# Dynamic Simulation Tool for Seismic Analysis

by

Chun-Yi Wang

B.S., Civil Engineering
National Taiwan University, 1993

Submitted to the Department of Civil and Environmental Engineering
in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Civil and Environmental Engineering

at the

Massachusetts Institute of Technology

May 1998

Signature of Author ........[illegible]........
Department of Civil and Environmental Engineering
May 8, 1998

Certified by ........[illegible]........
Professor Jerome J. Connor
Department of Civil and Environmental Engineering
Thesis Supervisor

Accepted by ........[illegible]........
Joseph M. Sussman
Chairman, Department Committee on Graduate Students

# Dynamic Simulation Tool for Seismic Analysis

by

Chun-Yi Wang

Submitted to the Department of Civil and Environmental Engineering
May, 1998 in Partial Fulfillment of the
Requirements for the Degree of Master of Science in
Civil and Environmental Engineering

## ABSTRACT

Time history analysis of multi-degree-of-freedom structures subjected to general dynamic loading is a convenient and accurate way of representing the dynamic behavior of structures. It is widely used for determining the earthquake resistance in seismic engineering. However, the high computational requirement makes it difficult to implement by simple calculation. Thus, spectrum analysis and other simplified less accurate procedures are used instead of response history analysis. With the great improvement in personal computing capability in recent years, response history analysis with personal computers is now feasible. Furthermore, more complex and realistic problems such as non-proportional damping and highly damped structures can be handled. The thesis presents the mathematical formulation in state space that can handle a shear beam type of structure with arbitrary properties. The implementation of the mathematical formulation in a program named Motion Lab is described. This program provides a dynamic simulation environment for rapid prototyping and assessment of structural response. Key features are the graphical user interface and visual display tools for time history response. This program is intended to be used primarily as a learning tool for structural dynamics. It is also useful for preliminary structural design.

Thesis Supervisor: Dr. Jerome J. Connor
Title: Professor of Civil and Environmental Engineering

# Acknowledgments

I would like to thank Professor Connor for giving me this chance to implement the simulation environment. His guidance and encouragement inspired me throughout this thesis. I would also like to thank Joy for her great support, and my friends in MIT, who gave me help in both academics and daily life.

Of course, I am grateful to my parents for their love and support.

# Table of Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 Motivation

Matrix structural dynamic analysis is a complex procedure that involves many matrix operations and numerical computations. The extent of the complexity makes it difficult or even impossible to calculate by hand. Thus, the actual problem is either simplified to one that can be solved by simple computation or people have to resort to dynamic analysis program to get an accurate result. In the first case, the problem simplified may behaves far from the original system, thus it may not meet the accuracy requirement. On the other hand, spending hours to setup a dynamic analysis program such as ADINA or SAP is not efficient for people who mainly want to experiment with the structure and therefore need a simple setup procedure. For example, in the preliminary design phase, one needs a tool with simplified interface and good numerical accuracy that can be set up within minutes and return the result immediately. One needs to play around with several scenarios and have the feeling of the actual behavior of the structure within a limited amount of time.

Thus, a structural analysis tool that is simple to use and handles intensive computations is needed in two major areas. The first area is the preliminary earthquake design of structures for seismic excitation, the second area is mechanical vibration and structural dynamics education.

To understand how the tool may improve the preliminary design process, the traditional design procedure for earthquakes is examined. The procedure is divided into several steps. First, the structure is discretized into a lumped parameter system. Then the building code is introduced. From the properties of the structure and the seismicity of the region, the equivalent static lateral force each floor will take during earthquake is determined.

Most codes also permit dynamic analysis procedures of both response spectrum analysis (RSA) and response history analysis (RHA). The response history analysis calculates the time history of the structural response subjected to a given ground acceleration $\ddot{u}_g(t)$. The response spectrum analysis computes the peak response of the structure during an earthquake directly from the earthquake response spectrum.

The earthquake design procedures presented above can be improved in several aspects by applying the tool. First, the building earthquake design code can be verified or even substituted by the response calculated by the tool. The reason is that the code is only an approximation and a safety limit to normal structures; it does not take the property of the entire structure or the equivalent discretized structure into account. For example, only the fundamental frequency and maximum response properties from the design spectrum chart are considered to represent a structure and an earthquake impact. The same problem happens to the response spectrum analysis procedure. The structure peak response varies with the property of the structure, thus the effect of the summation of peak responses of several modes is not the real structural response because of the difference in phase lag between modes. With the complete response analysis procedure built into a standard computing process, the preliminary design procedure can have a great support.

Other improvements can be made on the design procedure with the tool. First, the response of general structure with non-proportional damping can be evaluated. The state-space formulation works with complex modes and frequencies, and thus can deal with arbitrary damping In addition to accurately predicting the response of a highly damped structure, the influence of damping on the mode shapes and frequencies can be examined.

Secondly, several earthquake spectrums can be imposed on the structure to get a complete portrait of the response characteristics of the structure. For example, responses of the structure to different earthquakes with different frequency contents can be used to establish an improved estimate of the required rigidity.

The other application area of great potential is education. Textbooks on vibration analysis and structural dynamics give examples based on proportional damping, because real quantities are easier to comprehend and compute. However, non-proportional damping is generally used in buildings. With the help of this analysis tool, a more realistic treatment of the subject can be presented. Realistic studies of highly damped structures with non-proportional damping can be carried out.

The main functional requirements and corresponding design parameters for the simulation environment are listed in Figure 1.1.

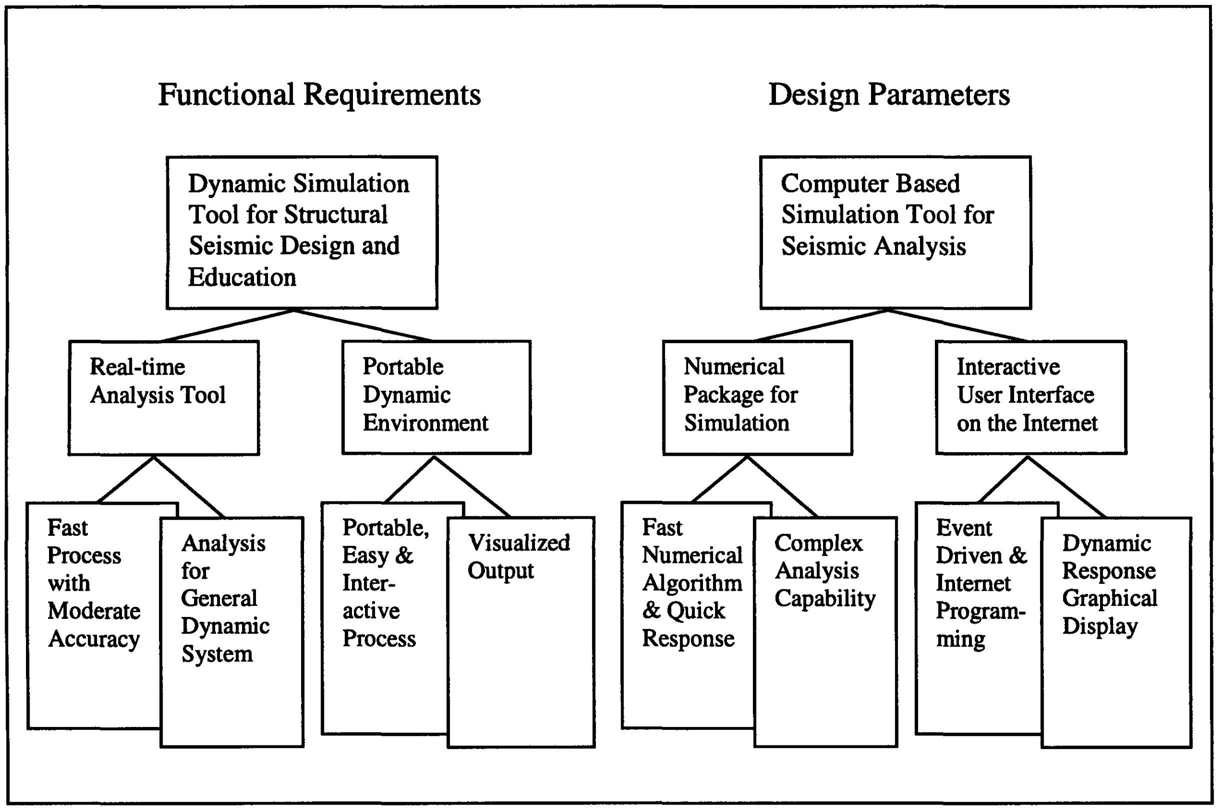

Functional Requirements
Design Parameters
Dynamic Simulation Tool for Structural Seismic Design and Education
Computer Based Simulation Tool for Seismic Analysis
Real-time Analysis Tool
Portable Dynamic Environment
Numerical Package for Simulation
Interactive User Interface on the Internet
Fast Process with Moderate Accuracy
Analysis for General Dynamic System
Portable, Easy & Inter-active Process
Visualized Output
Fast Numerical Algorithm & Quick Response
Complex Analysis Capability
Event Driven & Internet Program-ming
Dynamic Response Graphical Display

**Figure 1.1 Functional Requirements and Corresponding Design Parameters of the Dynamic Simulation Tool**

## 1.2 Review of Previous Work

Previous works related to the implementation of the simulation environment fall into two categories. One category is research on general formulation and numerical computation methods in structural dynamics. The other category is computer-aided design and education system for mechanics.

Research efforts on the application of state-space formulation to structural analysis are described in [1-3]. The capability of the state-space formulation in the dynamic analysis of structures has been proved. Also, the basic transformations between real and complex system have been derived.

On the other hand, two different types of software are developed for computer-aided design and education. The first are the tools for accurate structural analysis. Software systems such as ADINA, ANSYS and SAP are out of the scope of the thesis and will not be discussed. The other types of software for preliminary design and education are still focused on statics problem. Educational programs for the analysis of simple beams and trusses are developed in [4][5]. They are all portable on the internet and have a user friendly interface.

## 1.3 Organization of the Thesis

Comparing the functional requirements to the previous work, a more general structural dynamic simulation environment with easy to use interface and short responding time is needed. The following chapters discuss the underlying theories and the implementation of the proposed environment.

In the next chapter (Chapter 2), the fundamental state-space formulation is stated, and then an uncoupled version is derived by solving the associated eigenvalue problem.

Applying the uncoupled complex equations, responses for free and forced vibrations are generated. After that, modal properties and mode shapes of complex modes are discussed.

Chapter 3 describes the numerical algorithms used in dynamics analysis. A standard numerical procedure for obtaining the response of a general dynamic system is proposed. Then, the numerical algorithm for generating complex eigenvalues and eigenvectors of a general matrix is described. Both frequency domain analysis and direct integration method are discussed for complex modal analysis.

Chapter 4 focuses on the architecture and implementation of the environment. First, the design of the objects is discussed. Then, the strategy for optimizing the computations for dynamic simulation is proposed. Finally, the implementation and the operation of the system is described.

Chapter 5 discusses the evaluation and testing of the system. Several scenarios are provided. Interactive process between the program and user is presented. The eigensystems and response histories are also verified by several MATLAB programs.

Chapter 6 summarizes the research work and proposes some future directions.

# Chapter 2

# Fundamental Formulation for General Dynamic System

## 2.1 Introduction to State-space Formulation

The governing equations of motion for a multi-degree-of-freedom mechanical system are given by

$$\mathbf{M}\,\ddot{\mathbf{u}}(t) + \mathbf{C}\,\dot{\mathbf{u}}(t) + \mathbf{K}\,\mathbf{u}(t) = \mathbf{P}(t) \tag{2.1}$$

where **M** is the mass matrix, **C** is the damping matrix, and **K** is the stiffness matrix. Vectors $\mathbf{u}(t), \dot{\mathbf{u}}(t)$, and $\ddot{\mathbf{u}}(t)$ are the displacement, velocity, and acceleration vectors of the degrees of freedom in the system. Vector **P** (t) is the external forcing function as a function of time t. If **P** (t) is zero at all time t and $\mathbf{u}(0), \dot{\mathbf{u}}(0)$ are given as initial conditions, (2.1) is often referred to as the free vibration problem. Otherwise, if **P** (t) is not zero, the problem is categorized as the forced vibration problem.

Equation 2.1 is recognized as a set of coupled second order differential equations. One solution strategy is based on applying a linear coordinate transformations to uncouple the equations. For the undamped (**C** = **0**) and proportional damping[1] cases, the uncoupled equations can be obtained easily by first solving first order eigenvalue problem for the

[1] If $\phi_m^T, \phi_n^T$ are the *m*th and *n*th mode-shape vector of the undamped system, and if for the damping matrix **C**, the orthogonality holds, i.e., $\phi_m^T\,\mathbf{C}\,\phi_n = 0$ for all $m \neq n$, then **C** is called the proportional damping.

system, and using the mutually orthogonal eigenvectors as the basis functions for the coordinate transformation. Then, use the solution procedure for second order differential equation to solve each of the uncoupled equations separately.

However, when dealing with dynamic systems with non-proportional damping, orthogonal eigenvectors of the real system does not exist. The procedure for uncoupling the system with proportional damping is not valid. Thus, to obtain an uncoupled set of equations for the system, one has transform the system to state-space system to reduce the problem to a first order eigenvalue problem.

The transformation from the real system as shown in Equation 2.1 to the state-space formulation is performed by considering the velocity vector another unknown variable of the system. The vector composed of both displacement and velocity vectors of an n-degree-of-freedom system defines the state of the system thus named the state vector. The state vector $\mathbf{X}$ (t) is arranged in a 2n-dimensional vector of the form

$$\mathbf{X}(t) = \begin{bmatrix} \mathbf{u(t)} \\ \dot{\mathbf{u}}\mathbf{(t)} \end{bmatrix} \tag{2.2}$$

Similarly, the 2n-dimensional excitation vector F (t) is in the form

$$\mathbf{F}(t) = \begin{bmatrix} 0 \\ \mathbf{P(t)} \end{bmatrix} \tag{2.3}$$

Then, the equation of motion of an n-degree-of-freedom linear system can be written in the state-space form

$$\dot{\mathbf{X}}\mathbf{(t)} = \mathbf{A}\ \mathbf{X(t)} + \mathbf{B}\ \mathbf{F}(t) \tag{2.4}$$

Where matrix **A** and **B** contain the parameters of the system and have the following form

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{M}^{-1}\mathbf{K} & -\mathbf{M}^{-1}\mathbf{C} \end{bmatrix} \tag{2.5}$$

and

$$\mathbf{B} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}^{-1} \end{bmatrix} \tag{2.6}$$

It is clear that the state-space formulation in Equation 2.4 is similar in structure to the first order differential equation system. Also, first order eigenvalue solution procedure can be applied to the state-space formulation, except the system has been turned into a complex system.

