ABSTRACT

Modal Control of Flexible Beam Using Smart Materials

by

Sujata Mallepalle

In this work, the dynamics and control aspects of a vibrating flexible beam using modal analysis are studied. To damp the vibrations of the system, the vibrations of each mode has to be controlled, which can be done if we know the individual mode shape, and the resonance frequency. These quantities can be derived mathematically or measured experimentally with a spectrum analyzer. Individual modal velocities can be computed by integration of the product of beam velocity and the mode shape, over the interval of beam length. The integration is carried out using numerical methods. The necessary discrete ordinates are obtained by measuring the system velocity at several points on the beam. This mode velocity estimation method constitutes the mode separation scheme which is the principle feature of this thesis. Controlling of the system vibrations can be achieved by controlling individual mode vibrations. The control action for each mode are decoupled from others, because of the frequency separation. So the resulting the controller is modular, consisting of N_o (N_o = number of outputs to be regulated) structurally identical modules. The combined mode separation scheme and modular controller are the desired modal controllers that stabilize, and regulate the beam dynamics.

MODAL CONTROL OF FLEXIBLE BEAM USING SMART MATERIALS

by
Sujata Mallepalle

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science

Department of Electrical and Computer Engineering

January, 1993

APPROVAL PAGE

Modal Control of Flexible Beam Using Smart Materials

Sujata Mallepalle

Dr. Timothy N. Chang, Thesis Advisor

Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. Nirwan Ansari, Committee Member

Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. Durga Misra, Committee Member

Assistant Professor of Electrical and Computer Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Sujata Mallepalle

Degree: Master of Science in Electrical and Computer Engineering

Date: January, 1993

Undergraduate and Graduate Educations:

- Master of Science in Electrical and Computer Engineering,
  New Jersey Institute of Technology, Newark, NJ, 1993
- Master of Arts in Mathematics,
  Trenton State College, Trenton, NJ, 1991
- Bachelor of Science in Electronics and Communication Engineering,
  Venkateswara University, Tirupati, India, 1984

Major: Electrical and Computer Engineering

ACKNOWLEDGMENT

I am indebted to Dr. Timothy N. Chang, my thesis advisor for his invaluable support and guidance throughout the research and development of this thesis. I thank him for his patience and for all of his suggestions to refine and improve this research work as well as manuscript. My sincere appreciation to Dr. Nirwan Ansari and Dr. Durga Misra who agree to work on the thesis committee, given the very short notice. My sincere thanks to Dr. Chang and N.J.I.T. for providing the necessary financial support for my studies. I would like to thank all teachers at NJIT, especially the ECE faculty and fellow students for their support and encouragement. Finally, to Qian Wang , I express my deep appreciation for her expert typing of the manuscript.

TABLE OF CONTENTS

Chapter **Page**

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

Table Page

CHAPTER 1

INTRODUCTION

Flexible systems are characterized by the presence of many modes. By means of parallel decomposition, it is evident that the modes of vibration, each having its own frequency, behave essentially as second order systems. This allows us to express the motion of the system in terms of the modal vibrations, each proceeding at its own frequency, completely independent of the other, the amplitudes and phases being determined by the initial and excitation conditions. The total motion of the system is given by superposition of the modal harmonic vibrations. In Chapter 1, the dynamics of a flexible structure undergoing transversal motion is modelled. From the mathematical model, natural frequencies and mode shapes are calculated.

To damp the system vibrations, individual mode vibrations have to be damped. In this work, damping of the individual mode velocities is implemented by feeding back the individual mode velocities with proper gain. For this we need have individual mode velocities, which can be estimated by numerical methods by placing several number of sensors on the beam. In Chapter 2, how to estimate the modal velocities using numerical methods is discussed. And a method to estimate the minimum number of sensors we need to use for good estimation of modal velocities is given. Finally, these results are simulated using ALSIM software, for open loop dynamics, closed loop dynamics with good estimation of modal velocities, and closed loop dynamics without good estimation of modal velocities.

In Chapter 4, how to implement the estimation of modal velocities using smart materials is discussed. Finally, in Chapter 5, conclusions are stated and directions for further development are given.

CHAPTER 2

MATHEMATICAL MODEL OF FLEXIBLE BEAM DYNAMICS

The equation of motion of long thin members undergoing transverse vibrations can be described by (2.1), which upon solution, generates an infinite natural modes of vibration. Figure(2.1) shows the freebody diagram of a beam undergoing transverse motion.

In this work, the Bernoulli-Euler assumptions of elementary beam theory are employed, namely:

1. There is an axis of the beam which undergoes no extension or contraction, which is a neutral-axis.

2. Cross sections perpendicular to the neutral axis in the undeformed beam remain plane and remain perpendicular to the deformed neutral axis, that is transverse shear deformation is neglected.

3. The material is linearly elastic and the beam is homogeneous at any cross section.

2.1 Mathematical Model of a Flexible Beam

The equation of motion for transverse vibrations of a beam, neglecting shear deflection and rotary inertia is given by

$$E\,I\,y'''' = -\rho\,\ddot{y} \tag{2.1}$$

where E is the Young's Modulus

I is the moment of Inertia

ρ is linear density

y is displacement

y' is differentiation with respect to x

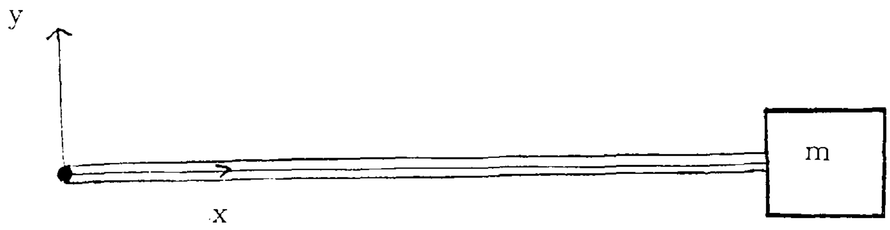

Figure 2.1 Free Body Diagram of the Flexible Beam

$\dot{y}$ is differentiation with respect to t

The left end ($x = 0$) of the beam is hinged to the motor. The concentrated mass m is attached at the right end ($x = l$).

The appropriate boundary conditions are:

$$\begin{aligned} y(0,t) &= 0 \\ y''(0,t) &= 0 \\ y''(l,t) &= 0 \\ E\,I\,y'''(l,t) &= m\,\ddot{y}(l,t) \end{aligned} \tag{2.2}$$

The solution to (2.1) is given by the method of separation of variables:

$$y(x,t) = \sum_{i=1}^{\infty} X_i(x)\,\Phi_i(t) \tag{2.3}$$

then

$$X_i'''' = -k_i\,X_i \tag{2.4}$$

$$\ddot{\Phi}_i = -\omega_i^2\,\Phi_i \tag{2.5}$$

where

$$k_i = \frac{\mu_i}{l}$$

$$\omega_i^2 = \frac{E\,I}{\rho}\,k_i^4$$

μ_i is mode shape frequency.

The solution to (2.4) can be given as

$$X(x) = A_i cos(k_i x) + B_i sin(k_i x) + C_i cosh(k_i x) + D_i sinh(k_i x) \tag{2.6}$$

Applying boundary conditions (2.2) leads to

$$A_i = 0$$

$$C_i = 0$$

and $X(x_i) = B_i sin(k_i x) + D_i sinh(k_i x)$.

with

$$B_i = D_i\,\frac{sinh(k_i l)}{sin(k_i l)}$$

The resulting frequency equation is

$$-\,2\,K\;\mu_i\,sin\mu_i\,sinh\mu_i = sin\mu_i\,cosh\mu_i - sinh\mu_i\,cos\mu_i \tag{2.7}$$

where K is the ratio of end mass to bean mass.

Combining (2.6) and (2.7) yields

$$X_i(x) = B_i\left[\frac{sin(\mu_i\,x\,/l)}{sin(\mu_i)} + \frac{sinh(\mu_i\,x\,/l)}{\sinh(\mu_i)}\right] \tag{2.8}$$

The orthogonality condition of the mode shapes $X_i(x)$ are expressed as:

$$\int_0^l r(x)\,X_i(x)\,X_j(x)\,dx = \left\{\begin{array}{cc} 0 & i \neq j \\ 1 & i = j \end{array}\right. \tag{2.9}$$

with the generalized weighting function $r(x)$ given by

$$r(x) = 1 + K\,l\,\delta(x - l) \tag{2.10}$$

where $\delta(x-l)$ is the unit impulse function.

Verification of the orthogonality of mode shapes:

$$\int_0^l r(x)\,X_i(x)\,X_j(x)\,dx = 0; \qquad i \neq j$$

$$\begin{aligned}&\int_0^l [1 + K\,l\,\delta(x-l)]X_i(x)\,X_j(x)\,dx \\ &= \int_0^l X_i(x)\,X_j(x)\,dx + K\,l \int_0^l \delta(x-l)\,X_i(x)\,X_j(x)\,dx\end{aligned} \tag{2.11}$$

$$\int_0^l \delta(x-l)\,X_i(x)\,X_j(x)\,dx = -4B_iB_j \tag{2.12}$$

Now since

$$\begin{aligned}&\int_0^l X_i(x)\,X_j(x)\,dx \\ &= \int_0^l B_iB_j \left[\frac{sin(\mu_i x/l)}{sin(\mu_i)} + \frac{sinh(\mu_i x/l)}{sinh(\mu_i)}\right]\left[\frac{sin(\mu_j x/l)}{sin(\mu_j)} + \frac{sinh(\mu_j x/l)}{sinh(\mu_j)}\right] dx\end{aligned} \tag{2.13}$$

$$\int_0^l \frac{sin(\mu_i x/l)}{sin(\mu_i)}\,\frac{sin(\mu_j x/l)}{sin(\mu_j)}\,dx = \frac{1}{2sin(\mu_i)sin(\mu_j)}\left[\frac{sin(\mu_i\mu_j)}{\mu_i-\mu_j} - \frac{sin(\mu_i+\mu_j)}{\mu_i+\mu_j}\right] \tag{2.14}$$

$$\int_0^l \frac{sinh(\mu_i x/l)}{sinh(\mu_i)}\,\frac{sin(\mu_j x/l)}{sin(\mu_j)}\,dx = \frac{\mu_j^2}{\mu_i^2+\mu_j^2}\left[\frac{-1}{\mu_j^2}\,cot(\mu_j) + \frac{\mu_i\,l}{\mu_j^2}coth(\mu_i)\right] \tag{2.15}$$

$$\int_0^l \frac{sinh(\mu_j x/l)}{sinh(\mu_j)}\,\frac{sin(\mu_i x/l)}{sin(\mu_i)}\,dx = \frac{\mu_i^2}{\mu_i^2+\mu_j^2}\left[\frac{-1}{\mu_i^2}\,cot(\mu_i) + \frac{\mu_i\,l}{\mu_j^2}coth(\mu_i)\right] \tag{2.16}$$

$$\begin{aligned}&\int_0^l \frac{sinh(\mu_i x/l)}{sinh(\mu_i)}\,\frac{sinh(\mu_j x/l)}{sinh(\mu_j)}\,dx \\ &= \frac{1}{2(\mu_i+\mu_j)}\,[cot(\mu_j) - cot(\mu_i)] - \frac{1}{2(\mu_i-\mu_j)}\,[cot(\mu_i) - cot(\mu_j)]\end{aligned} \tag{2.17}$$

substituting (2.13),(2.15),(2.16),(2.17),(2.18) together with the relation (2.8) in (2.13) will prove the orthogonality condition (2.9).

Table 2.1 Natural Frequencies of the Flexible Beam

i	μ_i	ω_i	B_i
1	3.1711	0.3128	0.0035
2	6.2983	1.2341	0.0021
3	9.4349	2.7694	0.0006
4	12.5740	4.9188	0.0041
5	15.7141	7.6822	0.0010
6	18.8546	11.0597	0.0061
7	21.9955	15.0514	0.00111
8	25.1366	19.6542	0.0040
9	28.2777	24.8770	0.0031
10	31.4190	30.7110	0.0020
11	34.5603	37.1590	0.0052
12	37.7017	44.2212	0.0001
13	40.8431	51.8975	0.0049
14	43.9845	60.1878	0.0022
15	47.1259	69.0921	0.0029
16	50.2674	78.6108	0.0042
17	53.4089	88.7435	0.0009
18	56.5504	99.4902	0.0062
19	59.6919	110.8511	0.0012
20	62.8334	122.8260	0.0039

Therefore,

$$B_i = [l\,(K + 1 + 0.5\,(cot\mu_i^2 - coth\mu_i^2)\,)\,]^{-\frac{1}{2}} \tag{2.18}$$

The natural frequencies are calculated from the frequency equation and are tabulated in Table (2.1).

Assuming concentrated moment is applied at $x = x_o$, the resulting equation of motion can be obtained by virtual work, refer [1] as :

$$\ddot{\Phi}_i + \omega_i^2\,\Phi_i = \frac{M(t)}{\rho}\,X_i'(x_o) \tag{2.19}$$

Equation (2.19) represents the dynamics of an undamped system.

By considering damping effects, the dynamic equation of motion becomes

$$\ddot{\Phi}_i + 2\,\xi_i\,\omega_i\,\dot{\Phi}_i + \omega_i^2\Phi_i^2 = \frac{M(t)}{q}\,X_i'(x_o) \tag{2.20}$$

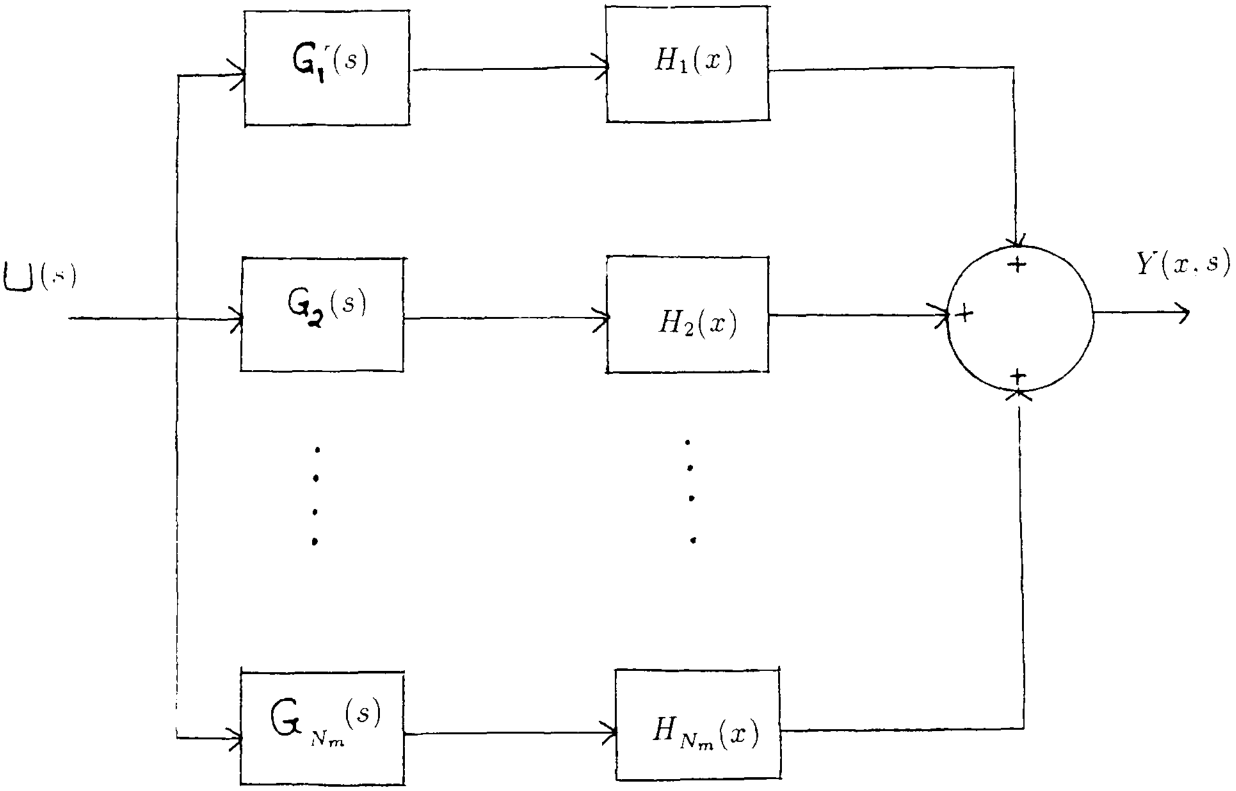

Figure 2.2 Internal Structure of Flexible Beam Dynamics

The transfer function between $y(x,t)$ and $u(t) = M(t)$ is given by

$$T(s) = \frac{Y(x,s)}{u(s)} = \sum_{i=1} \frac{H_i(x)}{s^2 + 2\xi_i\,\omega_i\,s + \omega_i^2} \tag{2.21}$$

where

$$H_i(x) = \frac{X_i(x)\,X_i'(x_o)}{q} \qquad x \in (0, l] \tag{2.22}$$

The internal structure of the beam dynamics is shown in Figure(2.2).

Assuming a fourth order model, the finite dimensional model of the flexible beam is given by:

$$T(s) = \frac{H_1(x)}{s^2 + 2\xi_1\,\omega_1\,s + \omega_1^2} + \frac{H_2(x)}{s^2 + 2\xi_2\,\omega_2\,s + \omega_2^2} \tag{2.23}$$

Now, the transfer function model is converted to a state-space model of the following form:

$$\begin{aligned} \dot{q} &= Aq + Bu \\ y &= Cq \end{aligned} \tag{2.24}$$

where q, u, y are the state vector, input, and output respectively.

The A, B, C matrices are given by

$$\begin{aligned} A &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\omega_1^2 & -\xi_1\,\omega_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\omega_2^2 & -\xi_2\,\omega_2 \end{bmatrix} \\ B &= \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \\ C &= \begin{bmatrix} H_1(x) & 0 & H_2(x) & 0 \end{bmatrix} \end{aligned} \tag{2.25}$$

Since, (2.19) is a minimal realization, (2.18) is clearly controllable. Further details may be found in [1].

CHAPTER 3

NUMERICAL METHODS FOR ESTIMATION OF VELOCITY FROM MODAL ANALYSIS

For an elastic continuum such as a flexible beam, undergoing a transversal motion, total vibration of the system is a sum of the individual mode vibrations. Therefore, to control the vibrations of the system, individual mode vibration has to be controlled. This objective can be achieved, if we feedback individual mode velocities, with proper control gain. For this, we need to have the mode velocities. Since the mode shapes representing the system are orthogonal, we can get the individual mode velocity by integrating the product of system velocity and the mode shape function over the interval of beam length. That is, since

$$\int_0^l r(x)\, X_i(x)\, X_j(x)\, dx = \delta_{ij}$$

$$\delta_{ij} = \{ \begin{array}{ll} 1 & i = j \\ 0 & i \neq j \end{array}$$

Thus, the mode velocity is obtained as

$$\dot{\Phi}_i = \int_0^l r(x)\dot{y}(x,t)X_i(x)\, dx \tag{3.1}$$

where $\dot{\Phi}_i$ is velocity of the i th mode,

$\dot{y}$ is velocity of the system at a distance x on the beam

X_i is mode shape.

$r(x)$ is the weighting function given by

$$r(x) = 1 + K\, l\, \delta(x - l) \tag{3.2}$$

where K is the ratio of end mass to end mass

l is the length of the beam

$\delta(x - l)$ is the impulse function

By substituting equation (3.2) in equation (3.1), we will get

$$\dot{\Phi}_i = \int_0^l \dot{y}(x,t)\, X_i(x)\, dx + 2\, K\, l\, \dot{y}(l,t) \qquad (3.3)$$

The integral given by (3.3) is evaluated, using standard numerical methods.

The numerical methods used in this work, to estimate $\int_0^l \dot{y}(x,t)\, X_i(x)\, dx$ are 1) Euler's Rule and 2) Simpson's Rule.

These two methods are now described below.

1. Euler's Estimation:

$$\int_0^l \dot{y}(x,t)\, X_i(x)\, dx = \sum_{k=1}^{N_s} \dot{y}(kh)\; X_i(kh)\, h \qquad (3.4)$$

where N_s is the number of sampling points, in other words, the number of sensors

h is the sampling interval given by l/N_s.

2. Simpson's Estimation:

$$\int_0^l \dot{y}(x,t)\, X_i(x)\, dx = \frac{h}{3}\, (\dot{y}(0,t) X_i(0) + \sum_{n=1,3,5} 4\, \dot{y}(nh,t)\, X_i(nh)$$
$$+ \sum_{n=2,4,6} 2\, \dot{y}(nh,t)\, X_i(nh) + \dot{y}(N_s\, h,t)\, X_i(N_s\, h)) \qquad (3.5)$$

where N_s is the number of sampling points, in other words, number of sensors, an even integer

h is the sampling interval given by l/N_s.

Let the estimated mode velocities be $\dot{\Psi}_i(t)$.

Using Euler's Estimation:

$$\dot{\Psi}_i(t) = \sum_{k}^{N_s} \dot{y}_i(kh)\; X_i(kh) + 2\, K\, l\, \dot{y}(l,t) \qquad (3.6)$$

where $i = 1, 2, \cdots, N_o$

It is to be noted that N_o is the number of modes to be controlled and $\dot{y}(l,t)$ is the velocity of the beam end.

Using Simpson's Estimation

$$\dot{\Psi}_i(t) = \frac{h}{3}\,(\dot{y}(0,t)\,X_i(0) + \sum_{n=1,3,5} 4\,\dot{y}(nh,t)\,X_i(nh)$$

$$+ \sum_{n=2,4,6} 2\,\dot{y}(nh,t)\,X_i(nh) + \dot{y}(N_s\,h,t)\,X_i(N_s h)) + 2\,K\,l\,\dot{y}(l,t) \qquad (3.7)$$

In either cases, the derivation of $\dot{\Psi}_i(t)$ requires the knowledge of $\dot{y}(x,t)$ and $X_i(x)$, for $i = 1, 2, \cdots, N_o$, at locations $x = kh$, $k = 1, 2, \cdots, N_s$.

The system velocity $\dot{y}(x,t)$ can be readily determined using N_s rate sensors placed on the beam, whereas the mode shapes $X_i(x)$ are generally determined mathematically. In the case of a flexible beam, the mode shapes are given by (2.9).

To reduce implementational complexity, it is desired that N_s, the number of rate sensors, be kept low.

An estimate of the lower bound of N_s is now carried out.

Since $\dot{\Phi}_i = \int_0^l \dot{y}(x,t)\,X_i(x)\,dx + 2\,K\,l\,\dot{y}(l,t)$

and $\dot{\Psi}_i = NumInt_x \sum_k[\dot{\Phi}_k(t)\,X_k(x)\,X_i(x)] + 2\,K\,l\,\dot{y}(l,t)$

where $NumInt_x$ denotes the numerical integration of $\sum_k[\dot{\Phi}_k(t)\,X_k(x)\,X_i(x)]$ with respect to x.

$$\dot{\Psi}_i = NumInt_x \sum_k [X_k(x)\,X_i(x)]\,\dot{\Phi}_k(t) + 2\,K\,l\,\dot{y}(l,t) \qquad (3.8)$$

Denote $m_{ij} = NumInt_x\,[X_i(x)\,X_j(x)]$, then

$$\dot{\Psi}_i(t) = \begin{bmatrix} m_{i1} & m_{i2} & m_{i3} & \cdots & m_{iN_m} \end{bmatrix} \begin{bmatrix} \dot{\Phi}_1(t) \\ \dot{\Phi}_2(t) \\ \dot{\Phi}_3(t) \\ \vdots \\ \dot{\Phi}_{N_m}(t) \end{bmatrix} + 2\,K\,l\,\dot{y}(l,t)$$

let

$$\dot{\Psi}(t) = \begin{bmatrix} \dot{\Psi}_1(t) & \dot{\Psi}_2(t) & \dot{\Psi}_3(t) & \cdots & \dot{\Psi}_{N_o}(t) \end{bmatrix}$$

$$\dot{\Phi}(t) = \begin{bmatrix} \dot{\Phi}_1(t) & \dot{\Phi}_2(t) & \dot{\Phi}_3(t) & \cdots & \dot{\Phi}_{N_m}(t) \end{bmatrix}$$

$$\epsilon(t) = \begin{bmatrix} \dot{\Psi}_1(t) - \dot{\Phi}_1(t) & \dot{\Psi}_2(t) - \dot{\Phi}_2(t) & \cdots & \dot{\Psi}_{N_o}(t) - \dot{\Phi}_{N_o}(t) \end{bmatrix}$$

then

$$e(t) = M\,\dot{\Phi}(t) - I\,\Phi(t) = (M - I)\,\Phi(t)$$

$$M = [m_{ij}] \in R^{N_o \times N_m}$$

$$I = [\delta_{ij} - 4\,K\,l] \in R^{N_o \times N_m} \qquad (3.9)$$

The estimation error bound can now be determined by computing the spectral norm of $M - I$ as follows:

$$Sup_{\Phi \neq 0} \frac{||e||}{||\dot{\Phi}||} = Sup_{||\dot{\Phi}||=1} \frac{||[M - I]\,\Phi||}{||\dot{\Phi}||} = \sigma_s(M - I)$$

where σ_s = maximum singular value of $(M - I)$

Computation of $\sigma_s(M - I)$ entails the following steps:

1. Determine N_m, N_o, N_s
2. Calculate M matrix according to equation (3.9)
3. Form the matrix $M - I$
4. Apply standard singular value decomposition techniques to obtain the maximum singular value.

For example, the I matrix and the M matrix for 4 modes, 4 sensors for controlling 2 modes are calculated using Euler's approximation and are given below.

Given $N_m = 4$, $N_o = 2$, $N_s = 4$, $l = 133$.

Using the frequencies given in Table (2.1), the values of mode shapes are calculated using the formula (2.9) and are given in Table (3.1).

For Euler's estimation, the values of m_{ij} are calculated using Equation 3.9 as follows:

$$M_w = \begin{bmatrix} 1.0049 & .0056 & .0075 & .0103 \\ .0056 & 1.0045 & .0051 & .0067 \end{bmatrix}$$

$$\delta_{ij} = \begin{bmatrix} 1.0 & 0 & 0 & 0 \\ 0 & 1.0 & 0 & 0 \end{bmatrix}$$

Table 3.1 Mode Shape Values at 4 Sensors Placed at Equidistant on the Beam

x	$X_1(x)$	$X_2(x)$	$X_3(x)$	$X_4(x)$
33.5	-.0867	.1225	-.0864	-.0002
66.5	-.1214	-.0008	.1226	.0005
97.5	-.0828	-.1221	-.0828	-.0007
133.0	.0072	.0037	.0025	.0019

maximum singular value of $(M - I) = .0184$.

By increasing N_s, we can make M matrix approach I matrix. Therefore, the maximum singular value of $(M - I)$ will approach zero, as M approaches I matrix.

For the purpose of comparison, the maximum singular values of $(M - I)$ for different number of sensors are calculated using Euler's and Simpson's approximation and summarized in the tables (3.2 thru 2.7). These values are plotted and are shown In Figures (3.1 thru 2.6).

For example, in a system containing 4 modes, suppose we want to control

- 2 modes

  1. For Euler's approximation From Table(3.1) and Figure(3.1), we can see a steep decline in maximum singular value at 4 sensors, so we need 4 sensors for good estimation.

  2. For Simpson's approximation From Table (3.4) and Figure(3.4), it can be shown we need 6 sensors for good estimation.

- 4 modes

  1. For Euler's approximation From Table(3.1) and Figure(3.1), we can see a steep decline in maximum singular value at 6 sensors, so we need 6 sensors for good estimation.

  2. For Simpson's approximation From Table (3.4) and Figure(3.4), it can be shown that we need 8 sensors for good estimation.

By comparing,Tables (3.2) and (3.5), (3.3) and (3.6), (3.4) and (3.7) we can see Euler's approximation gives fewer number of sensors for a given number of modes. This is because of the nature of the mode shapes.

But, if the number of sensors is high, Simpson's rule gives good approximation compare to Euler's approximation. For example, for a system containing 8 modes, with 14 sensors, the maximum singular value

1. using Euler's approximation is .0030 from Table (3.3)

2. using Simpson's approximation is $.6177 \times e^{-3}$ from Table (3.6).

From the tabulated data we can form an empirical formula for calculating number of sensors as follows.

1. for Euler's estimation

$$N_s = 1.28 * N_m - .7 * (N_m - N_o)$$

2. for Simpson's estimation

$$N_s = 2.3 * N_m - (N_m - N_o)$$

where N_s is number of sensors

N_m is number of modes present in the system

N_o is number of modes we want to control

Mode velocities $\dot{\Phi}$ and estimated mode velocities $\dot{\Psi}$ are plotted for different number of modes and sensors using Euler's approximation and Simpson's approximation and are shown in figures (3.7) thru (3.12). From these figures, we can see the estimation is good when the maximum is singular value of $(M - I)$ matrix is should be around 0.01 or less.

Table 3.2 Maximum Singular Values of (M-I) Matrix of a System Containing 4 Modes, Using Euler's Estimation.

	Maximum Singular Value	
N_s	$N_o = 2$	$N_o = 4$
2	.998	1.0
4	.0184	.9977
6	.0077	.0105
8	.0044	.0059
10	.0029	.0038

Table 3.3 Maximum Singular Values of (M-I) Matrix of a System Containing 8 Modes, Using Euler's Estimation

	Maximum Singular Value			
N_s	$N_o = 2$	$N_o = 4$	$N_o = 6$	$N_o = 8$
4	.9999	1.0	1.0	1.0
6	.0199	.9984	1.0	1.0
8	.0098	.0112	.0129	.9984
10	.006	.0069	.0078	.0092
12	.0041	.0048	.0054	.0063
14	.0030	.0035	.0039	.0046
16	.0023	.0027	.0030	.0035
18	.0019	.0021	.0024	.0028

Table 3.4 Maximum Singular Values of (M-I) Matrix of a System Containing 20 Modes, Using Euler's Estimation

	Maximum Singular Value			
N_s	$N_o = 2$	$N_o = 4$	$N_o = 6$	$N_o = 20$
8	1.4142	1.4142	1.4142	2.0001
10	1.0	1.0	1.0	1.0001
12	.0224	.9986	1.0	1.0
14	.0132	.0143	.0149	1.0
16	.0092	.0099	.0103	1.0
18	.0069	.0075	.0077	1.0
20	.0054	.0059	.0061	.9999
22	.0044	.0048	.0049	.0066
24	.0037	.0040	.0041	.0054
26	.0031	.0034	.0035	.0046
28	.0027	.0029	.003	.0039
30	.0023	.0025	.0026	.0034
32	.0020	.0022	.0023	.0030
36	.0016	.0017	.0018	.0024
40	.0015	.0014	.0015	.0019
42	.0012	.0013	.0013	.0016
44	.0011	.0012	.0012	.0016

Table 3.5 Maximum Singular Values of (M-I) Matrix of a System Containing 4 Modes, Using Simpson's Estimation.

