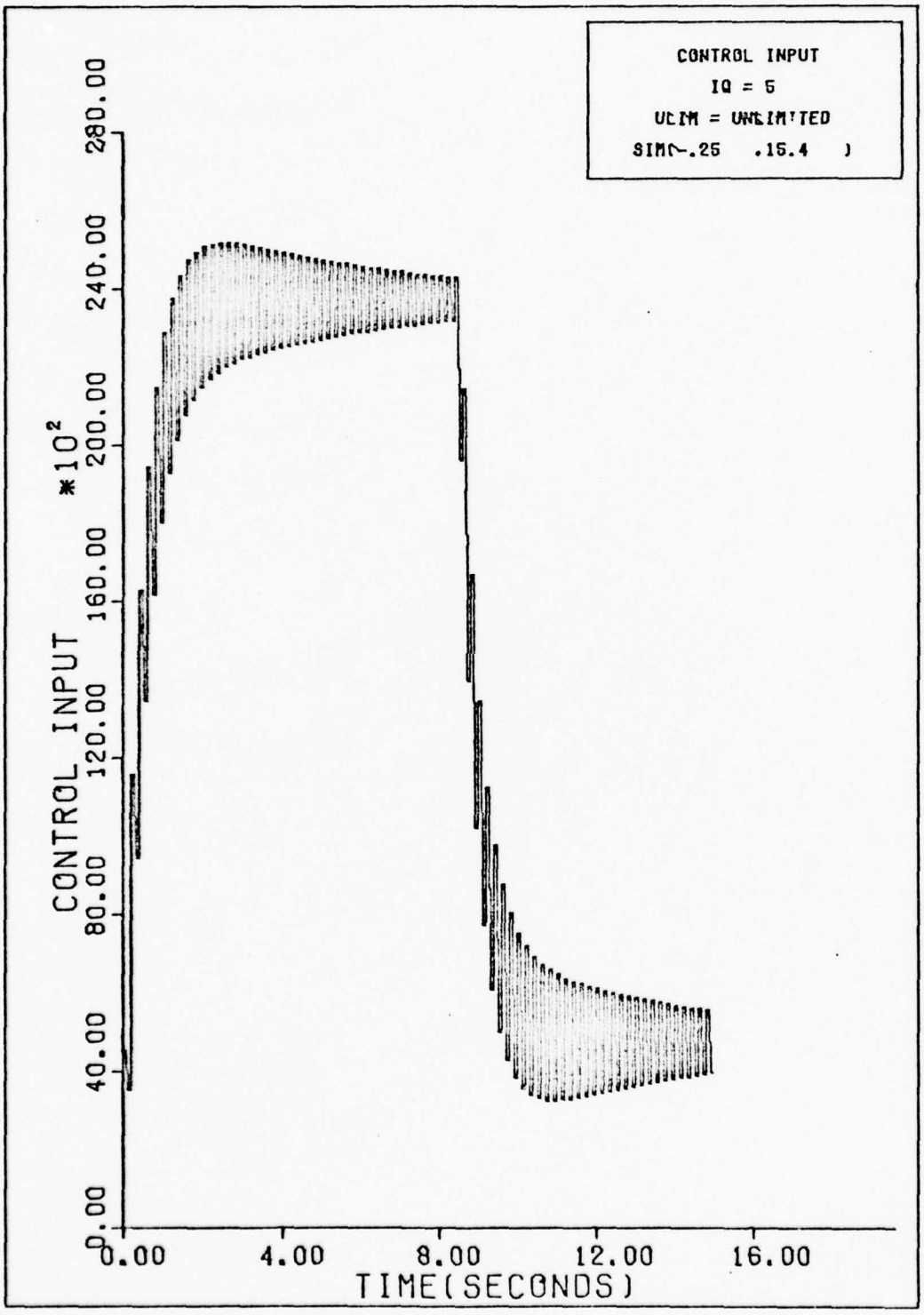


Figure 7. Control Inputs for DT = 0.10

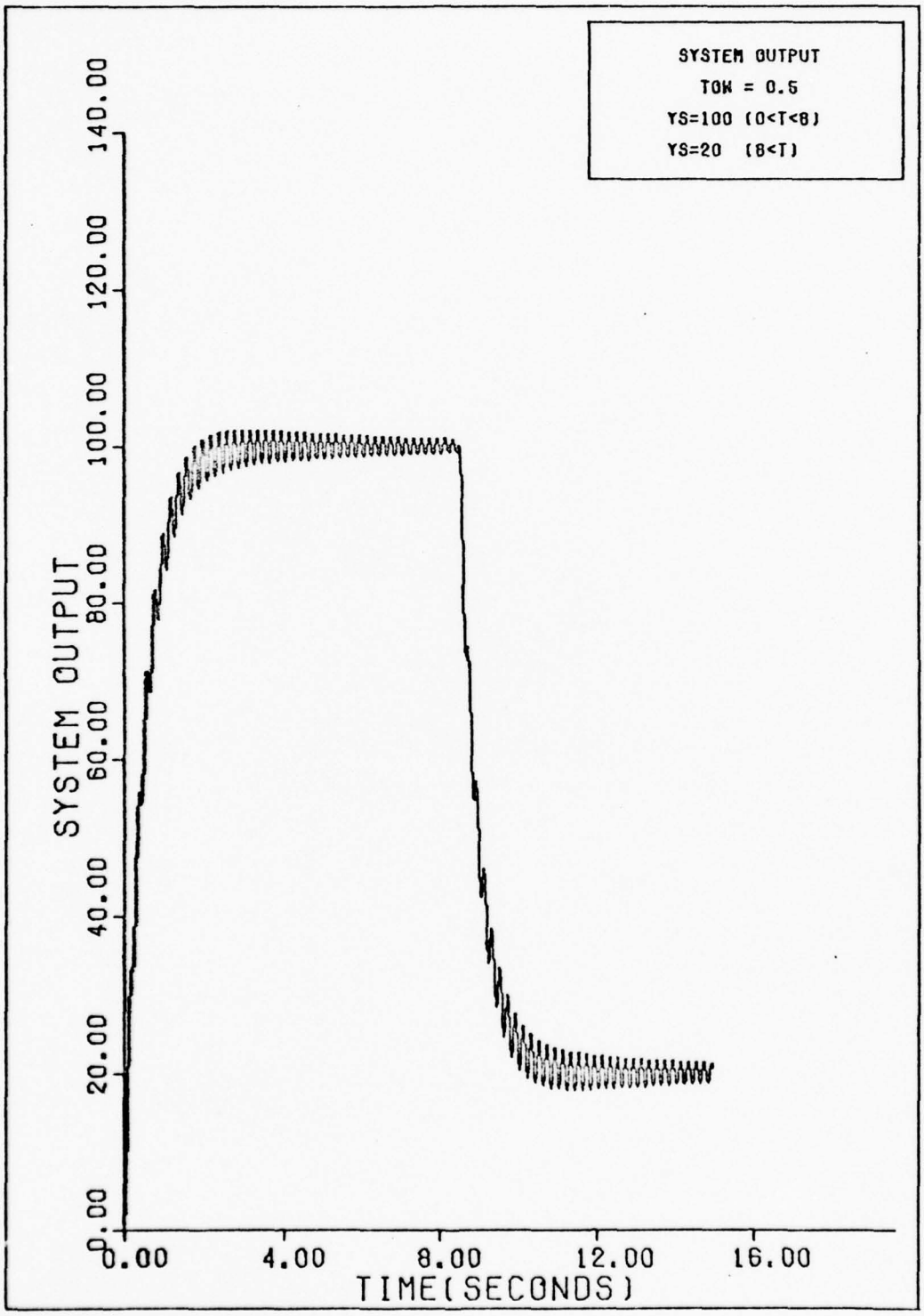


Figure 8. System Output with DT = 0.10

Figures 9 through 11 illustrate an interesting result. The impulse response function of Figure 9 illustrates the effect of "folding" at DT = 0.195, as the Shannon Sampling Frequency (Ref 6: III-18) is approached; however, the control inputs become almost stairstep in nature (Figure 10).

Corresponding to the eliminated oscillatory nature of the control inputs, the system output has become less oscillatory (in both frequency and magnitude). In fact, the output frequency is very near the natural frequency of the system.

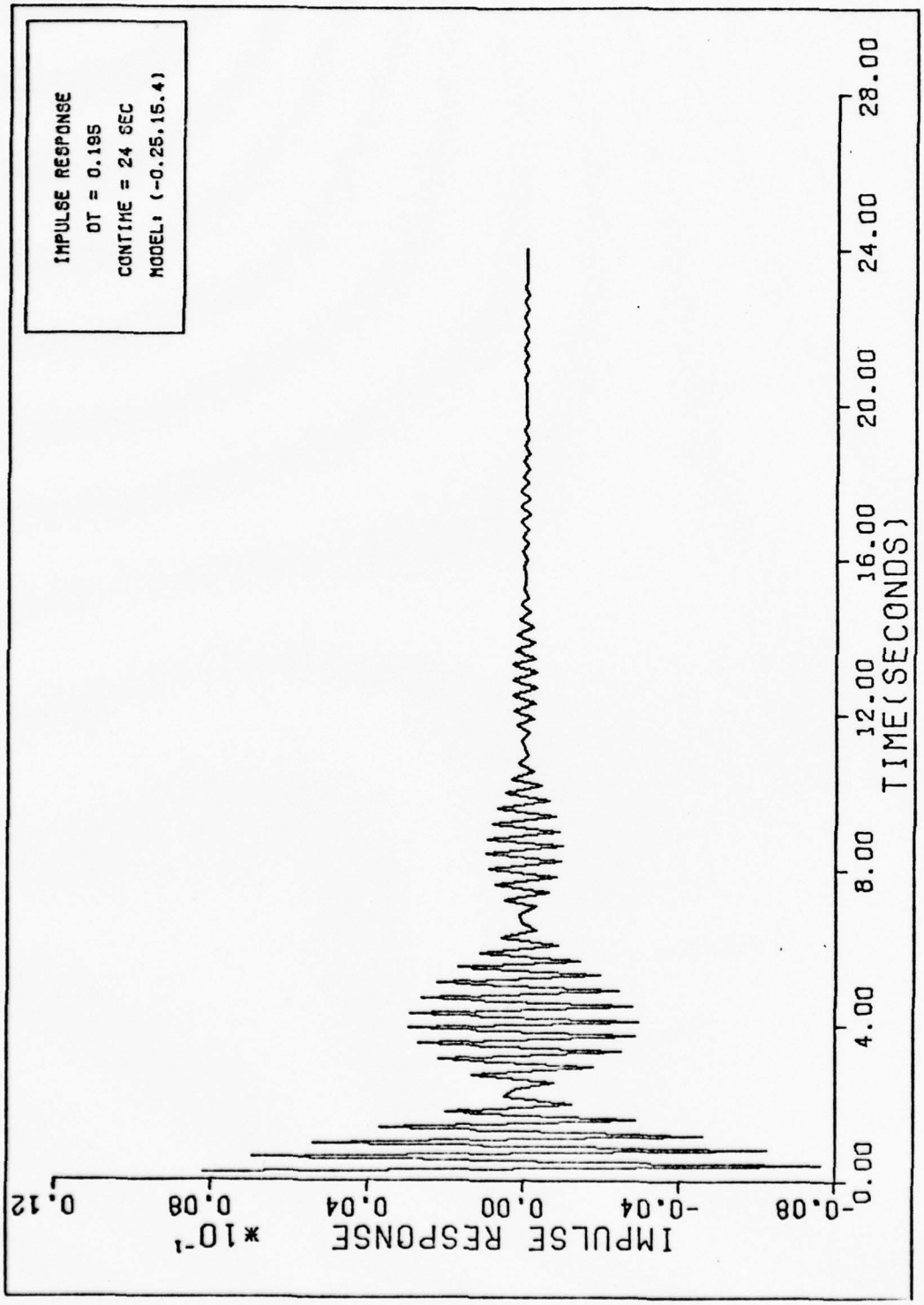


Figure 9. Discrete Impulse Response with DT = 0.195

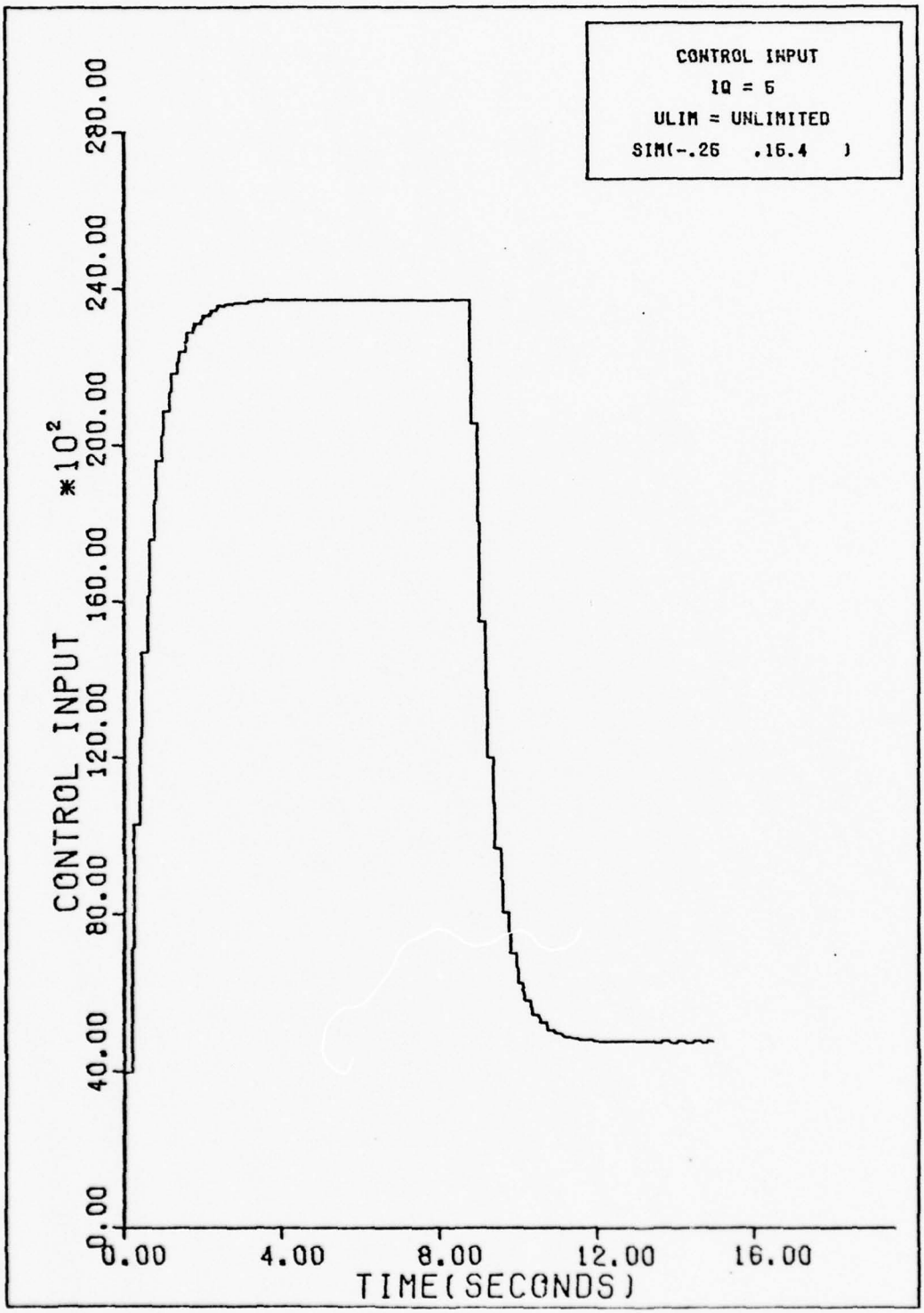


Figure 10. Control Inputs for DT = 0.195

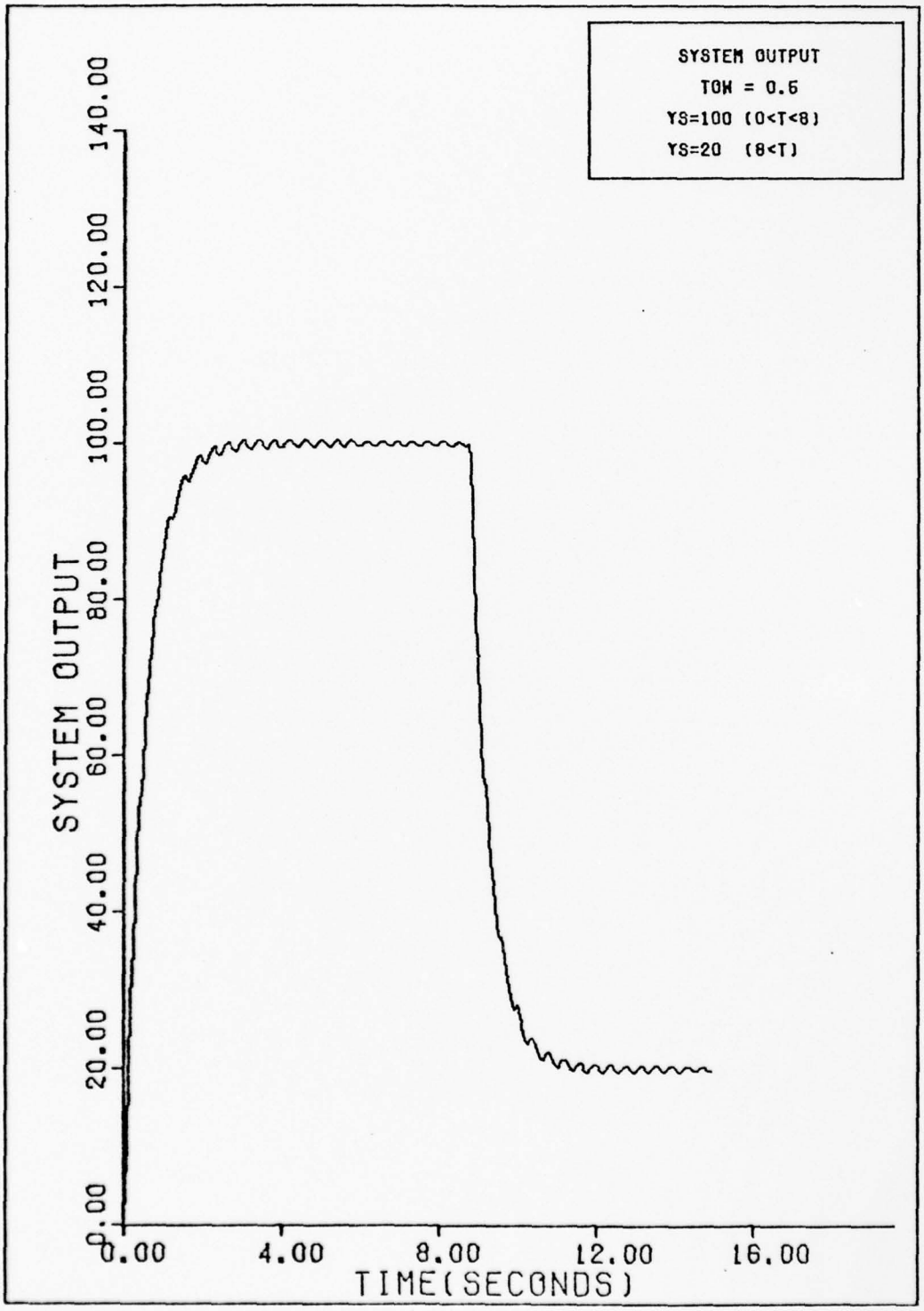


Figure 11. System Output with DT = 0.195

Perhaps the most interesting results of all are indicated in Figures 12 through 14. In that case DT was chosen as π/ω, which was the same as that of the Shannon sampling rate (Ref 6: III-18). Here, the impulse response function exhibited a "clean" exponentially decaying envelope (Figure 12). The control inputs became more stair-step in nature and the system output was devoid of any oscillation. This resulted in DT = π/ω being chosen as the "optimum sampling time".