## 2.2 Eigenproblem of a Nonsymmetric Matrix

As presented by Equation 2.5, the matrix **A** is a 2n by 2n, nonsymmetric matrix. To solve the response of Equation 2.4, it is necessary to consider first the following eigenvalue problem

$$\mathbf{A}\,\mathbf{V}_i = \lambda_i \mathbf{V}_i \qquad i = 1,2,\ldots,2n \tag{2.7}$$

where $\lambda_i$ and V$i$ are the eigenvalues and eigenvectors of **A**. Because **A** is not symmetric, the eigenvalues and eigenvectors are in general both complex, and the eigenvectors are not mutually orthogonal. Nevertheless, they do satisfy some orthogonality relations. The eigenvalue problem,

$$\mathbf{A}^T\ \mathbf{W}_j = \lambda_j \mathbf{W}_j \qquad j = 1,2,...,2\text{n} \tag{2.8}$$

is called adjoint eigenvalue problem. **V** is often referred to as the right eigenvector and **W** is called the left eigenvector. It has been proved [2] that the two sets of eigenvectors satisfy the biorthogonality property, that is

$$\mathbf{W}_j^T\ \mathbf{V}_i = 0 \qquad \lambda_i \neq \lambda_j \qquad i, j = 1,2,...,2\text{n} \tag{2.9a}$$

$$\mathbf{W}_j^T\ \mathbf{V}_i = \alpha_i \qquad \lambda_i = \lambda_j \qquad i, j = 1,2,...,2\text{n} \tag{2.9b}$$

where $\alpha_i$ is the product of the right and left eigenvectors that possess the same eigenvalue $\lambda_i$. The length of $\mathbf{W}_j$ and $\mathbf{V}_i$ can be normalized to unity for obtaining 1 for all $\alpha_i$. Multiplying both side of Equation 2.7 by $\mathbf{W}_j^T$ and substituting in Equation 2.9, the equation becomes

$$\mathbf{W}_j^T\mathbf{A}\ \mathbf{V}_i = \lambda_i\, \alpha_i\, \delta_{ij} \qquad i, j = 1,2,...,2\text{n} \tag{2.10}$$

where $\delta$ is the Kronecker delta. Equation 2.10 shows the eigenvectors $W_j$ and $V_i$ are biorthogonal with respect to the matrix A as well. Using the relationship shown in Equation 2.10, the uncoupled set of equations is obtained by first assuming the solution of Equation 2.4 as

$$\mathbf{X} = \sum_{i=1}^{2\text{n}} \mathbf{V}_i\, q_i(\text{t}) \tag{2.11}$$

or in matrix notation

$$\mathbf{X} = \mathbf{V}\,\mathbf{q} \tag{2.12}$$

with the initial conditions

$$\mathbf{X}(0) = \begin{bmatrix} \mathbf{u}(0) \\ \dot{\mathbf{u}}(0) \end{bmatrix} \tag{2.13}$$

where **V** is the right eigenvector serving as a transformation vector from the generalized coordinate **q** to the geometric coordinate **X,** and **X**(0) is the initial state vector composed of initial displacement vector **u**(0) and initial velocity vector **u**(0). Substituting Equation 2.12 in Equation 2.4 and multiplying both sides by $\mathbf{W}^T$, the following equation is obtained

$$\mathbf{W}^T\,\mathbf{V}\,\dot{\mathbf{q}} = \mathbf{W}^T\,\mathbf{A}\,\mathbf{V}\,\mathbf{q} + \mathbf{W}^T\,\mathbf{B}\;\mathbf{F} \tag{2.14}$$

Inserting Equations 2.9 and 2.10 into Equation 2.14 leads to

$$\boldsymbol{\alpha}\;\dot{\mathbf{q}} = \boldsymbol{\alpha}\;\boldsymbol{\Lambda}\;\;\mathbf{q} + \mathbf{W}^T\,\mathbf{B}\;\mathbf{F} \tag{2.15}$$

where

$$\boldsymbol{\alpha}\;= diag\;\begin{bmatrix}\alpha_1 & \alpha_2 & \cdots & \alpha_{2n}\end{bmatrix} \tag{2.16}$$

$$\mathbf{q}^T\;=\;\begin{bmatrix}q_1 & q_2 & \cdots & q_{2n}\end{bmatrix} \tag{2.17}$$

and

$$\boldsymbol{\Lambda}\;= diag\;\begin{bmatrix}\lambda_1 & \lambda_2 & \cdots & \lambda_{2n}\end{bmatrix} \tag{2.18}$$

As shown in Equations 2.15 to 2.18, both $\alpha$ and $\Lambda$ are complex diagonal matrixes, and $\mathbf{W}^T\mathbf{B}\,\mathbf{F}$ the product of is a vector, thus Equation 2.15 is decomposed as 2n independent first order differential equations, with the solution vector **q** in the complex plane.

## 2.3 State-space Solution for Free Vibration Problem

The free vibration problem follows from Equation 2.15 by setting **F** = 0.

$$\dot{\mathbf{q}} = \boldsymbol{\Lambda}\;\;\mathbf{q} \tag{2.19}$$

with the initial conditions shown in Equation 2.13. The assumption of **q** in Equation 2.19 is in the form of Equation 2.20

$$q_i(t) = c_i\, e^{\lambda_i t} \tag{2.20}$$

To solve for the coefficients in Equation 2.20, the transformation of the geometric initial condition **X**(0) to the generalized coordinate **q** has to be performed. The transformation is done by inserting Equations 2.19 and 2.20 into 2.11 at t = 0

$$\mathbf{X}(0) = \sum_{i=1}^{2n} \mathbf{V}_i\, q_i(0) \;= \sum_{i=1}^{2n} \mathbf{V}_i\, c_i \tag{2.21}$$

Multiplying $\mathbf{W}^T$ on both sides of Equation 2.21 and applying Equation 2.9, the following relationship is obtained

$$\mathbf{W}_i^T \mathbf{X}(0) = \mathbf{W}_i^T \mathbf{V}_i\, c_i = \alpha_i c_i \tag{2.22}$$

Thus, the coefficients $c$ can be obtained from dividing both sides of Equation 2.22 by $\alpha_i$

$$c_i = \frac{\mathbf{W}_i^T \mathbf{X}(0)}{\alpha_i} \tag{2.23}$$

Note that for **X** to be real throughout the time domain, all $\mathbf{V}_i$, $q_i$ and $\lambda_i$ have to appear in complex conjugate pairs. Equation 2.21 is rewritten to incorporate the complex conjugate pairs as follows

$$\sum_{i=1}^{n} \mathbf{V}_i\, c_i + \tilde{\mathbf{V}}_i\, \tilde{c}_i = \mathbf{X}(0) \tag{2.24}$$

where $\tilde{\mathbf{V}}_i$ and $\tilde{c}_i$ are the complex conjugate vectors of $\mathbf{V}_i$, $c_i$ separately. Letting

$$\mathbf{V}_i = \mathbf{V}_{iR} + i\,\mathbf{V}_{iI} \tag{2.25a}$$

$$c_i = c_{iR} + i\, c_{iI} \tag{2.25b}$$

and inserting Equations 2.25 into Equation 2.24, leads to the following

$$\begin{aligned}\sum_{i=1}^{n} \mathbf{V}_i\, c_i + \tilde{\mathbf{V}}_i\, \tilde{c}_i &= \sum_{i=1}^{n} (\mathbf{V}_{iR} + i\mathbf{V}_{iI})(c_{iR} + i c_{iI}) + (\mathbf{V}_{iR} - i\mathbf{V}_{iI})(c_{iR} - i c_{iI}) \\ &= 2\sum_{i=1}^{n} (\mathbf{V}_{iR} c_{iR} - \mathbf{V}_{iI} c_{iI}) = \mathbf{X}(0)\end{aligned} \tag{2.26}$$

Equation 2.26 can be rewritten in matrix form as follows

$$2\cdot \begin{bmatrix} \mathbf{V}_{1R} & -\mathbf{V}_{1I} & \mathbf{V}_{2R} & -\mathbf{V}_{2I} & \cdots & \mathbf{V}_{nR} & -\mathbf{V}_{nI} \end{bmatrix}_{2n\times 2n} \begin{bmatrix} c_{1R} \\ c_{1I} \\ c_{2R} \\ c_{2I} \\ \vdots \\ c_{nR} \\ c_{nI} \end{bmatrix}_{2n\times 1} = \mathbf{X}(0)_{2n\times 1} \tag{2.27}$$

Equation 2.27 is the alternative formulation for solving the coefficient $c$ when the left eigenvector $\mathbf{W}_i$ in Equation 2.23 is unknown. Substituting $q$ into Equation 2.11 to obtain the state vector **X**. The state vector is written as follows

$$\mathbf{X}(t) = \sum_{i=1}^{n} \mathbf{V}_i\, q_i = \sum_{i=1}^{n} \mathbf{V}_i\, c_i\, e^{\lambda_i t} + \tilde{\mathbf{V}}_i\, \tilde{c}_i\, e^{\tilde{\lambda}_i t} \tag{2.28}$$

The imaginary part of Equation 2.28 cancelled out and leaves the real coefficients, which corresponds to the real displacements and velocities in **X**.

$$\mathbf{X}(t) = \sum_{i=1}^{n} 2\, e^{\lambda_{iR} t} \cdot [\, (\mathbf{V}_{iR}\, c_{iR} - \mathbf{V}_{iI}\, c_{iI}) \cdot \cos\lambda_{iI}\, t - (\mathbf{V}_{iR}\, c_{iI} + \mathbf{V}_{iI}\, c_{iR}) \cdot \sin\lambda_{iI}\, t\,] \tag{2.29}$$

Notice that the displacement response of the system is stored in the upper n elements, and the velocity response is stored in the lower n elements in the **X** vector.

## 2.4 State-space Solution for Forced Vibration Problem

Two procedures are often applied to evaluate the dynamic response of a multi-degree-of-freedom structure subjected to arbitrary forcing. The first is the mode superposition method, which uncouples the equations by making transformation to the generalized coordinate. The response from each mode is summed up to generate the response of the entire system. The second procedure is the direct integration method, which converts the entire system of equations into a difference form and evaluates the response of the system using a numerical step-by-step integration procedure.

The direct integration method is sometimes more suitable for calculating the response of a general non-proportional damping system, because the coupled equations need not to be uncoupled in order to carry out modal analysis. In addition, it is easier to apply direct integration in a non-linear system with its parameters varying from time to time. However, for linear systems with moderate degree of freedoms, modal analysis is still applicable. It is not necessary to calculate all the modal properties and response histories in modal analysis. Instead, several significant modes are summed up to represent the response of a structure. Thus, computational cost is minimized even though more effort is made to uncouple the equations by solving the eigenvalue problem.

Since our primary interest is in implementing the system as an educational tool, the effort is focused here on the mode superposition method. Modal analysis needs to be performed in order to obtain the insight of the dynamic modal characteristics. Thus, the effort of the following sections will be focused on mode superposition method.

### 2.4.1 Formulation of Forced Vibration Problem in State-space

The first step of the mode superposition method for solving a coupled multi-degree-of-freedom system is to uncouple the equations, thus the same formulation in Section 2.2 can be applied. Rewrite Equation 2.14 as follows

$$\alpha_i \dot{q}_i = \alpha_i \ \lambda_i \ q_i + \mathbf{W}_i^T \mathbf{B} \ \mathbf{F} \qquad i = 1,2,\cdots,2\mathrm{n} \tag{2.30}$$

It can be proved that $\alpha_i$ appear in complex conjugate pairs by simply comparing the vectors $\mathbf{W}_j^T \mathbf{V}_j$ and $\tilde{\mathbf{W}}_j^T \tilde{\mathbf{V}}_j$. Notice that the modal forces $\mathbf{W}_j^T \mathbf{B} \mathbf{F} / \alpha_j$ appear in complex conjugate pairs as well. Separate the conjugates and rewrite Equation 2.30 as follows

$$\dot{q}_i = \lambda_i \ q_i + f_i \qquad i = 1,2,\cdots,\mathrm{n} \tag{2.31a}$$

$$\dot{\tilde{q}}_i = \tilde{\lambda}_i \ \tilde{q}_i + \tilde{f}_i \qquad i = 1,2,\cdots,\mathrm{n} \tag{2.31b}$$

where

$$f_i = \mathbf{W}_i^T \mathbf{B} \ \mathbf{F} \ / \ \alpha_i \qquad i = 1,2,\cdots,\mathrm{n} \tag{2.32}$$

and $\tilde{f}_i$ is the complex conjugate of $f_i$.

It is obvious that only either Equation 2.31a or Equation 2.31b has to be considered instead of both, because the solution of $q$ comes in complex pairs. However, this does not make the computation work easier. By separating the real parts and imaginary parts, the equation number is doubled as before.

Three alternative methods can be applied to solve for Equation 2.31, i.e., time domain analysis, frequency domain analysis, and direct integration method. Time domain analysis is performed by first evaluating the impulse response of the system. The forcing function is then treated as a series of impulse. Summing up the impulse responses at different time by different forcing magnitude comes to the response history of the system. Convolution integral is applied for the summing up process. On the other hand, direct integration method applies numerical integration process to solve for the response of the system.

Time domain method is not suitable for the area of interest because it needs convolution integration, which is computationally intensive. Alternative way of carrying out convolution integral is to treat it as an equivalent frequency method to cut down the computational cost [6]. Thus, it is more suitable to apply frequency domain method instead of time domain method. Direct integration is also a suitable method for response history analysis. The method is a numerical algorithm based on finite difference formulation. Thus, it belongs to the category of numerical algorithms and will be discussed in Chapter 3.

Frequency domain analysis is different from the time domain method in that it sums up the effect of the forcing function in the frequency domain. That is, the forcing function is represented as a summation of harmonic functions of different frequencies. The summation is done by first taking the forcing term into Fourier transformation. The transformation of the forcing term *f* in Equation 2.31 is represented as follows

$$F_j(i\omega) = \int_{-\infty}^{\infty} f_j(t)\, e^{-i\omega t} dt \qquad j = 1,2,\cdots,\mathrm{n} \tag{2.33}$$

Then, all forcing terms in the frequency domain are combined by the inverse Fourier transformation to obtain the total response as follows

$$q_j(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} H_j(i\omega) F_j(i\omega)\, e^{i\omega t} d\omega \qquad j = 1,2,\cdots,\mathrm{n} \tag{2.34}$$

where $H_j(i\omega)$ is the complex transfer function and will be derived later in the next section.

### 2.4.2 Complex Transfer Function in the Frequency Domain

The complex transfer function in the frequency domain transfers the periodic force into the response of the system. Thus, to obtain the transfer function for the state-space system presented in Equation 2.31, the response to general periodic functions has to be solved. Consider a periodic force exerts on a system similar to Equation 2.31 as follows

$$\dot{x} = a\, x + p_0\, e^{i\bar{\omega} t} \tag{2.35}$$

Assume the solution for *x* has the following form

$$x(t) = x_0\, e^{i\bar{\omega} t} \tag{2.36}$$

Substituting *x*'s in Equation 2.35 by *x*(*t*) in Equation 2.36 obtains the following

$$i\bar{\omega}\, x_0\, e^{i\bar{\omega} t} = a\, x_0\, e^{i\bar{\omega} t} + p_0\, e^{i\bar{\omega} t} \tag{2.37}$$

$x_0$ is obtained from Equation 2.37 as follows

$$x_0 = \frac{p_0}{i\overline{\omega} - a} \tag{2.38}$$

The transfer function $H(i\omega)$ is obtained from dividing the response by the excitation force as follows

$$H(i\overline{\omega}) = x_0\, e^{i\overline{\omega}t} / p_0\, e^{i\overline{\omega}t} = \frac{1}{i\overline{\omega} - a} \tag{2.39}$$

Applying the result to the state-space system in Equation 2.31, $H_j(i\omega)$ is obtained as follows

$$H_j(i\omega) = \frac{1}{i\omega - \lambda_j} \qquad j = 1,2,\cdots,\mathrm{n} \tag{2.40}$$

## 2.4.3 Forced Vibration Response in State-space Formulation

After the frequency domain transfer function is obtained by Equation 2.40, the inverse Fourier transformation defined by Equation 2.34 is performed to combine the effect contributed by every harmonic forcing terms. The next step is to transform the complex state-space vectors back to the geometric coordinates.