	Maximum Singular Value	
N_s	$N_o = 2$	$N_o = 4$
2	1.3508	1.6460
4	.3381	.9977
6	.3344	.3366
8	.7156e-3	.3339
10	.2143e-3	.5087e-3

Table 3.6 Maximum Singular Values of (M-I) Matrix of a System Containing 8 Modes, Using Simpson's Estimation

	Maximum Singular Value			
N_s	$N_o = 2$	$N_o = 4$	$N_o = 6$	$N_o = 8$
4	1.3690	1.3744	1.6620	1.6667
6	.4714	1.1026	1.1054	1.1870
8	.3338	.3352	.3366	.9984
10	.3334	.3343	.3352	.3360
12	.0015	.3336	.3341	.3347
14	.6177e-3	.7986e-3	.3335	.334
16	.3256e-3	.3800e-3	.5548e-3	.3335
18	.2014e-3	.2263e-3	.2732e-3	.4606e-3

Table 3.7 Maximum Singular Values of (M-I) Matrix of a System Containing 20 Modes, Using Simpson's Estimation

	Maximum Singular Value			
N_s	$N_o = 2$	$N_o = 4$	$N_o = 6$	$N_o = 20$
8	1.4907	1.9136	1.0136	2.9988
10	1.1056	1.1056	1.3744	1.6668
12	.4714	1.1037	1.3737	1.6667
14	.4714	.4714	.4714	1.1872
16	.4714	.4714	.4714	1.1872
18	.4714	.4714	.4714	1.1872
20	.3334	.3335	.3337	.9999
22	.3346	.3334	.3326	.3346
24	.0030	.3333	.3334	.3345
26	.0016	.0018	.3334	.3342
28	.0010	.0011	.0012	.3340
30	.6793e-3	.7376e-3	.7768e-3	.3338
32	.4778e-3	.5171e-3	.5390e-3	.3337
36	.2710e-3	.2922e-3	.3019e-3	.3335
40	.1873e-3	.2015e-3	.2076e-3	.3334
42	.1678e-3	.1804e-3	.1860e-3	.3010
44	.1566e-3	.1683e-3	.1737e-3	.1952e-3

For example, using Euler's approximation, the maximum singular value for a system containing 4 modes, for $N_s = 4$, $N_o = 4$ is 0.9977, $N_s = 6$, $N_o = 4$ is 0.0105.

From Figures (3.7) and (3.8), we can see the error in the estimation of third and fourth modes is minimized when N_s is increased from 4 to 6.

For Euler's approximation:

1) Figure(3.7) shows the velocities of first four modes and their estimated values for a system containing 4 modes and 3 sensors.

2) Figure(3.8) shows the velocities of first four modes and their estimated values for a system containing 4 modes and 6 sensors.

3) Figure(3.9) shows the velocities of first four modes and their estimated values for a system containing 8 modes and 10 sensors.

4) Figure(3.10) shows the velocities of first four modes and their estimated values for a system containing 20 modes and 22 sensors.

For Simpson's approximation:

1) Figure(3.11) shows the velocities of first four modes and their estimated values for a system containing 4 modes and 10 sensors.

2) Figure(3.12) shows the velocities of first four modes and their estimated values for a system containing 8 modes and 18 sensors.

3) Figure(3.13) shows the velocities of first four modes and their estimated values for a system containing 20 modes and 42 sensors.

In all cases, excellent agreement between $\dot{\Phi}$ and $\dot{\Psi}$ is observed.

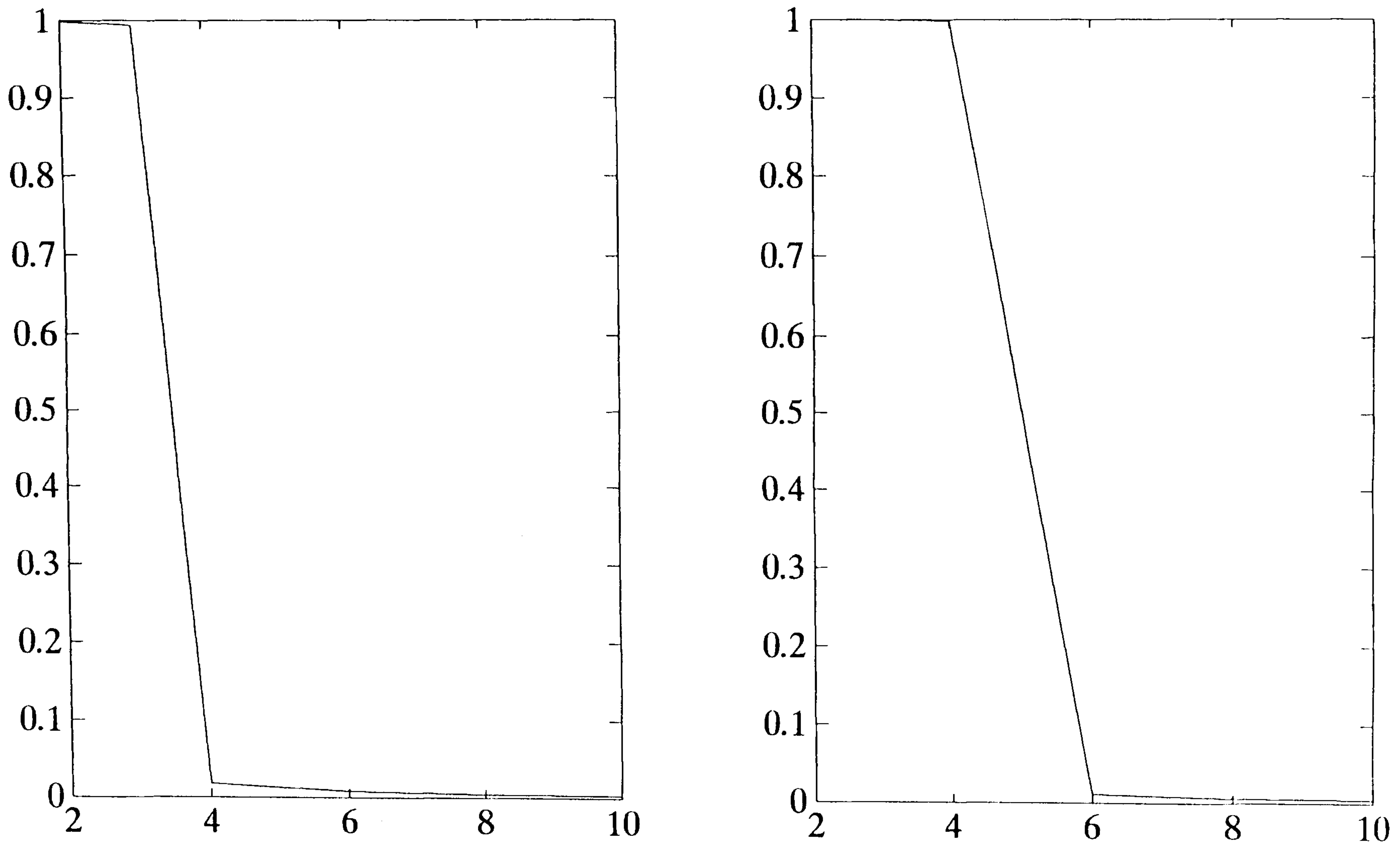

Figure 3.1 Maximum Singular Values vs Number of Sensors for $N_m = 4$, $N_o = 2, 4$, Using Euler's Approximation

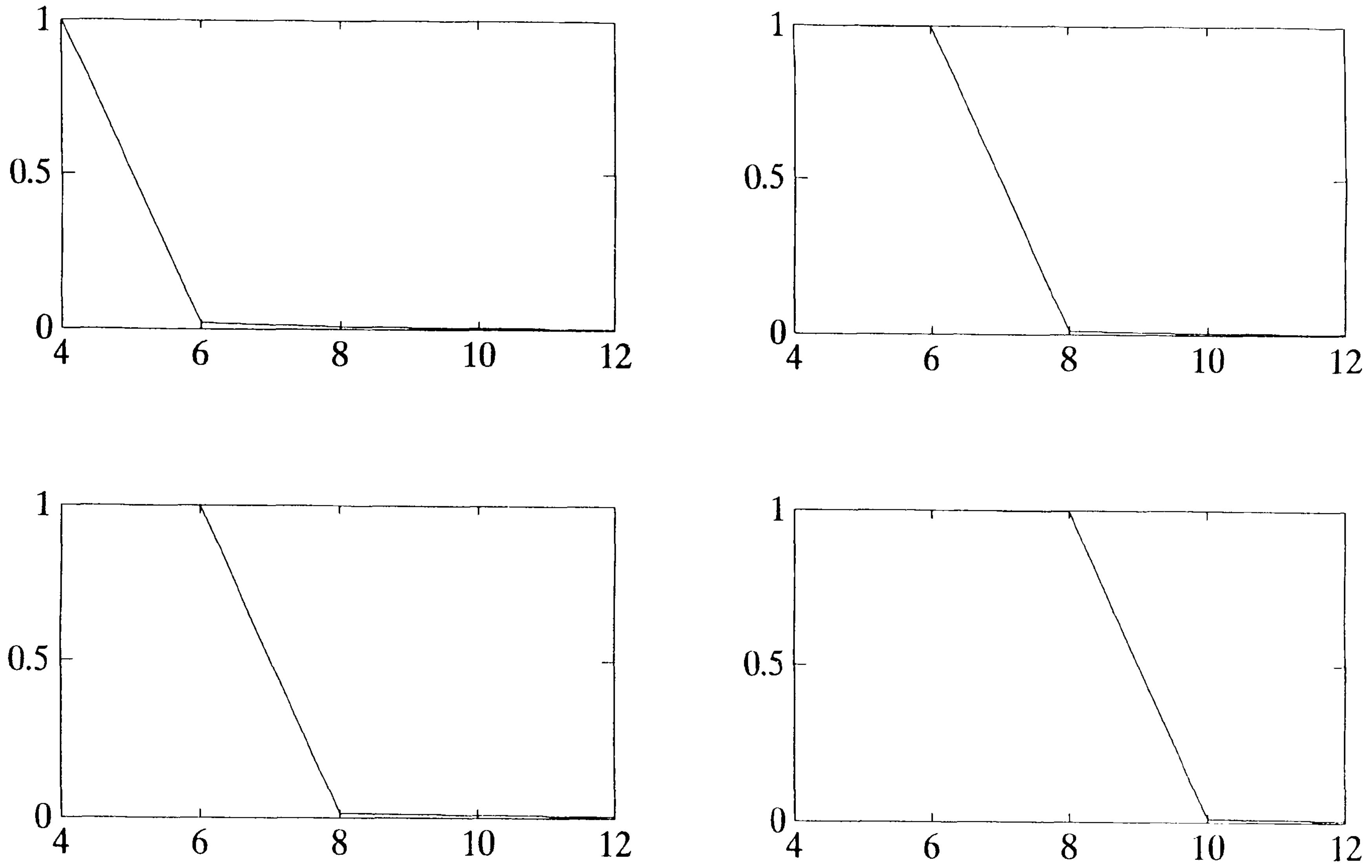

Figure 3.2 Maximum Singular Values vs Number of Sensors for $N_m = 8$, $N_o = 2, 4, 6, 8$, Using Euler's Approximation

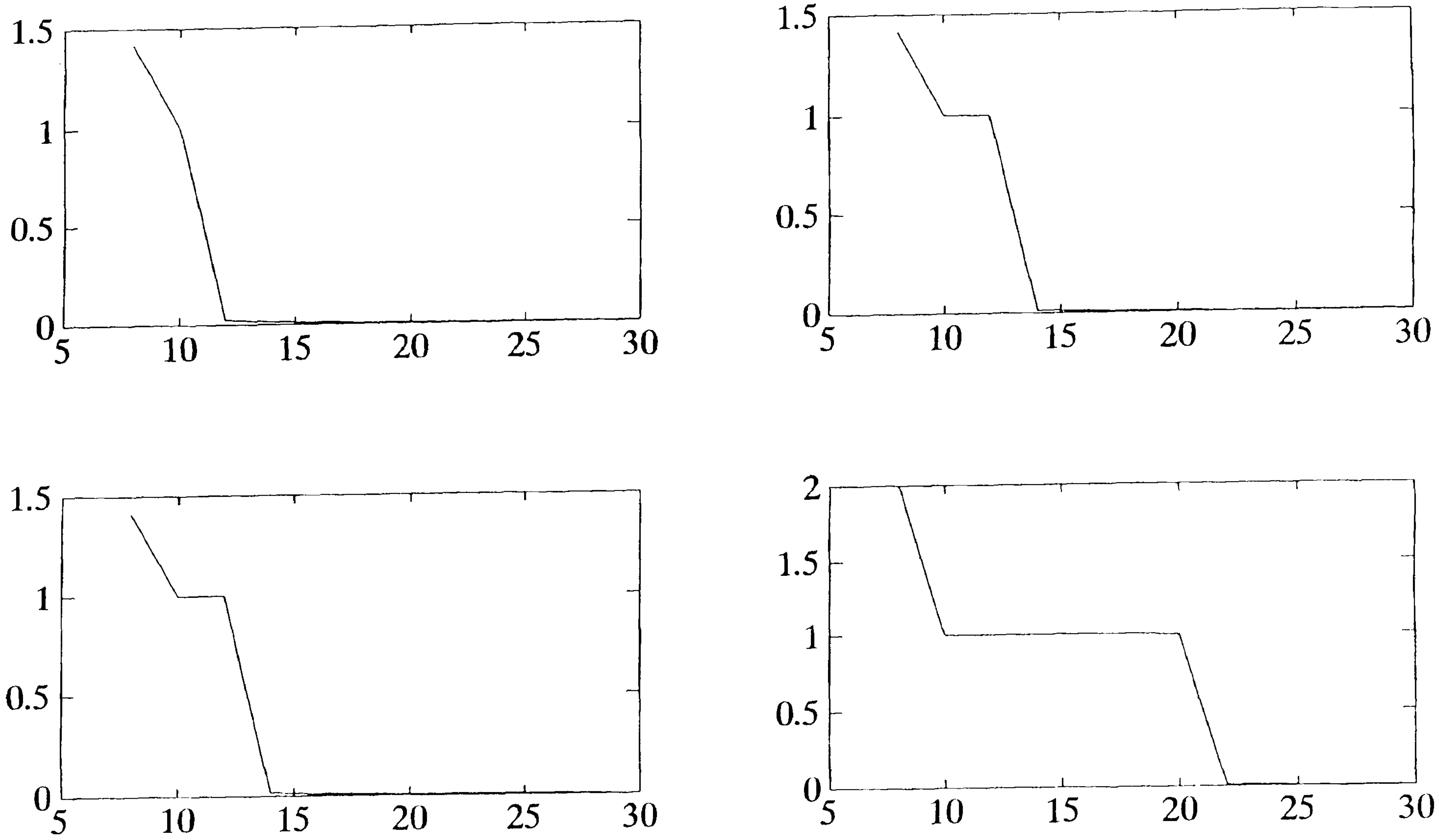

Figure 3.3 Maximum Singular Values vs Number of Sensors for $N_m = 20$, $N_o = 2, 4, 6, 20$, Using Euler's Approximation

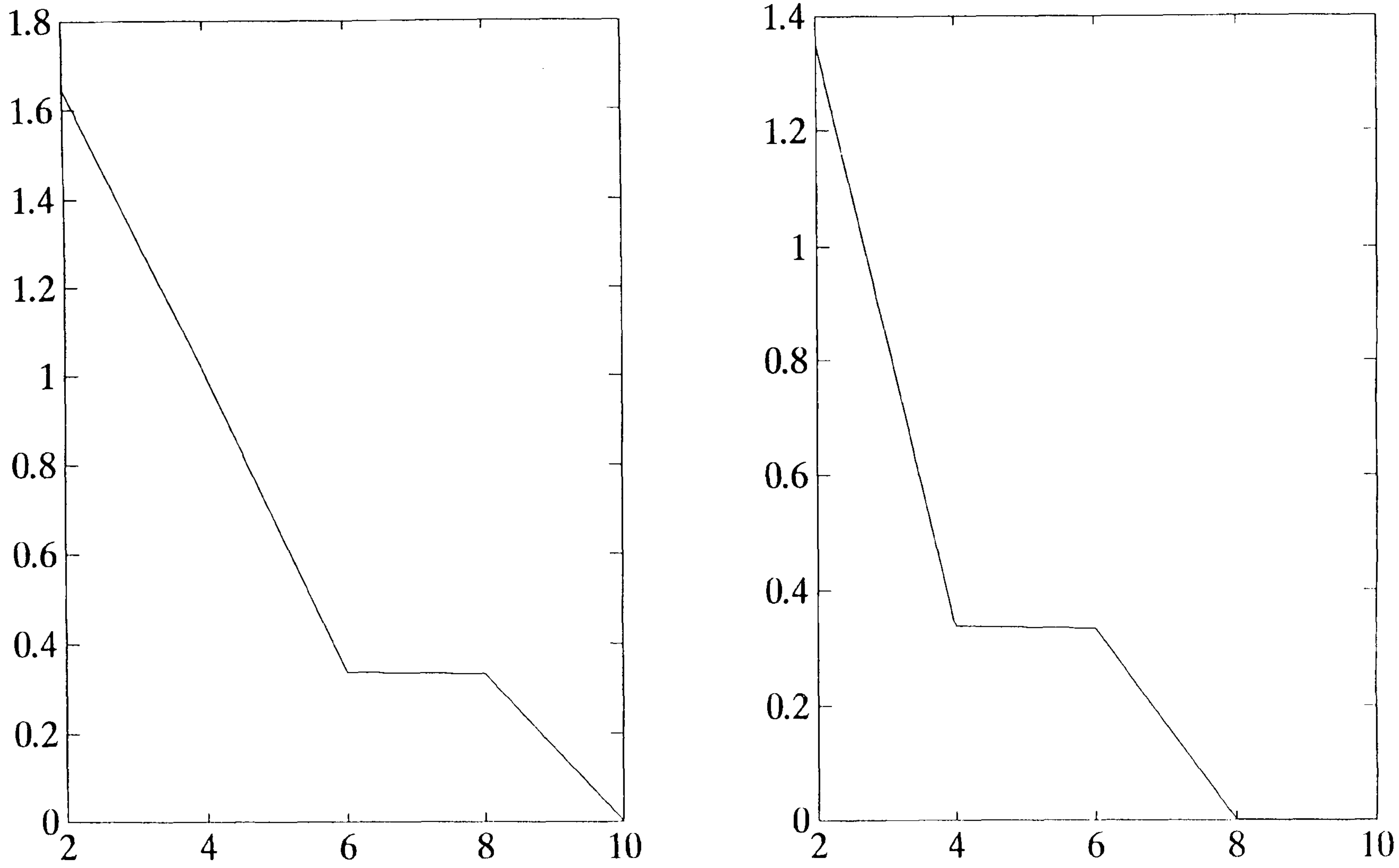

Figure 3.4 Maximum Singular Values vs Number of Sensors for $N_m = 4$, $N_o = 2, 4$, Using Simpson's Approximation

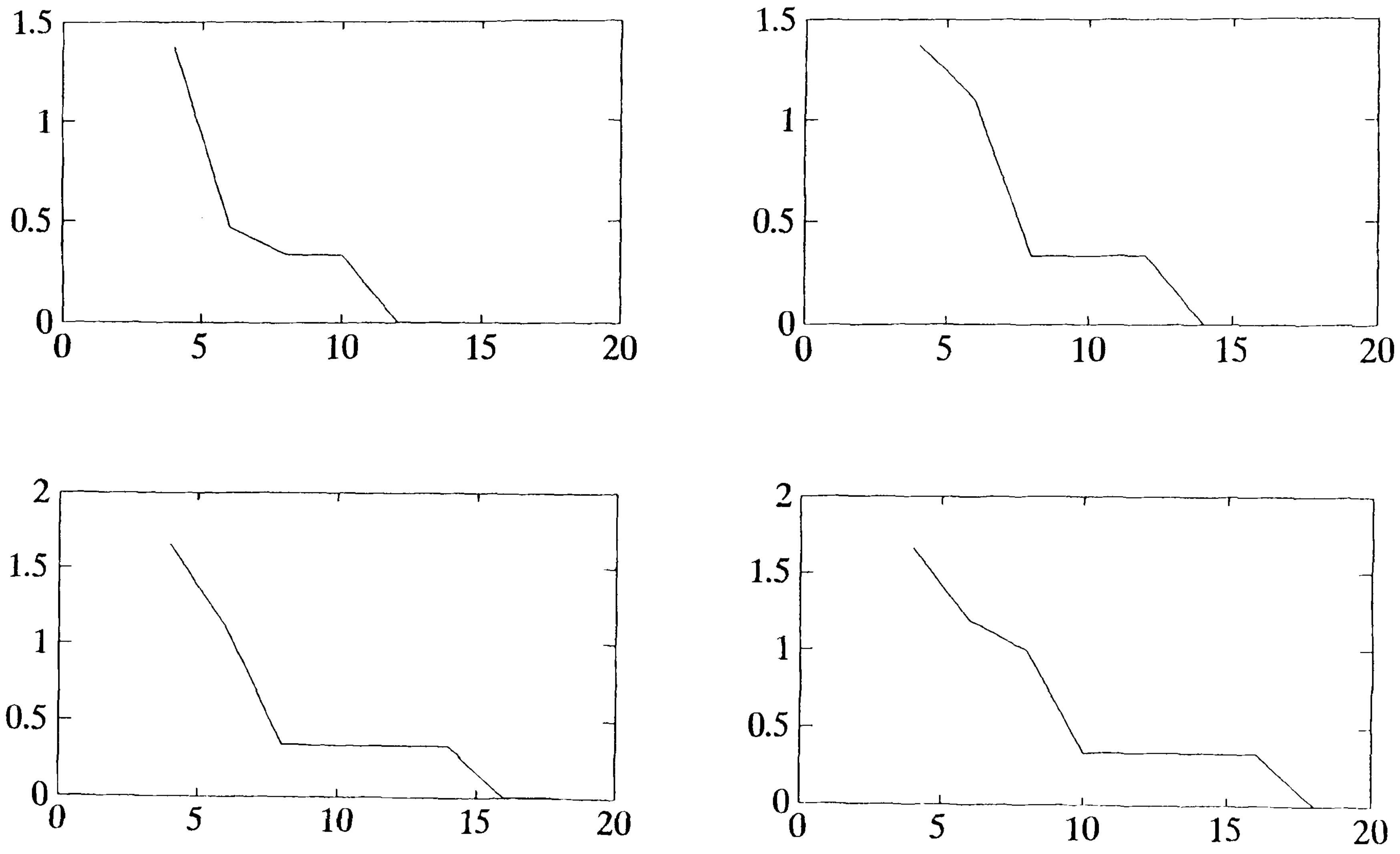

Figure 3.5 Maximum Singular Values vs Number of Sensors for $N_m = 8$, $N_o = 2, 4, 6, 8$, Using Simpson's Approximation

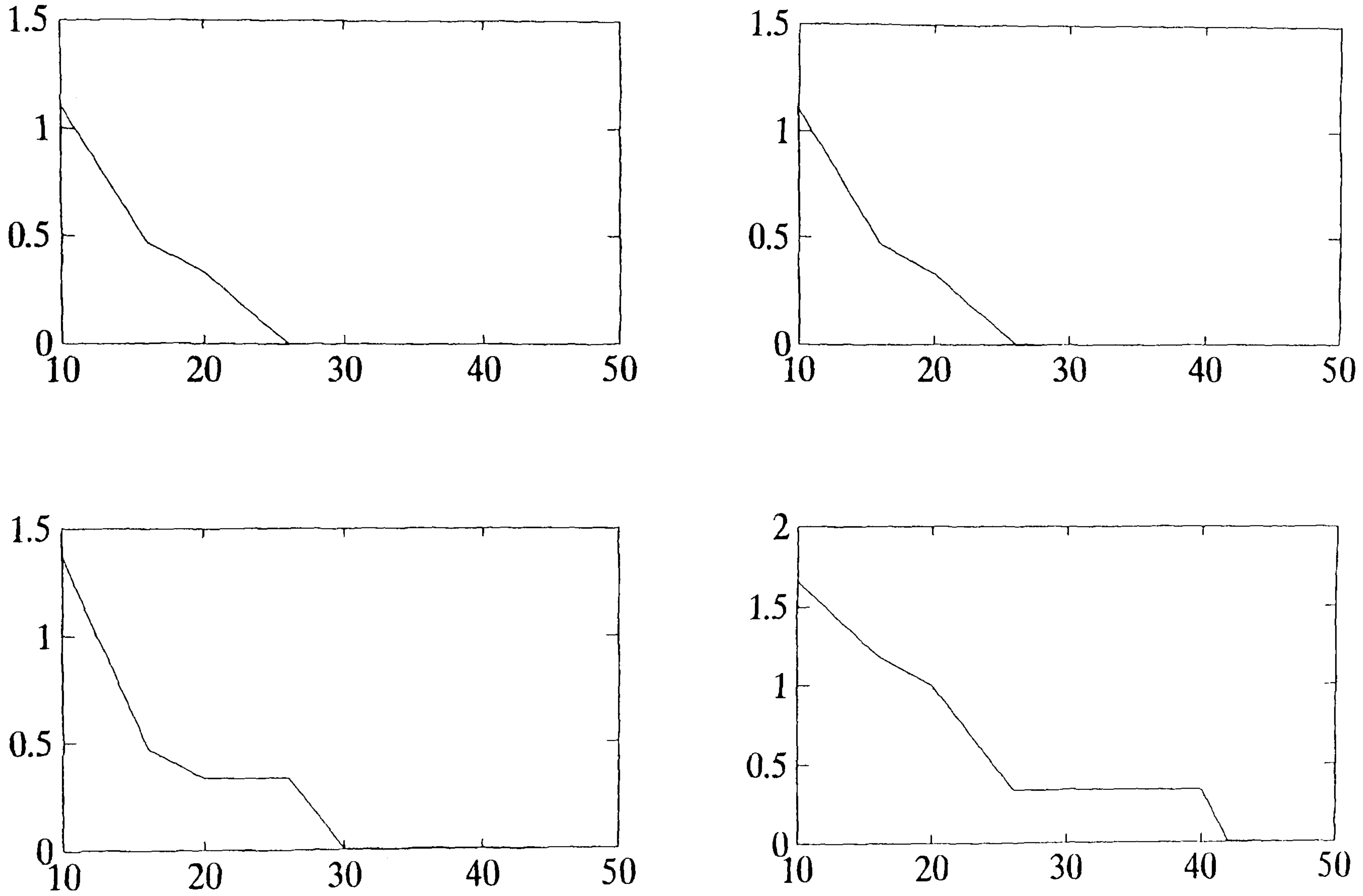

Figure 3.6 Maximum Singular Values vs Number of Sensors for $N_m = 20$, $N_o = 2, 4, 6, 20$, Using Simpson's Approximation

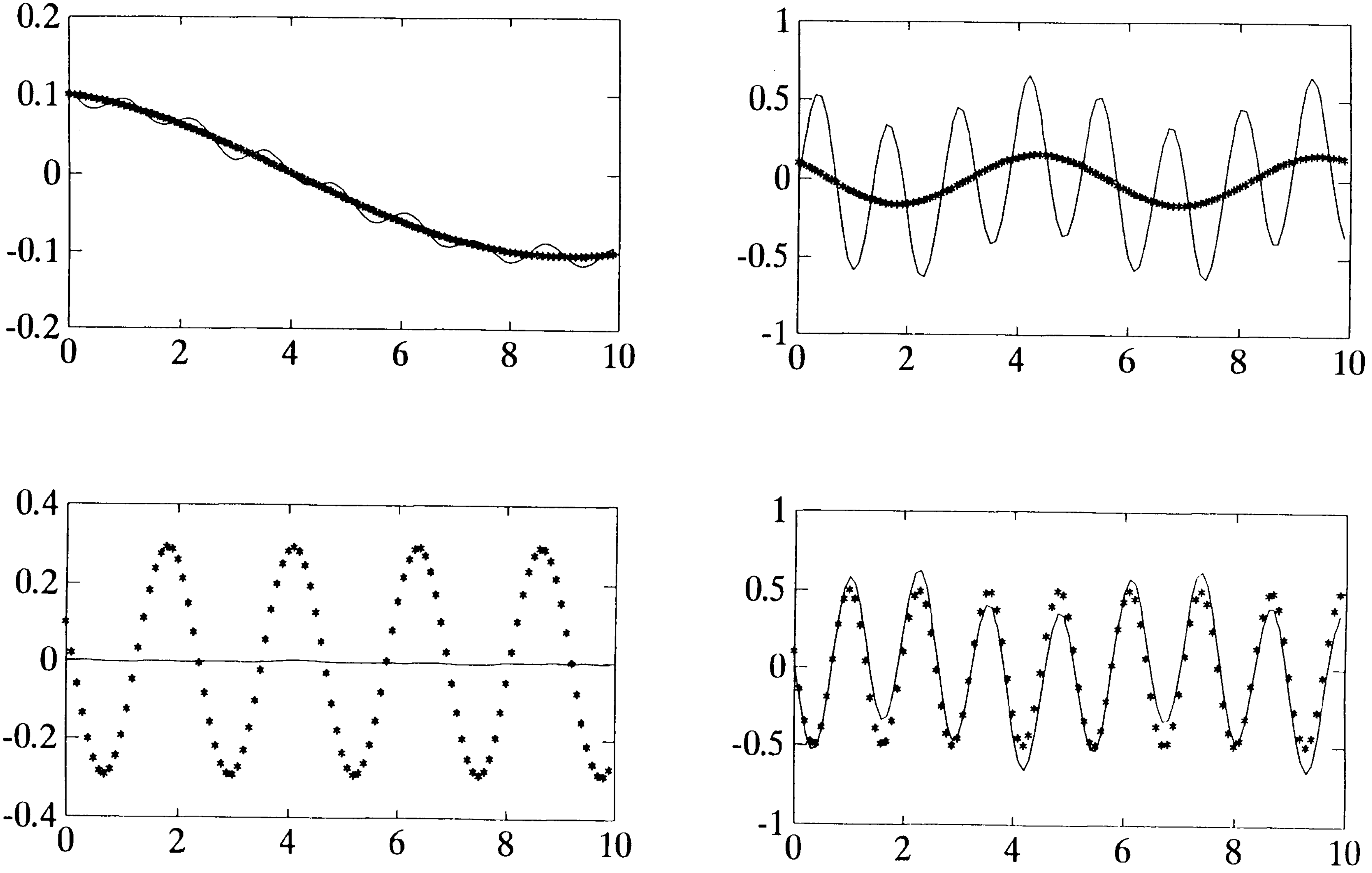

Figure 3.7 Velocities of First 4 Modes and Their Estimated Values for $N_m = 4$, $N_s = 3$, Using Euler's Approximation

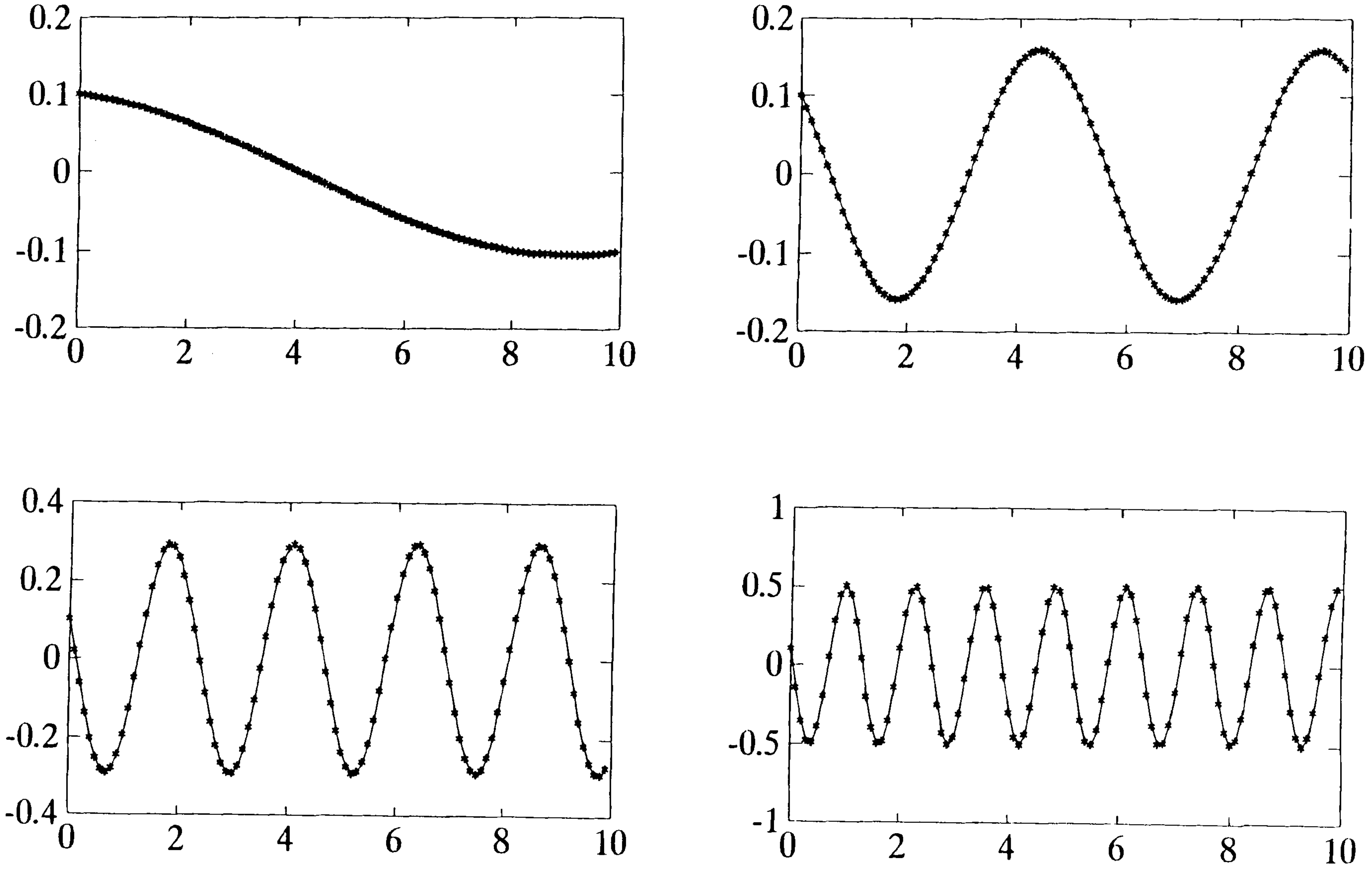

Figure 3.8 Velocities of First 4 Modes and Their Estimated Values for $N_m = 4$, $N_s = 6$, Using Euler's Approximation