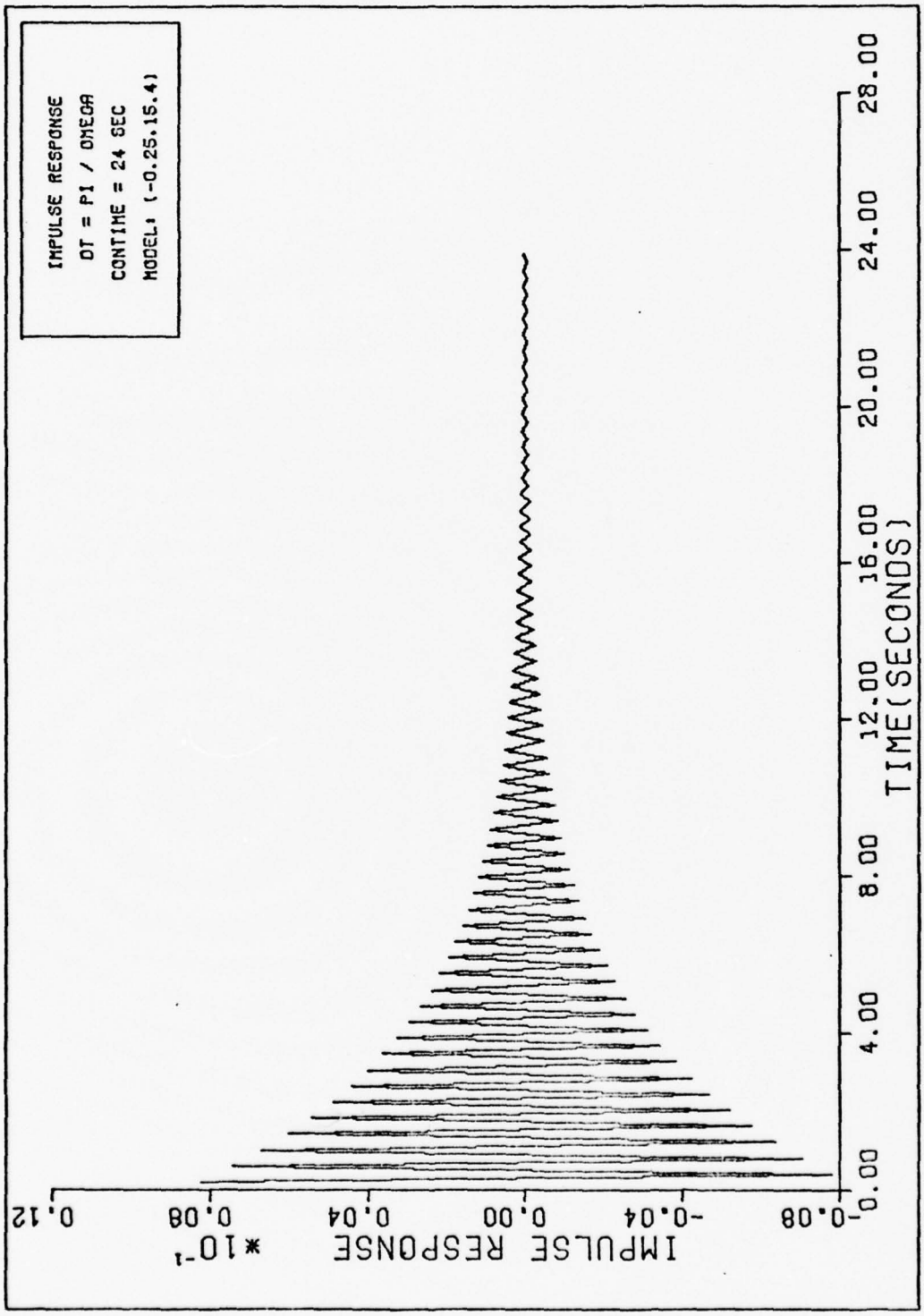


Figure 12. Discrete Impulse Response with DT = π/ω = 0.203999523

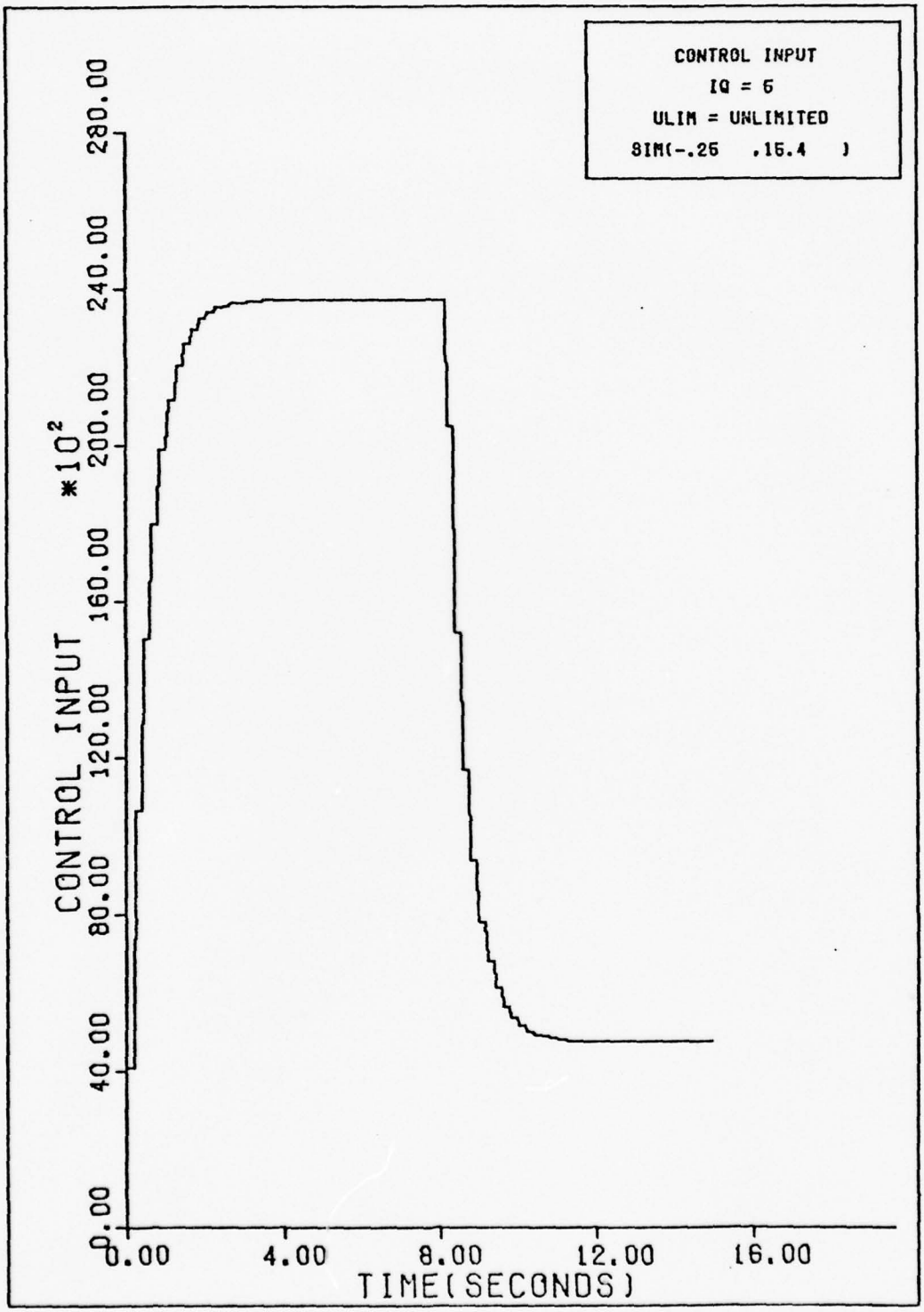


Figure 13. Control Input for DT = 0.203999523

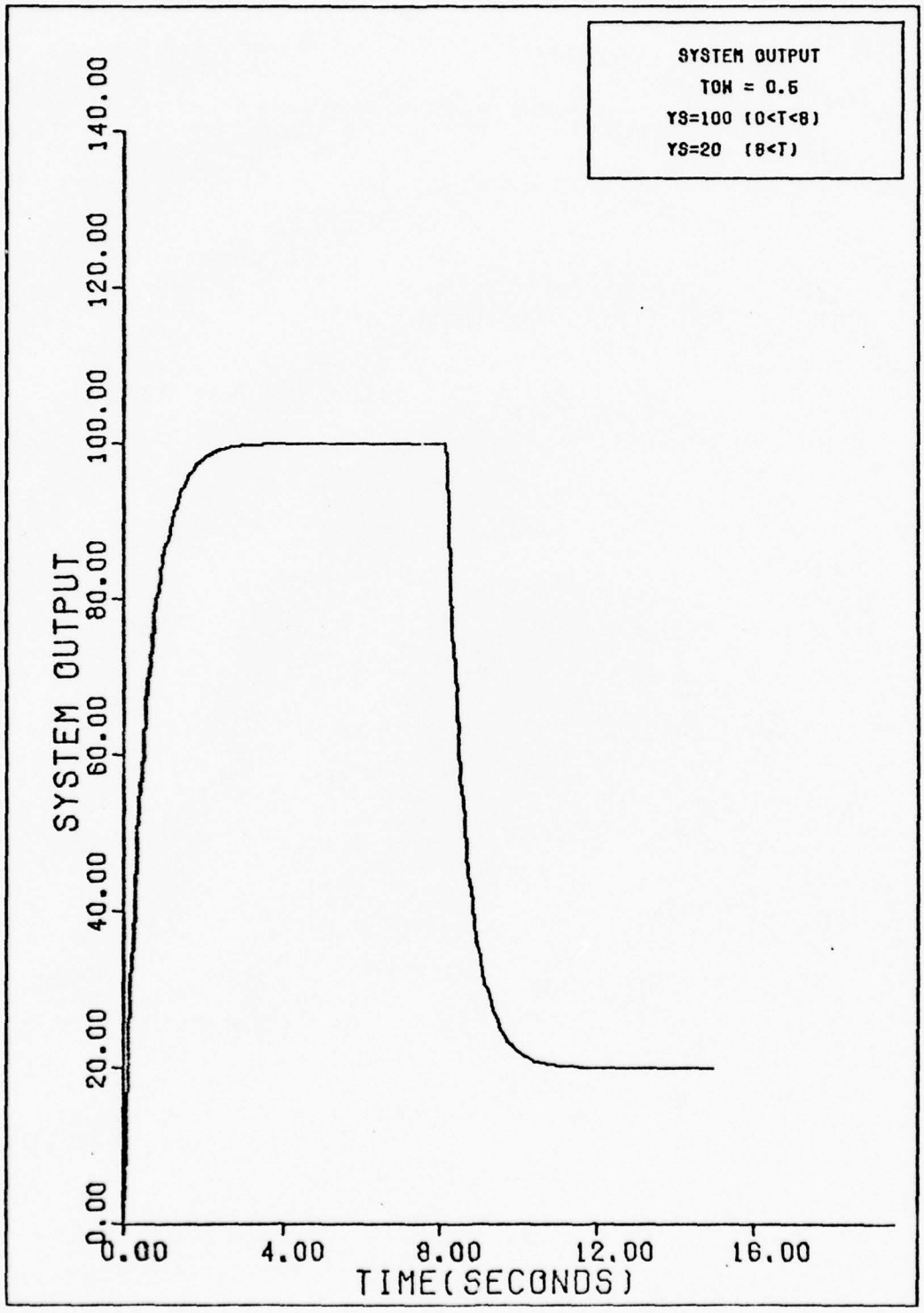


Figure 14. System Output for DT = 0.203999523

Increasing the sampling time beyond DT = π/ω resulted in the effects shown in Figures 15 through 17 for the case with DT = 0.3. The degradation of the impulse response function is apparent in that the natural system frequency is no longer portrayed. Instead, a frequency of approximately half that of the preceding impulse response representations results. The control inputs have become less oscillatory, as shown in Figure 16, and the system output (Figure 17) now displays larger magnitude changes than any of the preceding cases.

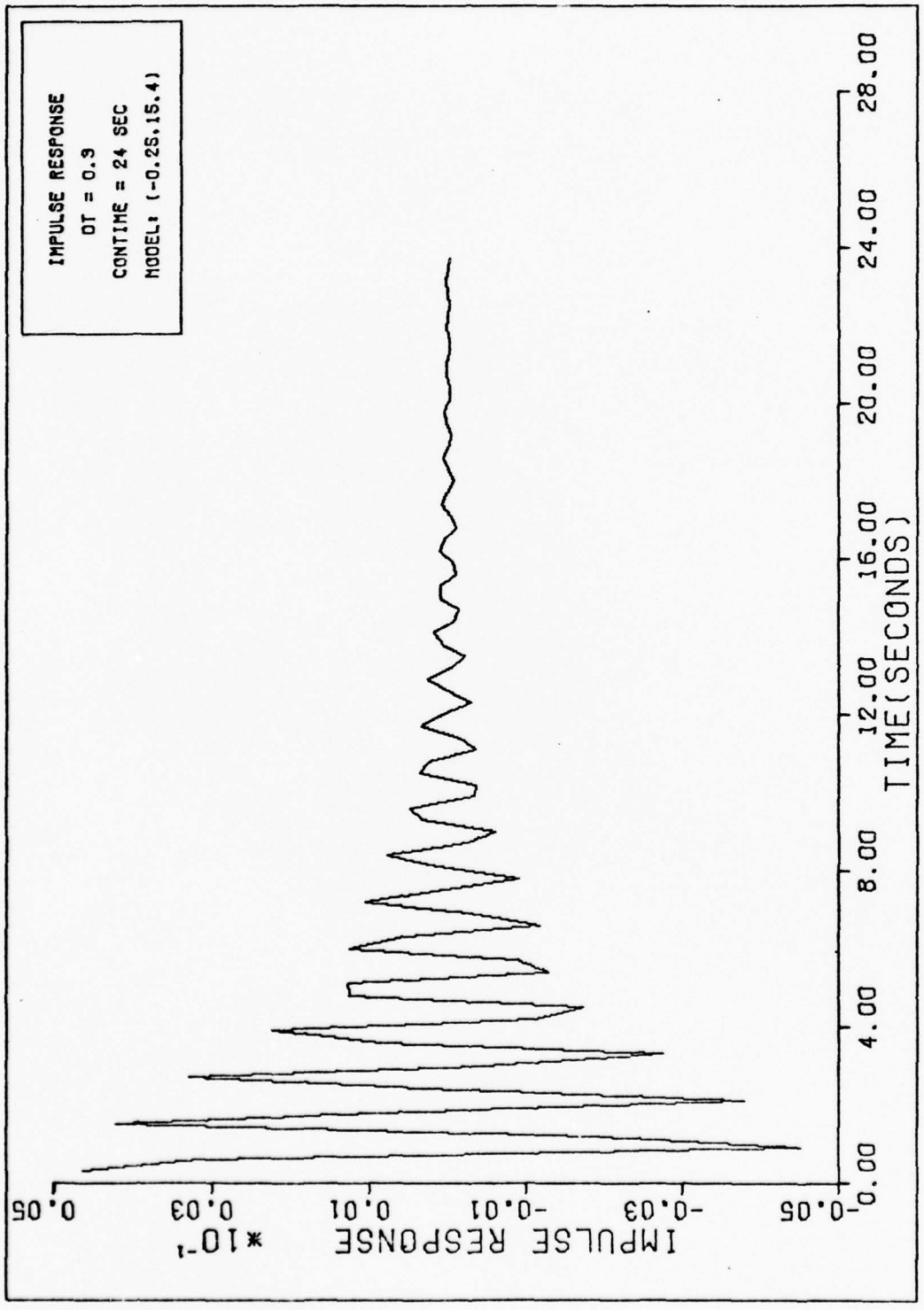


Figure 15. Discrete Impulse Response with DT = 0.3

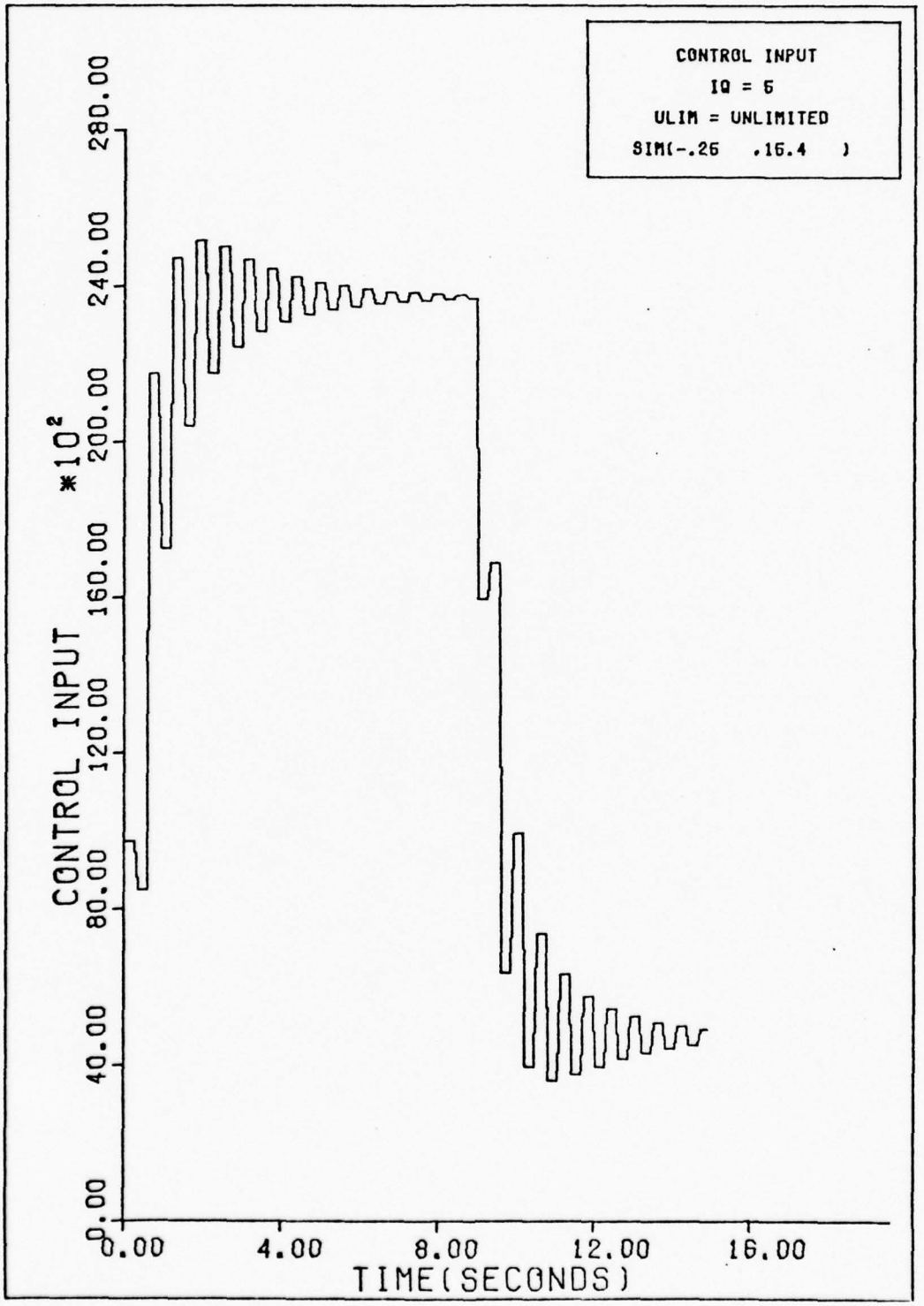


Figure 16. Control Input for DT = 0.3

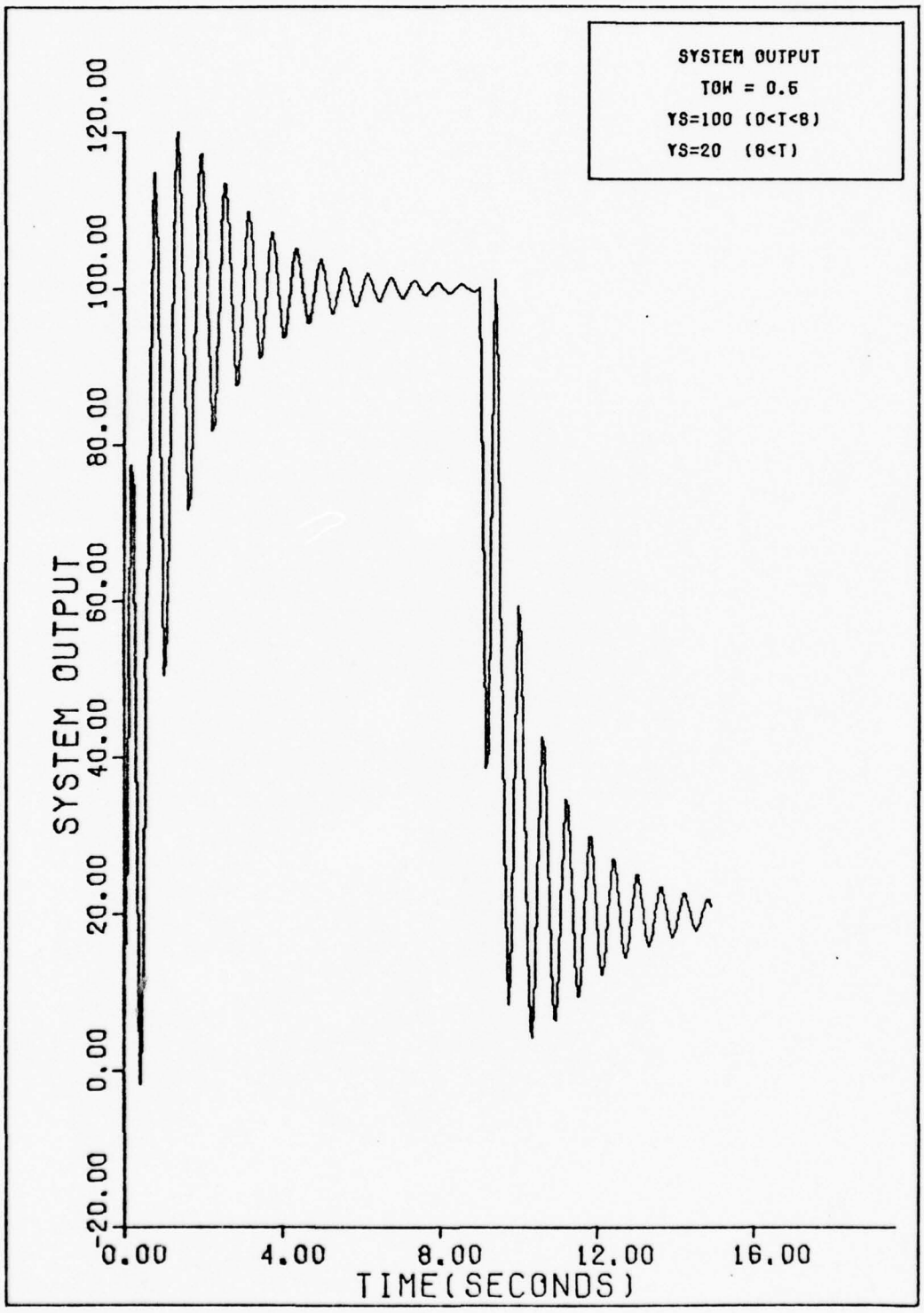


Figure 17. System Output for DT = 0.3

Although only the plots for five selected values of DT are presented here, the progression was fairly consistent, with better results being obtained the closer DT came to π/ω. In fact, DT = π/ω gave the best results of all; however, exceeding this value produced oscillatory outputs (see Figures 15 through 17). Any deviation (greater than $\pm$ 1%) from that value degraded the output response significantly; i.e., oscillations in the output response occurred. These results were unexpected, since the value of DT = π/ω is identically the period of the Shannon sampling rate (Ref 6: III-18). It is also noteworthy that this value of DT yields a value of N = $\frac{24}{DT}$ = 118, which implies that more impulse response information than initially assumed can be stored if so desired.

Figures 18 and 19 illustrate the performance of the controller out to 40 seconds. As can be seen from these figures, a slight deviation from the set point, y_s = 20, occurs at approximately 24 seconds.

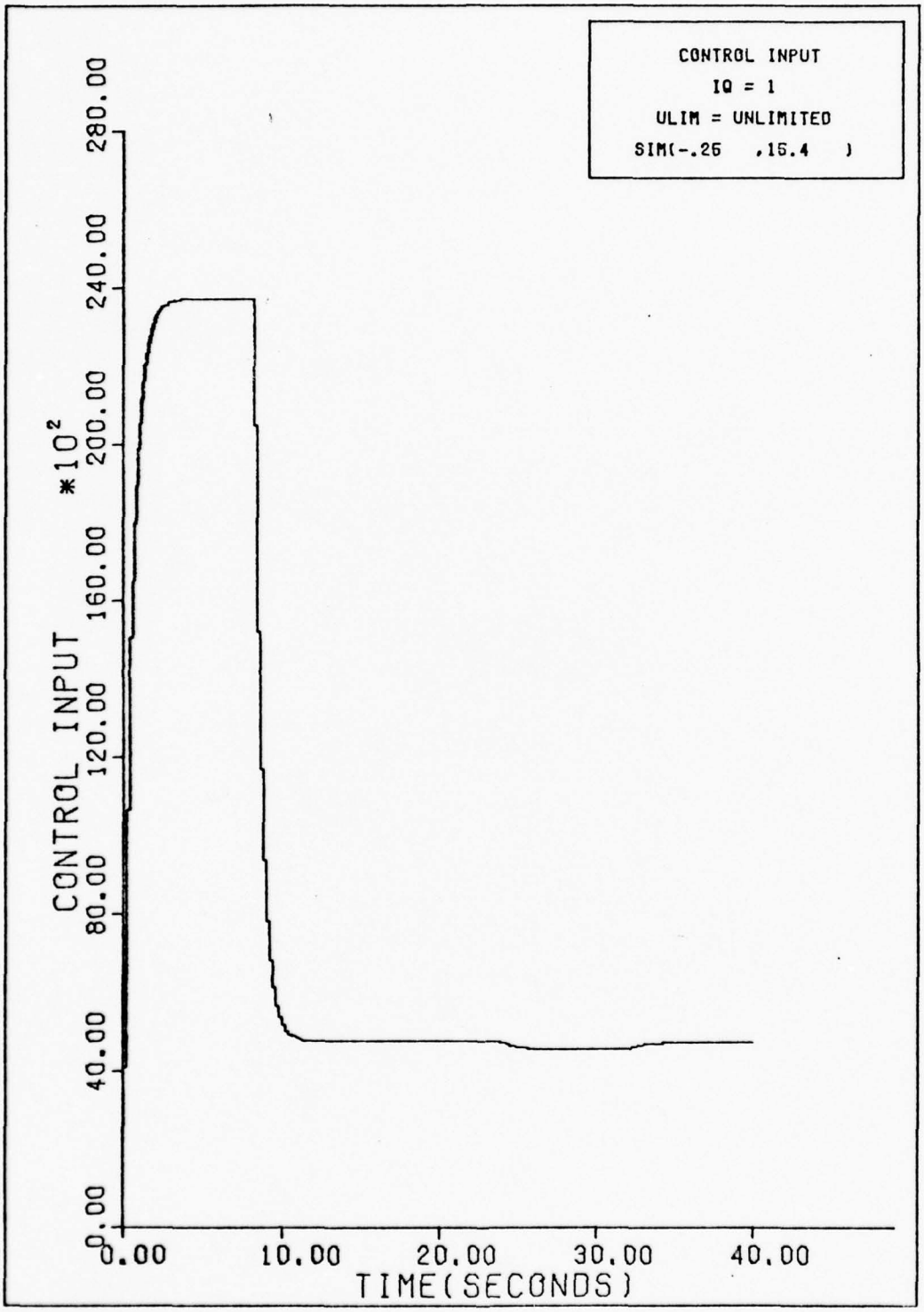


Figure 18. Control Input, 40 second Simulation

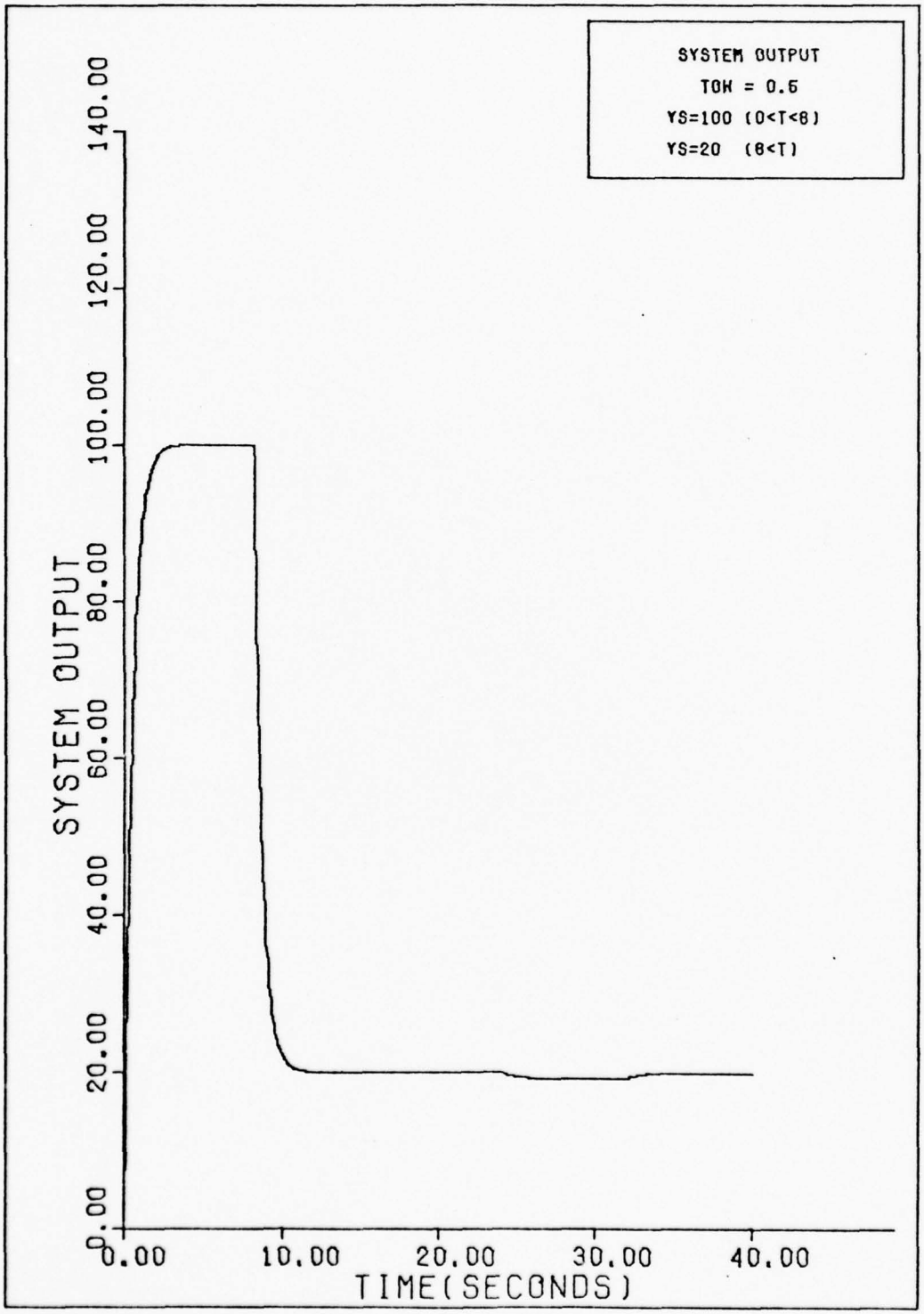


Figure 19. Systems Output, 40 Second Simulation

Variation of CONTIME

Since the 24 seconds mentioned in the preceding section corresponded to the impulse response information stored (CONTIME), an increase in CONTIME was made in an attempt to improve the results. The minimum value of CONTIME that eliminated the y_s deviation was 33 seconds. Figures 20 through 22 illustrate the effect of this change.

For contrast, the effects of shortening CONTIME by one-half the initially assumed value are shown in Figures 23 through 25. From these figures it can be seen that the system performed satisfactorily for the first 12 seconds; i.e., the first set point $y_s = 100$ was attained smoothly and so was the second set point $y_s = 20$. However, at 12 seconds the system output departed the second set point and never re-established it. This was the result of disregarding a significant portion of the system dynamics in the computation of required inputs to follow a specified trajectory; i.e., the impulse response (see Figure 23) was assumed to be zero after 12 seconds. Since an error was introduced in any interval shorter than 33 seconds, this result was not too surprising.