The transform is similar to what has been done in the free vibration response analysis in Section 2.3. From the free vibration analysis, $\mathbf{V}_i$ and $q_i$ have to appear in conjugate pairs in order to keep the response in the real axis. Rewrite Equation 2.11 to accommodate conjugate pairs as follows

$$\mathbf{X}(t) = \sum_{i=1}^{\mathrm{n}} \mathbf{V}_i\, q_i(t) + \tilde{\mathbf{V}}_i\, \tilde{q}_i(t) \tag{2.41}$$

Notice that the eigenvectors in the forced vibration problem are the same used in the free vibration problem because the eigenproblem is the same. Thus, apply Equation 2.25a directly in Equation 2.41 and separate the real and imaginary parts for $q_i$ as follows

$$q_i(t) = q_{iR}(t) + i\, q_{iI}(t) \tag{2.42}$$

Inserting in Equation 2.42 into Equation 2.41 obtains the following

$$\mathbf{X}(t) = \sum_{i=1}^{n} 2 \cdot \left[ \mathbf{V}_{iR}\, q_{iR}(t) - \mathbf{V}_{iI}\, q_{iI}(t) \right] \tag{2.43}$$

The state vector $\mathbf{X}(t)$ in Equation 2.41 contains all complex modes of the system, thus it is an exact solution for the state-space system. However, it is not efficient to incorporate all modes in modal analysis. Only the combination of several basic modes is enough for moderate accuracy. The mode superposition theory has been developed very well for systems with proportional damping [7]. However, very few discussions have been focused on the dynamic systems with non-proportional damping. Thus the next section will emphasis on the modal properties and the mode superposition of the state-space systems.

## 2.5 Modal properties of Dynamic Systems with Coupled Damping

Consider the eigenvalue problem of state-space formulation in Equation 2.4 with $\mathbf{F} = 0$. The eigenvalues can be obtained by letting

$$\mathbf{X}(t) = \mathbf{V} e^{-\lambda t} \tag{2.44}$$

and taking the determinant as follows

$$\det(\mathbf{A}+\lambda\mathbf{I}_{2n\times 2n}) = \begin{vmatrix} \lambda\mathbf{I}_{n\times n} & \mathbf{I}_{n\times n} \\ -\mathbf{M}^{-1}\mathbf{K} & -\mathbf{M}^{-1}\mathbf{C}+\lambda\mathbf{I}_{n\times n} \end{vmatrix} = 0 \tag{2.45}$$

Multiplying out the determinant obtains the following

$$\lambda^2\mathbf{I}-\lambda\mathbf{M}^{-1}\mathbf{C}+\mathbf{M}^{-1}\mathbf{K}=0 \tag{2.46}$$

Define $\lambda = -i\omega$ as a complex frequency term to incorporate the damping effect and rewrite Equation 2.46 as follows

$$-\omega^2\mathbf{M}+i\omega\mathbf{C}+\mathbf{K}=0 \tag{2.47}$$

The eigenvalues can be obtained by solving Equation 2.46 as a quadratic eigenvalue problem and assuming $\mathbf{z}_j$ is the j-th eigenvector of it. Notice that $\mathbf{z}_j$ is not the eigenvector of the state-space system described by Equation 2.7. In fact, **z** and **z*** are not mutually orthogonal, thus the matrices in Equation 2.47 will not be uncoupled by **z**. However, **z** is still valuable in the discussion of complex modal properties.

To obtain further information of complex modal properties, first multiply $\mathbf{z}_j$ and its Hermitian $\mathbf{z}_j^*$ on the left hand side of Equation 2.47

$$\mathbf{z}_j^*\cdot(-\omega^2\mathbf{M}+i\omega\mathbf{C}+\mathbf{K})\cdot\mathbf{z}_j=0 \tag{2.48}$$

Notice that the procedure is not the procedure for diagonalizing the system as provided in Equation 2.14, and **z** vectors are not mutually orthogonal. The procedure in Equation 2.48 is merely for determining the modal properties of the system. Multiplying out $\mathbf{z}_j$ and $\mathbf{z}_j^*$ into the bracket obtains the following

$$-\omega^2 \mu_j + i\omega\eta_j + \kappa_j = 0 \tag{2.49}$$

where

$$\mu_j = \mathbf{z}_j^* \mathbf{M} \mathbf{z}_j > 0 \tag{2.50a}$$

$$\eta_j = \mathbf{z}_j^* \mathbf{C} \mathbf{z}_j \geq 0 \tag{2.50b}$$

$$\kappa_j = \mathbf{z}_j^* \mathbf{K} \mathbf{z}_j \geq 0 \tag{2.50c}$$

The above terms are all real and non-negative because **M** is positive definite and **C, K** are positive semi-definite. Define the Rayleigh quotients *b* and *r* as follows

$$b = \frac{1}{2}\frac{\eta_j}{\mu_j} \geq 0 \tag{2.51a}$$

$$r^2 = \frac{\kappa_j}{\mu_j} \geq 0 \tag{2.51b}$$

Substituting *b* and *r* in Equation 2.46 obtains the following

$$-\omega^2 + 2ib\omega + r^2 = 0 \tag{2.52}$$

The solution of Equation 2.52 is

$$\omega = ib \pm \sqrt{r^2 - b^2} \tag{2.53}$$

Substitute Equation 2.52 by letting $b = \xi_j \omega_j$ and $r = \omega_j$,

$$\omega = i\xi_j\omega_j \pm \omega_j\sqrt{1-\xi_j^2} \tag{2.54}$$

where $\xi_j$ and $\omega_j$ are defined as the modal damping coefficient and modal frequency of the *j*-th mode. Comparing Equation 2.51 to the modal formulation of proportional damping, $\xi_j$ and $\omega_j$ are identical to the modal damping and frequency defined in proportional damping structures. On the other hand, the definition is still valid for non-proportional damping case. Recall that the eigenvalue λ = -*i*ω; substituting λ in Equation 2.54 obtains the following

$$\lambda = \xi_j \omega_j \pm i\omega_j \sqrt{1-\xi_j^2} \tag{2.55}$$

The real term of the right hand side in Equation 2.55 represents an exponentially decay term of the state vector **X** in Equation 2.44, which is controlled by the modal damping term $\xi_j \omega_j$; On the other hand, the imaginary term stands for the oscillation term in **X**, which is controlled by the damped modal frequency $\omega_j \sqrt{1-\xi_j^2}$. Figure 2.1 shows this relationship in the complex plane. For the case $\xi_j < 0$, the system is unstable because the response is always amplified by the positive exponential term. When $\xi_j = 0$, the system is undamped because no exponential decay term appears. When $0 < \xi_j < 1$, the system is underdamped because both exponential decay and oscillation term appear. When $\xi_j \geq 1$, the system is overdamped, and only exponential decay term left.

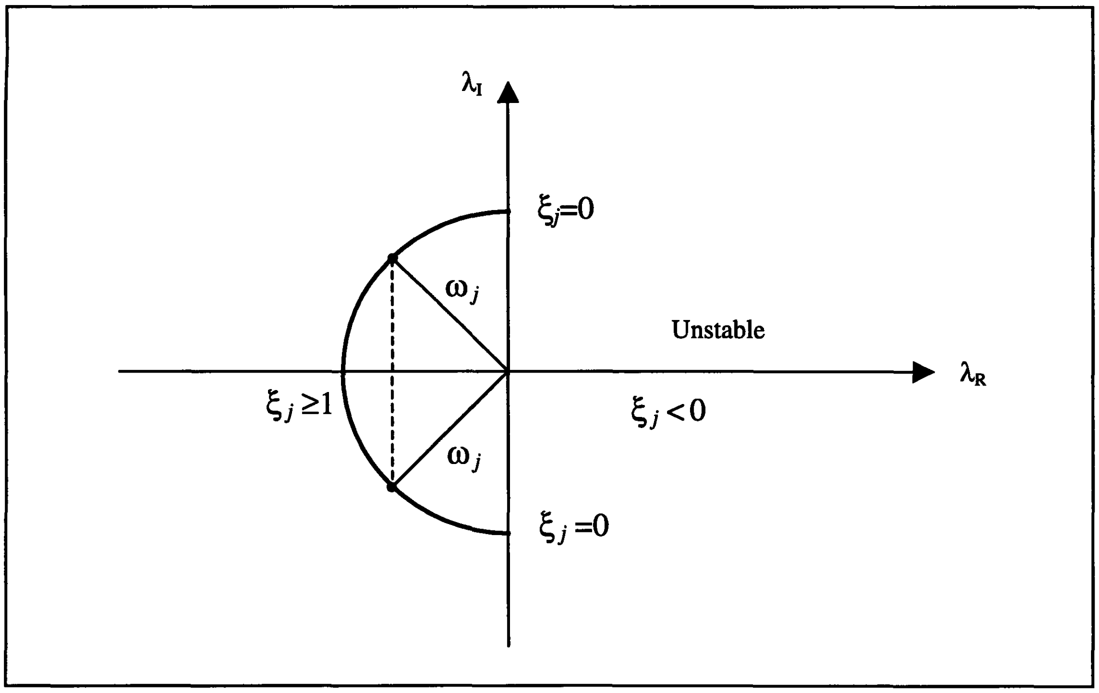


**Figure 2.1 Plot of λ in complex plane**

Mode superposition in state-space can be further simplified with the discussion of λ. From Equation 2.55, λ appears in complex conjugate pair, and each pair share the same modal damping and modal frequency. Thus, the modes with conjugate pair of eigenvalues have the same properties of dynamic response in the real plane. The only difference they have is the difference in sign of the imaginary oscillation term. The eigenvectors associated with the conjugate pair of eigenvalues have to form conjugate pair in order to produce real state vectors in Equation 2.44 for any time t. Thus, the complex oscillation term cancelled out each other if the effect of the two conjugate responses are combined together. From the above discussion, the dynamic properties of the two conjugate terms are the same, thus the combined effect of response can be treated as a modal response for the system.

Modal analysis is made on the forced vibration response by introducing this concept in Section 2.4.3. From Equation 2.44, the complex conjugate pairs of $\boldsymbol{V}$ and $q$ are identical to the conjugate eigenvalues and eigenvectors mentioned previously. They share the same dynamic properties and can be combined together to represent one modal contribution for the system. Thus, the approximate response can be represented by taking the partial sum of Equation 2.43 as follows

$$\mathbf{X}(t) = \sum_{i=1}^{m} 2\cdot\left[\mathbf{V}_{iR}\, q_{iR}(t) - \mathbf{V}_{iI}\, q_{iI}(t)\right] \quad , 1 \le m \le \mathrm{n} \tag{2.56}$$

where $\mathbf{X}$ is the sum of responses from lower $m$ modes if the eigenvalues and eigenvectors were presorted with the modal frequency ω at increasing order. In addition, the same procedure can be applied to the analysis of free vibration response by changing the upper limit of the sum of Equation 2.29 to *m*.

The procedure above is not suitable for all types of analysis. Generally speaking, if the modal frequencies of the modes included in the summation cover the frequency spectrum of the excitation, and the need for response accuracy is not very high, it is acceptable to have only a combination of first several modes. For the example of earthquake, the frequency content is not very high (usually lower than 30 Hz), thus usually five modes are enough to represent the response to an earthquake.

If only the responses of several modes are desired, the computation effort can be further reduced in uncoupling the equations. If first *m* out of n modes are to be computed, then *m* right and left eigenvectors are needed to decompose the system. The computation speed is greatly improved by reducing the eigenvalue problem for  non-symmetric matrices.

# Chapter 3

# Numerical Methods for General Dynamic System

## 3.1 Numerical Procedure for the Analysis of General Dynamic Systems

The fundamental formulation derived in Chapter 2 is implemented in the numerical algorithms provided in this chapter. The implementation is divided into four major steps. First, the real system parameters are reorganized to form the equivalent state-space formulation. Then the state-space system is transformed into generalized coordinates in complex plane in order to obtain the uncoupled modal system. Then each uncoupled modal system is solved by either direct integration method or frequency domain analysis. The response in each mode is then transform back to real coordinate and sum up to get the total response of the system. The steps are presented below in Figure 3.1.

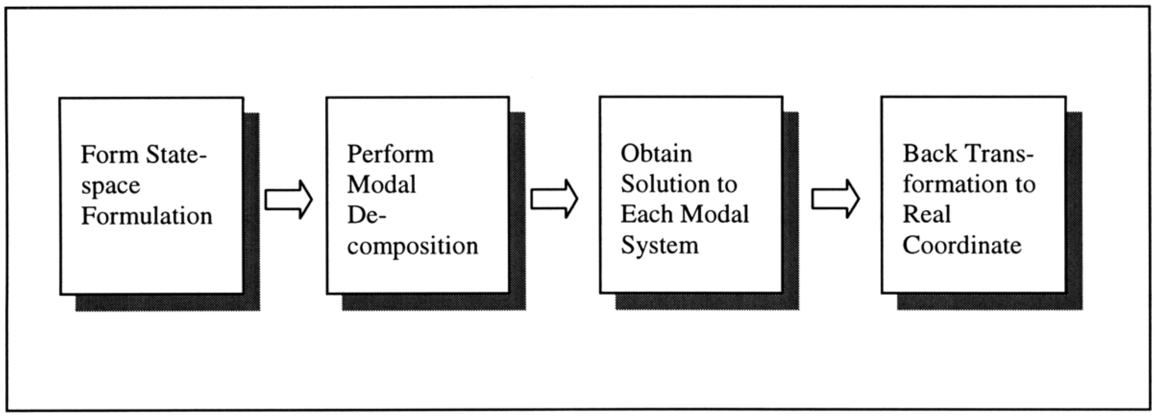

Form State-space Formulation
Perform Modal De-composition
Obtain Solution to Each Modal System
Back Trans-formation to Real Coordinate

**Figure 3.1 Procedure for the Analysis of General Dynamic Systems**

| | Step 1 | Step 2 | Step 3 | Step 4 |
|---|---|---|---|---|
| Procedure | Form State-space Formulation | Perform Modal Decomposition | Obtain Solution to Each Modal System | Back Trans-formation to Real Coordinate |
| Numerical Methods Required | Real Matrix Operation and Linear Equation Solver | Complex Eigenvalue Problem Solver | Direct Integration or Discrete Fourier Transformation | Complex Matrix Operation and/or Linear Equation Solver |

**Table 3.1 Requirement for Numerical Methods in the Analysis Procedure**

Table 3.1 shows the numerical methods required in each step of the procedure. The first step corresponds to the state-space formulations in Section 2.1. As shown in Equation 2.2 to 2.6, the inverse of the mass matrix have to be computed, then several matrix multiplication is needed to form matrix **A** and the forcing term. After the elements in Equation 2.4 are formed, the system is transformed into uncoupled modal equations in step 2. In order to uncouple the system, procedure for complex eigenproblem in Section 2.2 is performed to obtain the modal system described in Equation 2.15. Thus, complex eigenproblem solver is needed in this step. The solution to each modal equation is obtained after the system is uncoupled into its modal formulation. In this step, either direct integration or discrete Fourier transformation is performed to obtain the response of arbitrary excitation. If free vibration problem is considered, step 3 will not be performed. Instead, linear equation solver in step 4 is performed to obtain the solution to the initial value problem. Then, in both of the response cases, i.e., both forced and free response, complex modal effects are combined to produce the real response history of the system as presented in Section 2.5.

The next sections provide the numerical procedures applied for the analysis. Because the matrix operations and matrix inversion is easier to implement, thus, focus of the following section will be on the complex eigenproblem solver, discrete Fourier transformation and direct integration.

## 3.2 Numerical Methods for Complex Eigenproblem

The numerical procedure for solving complex eigenproblem is rather difficult compare to its conceptual formulation. Moreover, the eigenvalue of the nonsymmetric matrices are often sensitive to small changes and are even defective. Thus, it is impossible to guarantee an accurate solution to such a problem as is to symmetric eigenproblem [8].