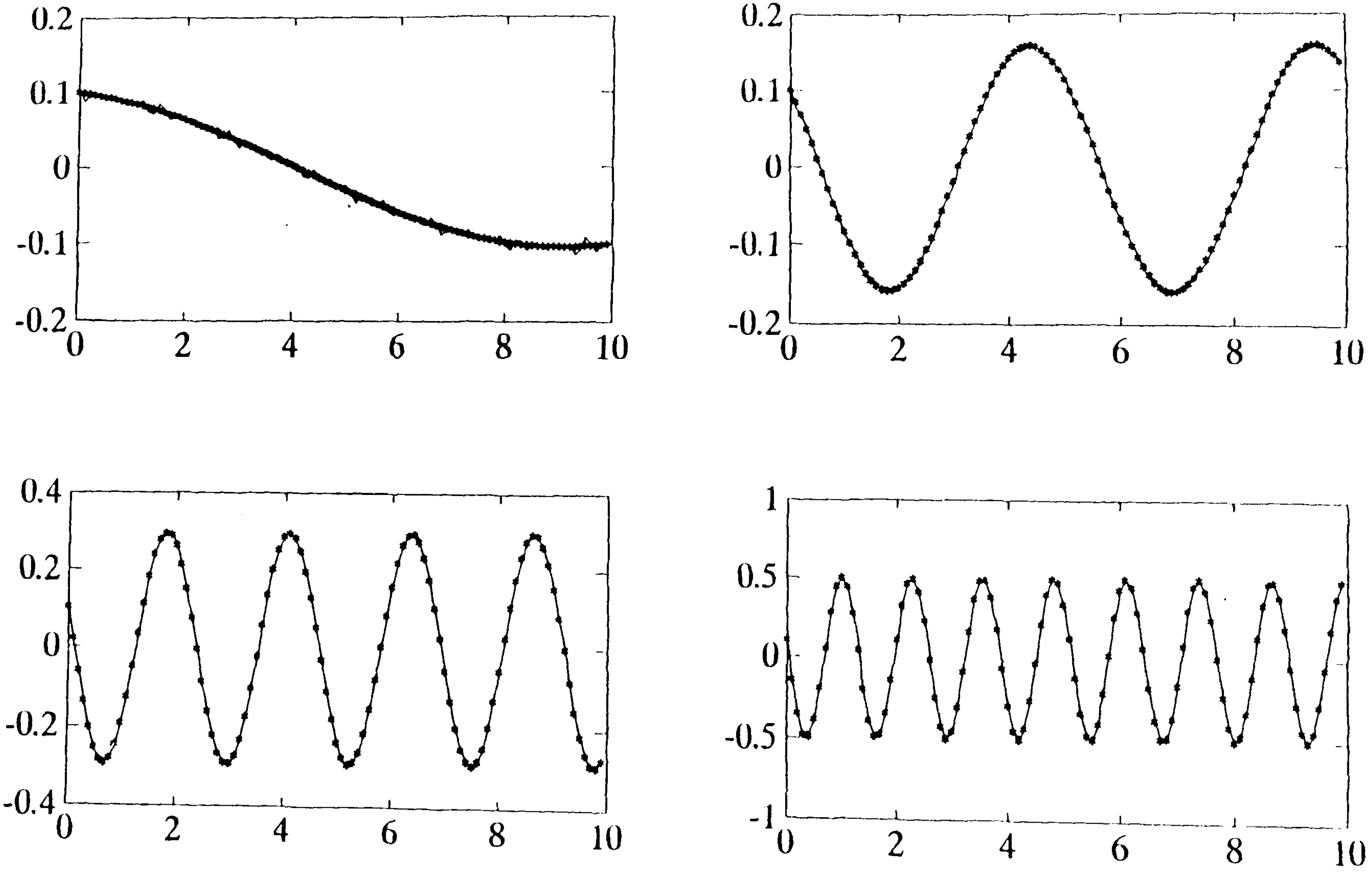

Figure 3.9 Velocities of First 4 Modes and Their Estimated Values for $N_m = 8$, $N_s = 10$, Using Euler's Approximation

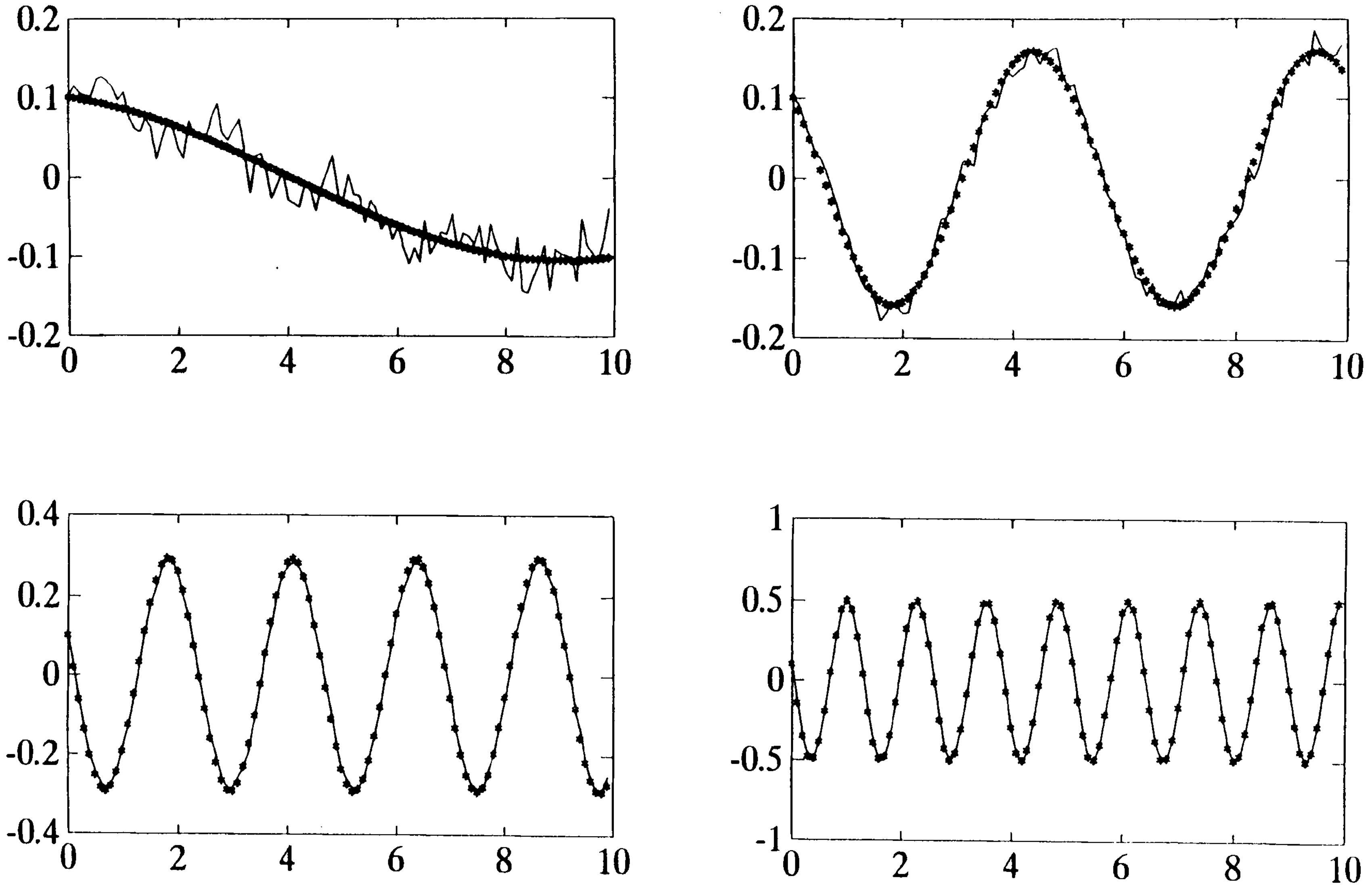

Figure 3.10 Velocities of First 4 Modes and Their Estimated Values for $N_m = 20$, $N_s = 22$, Using Euler's Approximation

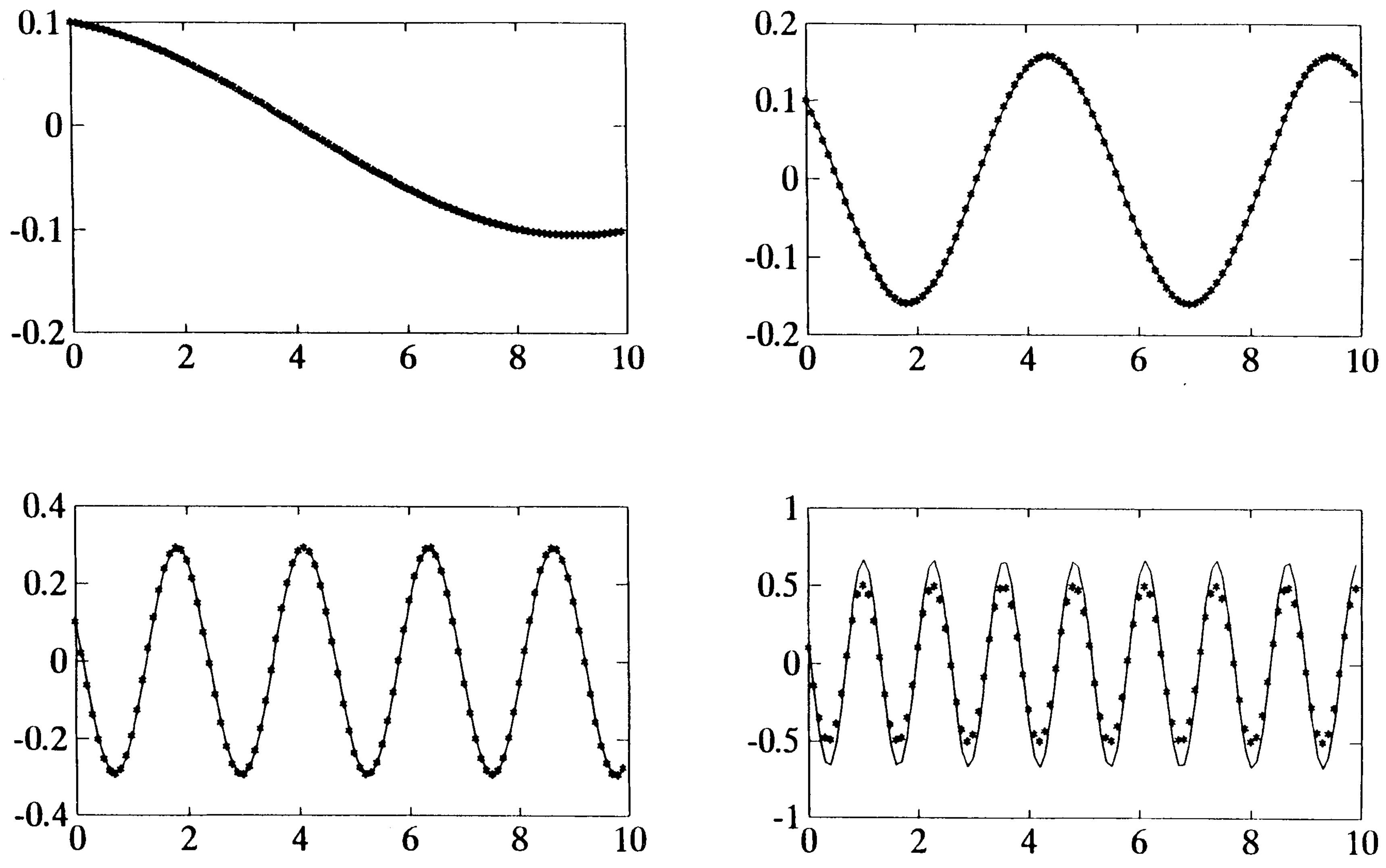

Figure 3.11 Velocities of First 4 Modes and Their Estimated Values for $N_m = 4$, $N_s = 10$, Using Simpson's Approximation

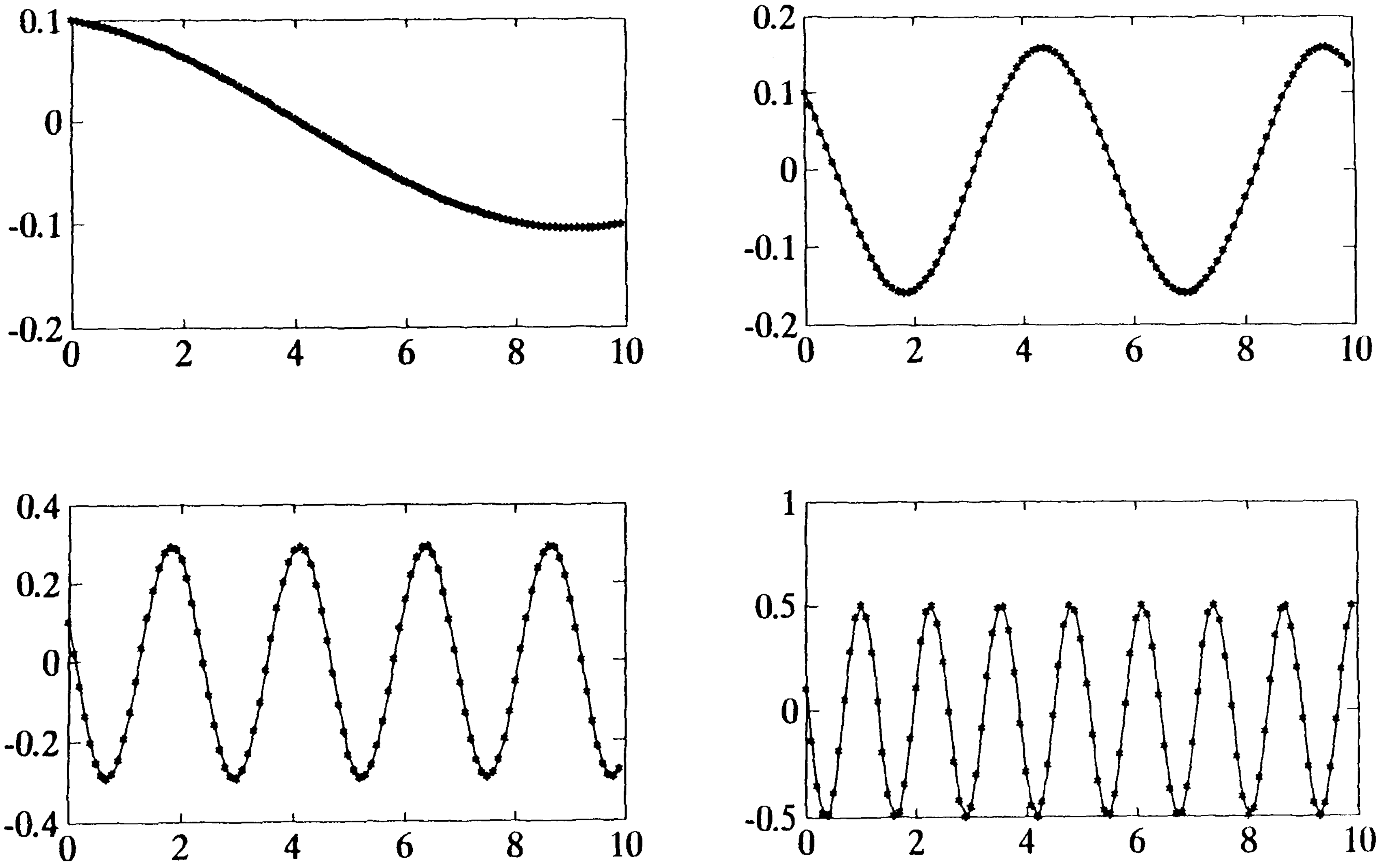

Figure 3.12 Velocities of First 4 Modes and Their Estimated Values for $N_m = 8$, $N_s = 18$, Using Simpson's Approximation

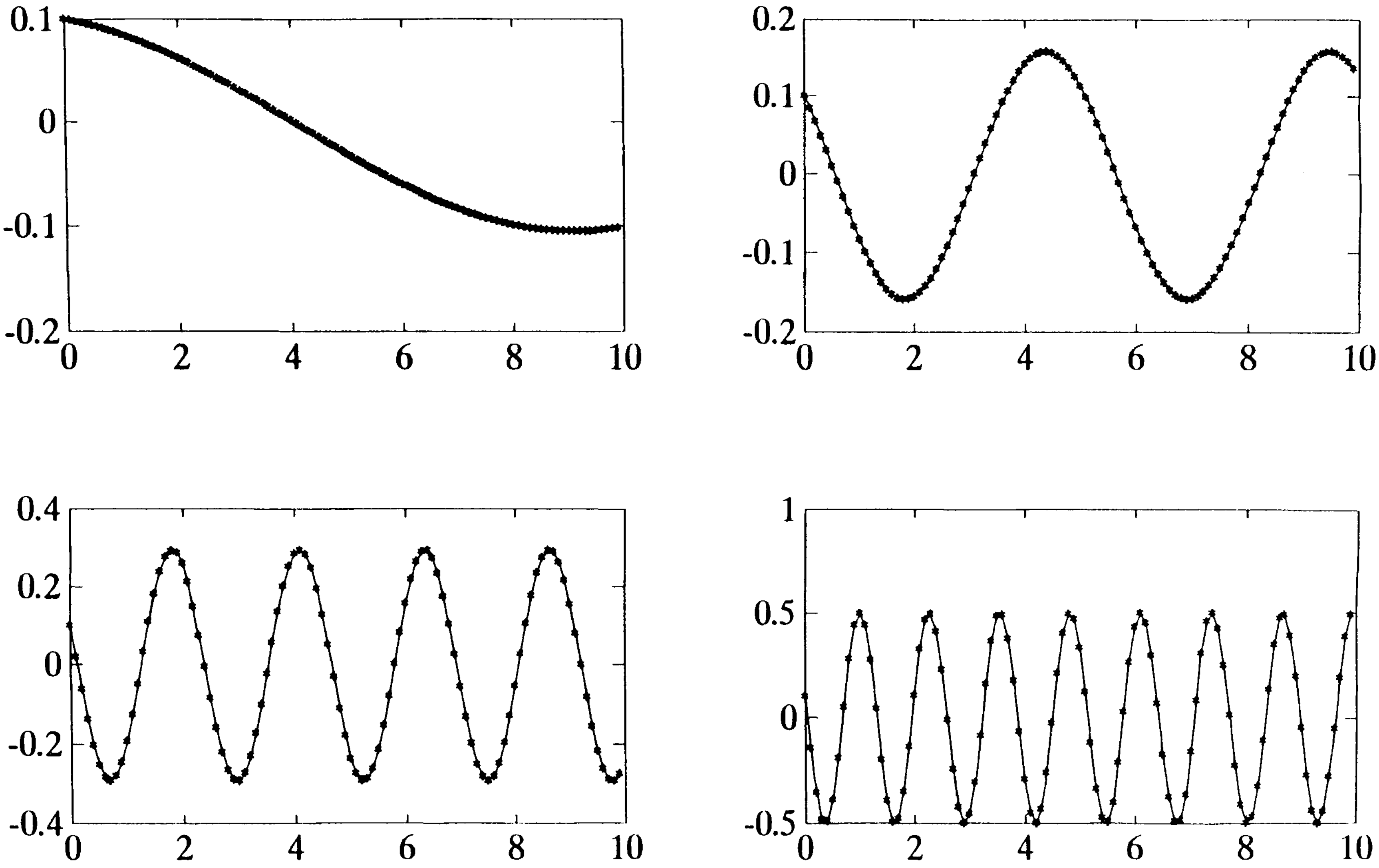

Figure 3.13 Velocities of First 4 Modes and Their Estimated Values for $N_m = 20$, $N_s = 42$, Using Simpson's Approximation

CHAPTER 4

CONTROL AND REGULATION OF FLEXIBLE BEAM

In chapter 1, the dynamics of a flexible beam undergoing transversal motion has been modelled. The equation of motion is given by (2.1). It has been shown that the flexible beam dynamics are equivalent to N_m second order modules connected in parallel. Although these second order modules are reachable by the same control input (e.g. the torque generated by a *dc* motor), they may be decoupled in the frequency domain on observing that each second order substructure occupies a unique passband with center frequency ω_i given by Table(2.1). The control system described in [2] exploits this property to provide stabilization and regulation of each second order substructure based on applying N_o structurally identical control module to the flexible structure. A critical requirement in [2] is that the individual mode velocities $(\dot{\Phi}_i)$ be available, by means of bandpass filtering of the beam velocity.

In this thesis, the mode separation method described in Chapter 2 is used in lieu of the bandpass filtering operation to generate the mode velocity estimates $\dot{\Psi}_i$. The quantitative effects of using $\dot{\Psi}_i$ rather than $\dot{\Phi}_i$ are now explored by means of numerical simulation.

The following control objectives are now described:

1. Stabilization: Velocity of each mode $\dot{\Phi}_i(t)$ goes to zero as t tends to infinity, for $i = 1, 2, \cdots N_o$

2. Regulation: Let amp$(\dot{\Phi}_i(t))$, be the amplitude of $(\dot{\Phi}_i(t))$ and $e_i(t) = amp(\dot{\Phi}_i(t)) - \dot{\Phi}_i^{ref}$ be the regulation error. For regulation purposes, it is desired that

$$\lim_{t \to \infty} e_i(t) = 0 \quad for\ i = 1,\ 2,\ \cdots N_o \tag{4.1}$$

Stabilization is a special case of regulation by setting $\dot{\Phi}_i^{ref} = 0$.

4.1 Controller Synthesis

In order to increase the damping of the flexible beam, each velocity component $\dot{\Phi}_i$ has to be estimated; and then fed back to the system using proper gain. A scheme for estimating the mode velocities has been described in Chapter 2.

Assuming there is no or little damping present in the system, our aim is to stabilize the required modes, using the mode velocity. We are introducing damping into the system, by feeding back the velocity component, and therefore, the structure corresponding to each mode can be described as

$$\ddot{\Phi}_i + \omega_i^2 \, \Phi_i = -K \, \dot{\Phi}_i \tag{4.2}$$

where the $-K \, \dot{\Phi}_i$ terms are to be synthesized by active feedback control. The overall control system to control a flexible beam with N_m modes, N_o outputs and N_s sensors are shown in Figure(4.1)

A block diagram showing the closed loop control of a single mode is shown in Figure(4.2). A similar control module is implemented for each mode we want to control. [2]

Each Control module consists of 6 components.

1. Demodulator : For converting $\dot{\Phi}_i(t)$ to $|\dot{\Phi}_i(t)|$.

2. Lowpass Filter : For a given natural frequency ω_i, a second order Butterworth filter is used to remove the ripple. The transfer function of the Lowpass filter is given by

$$H_i(s) = \frac{(\omega_i^c)^2}{s^2 + 2\omega_i^c \, s + (\omega_i^c)^2}$$

where ω_i^c , the cutoff frequency, is generally set to one tenth of the value of ω_i .

3. Multiplier : The function of the multiplier is to translate the controller output back to the passband.

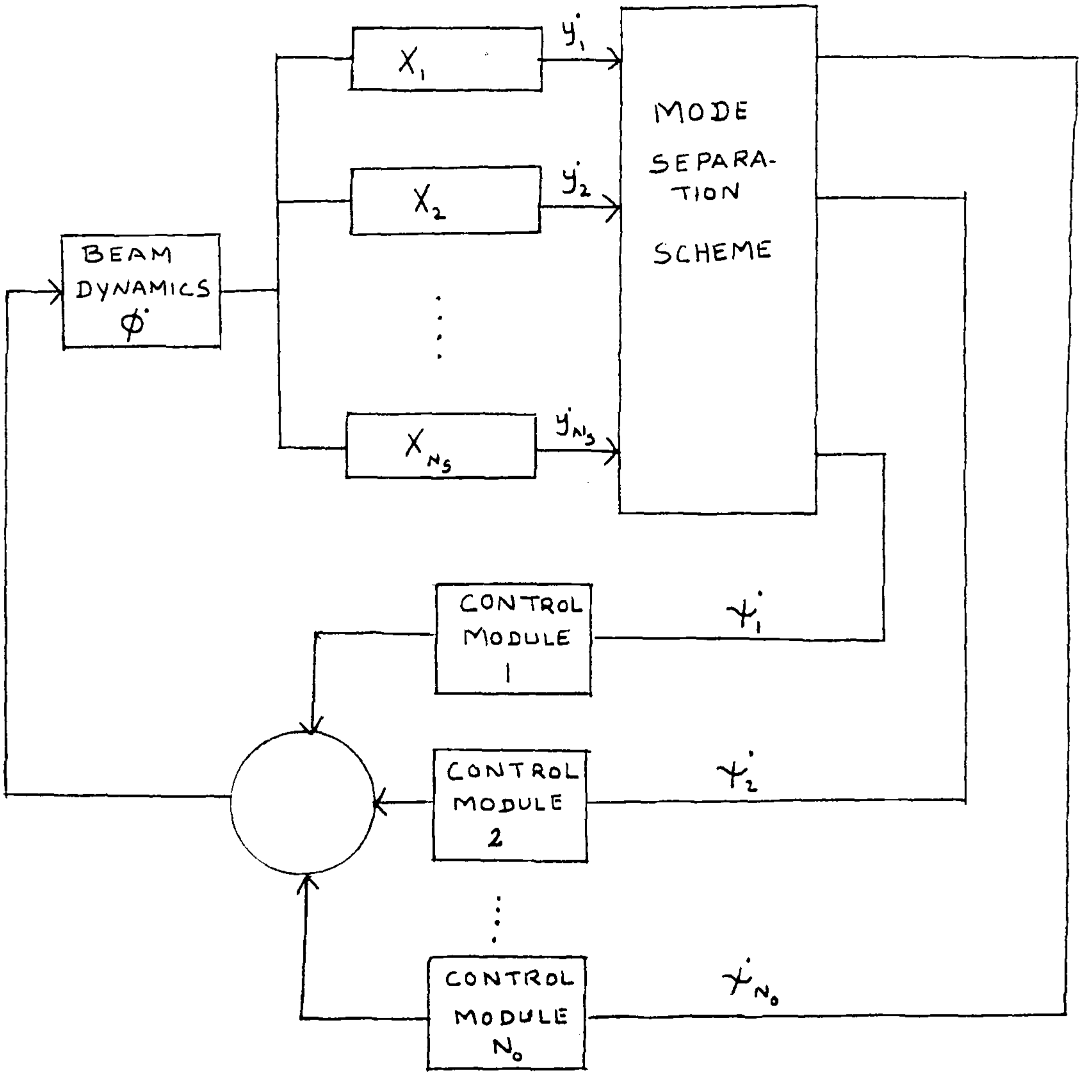

Figure 4.1 Flexible Beam Control System Using Modular Controller [2] and Mode Separation

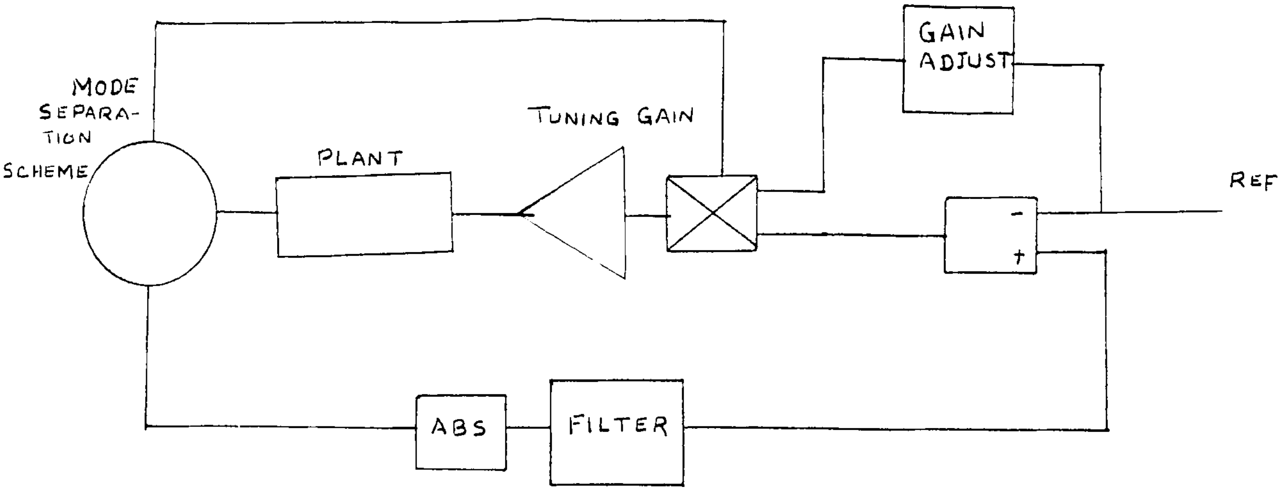

Figure 4.2 Control Module [2]

4. Gain Adjust : Since the module output is multiplicative to the rate signal, loop gain is scaled by the reference signal, $\dot{\Phi}_i^{ref}$ and must be adjusted. This is carried out by inserting the scaling factor

$$\frac{1}{\dot{\Phi}_i^{ref} + 0.01} \tag{4.3}$$

in the loop. The 0.01 factor is included to prevent singularities as $\dot{\Phi}_i^{ref}$ goes to zero.

5. PI Controller : Stabilization and regulation of $\dot{\Phi}_i(t)$ is carried out by a PI controller having the form

$$K_p + \frac{K_I}{s} \tag{4.4}$$

where K_p is the proportional gain and K_I is the integral gain.

6. Tuning gain ε_i : The magnitude of this gain element is determined by on-line tuning to obtain satisfactory transient response.

4.2 Simulation Results

Simulation is carried out for the following cases using Euler's approximation for the mode separation scheme.

4.2.1 Simulation of open loop dynamics

The state space model for the plant containing 4 modes is given by

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -w_1^2\, x_1 \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= -w_2^2\, x_3 \\
\dot{x}_5 &= x_6 \\
\dot{x}_6 &= -w_3^2\, x_5 \\
\dot{x}_7 &= x_8 \\
\dot{x}_8 &= -w_4^2\, x_7 \qquad (4.5)
\end{aligned}$$

The frequency values are given in Table(2.1). Figure(4.3) to Figure(4.6) show the 4 modal velocities. We can see they are not stabilized as there is no control.