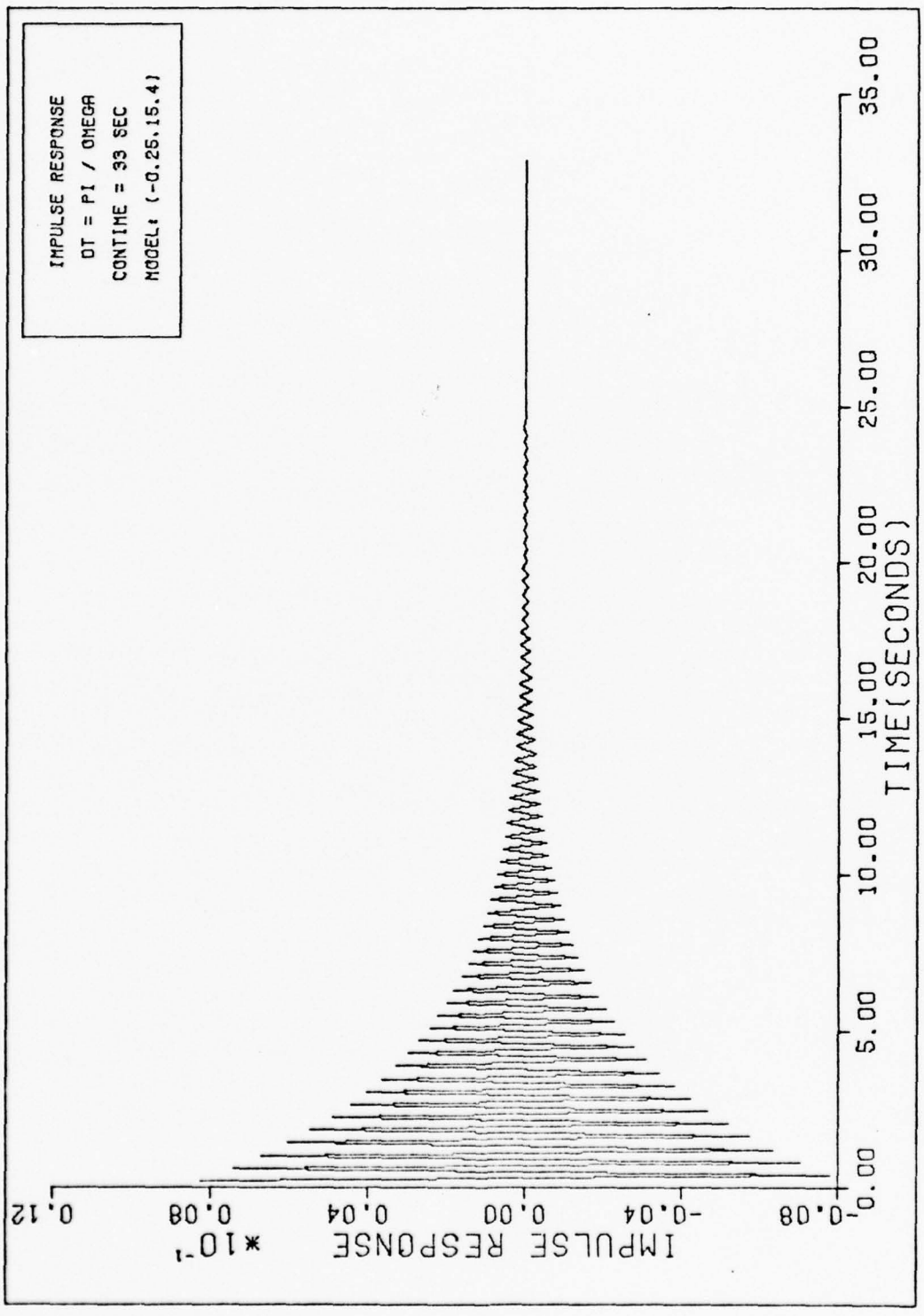


Figure 20. Discrete Impulse Response for CONTIME = 33 Seconds

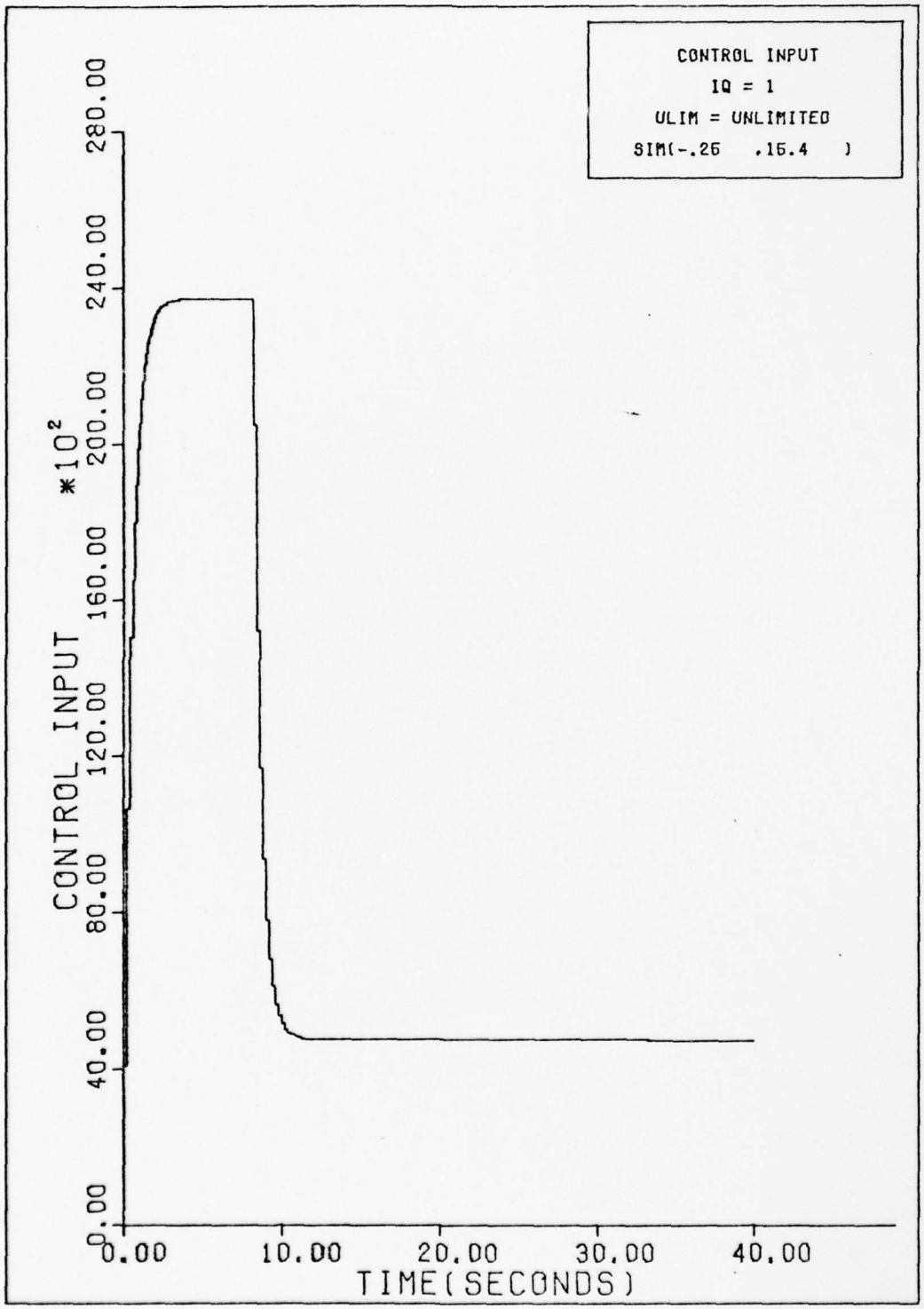


Figure 21. Control Inputs for CONTIME = 33 Seconds

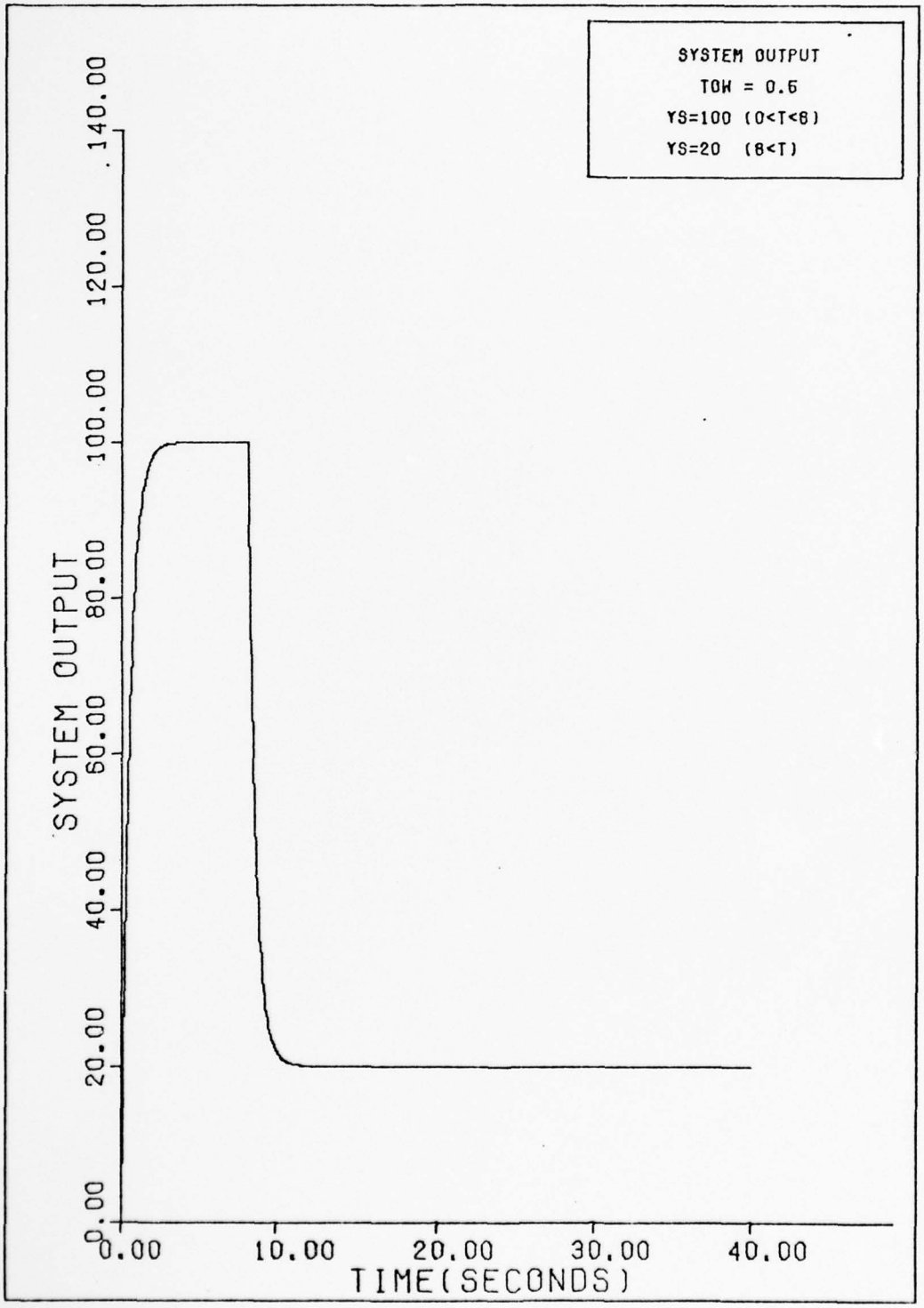


Figure 22. System Output with CONTIME = 33 Seconds

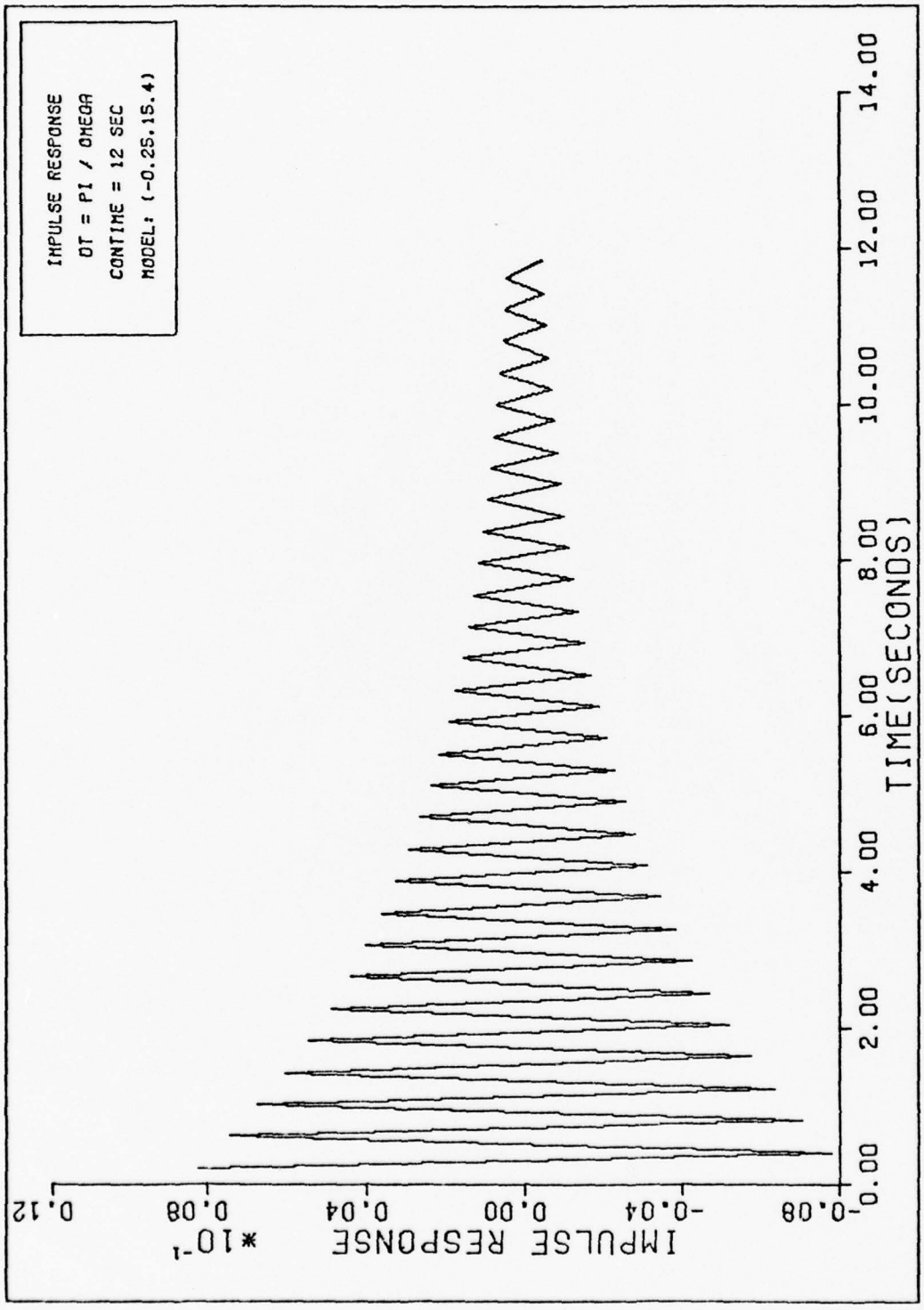


Figure 23. Discrete Impulse Response for CONTIME = 12 Seconds

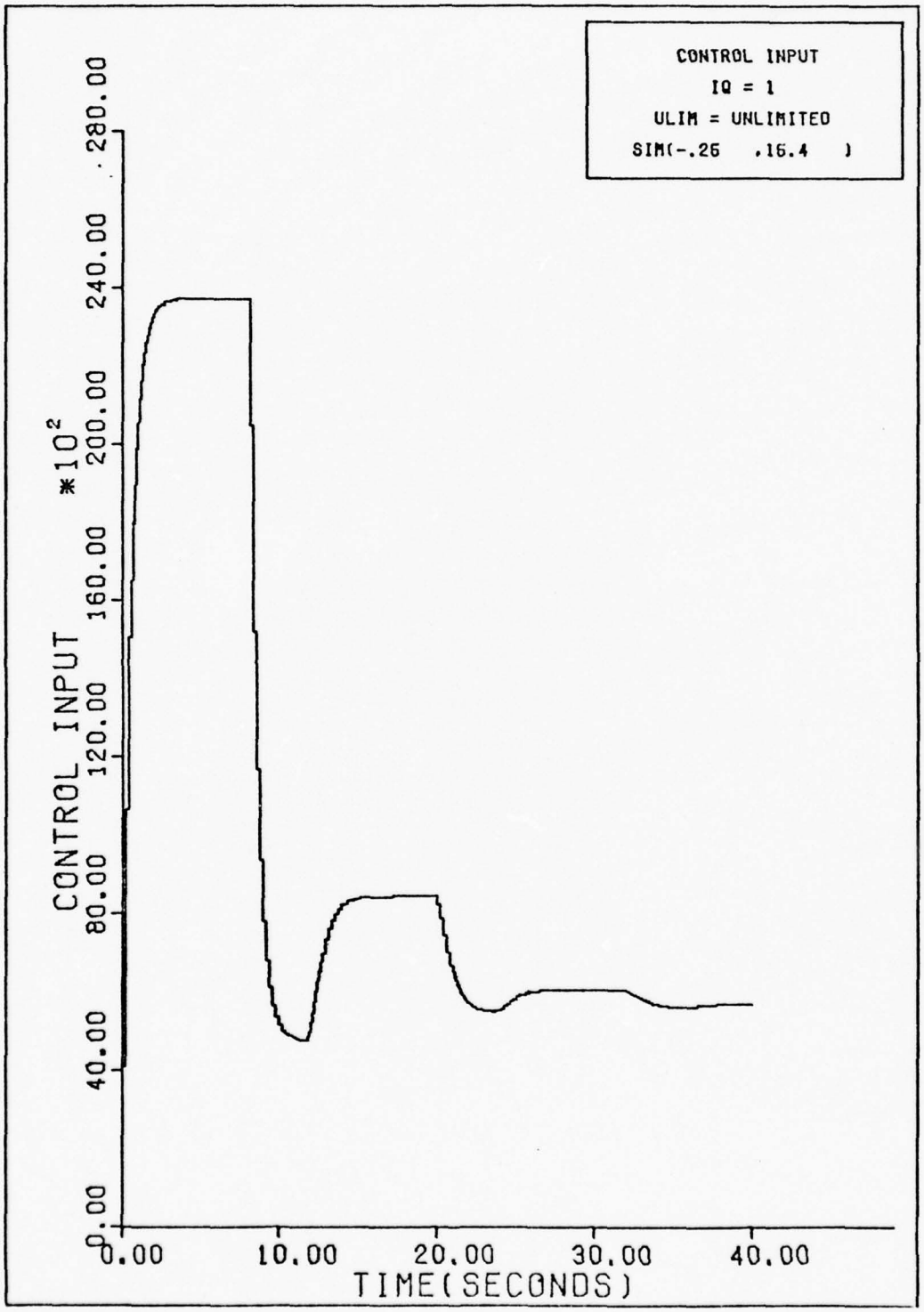


Figure 24. Control Inputs with CONTIME = 12 Seconds

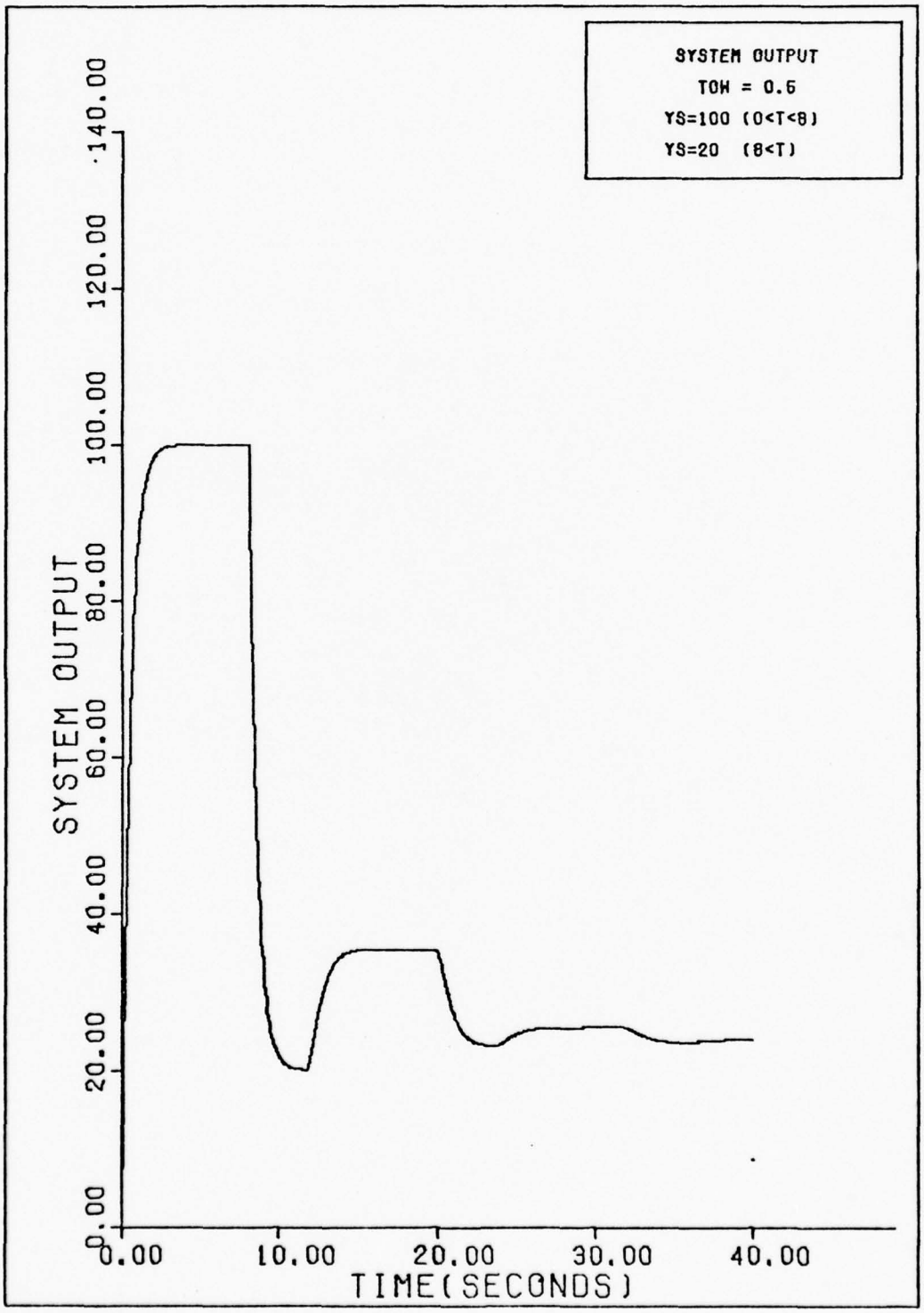


Figure 25. System Output with CONTIME = 12 Seconds

Introduction of Model/Simulation Mismatch

In the next phase of the investigation, an eigenvalue mismatch between the mathematical model and the simulated system was introduced. This was implemented by specifying different eigenvalues for the model after the discrete impulse response was calculated and stored. The object of this change was to see how well the controller attained and then maintained the specified set points, given an incorrect discrete impulse response function. Figure 12 was still applicable for the discrete impulse response; however, Figures 26 through 29 illustrate the effects of 1.0% and 5.0% eigenvalue changes. An inspection of these figures reveals that the magnitude of the system's deviation from the specified set points increases as the eigenvalue error increases.

Table I compares the line printer output for four cases. Apparently, the magnitude of the output error depends also on the value of the set point, as the percent error was not exactly the same for the high and low set points associated with a particular percent eigenvalue error; however, the results are similar. Additionally, the observed effect was noticed primarily in the steady-state response, with very little effect introduced in the transient response.

These results were unexpected because of the MAC's robustness property as demonstrated in IDCOM. In fact, they conflict with the findings in references 1, 2, and 3.