The solution procedure of nonsymmetric eigenproblem is divided into four major steps as shown in Figure 3.2. The first step is balancing, which helps preventing the sensitive problem in the following step. The second step is the reduction procedure to upper

Hessenberg form. Gaussian elimination with pivoting is introduced to eliminate the lower triangular elements except for the off-diagonal elements. Then, the QR algorithm for real Hessenberg matrices is applied to find the complex conjugate eigenvalues of the matrix. After all the eigenvalues are obtained, several eigenvalues needed proceed with inverse iteration procedure to obtain their corresponding right and left eigenvectors. Descriptions of the procedures are provided in Sections 3.2.1 to 3.2.3.

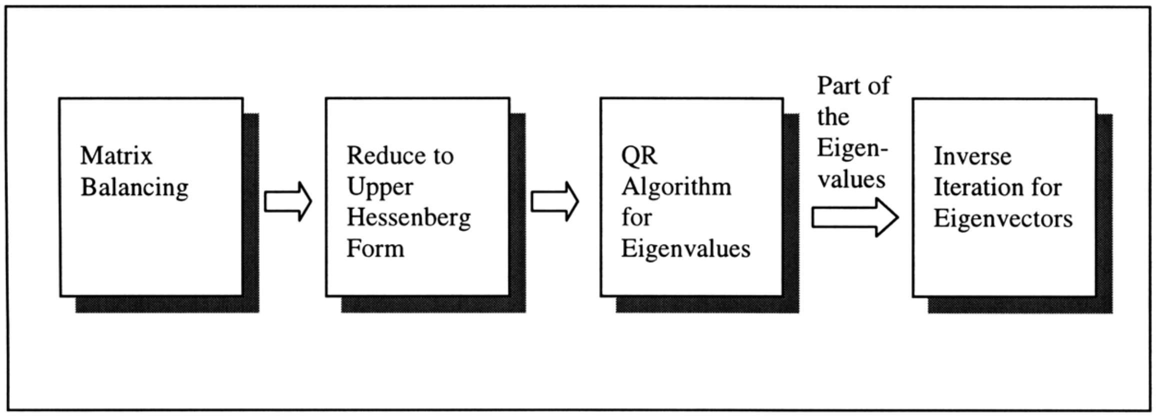

Matrix Balancing
Reduce to Upper Hessenberg Form
QR Algorithm for Eigenvalues
Part of the Eigen-values
Inverse Iteration for Eigenvectors

**Figure 3.2 Procedures for Finding Eigenvalues and Eigenvectors for nonsymmetric matrix**

### 3.2.1 Matrix Balancing and Reduction to Upper Hessenberg Form

Matrix balancing provides a good way to deal with sensitivity with rounding errors and machine accuracy for nonsymmetric matrix. Based on the fact that the errors produced by the system are generally proportional to the Euclidean norm of the matrix, the idea of matrix balancing is to use similarity transformations to obtain a matrix with equal norm in corresponding rows and columns to their pivot elements, thus reduce the norm of the matrix. The reason why similarity transformations are applied is to keep the original eigenvalues of the matrix.

The algorithm of balancing comes from Osborne in [8]. It first calculates the row and column norms of the matrix, then calculates the transformation matrix that balances the norm of rows and columns. In order not to worsen the accuracy by the transformations, the elements in the transformation are selected powers of machine radix base.

Notice that if all the eigenvectors are desired, or the inverse iteration is not applied to find the eigenvectors, the similarity transformations have to be kept track of. Because the multiplication of all the similarity transformations made to produce the eigenvalues is the matrix of eigenvectors.

After the balancing procedure is done, the matrix is reduced to upper Hessenberg form. The reason why nonsymmetric matrix can not be directly reduced to diagonal matrix is that the matrix is not symmetrical, the similarity transformation applied on the matrix to cancel out some part of the matrix will not cancel out the symmetric part of it. If one wants to cancel out the other part when one part has become zeros before, then usually the part that has become zero will grow back again because of the transformation. Thus, the upper Hessenberg matrix looks like the following in Figure 3.3 has to be the intermediate state of the solution process. Then different kind of procedure is introduced to search for eigenvalues of the system.

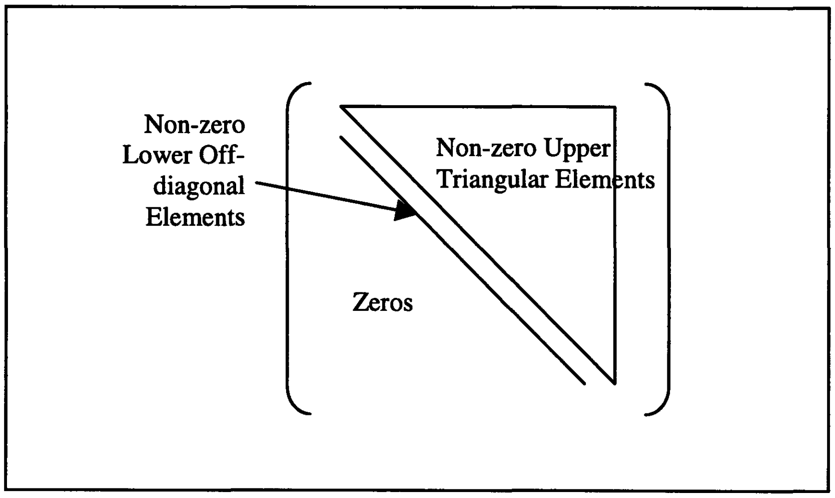

Non-zero Lower Off-diagonal Elements
Non-zero Upper Triangular Elements
Zeros

**Figure 3.3 Elements of Upper Hessenberg Matrix**

The transformation of the matrix into its upper Hessenberg form can be achieved by any methods for reducing symmetric matrix to its tridiagonal form. Thus, Givens method, Householder transformation and Gaussian elimination with pivoting are all qualified for the transformation.

The most efficient method among all is Gaussian elimination with pivoting. The reason why Gaussian elimination has to be performed with pivoting is that Gaussian elimination itself is not a similarity transformation. To make the transformation a similarity transformation, column exchanges have to come with corresponding row changes in the transformation of finding pivot element. The procedure of Gaussian elimination with pivoting is as follows:

1. For stage $i$ of the procedure, the absolute value of the elements below the diagonal elements of the $i$th column are compared with the diagonal elements, assume the largest element is $i+r$.
2. Interchange rows $i+1$ and $i+r$, also interchange column $i+1$ and $i+r$ for similarity transformation.
3. For row $k$ greater than $i+1$, subtract $a_{k,i}/a_{i+1,i}$ times row $i+1$ from row $k$. Also add $a_{k,i}/a_{i+1,i}$ times column $k$ to column $i+1$ for similarity transformation.

### 3.2.2 The QR Algorithm for Real Hessenberg Matrix

The QR algorithm of real Hessenberg matrix is adapted from the QR algorithm for tridiagonal matrix. The idea of QR algorithm is based on the fact that any real matrix **A** can be decomposed by the Householder transformation to the form

$$\mathbf{A} = \mathbf{Q} \cdot \mathbf{R} \tag{3.1}$$

while keeping its eigenvalues unchanged. The **Q** matrix in Equation 3.1 is orthogonal and **R** is upper triangular, assume the next step of Equation 3.1 is

$$\begin{aligned} \mathbf{A}_1 &= \mathbf{R} \cdot \mathbf{Q} = \mathbf{Q}^T \cdot \mathbf{A} \cdot \mathbf{Q} \\ \mathbf{A}_1 &= \mathbf{Q}_1 \cdot \mathbf{R}_1 \\ \mathbf{A}_2 &= \mathbf{R}_1 \cdot \mathbf{Q}_1 = \mathbf{Q}_1^T \cdot \mathbf{A}_1 \cdot \mathbf{Q}_1 \end{aligned} \tag{3.2}$$

From Equation 3.2, $\mathbf{A}_s$ can be found by repeating the process s times. An important theorem is applied in Equation 3.2 providing a connection of the transform to the solution of eigenvectors. The theorem is stated as follows

*If* **A** *has eigenvalues of different absolute value* $|\lambda_i|$, *then* $\mathbf{A}_s$ *approaches the upper triangular form as* $s \to \infty$. *The eigenvalues appear on the diagonal in decreasing order of absolute magnitude.*

Thus, keep doing the QR decomposition and multiply them in the reverse manner will make the target matrix approaches its eigenvalues. Another useful information in the process of proving the theorem is that the lower-diagonal terms will vanish at the order as follows

$$a_{ij}^{(s)} \propto \left( \frac{\lambda_i}{\lambda_j} \right)^s \tag{3.3}$$

where *i>j* for lower triangular elements. Equation 3.3 points out a pitfall of the algorithm, that is, if the *i*th eigenvalue is very close to *j*th eigenvalue, the convergence may be slow even if the steps s is large. Thus, shifting is introduced to accelerate the convergence rate. Introduce the following equation

$$\begin{aligned} \mathbf{A}_s - k_s\mathbf{I} &= \mathbf{Q}_s \cdot \mathbf{R}_s \\ \mathbf{A}_{s+1} &= \mathbf{R}_s \cdot \mathbf{Q}_s + k_s\mathbf{I} = \mathbf{Q}_s{}^T \cdot \mathbf{A}_s \cdot \mathbf{Q}_s \end{aligned} \tag{3.4}$$

with all eigenvalues shifted by $\lambda_i$ -$k_s$. The ratio of eigenvalues in Equation 3.3 becomes

$$a_{ij}^{(s)} \propto \left( \frac{\lambda_i - k_s}{\lambda_j - k_s} \right)^s \tag{3.5}$$

Thus, the procedure can have a better convergence rate if $k_s$ is chosen close to $\lambda_i$.

Another method named implicit shift similar to the shift procedure is developed to prevent the loss of accuracy of small eigenvalues from subtracting operation. The difference between the implicit shift and shift is that the shift is embedded in the transformation matrices instead of directly subtract one of the eigenvalues from the **A** matrix. The detailed proof of implicit shift can be found from [6] and [8].

The QR algorithm for real Hessenberg matrix is slightly different from the above QR algorithm. This is because complex eigenvalues are expected instead of real eigenvalues. There may exist 2x2 isolated diagonal elements represent complex pairs of eigenvalues. To incorporate this situation, two steps of QR algorithm are combined together using implicit shift to obtain 2x2 diagonals.

Combining two steps of the QR algorithm with shifts in Equation 3.4, Equation 3.6 is obtained

$$\mathbf{A}_s \cdot \mathbf{Q}^T = \mathbf{Q}^T \cdot \mathbf{A}_{s+2} \tag{3.6}$$

Comparing Equation 3.6 to the form of implicit shift as follows

$$\mathbf{A}_s \cdot \overline{\mathbf{Q}}^T = \overline{\mathbf{Q}}^T \cdot \mathbf{H} \tag{3.7}$$

From the theorem of implicit shift, if the first column of $\mathbf{Q}^T$ and $\overline{\mathbf{Q}}^T$ are the same, then matrix $\mathbf{Q}^T$ and $\overline{\mathbf{Q}}^T$ are the same and $\mathbf{A}_{s+2}$ is equal to **H**. Thus, the strategy is to find the first row of **Q** that matches the first row of $\overline{\mathbf{Q}}$, insert Equation 3.7 into Equation 3.6 to obtain $\mathbf{A}_{s+2}$ for the next iteration.

The matrix **Q** is constructed by a multiplication of series of Householder matrix $\mathbf{P}_{n-1}$, $\mathbf{P}_{n-2}$ to $\mathbf{P}_1$, where $\mathbf{P}_1$ determines the first row of **Q**. From the formulation of Householder transformation and the formulation of double shifts, $\mathbf{P}_1$ is determined as follows

$$\mathbf{P}_1 = \mathbf{I} - 2\mathbf{w}_1 \cdot \mathbf{w}_1^{\mathrm{T}} \tag{3.8}$$

where

$$2\mathbf{w}_1 \cdot \mathbf{w}_1^{\mathrm{T}} = \begin{bmatrix} (p_1 \pm s_1)/\pm s_1 \\ q_1/\pm s_1 \\ r_1/\pm s_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & q_1/(p_1 \pm s_1) & r/(p \pm s_1) & 0 & \cdots & 0 \end{bmatrix} \tag{3.9}$$

and

$$\begin{aligned} p_1 &= (a_{nn} - a_{11})(a_{n-1,n-1} - a_{11}) - a_{n-1,n}a_{n,n-1} + a_{12}a_{21} \\ q_1 &= a_{21}[a_{22} - a_{11} - (a_{nn} - a_{11}) - (a_{n-1,n-1} - a_{11})] \\ r_1 &= a_{21}a_{32} \\ s_1^2 &= p_1^2 + q_1^2 + r_1^2 \end{aligned} \tag{3.10}$$

$\mathbf{P}_{n-1}$ to $\mathbf{P}_2$ are obtained by the same manner. For example, substitute $a_{11}$, $a_{12}$, $a_{21}$ and $a_{32}$ by $a_{22}$, $a_{23}$, $a_{32}$ and $a_{43}$ obtains $p_2$, $q_2$, $r_2$ and $s_2$. Let the normalized $p_1$, $q_1$ and $r_1$ occupy the 2,3,4 row of the $w_2$ vector, substitute in Equation 3.8 to obtain $\mathbf{P}_2$. The detailed derivation of how to come up with the formulation can also be found from [6] and [8].

The criterion for testing if any eigenvalue has been found is on the test of subdiagonal elements. If the dimension of the matrix is n and if $a_{n,n-1}$ is negligible, then a real eigenvalue is found on $a_{n,n}$. Else, if $a_{n-2,n-1}$ is negligible, then there are two real eigenvalues or two conjugate complex eigenvalues on the 2x2, $a_{n-1,n-1}$ to $a_{n,n}$ diagonal block. Iterate until all the diagonal blocks are found and the corresponding eigenvalues are obtained from the diagonal blocks.

### 3.2.3 Inverse Iteration for Eigenvectors

After all the eigenvalues are found from QR algorithm for Hessenberg matrix, some selected eigenvalues are thrown into the inverse iteration procedure to obtain their corresponding eigenvectors. In the case of vibration analysis, usually only the response of first five modes are desired. Thus the eigenvalues are first sorted by their absolute value, then first five conjugate pairs of eigenvalues are selected to proceed with inverse iteration.

The idea of inverse iteration comes from the following equation

$$(\mathbf{A} - \chi\,\mathbf{I}) \cdot \mathbf{y} = \mathbf{b} \tag{3.11}$$

where $\chi$ is close to one of the eigenvalue $\lambda$ of **A**, **b** is a random vector and **y** is the solution to the equation. The solution **y** will be close to the eigenvector of $\lambda$, if $\chi$ is close enough to it. Thus, the strategy is to iterate for Equation 3.11 by repeatedly replace **b** by **y** and solve for the new **y**. The result of **y** will be closer and closer to the desired eigenvector.

Notice that if over half of the eigenvectors are desired, recording all the similarity transformation is faster than inverse iteration. Another issue of the inverse iteration method is its convergence and accuracy. Because it is an iterative method and the matrix can be defective or has close eigenvectors, good convergence often relies on good initial guess. In addition, because the exact eigenvalues and eigenvectors are not known, when to stop at a reasonable accuracy, or say, making large improvement from the original vector is a big problem. Generally speaking, the strategy is, for large initial growth of $|\mathbf{y}|$, the good guess of initial vector is obtained. If $|\mathbf{y}|$ does not grow fast enough, then try another initial vector. After a few iterations, if **b** does not change much, then the vector is assumed converging, else, recalculate the eigenvalue by Equation 3.12

$$\chi_{k+1} = \chi_k + \frac{1}{\mathbf{b}_k \mathbf{y}} \tag{3.12}$$

The criteria presented above ought to be suitable for most of the normal cases. For the defected matrix or closed eigenvectors, some other tricks are provided to deal with the problem. There are discussions about the convergence and accuracy of the method (see [6] and [8]) that provide some answers to those questions.