4.2.2 Simulation of closed loop dynamics with: $N_m = 4$, $N_o = 4$, $N_s = 6$.

From Table (3.3), we can see that for the given system to get a correct estimation, we need 6 sensors, because at this value the maximum singular value is dropped from .9977 to .0101.

From Figure (3.8) we can see, the estimated modal velocities are the same as the modal velocities. Therefore, the system can be stabilized when the estimated modal velocities are fed back with proper gain. Table (4.1) contains the values of the 4 modes at 6 sensors.

Table 4.1 Mode Shape Values at 6 Sensors Placed at Equidistant on the Beam

x	$X_1(x)$	$X_2(x)$	$X_3(x)$	$X_4(x)$
22.66	-.0614	.1062	-.0864	-.0002
44.33	-.1059	.1058	.1226	.0005
66.5	-.1214	-.0008	-.0872	-.0007
88.6	-.1033	-.1065	.0025	.0019
110.83	-.0563	-.1046	-.0872	-.0007
133	.0072	.0037	.0025	.0019

The state space model of the plant:

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -\omega_1^2\ x_1 + u_1 \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= -(\omega_1^c)^2\ x_3 - 2\ \omega_1^c\ x_4 + (\omega_1^c)^2\ |x_2| \\
e_1 &= x_3 - refer \\
\dot{x}_5 &= e_1 \\
\dot{x}_6 &= x_7 \\
\dot{x}_7 &= -\omega_2^2\ x_6 + u_1 \\
\dot{x}_8 &= x_9 \\
\dot{x}_9 &= -(\omega_2^c)^2\ x_8 - 2\ \omega_2^c\ x_9 + (\omega_2^c)^2\ |x_7| \\
e_2 &= x_8 - refer \\
\dot{x}_{10} &= e_2 \\
\dot{x}_{11} &= x_{12} \\
\dot{x}_{12} &= -\omega_3^2\ x_{11} + u_1 \\
\dot{x}_{13} &= x_{14} \\
\dot{x}_{14} &= -(\omega_3^c)^2\ x_{13} - 2\ \omega_3^c\ x_{14} + (\omega_3^c)^2\ |x_{12}| \\
e_3 &= x_{13} - refer
\end{aligned} \tag{4.6}$$

$$
\begin{aligned}
\dot{x}_{15} &= e_3 \\
\dot{x}_{16} &= x_{17} \\
\dot{x}_{17} &= -\omega_4^2 \; x_{16} + \mu_1 \\
\dot{x}_{18} &= x_{19} \\
\dot{x}_{19} &= -(\omega_4^c)^2 \; x_{18} - 2 \; \omega_4^c \; x_{19} + (\omega_4^c)^2 \; |x_{17}| \\
e_4 &= x_{18} - refer \\
\dot{x}_{20} &= e_4
\end{aligned}
\tag{4.7}
$$

$$
\begin{aligned}
\dot{y}_1 &= -.0614 \; x_2 + .1062 \; x_7 - .1226 \; x_{12} + .1061 \; x_{17} \\
\dot{y}_2 &= -.1059 \; x_2 + .1058 \; x_7 + .0004 \; x_{12} - .1063 \; x_{17} \\
\dot{y}_3 &= -.1214 \; x_2 - .0008 \; x_7 + .1226 \; x_{12} + .0005 \; x_{17} \\
\dot{y}_4 &= -.1033 \; x_2 - .1065 \; x_7 - .0008 \; x_{12} + .1059 \; x_{17} \\
\dot{y}_5 &= -.0563 \; x_2 - .1046 \; x_7 - .1223 \; x_{12} - .1064 \; x \\
\dot{y}_6 &= +.0072 \; x_2 + .0037 \; x_7 + .0025 \; x_{12} + .0019 \; x_{17}
\end{aligned}
$$

$$
\begin{aligned}
cal(x_2) &= 22.17 \; (-.0614 \; \dot{y}_1 - .1059 \; \dot{y}_2 - .1214 \; \dot{y}_3 \\
&\quad -.1033 \; \dot{y}_4 - .0563 \; \dot{y}_5 + .0072 \; \dot{y}_6) \\
\dot{\Psi}_1 &= cal(x_2) + 5.18 \; 133 \; .0072 \; \dot{y}_6 \\
cal(x_7) &= 22.17 \; (.1062 \; \dot{y}_1 + .1058 \; \dot{y}_2 - .0008 \; \dot{y}_3 \\
&\quad -.1065 \; \dot{y}_4 - .1046 \; \dot{y}_5 + .0037 \; \dot{y}_6) \\
\dot{\Psi}_2 &= cal(x_7) + 5.18 \; 133 \; .0037 \; \dot{y}_6 \\
cal(x_{12}) &= 22.17 \; (-.1226 \; \dot{y}_1 + .0004 \; \dot{y}_2 + .1226 \; \dot{y}_3 \\
&\quad -.0008 \; \dot{y}_4 - .1223 \; \dot{y}_5 + .0025 \; \dot{y}_6) \\
\dot{\Psi}_3 &= cal(x_{12}) + 5.18 \; 133 \; .0025 \; \dot{y}_6 \\
cal(x_{17}) &= 22.17 \; (.1061 \; \dot{y}_1 - .1063 \; \dot{y}_2 + .0005 \; \dot{y}_3
\end{aligned}
$$

$$
\begin{aligned}
& +.1059\ \dot{y}_4 - .1064\ \dot{y}_5 + .0019\ \dot{y}_6) \\
\dot{\Psi}_4 &= cal(x_{17}) + 5.18\ 133\ .0019\ y_6 \\
u_1 &= refscale\ (k_1\ \dot{\Psi}_1\ e_1 + k_2\ \dot{\Psi}_2\ e_2 \\
& +k_3\ \dot{\Psi}_3\ e_3 + k_4\ \dot{\Psi}_4\ e_4)
\end{aligned}
$$

The states x_1 thru x_5 represent the first mode and the its lowpass filter. similarly x_6 thru x_{10} represent the second mode and its lowpass filter. and x_{11} thru x_{15} represent the third mode and the filter. and x_{16} thru x_{20} represent the fourth mode and its filter.

The variables $\dot{y}_1$ thru $\dot{y}_6$ represent the output from the sensors. u_1 is the input formed from the estimated the estimated mode velocities. $\dot{\Psi}_1$ thru $\dot{\Psi}_3$ represent the estimated mode velocities. k_1 thru k_4 terms are controller gains for modes $i = 1, 2 \cdots N_o$.

From Figures (4.7) thru (4.14), we can see that all the modes are stabilized. The gains used for this run together with the gain adjust are

1. $k_1 = -50.$

2. $k_2 = -40.$

3. $k_3 = -30.$

4. $k_4 = -20.$

4.2.3 Simulation of closed loop dynamics with: $N_m = 4$, $N_o = 2$, $N_s = 4$

State space model is the same as (4.17), except the difference in the output values, and the control. The mode shape values at 4 sensors are given in Table (3.1). The output of the sensors, and the control term can be calculated as:

$$\dot{y}_1 = -.0867\ x_2 + .1225\ x_7 - .0864\ x_{12} - .0002\ x_{14}$$

$$
\begin{aligned}
\dot{y}_2 &= -.1214\ x_2 - .0008\ x_7 + .1226\ x_{12} + .0005\ x_{14} \\
\dot{y}_3 &= -.0828\ x_2 - .1221\ x_7 - .0872\ x_{12} - .0007\ x_{14} \\
\dot{y}_4 &= .0072\ x_2 + .0037\ x_7 + .0025\ x_{12} + .0019\ x_{14} \\
cal(x_2) &= 33.25\ (-.0867\ \dot{y}_1 - .1214\ \dot{y}_2 - .0828\ y_3 +\ 0072\ \dot{y}_4) \\
\dot{\Psi}_1 &= cal(x_2) + 5.18\ 133\ .0072\ \dot{y}_4 \\
cal(x_7) &= 33.25\ (.1225\ \dot{y}_1 - .0008\ \dot{y}_2 - .1221\ \dot{y}_3 + .0037\ \dot{y}_4) \\
\dot{\Psi}_2 &= cal(x_7) + 5.18\ 133\ .0037\ \dot{y}_4
\end{aligned}
$$

$$u_1 = refscale\ (\dot{\Psi}_1\ k_1\ e_1 + \dot{\Psi}_2\ k_2\ e_2)$$

From Table(3.2), we can see that for the given system to get a correct estimation, we need 4 sensors, because at this value the maximum singuler value is dropped from .998 to .0184.

From Figure(4.15) thru (4.22), we can see the first two modes and their estmations are good and therefore we are able to control those two modes.

The gains used for this run together with the gain adjust are

1. $k_1 = -90.$

2. $k_2 = -50.$

4.2.4 Simulation of closed loop dynamics with: $N_m = 4$, $N_o = 2$, $N_s = 3$

The plant state space model is the same as (3.7). The output from the sensors and the input can be calculated as:

$$
\begin{aligned}
\dot{y}_1 &= -.1059\ x_2 + .1058\ x_7 + .0004\ x_{12} - .1063\ x_{14} \\
\dot{y}_2 &= -.1033\ x_2 - .1065\ x_7 - .0008\ x_{12} + .1059\ x_{14} \\
\dot{y}_3 &= .0072\ x_2 + .0037\ x_7 + .0025\ x_{12} + .0019\ x_{14} \\
cal(x_2) &= 33.25\ (-.1059\ \dot{y}_1 - .1033\ \dot{y}_2 + .0072\ \dot{y}_3)
\end{aligned}
$$

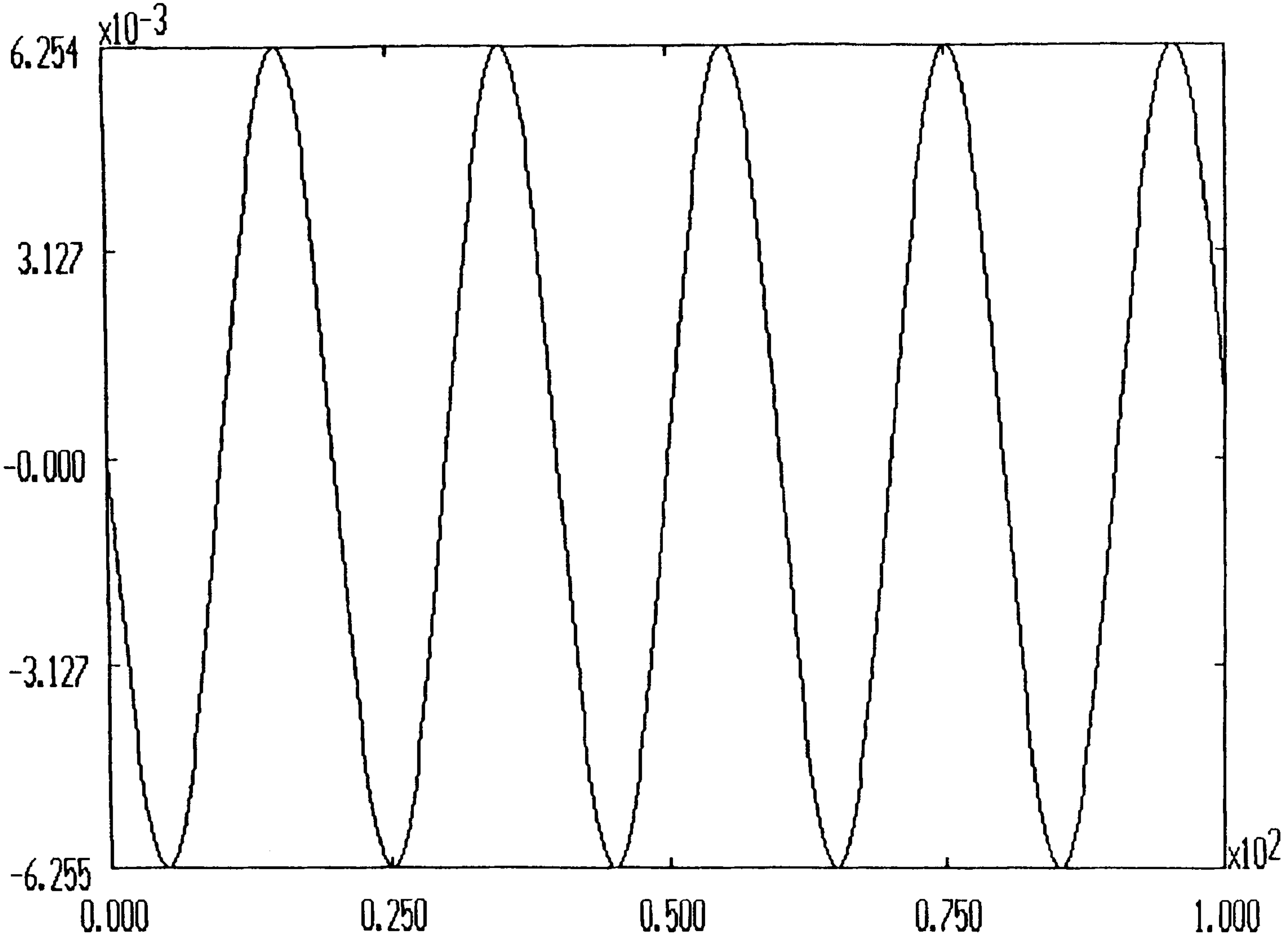

Figure 4.3 Velocity of the First Mode (Open Loop)

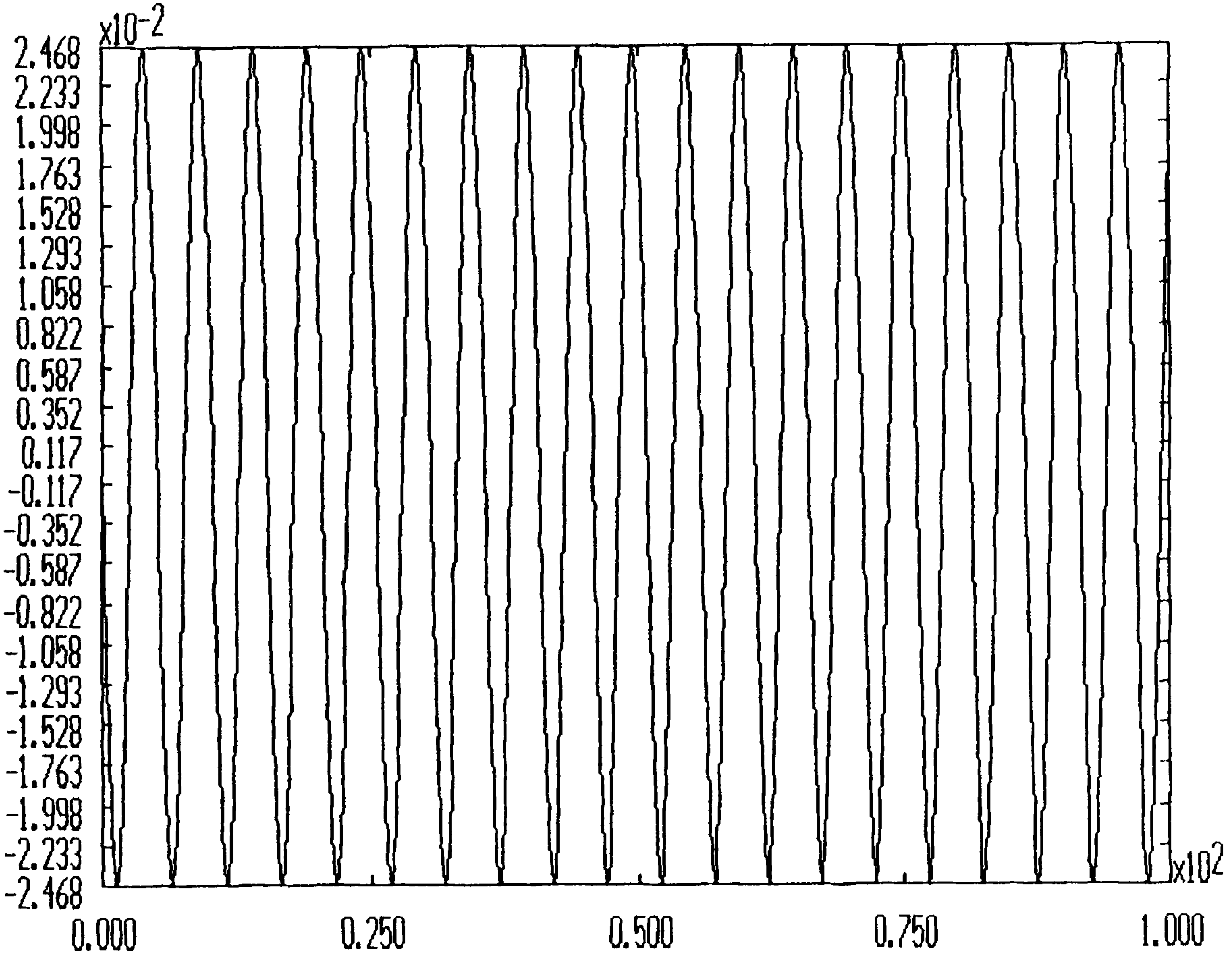

Figure 4.4 Velocity of the Second Mode (Open Loop)

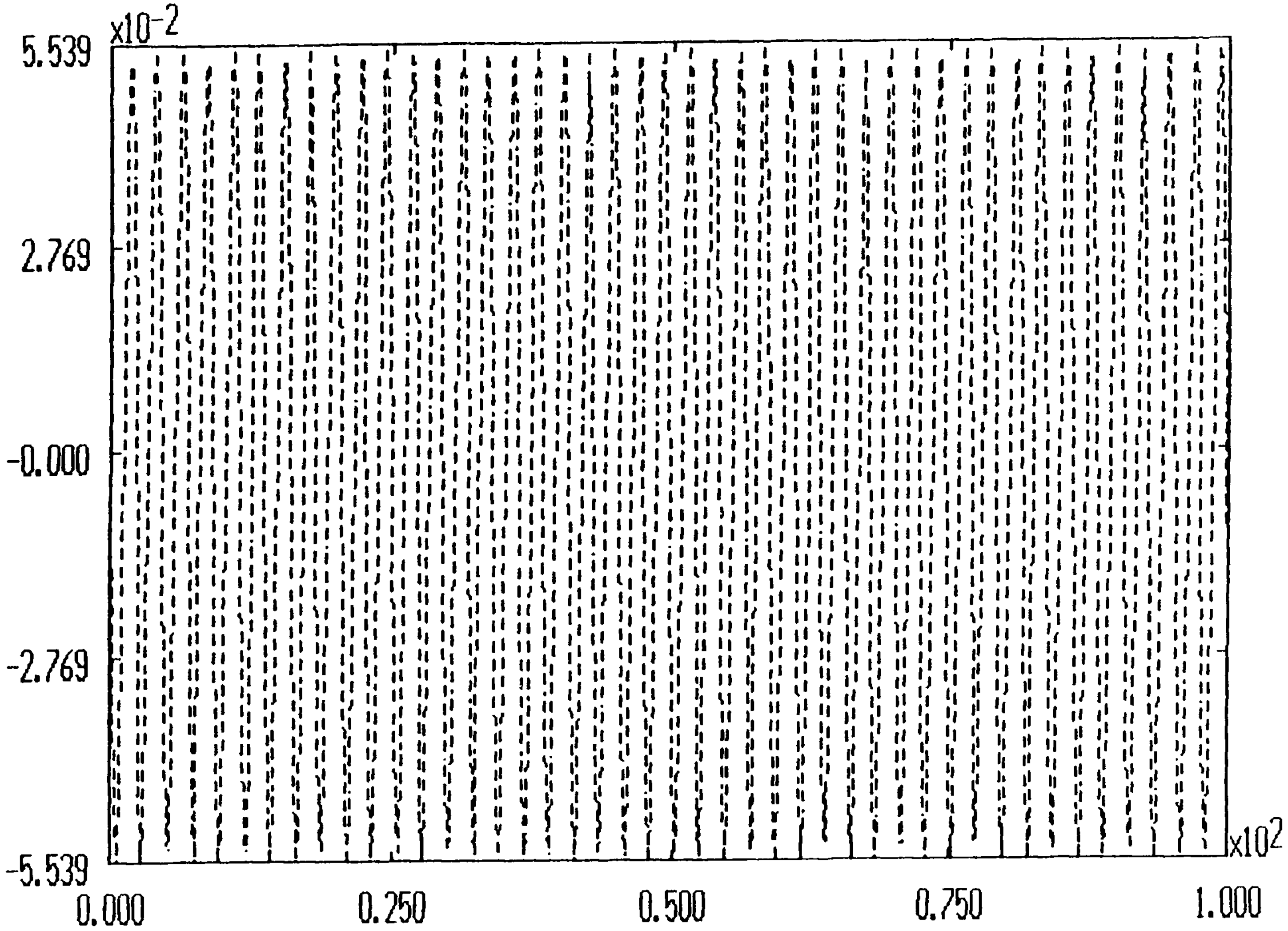

Figure 4.5 Velocity of the Third Mode (Open Loop)

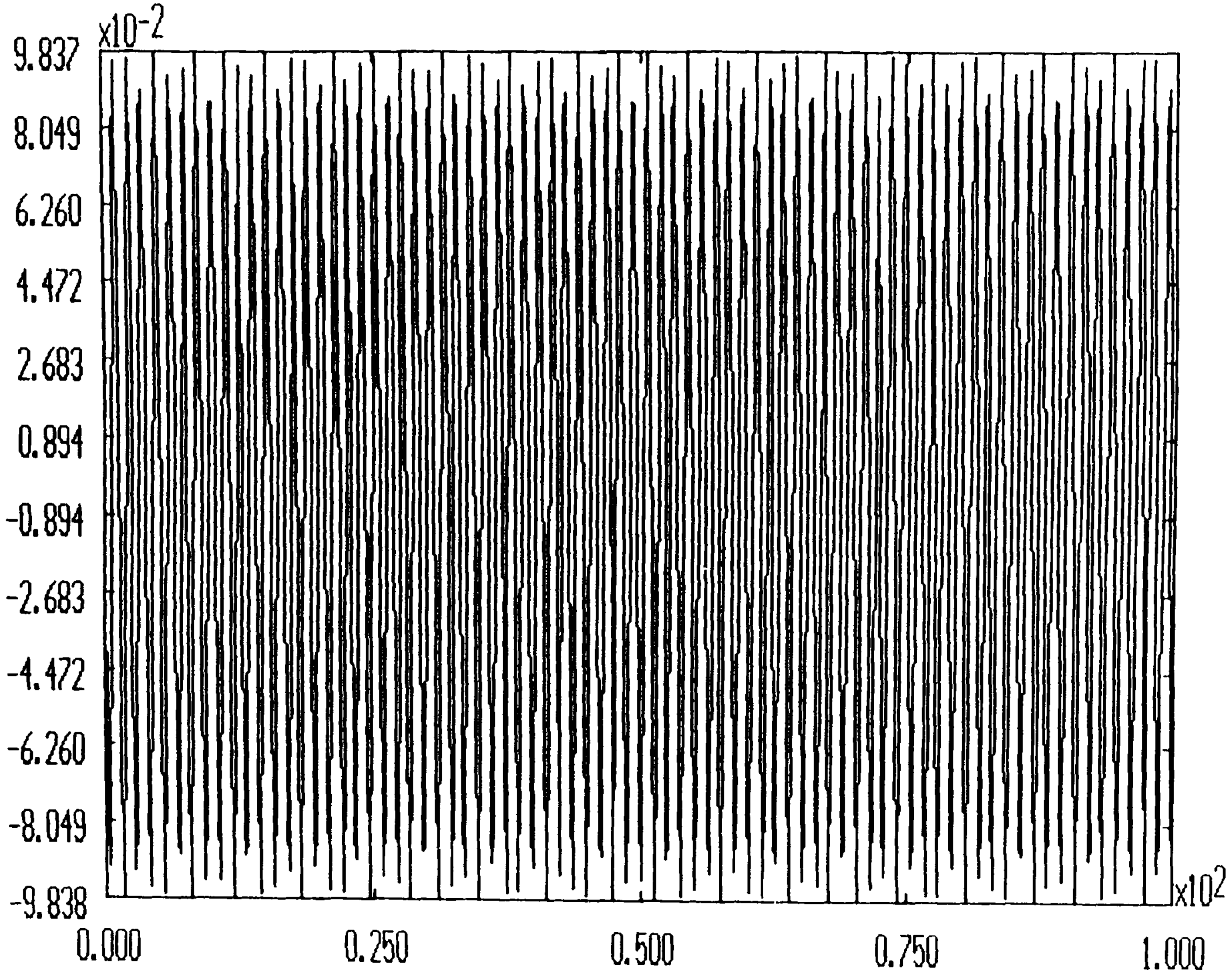

Figure 4.6 Velocity of the Fourth Mode (Open Loop)

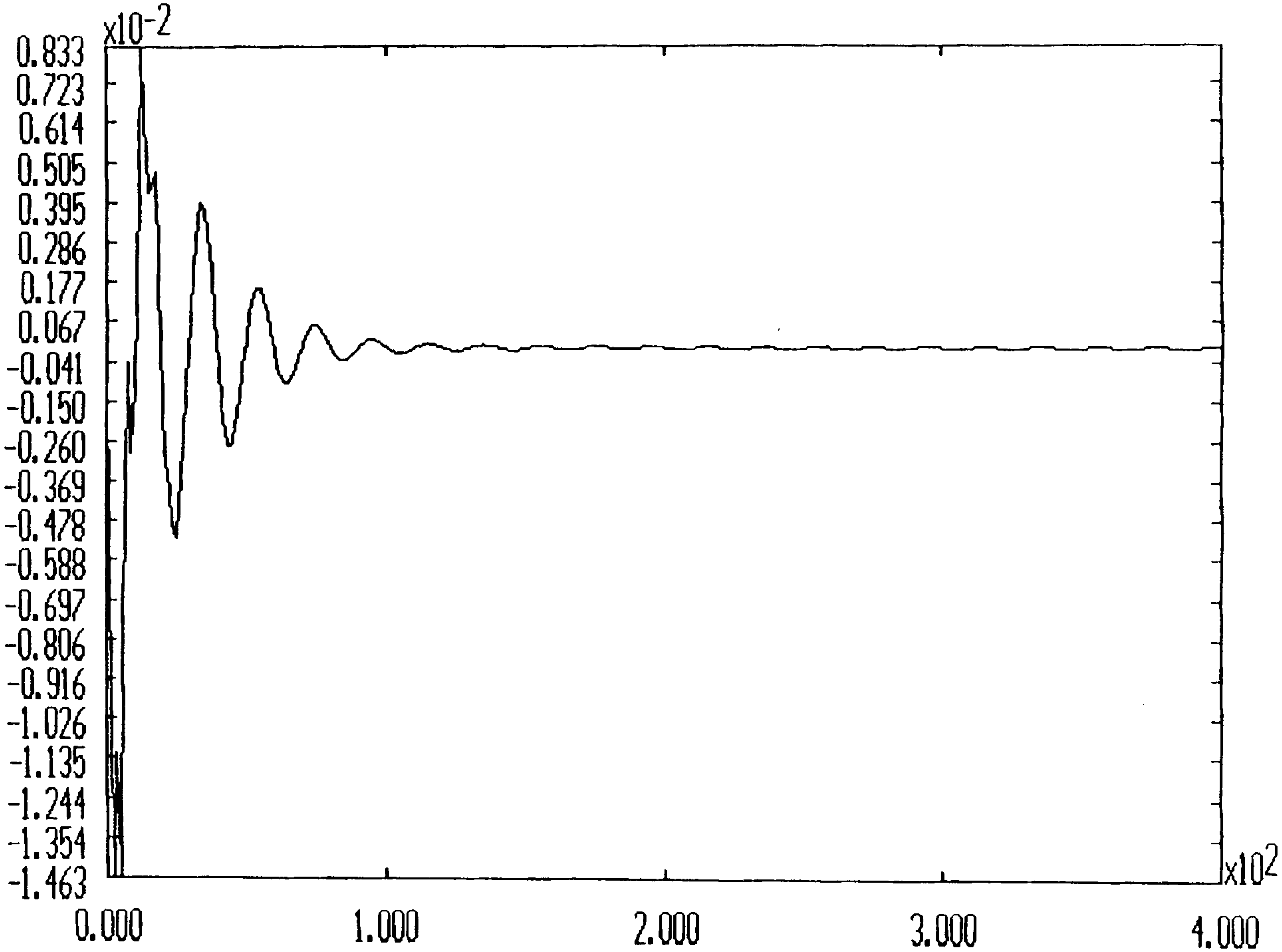

Figure 4.7 Velocity of the First Mode for $N_m = 4$, $N_s = 6$, $N_o = 4$

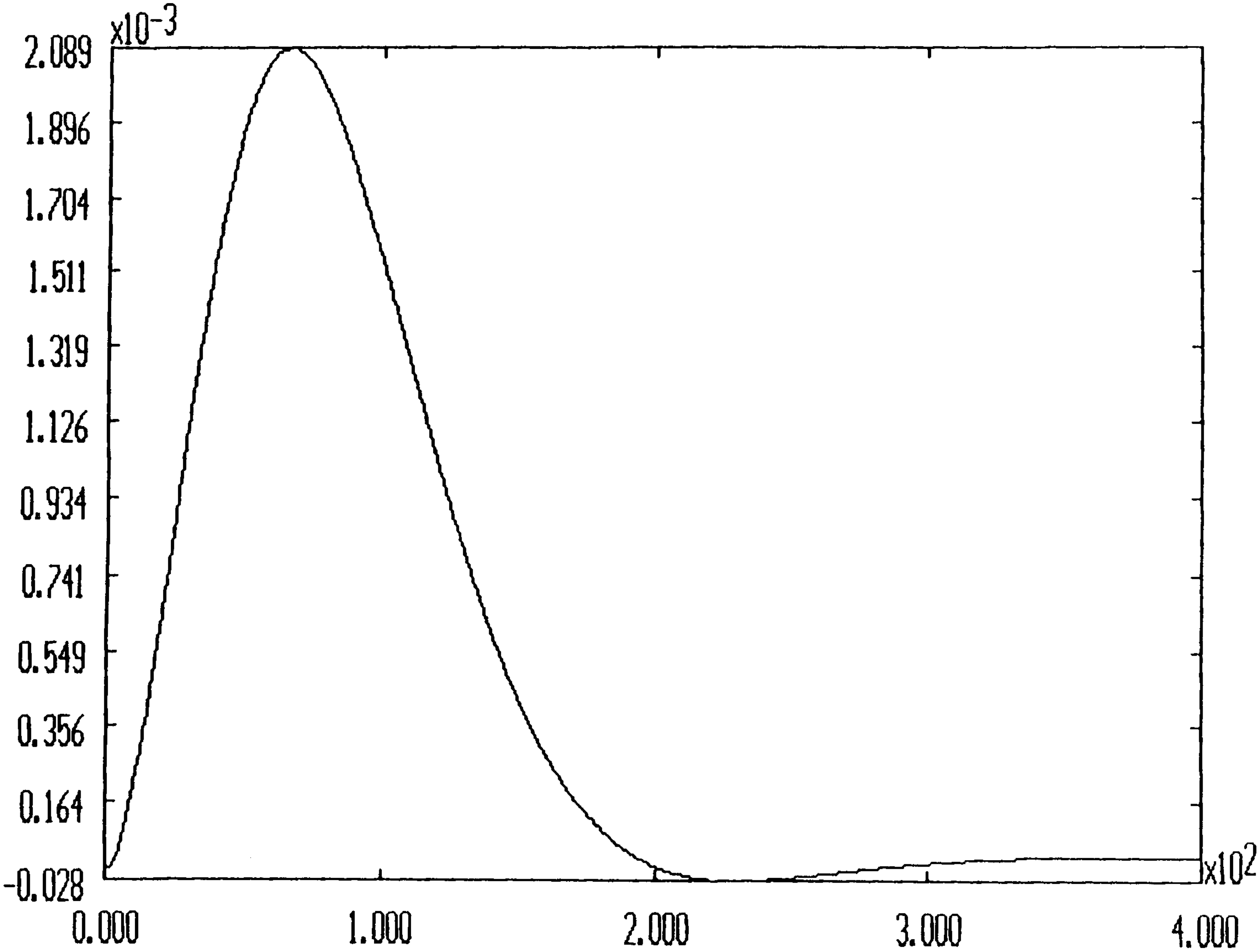

Figure 4.8 Amplitude of Velocity of the First Mode for $N_m = 4$, $N_s = 6$, $N_o = 4$