Table I Effects of Eigenvalue Change on Output

Percentage Eigenvalue Change $\Delta\lambda$ (Decrease)	Set Point y_s	Average Actual Output y $\lvert y_s - y \rvert$	Average Deviation from Set Point	Average Per Cent Steady-State Error %e
0.1%	100.0	100.6	0.6	0.6%
	20.0	20.13	0.13	0.7%
0.5%	100.0	103.1	3.1	3.1%
	20.0	20.7	0.7	3.4%
1.0%	100.0	106.4	6.4	6.4%
	20.0	21.3	1.3	6.5%
5.0%	100.0	141.4	41.4	41.4%
	20.0	28.6	8.6	43.0%

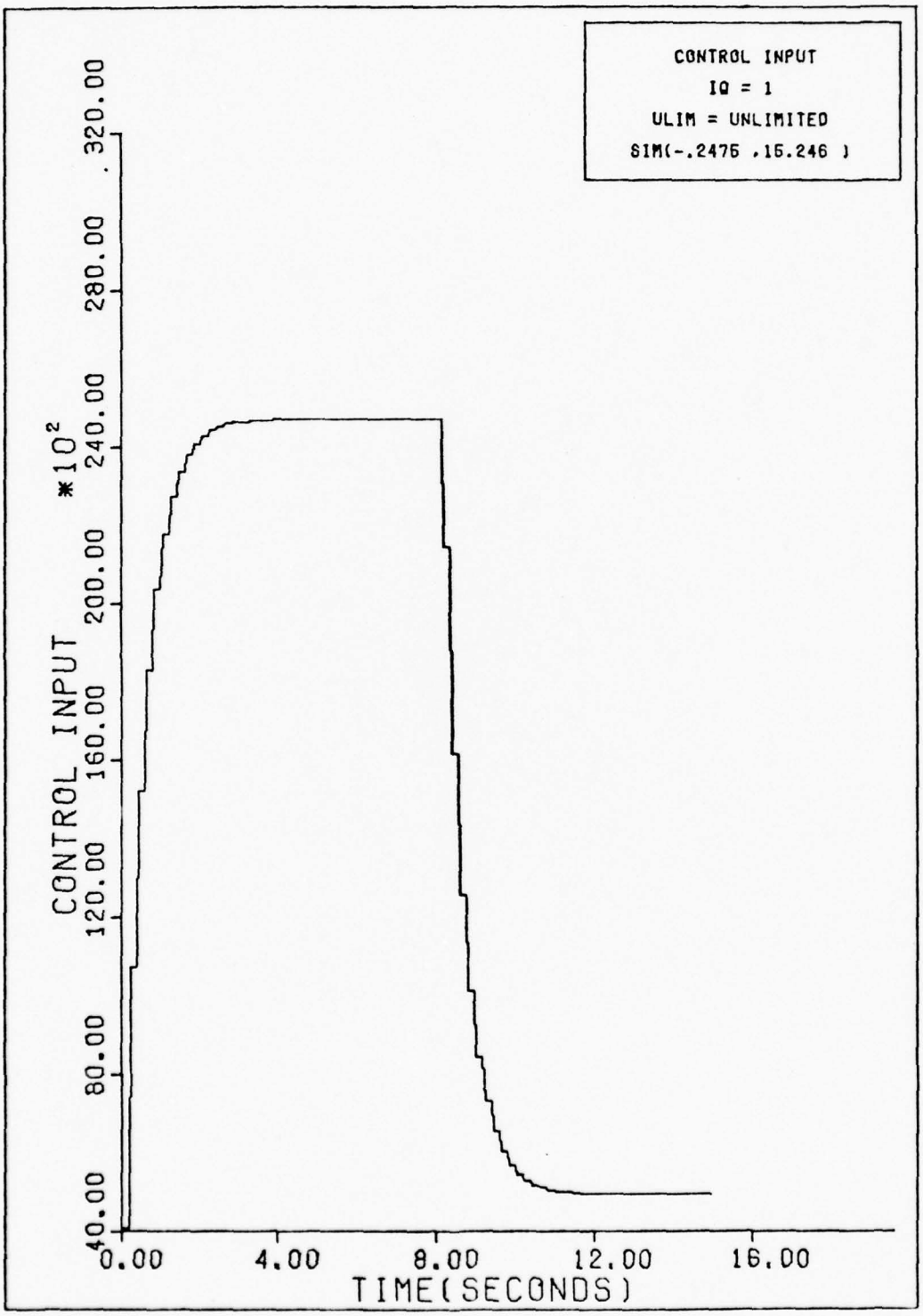


Figure 26. Control Inputs with 1.0% Eigenvalue Change

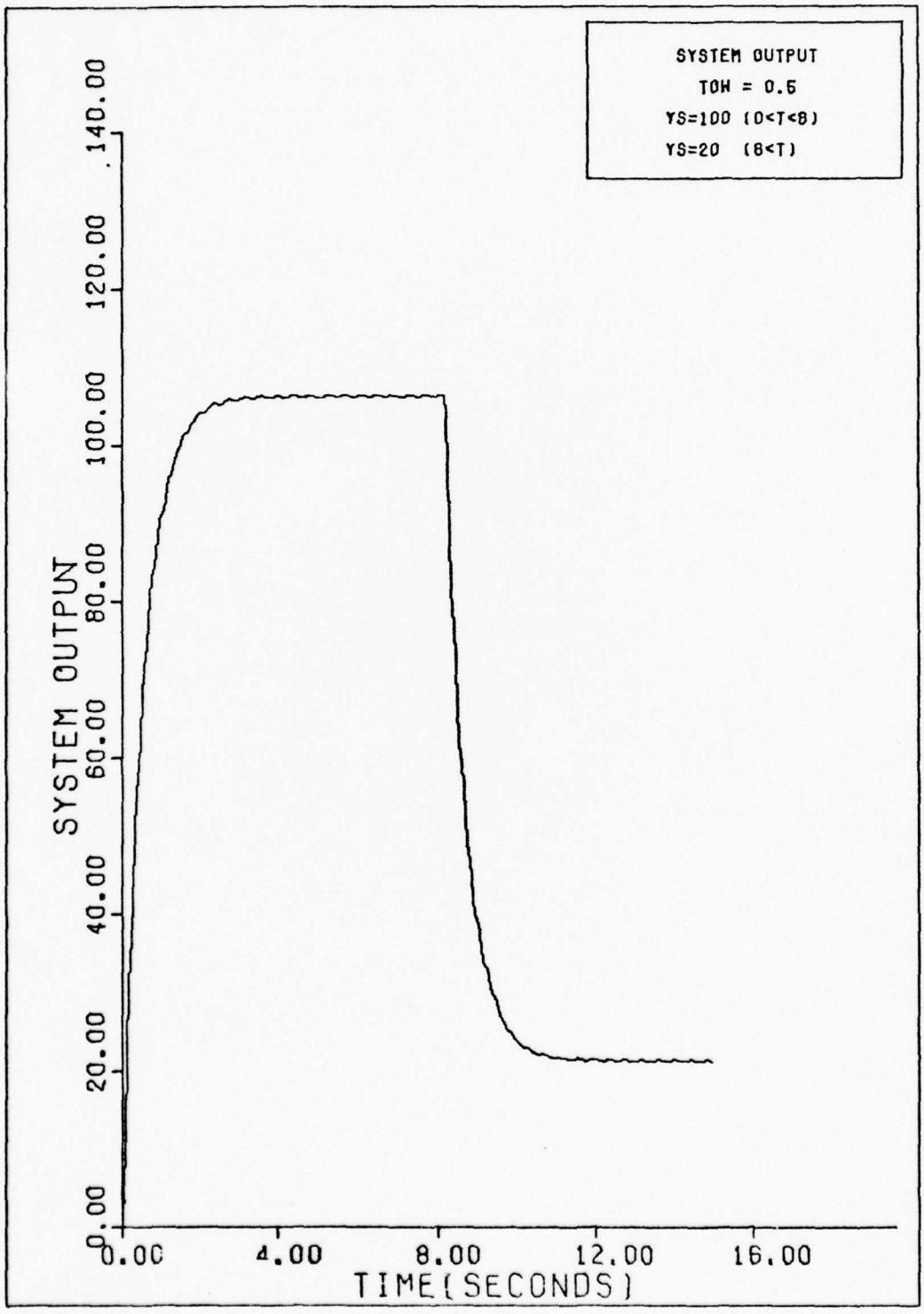


Figure 27. System Output with 1.0% Eignevalue Change

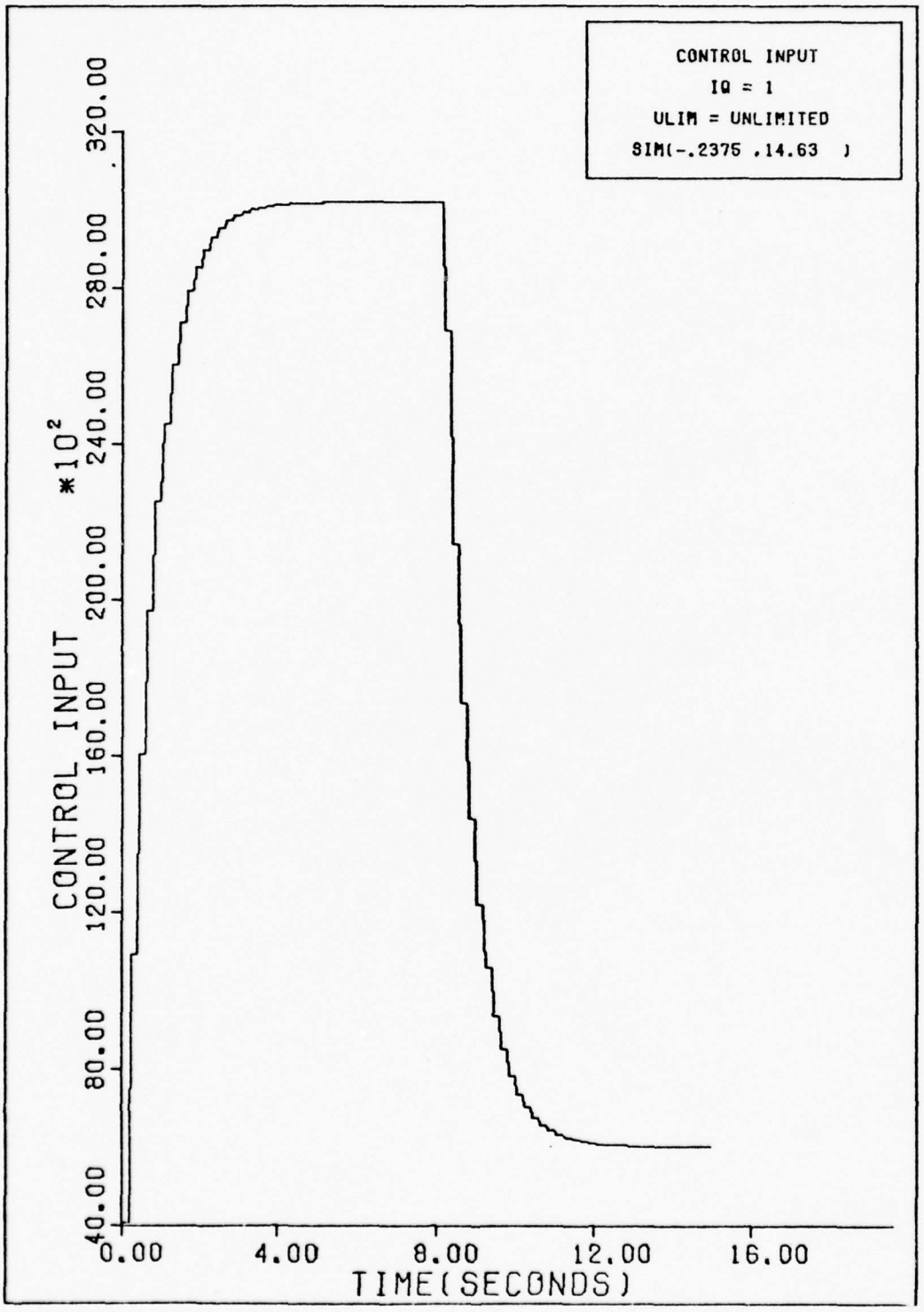


Figure 28. Control Inputs with 5.0% Eigenvalue Change

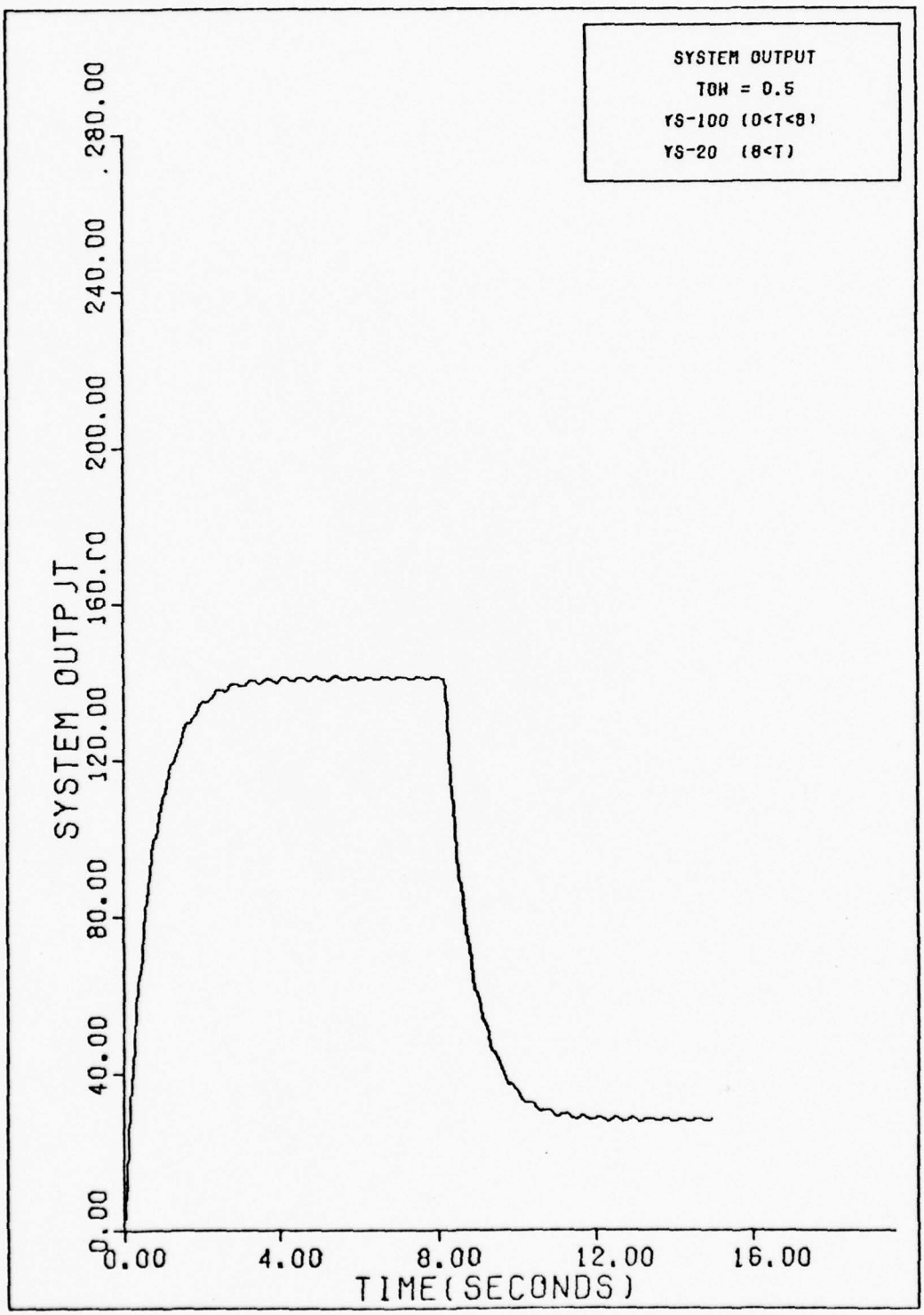


Figure 29. System Output with 5.0% Eigenvalue Change

Introduction of Control Input Limiter

To illustrate the effect of a limiter on the allowable input values, an arbitrary value, less than the steady-state value indicated in previous runs, was specified. As expected in the case where $DT = \pi/\omega$ (and control inputs never exceeded the steady-state value in the first place), the effect was that the system did not reach the high valued set point ($y_s = 100$), but attained and maintained a value corresponding to the steady-state value of the control applied. This is illustrated in Figures 30 and 31.

In a regression to the case where DT = 0.08 (oscillatory inputs and outputs as previously shown in Figures 4 and 5), applying an input limiter had the effect of actually improving the output response at the $y_s = 100$ set point. This is shown in Figures 32 and 33. However, there was no apparent effect on the output at the $y_s = 20$ point.

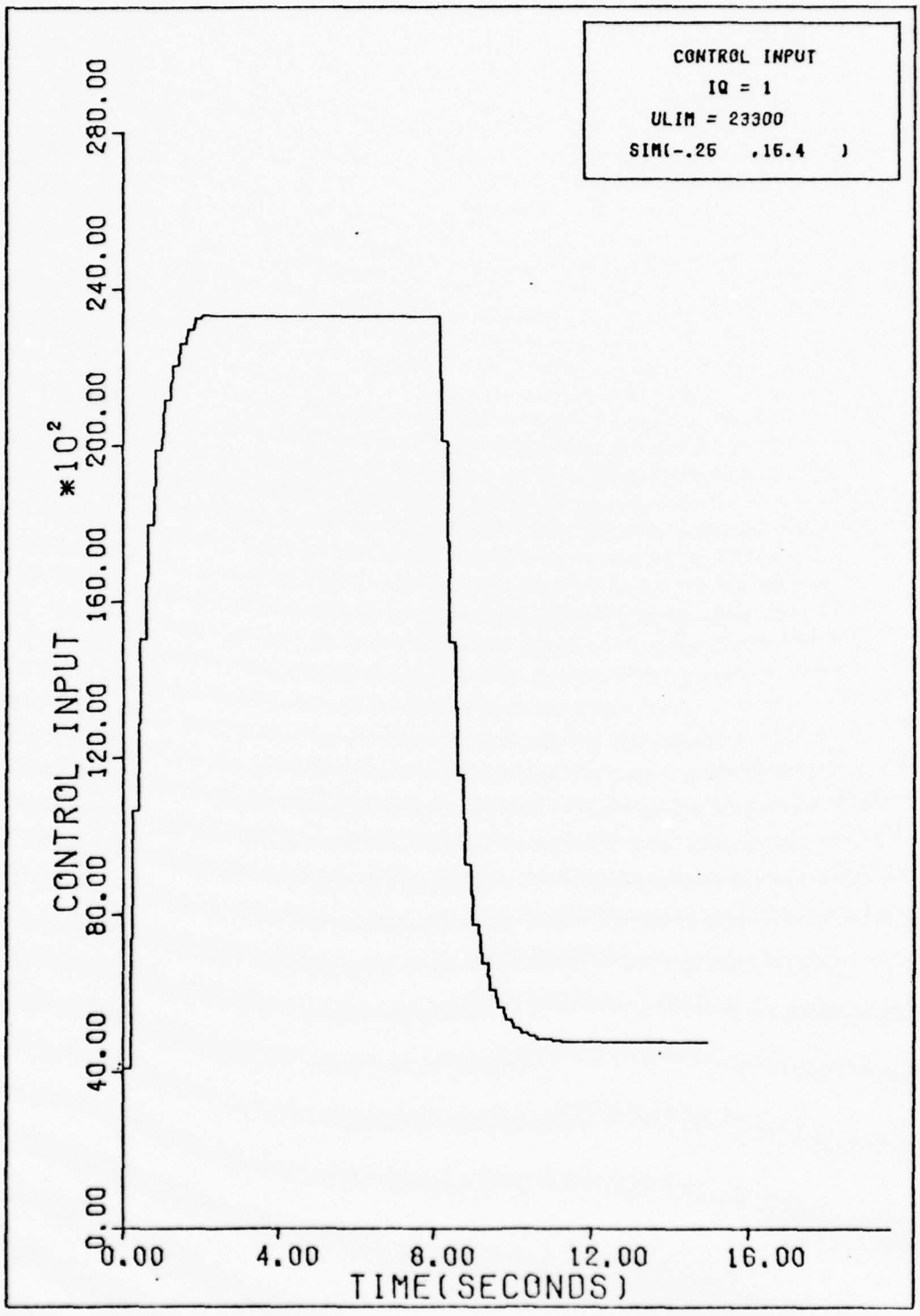


Figure 30. Control Inputs Limited to 23,300 (DT = π/ω)

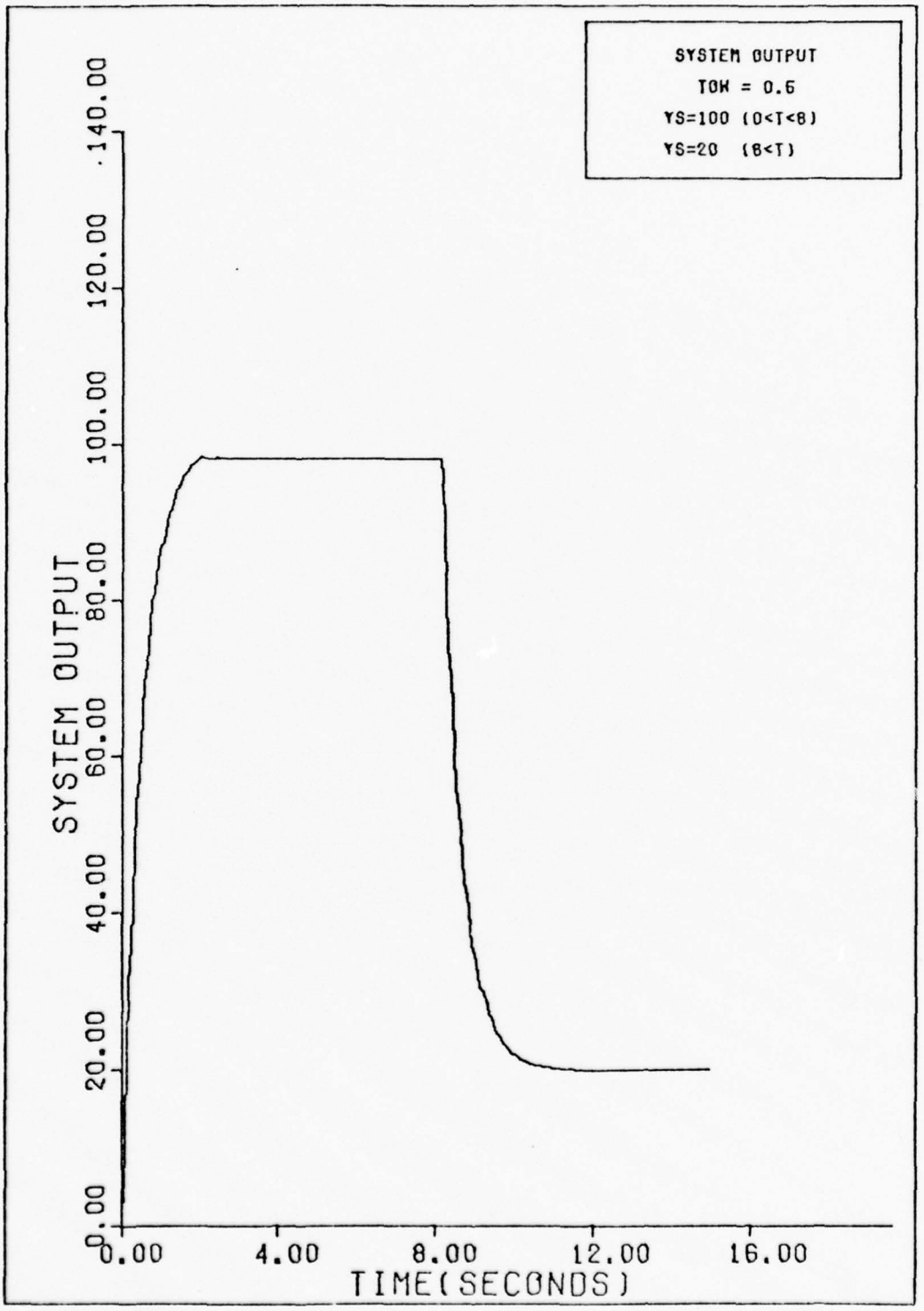


Figure 31. System Output with Control Input Limited to 23,300 (DT = π/ω)

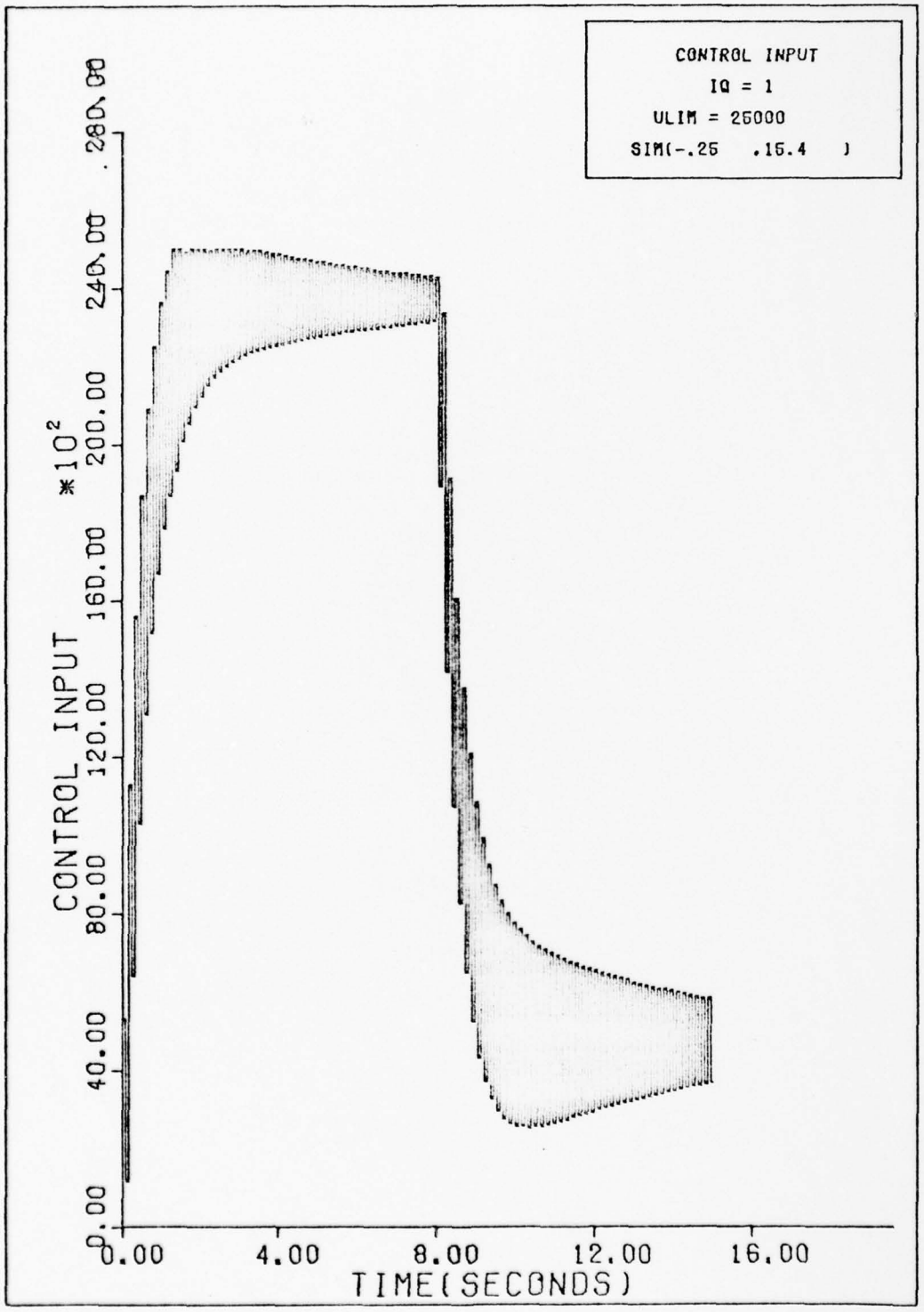


Figure 32. Control Inputs Limited to 25,000 (DT = 0.08)

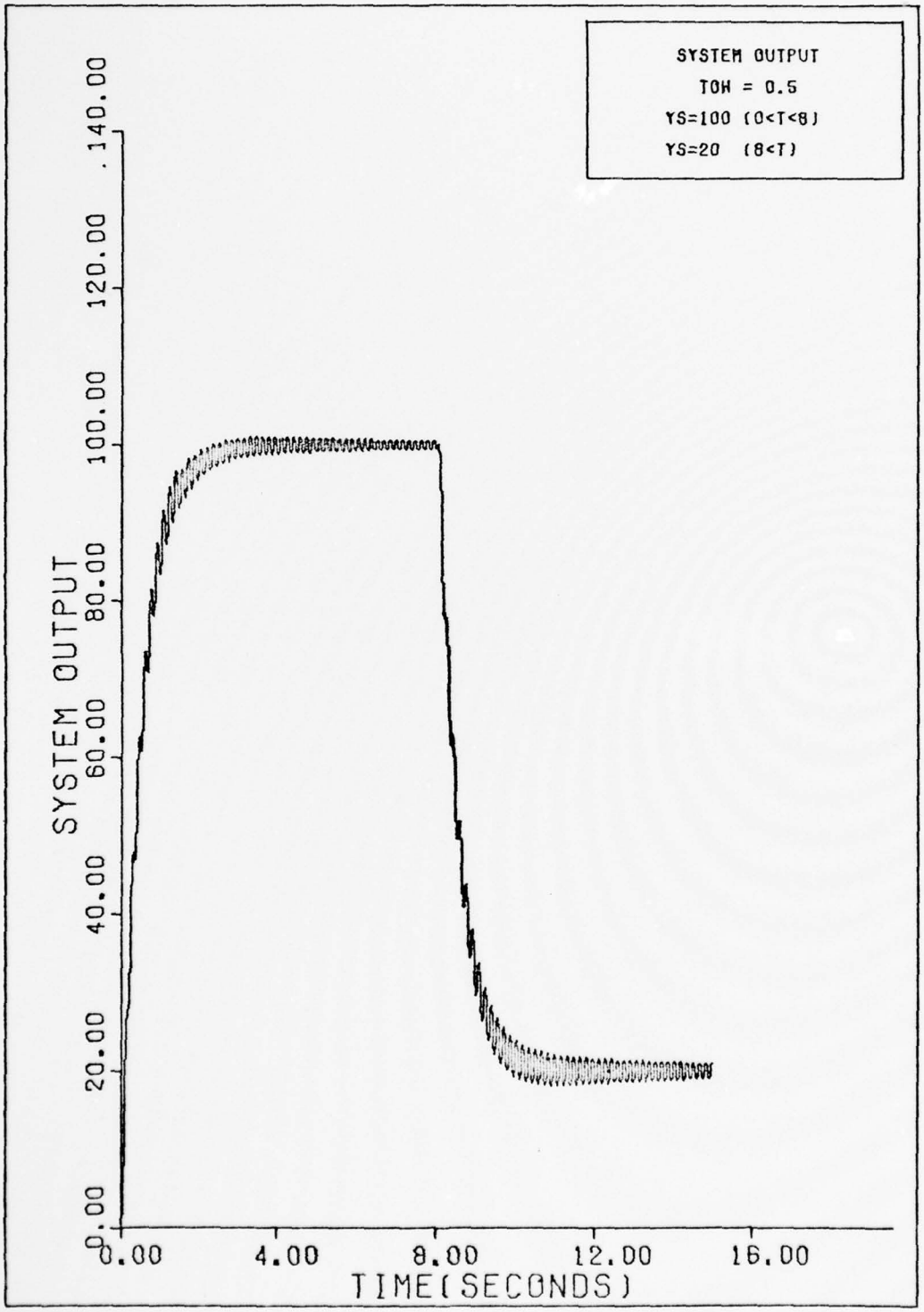


Figure 33. System Output with Control Input Limited to 25,000 (DT = 0.08)

Variation of the First-Order Trajectory Time Constant

To illustrate the effect of changing the time constant (TOW) of the first-order model, a new value of TOW = 0.1 was selected. This had a predictable effect in that the system merely attained its set points faster. As seen in Figures 34 and 35, there was no noticeable system response degradation with the faster time constant specification. With a limiter applied (Figures 36 and 37), as in the case where TOW = 0.5, the system attained a value lower than the high set point. This occurred because the system required a steady-state value of the input to obtain a desired steady-state output, and since the maximum output input was limited, the output did not reach the desired output value.