## 3.3 Algorithm for Fast Fourier Transformation

As mentioned in Section 2.4.1, Fourier transformation is carried out by integration back and forth from time domain to frequency domain. The response analysis of general dynamic calculates the transfer function as mentioned in Section 2.4.2, multiply it to the corresponding frequency domain excitation to produce a response. The procedure is presented in Equation 2.33 and 2.34.

Because the input for the system is discrete-time earthquake signals, plus the algorithm for digital computer has to deal with discrete time data, thus the equations have to turn into their discrete form. For Equation 2.31, approximate the integration to finite bounds and rewrite as follows

$$F_{jn} \approx \frac{1}{n}\sum_{m=0}^{N-1} f_j(t_m)\, e^{-i\frac{2\pi nm}{N}} \qquad n = 0, 1, 2\cdots, N-1 \tag{3.13}$$

where $t_m$=$m\,\Delta t$ and $\omega$=$2\pi n/N\Delta t$. Equation 2.32 can also be rewritten as follows in discrete form

$$q_{jm} \approx \sum_{n=0}^{N-1} H_{jn}\, F_{jn}\, e^{i\frac{2\pi nm}{N}} \qquad m = 0, 1, 2\cdots, N-1 \tag{3.14}$$

Equation 3.13 together with Equation 3.14 forms a complete discrete Fourier transform pair for mode $j$.

Looking into the two discrete equations, we can observe that the computation is an O($N^2$) process. That means if there are thousands of sampled data, which is normal for seismic analysis, the effort used in carrying the computations is exorbitant. However, by reevaluate the exponential terms in the two equations, the process can be reduced to O($N\ log_2N$).

The reduction proposed by Danielson and Lanczos in 1942 is called Danielson-Lanczos Lemma. The lemma showed that a discrete Fourier transform of length N can be rewritten as the sum of two discrete Fourier transforms each of length N/2. Each subset contains either even numbered data or odd numbered data. The proof of the lemma can be accomplished by separating the even and odd numbered points. (Details of the proof and the theory of fast Fourier transformation can be found on [6] and [9].) Furthermore, if N is an integer power of 2, the lemma can be further applied, then the four subsets will have four combination of input data, i.e., even-even, even-odd, odd-even, odd-odd. Continue this process until the data have been subdivided all the way down to transforms of length 1. The transformation of length one is a combination of even and odd data.

The strategy here is to compute all the possible one-point transform we need. However, the question of this strategy is how to figure out which one-point transformation corresponds to which output. The answer is provided by looking at the pattern of even and odd data combinations in the one-point transformation. For the discrete transformation in Equation 3.13, the way to figure out which *m* corresponding to which *n* is to take the bit reversal of the binary value of *n*. The example of the data ordering for an eight-point Fourier transformation is in Figure 3.4.

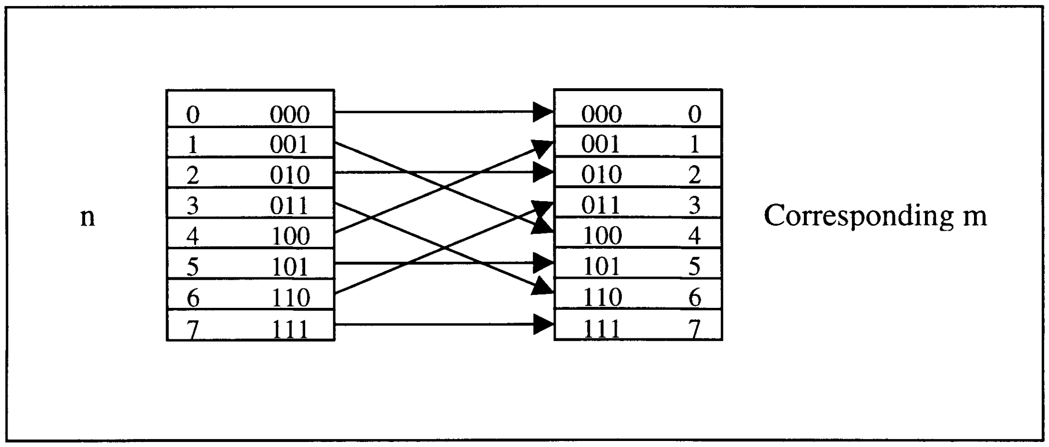


**Figure 3.4 Data Ordering for an Eight-point Fourier Transformation**

Previous procedure is the basic algorithm of fast Fourier transformation(FFT) proposed by Cooley and Tukey. Notice that this algorithm is for the data length of integer power 2. There are other algorithms dealing with some small power of 2 as a basis of the transformation, for example, 4 (base-4 FFT) and 8 (base-8 FFT). They are also provided in [6] and [9].

There exist problems in the basic FFT algorithm. The first is concerned with the input data. It is very often that the length of the input data set is not equal the integer power of 2. The second is rather a physics problem. If the response of excitation of the system has not died out after the length of FFT, the response near the beginning will be influenced by the end excitation. This is because when the infinite integral is cut into a finite summation form, it is assumed that the excitation is a periodic function. This procedure is valid because our interest is only in a finite period instead of infinite. Thus, although the excitation has been turned into periodic function and the response of it is also periodic, we neglect it and treat it as zero because it is out of the range of our interest. However, problem comes in when there is not enough time for the response of the previous period of excitation to die out and a new one comes in. This is called the wrap-around effect.

There is one simple solution to these problems, which is zero padding. For the first case where the data is not the integer power of 2, padding zeros at the end of input data set to form a complete set with length integer power of 2. For the second case, padding zeros at the end of input data set so that the response dies out before a new period starts. The length of padding should be determined by the larger one of the previous two padding cases. Combine the two requirements for zero padding, a criteria comes up as follows

*The original length plus the padding length should be the larger length of integer-power-of-two than the length for the response to die out.*

To find the padding length meeting this criteria, the length for the response to die out has to be determined first. Equation 3.15 shows the relationship between the decay rate of free response to the number of cycles and the damping ratio.

$$\ln\frac{v_n}{v_{n+m}} = \frac{2m\pi\xi}{\sqrt{1-\xi^2}} \tag{3.15}$$

where $n$ and $m$ are the number of cycles, $v$ is the peak response in the cycle and $\xi$ is the damping ratio. If the desired percentage of decay is 90%, then from equation 3.15, the number of cycle $m$ to die out is as follows

$$m = \frac{\ln 10 \cdot \sqrt{1-\xi^2}}{2\pi\xi} \tag{3.16}$$

where the modal damping ratio $\xi$ can be obtained from the modal analysis of the general dynamic system as presented in Section 2.5. The time interval requires to damp out the response is thus $m\Delta t$, where $\Delta t$ is the time step of the discrete system. Thus, if the original time duration is $t_i$, then the total padding time $t_p$ needed is as follows

$$t_p = 2^{ceil(\log_2(t_i + m\Delta t))} - t_i \tag{3.17}$$

where function *ceil(x)* rounds x to the nearest integer towards infinity.

The application of FFT in the response of general dynamic system is through Equation 3.13 and 3.14. First, select several modes of interest, throw the data from the eigensystem solver into the FFT algorithm for each mode. Then, multiply the frequency domain transfer functions to each of the corresponding frequency content. Then, do inverse FFT back to the time domain and sum up the response from each mode.

## 3.4 Algorithm for Direct Integration

Direct integration method is a step-by-step approach that divides the loading and response history into a series of time intervals or "steps". Each of the steps is treated as an independent analysis problem thus only the initial conditions from the previous time step are taken into account. There are generally two kinds of direct integration methods, that is, explicit integration method and implicit integration method. Explicit integration method takes the equilibrium condition at current time step, whereas the implicit integration method takes the equilibrium condition at the next time step. The explicit integration is easier to formulate, and the computation for each time step is less. However, the convergence of the explicit integration method is conditional. Generally speaking, the time interval in the explicit integration method has to be small enough to meet the convergence criteria. For the response history analysis, the time history is usually larger than 15 seconds, and the minimum time interval is limited by the time interval of recorded earthquake signal. Thus, the explicit integration method is not suitable for the analysis.

On the other hand, one of the implicit integration method proposed by Newmark provided an unconditionally stable scheme. The method named Newmark method is based on the following assumption on the displacement and velocity in the next time step

$$^{t+\Delta t}\dot{\mathbf{U}} = {}^{t}\dot{\mathbf{U}} + [(1-\delta)\ {}^{t}\ddot{\mathbf{U}} + \delta\ {}^{t+\Delta t}\ddot{\mathbf{U}}]\,\Delta t \tag{3.18}$$

$$^{t+\Delta t}\mathbf{U} = {}^{t}\mathbf{U} + {}^{t}\dot{\mathbf{U}}\Delta t + [(\frac{1}{2}-\alpha)\ {}^{t}\ddot{\mathbf{U}} + \alpha\ {}^{t+\Delta t}\ddot{\mathbf{U}}]\,\Delta t^2 \tag{3.19}$$

where ${}^{t}\mathbf{U}$, ${}^{t}\dot{\mathbf{U}}$ and ${}^{t}\ddot{\mathbf{U}}$ are the displacement, velocity and acceleration at time t, and ${}^{t+\Delta t}\mathbf{U}$, ${}^{t+\Delta t}\dot{\mathbf{U}}$ and ${}^{t+\Delta t}\ddot{\mathbf{U}}$ are the displacement, velocity and acceleration at time t+Δt. α and δ are the control parameters for stability and accuracy. When α = 0.25 and δ = 0.5, the method becomes unconditionally stable.

In addition to Equation 3.18 and 3.19, the equilibrium equation is taken at time t+Δt in Equation 3.20

$$\mathbf{M}\ {}^{t+\Delta t}\ddot{\mathbf{U}} + \mathbf{C}\ {}^{t+\Delta t}\dot{\mathbf{U}} + \mathbf{K}\ {}^{t+\Delta t}\mathbf{U} = {}^{t+\Delta t}\mathbf{F} \tag{3.20}$$

Thus three equations, Equation 3.18 to 3.20, are able to solve for three unknowns, the displacement, velocity and acceleration of time t+Δt. The following procedure is provided as the algorithm of the Newmark method

1. Initialize ${}^{0}\mathbf{U}$, ${}^{0}\dot{\mathbf{U}}$ and ${}^{0}\ddot{\mathbf{U}}$ from the initial conditions. Check if the initial conditions satisfy the equations of motion at time 0.
2. Select time step Δt and parameters α and δ. Calculate the following constants:

$$a_0 = \frac{1}{\alpha \Delta t^2};\quad a_1 = \frac{\delta}{\alpha \Delta t};\quad a_2 = \frac{1}{\alpha \Delta t};\quad a_3 = \frac{1}{2\alpha} - 1;$$
$$a_4 = \frac{\delta}{\alpha} - 1;\quad a_5 = \frac{\Delta t}{2}(\frac{1}{2\alpha}) - 2;\quad a_6 = \Delta t(1-\delta);\quad a_7 = \delta \Delta t; \tag{3.21}$$

3. For given mass matrix **M**, damping matrix **C**, and stiffness matrix **K**, calculate the effective stiffness matrix $\tilde{\mathbf{K}}$.

$$\tilde{\mathbf{K}} = a_0\mathbf{M} + a_1\mathbf{C} + \mathbf{K} \tag{3.22}$$

4. Decompose $\tilde{\mathbf{K}}$.
5. For each time step:
   a. Calculate the effective load:

   $${}^{t+\Delta t}\hat{\mathbf{F}} = {}^{t+\Delta t}\mathbf{F} + \mathbf{M}(a_0\,{}^{t}\mathbf{U} + a_2\,{}^{t}\dot{\mathbf{U}} + a_3\,{}^{t}\ddot{\mathbf{U}}) + \mathbf{C}(a_1\,{}^{t}\mathbf{U} + a_4\,{}^{t}\dot{\mathbf{U}} + a_5\,{}^{t}\ddot{\mathbf{U}}) \tag{3.23}$$

   b. Solve for the displacement matrix at time t+Δt

   $$\tilde{\mathbf{K}}\;{}^{t+\Delta t}\mathbf{U} = {}^{t+\Delta t}\tilde{\mathbf{R}} \tag{3.24}$$

   c. Calculate accelerations and velocities at time t+Δt for the next time step.

The procedure can be applied to modal analysis as an alternative method to fast Fourier transformation. For the modal system described in Equation 2.31, the procedure is simplified into a one dimensional problem. After assigning **M** = 0, **C** = 1, and **K** =$\lambda_i$, the response is obtained by simply iterating through the procedure.

# Chapter 4

# Architecture and Implementation of the Dynamic Simulation Tool

## 4.1 Object-oriented Approach for System Design

The object-oriented approach for design is characterized by organizing and structuring a system by defining several entities, their relations and the functions they perform. The entities mentioned previously are called objects. Everything in the real world can be defined as object as long as the relation and functions are well defined. For example, a car can be defined as an object. It has several properties such as year, make and mileage. It also has several functions such as start, turn and stop. It has many objects on it such as wheels, windshield and seats.

The reason the object-oriented approach is applied in developing the tool instead of traditional programming and design style is presented as the following:

1. The object-oriented approach represents the entities of the real world without distorting or decomposing them: For a simulation tool for seismic analysis, the structures and their associated properties are very complex. If the tool is designed in non object-oriented way, the properties will not associate with the structures, instead,

they appear whenever the computation or demonstration needs them. This way obeys human's way of thinking. In addition, it will cause the program very hard to develop and maintain.

2. Easy to develop and test new application: The new application is done by specifying the interaction between its objects. Thus, like the real world, different parts (or objects) can be developed separately, or simply use product develop previously or from other developer. The testing is also similar to the case of the real world. Each object is tested separately instead of having them in a big main program and do testing together. Thus, the problem is uncoupled into independent objects.

3. Easy for re-use or make extension of existing software: Because each entity is defined as an object, once the function or attribute of the entity is changed, the change can be made within the object. If the attribute is changed in traditional structured program, much more effort will be made at modifying each section and function of the program. In addition, if a new entity is created and used in the program, the object-orient approach is to just define its relationship with others as well as its function. Then, the new entity can be immediately adopted by the program group. However, in structured programming, it is possible that the entire program has to be changed in order to fit in the new entity.

4. Easy to create high quality interactive man-machine interface: The external events are easier to be represented in the object-orient approach. For example, the reaction of a

mouse-click on an icon can be bind to an object that needs to react to this event, instead of the old style to explicitly check it in every loop of the process.

Thus, for the dynamic simulation environment for seismic analysis, the optimum development strategy is to introduce the object oriented approach. First, create all the engineering structure objects and define their properties. Also, create a numerical system object to represent the structure elements. Then create functions belong to the numerical system to produce the solution to excitations. Finally, bind the objects to the graphical user interface for reacting to the events. The previous procedures are the methodology for developing the dynamic environment for seismic analysis and will be presented in Section 4,2 to 4.4.

## 4.2 Fundamental Class Hierarchy of the Dynamic Environment

The first step of implementing the dynamic simulation environment in the object-oriented approach is to analyze and design the system, that is, to specify the objects in the system and their relationships. Imagine the dynamic simulation environment for seismic analysis is a laboratory named Motion Lab, then the equipment and facilities needed to run the lab are the objects needed in the environment. As shown in Figure 4.1, to construct a Motion Lab needs four components in the real world. First, there should be several structures to be tested for the response of different kinds of excitations. Secondly, there should be different kinds of excitations to exert on the structure. Thirdly, there should be a modal system analyzer to calculate and analyze the response of the structure. In addition, there should be a user interface to display the analyzed result to the people experimenting with the structures.