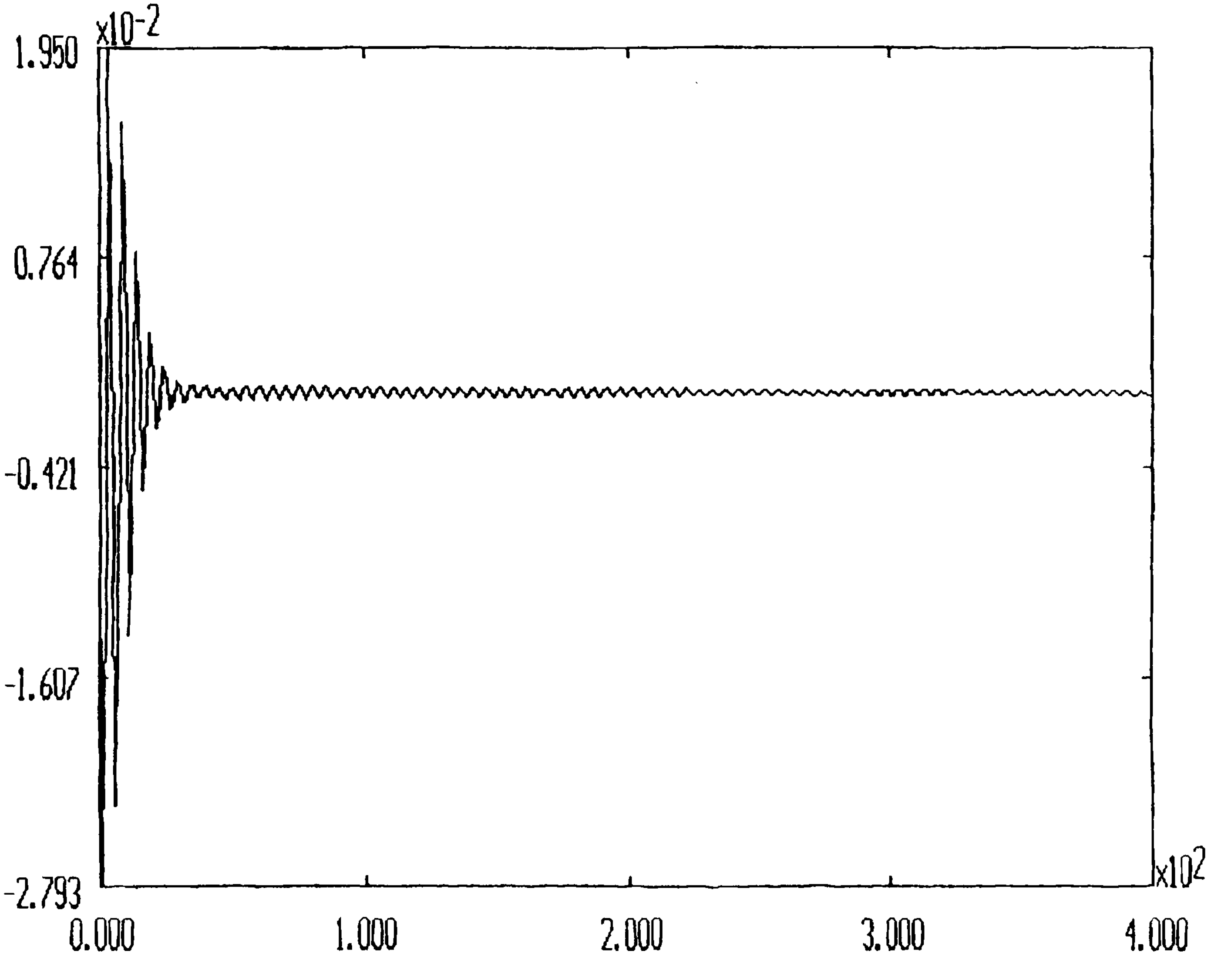

Figure 4.9 Velocity of the Second Mode for $N_m = 4$, $N_s = 6$, $N_v = 4$

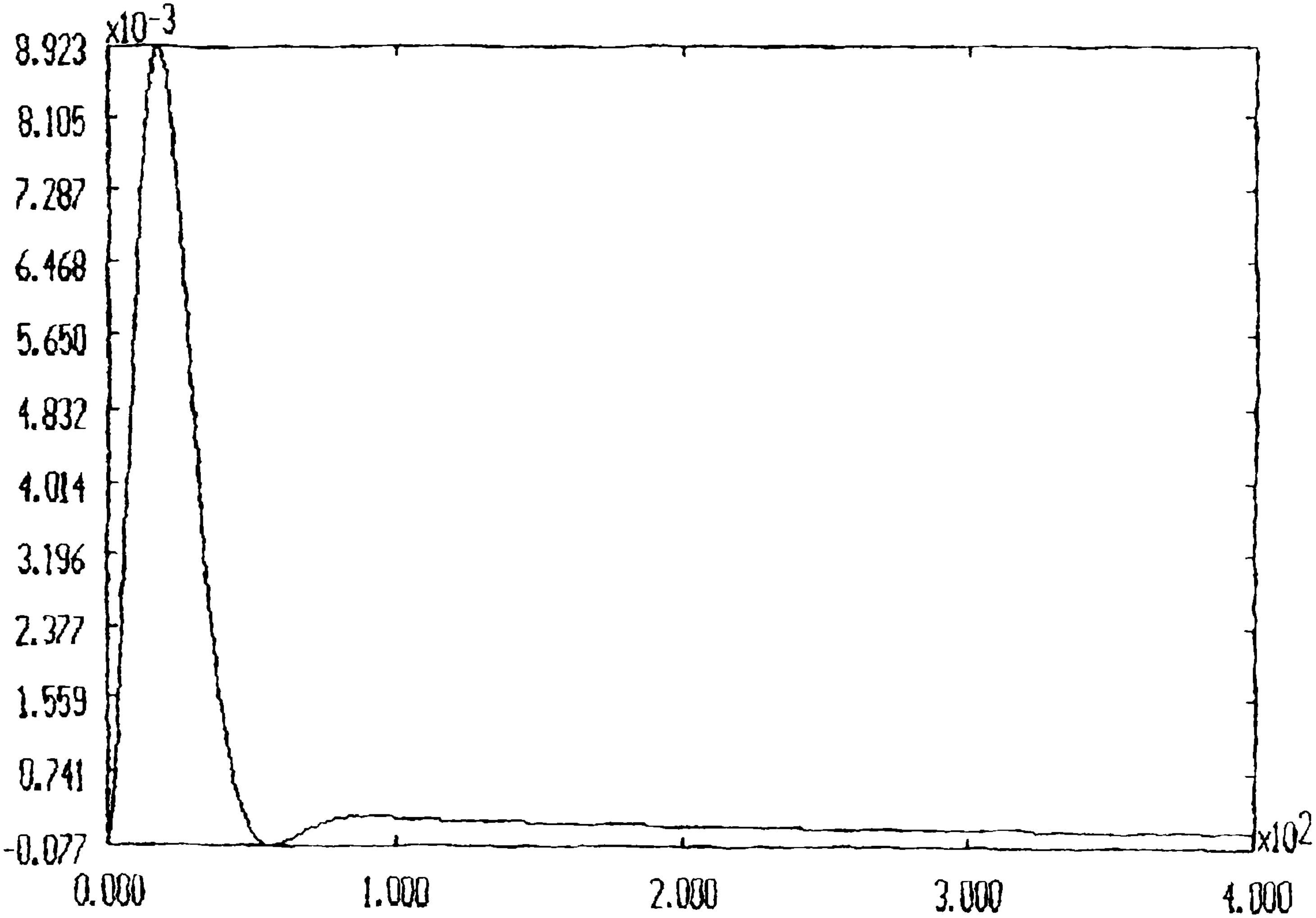

Figure 4.10 Amplitude of Velocity of the Second Mode for $N_m = 4$, $N_s = 6$, $N_a = 4$

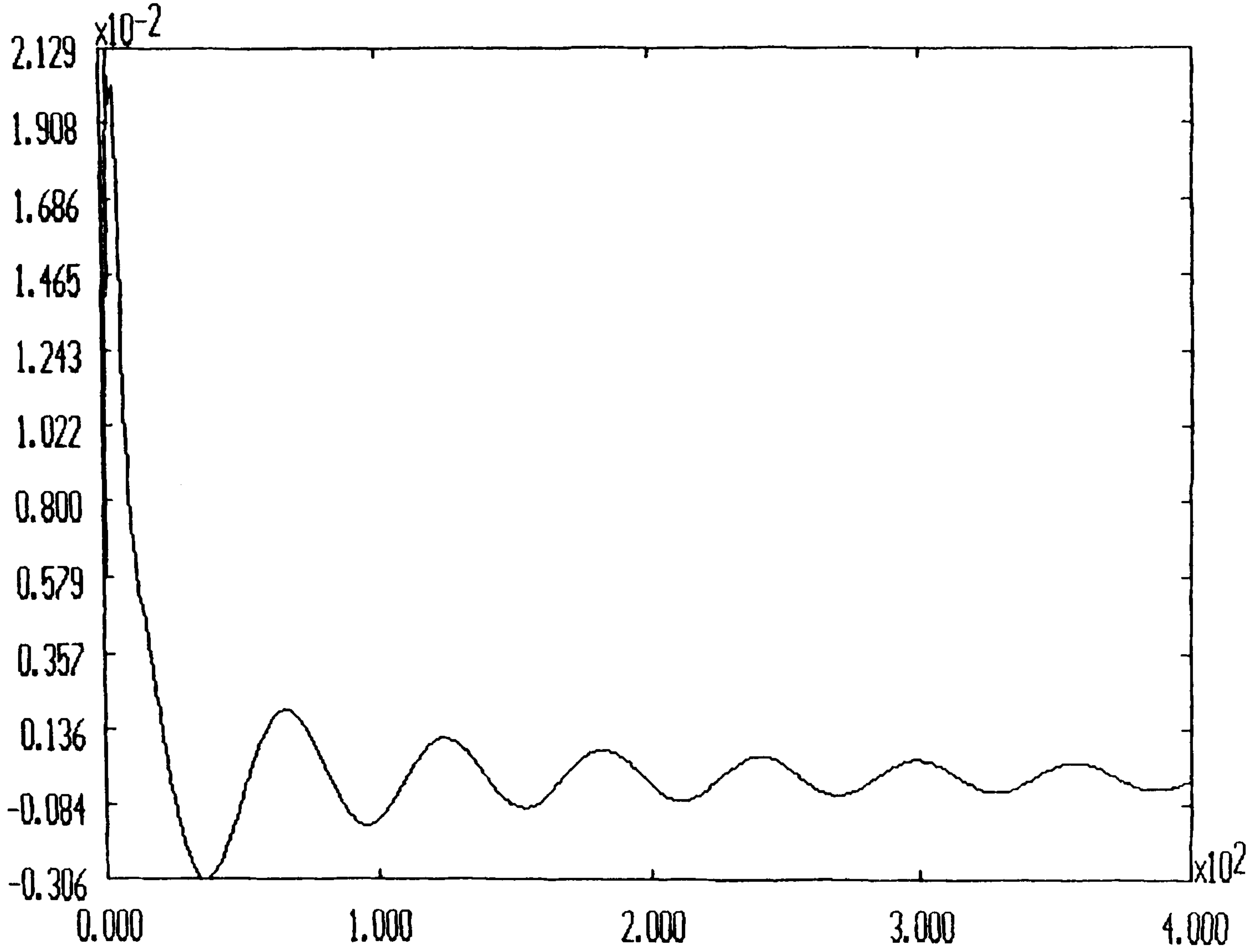

Figure 4.11 Velocity of the Third Mode for $N_m = 4$, $N_s = 6$, $N_o = 4$

$$
\begin{aligned}
\dot{\Psi}_1 &= cal(x_2) + 5.18\ 133\ .0072\ \dot{y}_3 \\
cal(x_7) &= 33.25\ (.1058\ \dot{y}_1 - .1065\ \dot{y}_2 + .0037\ \dot{y}_3) \\
\dot{\Psi}_2 &= cal(x_7) + 5.18\ 133\ .0037\ \dot{y}_3
\end{aligned}
$$

$$u_1 = refscale\ (\dot{\Psi}_1\ k_1\ e_1 + \dot{\Psi}_2\ k_2\ \epsilon_2)$$

The maximum singular for this system is calculated as .9937. From chapter 2, we know that the for good estimation we need, at least 4 sensors. otherwise estimated velocities do not agree with the mode velocities. From Figure(4.23) thru (4.26), we can see that the estimation is wrong, and the system can not be stabilized.

The gains used for this run together with the gain adjust are

1. $k_1 = -4.7$.

2. $k_2 = -1$.

4.3 Discussions

From simulation results, we can see, when the maximum singular value is less, we will get correct estimation of mode velocities, and by feeding back the correct estimated value, we can stabilize the system from sections (4.2.2. and 3.2.3).

From section 3.2.4, we can see when the maximum singular value is around 1, we will not correct estimation of mode velocities, thereby unable to stabilize the system.

The theoretical results obtained in chapter 2 are verified through simulation.

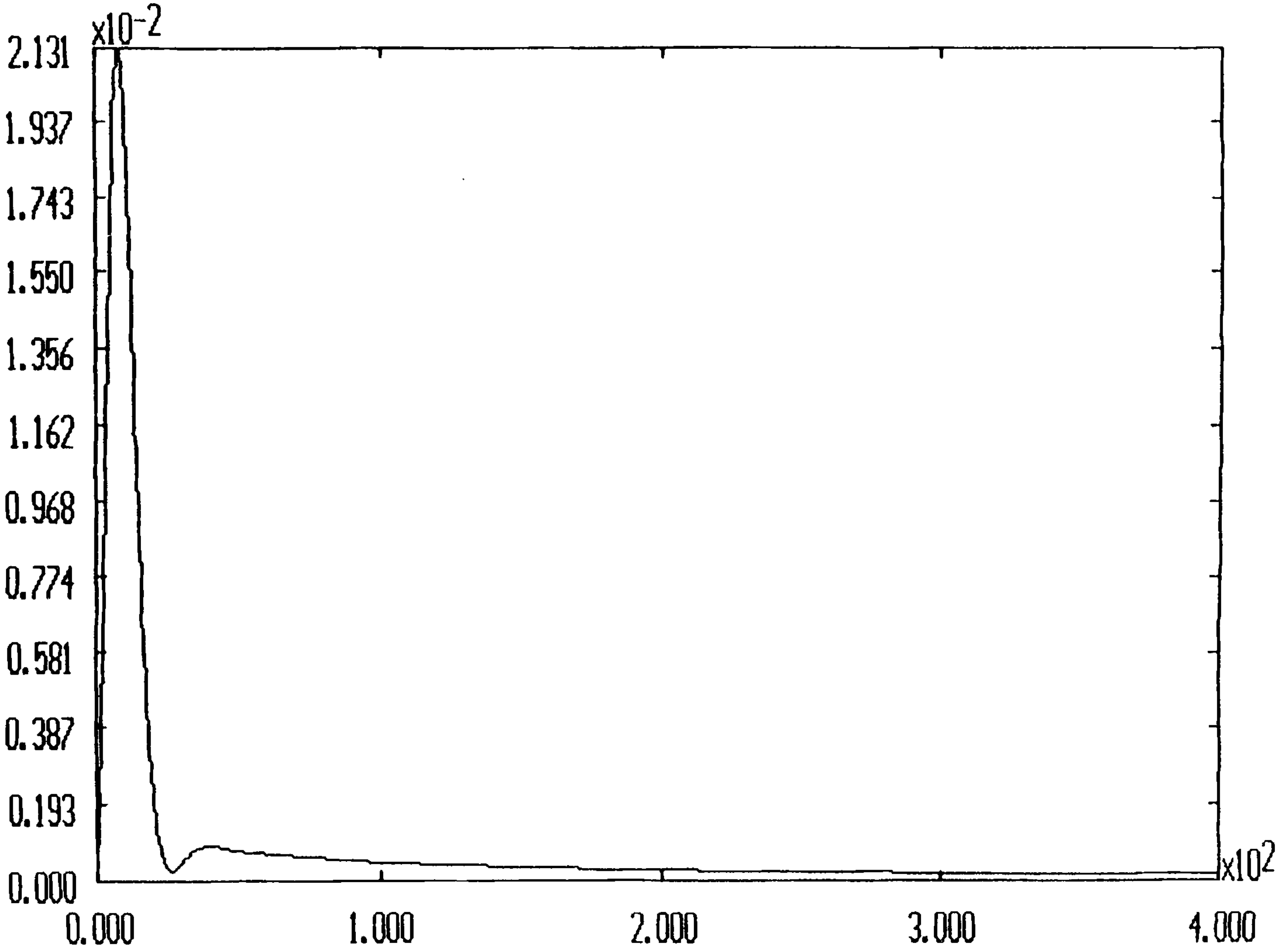

Figure 4.12 Amplitude of Velocity of the Third Mode for $N_m = 4$, $N_s = 6$, $N_\omega = 4$

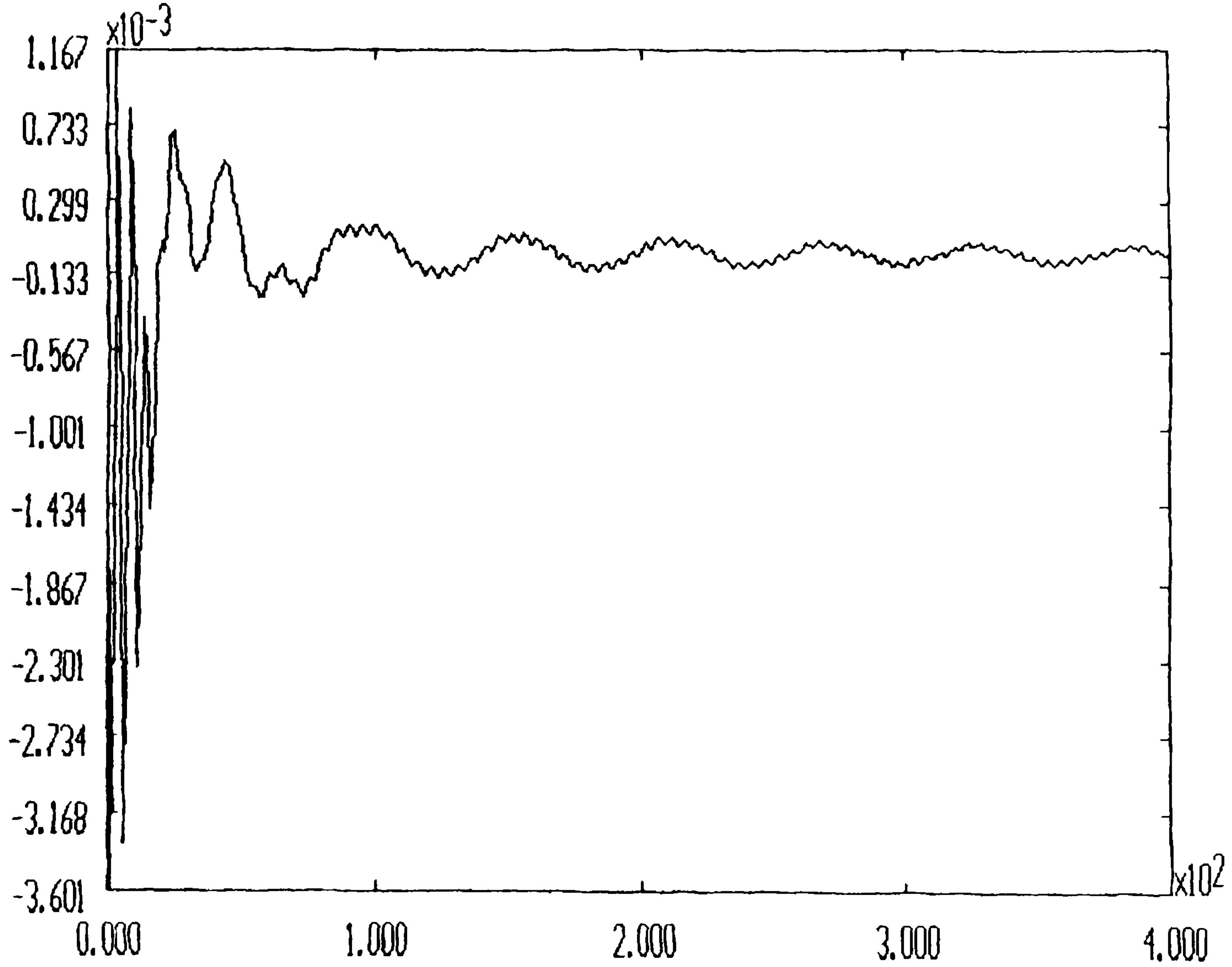

Figure 4.13 Velocity of the Fourth Mode for $N_m = 4$, $N_s = 6$, $N_o = 4$

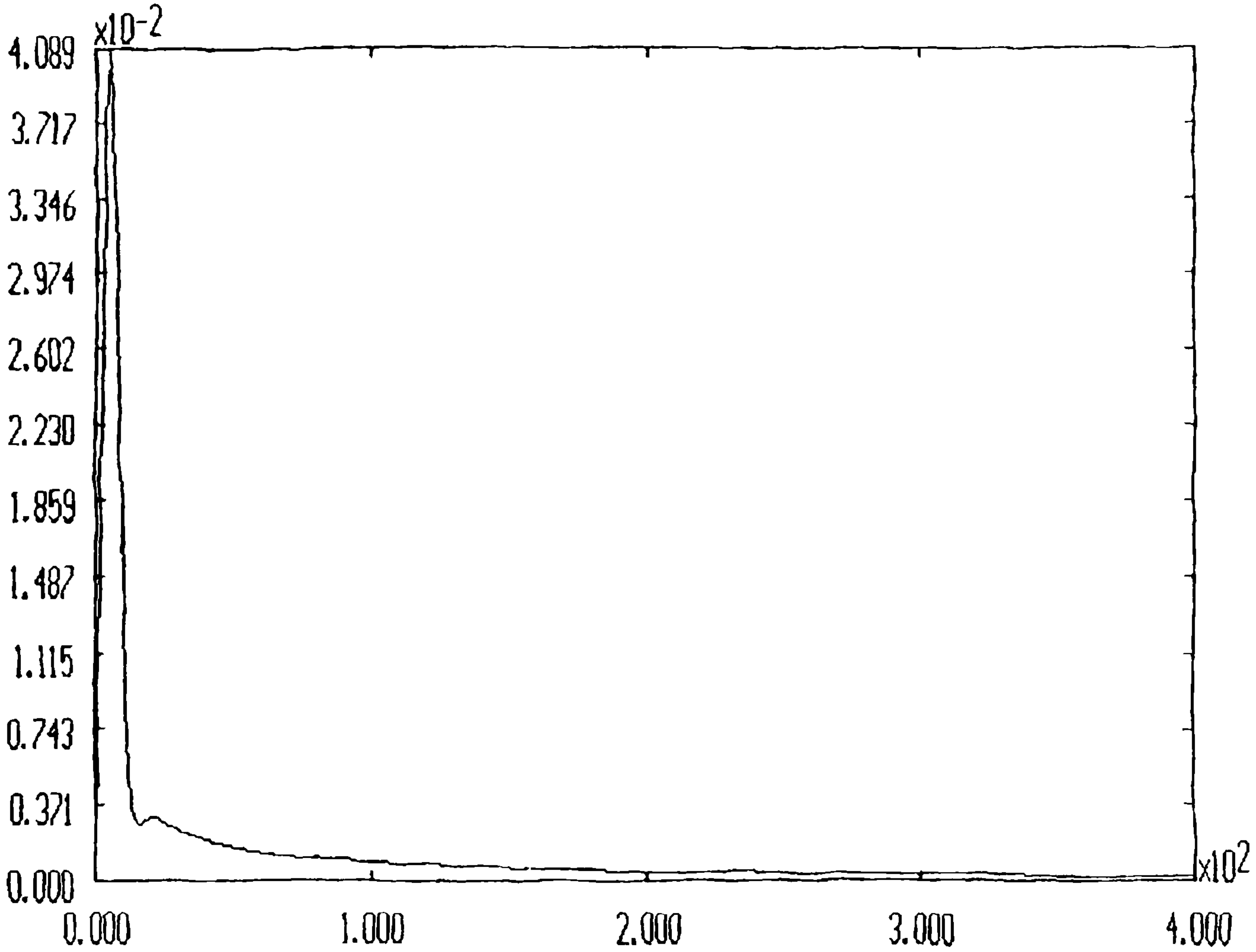

Figure 4.14 Amplitude of Velocity of the Fourth Mode for $N_m = 4$, $N_s = 6$, $N_a = 4$

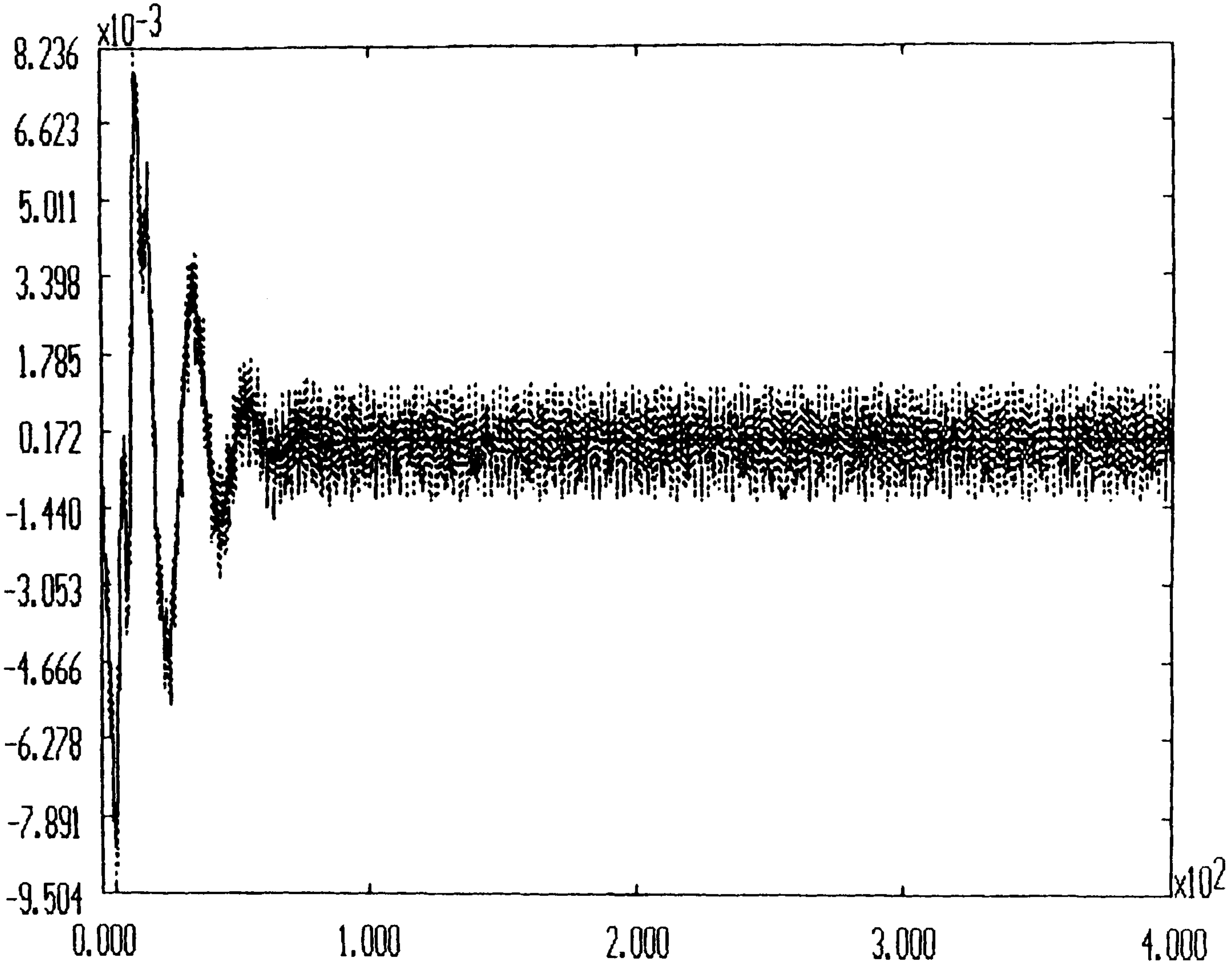

Figure 4.15 Velocity of the First Mode and It's Estimation for $N_m = 4$, $N_s = 4$, $N_o = 2$

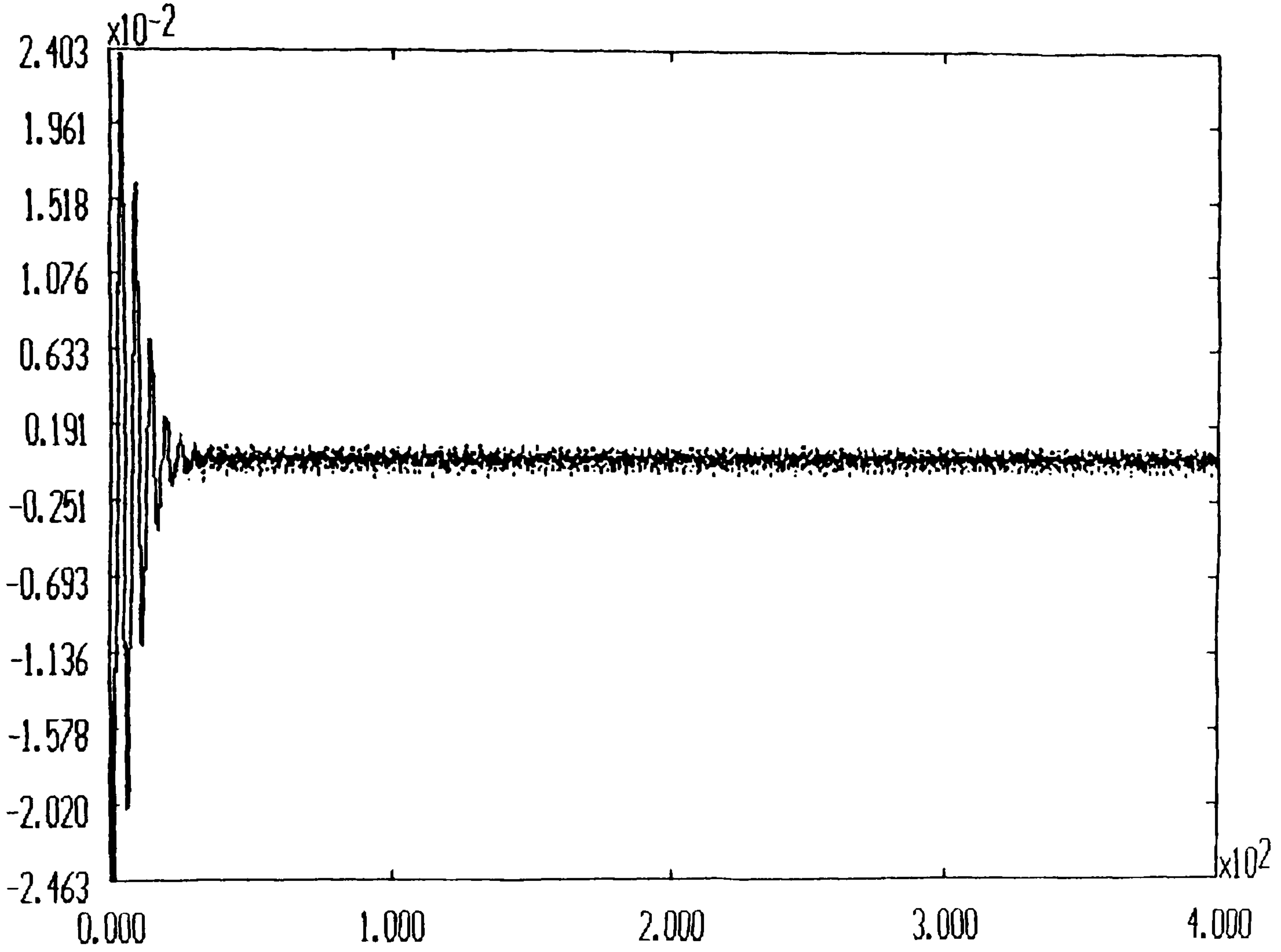

Figure 4.16 Velocity of the Second Mode and It's Estimation for $N_m = 4$, $N_s = 4$, $N_o = 2$