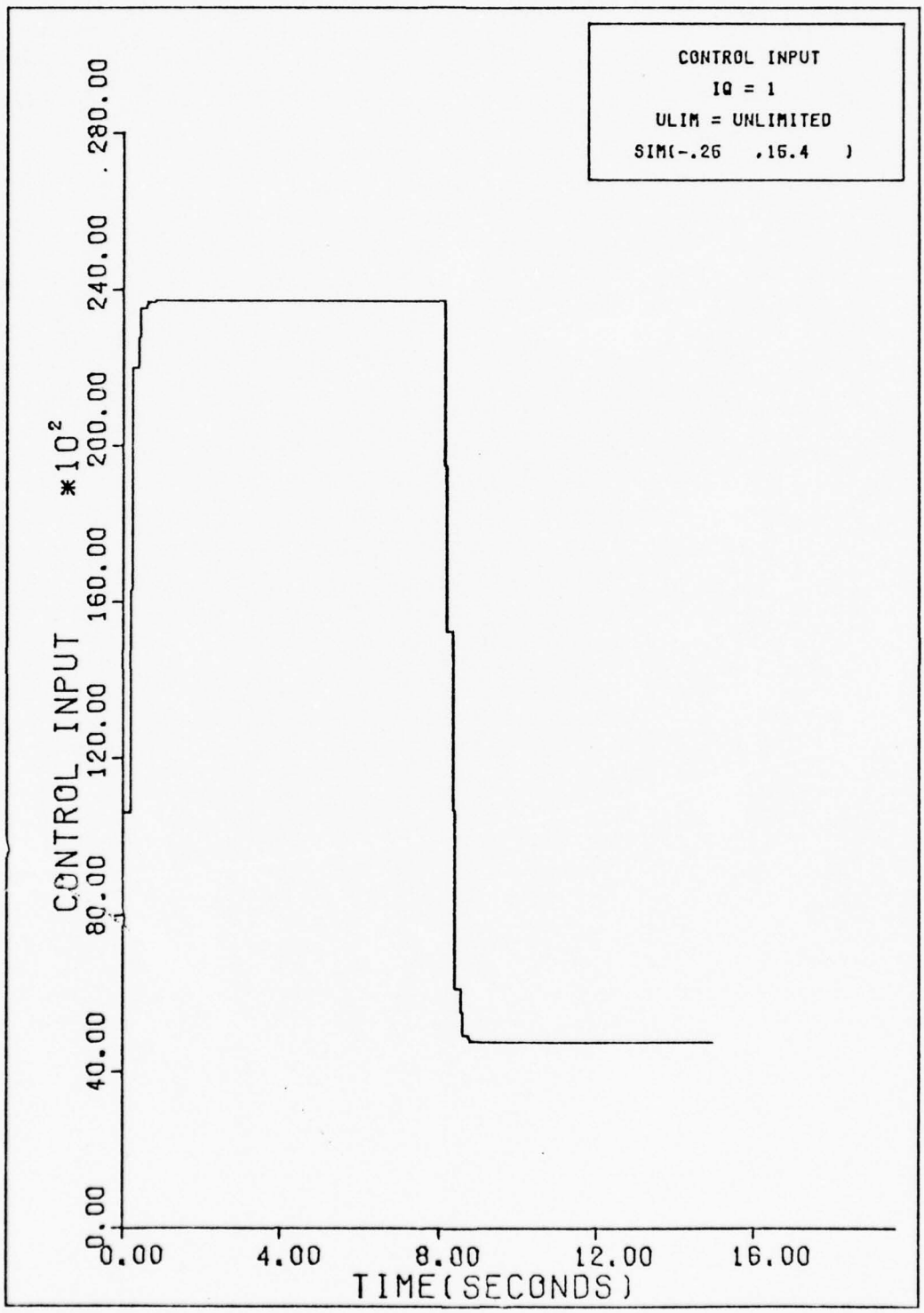


Figure 34. Control Input with TOW = 0.1 and DT = π/ω

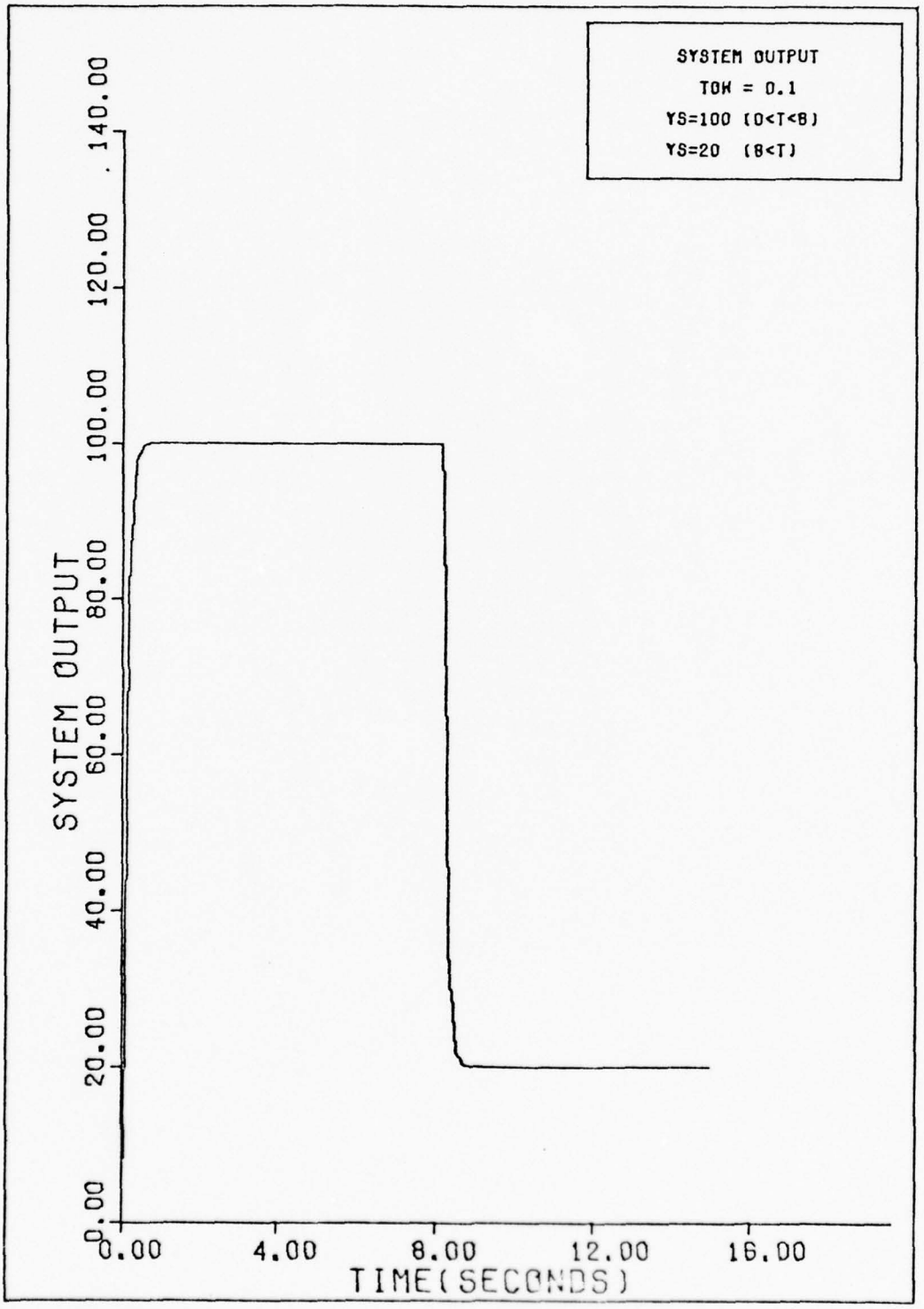


Figure 35. System Output with TOW = 0.1 and DT = π/ω

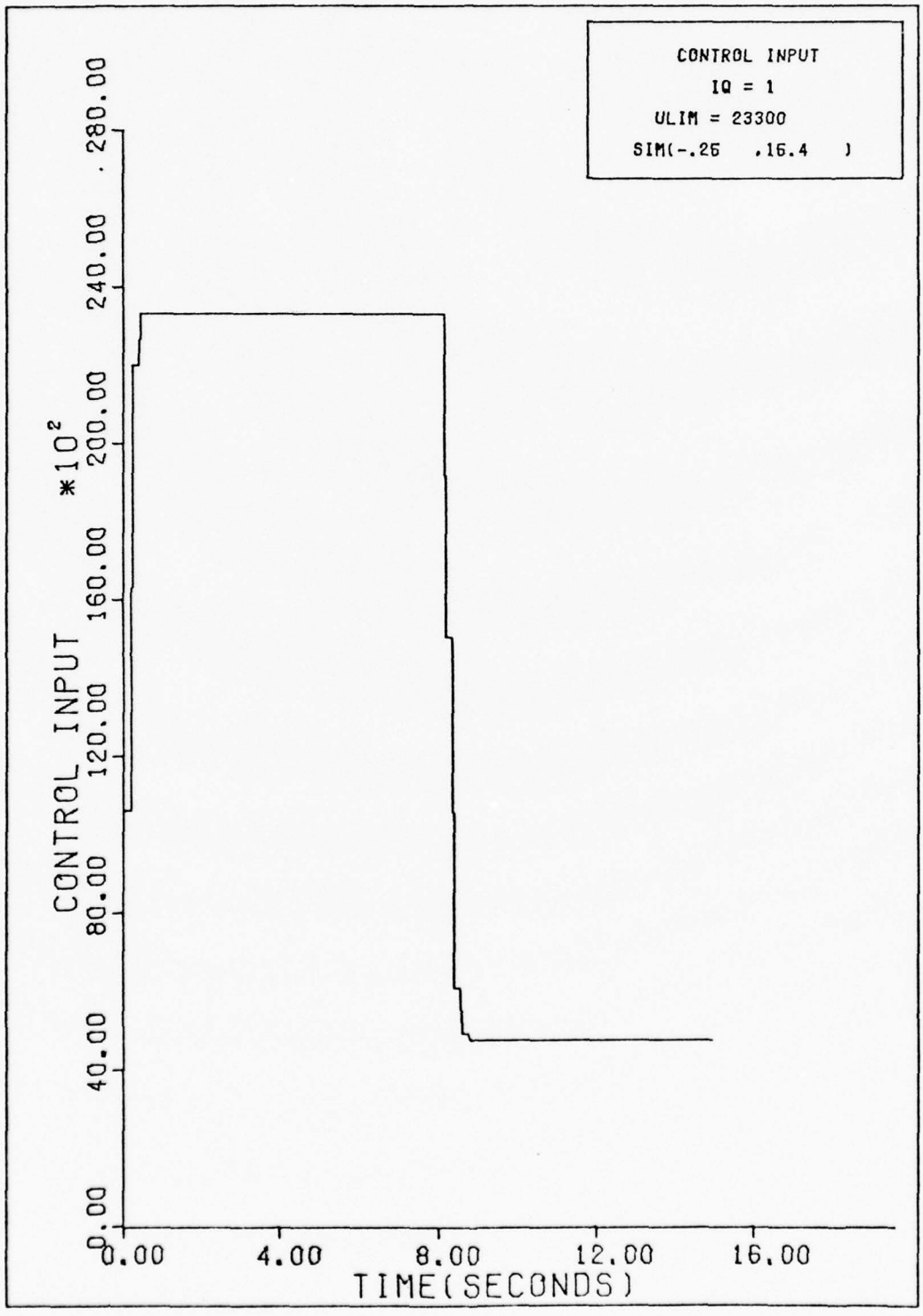


Figure 36. Control Input with ULIM = 23,300, TOW = 0.1 and DT = π/ω

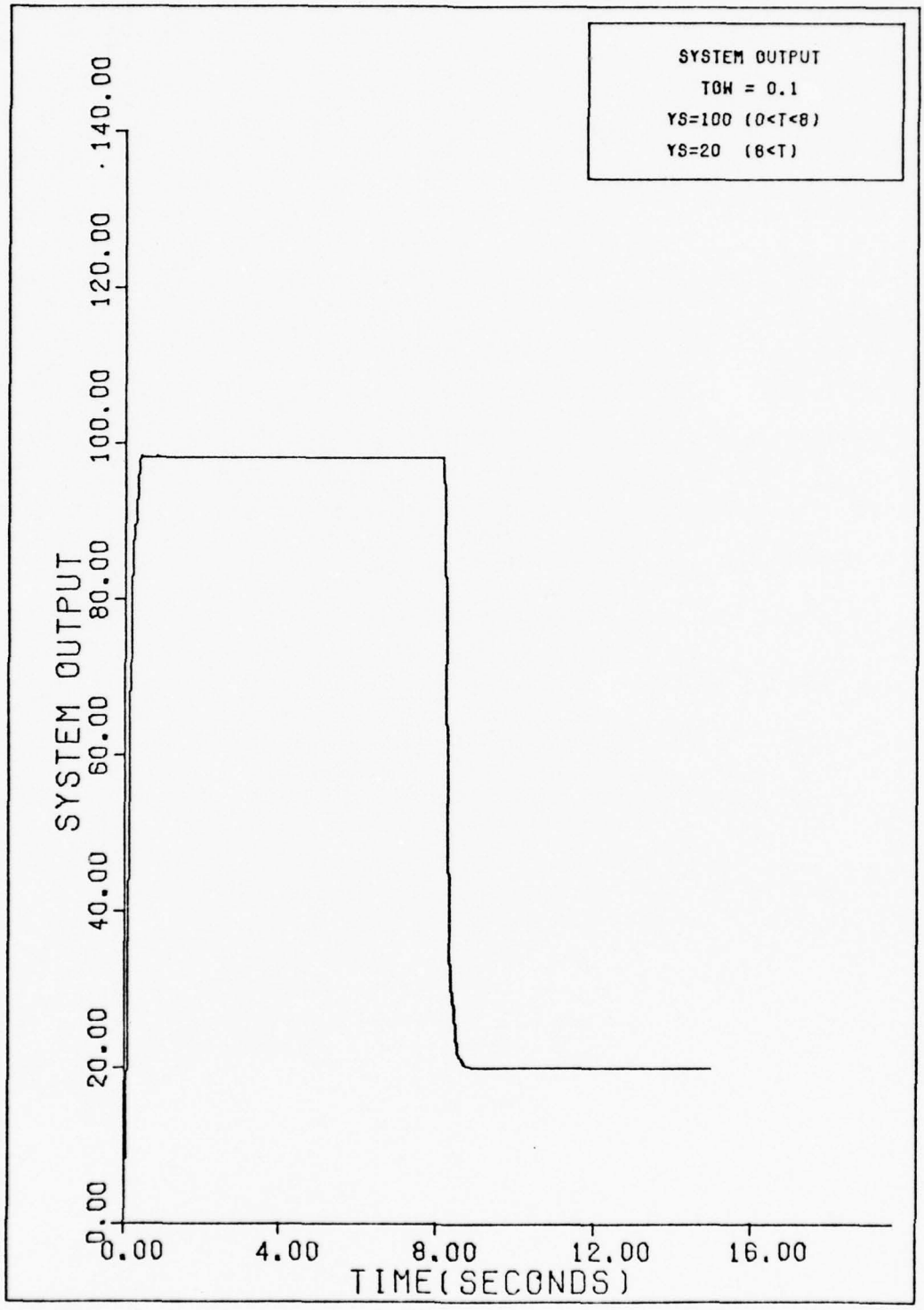


Figure 37. System Output with ULIM = 23,300, TOW = 0.1 and DT = π/ω

The Third-Order Model

Section III describes the change to the system model made by adding a real eigenvalue. The results of this change are shown in Figures 38 through 40. In these results, only the optimum value of DT determined in the preceding second-order study was used. As seen in these figures, the real root dominated the discrete impulse response function. Additionally, a re-introduction of system output oscillation is noted. No attempts were made to determine an optimum DT for this case. Instead, the order of the model was increased to four, and the result of using DT as previously determined was observed.

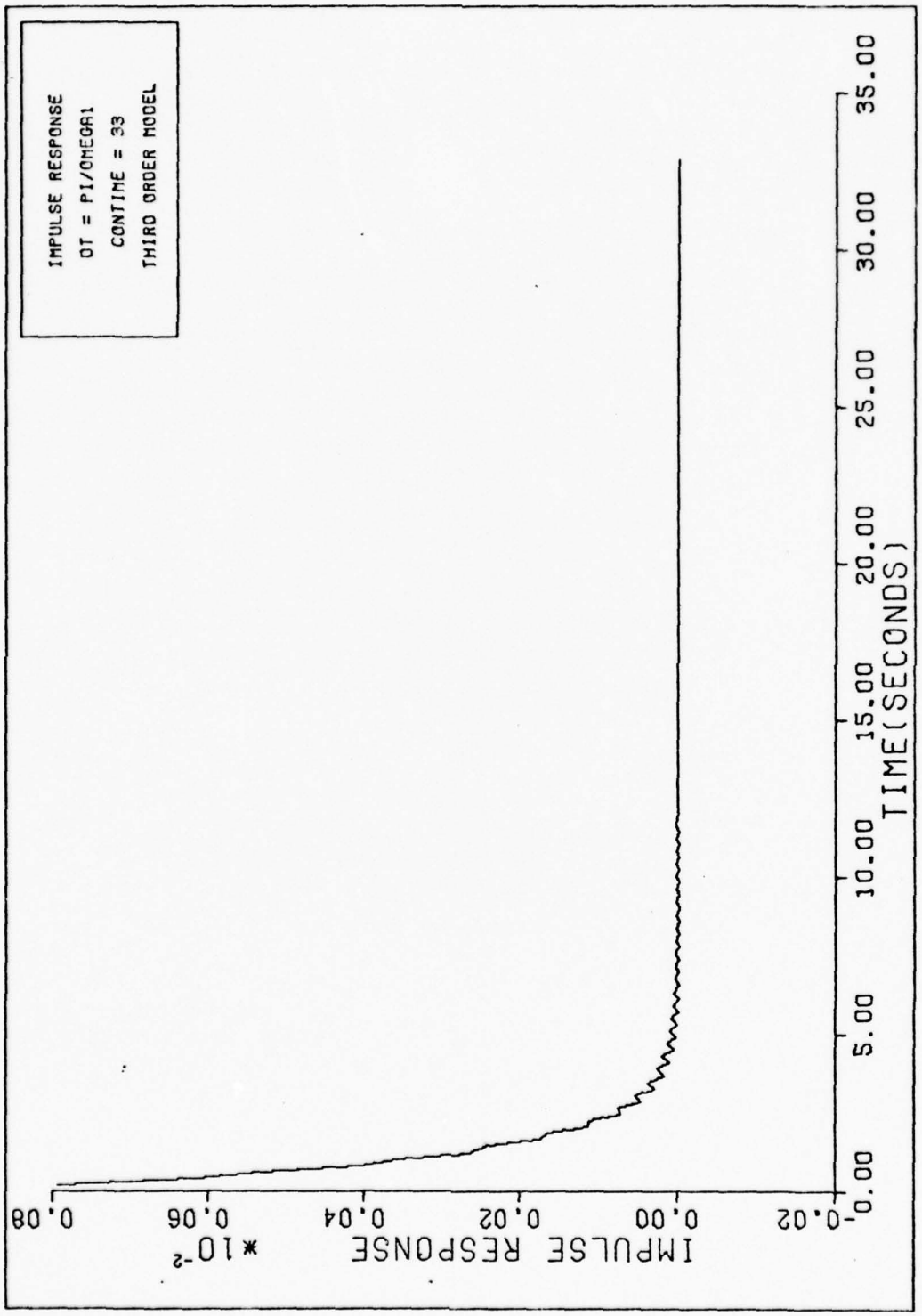


Figure 38. Discrete Impulse Response of Third Order Model (DT = π/ω)