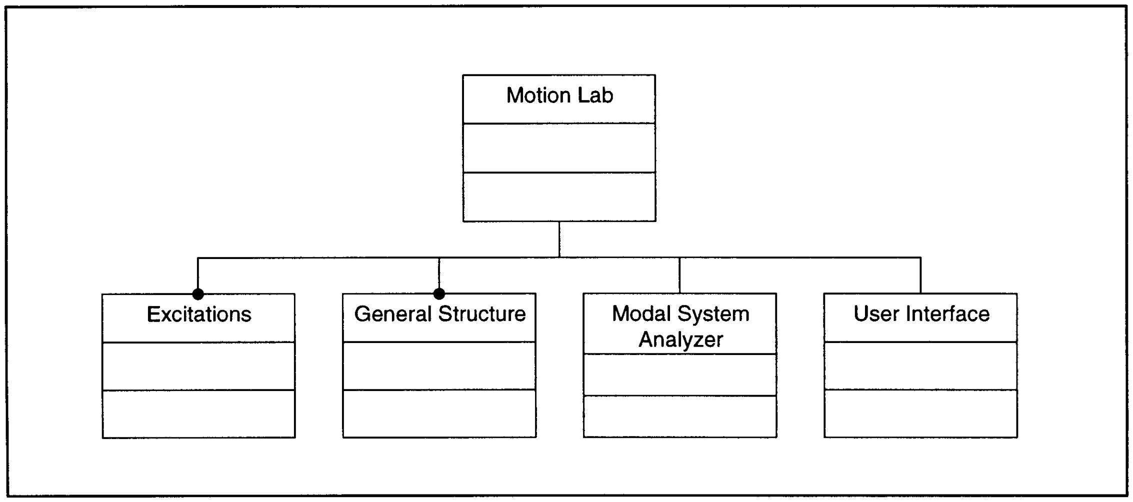


**Figure 4.1 Static model for Motion Lab**

The four objects have their associated objects as well. Excitations are classified as initial displacements and/or velocities, and general forcing functions. The structures are classified by its dynamic behavior. Shear structure stands for the kind of structure whose shear force between its floors or structural elements dominate the behavior of it, and its bending moment is negligible. On the other hand, bending structure stands for the kind of structure whose bending moment in its structural elements dominant the behavior of it, and its shear force is negligible. Generally speaking, the behavior of high-rise buildings is more similar to bending structure and the low-rise buildings are like shear structures. Figure 4.2 shows the static model of the excitation and general structure objects.

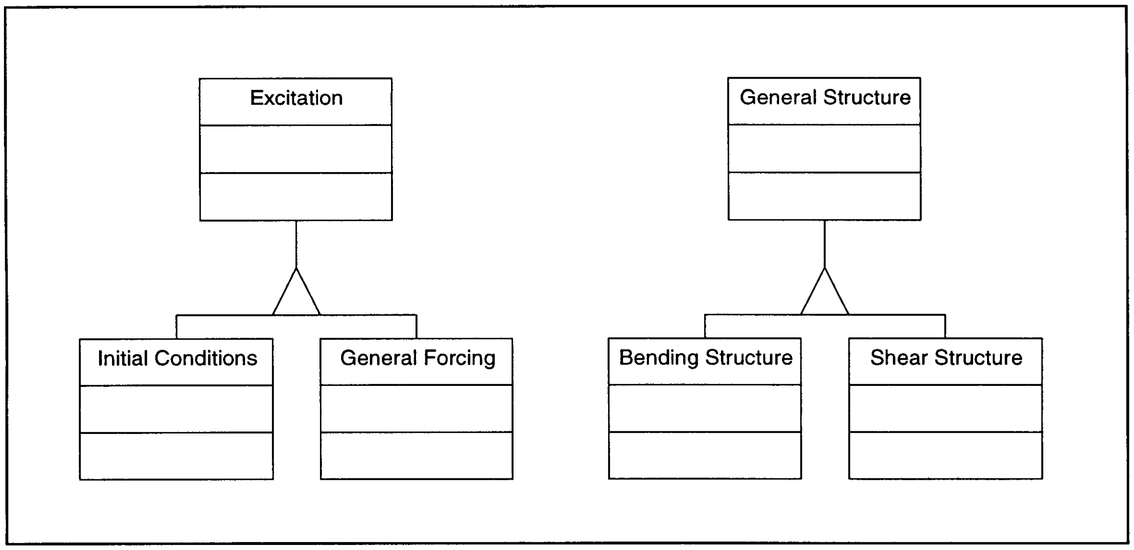


**Figure 4.2 Static model for the Excitation and General Structure**

The modal system analyzer is a dynamic response simulator using the numerical procedures provided in Chapter 3. All the functions needed for modal analysis of a general dynamic system, including eigenproblem solver, linear equations solver, fast Fourier transformation and direct integration procedures are provided in the analyzer. Figure 4.3 shows the static model for the modal system analyzer.

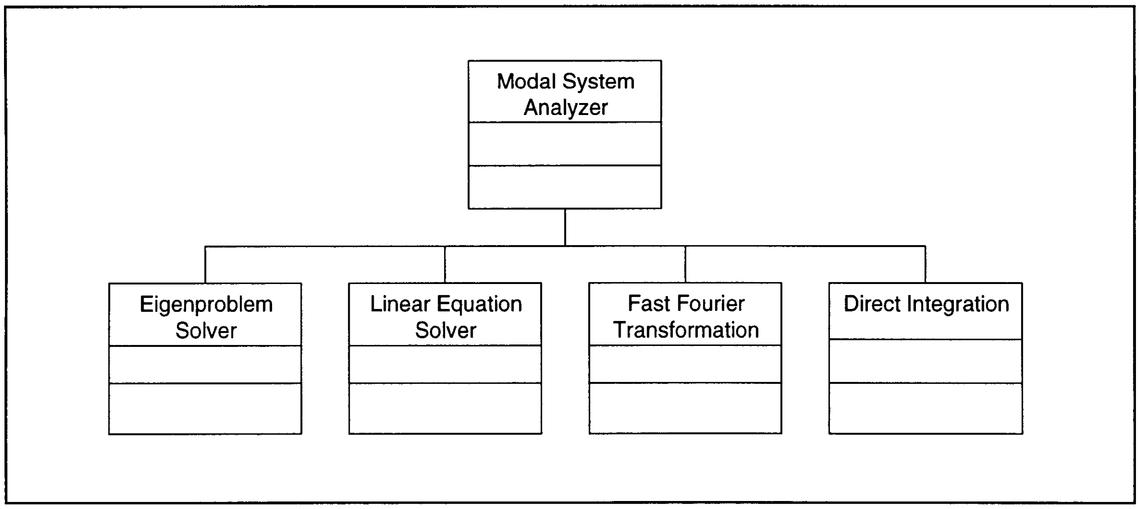


**Figure 4.3 Static model for the Modal System Analyzer**

The user interface is for the user to communicate with the environment and for the environment to display result to the user. For this purpose, the user interface is divided into two main functions: input and output. For the input function, three objects are created to accept three types of input event. First, structure properties such as mass, damping, and stiffness are obtained from users through the structural property frame. Second, The system properties such as display control, system parameters are obtained from the system attribute frame. The demo button frame is in charge of the starting signal of the response simulation.

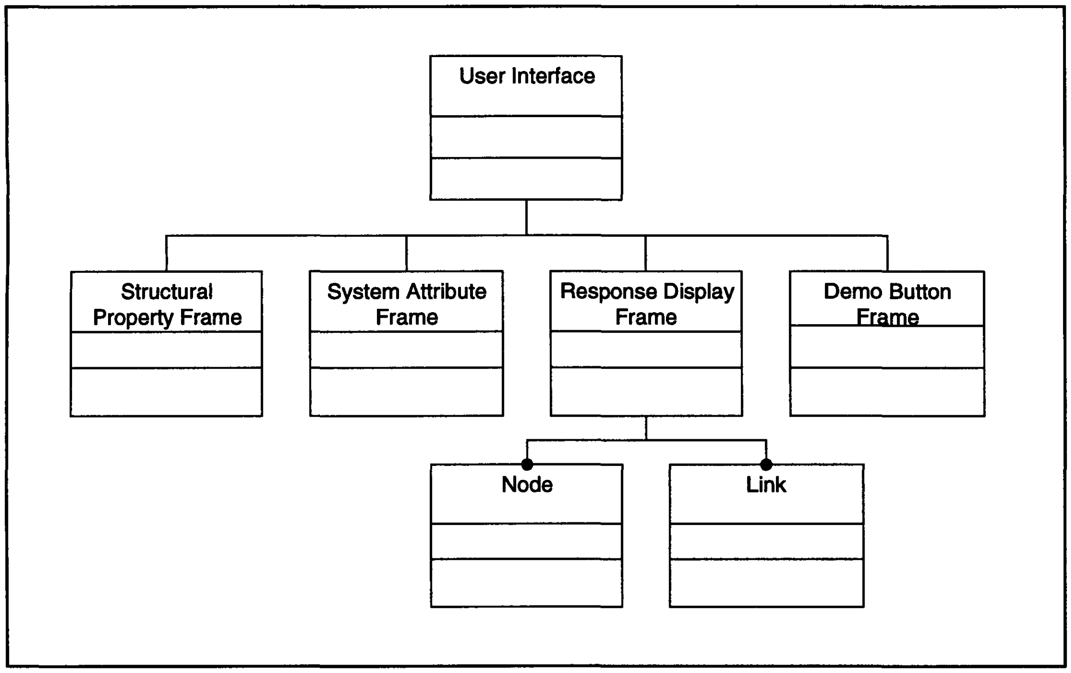


**Figure 4.4 Static model for the User Interface**

On the other hand, the simulation output of the system to the user is through the response display frame. Several node and link objects compose of a simple representation of a structural system is displayed through the frame.

## 4.3 Optimization of Computation in the Event Driven Programming

As stated in Section 1.1, part of the functional requirements for the environment is to provide immediate output with moderate accuracy. Thus, the amount of computation should be reasonable, or somehow, there is a way to obtain immediate response after input all the parameters of the system.

The response time problem can be alleviated by several strategies. From the scope of the environment provided in Chapter 1, the tool is for preliminary design or education, thus the degree of freedoms should not be too many. Usually less than 50 degrees of freedom is acceptable for this purpose. However, limited degrees of freedom does not make the response fast enough. Although the numerical procedure provided in Chapter 3 is known as the optimal procedure in computation, it still takes more than seconds to produce an output.

Thus, different perspective have to be proposed as a strategy to the problem. We know that the system is for preliminary design and education, people usually input their idea directly into the computer instead of read system parameters from file. Thus, it may take minutes to translate its idea and experimenting with the tool. The basic idea of the strategy is that if the computation can be somehow embedded in or parallel to the interactive input process between the environment an the user, then the immediate response is obtained. The idea is presented in Figure 4.5.

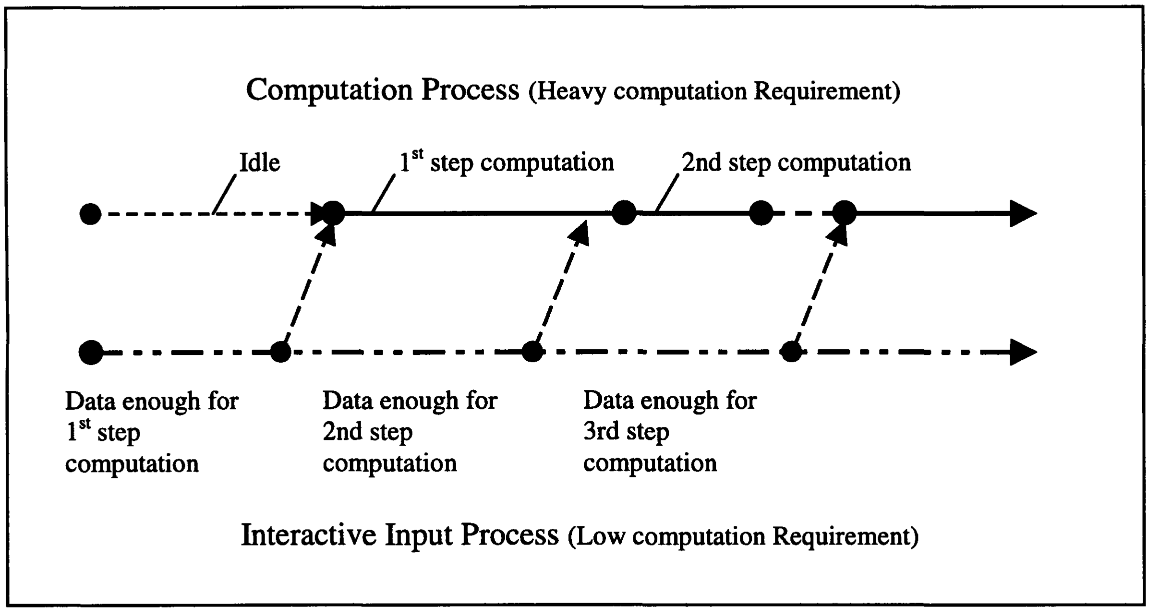

Computation Process (Heavy computation Requirement)
Idle
1st step computation
2nd step computation
Data enough for 1st step computation
Data enough for 2nd step computation
Data enough for 3rd step computation
Interactive Input Process (Low computation Requirement)

**Figure 4.5 The idea of parallel computation for the input and computation processes**

The strategy is based on the fact that the interactive input process requires less computation. Assume the numerical computation can be broken into smaller sub-processes and each sub-process only needs part of the parameters of the system. To implement the strategy, the numerical procedure provided in Chapter 3 is examined to see if it is divisible into smaller sections.

According to Table 3.1, there are four divisible steps in the numerical procedure: form state-space formulation, perform modal decomposition, obtain solution to each modal system, and back transformation to real coordinate. Though they can be performed as separate units, their sequence could not be changed. Next, look into the computation cost of the four steps. Most of the computation is spent in solving the eigenvalue problem, which is the procedure for modal decomposition. The second is the fast Fourier transformation for solving the modal equations. The first and the last steps cost very little computation. The first, which forms the state-space formulation, is the step for putting the input structure properties into the corresponding place in the state-space matrices. The last step is a transformation from complex to real coordinates. Thus the parallel strategy does not need to apply on these two steps.

Next, the interactive input process is examined to see which input is needed for the modal decomposition and the solution process to modal equations. Up to the step of modal decomposition, all the parameters of the system is needed including the degrees of freedom, the distribution of mass, stiffness, and damping of the structure system. To proceed the step of solving individual modal equations, the forcing data and the system setting have to be obtained. Thus, the precedence relationship of the two processes are in Figure 4.6.

The idea of having multiple processes running in the same program group is called threads. Threads are widely used in the application programming. Thus, the implementation of the processes in Figure 4.6 is available for almost all programming languages.