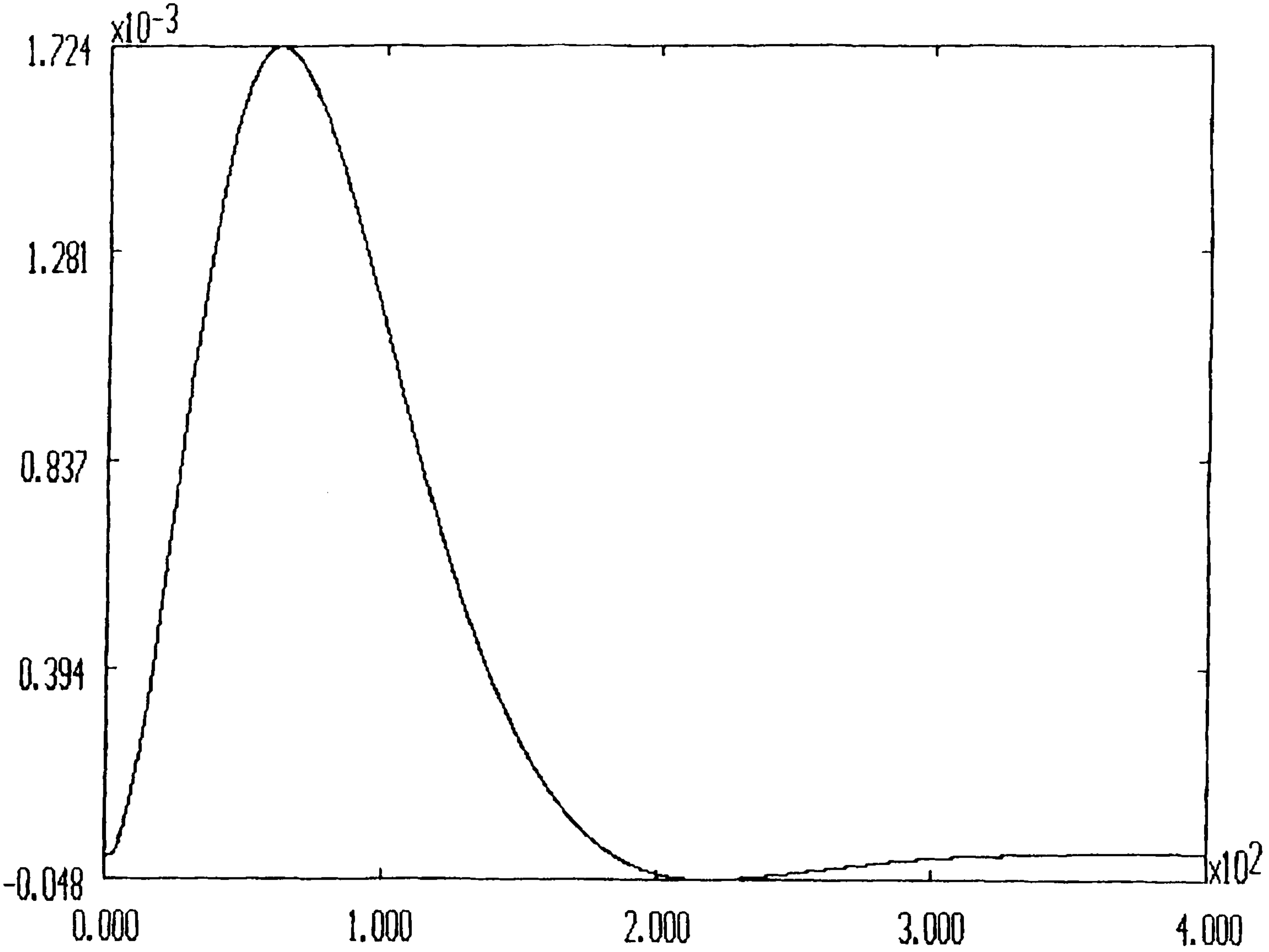

Figure 4.17 Amplitude of Velocity of the First Mode for $N_m = 4$, $N_s = 4$, $N_o = 2$

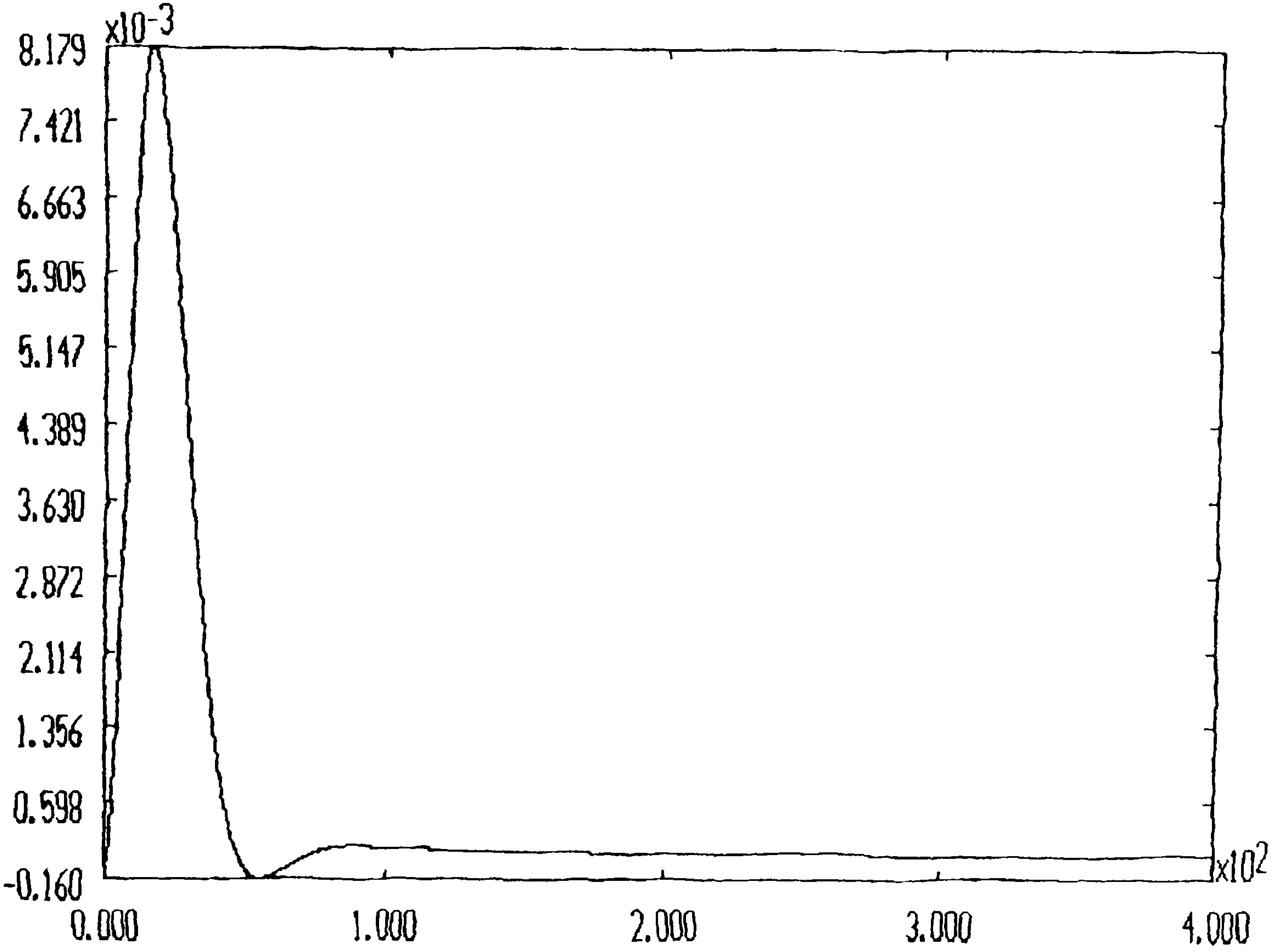

Figure 4.18 Amplitude of Velocity of the Second Mode for $N_m = 4$, $N_s = 4$, $N_o = 2$

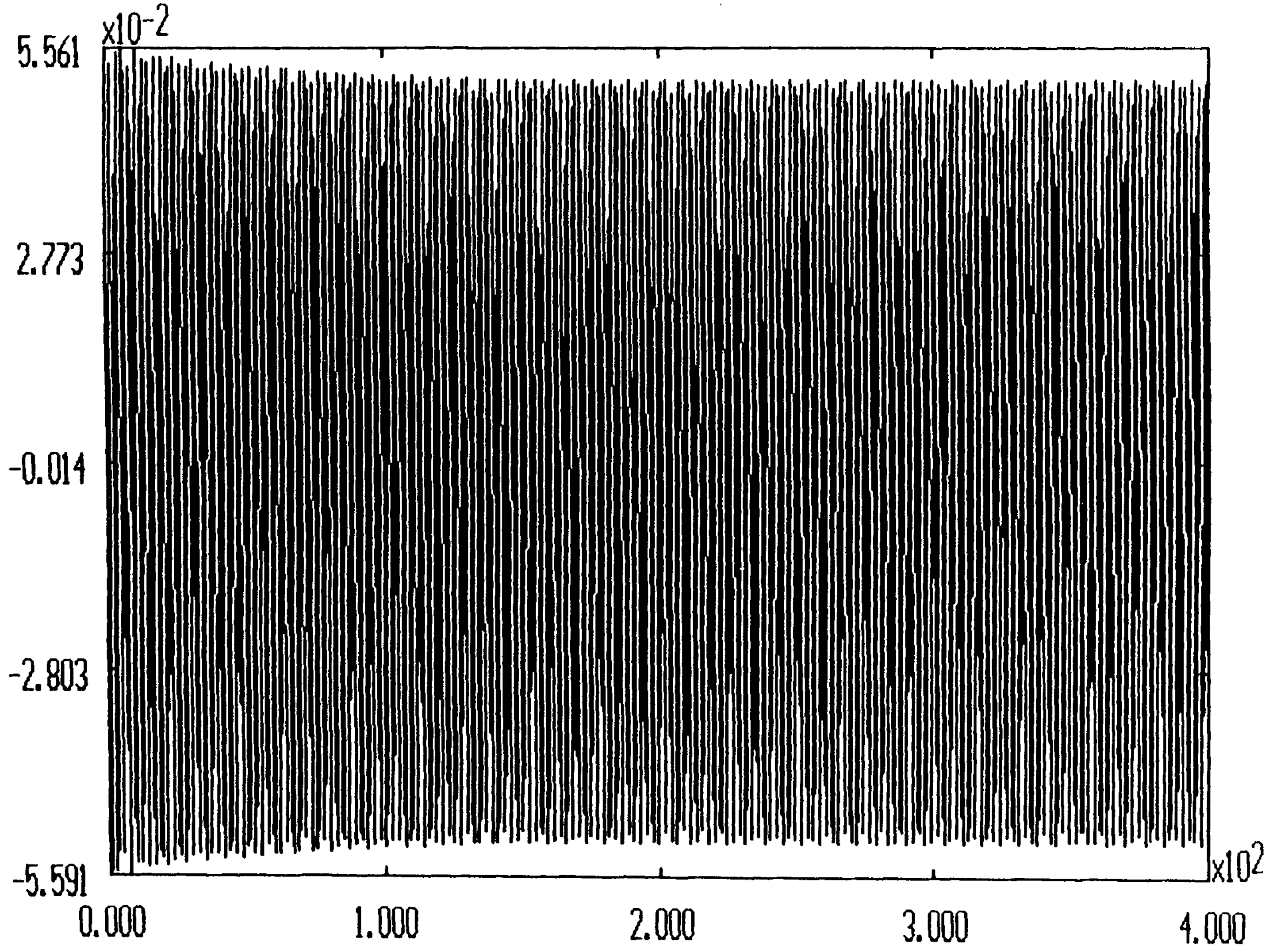

Figure 4.19 Velocity of the Third Mode $N_m = 4$, $N_s = 4$, $N_o = 2$

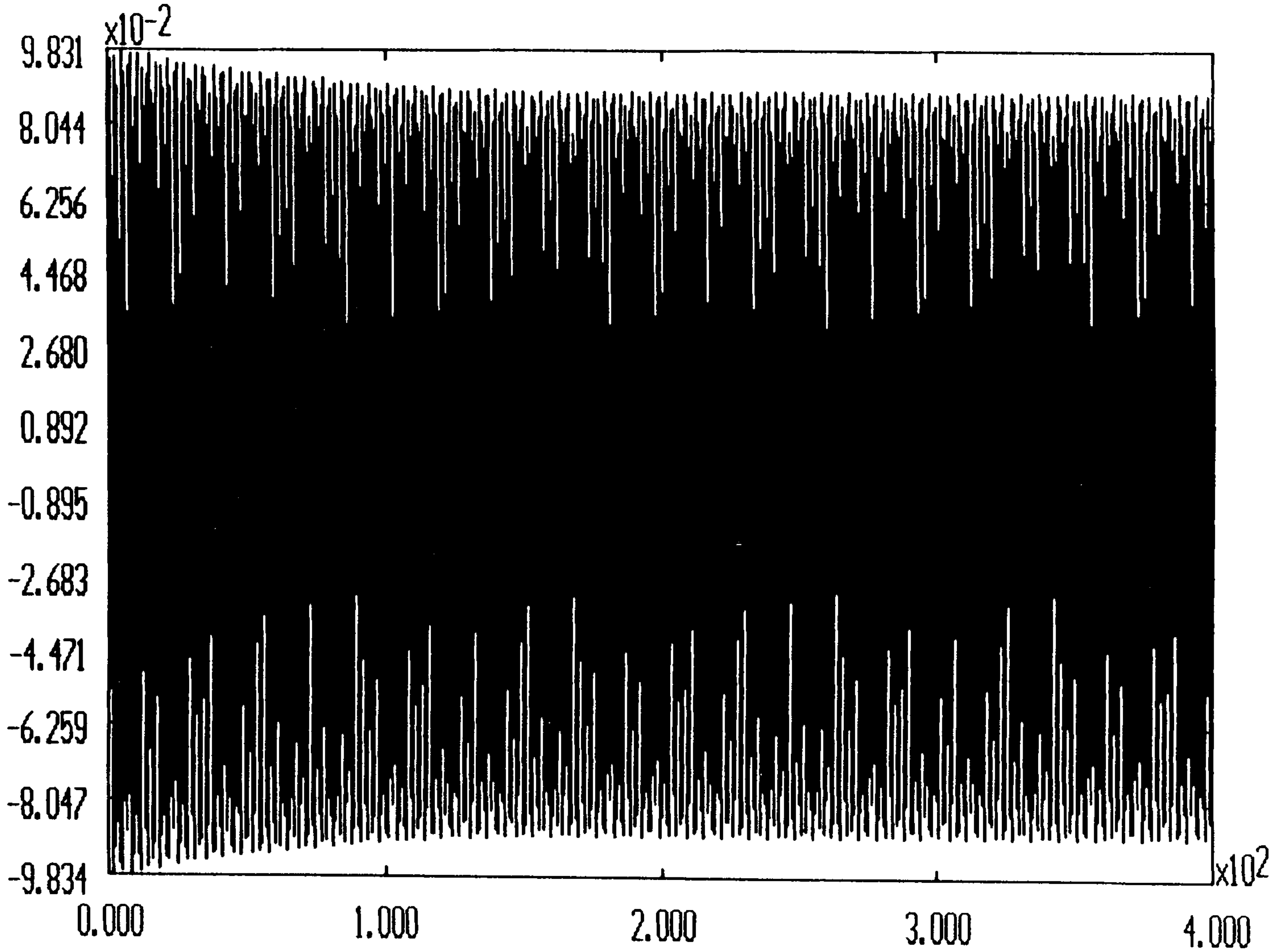

Figure 4.20 Velocity of the Fourth Mode $N_m = 4$, $N_s = 4$, $N_o = 2$

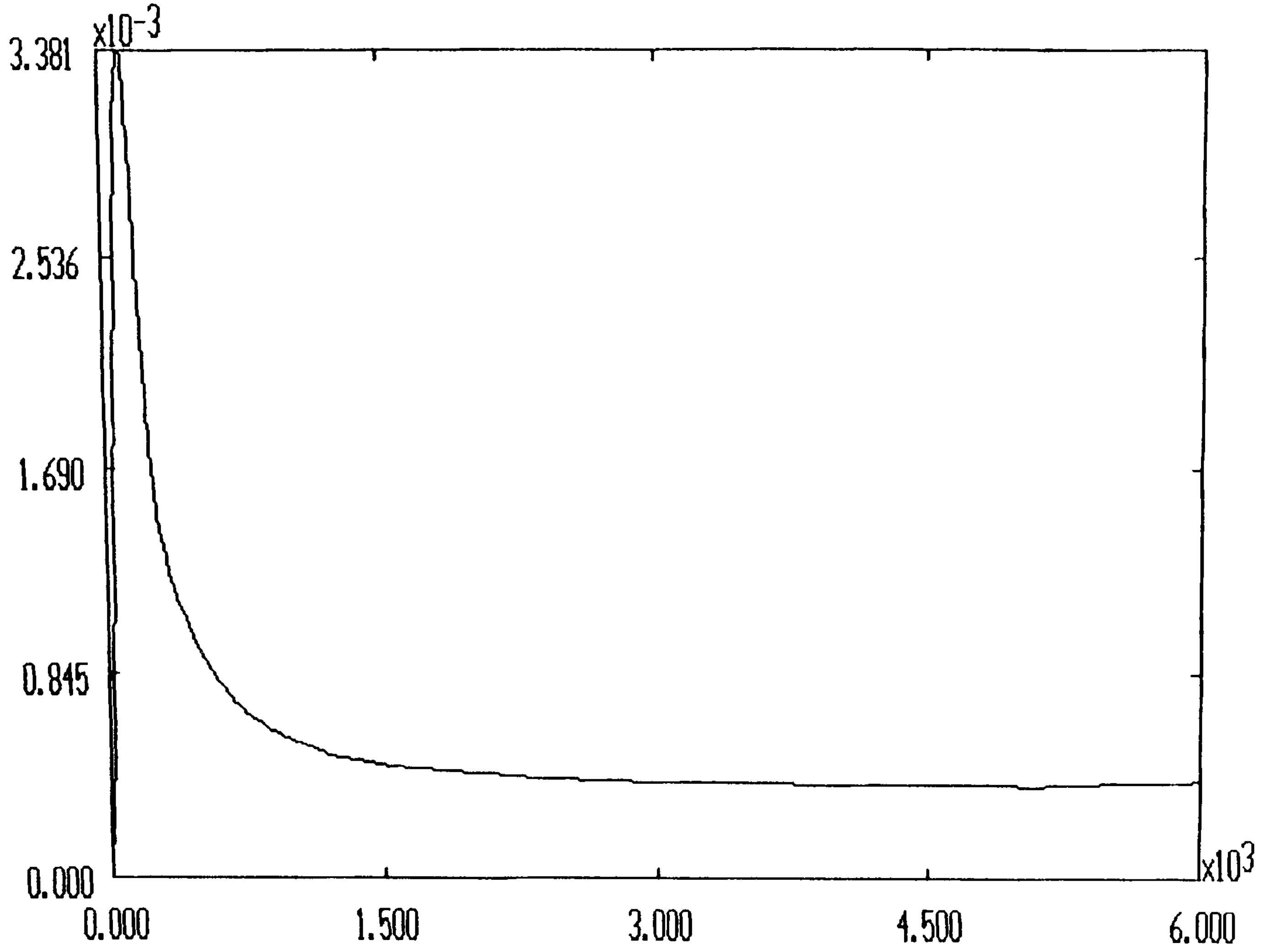

Figure 4.21 Amplitude of Velocity of the First Mode for $N_m = 4$, $N_s = 3$, $N_o = 2$

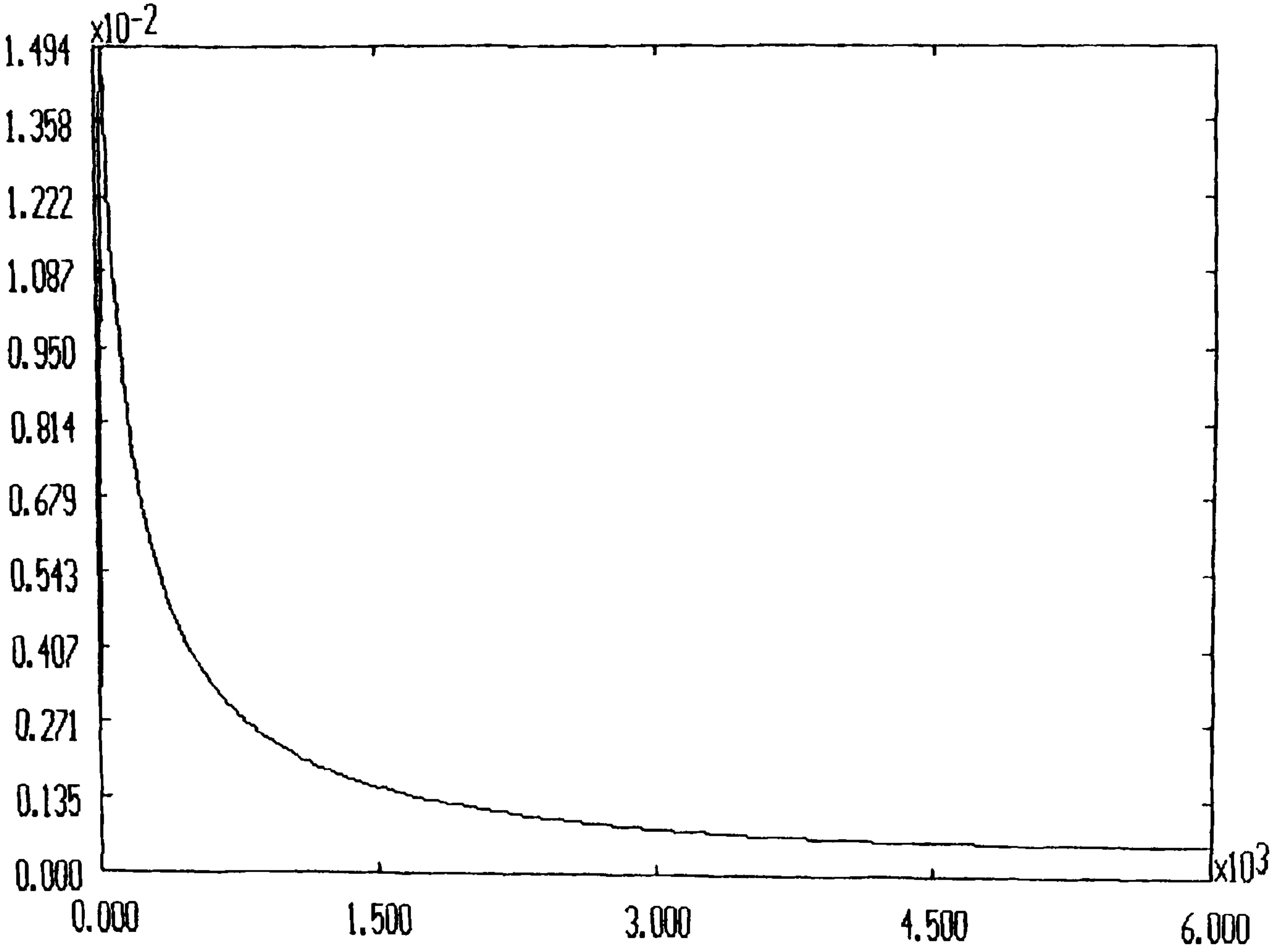

Figure 4.22 Amplitude of Velocity of the Second Mode for $N_m = 4$, $N_s = 3$, $N_c = 2$

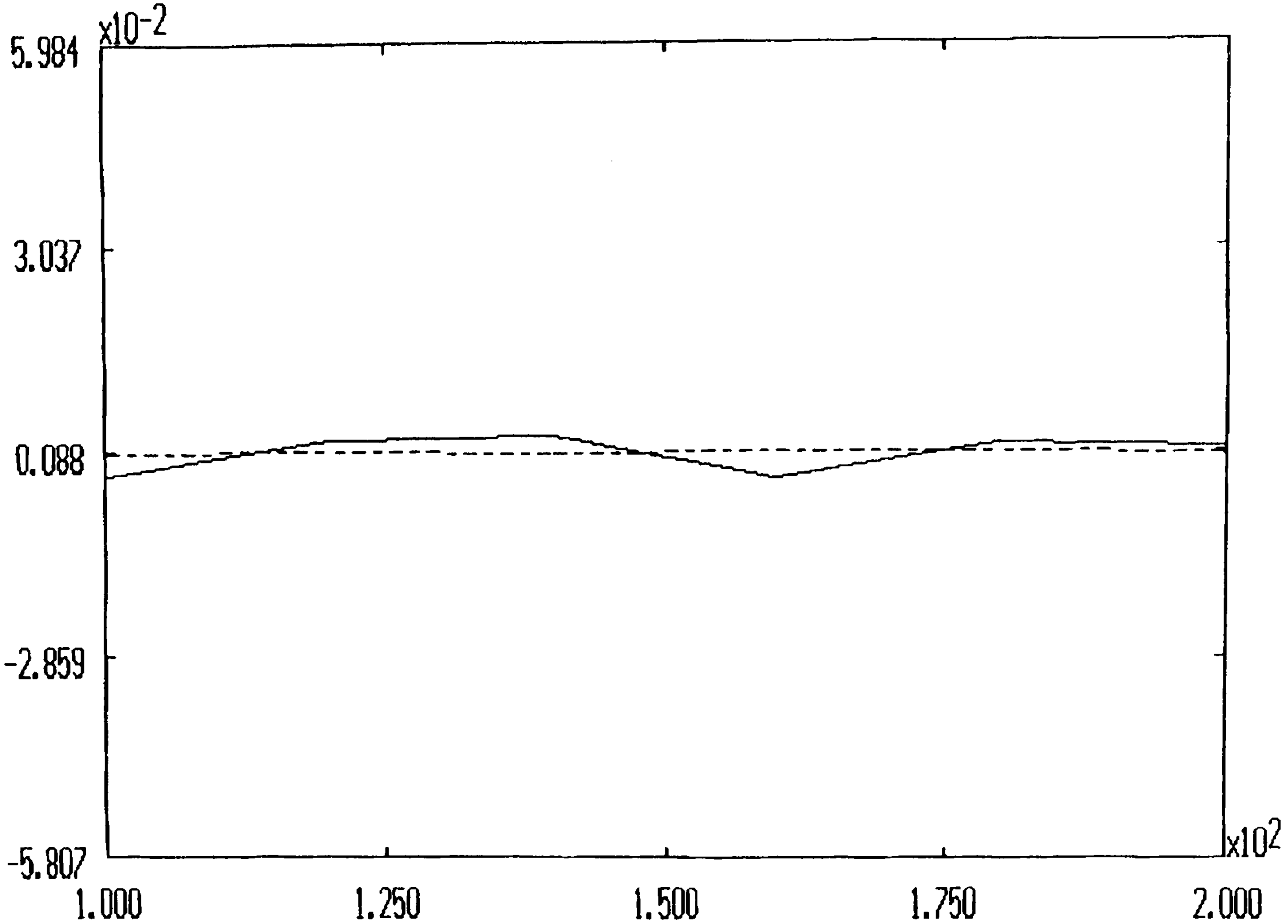

Figure 4.23 Velocity of the First Mode and It's Estimation for $N_m = 4$, $N_s = 3$, $N_\upsilon = 2$

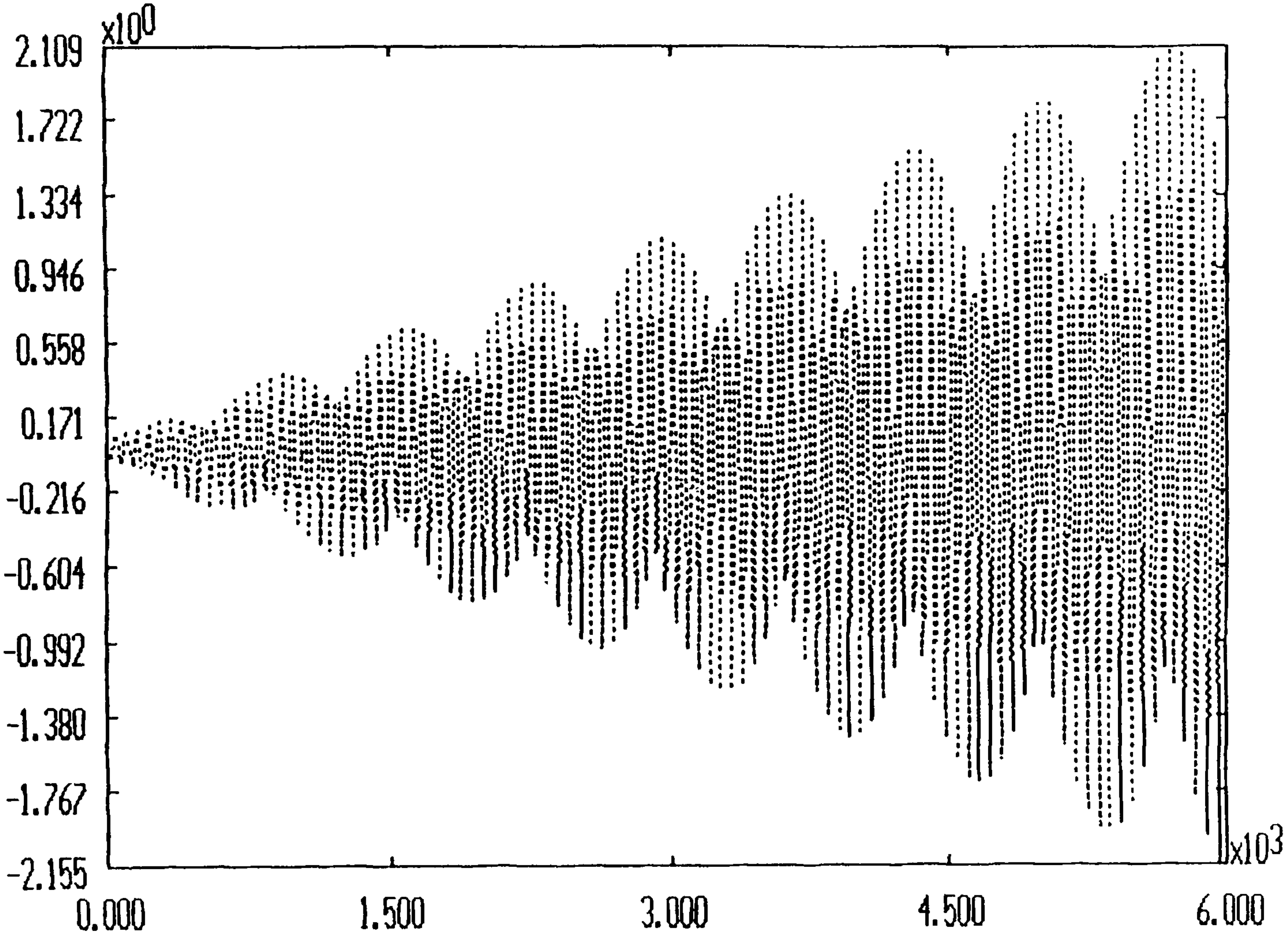

Figure 4.24 Velocity of the Second Mode and It's Estimation for $N_m = 4$, $N_s = 3$, $N_o = 2$