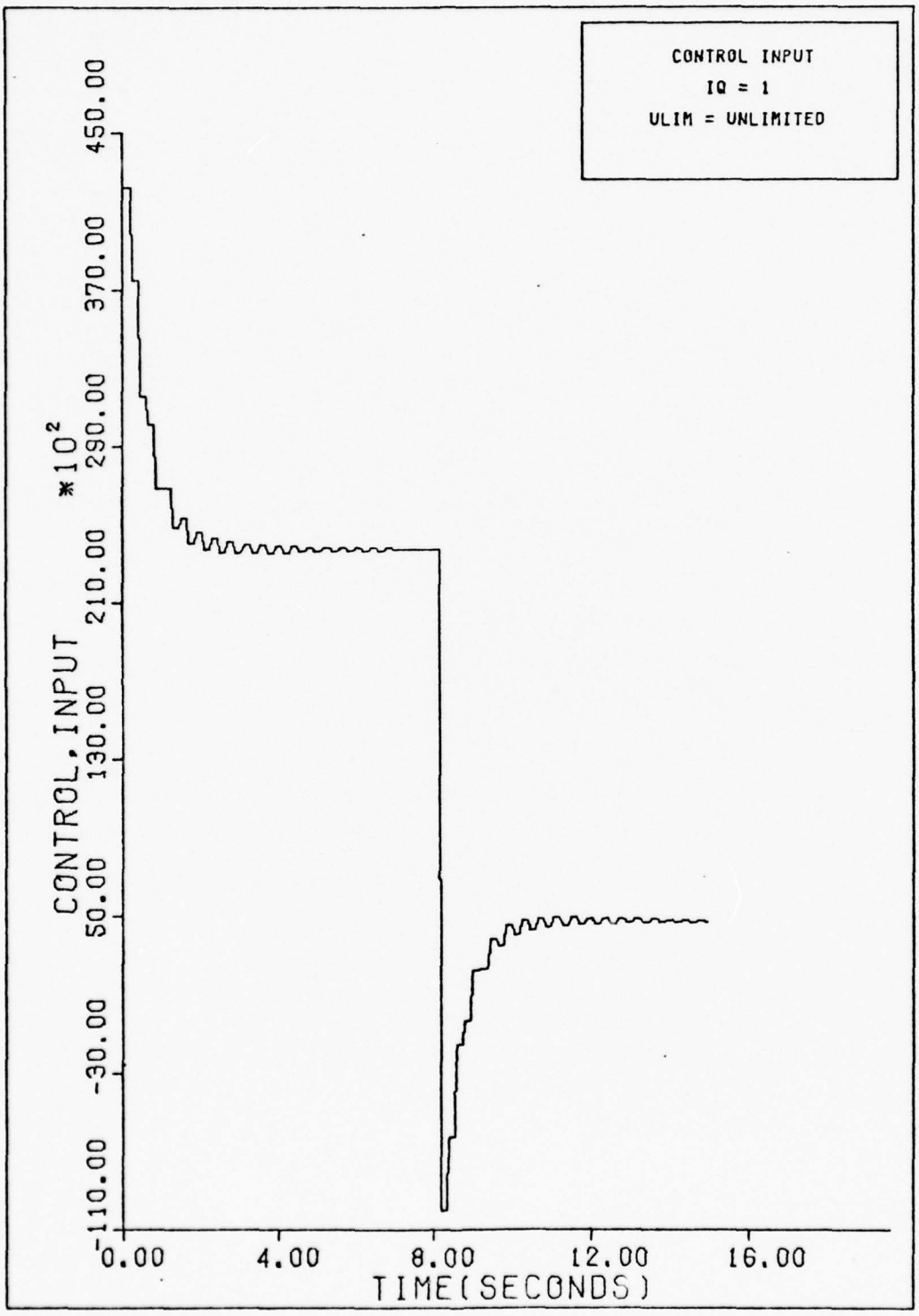


Figure 39. Control Input for Third Order Model

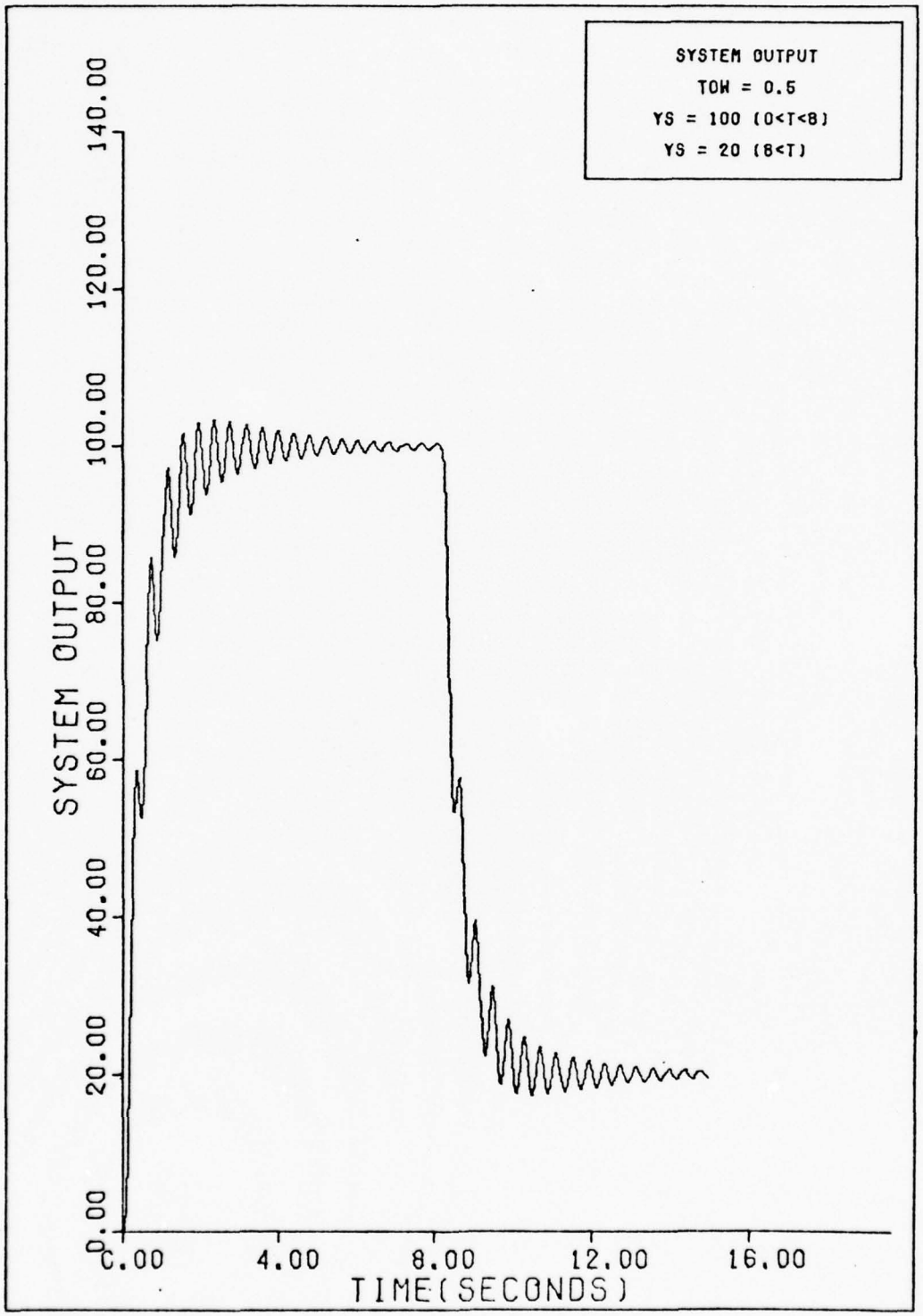


Figure 40. System Output for Third Order Model

The Fourth-Order Model

The effects of increasing the system model order to four are shown in Figures 41 through 46. In Figures 41 through 43, the radian frequency used in the calculation of DT was ω_1 = 15.4 rad/sec, corresponding to the original set of complex eigenvalues. Figures 44 through 46 show the results of using the radian frequency of the other eigenvalue set, ω_2 = 6.0 rad/sec. In the first case, the discrete impulse response appears to display more information than the latter case; however, using the first case inputs (at DT = π/ω_1 intervals) results in driving the system unstable, whereas using the latter case imputs (at DT = π/ω_2) results in more acceptable performance. Further attempts at finding an optimum DT interval were attempted but proved to be unsuccessful.

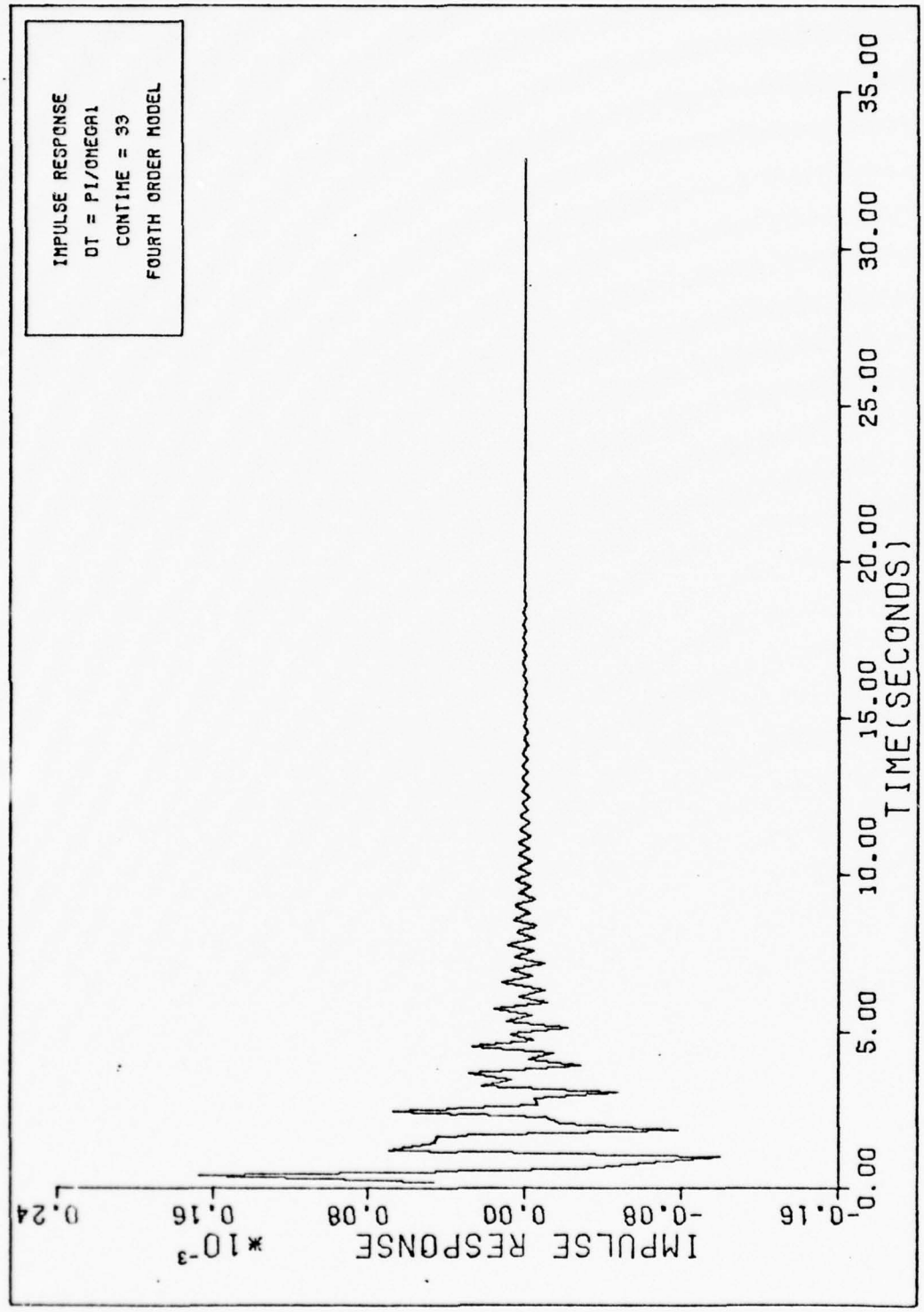


Figure 41. Discrete Impulse Response of Fourth Order Model (DT = π/ω)

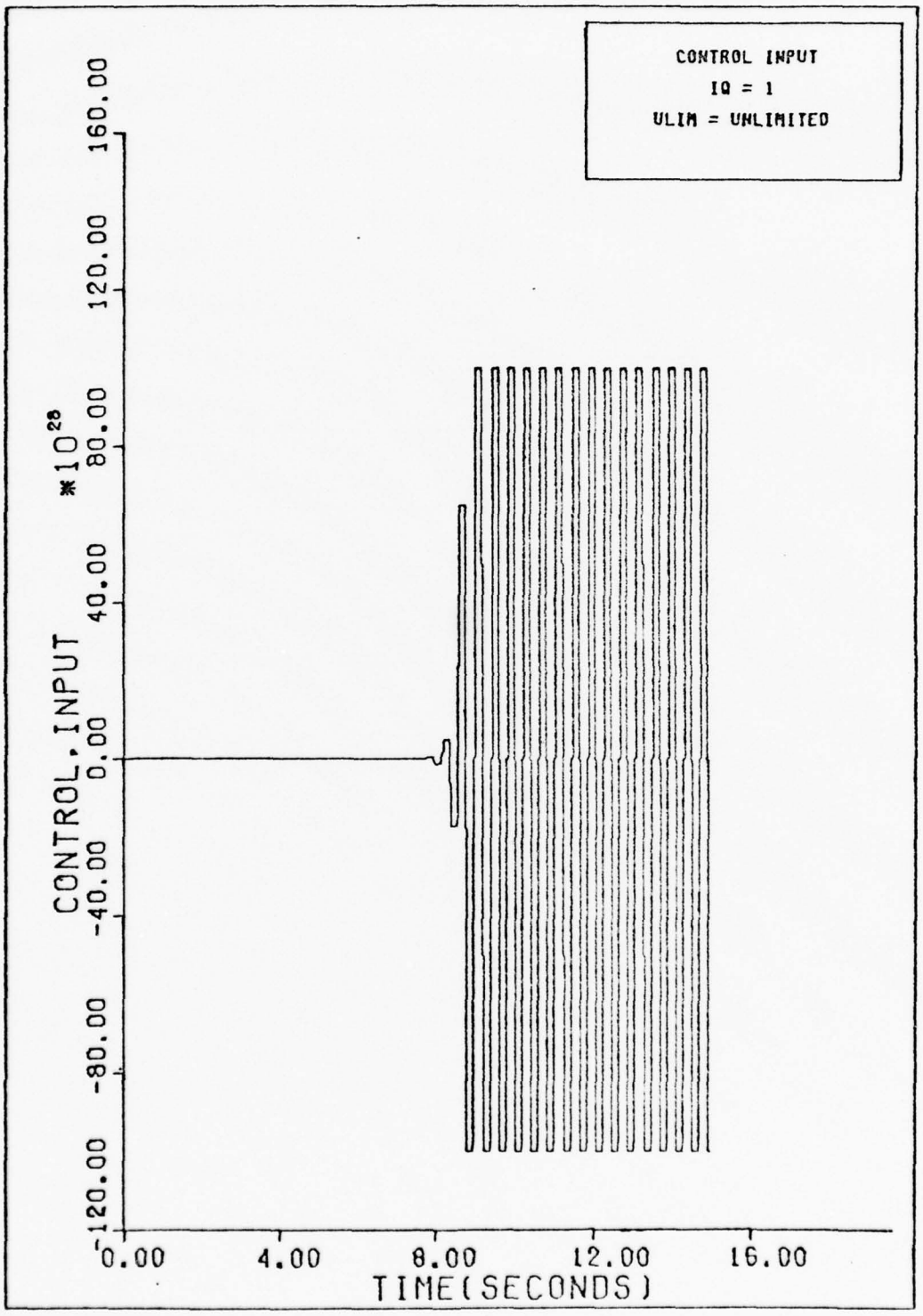


Figure 42. Control Input for Fourth Order Model ($DT = \pi/\omega_1$)

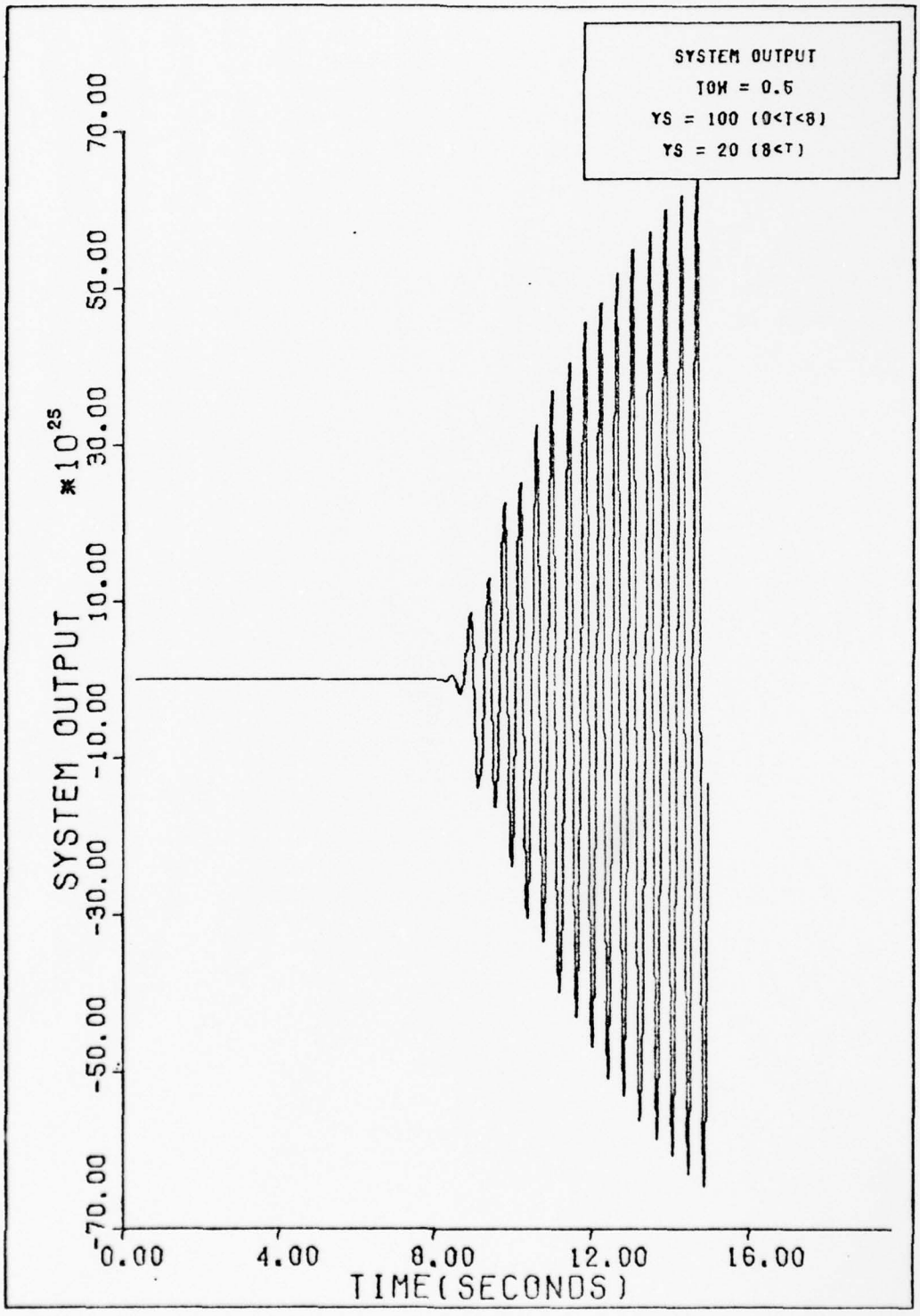


Figure 43. System Output for Fourth Order Model (DT = π/ω_1)

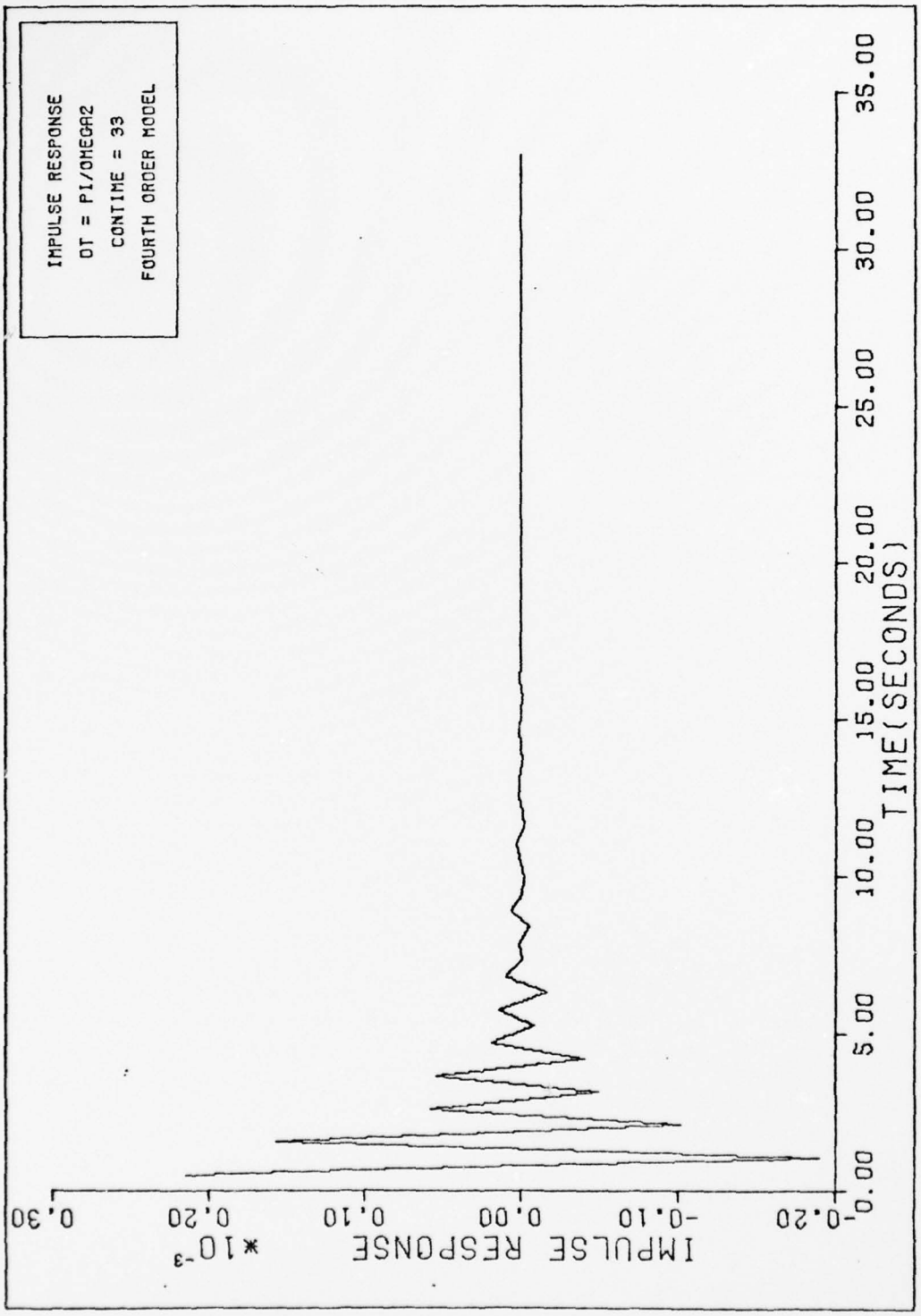


Figure 44. Discrete Impulse Response of Fourth Order Model (DT = π/ω_2)

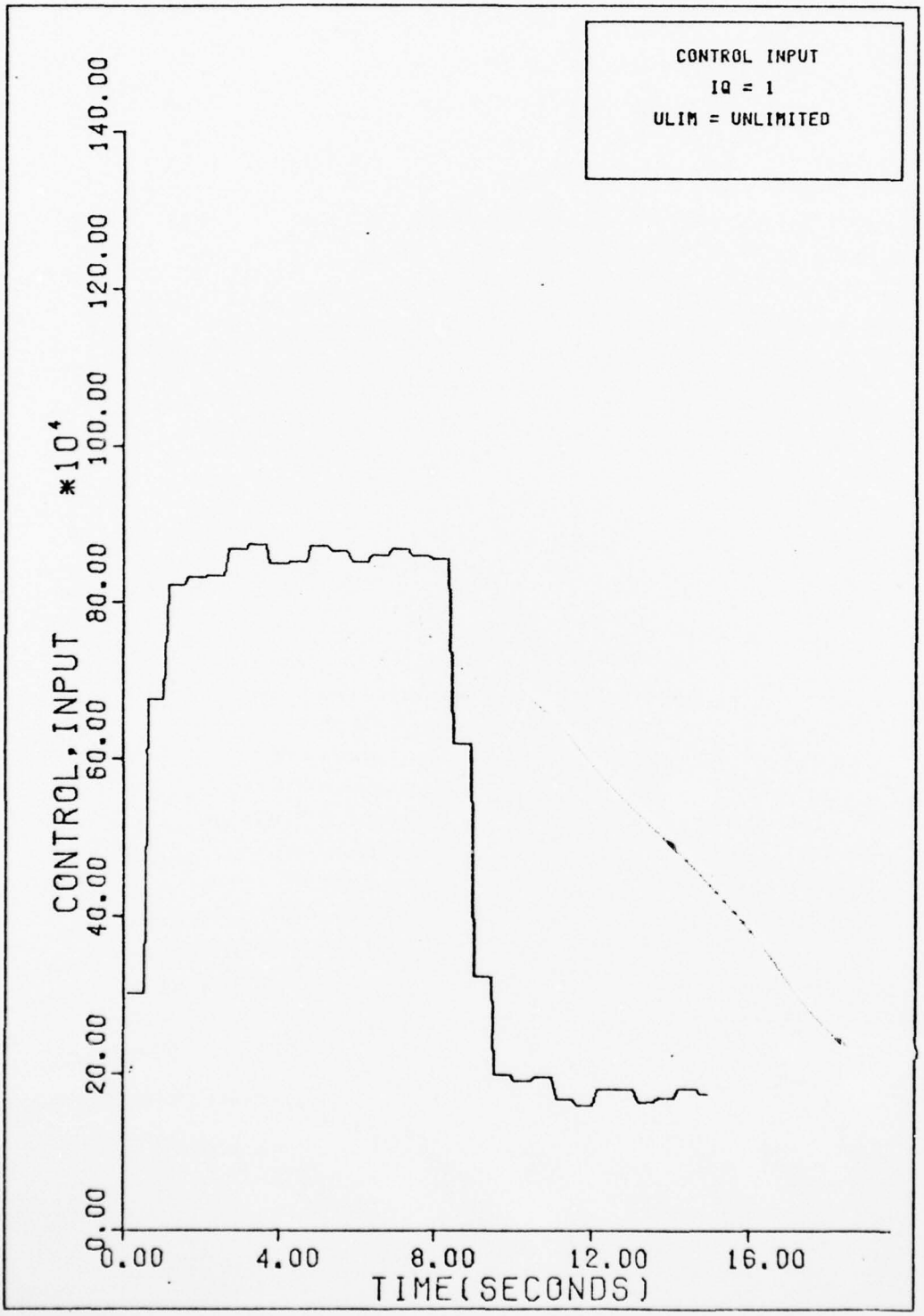


Figure 45. Control Input for Fourth Order Model (DT = π/ω_2)

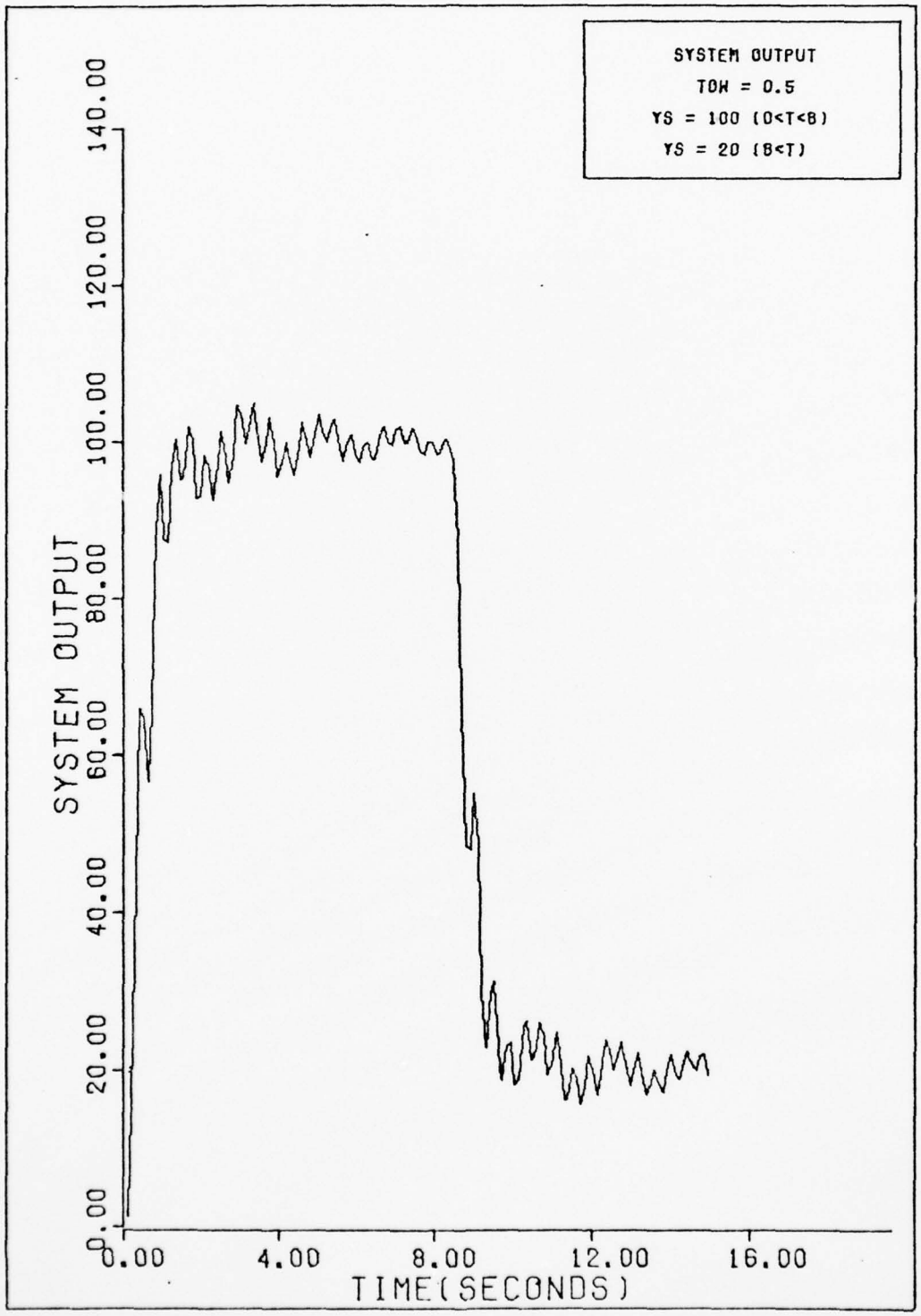


Figure 46. System Output for Fourth Order Model (DT = π/ω_2)

V Revised Algorithm

Since the results in Section IV indicated that the MAC algorithm as implemented is highly sensitive to parameter changes (which contradicted the results obtained by Mehra, et al), it was felt that a modified algorithm more closely approximating the IDCOM algorithm (Ref 1, 2, and 3) could alleviate the problem. It was hypothesized that an algorithm which recomputed the required inputs after several sampling periods had elapsed could be a plausible solution. In other words, in the initial realization of the MAC algorithm, the controls computed were those for which a unique solution existed at discrete points in time (corresponding to sampling times). In that particular realization, there was no "control" over the internal states of the system in-between those sampling times. (See Figure 47). The only constraint was that the output equalled the desired output at each discrete sampling time.