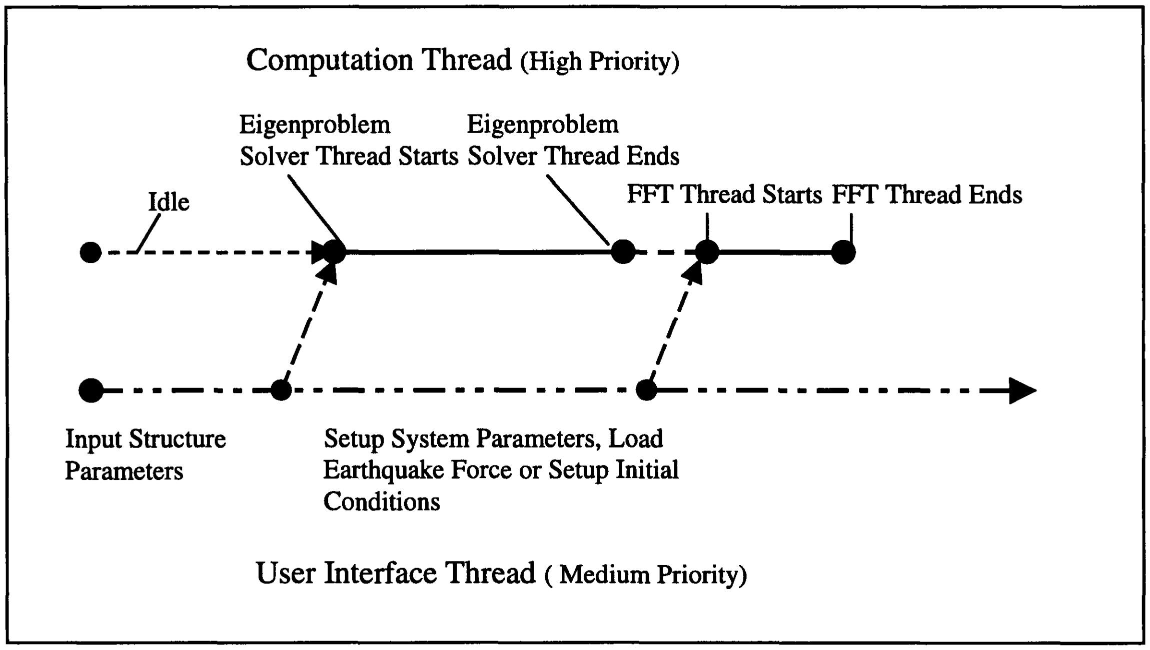

Computation Thread (High Priority)
Eigenproblem Solver Thread Starts
Eigenproblem Solver Thread Ends
Idle
FFT Thread Starts
FFT Thread Ends
Input Structure Parameters
Setup System Parameters, Load Earthquake Force or Setup Initial Conditions
User Interface Thread ( Medium Priority)

**Figure 4.6 Precedence diagram of user interface and computation threads**

Another strategy for optimize the computation performance is concerned with the accuracy. In the eigenproblem solver, the machine precision is highly demanded in order not to worsen the sensitivity problem. Thus, double precision floating point arrays are applied for the solver. In the fast Fourier transformation problem, the precision requirement is not that much as in the eigenproblem solver. The reason is that the earthquake forcing provided is only single precision and the output precision is accuracy enough with single precision value. Thus, apply single precision matrix operation in the fast Fourier transformation procedure is a better choice for both accuracy and computation cost considerations.

## 4.4 The Implementation of the Dynamic Simulation Environment

The tool named Motion Lab is implemented in JAVA programming language. The reason of selecting JAVA as the programming language is its advantage of platform portability. The program is mainly divided into two kinds of objects, one is the numerical objects for dynamics simulation, and the other is the user interface objects. Because Java is not very suitable for computationally intensive usage, thus no previous works has been done to implement the numerical methods for dynamic usage. Thus, the implementation of numerical object group have to start from scratch. From the algorithms provided in Chapter 3, non-symmetric matrix eigenproblem solver, fast Fourier transformation, and linear equation solver are built for the tool.

On the other hand, the user interface objects are built to provide connection between users and the numerical objects for dynamics. As presented in Figure 4.4, the user interface of the tool contains four major components: structural property frame, system attribute frame, response display frame, and demo button frame. Figure 4.7 shows the user interface of Motion Lab on the Netscape browser.

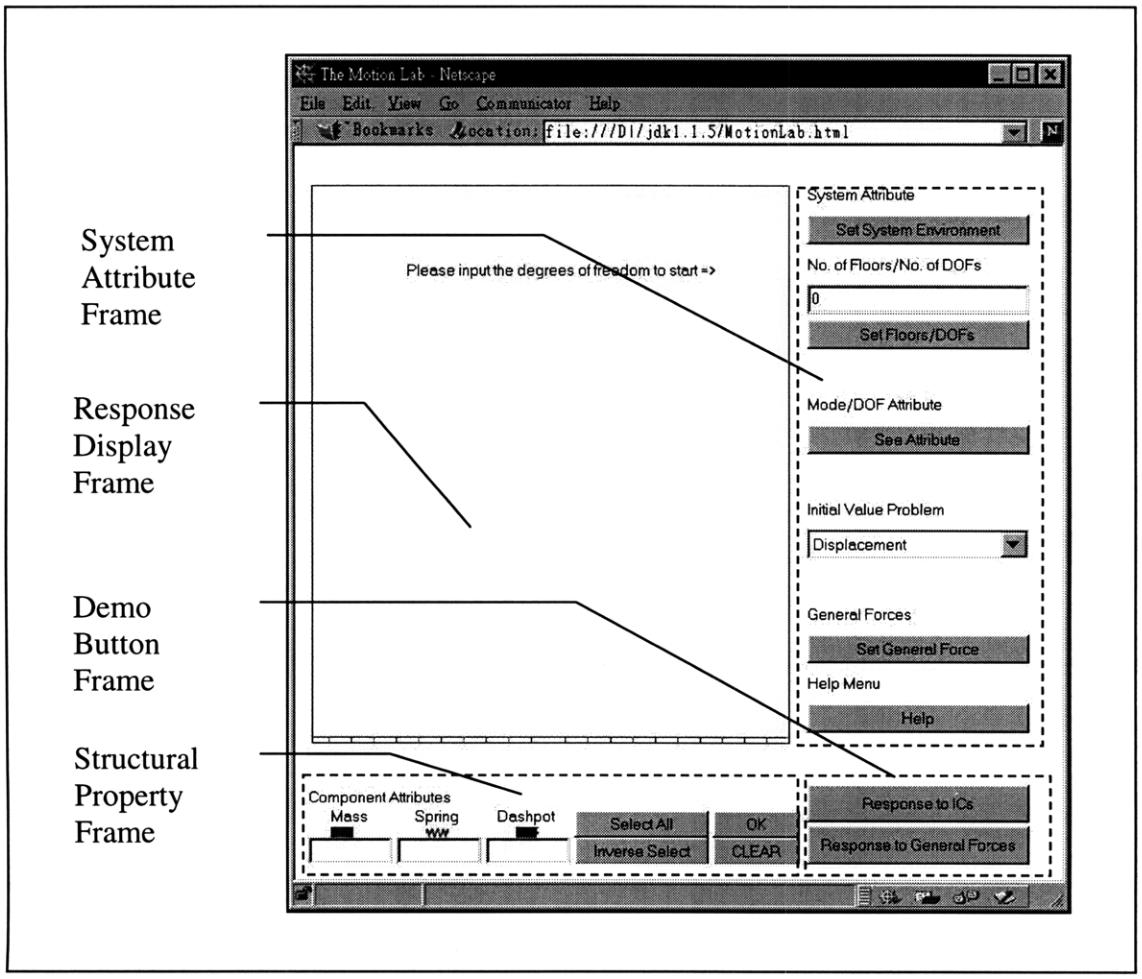
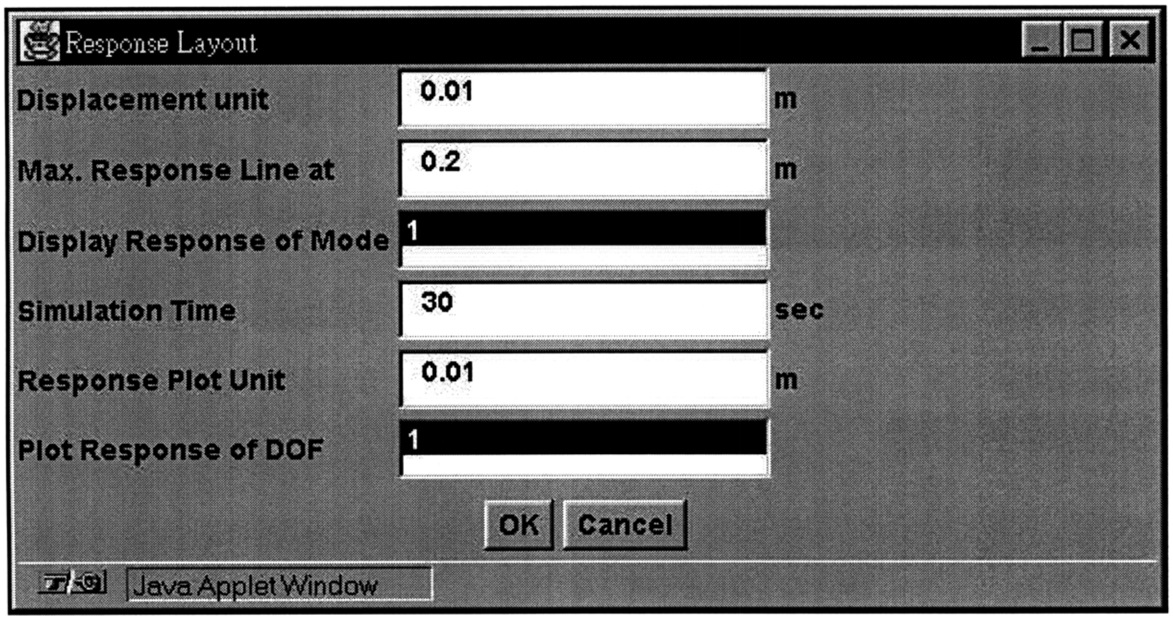


**Figure 4.7 User interface of Motion Lab on Netscape browser**

The four components each provides a main function in Section 4.2. To understand how the function of each component works, the detailed operations of each frame are provided as follows.

The system attribute frame is for collecting and displaying the information of the entire system. Several individual functions are provided in the frame. The functions including system parameters setup, degrees-of-freedom setup, mode/DOF attribute display, initial condition switch, and general force setup. The system parameter function starts when the

Set System Environment button is pressed. Figure 4.8 shows the system parameter dialog box pops up after the button is pressed.


Response Layout

| | | |
|---|---|---|
| Displacement unit | 0.01 | m |
| Max. Response Line at | 0.2 | m |
| Display Response of Mode | 1 | |
| Simulation Time | 30 | sec |
| Response Plot Unit | 0.01 | m |
| Plot Response of DOF | 1 | |

**Figure 4.8 System parameter setting dialog box**

The first item in the dialog box is for the displacement unit setting. Notice that the unit used for the system is SI unit and the displacement unit is meter per pixel for the dynamic simulation and display purpose. The user can adjust suitable screen display unit for the simulation.

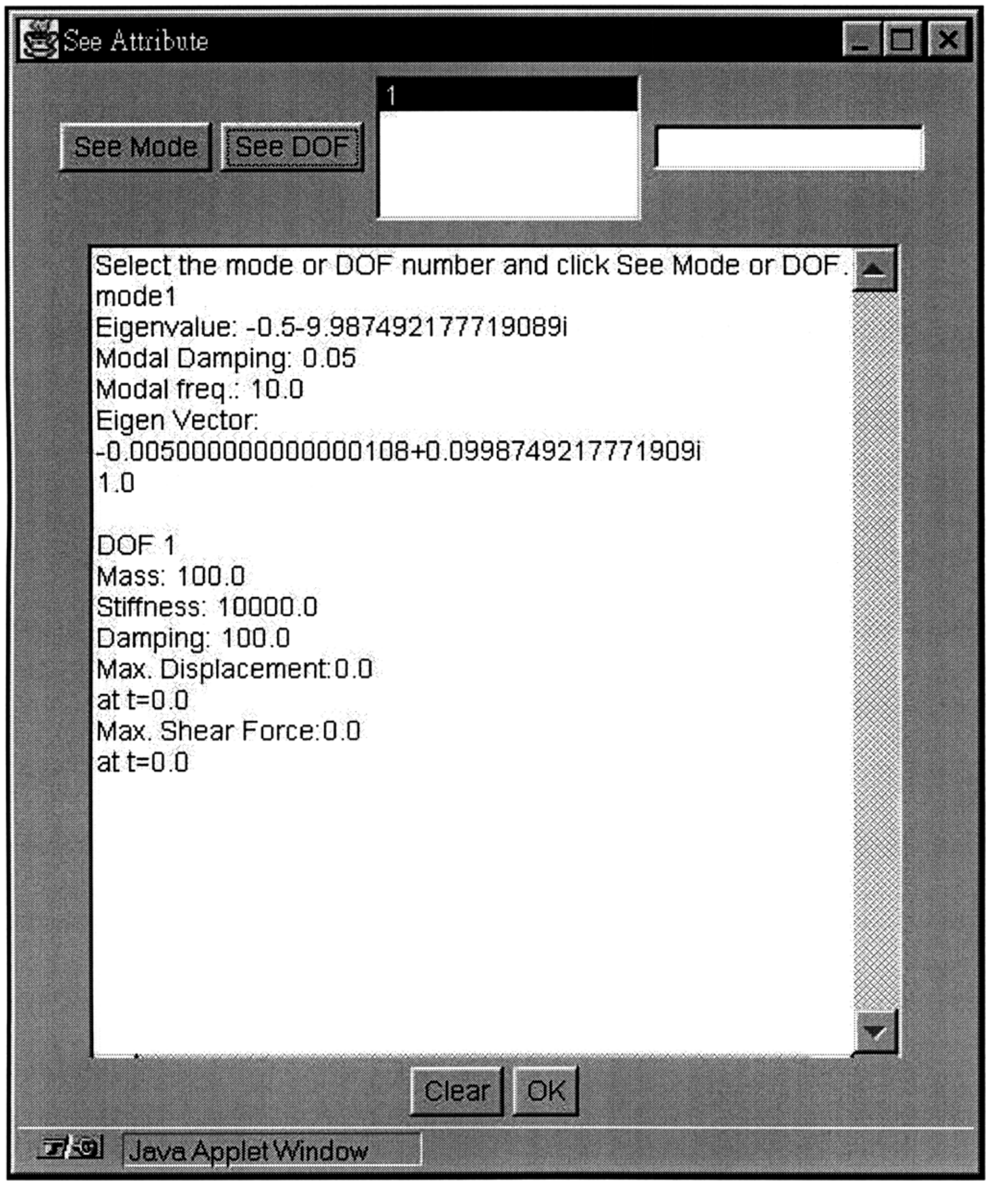
The next item is the maximum response line input field. It is an useful tool when the user experiments with different structural properties to get the optimum response. It draws two lines at the distance corresponding to the input on both sides of the structure. Thus, users can have the idea when response goes out of limit during the simulation.

The next is a mode superposition choice list. The user can specify the combination of several desired modal response by selecting them in the choice. The information of selected modes is then sent to the numerical analyzer for mode superposition.

The rest components are the simulation time, response plot unit and the degrees of freedom to display. The item of simulation time lets the user specify the length of simulation. The response plot unit setting and degrees of freedom setting let the user set the response of each pixel and the degrees of freedom desired on the response display dialog.

The next function of the system attribute frame is the setup for the degrees of freedom of the system. After the user input the degrees of freedom desired and presses the "Set Floors/DOFs" button, a structure composed of nodes and links corresponding to the desired degrees of freedom is displayed on the response display frame. This should be the first procedure of the input process because all of the system settings are related to the degrees of freedom of the system.