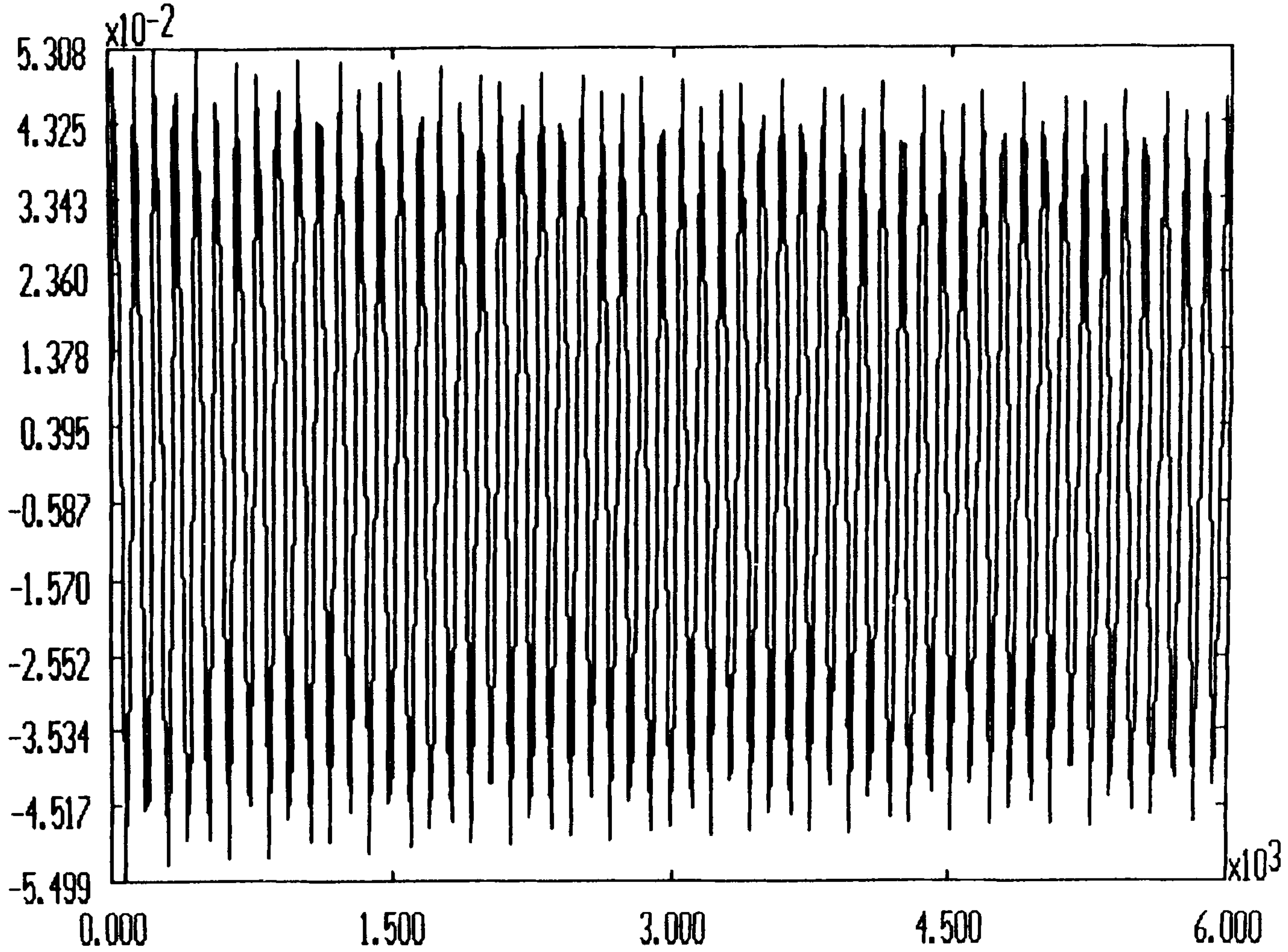

Figure 4.25 Velocity of the Third Mode $N_m = 4$, $N_s = 3$, $N_o = 2$

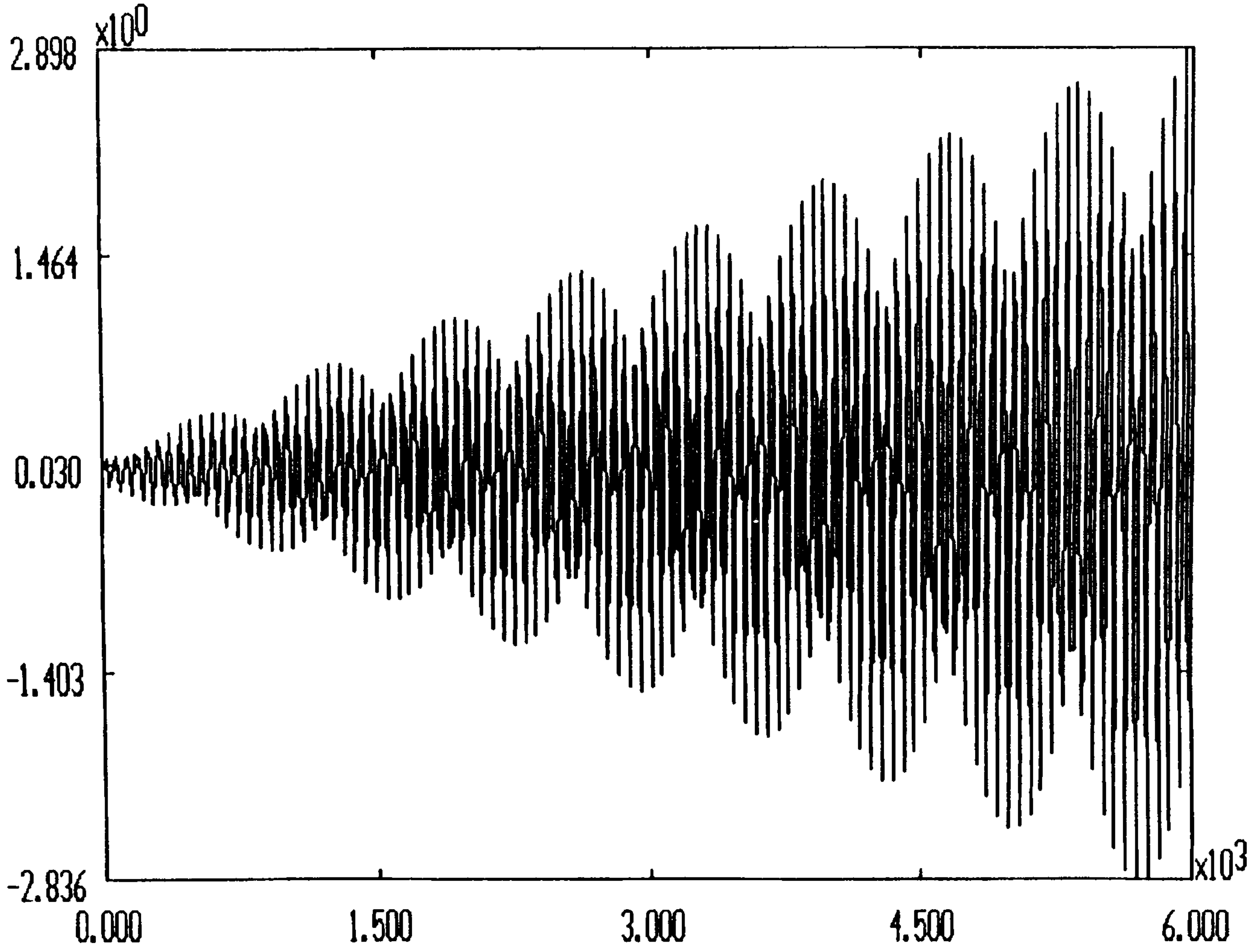

Figure 4.26 Velocity of the Fourth Mode $N_m = 4$, $N_s = 3$, $N_o = 2$

CHAPTER 5
SENSORS

Certain solid-state materials are electrically responsive to mechanical force. These materials are often used in the transduction of mechanical phenomena to electrical phenomena. These materials can be divided into two main categories:

1. Self generating type - applied force generates electrical charge;

2. Passive circuit type - applied force causes a change in the electrical characteristics of the material.

Piezoelectric materials are of the self-generating type. The piezoelectric effect arises because when an asymmetric crystal lattice is distorted, an internal charge reorientation takes place, and this causes a relative displacement of positive and negative charges to opposite outer surfaces of the crystal.

The piezoelectric charge constant relates stress to charge density and piezoelectric force constant relates strain to electric field. Piezoelectric sensors are basically dielectrics with a high but finite leakage resistance. This insulating property allows the sensor to be modeled as a parallel-plate capacitor. The internal parallel plate structure of the sensor with lossy medium, characterized by conductivity σ, permitivity ε is shown in Figure(5.1).

5.1 Low-frequency Equivalent Circuit

The total induced charge produced on the sensor is directly proportional to the applied force:

$$q = p\,F = k_1\,F \tag{5.1}$$

where p is the piezoelectric constant, in coulombs per newton.

Within elastic limits, a force applied to a sensor surface deflects it according to

$$F = k_2\,x \tag{5.2}$$

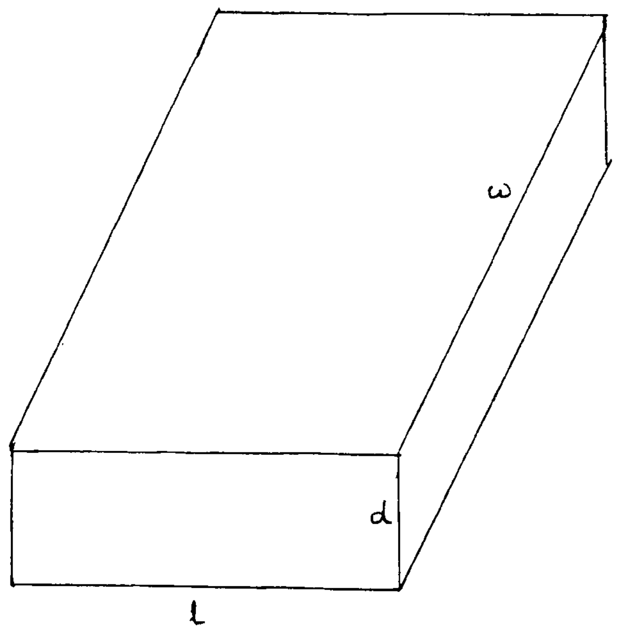

Figure 5.1 Parallel Plate Structure of Piezoelectric Transducer

where x is displacement.

By substituting equation (5.2) into equation (5.1)

$$q = k_1\, k_2\, x = K\, x \tag{5.3}$$

The charge generator can be converted to a current generator by differentiation and we can model an equivalent circuit as shown in Figure(5.2)

$$i_t = \frac{dq}{dt} = \frac{K\, dx}{dt} \tag{5.4}$$

where $\frac{dx}{dt}$ is the pick off velocity , we are interested.

The capacitance between two parallel plates of length l, width w, separated by a distance d is

$$C = \frac{\varepsilon\, l\, w}{d} \tag{5.5}$$

where ε is the permitivity constant of the medium.

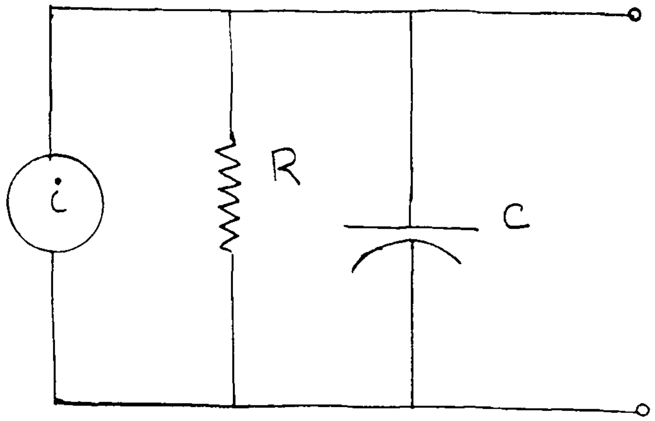

Figure 5.2 Low Frequency Equivalent Circuit of Piezoelectric Transducer

Similarly, resistance is given by

$$R = \frac{d}{\sigma\, w\, l} \tag{5.6}$$

where σ is conductivity of the medium.

In frequency domain, the input admittance Y of the structure is given by

$$Y = \frac{I}{V} = j\omega C + \frac{1}{R} \tag{5.7}$$

$$\begin{aligned} V &= I\frac{R}{1 + j\omega RC} \\ &= I\frac{R - j\omega RC}{1 + (\omega RC)^2} \end{aligned} \tag{5.8}$$

At low frequencies, $\omega RC <<< 1$, reducing (5.8) to

$$V = IR \tag{5.9}$$

Figure(5.3) shows frequency response of a piezoelectric transducer.

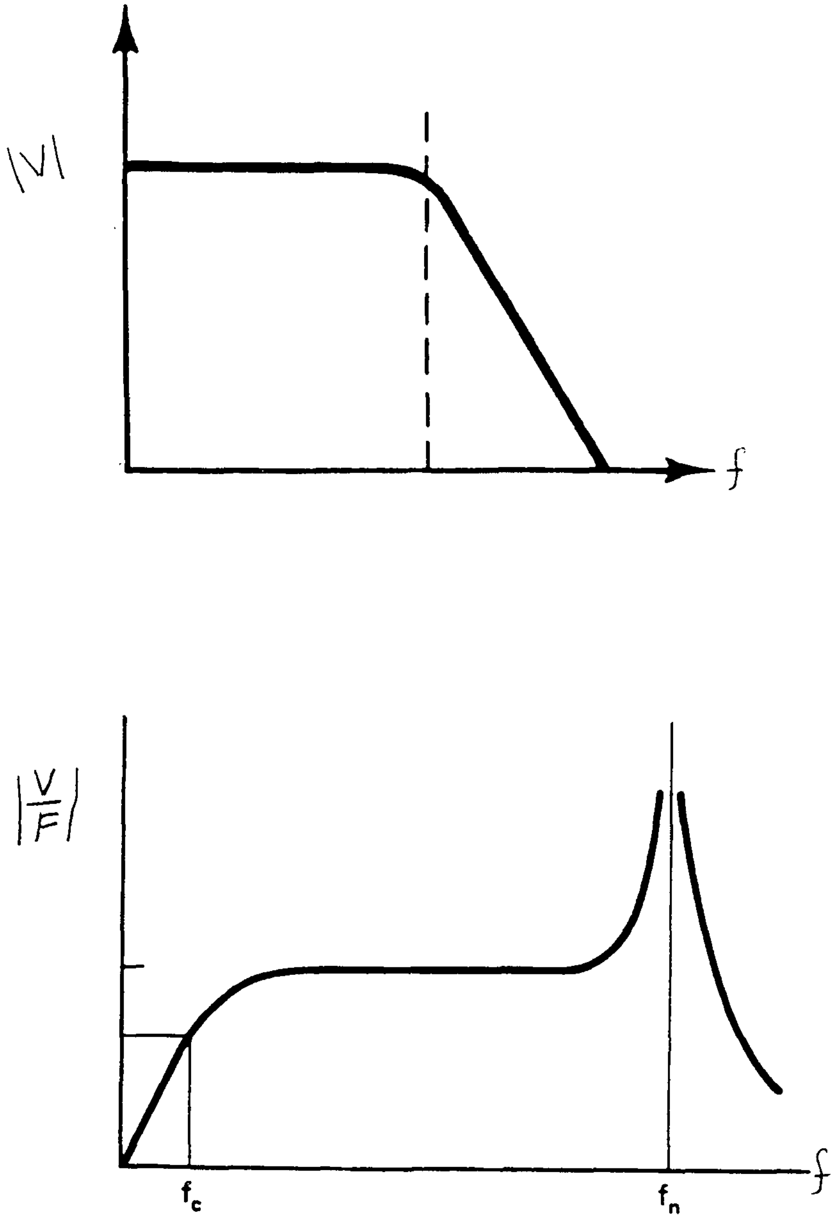

Figure 5.3 Frequency Response of Piezoelectric Transducer

5.2 Application to the Beam Experiment

Each sensor is placed on the beam where we want to measure the velocity. Suppose we want to control the first mode. The value of the ith mode shape can be calculated from equation (2.9), at that particular point on the beam where the sensor is placed, and incorporated in the sensor as an scale factor as described below.

Using Euler's estimation from equation 3.4, we have

$$\int_0^l \dot{y}_i(x,t)\, X_i(x)\, dx = \sum_{k=1}^{N_s} \dot{y}(kh)\ X_i(kh)\ h$$

Now the voltage from that particular sensor is proportional to the term $\dot{y}(kh,t)$, where $x = k\,h$ is the pick off point, the sensor is placed on the beam. Now our aim is to incorporate the value of $X_i(k\,h)$ as a gain on the sensor. We can do this by changing the values of R in equation 5.9.

Depending on the output signal, either voltage or current we are interested to measure, $X_i(x)$ can be made either directly proportional to R, or inversely proportional to R.

Suppose we are interested in measuring voltage signal, from (5.9) and (5.4), we have

$$V = K\,\frac{dx}{dt}\,R \tag{5.10}$$

By comparing (5.10) and (3.4)

$$X_i(x) \propto R \tag{5.11}$$

Substituting the value of R from (5.6), we have

$$X_i(x) \propto \frac{d}{\sigma\,w\,l} \tag{5.12}$$

Now, if we connect all the sensors in series, as shown in Figure(5.4), the resulting voltage will give the velocity of the ith mode.

Suppose, if we want to measure current signal, from (5.9) and (5.4), we have

$$I_s = K\,\frac{dx}{dt} = I\,\frac{R}{R_s + R} \tag{5.13}$$

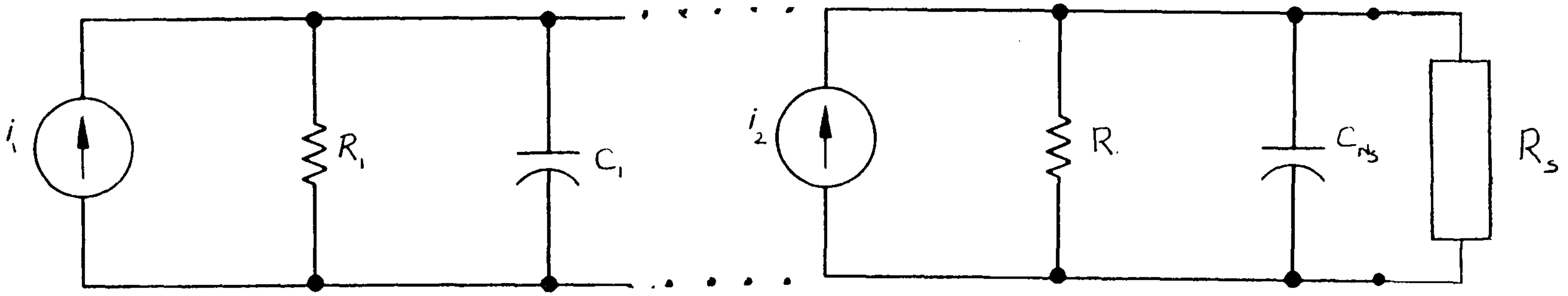

Figure 5.4 Serial Implementation for Measuring ith Mode Velocity

Since $R_s >>> R$

$$X_i(x) \propto R \tag{5.14}$$

Now, if we connect all the sensors in parallel, as shown in Figure(5.5), the resulting current will give the velocity of the ith mode.

Therefore, changing the effective surface area (i.e. w or l) of the piezoelectric wafer can provide the necessary gain (mode shape at kh, $k = 1, 2, \cdots, N_s$ for integration.

Suppose, we want to control multiple number modes. Instead of using one sensor for each mode, we can take advantage of smart materials and reduce the number of sensors by a factor, equal to the number of modes, to be controlled. Using smart materials, we can implant multiple number of sensors in a single chip. On each sensor the value of mode we want to control will be placed by fetching different surface areas. And all the sensors that belong to a particular mode are connected in either series or parallel depending on the signal we want to measure. Figure (5.6) shows a piezoelectric wafer, containing multiple number of sensors.

Signal Conditioning:

There are two methods for conditioning the signals from piezoelectric transducer.

1. Voltage amplification

2. Charge amplification

Voltage Amplification:

In voltage amplification, the amplifier must have a high input impedance, because the addiction of cable capacitance reduces the voltage signal seen by the amplifier. This can be avoided, by making use of a voltage follower circuit with high input impedance, which converts the signal from the high-impedance transducer to a

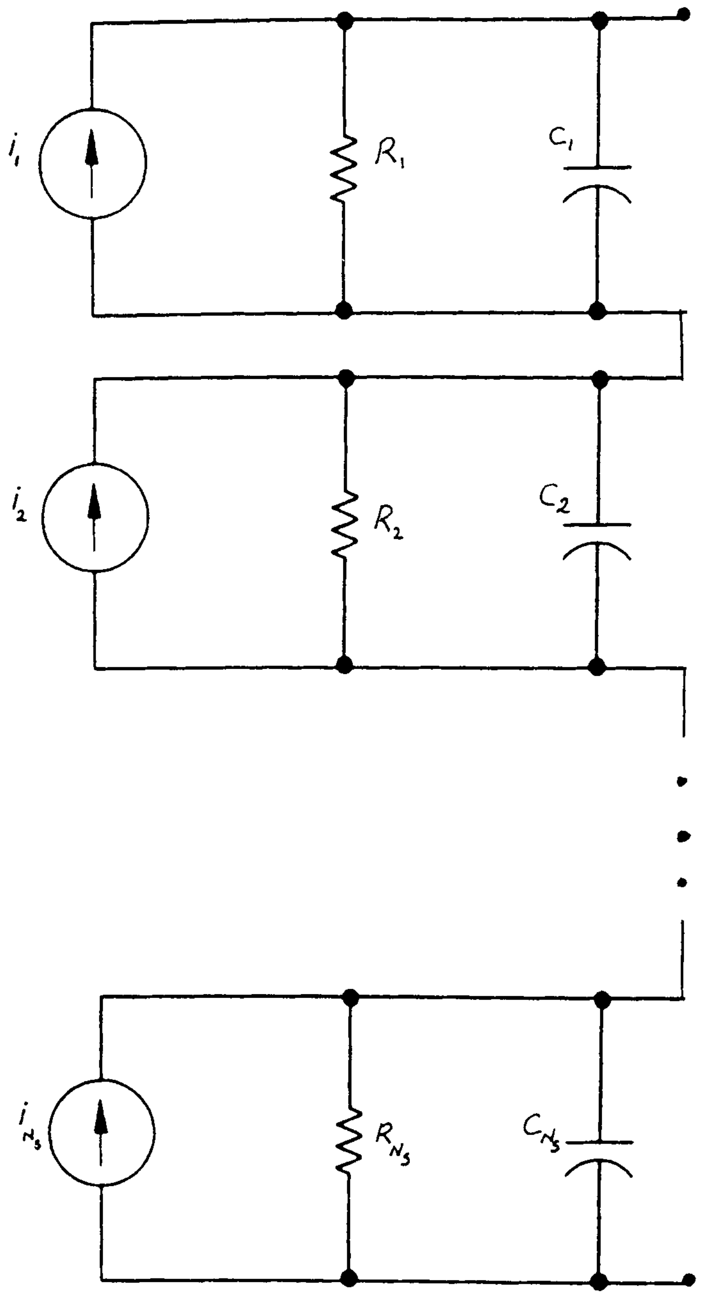

Figure 5.5 Parallel Implementation for Measuring ith Mode Velocity

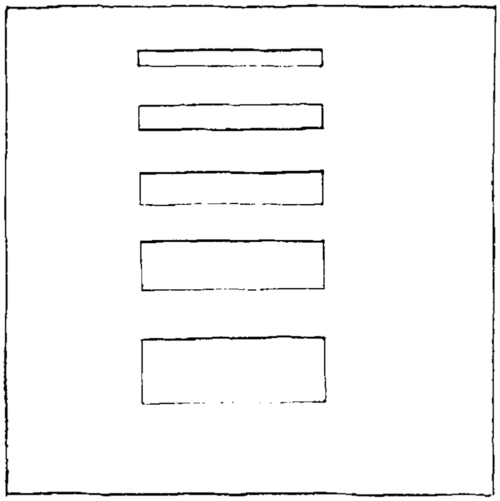

Figure 5.6 Piezoelectric Wafer Containing Multiple Number of Sensors

voltage output at low impedance. The voltage follower (unity gain), voltage amplifier with gain equal to $(R_2 + R_1)/R_1$ is shown in Figure(5.7). Sensor signal conditioning devices such as the TLC2272/4 opamp features a 10^{12} Ω input impedance could be used in this application.

Charge Amplification:

This method makes use of the fact that the low frequency response of the transducer-amplifier system is independent of transducer and cable capacitance. As shown in Figure(5.8), the input of the high-impedance amplifier is a virtual ground. Therefore, all of the charge generated then flows to the feedback capacitor, and the output voltage is the negative of the voltage on the capacitor. A large feedback resistor must be added across the capacitor, to avoid the output voltage drift, caused by opamp bias currents.

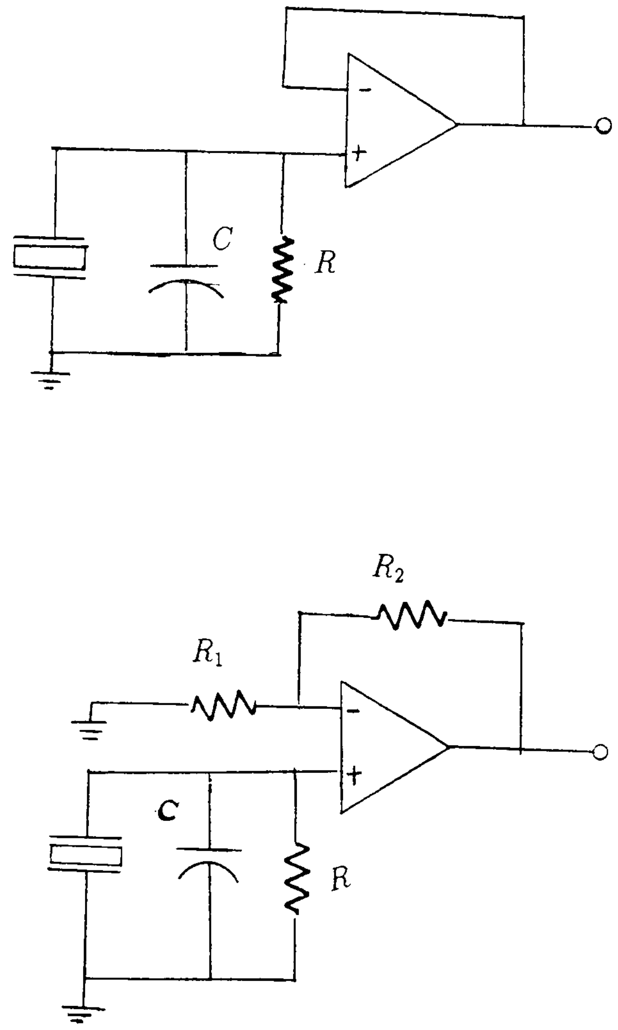

Figure 5.7 Voltage Amplifier: a) unity gain; b) with gain

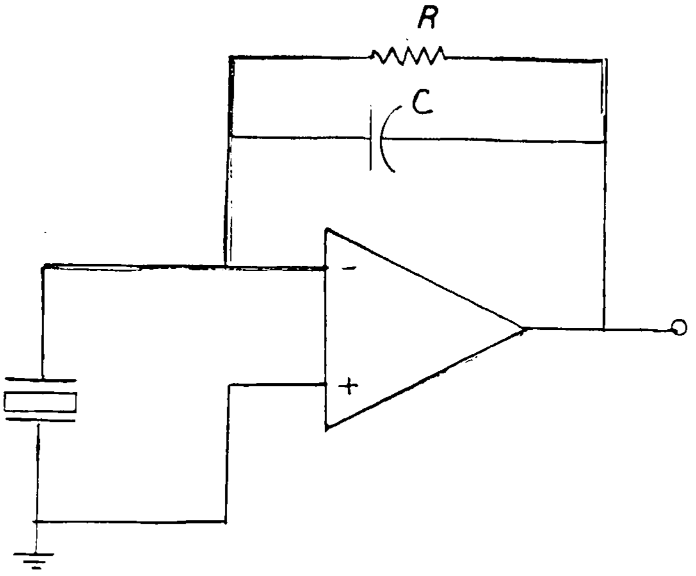

Figure 5.8 Charge Amplifier

CHAPTER 6

CONCLUSIONS AND SUGGESTIONS FOR FUTURE RESEARCH

Three objectives have been attained in this work: modelling of flexible beam dynamics, estimation of modal velocities using numerical methods, and stabilization of a flexible beam. The approach used in this work consists of 3 stages :

1. Determine the model structure and plant parameters, such as the natural frequencies, and mode shapes.

2. Estimating modal velocities by using numerical methods.

3. Stabilize and regulate beam dynamics by feeding back the estimated modal velocities, with proper gain.

Stage 1 is carried out, by applying classical beam theory. In stage 2, Euler's and Simpson's methods are used for estimating modal velocities, and a method for finding the number of sensors needed for good estimation is described. In stage 3, stabilization regulation of beam dynamics is carried out, by employing the scheme described in stage 2, and simulations have been carried out, using the ALSIM software. In conclusion, the simulation results, established the scheme described in stage 2.

As for future development, the above scheme can be implemented using a smart material beam, sensors and a properly set up hardware. For example, a TMS320C25 DSP card can be used for signal processing and interfacing. The number of sensors required can be further reduced, by coming up with a more efficient integration technique.

APPENDIX

1. BEAM PROPERTIES

Dimension	:	133" x 3" x 3/16"
Liner density	:	$p = 0.158$ lb/in.
Young's Modulus	:	$E = 2.9x10^7 lb/in^2$.
Area Moment of Inertia	:	$I = bh^3/12 = 1.65x10^{-3} in^4$.
End mass to Beam Mass Ratio	:	$K = 5.18$.