In the revised philosophy, it was hypothesized that more control over the system could be attained in between sample points, if some relaxation of the requirement to satisfy the first-order trajectory exactly was allowed. This would amount to specifying values of outputs between sample times, which would no longer produce a linear problem for which a unique solution exists, but in fact, an overdetermined system for which only a "best" fit solution could be obtained (See Figure 48). The least squares technique was chosen as the method to produce a solution yielding the best fit of the output values to the reference trajectory. Additionally, this type of solution would make it possible to assign a cost function or weighting matrix to specify

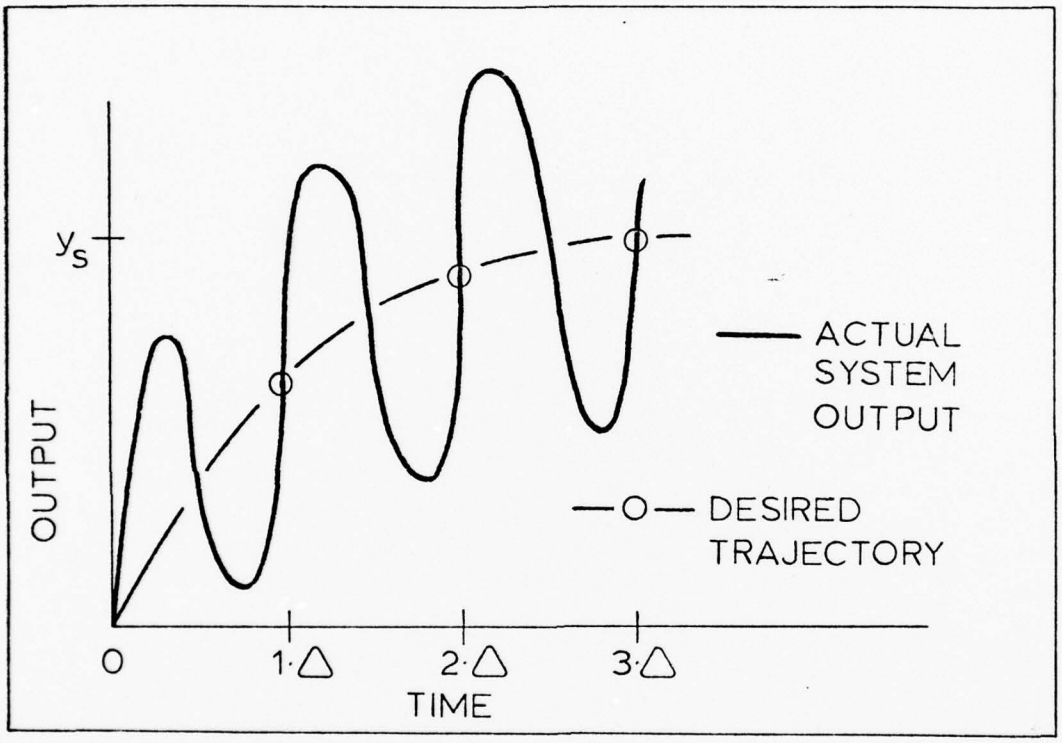


Figure 47. Output for Unique Solution Case

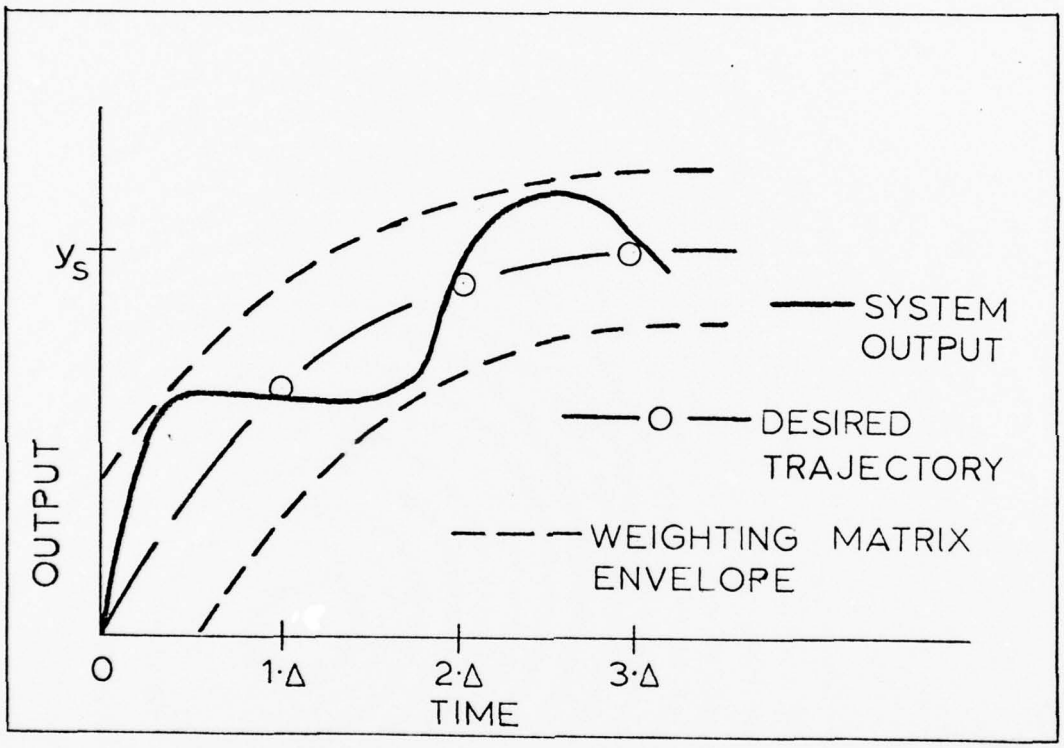


Figure 48. Output for Best Fit Case (Uniform Weighting)

how closely the future trajectory was to be tracked. For example, Figure 48 shows the envelope of acceptable output values for one choice of weighting matrix, where tracking of the first-order trajectory was of uniform importance. Figure 49 shows the effect of specifying a weighting matrix to more closely track the first-order trajectory nearer the end of the interval of interest than at the beginning. Figure 50 illustrates a situation where a weighting matrix was specified so that the first-order trajectory was tracked more closely at the beginning than at the end of the interval of interest. This approach would also have merit since a smaller sampling time could be selected to produce a more accurate discrete impulse response, yet a longer time between control inputs could be chosen so that rapidly changing inputs could be avoided.

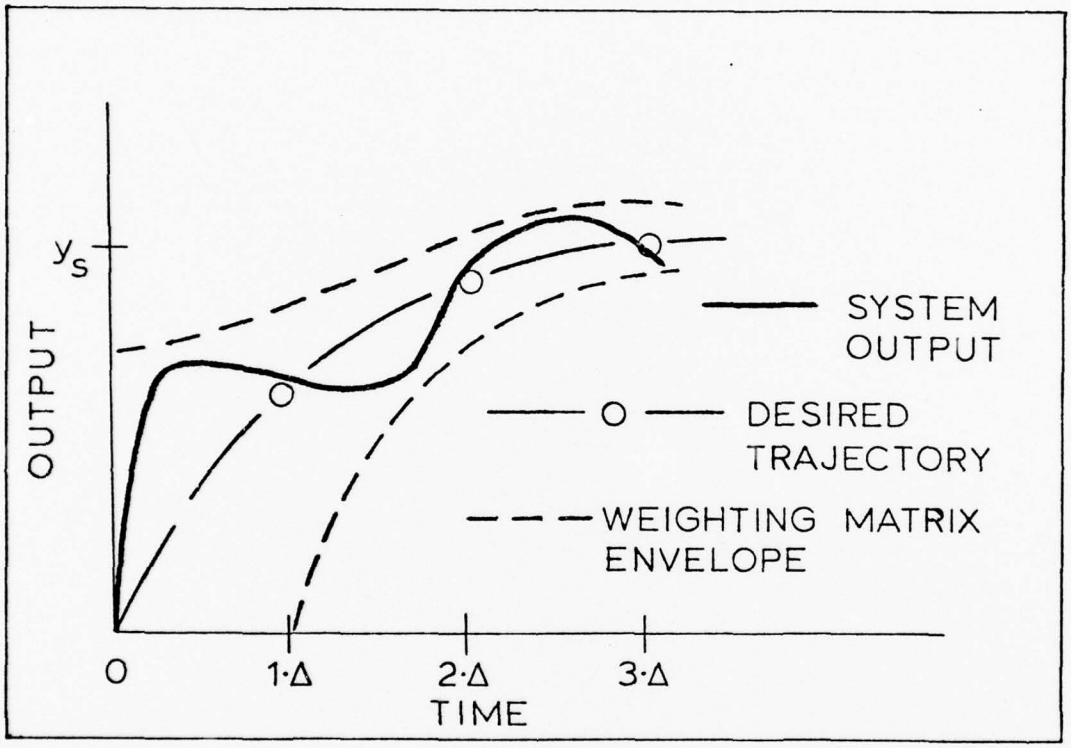


Figure 49. Output for Best Fit (Closer Fit at End of Interval)

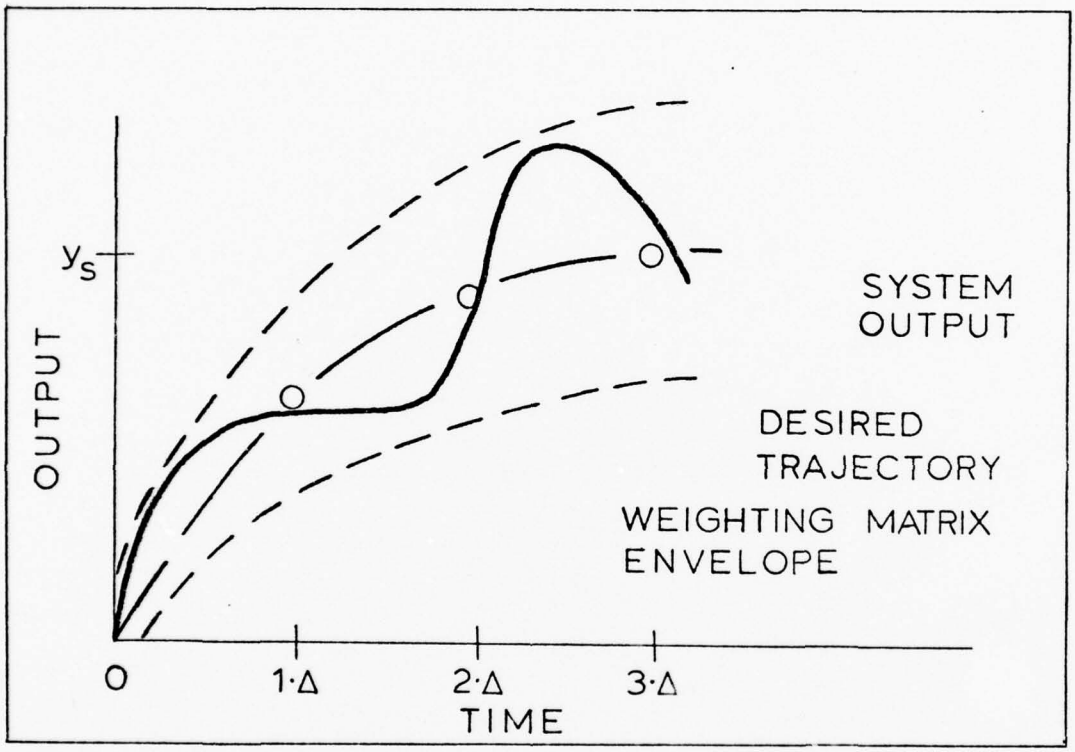


Figure 50. Output for Best Fit (Closer Fit at Beginning of Interval)

VI Conclusions and Recommendations

This section contains the resultant conclusions drawn from this particular study of Model Algorithmic Control (MAC) and some recommendations for further study in this area.

Conclusions

In this implementation of MAC, the selection of the sampling time (DT) became a critical element in producing satisfactory results. The selection process for DT tended to be one with conflicting requirements: a small DT was desired so that an accurate discrete impulse response representation could be obtained, but a large DT was desired to avoid high frequency control inputs (since controls were applied each DT second). An optimum DT for the second-order model hypothesized in this thesis was obtained where $DT = \pi/\omega$, or the Shannon Sampling period. At this point it should be pointed out that the selection of $DT = \pi/\omega$ produced, through the deconvolution process, a "clean" impulse response representation (See Figure 12) containing the minimum essential information for applying the control algorithm successfully. Additionally, this particular value of DT allowed controls to be applied at a frequency on the order of the natural system frequency. Unfortunately, a change of the system model to a third and then a fourth-order system produced oscillatory results at that particular selection of DT. After some preliminary searching for appropriate values for DT in higher order models, it was found that a simple relationship (such as that which applied in the second-order model) was not immediately apparent.

Another factor in producing satisfactory results was the selection of enough storage locations (N) for the discrete impulse response function. This value is directly related to the value chosen for DT through the relationship CONTIME = N·DT, where CONTIME represents the time interval over which the impulse response is computed. From the results of Section IV, it appears that this interval should contain at least enough impulse response information to correspond to the time it takes for the exponential part of the time function to decay to approximately 0.1% of its maximum value.

Specification of values for the control input limiter (ULIM) and the first-order model time constant (TOW) produced predictable results in the case where the best model following occurred (DT = π/ω). In this situation, the selection of ULIM prevented the system from attaining its maximum desired set point. Selecting a faster time constant (TOW) merely produced a faster system response.

Apparently, from the results of Section IV, the ability of the controller to follow a specified (first-order) trajectory accurately is very dependent upon an accurate representation of the system model. In fact, if eigenvalues of the system model and those of the "actual" system differ by more than 1.0%, the approach yields a steady-state error greater than 6%. A significant point about these results is the following:

Mehra, et al (Ref 1:3) hypothesized that the robustness of IDCOM and the MAC control philosophy may be attributed to using the impulse response function for output prediction. They suggest that the nonminimality of this description may be a key factor which contributes to the robustness.

However, these results would indicate that the robustness is not attributed to the impulse response prediction technique, but rather must be attributed to the control algorithm as they have implemented it. The preceding section outlined the changes that were begun but not completed on the MAC algorithm. Upon successful implementation of these changes, the modified algorithm would more closely resemble what they have implemented. It is felt that if these changes were successfully implemented, the eigenvalue sensitivity problem would be solved, at least in part. This hypothesis cannot be tested until the modified program has been fully checked out. It is, however, apparent that the robustness property does not evolve from the impulse response function alone.

Conceptually, the MAC approach is intuitively pleasing in that once the "ideal" controlling program is developed, it is necessary to specify certain parameters. Then the model following controller computes and applies the inputs required to cause the system to follow the specified trajectory and to meet whatever specifications are inherent in the parameter specification. However, the key word "ideal" implies a considerable amount of sophistication and futher development than exists in the implementing program used in this thesis. As indicated in the Introduction, however, the MAC algorithm used in this study was a simplified form of the MAC algorithm used by IDCOM; therefore, it bears re-emphasizing that this study should not be interpreted as casting unfavorable reflections on the ADERSA/GERBIOS algorithm.

Recommendations for Further Study

Some of the conclusions in the preceding section point to areas in this MAC study which could be expanded further. This sub-section

covers those areas and also some areas the writer wished to explore had there been more time.

Essentially, the controlling program used in this thesis demonstrated only the very basic principles behind the MAC concept and how they worked in a very specific case: that of a SISO, second-order, lightly damped system. To fully exploit the MAC concept, the controlling program should be made considerably more sophisticated and generalized, in line with the IDCOM program's level of sophistication. Suggested areas for development would be:

1. Continuing the investigation into the feasibility of recomputing required control inputs after several sampling periods have elapsed. As alluded to in the conclusion, this may be the answer to the problem of eigenvalue sensitivity, and perhaps the solution to the other problem regarding the selection of appropriate values of DT.

2. Incorporating the on-line identification feature. This should also help take care of the eigenvalue sensitivity problem addressed above.

3. Investigating the effects of noise on the model-following controller.

4. Extending the controlling program to a multiple-input multiple-output capability.

5. Investigating the possibility of having the sampling rate determined automatically.

6. Investigating the controller's performance when used on a non-linear system.

Bibliography

1. Mehra, R. K.; Kessel, W. C.; Rault, A.; Richalet, J.; and Papon, J. "Model Algorithmic Control Using IDCOM for the F100 Jet Engine Multivariable Control Design Problem," paper presented at the International Forum on Alternatives for Multivariable Control, Chicago, Illinois, 13-14 October 1977.

2. Richalet, J.; Rault, A.; Testud, J. L.; and Papon, J. "Model Predictive Heuristic Control: Applications to Industrial Processes," paper for ADERSA/GERBOIS, 53 avenue de 1'Europe, F78140 Vélizy France.

3. Mereau, P.; Guillaume, D.; and Mehra, R. K. "Flight Control Application of Model Algorithmic Control with IDCOM (Identification and Command)," pre-print of paper to be presented at the 1978 Decision and Control Conference to be held January 1979 at San Diego, California.

4. Hotz, Anthony F. "B-52E CCV Flutter Mode Control System Analysis Using Level 2.02 FLEXSTAB," AFFDL Technical Memorandum TM-77-46-FGC. Wright-Patterson AFB, Ohio: Air Force Flight Dynamics Laboratory, 1977.

5. Reid, J. G Course notes for EE 5.10, Linear Systems Theory and Digital Computation Methods. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, October 1978.

6. Houpis, Constantine H. and G. B. Lamont. Lecture Notes on Digital Control Systems/Information Processing. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1977.

AD-A064 222 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 1/3
APPLICATION OF MODEL ALGORITHMIC CONTROL TO A LIGHTLY DAMPED SI--ETC(U)
DEC 78 H J COLSON

UNCLASSIFIED AFIT/GE/EE/78-21 NL

2 OF 2
AD A064222

END
DATE FILMED
4-79
DDC

Appendix A

The FORTRAN source code listed in this appendix is the MAC implementation for a second-order system model. To run the program, the user must first specify the variables listed in the initial comments section. The data cards at the beginning of the program are for Calcomp plot titles only and as a consequence may be left unspecified if the user so desires. Once the appropriate variables are specified, the program may then be compiled and executed.

```
      PROGRAM STMCONT(INPUT,OUTPUT,PLOT)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     THIS PROGRAM COMPUTES THE DISCRETE IMPULSE RESPONSE
C     FUNCTION FOR A SPECIFIED SINGLE INPUT / SINGLE OUTPUT
C     (SISO) SYSTEM.  THE RESULT IS THEN USED BY MODEL
C     ALGORITHMIC CONTROL (MAC) TO DETERMINE THE CONTROLS
C     REQUIRED TO FOLLOW THE DESIRED OUTPUT.  OUTPUT DATA
C     CONSISTS OF PRINTOUTS AND PLOTS FOR VALUES OF DISCRETE
C     IMPULSE RESPONSE (H-DECONVOLUTION), COMPUTED CONTRO-
C     INPUTS, AND ACTUAL OUTPUT (SAMPLED AT ONE-FIFTH THE
C     DISCRETE TIME INTERVAL USED IN DECONVOLUTION).  USER
C     SUPPLIES THE FOLLOWING INFORMATION:
C               TOW = DESIRED TIME CONSTANT OF
C                     FIRST ORDER SYSTEM MODEL
C               IQ = NUMBER OF FUTURE CONTROLS
C                    TO BE COMPUTED EACH CYCLE
C               YS = SET POINTS OF FIRST ORDER
C                    SYSTEM MODEL
C               CONTIME = DESIRED DURATION OF DECONVOLUTION
C               SIGMA = REAL PART OF EIGENVALUE
C                       FOR SECOND ORDER SISO
C                       SYSTEM TO BE CONTROLED
C               OMEGA = POSITIVE IMAGINARY PART
C                       OF EIGENVALUE FOR SECOND
C                       ORDER SISO SYSTEM TO BE
C                       CONTROLED
C               SIMTIME = DESIRED DURATION OF
C                         SIMULATION
C               ULIM = MAXIMUM CONTROL VALUE ALLOWED
C               N = NUMBER OF INCREMENTS STORED
C                   IN DECONVOLUTION
C               DT = DISCRETE TIME INCREMENT USED
C                    IN DECONVOLUTION
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      DIMENSION H(302),YZI(5),YD(5),Z(5),UF(5),UP(302)
      DIMENSION X(2),U(302),TA(302),Y(302),TAO(2102)
      DIMENSION IDU(17),UA(2102),FEE(2,2),DELAH(2),DX(2)
      DIMENSION IDH(17),IDY(17),YA(2102)
      COMMON L1,V
      DATA IDH(1)/20H  IMPULSE RESPONSE  /
      DATA IDH(3)/20H    DT = PI / OMEGA /
      DATA IDH(5)/20H CONTIME = 33 SEC   /
      DATA IDH(7)/20H MODEL: 1(-0.25,15.4)/
      DATA IDH(9)/20H   TIME(SECONDS)    /
      DATA IDH(11)/20H   IMPULSE RESPONSE /
      DATA IDH(13)/50H
     1/
      DATA IDY(1)/20H    SYSTEM OUTPUT   /
      DATA IDY(3)/20H      TOW = 0.5     /
      DATA IDY(5)/20H   YS=100 (0<T<8)   /
      DATA IDY(7)/20H   YS=20   (8<T)    /
      DATA IDY(9)/20H   TIME(SECONDS)    /
```

```
      DATA IDY(11)/20H   SYSTEM OUTPUT    /
      DATA IDY(13)/50H
     1/
      DATA IDU(1)/20H    CONTROL INPUT    /
      DATA IDU(3)/20H       IC = 1        /
      DATA IDU(5)/20H  ULIM = UNLIMITED   /
      DATA IDU(7)/20HSIM(-.25   ,15.4    )/
      DATA IDU(9)/20H   TIME(SECONDS)     /
      DATA IDU(11)/20H   CONTROL INPUT    /
      DATA IDU(13)/50H
     1/
      CALL PLOT(0.,-4.,-3) $ CALL PLOT(0.,0.13,-3).
C          INITIALIZE VARIABLES
      TOW=.5
      SIGMA = -0.25
      OMEGA = 15.4
      CONTIME = 33
      SIMTIME = 15
      IC=1
      YS=100.
      ULIM = 1.E+30
      L1 = 0
      T = 0
      M = 2
      PI=3.14159265358979
      DT = PI / OMEGA
      N = CONTIME / DT
      IF(N.GE.300) N = 300
      D=DT/5.
      IC2 = 5 * IC
      N2 = SIMTIME * 5 / DT
      ICMAX = SIMTIME / ( DT * IC ) + 5
      CALL STMLAM(FEE,DELAM,DT,SIGMA,OMEGA)
      DO 1 J=1,N
      U(J) = UI( J * DT - D )
1     CONTINUE
      DO 2 J=1,20
      X(J) = 0.
2     CONTINUE
      DO 3 J=1,N
      TA(J) = 0.0
3     CONTINUE
      DO 5 J= 1,N
      TA(J) = J*DT
5     CONTINUE
C          CALL SUBROUTINES TO CALCULATE THE DISCRETE
C     IMPULSE RESPONSE
      CALL OUTPUT(Y,X,N,DT,T,FEE,DELAM)
      CALL DECON(Y,U,M,N,DT)
      PRINT 101
      PRINT*,"  THE VARIABLES OF INTEREST FOR THIS RUN ARE:"
      PRINT*,"  N =  ",N,"  DT =  ",DT,"  TOW =  ",TOW,
      PRINT*,"  ULIM =  ",ULIM
      PRINT 101
```

```
      PRINT*,"  H - DECONVOLUTION              TIME "
      PRINT*," "
102   FORMAT(2(5X,G15.7))
101   FORMAT(1H1)
      DO 8 J = 1,N
      H(J) = H(J) * DT
8     CONTINUE
      PRINT 102,(H(I),TA(I),I=1,N)
      PRINT 101
      CALL HGRAPH(TA,H,N,IOH,1,0,1)
C          RE-INITIALIZE VARIABLES
      T = 0.0
      L1 = 1
      DO 6 J = 1,N
      UP(J)=0.
6     CONTINUE
      DO 9 J = 1,M
      X(J) = 0.0
9     CONTINUE
      Y1 = 0.0
      PRINT*,"   THE MODEL EIGENVALUES HAVE THE FOLLOWING"
      PRINT*,"REAL AND IMAGINARY PARTS "
      PRINT*," "
      PRINT*," SIGMA = ",SIGMA,"     OMEGA = +OR- ",OMEGA
      PRINT*," "
      SIGMA = -0.25
      OMEGA = 15.4
      PRINT*,"   THE SIMULATION EIGENVALUES HAVE THE "
      PRINT*,"FOLLOWING REAL AND IMAGINARY PARTS"
      PRINT*," "
      PRINT*,"SIGMA = ",SIGMA,"    OMEGA = +OR-",OMEGA
      PRINT 101
      CALL STHLAM(FEE,DELAM,0,SIGMA,OMEGA)
C          APPLY THE CONTROL ALGORITHM TO FOLLOW SPECIFIED
C          FIRST ORDER MODEL
      DO 99 IO=1,IOMAX
      IF(T .GE. 0.) YS=20.
      DO 10 J=1,IO
      NJ=N-J
      YZI(J)=0.
      DO 11 I=1,NJ
      YZI(J)=YZI(J)+H(I+J)*UP(I)
11    CONTINUE
      YD(J) = (EXP(-J*DT/TOM)) * (Y1 - YS) + YS
      Z(J)=YD(J)-YZI(J)
10    CONTINUE
      UF(1)=Z(1)/H(1)
      IF(ABS(UF(1)).GT.ULIM) UF(1)=ULIM*(ABS(UF(1)))/UF(1)
      DO 12 J=2,IO
      UF(J)=Z(J)
      J1=J-1
      DO 13 I=1,J1
      UF(J)=UF(J)-H(J-I+1)*UF(I)
13    CONTINUE
```

```
      UF(J)=UF(J)/H(1)
      IF(ABS(UF(J)).GT.ULIM) UF(J)=ULIM*(ABS(UF(J)))/UF(J)
12    CONTINUE
      NIO=N-IO
      DO 14 J=1,NIO
      K=N+1-J
      UP(K)=UP(K-IO)
14    CONTINUE
      DO 15 J=1,IO2
      J2 = (J + 4) / 5
      V = UF(J2)
      CALL OUTPUT(Y,X,1,D,T,FEE,DELAM)
      JC=(IO-1)*IO2+J
      YA(JC) = X(1)
      UP(J2)=UF(IO+1-J2)
      UA(JC) = V
15    CONTINUE
      Y1 = X(1)
99    CONTINUE
      CALL PLOT(0.,-4.,-3) $ CALL PLOT(0.,0.03,-3)
      DO 20 J = 1,N2
      TAD(J) = 0.0
20    CONTINUE
      DO 30 J = 1,N2
      TAD(J) = J * C
30    CONTINUE
      CALL VGRAPH(TAD,YA,N2,IDY,1,0,1)
      CALL PLOT(0.,-4.,-3) $ CALL PLOT(0.,0.03,-3)
      CALL VGRAPH(TAD,UA,N2,IDU,1,0,1)
      PRINT 101
      PRINT*,"   CONTROL INPUT          ACTUAL OUTPUT
     1     TIME"
      PRINT*," "
      PRINT 103,(UA(I),YA(I),TAD(I),I=1,N2)
103   FORMAT(3(5X,G15.7))
      CALL PLOTE(H)
      STOP
      END
      FUNCTION UI(T)
      UI = 1.
      RETURN
      END
      SUBROUTINE DECON (Y,U,H,N,DT )
      DIMENSION  Y(302),U(302),H(302)
      H(1) = Y(1)/(DT*U(1) )
      DO 1 J=2,N
      H(J) = Y(J)/DT
      J1 = J - 1
      DO 2 I=1,J1
      H(J) = H(J) - H(J-I)*U(I+1)
2     CONTINUE
      H(J) = H(J)/U(1)
1     CONTINUE
      RETURN
      END
```

```
      SUBROUTINE HGRAPH(X,Y,N,ID,NO,NP,NS)
      DIMENSION X(1),Y(1),ID(1) $ IF(NO.EQ.2) GO TO 30
      IF (NO.LT.0) GO TO 10
      CALL SCALE(X,7.,N,1)       $ CALL SCALE(Y,5.,N,1)
   10 CALL PLOT(8.5,0.,-3) $   CALL PLOT(0.,11.,3)
      CALL PLOT(-1.35,1.35,3)
      CALL PLOT(-7.15,1.35,2)  $  CALL PLOT(-7.15,9.65,2)
      IF(ID(1).EQ.000) GO TO 25
      CALL PLOT(-7.05,9.55,3)     $ CALL PLOT(-7.05,7.55,2)
      DO 20 I=1,7,2
   20 CALL SYMBOL(I*.1-6.9,7.85 ,.07,ID(I),90.,20)
      CALL PLOT(-7.05,7.55,3)     $ CALL PLOT(-6.05,7.55,2)
      CALL PLOT(-6.05,9.55,2)     $ CALL PLOT(-7.05,9.55,2)
      CALL PLOT(-7.15,9.65,3)
   25 CALL PLOT(-1.35,9.65,2)    $ CALL PLOT(-1.35,1.35,2)
      CALL SYMBOL(-6.65,1.15,.1,ID(13),0.,50)
      CALL AXIS(-1.85,2.1,ID(9),-20,7.,90.,X(N+1),X(N+2))
      CALL AXIS(-1.85,2.1,ID(11),20,5.,180.,Y(N+1),Y(N+2))
   30 Y(N+2)=-Y(N+2)
      X(N+1)=X(N+1)-2.1*X(N+2) $ Y(N+1)=Y(N+1)+1.85*Y(N+2)
      CALL LINE(Y,X,N,1,NP,NS)
      X(N+1)=X(N+1)+2.1*X(N+2) $ Y(N+1)=Y(N+1)-1.85*Y(N+2)
      Y(N+2)=-Y(N+2)
      RETURN                          $ END
      SUBROUTINE VGRAPH(X,Y,N,ID,NO,NP,NS)
      DIMENSION X(1),Y(1),ID(1) $ IF(NO.EQ.2) GO TO 30
      IF (NO.LT.0) GO TO 10
      CALL SCALE(X,4.9,N,1)       $ CALL SCALE(Y,7.0,N,1)
   10 CALL PLOT(8.5,0.,-3) $   CALL PLOT(0.,11.,3)
      CALL PLOT(-1.35,1.35,3)
      CALL PLOT(-7.15,1.35,2)  $  CALL PLOT(-7.15,9.65,2)
      CALL PLOT(-1.35,9.65,2) $ IF(ID(13).EQ.000) GO TO 25
      CALL PLOT(-1.45,9.55,3) $ CALL PLOT(-3.45,9.55,2)
      DO 20 I= 1,7,2
   20 CALL SYMBOL(-3.15,9.4-I* .10,.07,ID(I),0.,20)
      CALL PLOT(-3.45,9.55,3) $ CALL PLOT(-3.45,8.55,2)
      CALL PLOT(-1.65,8.55,2) $ CALL PLOT(-1.45,9.55,2)
      CALL PLOT(-1.35,9.65,3)
   25 CALL PLOT(-1.35,1.35,2)
      CALL SYMBOL(-6.65,1.15,.1,ID(13),0.,50)
      CALL AXIS(-6.4,1.85,ID(9),-20,4.9,0.,X(N+1),X(N+2))
      CALL AXIS(-6.4,1.85,ID(11),20,7.0,90.,Y(N+1),Y(N+2))
   30 X(N+1)=X(N+1)+6.4*X(N+2)  $ Y(N+1)=Y(N+1)-1.85*Y(N+2)
      CALL LINE(X,Y,N,1,NP,NS)
      X(N+1)=X(N+1)-6.4*X(N+2)  $ Y(N+1)=Y(N+1)+1.85*Y(N+2)
      RETURN      $ END
      SUBROUTINE OUTPUT(Y,X,N,DT,T,FEE,DELAM)
      DIMENSION Y(2102),X(2),DX(2),FEE(2,2),DELAM(2)
      COMMON L1,V
      IF(L1.EQ.1) GO TO 2
      V=1.
    2 CONTINUE
      DO 1 J=1,N
      T=T+DT
```

```
      DX(1)=FEE(1,1)*X(1)+FEE(1,2)*Y(2)+DELAM(1)*V
      DX(2)=FEE(2,1)*X(1)+FEE(2,2)*X(2)+DELAM(2)*V
      X(1)=DX(1)
      X(2)=DX(2)
      Y(J)=X(1)
    1 CONTINUE
      RETURN
      END
      SUBROUTINE STMLAM(FEE,DELAM,DT,SIGMA,OMEGA)
      DIMENSION FEE(2,2),DELAM(2)
      A21 = -(SIGMA**2 + OMEGA**2)
      A22 = -2.0 * ABS(SIGMA)
      B12 = -SIGMA/OMEGA
      B22 = 1.0/OMEGA
      EDEL = EXP(SIGMA * DT)
      CODEL = COS(OMEGA * DT)
      SIDEL = SIN(OMEGA * DT)
      ECODEL = EDEL * CODEL
      ESIDEL = EDEL * SIDEL
      A1T = ECODEL + B12 * ESIDEL
      A2T = B22 * ESIDEL
      FEE(1,1) = A1T
      FEE(1,2) = A2T
      FEE(2,1) = A21 * A2T
      FEE(2,2) = A1T + A22 * A2T
      AINV11 = -A22/A21
      AINV12 = 1./A21
      EATM11 = FEE(1,1) - 1.0
      EATM12 = FEE(1,2)
      EATM21 = FEE(2,1)
      EATM22 = FEE(2,2) - 1.0
      DELAM(1) = AINV11 * EATM12 + AINV12 * EATM22
      DELAM(2) = EATM12
      RETURN
      END
```

Appendix B

The FORTRAN source code listed in this appendix differs from the one in Appendix A in that an m^{th}-order system model may now be specified. Besides specifying the variables as presented in the initial comments section, the user must specify the appropriate differential equation in the format shown in SUBROUTINE F(T,X,DX) on page 99. Once these specifications are made, the source code may be compiled; however, before execution, the library containing the differential equation solver must be attached.

```
      PROGRAM CONTRCL(INPUT,OUTPUT,PLOT)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     THIS PROGRAM COMPUTES THE DISCRETE IMPULSE RESPONSE
C     FUNCTION FOR A SPECIFIED SINGLE INPUT / SINGLE
C     OUTPUT (SISO) SYSTEM.  THE RESULT IS THEN USED
C     BY MODEL ALGORITHMIC CONTROL (MAC) TO DETERMINE
C     THE CONTROLS REQUIRED TO FOLLOW THE DESIRED
C     OUTPUT.  OUTPUT DATA CONSISTS OF PRINTOUTS AND
C     PLOTS FOR VALUES OF DISCRETE IMPULSE RESPONSE
C     (H-DECONVOLUTION), COMPUTED CONTROL INPUTS, AND
C     ACTUAL OUTPUT (SAMPLED AT ONE-FIFTH THE DISCRETE
C     TIME INTERVAL USED IN DECONVOLUTION).  USER
C     SUPPLIES THE FOLLOWING INFORMATION:
C               TOW = DESIRED TIME CONSTANT OF
C                     FIRST ORDER SYSTEM MODEL
C               IO = NUMBER OF FUTURE CONTROLS
C                    TO BE COMPUTED EACH CYCLE
C               YS = SET POINTS OF FIRST ORDER MODEL
C               ULIM = MAXIMUM CONTROL VALUE ALLOWED
C               M = ORDER OF SISO SYSTEM
C               CONTIME = DESIRED DURATION (IN SECONDS)
C                         OF DISCRETE IMPULSE RESPONSE
C               SIMTIME = DESIRED DURATION (IN SECONDS)
C                         OF SIMULATION
C               DT = DISCRETE TIME INCREMENT USED
C                    IN DECONVOLUTION
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      DIMENSION H(200),YZI(5),YD(5),Z(5),UF(5),UP(200)
      DIMENSION X(20),U(200),TA(200),Y(200),TAD(1000)
      DIMENSION IDU(17),UA(1000)
      DIMENSION IDH(17),IDY(17),YA(1000)
      COMMON L1,V
      DATA IDH(1)/20H  IMPULSE RESPONSE  /
      DATA IDH(3)/20H    DT = PI/OMEGA2  /
      DATA IDH(5)/20H     CONTIME = 33   /
      DATA IDH(7)/20H FOURTH ORDER MODEL /
      DATA IDH(9)/20H  TIME(SECONDS)     /
      DATA IDH(11)/20H  IMPULSE RESPONSE /
      DATA IDH(13)/50H
     1/
      DATA IDY(1)/20H     SYSTEM OUTPUT  /
      DATA IDY(3)/20H       TOW = 0.5    /
      DATA IDY(5)/20H  YS = 100 (0<T<8)  /
      DATA IDY(7)/20H   YS = 20 (8<T)    /
      DATA IDY(9)/20H   TIME(SECONDS)    /
      DATA IDY(11)/20H    SYSTEM OUTPUT  /
      DATA IDY(13)/50H
     1/
      DATA IDU(1)/20H     CONTROL INPUT  /
      DATA IDU(3)/20H        IO = 1      /
      DATA IDU(5)/20H  ULIM = UNLIMITED  /
      DATA IDU(7)/20H                    /
```

```
      DATA IDU(9)/20H    TIME(SECONDS)    /
      DATA IDU(11)/20H   CONTROL,INPUT    /
      DATA IDU(13)/50H
     1/
      CALL PLOT(0.,-4.,-3) $ CALL PLOT(0.,0.03,-3)
C          INITIALIZE VARIABLES
      TOW=.5
      OMEGA = 5.0
      CONTIME = 33
      SIMTIME = 15
      IO=1
      YS=100.
      PI=3.14159265358898
      ULIM=1.E+30
      L1 = 0
      T = 0
      M = 4
      DT = PI /   OMEGA
      N = CONTIME / DT
      D=DT/5.
      IO2 = 5 * IO
      N2 = SIMTIME * 5 / DT
      IOMAX = SIMTIME / (DT * IO) + 5
      DO 1 J=1,N
      U(J) = UI( J * DT - D )
1     CONTINUE
      DO 2 J=1,M
      X(J) = 0.
2     CONTINUE
      DO 3 J=1,M
      TA(J) = 0.0
3     CONTINUE
      DO 5 J= 1,N
      TA(J) = J*DT
5     CONTINUE
C          CALL SUBROUTINES TO CALCULATE THE DISCRETE
C          IMPULSE RESPONSE
      CALL OUTPUT(Y,X,M,N,DT,T)
      CALL DECON(Y,U,M,N,DT)
      PRINT 101
      PRINT*,"  THE VARIABLES OF INTEREST FOR THIS RUN ARE:"
      PRINT*,"  N =  ",N,"  DT =  ",DT,"  TOW =  ",TOW,
      PRINT*,"  ULIM = ",ULIM
      PRINT 101
      PRINT*,"  H - DECONVOLUTION          TIME "
      PRINT*," "
102   FORMAT(2(5X,G15.7))
101   FORMAT(1H1)
      DO 8 J = 1,N
      H(J) = H(J) * DT
8     CONTINUE
      PRINT 102,(H(I),TA(I),I=1,N)
      PRINT 101
      CALL HGRAPH(TA,H,N,IOM,1,0,1)
```

```
C           RE-INITIALIZE VARIABLES
      T = 0.0
      L1 = 1
      DO 6 J = 1,N
      UP(J)=0.
    6 CONTINUE
      DO 9 J = 1,M
      X(J) = 0.0
    9 CONTINUE
      Y1 = 0.0
C           APPLY THE CONTROL ALGORITHM TO FOLLOW SPECIFIED
C           FIRST ORDER MODEL
      DO 99 IO=1,IOMAX
      IF(T.GE.8) YS = 20.
      DO 10 J=1,IO
      NJ=N-J
      YZI(J)=0.
      DO 11 I=1,NJ
      YZI(J)=YZI(J)+H(I+J)*UP(I)
   11 CONTINUE
      YD(J) = (EXP(-J*DT/TOW)) * (Y1 - YS) + YS
      Z(J)=YD(J)-YZI(J)
   10 CONTINUE
      UF(1)=Z(1)/H(1)
      IF(ABS(UF(1)).GT.ULIM) UF(1)=ULIM*(ABS(UF(1)))/UF(1)
      DO 12 J=2,IO
      UF(J)=Z(J)
      J1=J-1
      DO 13 I=1,J1
      UF(J)=UF(J)-H(J-I+1)*UF(I)
   13 CONTINUE
      UF(J)=UF(J)/H(1)
      IF(ABS(UF(J)).GT.ULIM) UF(J)=ULIM*(ABS(UF(J)))/UF(J)
   12 CONTINUE
      NIO=N-IO
      DO 14 J=1,NIO
      K=N+1-J
      UP(K)=UP(K-IO)
   14 CONTINUE
      DO 15 J=1,IO2
      J2 = (J + 4) / 5
      V = UF(J2)
      CALL OUTPUT(Y,X,M,1,0,T)
      JC=(IO-1)*IO2+J
      YA(JC) = X(1)
      UP(J2)=UF(IO+1-J2)
      UA(JC) = V
   15 CONTINUE
      Y1 = X(1)
   99 CONTINUE
      CALL PLOT(0.,-4.,-3) $ CALL PLOT(0.,0.03,-3)
      DO 20 J = 1,N2
      TAD(J) = 0.0
   20 CONTINUE
```

```
      DO 30 J = 1,N2
      TAD(J) = J * C
30    CONTINUE
      CALL VGRAPH(TAD,YA,N2,IDY,1.0,1)
      CALL PLOT(0.,-4.,-3) & CALL PLOT(0.,0.0?,-3)
      CALL VGRAPH(TAD,UA,N2,IDU,1.0,1)
      PRINT 101
      PRINT*,"   CONTROL INPUT          ACTUAL OUTPUT
     1    TIME"
      PRINT*," "
      PRINT 103,(UA(I),YA(I),TAD(I),I=1,N2)
103   FORMAT(3(5X,G15.7))
      CALL PLOTE(M)
      STOP
      END
      SUBROUTINE OUTPUT(Y,X,M,N,DT,T)
      EXTERNAL F
      DIMENSION WORK(200),IWORK(5),X(20),Y(200)
      A = 1.E-10
      B = 1.E-10
      DO 1 J=1,N
      TOUT = T + DT
      IFLAG = 1
      CALL ODE(F,M,X,T,TOUT,A,B,IFLAG,WORK,IWORK)
      Y(J) = X(1)
1     CONTINUE
      RETURN
      END
      SUBROUTINE F(T,X,DX)
C        UF(T) IS A USER SUPPLIED INPUT FUNCTION
      DIMENSION X(20),DX(20)
      COMMON L1,V
      IF ( L1.EQ.1) GO TO 1
      V = UI(T)
1     CONTINUE
      DX(1) = X(2)
      DX(2) = X(3)
      DX(3) = X(4)
      DX(4)=-8611.769806*X(1)-279.096*X(2)-274.075*X(3)-
     11.6*X(4)+V
      RETURN
      END
      FUNCTION UI(T)
      UI = 1.
      RETURN
      END
      SUBROUTINE DECON (Y,U,H,N,DT )
      DIMENSION  Y(200),U(200),H(200)
      H(1) = Y(1)/(DT*U(1) )
      DO 1 J=2,N
      H(J) = Y(J)/DT
      J1 = J - 1
      DO 2 I=1,J1
      H(J) = H(J) - H(J-I)*U(I+1)
```

```
2       CONTINUE
        H(J) = H(J)/U(1)
1       CONTINUE
        RETURN
        END
        SUBROUTINE HGRAPH(X,Y,N,ID,NO,NP,NS)
        DIMENSION X(1),Y(1),ID(1) $ IF(NO.EQ.2) GO TO 30
        IF (NO.LT.0) GO TO 10
        CALL SCALE(X,7.,N,1)       $ CALL SCALE(Y,5.,N,1)
10      CALL PLOT(8.5,0.,-3) $   CALL PLOT(0.,11.,3)
        CALL PLOT(-1.35,1.35,3)
        CALL PLOT(-7.15,1.35,2)  $  CALL PLOT(-7.15,9.65,2)
        IF(ID(1).EQ.000) GO TO 25
        CALL PLOT(-7.05,9.55,3)    $ CALL PLOT(-7.05,7.55,2)
        DO 20 I=1,7,2
20      CALL SYMBOL(I*.1-6.9,7.85 ,.07,ID(I),90.,20)
        CALL PLOT(-7.05,7.55,3)    $ CALL PLOT(-6.05,7.55,2)
        CALL PLOT(-6.05,9.55,2)    $ CALL PLOT(-7.05,9.55,2)
        CALL PLOT(-7.15,9.65,3)
25      CALL PLOT(-1.35,9.65,2)   $ CALL PLOT(-1.35,1.35,2)
        CALL SYMBOL(-6.65,1.15,.1,ID(13),0.,30)
        CALL AXIS(-1.85,2.1,ID(9),-20,7.,90.,X(N+1),X(N+2))
        CALL AXIS(-1.85,2.1,ID(11),20,5.,180.,Y(N+1),Y(N+2))
 30     Y(N+2)=-Y(N+2)
        X(N+1)=X(N+1)-2.1*X(N+2) $ Y(N+1)=Y(N+1)+1.85*Y(N+2)
        CALL LINE(Y,X,N,1,NP,NS)
        X(N+1)=X(N+1)+2.1*X(N+2) $ Y(N+1)=Y(N+1)-1.85*Y(N+2)
        Y(N+2)=-Y(N+2)
        RETURN                     $ END
        SUBROUTINE VGRAPH(X,Y,N,ID,NO,NP,NS)
        DIMENSION X(1),Y(1),ID(1) $ IF(NO.EQ.2) GO TO 30
        IF (NO.LT.0) GO TO 10
        CALL SCALE(X,4.9,N,1)      $ CALL SCALE(Y,7.0,N,1)
10      CALL PLOT(8.5,0.,-3) $   CALL PLOT(0.,11.,3)
        CALL PLOT(-1.35,1.35,3)
        CALL PLOT(-7.15,1.35,2)  $  CALL PLOT(-7.15,9.65,2)
        CALL PLOT(-1.35,9.65,2) $ IF(ID(1).EQ.000) GO TO 25
        CALL PLOT(-1.45,9.55,3) $ CALL PLOT(-3.45,9.55,2)
        DO 20 I= 1,7,2
20      CALL SYMBOL(-3.15,9.4-I*.10,.07,ID(I),0.,20)
        CALL PLOT(-3.45,9.55,3) $ CALL PLOT(-3.45,8.55,2)
        CALL PLOT(-1.45,8.55,2) $ CALL PLOT(-1.45,9.55,2)
        CALL PLOT(-1.35,9.65,3)
25      CALL PLOT(-1.35,1.35,2)
        CALL SYMBOL(-6.65,1.15,.1,ID(13),0.,30)
        CALL AXIS(-6.4,1.85,ID(9),-20,4.9,0.,X(N+1),X(N+2))
        CALL AXIS(-6.4,1.85,ID(11),20,7.0,90.,Y(N+1),Y(N+2))
 30     X(N+1)=X(N+1)+6.4*X(N+2)  $ Y(N+1)=Y(N+1)-1.85*Y(N+2)
        CALL LINE(X,Y,N,1,NP,NS)
        X(N+1)=X(N+1)-6.4*X(N+2)  $ Y(N+1)=Y(N+1)+1.85*Y(N+2)
        RETURN      $ END
```

VITA

Howard J. Colson, Jr. was born in El Paso, Texas in January 1947. In June 1969 he graduated from Texas Tech University with a Bachelor of Science degree in Electrical Engineering. Following his graduation from Texas Tech, he underwent the Officers Training School (OTS) commissioning program. After being commissioned a Second Lieutenant, he was assigned to Undergraduate Pilot Training at Craig AFB, Alabama. Upon receipt of his pilot's wings in October 1970, he reported to the 4900th Test Group, Kirtland AFB, New Mexico where he flew the C-131, C-130, and T-39 on various test missions. He was then assigned to SCATBACK, NKP, RTAFB, Thailand as a T-39 instructor pilot. His last assignment prior to reporting to AFIT was at Clark AB, Republic of the Philippines as a 3rd Tactical Fighter Wing, Aircrew Evaluation Division, T-39 Flight Evaluator.

Permanent address: P. O. Box 1462
601 North Gillis
Fort Stockton, Texas 79735

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GE/EE/78-21	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* APPLICATION OF MODEL ALGORITHMIC CONTROL TO A LIGHTLY DAMPED SINGLE INPUT SINGLE OUTPUT SYSTEM		5. TYPE OF REPORT & PERIOD COVERED MS Thesis 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) HOWARD J. COLSON, JR.		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT-EN) Wright-Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Flight Dynamics Laboratory (AFFDL-FGC) Air Force Systems Command Wright-Patterson AFB, Ohio 45433		12. REPORT DATE December, 1978 13. NUMBER OF PAGES 110
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

Approved for public release; IAW AFR 190-17

Joseph P. Hipps, Major, USAF
Director of Information 1-23-79

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Model Following
Model Algorithmic Control
Impulse Response
Time Domain Control Technique

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

A new digital control technique, called Model Algorithmic Control (MAC), was applied to a single-input single-output model containing complex eigenvalues. The MAC algorithm developed was a simplified version of the algorithm used by ADERSA/GERBIOS Corporation (France) in a complete computer program designated IDCOM (Identification and Command). A second-order model based on the dominant eigenvalues of the B-52E Flutter Mode was selected as the hypothetical system to be controlled. Data were obtained for various selections of control parameters in the implementing program, then a study was made of → next page

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

cont

the controlling program's "robustness" to eigenvalue changes. The second-order model was subsequently expanded to a third-order, then a fourth-order model, while maintaining the previously found "optimum" control parameters. Results obtained indicated that the "robustness" exhibited in the MAC concept as implemented in IDCOM is not attributed to the impulse response prediction technique, but rather must be attributed to the particular control algorithm used in IDCOM. Therefore, before conclusive results could be obtained for robustness and higher-order model studies, further refinement of the simplified MAC algorithm used in this thesis was necessary. A revision to the implementing program was begun, but due to time considerations, was not completed.