2. MATLAB ROUTINES

2.1 To calculate Maximum Singular Values Of (M -I) matrix Euler Approximation.

```
   l = 133;     % length of the beam
   K = 5.18;    % ratio of end mass to beam mass
   m = 20;      % number of modes
   n = 34;      % number of sensors
   h = l/n;     % spacing between sensors
   i = 1:1:20; % index for number of modes
   j = 1:1:n;  % index for number of sensors
   I = eye(m,m)
   I2 = [ eye(2,2) zeros(2,m-2) ]
   I4 = [ eye(4,4) zeros(4,m-4) ]
   I6 = [ eye(6,6) zeros(6,m-6) ]
   %MU(i) term
  U(i)=[ 3.1711 6.2983 9.4349 12.574 15.7141 18.8546 21.9955 25.1366 ..
    28.2777 31.419 34.5603 37.7017 40.8431 43.9845 47.1259 50.2674 ..
    53.4089 56.5504 59.6919 62.8334 ];
  % points at where the sensors are placed on the beam
  for j = 1:1:n
  x(j) = h*j;
  end;
%
% the following loop calulates the constant terms that are needed for
% calulating mode shapes, velocities
for i = 1:1:m
% square of cotangent term for calculating B(i)
   ctsqr(i) = (cos(U(i))*cos(U(i)))/(sin(U(i))*sin(U(i)));
% square of hyperbolic cotangent term for calculating B(i)
   cthsqr(i) = (cosh(U(i))*cosh(U(i)))/(sinh(U(i))*sinh(U(i)));
% B(i) term from your project for calculating mode shapes
   B(i) = 1/(sqrt(l*(K+1+.5*(ctsqr(i)-cthsqr(i))))); %
  end;
%
% the following loops calulate modes at differnt points on the beam
% the j loop is for different sensors(points on beam)
% the i loop is for different modes
for j = 1:1:n
 for i = 1:1:m
% sine term for calculating mode shapes
 sn(i,j) = (sin((U(i)*x(j))/l))/(sin(U(i)));
% heperbolic sine term for calculating mode shapes
 snh(i,j) = (sinh((U(i)*x(j))/l))/(sinh(U(i)));
% calculation of mode shapes at different points on the beam
 X(i,j) = B(i)*(sn(i,j)+snh(i,j));
 end;
 end;
for i = 1:1:m
for k = 1:1:m
Mode(i,k) = 0;
for j = 1:1:n
 Mode(i,k) = X(i,j)*X(k,j)*h + Mode(i,k);
end;
Mode(i,k) = Mode(i,k) + 4*K*l*B(i)*B(k);
end;
end;
for i = 1:1:m
for k = 1:1:m
M = Mode(1:m,1:m)
M2 = Mode(1:2,1:m)
M4 = Mode(1:4,1:m)
M6 = Mode(1:6,1:m)
end;
end;
SgM = (M -I)
SgM2 = (M2-I2)
SgM4 = (M4 - I4)
SgM6 = (M6 - I6)
Sgvl2034 = svd(SgM)     % singular values, for Nm = 20, Ns = 34, No =20.
Sgvl20342 = svd(SgM2)  % singular values, for Nm = 20, Ns = 34, No =2.
Sgvl20344 = svd(SgM4)  % singular values, for Nm = 20, Ns = 34, No =4.
Sgvl20346 = svd(SgM6)  % singular values, for Nm = 20, Ns = 34, No =6
```

2.2 To calculate Velocity of Each Mode and it's Estimation, Using Euler's Approximation.

```
s = 0;          % damping
Qo = .1;        % initial displacement
dervQo = .1;  % intial velocity
l = 133;
K = 5.18;
```

```
m = 20;
n = 22;
i = 1:1:20;
j = 1:1:n;
h = 1/n;
for j = 1:1:n
x(j)=j*h;
end;
%MU(i) term
U(i)=[ 3.1711 6.2983 9.4349 12.574 15.7141 18.8546 21.9955 25.1366 ..
     28.2777 31.419 34.5603 37.7017 40.8431 43.9845 47.1259 50.2674 ..
     53.4089 56.5504 59.6919 62.8334 ];
W(i) = [ .3128 1.2341 2.7694 4.9188 7.6822 11.0597 15.0514 19.6572 ..
       24.877 30.711 37.159 44.2212 51.8975 60.1878 69.0921 78.6108 ..
       88.7435 99.4902 110.8511 122.826 ];
for i = 1:1:m
   ctsqr(i) = (cos(U(i))*cos(U(i)))/(sin(U(i))*sin(U(i)));
   cthsqr(i) = (cosh(U(i))*cosh(U(i)))/(sinh(U(i))*sinh(U(i)));
   B(i) = 1/(sqrt(1*(K+1+.5*(ctsqr(i)-cthsqr(i)))));
   Wd(i) = W(i)*sqrt(1-(s*s));
   a(i) = (dervQo + (s*W(i)*Qo))/Wd(i);
   A(i) = sqrt((Qo*Qo) + (a(i)*a(i)));
   alph(i) = atan((dervQo +(s*W(i)*Qo))/(Wd(i)*Qo));
   end;
for i = 1:1:m
 for j = 1:1:n
 sn(i,j) = (sin((U(i)*x(j))/l))/(sin(U(i)));
 snh(i,j) = (sinh((U(i)*x(j))/l))/(sinh(U(i)));
 X(i,j) = B(i)*(sn(i,j)+snh(i,j));
 end;
 end;
%
t = 1:1:100; % index for time
T(1) = 0; % intializing time
% the following for loop decides at which instants(time) the derivative
% term (velocity) has to be calculated
for t = 2:1:100
T(t) = T(t-1)+.1; % time is incremented by .1
end;
%
% the following loops calculate different modes at difeeerent points
% on the beam at differnt instants of time
% t loop for differnt instants of time
% j loop for different sensors (points) on the beam
% i loop for differnt modes
  for t = 1:1:100
   for i = 1:m
%constant term for calculating velocity, PHIdot from your project
```

```
q(i,t)=((-A(i))*exp((-s)*W(i)*T(t)));
%calculation of velocity,PHIdot form your project
dervQ(i,t)=q(i,t)*((s*W(i)*cos(Wd(i)*T(t)-alph(i)))+(Wd(i)*sin(Wd(i)
*T(t)-alph(i))));
end;
end;
for t = 1:1:100
for j = 1:1:n
dervY(j,t) = 0;
%calculation ofvelocity of beam, Ydot from your project
for i = 1:1:m
 dervY(j,t) = (X(i,j)*dervQ(i,t)) + dervY(j,t);
 end;
 end;
 end;
%the following loops calculate the Estimating term of velocity
%which is SIGHdot from your notes
 for t = 1:1:100
 for i = 1:1:m
  dervchi(i,t) = 0;
   for j = 1:n
  dervchi(i,t)=(dervY(j,t)*X(i,j)*h)+dervchi(i,t);
   end;
dervchi(i,t) = K*l*dervY(n,t)*X(i,n) + dervchi(i,t);
    end;
end
```

2.3. To calculate Maximum Singular Values Of (M -I) matrix For Simpson's Approximation.

```
l = 133;  % length of the beam
K = 5.18; % ratio of end mass to beam mass
m = 20; % number of modes
n = 26; % number of sensors
h = l/n; %spacing between sensors
i = 1:1:20; % index for number of modes
j = 1:1:n; % index for number of sensors
I = eye(m,m)
I2 = [ eye(2,2) zeros(2,m-2) ]
I4 = [ eye(4,4) zeros(4,m-4) ]
I6 = [ eye(6,6) zeros(6,m-6) ]
%MU(i) term
U(i)=[ 3.1711 6.2983 9.4349 12.574 15.7141 18.8546 21.9955 25.1366 ..
     28.2777 31.419 34.5603 37.7017 40.8431 43.9845 47.1259 50.2674 ..
     53.4089 56.5504 59.6919 62.8334 ];
```

```
% points at where the sensors are placed on the beam
for j = 1:1:n
x(j) = h*j;
end;
%
% the following loop calulates the constant terms that are needed for
%calulating mode shapes, velocities
for i = 1:1:m
% square of cotangent term for calculating B(i)
   ctsqr(i) = (cos(U(i))*cos(U(i)))/(sin(U(i))*sin(U(i)));
% square of hyperbolic cotangent term for calculating B(i)
   cthsqr(i) = (cosh(U(i))*cosh(U(i)))/(sinh(U(i))*sinh(U(i)));
% B(i) term from your project for calculating mode shapes
   B(i) = 1/(sqrt(1*(K+1+.5*(ctsqr(i)-cthsqr(i))))); %
  end;
%
% the following loops calulate modes at differnt points on the beam
% the j loop is for different sensors(points on beam)
% the i loop is for different modes
for j = 1:1:n
 for i = 1:1:m
% sine term for calculating mode shapes
 sn(i,j) = (sin((U(i)*x(j))/l))/(sin(U(i)));
% heperbolic sine term for calculating mode shapes
 snh(i,j) = (sinh((U(i)*x(j))/l))/(sinh(U(i)));
% calculation of mode shapes at different points on the beam
 X(i,j) = B(i)*(sn(i,j)+snh(i,j));
 end;
 end;
for i = 1:1:m
for k = 1:1:m
Mode(i,k) = 0;
for j = 1:1:n
    if ((rem(j,2) == 0) & (j == n))
       Mode(i,k) = X(i,j)*X(k,j) + Mode(i,k)
   elseif ((rem(j,2) == 0) & (j ~= n))
     Mode(i,k) = 2*(X(i,j)*X(k,j)) + Mode(i,k)
   else
    Mode(i,k) = 4*(X(i,j)*X(k,j)) + Mode(i,k)
     end;
  end;
Mode(i,k) =(1/(3*n))*Mode(i,k) + 4*K*l*B(i)*B(k);
end;
end;
M = Mode(1:m,1:m)
M2 = Mode(1:2,1:m)
M4 = Mode(1:4,1:m)
M6 = Mode(1:6,1:m)
```

```
SgM = (M -I)
SgM2 = (M2-I2)
SgM4 = (M4 - I4)
SgM6 = (M6 - I6)
Sgvl2026 = svd(SgM)
Sgvl20262 = svd(SgM2)
Sgvl20264 = svd(SgM4)
Sgvl20266 = svd(SgM6)
```

2.4 To calculate Velocity of Each Mode and it's Estimation, Using Simpson's Approximation.

```
s = 0;
Qo = .1;
dervQo = .1;
l = 133;
K = 5.18;
m = 4;
n = 6;
i = 1:1:20;
j = 1:1:n;
h = l/n;
for j = 1:1:n
x(j)=j*h;
end;
%MU(i) term
U(i)=[ 3.1711 6.2983 9.4349 12.574 15.7141 18.8546 21.9955 25.1366 ..
     28.2777 31.419 34.5603 37.7017 40.8431 43.9845 47.1259 50.2674 ..
     53.4089 56.5504 59.6919 62.8334 ];
W(i) = [ .3128 1.2341 2.7694 4.9188 7.6822 11.0597 15.0514 19.6572 ..
       24.877 30.711 37.159 44.2212 51.8975 60.1878 69.0921 78.6108 ..
       88.7435 99.4902 110.8511 122.826 ];
for i = 1:1:m
   ctsqr(i) = (cos(U(i))*cos(U(i)))/(sin(U(i))*sin(U(i)));
   cthsqr(i) = (cosh(U(i))*cosh(U(i)))/(sinh(U(i))*sinh(U(i)));
   B(i) = 1/(sqrt(l*(K+1+.5*(ctsqr(i)-cthsqr(i)))));
   Wd(i) = W(i)*sqrt(1-(s*s));
   a(i) = (dervQo + (s*W(i)*Qo))/Wd(i);
   A(i) = sqrt((Qo*Qo) + (a(i)*a(i)));
   alph(i) = atan((dervQo +(s*W(i)*Qo))/(Wd(i)*Qo));
   end;
for i = 1:1:m
 for j = 1:1:n
 sn(i,j) = (sin((U(i)*x(j))/l))/(sin(U(i)));
 snh(i,j) = (sinh((U(i)*x(j))/l))/(sinh(U(i)));
 X(i,j) = B(i)*(sn(i,j)+snh(i,j));
 end;
 end;
%
t = 1:1:100; % index for time
T(1) = 0; % intializing time
% the following for loop decides at which instants(time) the derivative
% term (velocity) has to be calculated
for t = 2:1:100
T(t) = T(t-1)+.1; % time is incremented by .1
end;
%
% the following loops calculate different modes at difeerent points
% on the beam at differnt instants of time
% t loop for differnt instants of time
% j loop for different sensors (points) on the beam
% i loop for differnt modes
  for t = 1:1:100
   for i = 1:m
%constant term for calculating velocity, PHIdot from your project
q(i,t)=((-A(i))*exp((-s)*W(i)*T(t)));
%calculation of velocity,PHIdot form your project
dervQ(i,t)=q(i,t)*((s*W(i)*cos(Wd(i)*T(t)-alph(i)))+(Wd(i)*sin(Wd(i)
*T(t)-alph(i))));
end;
end;
for t = 1:1:100
for j = 1:1:n
dervY(j,t) = 0;
%calculation ofvelocity of beam, Ydot from your project
for i = 1:1:m
 dervY(j,t) = (X(i,j)*dervQ(i,t)) + dervY(j,t);
 end;
 end;
 end;
%the following loops calculate the Estimating term of velocity
%which is SIGHdot from your notes
 for t = 1:1:100
 for i = 1:1:m
  dervchi(i,t) = 0;
%summation of velocity(beam-Ydot) terms at differnt points on beam
  for j = 1:1:n
    if ((rem(j,2) == 0) & (j == n))
       dervchi(i,t) = dervY(j,t)*X(i,j) + dervchi(i,t)
   elseif ((rem(j,2) == 0) & (j ~= n))
          dervchi(i,t) = 2*(dervY(j,t)*X(i,j)) + dervchi(i,t)
        else
          dervchi(i,t) = 4*(dervY(j,t)*X(i,j)) + dervchi(i,t)
```

```
     end;
  end;
dervchi(i,t) = (1/(n*3))*dervchi(i,t) + (dervY(n,t)*X(i,n)*K*l);
 end;
 end
```

3. ALSIM FILES for SIMULATION

3.1 SIMULATION OF OPEN LOOP DYNAMICS for $N_m = 4$.

3.1.1 Dynamic File

```
#include "\ALSIM\ALSIM.H"
#include "MATH.H"
#include "STDIO.H"
#define omega12     fpar[1]
#define omega22     fpar[2]
#define omega32     fpar[3]
#define omega42     fpar[4]
/*
**  User state derivative function.
*/

derv(t, x, dxdt)
double t, *x, *dxdt;
{
dxdt[1] = x[2];
dxdt[2] = -omega12*x[1];
dxdt[3] = x[4];
dxdt[4] = -omega22*x[3];
dxdt[5] = x[6];
dxdt[6] = -omega32*x[5];
dxdt[7] = x[8];
dxdt[8] = -omega42*x[7];
}
```

3.1.2 Rundata File

```
0     ;initial time
100.    ;final time
0.01    ;maximum stepsize
1.0e-6  ;minimum stepsize
0.001   ;fractional error criterion
```

```
200      ;multiple of maximum stepsize for print output
20       ;multiple of maximum stepsize for plot output

8        ;number of plant states
0        ;number of plant inputs
0        ;number of plant outputs
0        ;number of controller states

0        ;size of user defined plot vector
0        ;size of user common area
0        ;size of gaussian random number vector
         ;vector multiplied by sqrt(hmax) to provide approx. uniform
         ;variance for variable stepsize
318      ;random number seed
272      ;random number seed
190      ;random number seed

0        ;number of user defined integer input parameters
0,0      ;end integer input parameters

4          ;number of user defined floating point input parameters
1,.0978    ;omega12
2,1.523    ;omega22
3,7.6696   ;omega32
4,24.1946  ;omega42
0,0        ;end floating point input parameters

1,.02    ;
3,.02    ;
5,.02    ;
7,.02    ;
0,0      ;end plant initial conditions

0,0      ;end controller initial conditions
```

3.2 SIMULATION OF CLOSED LOOP DYNAMICS for Nm = 4, No =4, Ns =6.

3.2.1 Dynamic File

```
#include "\ALSIM\ALSIM.H"
#include "MATH.H"
#include "STDIO.H"
#define omega12    fpar[1]
```

```
#define cutoff12   fpar[2]
#define cutoff1sq2 fpar[3]
#define refer      fpar[4]
#define ppgain1    fpar[5]
#define intgain1   fpar[6]
#define omega22    fpar[7]
#define cutoff22   fpar[8]
#define cutoff2sq2 fpar[9]
#define ppgain2    fpar[10]
#define intgain2   fpar[11]
#define omega32    fpar[12]
#define cutoff32   fpar[13]
#define cutoff3sq2 fpar[14]
#define ppgain3    fpar[15]
#define intgain3   fpar[16]
#define omega42    fpar[17]
#define cutoff42   fpar[18]
#define cutoff4sq2 fpar[19]
#define ppgain4    fpar[20]
#define intgain4   fpar[21]
#define refscale   fpar[22]
float est_x2,cal_x2,est_x7,cal_x7,est_x12;
float cal_x12,est_x17,cal_x17;
/*
**  User state derivative function.
*/

derv(t, x, dxdt)
double t, *x, *dxdt;
{
dxdt[1] = x[2];
dxdt[2] = -omega12*x[1]+ u[1];
plotout[1] = fabs(x[2]);
dxdt[3] = x[4];
dxdt[4] = -cutoff12*x[3] -cutoff1sq2*x[4] + cutoff12*plotout[1];
plotout[2] = x[3]-refer;
dxdt[5] = plotout[2];
dxdt[6] = x[7];
dxdt[7] = -omega22*x[6] + u[1];
plotout[3] = fabs(x[7]);
dxdt[8] = x[9];
dxdt[9] = -cutoff22*x[8] -cutoff2sq2*x[9] +cutoff22*plotout[3];
plotout[4] = x[8] - refer;
dxdt[10] = plotout[4];
dxdt[11] = x[12];
dxdt[12] = -omega32*x[11] + u[1];
plotout[5] = fabs(x[12]);
dxdt[13] = x[14];
```

```
dxdt[14] = -cutoff32*x[13] -cutoff3sq2*x[14] + cutoff32*plotout[5];
plotout[6] = x[13]-refer;
dxdt[15] = plotout[6];
dxdt[16] = x[17];
dxdt[17] = -omega42*x[16] + u[1];
plotout[7] = fabs(x[17]);
dxdt[18] = x[19];
dxdt[19] = -cutoff42*x[18] -cutoff4sq2*x[19] + cutoff42*plotout[7];
plotout[8] = x[18] - refer;
dxdt[20] = plotout[8];
y[1] = -.0614*x[2] +.1062*x[7] -.1226*x[12] +.1061*x[17];
y[2] = -.1059*x[2] +.1058*x[7] +.0004*x[12] -.1063*x[17];
y[3] = -.1214*x[2] -.0008*x[7] +.1226*x[12] +.0005*x[17];
y[4] = -.1033*x[2] -.1065*x[7] -.0008*x[12] +.1059*x[17];
y[5] = -.0563*x[2] -.1046*x[7] -.1223*x[12] -.1064*x[17];
y[6] = +.0072*x[2] +.0037*x[7] +.0025*x[12] +.0019*x[17];
cal_x2 =22.17*(-.0614*y[1]-.1059*y[2]-.1214*y[3]-.1033*y[4]-.0563*y[5]
+.0072*y[6]);
est_x2 = cal_x2  + 5.18*133*.0072*y[6];
cal_x7 =22.17*(.1062*y[1]+.1058*y[2]-.0008*y[3]-.1065*y[4]-.1046*y[5]
+.0037*y[6]);
est_x7 = cal_x7  + 5.18*133*.0037*y[6];
cal_x12=22.17*(-.1226*y[1]+.0004*y[2]+.1226*y[3]-.0008*y[4]-.1223*y[5]
+.0025*y[6]);
est_x12 = cal_x12  + 5.18*133*.0025*y[6];
cal_x17 =22.17*(.1061*y[1]-.1063*y[2]+.0005*y[3]+.1059*y[4]-.1064*y[5]
+.0019*y[6]);
est_x17 = cal_x17  + 5.18*133*.0019*y[6
u[1] = refscale*(ppgain1*est_x2*plotout[2]+ppgain2*est_x7*plotout[4]
+ ppgain3*est_x12*plotout[6] + ppgain4*est_x17*plotout[8] );
}
```

3.2.2 Rundata File

```
0       ;initial time
400.    ;final time
0.25    ;maximum stepsize
1.0e-6  ;minimum stepsize
0.001   ;fractional error criterion

200     ;multiple of maximum stepsize for print output
5       ;multiple of maximum stepsize for plot output

20      ;number of plant states
1       ;number of plant inputs
6       ;number of plant outputs
```

```
0       ;number of controller states

8       ;size of user defined plot vector
0       ;size of user common area
0       ;size of gaussian random number vector
        ;vector multiplied by sqrt(hmax) to provide approx. uniform
        ;variance for variable stepsize
318     ;random number seed
272     ;random number seed
190     ;random number seed

0       ;number of user defined integer input parameters
0,0     ;end integer input parameters

22        ;number of user defined floating point input parameters
1,.0978   ;omega12
2,.0009   ;cutoff12
3,.0442   ;cutoff1sq2
4,0       ;refer
5,-.5     ;ppgain1
6,0.      ;intgain1
7,1.523   ;omega22
8,.0152   ;cutoff22
9,.1745   ;cutoff2sq2
10,-.4    ;ppgain2
11,0.     ;intgain2
12,7.6696 ;omega32
13,.0767  ;cutoff32
14,.3916  ;cutoff3sq2
15,-.3    ;ppgain3
16,0.     ;intgain3
17,24.1946 ;omega42
18,.2419   ;cutoff42
19,.6955   ;cutoff4sq2
20,-.2     ;ppgain4
21,0.      ;intgain4
22,100     ;refscale=10 for 0refer, .9 for 1 refer,.476 for 2refer
0,0        ;end floating point input parameters

1,.02   ;
6,.02   ;
11,.02  ;
16,.02  ;
0,0     ;end plant initial conditions

0,0     ;end controller initial conditions
```

3.3 SIMULATION OF CLOSED LOOP DYNAMICS for Nm = 4, No =2, Ns =4.

3.3.1 Dynamic File

```
#include "\ALSIM\ALSIM.H"
#include "MATH.H"
#include "STDIO.H"
#define omega12     fpar[1]
#define cutoff12    fpar[2]
#define cutoff1sq2  fpar[3]
#define refer       fpar[4]
#define ppgain1     fpar[5]
#define intgain1    fpar[6]
#define omega22     fpar[7]
#define cutoff22    fpar[8]
#define cutoff2sq2  fpar[9]
#define ppgain2     fpar[10]
#define intgain2    fpar[11]
#define omega32     fpar[12]
#define cutoff32    fpar[13]
#define cutoff3sq2  fpar[14]
#define omega42     fpar[15]
#define cutoff42    fpar[16]
#define cutoff4sq2  fpar[17]
#define refscale    fpar[18]
/*
**  User state derivative function.
*/

derv(t, x, dxdt)
double t, *x, *dxdt;
{
static float est_x2 = 0.,cal_x2 = 0.,est_x7 =0. ,cal_x7 = 0.;
dxdt[1] = x[2];
dxdt[2] = -omega12*x[1]+u[1];
plotout[1] = fabs(x[2]);
dxdt[3] = x[4];
dxdt[4] = -cutoff12*x[3] -cutoff1sq2*x[4] + cutoff12*plotout[1];
plotout[2] = x[3]-refer;
dxdt[5] = plotout[2];
dxdt[6] = x[7];
dxdt[7] = -omega22*x[6] + u[1];
plotout[3] = fabs(x[7]);
dxdt[8] = x[9];
dxdt[9] = -cutoff22*x[8] -cutoff2sq2*x[9] +cutoff22*plotout[3];
plotout[4] = x[8] - refer;
dxdt[10] = plotout[4];
dxdt[11] = x[12];
dxdt[12] = -omega32*x[11] + u[1];
dxdt[13] = x[14];
dxdt[14] = -omega42*x[13] + u[i];
y[1] = -.0867*x[2] +.1225*x[7] -.0864*x[12] -.0002*x[14];
y[2] = -.1214*x[2] -.0008*x[7] +.1226*x[12] +.0005*x[14];
y[3] = -.0828*x[2] -.1221*x[7] -.0872*x[12] -.0007*x[14];
y[4] = .0072*x[2] +.0037*x[7] +.0025*x[12] +.0019*x[14];
cal_x2 = 33.25*(-.0867*y[1] -.1214*y[2]-.0828*y[3]+.0072*y[4]);
est_x2 = cal_x2  + 5.18*133*.0072*y[4];
cal_x7 = 33.25*(.1225*y[1] -.0008*y[2]-.1221*y[3]+.0037*y[4]);
est_x7 = cal_x7  + 5.18*133*.0037*y[4];
u[1] = refscale*(est_x2*ppgain1*plotout[2] +est_x7*ppgain2*plotout[4]);
plotout[5] = est_x2;
plotout[6] = x[2] - est_x2;
plotout[7] = est_x7;
plotout[8] = x[7] - est_x7;
}
```

3.3.2 Rundata File

```
0        ;initial time
400.     ;final time
0.01     ;maximum stepsize
1.0e-6   ;minimum stepsize
0.001    ;fractional error criterion
200      ;multiple of maximum stepsize for print output
20       ;multiple of maximum stepsize for plot output

14       ;number of plant states
1        ;number of plant inputs
4        ;number of plant outputs
0        ;number of controller states

8        ;size of user defined plot vector
0        ;size of user common area
0        ;size of gaussian random number vector
         ;vector multiplied by sqrt(hmax) to provide approx. uniform
         ;variance for variable stepsize
318      ;random number seed
272      ;random number seed
190      ;random number seed

0        ;number of user defined integer input parameters
0,0      ;end integer input parameters
```

```
18        ;number of user defined floating point input parameters
1,.0978   ;omega12
2,.0009   ;cutoff12
3,.0442   ;cutoff1sq2
4,0       ;refer
5,-.9     ;ppgain1
6,0       ;intgain1
7,1.523   ;omega22
8,.0152   ;cutoff22
9,.1745   ;cutoff2sq2
10,-.5    ;ppgain2
11,0      ;intgain2
12,7.6696 ;omega32
13,.0767  ;cutoff32
14,.3916  ;cutoff3sq2
15,24.1946 ;omega42
16,.2419   ;cutoff42
17,.6955  ;cutoff4sq2
18,100    ;refscale=10 for 0refer, .9 for 1 refer
0,0       ;end floating point input parameters

1,.02   ;
6,.02   ;
11,.02  ;
13,.02  ;
0,0     ;end plant initial conditions

0,0     ;end controller initial conditions
```

3.4 SIMULATION OF CLOSED LOOP DYNAMICS for Nm = 4, No =2, Ns =3.

3.3.1 Dynamic File

```
#include "\ALSIM\ALSIM.H"
#include "MATH.H"
#include "STDIO.H"
#define omega12     fpar[1]
#define cutoff12    fpar[2]
#define cutoff1sq2  fpar[3]
#define refer       fpar[4]
#define ppgain1     fpar[5]
#define intgain1    fpar[6]
#define omega22     fpar[7]
```

```
#define cutoff22   fpar[8]
#define cutoff2sq2 fpar[9]
#define ppgain2    fpar[10]
#define intgain2   fpar[11]
#define omega32    fpar[12]
#define cutoff32   fpar[13]
#define cutoff3sq2 fpar[14]
#define omega42    fpar[15]
#define cutoff42   fpar[16]
#define cutoff4sq2 fpar[17]
#define refscale   fpar[18]
/*
**  User state derivative function.
*/

derv(t, x, dxdt)
double t, *x, *dxdt;
{
static float est_x2 = 0.,cal_x2 = 0.,est_x7 =0. ,cal_x7 = 0.;
dxdt[1] = x[2];
dxdt[2] = -omega12*x[1] + u[1];
plotout[1] = fabs(x[2]);
dxdt[3] = x[4];
dxdt[4] = -cutoff12*x[3] -cutoff1sq2*x[4] + cutoff12*plotout[1];
plotout[2] = x[3]-refer;
dxdt[5] = plotout[2];
dxdt[6] = x[7];
dxdt[7] = -omega22*x[6] + u[1];
plotout[3] = fabs(x[7]);
dxdt[8] = x[9];
dxdt[9] = -cutoff22*x[8] -cutoff2sq2*x[9] +cutoff22*plotout[3];
plotout[4] = x[8] - refer;
dxdt[10] = plotout[4];
dxdt[11] = x[12];
dxdt[12] = -omega32*x[11] + u[1];
dxdt[13] = x[14];
dxdt[14] = -omega42*x[13] + u[1];
y[1] = -.1059*x[2] +.1058*x[7] +.0004*x[12] -.1063*x[14];
y[2] = -.1033*x[2] -.1065*x[7] -.0008*x[12] +.1059*x[14];
y[3] = .0072*x[2] +.0037*x[7] +.0025*x[12] +.0019*x[14];
cal_x2 = 33.25*(-.1059*y[1] -.1033*y[2]+.0072*y[3]);
est_x2 = cal_x2  + 5.18*133*.0072*y[3];
cal_x7 = 33.25*(.1058*y[1] -.1065*y[2]+.0037*y[3]);
est_x7 = cal_x7  + 5.18*133*.0037*y[3];
u[1] = refscale*(est_x2*ppgain1*plotout[2] + est_x7*ppgain2*plotout[4]);
plotout[5] = est_x2;
plotout[6] = x[2] - est_x2;
plotout[7] = est_x7;
plotout[8] = x[7] - est_x7;
}
```

3.3.2 Rundata File

```
0        ;initial time
6000.    ;final time
0.1      ;maximum stepsize
1.0e-6   ;minimum stepsize
0.001    ;fractional error criterion
200      ;multiple of maximum stepsize for print output
200      ;multiple of maximum stepsize for plot output

14       ;number of plant states
1        ;number of plant inputs
3        ;number of plant outputs
0        ;number of controller states

8        ;size of user defined plot vector
0        ;size of user common area
0        ;size of gaussian random number vector
         ;vector multiplied by sqrt(hmax) to provide approx. uniform
         ;variance for variable stepsize
318      ;random number seed
272      ;random number seed
190      ;random number seed

0        ;number of user defined integer input parameters
0,0      ;end integer input parameters

18         ;number of user defined floating point input parameters
1,.0978    ;omega12
2,.0009    ;cutoff12
3,.0442    ;cutoff1sq2
4,0        ;refer
5,-.047    ;ppgain1
6,0        ;intgain1
7,1.523    ;omega22
8,.0152    ;cutoff22
9,.1745    ;cutoff2sq2
10,-.01    ;ppgain2
11,0       ;intgain2
12,7.6696  ;omega32
13,.0767   ;cutoff32
14,.3916   ;cutoff3sq2
```

```
15,24.1946 ;omega42
16,.2419   ;cutoff42
17,.6955   ;cutoff4sq2
18,100     ;refscale=10 for 0refer, .9 for 1 refer
0,0        ;end floating point input parameters

1,.02   ;
6,.02   ;
11,.02  ;
13,.02  ;
0,0     ;end plant initial conditions

0,0     ;end controller initial conditions
```

REFERENCES

1. Kwong, R. H., and T. N Chang " Final Report of Development of Control System Hardware Demonstration for Third Generation Spacecrafts (phase III - Flexible Beam Structure Controller)." Department of Communications, Government of Canada.

2. Chang, T. N. "Nonlinear Tuning Regulator for the Control of Flexible Structures." To appear in the 1993 American Control Conference, San Francisco, CA, June, 1993.

3. Chang, T. N. "Decentralized Robust Control of Interconnected Resonators." Proceedings to the International Conference on Control Applications, Dayton, OH, September, 1992.

4. Davison E. J., and T. N. Chang. "Decentralized Controller Design Using Parameter Optimization Methods." *Control-Theory and Advanced Technology,* Vol. 2, No. 2 (1986): 131-154.

5. Yurkovich, S., F. E. Pacheco., and A. P. Tzes. "On-line Frequency Domain Information for Control of A Flexible-Link Robot with Varying Payload." *IEEE Transcations on Automatic Control,* Vol. 34, No. 12 (1989, December).

6. Omer, M. "Orientation and Stabilization of a Flexible Beam Attached to a Rigid Body: Planar Motion." *IEEE Transcations on Automatic Control,* Vol. 36, No. 8 (1991, August).

7. Tompkins, W. J., and J. G. Webster. *Interfacing Sensors to the IBM PC.* (1988): 282-290.

8. Newland, D. E. *Mechanical Vibration Analysis and Computation.* (1987): 122-176.

9. Strang, Gilbert. *Linear Algebra and Its Applications.* Academic Press (1980) 279-292.

10. Nannapaneni, N. R. *Elements of Engineering Electromagnetics.* Prentic Hall Inc. (1987): 235-243.

11. Katsuhiko, O. *Modern Control Engineering.* Prentice Hall Inc. (1990) 755-794.

12. Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C, The Art of Scientific Computing.* 111-196.

13. O'Neil, P. V. *Advanced Engineering Mathematics.* Wadworth Pubblishing Com pany (1991) 736-746, 1150-1153.

14. Lewis, F. L. *Applied Optimal Control and Estimation.* Prentic Hall (1992) 207-239

15. Roy, R. C. Jr. *Structural Dynamics, An Introduction to Computer Methods.* John Wiley & Sons Inc. (1981) 210-211