The next function is the Mode/DOF attribute display panel. The panel provides the quantitative output of every modes and degrees of freedom. As shown in Figure 4.9, the panel is composed of two main areas. The upper area is for the user to input desired modes or degrees of freedom to display and lower area is for display the system parameters of the desired modes or degrees of freedoms. The user can first select the desired modes or degrees of freedoms to display in the upper-center list, then press "show modes" or "show DOFs" to get the information displayed on the lower text panel. The associated mode properties includes eigenvalue and corresponding eigenvector of the mode, and modal damping and frequency of the mode. The DOF properties include mass, stiffness, and damping of the DOF, and maximum displacement and shear during excitation.



```
Select the mode or DOF number and click See Mode or DOF.
mode1
Eigenvalue: -0.5-9.987492177719089i
Modal Damping: 0.05
Modal freq.: 10.0
Eigen Vector:
-0.005000000000000108+0.0998749217771909i
1.0

DOF 1
Mass: 100.0
Stiffness: 10000.0
Damping: 100.0
Max. Displacement:0.0
at t=0.0
Max. Shear Force:0.0
at t=0.0
```

**Figure 4.9 Mode/DOF attribute display panel**

The next function is the initial condition choice list. There are two alternative items; one is displacements and the other is velocity. When displacement is selected from the choice list, the mode is in initial displacement input. The user can input interactively by drag-and-drop the nodes displayed on the response display frame. The original color of the nodes are red, while the selected nodes are highlighted with light green. Thus the user can first select the nodes he/she wants to setup for initial displacement, then drag the highlighted nodes and drop them to desired place. The mechanism for setting up the initial velocity condition is analogous to the setup procedure for displacement condition. First, select the velocity form the initial condition choice list. Select several nodes form the response display frame, then drag and drop them to the desired place. The difference is that when drag the nodes in the velocity setup mode, red arrows show up to represent the velocity initial condition instead of directly moving the nodes.

The next function is the excitation type selecting function. Different types of earthquakes and periodic functions can be selected from the list. The excitation data is transferred from the web server to the user through the internet.

The above are the functions of the system attribute frame. The next is the structure property frame for collecting properties of the structure components. As shown in Figure 4.7, there are three input text fields corresponding to three structure component properties: mass, damping and stiffness. The mass is associated with the node of the structure and the stiffness and damping are associated with the link. Thus, the input process of the structure properties is according to the above rule. For assigning the masses to the nodes, first input the desired mass on the text field for mass. Then, click on the nodes on the response display frame, the nodes selected will turn into light green after being selected. Click on the "OK" button on the upper right corner of the system attribute frame. The masses that have been assigned values will turn into dark gray. The process is the same for the input of stiffness and damping.

The next to introduce is the response display frame. There are two functions for the response display frame. The first is the function for interactively input the properties of the structure. After the degrees of freedom is determined, the nodes and links of the structure are drawn on the frame. The user can interactively input the initial conditions or the component properties of the structure by clicking on the desired nodes and links. The second function is the dynamic display function. After the system setting is completed and the demo button is pressed, the animation of the structure response is drawn on the response display panel.

The last is the demo button frame on the lower right corner of the user interface. There are two buttons on the frame, one is for activating the forced response demo, and the other is for the free response demo. The two functions will not respond if the setting of the system have not yet finished.

The above is the functions of all the components of the user interface. The functions provided above provide a complete environment for the simulation of a general dynamic system. The following figure shows the suggested operation of Motion Lab.

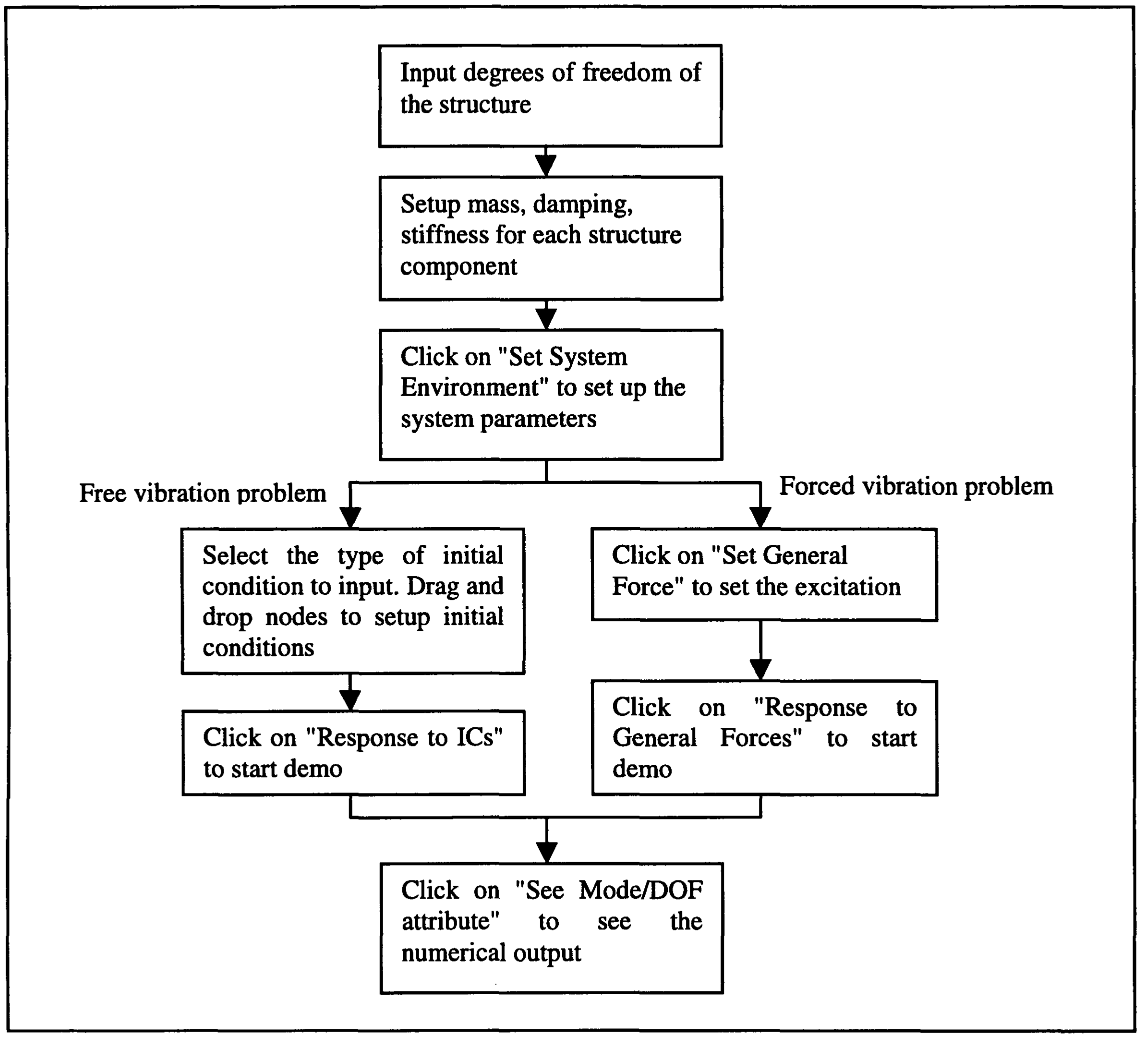

Input degrees of freedom of the structure
Setup mass, damping, stiffness for each structure component
Click on "Set System Environment" to set up the system parameters
Free vibration problem
Forced vibration problem
Select the type of initial condition to input. Drag and drop nodes to setup initial conditions
Click on "Response to ICs" to start demo
Click on "Set General Force" to set the excitation
Click on "Response to General Forces" to start demo
Click on "See Mode/DOF attribute" to see the numerical output

**Figure 4.10 Suggested operation of Motion Lab**

Notice that the procedure presented in Figure 4.10 is not the absolute procedure for the system. The sequence of setting up structure component, system attribute and forcing/initial conditions can be changed, but the optimal procedure is still the procedure suggested in Figure 4.10 for the consideration of optimization in computing.

# Chapter 5

# Testing Examples

## 5.1 Testing Example for Single-Degree-of-Freedom Structure

The first example is a single-degree-of-freedom structure. As shown in Figure 5.1, the structure can be transformed to an equivalent system on the right hand side of the figure. The mass is 100kg, the lateral stiffness is 5000 N/m, and the damping is 100 N·s/m.

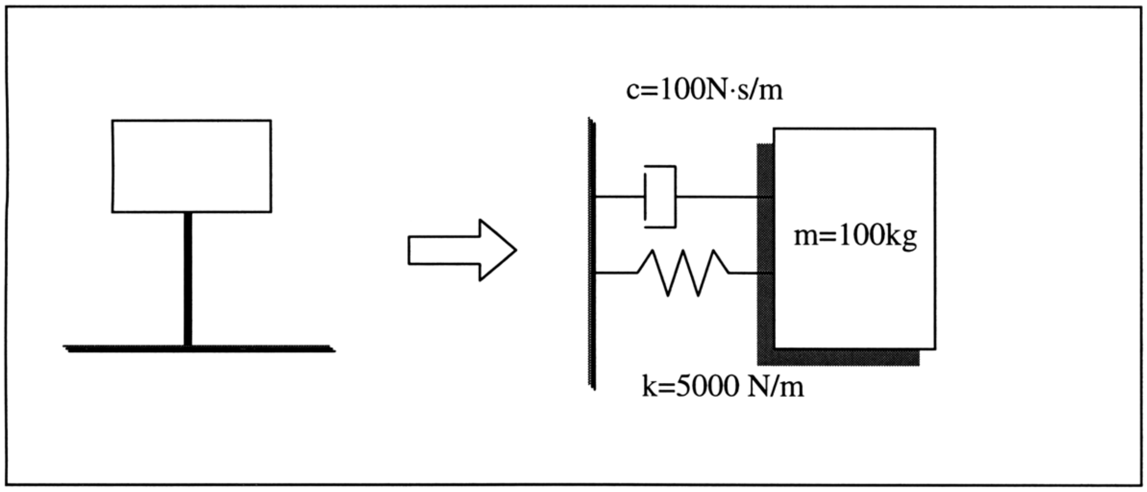


**Figure 5.1 The single-degree-of-freedom structure**

The natural frequency and the damping ratio are obtained in Equation 5.1a and 5.1b. According to Equation 2.2 to 2.5, the matrices represent the state-space system is formed in Equation 5.2a and 5.2b.

$$\omega_n = \sqrt{\frac{k}{m}} = \sqrt{\frac{5000}{100}} = 7.071068 \; rad/s \tag{5.1a}$$

$$\xi = \frac{c}{2m\omega_n} = \frac{100}{2\cdot 100\cdot 7.071068} = 0.0707 \tag{5.1b}$$

$$A = \begin{bmatrix} 0 & 1 \\ -k/m & -c/m \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -50 & -1 \end{bmatrix} \tag{5.2a}$$

$$X = \begin{bmatrix} u \\ \dot{u} \end{bmatrix} \tag{5.2b}$$

The next procedure is to solve for the eigenproblem for the state-space system. Feeding matrix A into Matlab, the eigenvector and eigenvalue matrices **V** and Λ are obtained in Equation 5.3a and 5.3b.

$$\mathbf{V} = \begin{bmatrix} 0.1397 - 0.0099i & 0.1397 + 0.0099i \\ 0.9901i & -0.9901i \end{bmatrix} \tag{5.3a}$$

$$\Lambda = \begin{bmatrix} -0.5 + 7.0534i & 0 \\ 0 & -0.5 - 7.0534i \end{bmatrix} \tag{5.3b}$$

Next, apply the north-south component of El Centro earthquake shown in Figure 5.2 to the single-degree-of-freedom structure and calculate its response. The response is obtained by the convolution of the impulse response of the system and the earthquake force.

The convolution process is written in a Matlab script. The result of the convolution is shown in Figure 5.3. The top frame of the figure shows the impulse response of the structure system, the center frame shows the earthquake force in N, and the bottom figure shows the response of the system. In addition, from the output of the Matlab script, the maximum response is 0.887m at t=5.94sec.

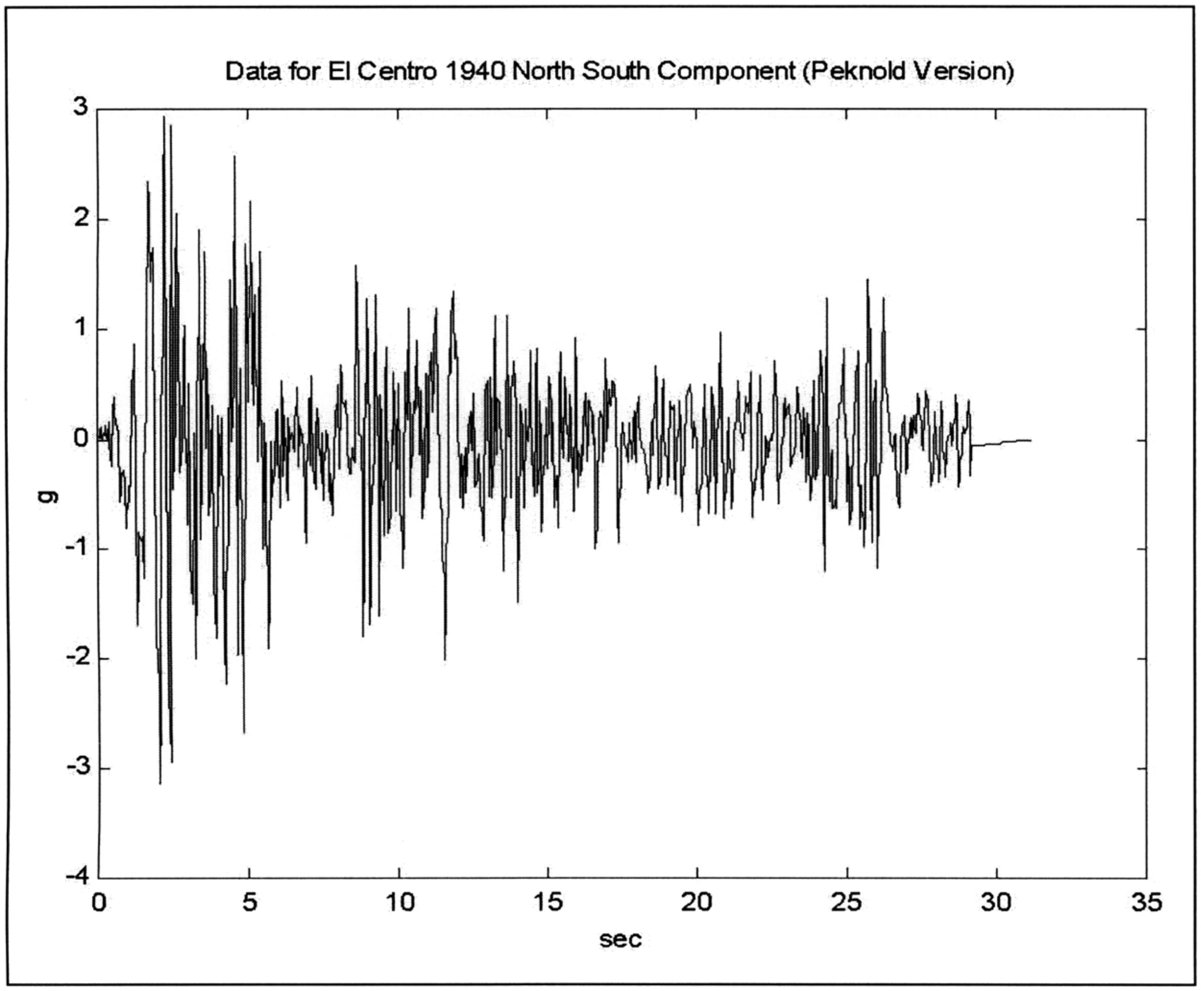


**Figure 5.2 North-south component of El Centro earthquake**

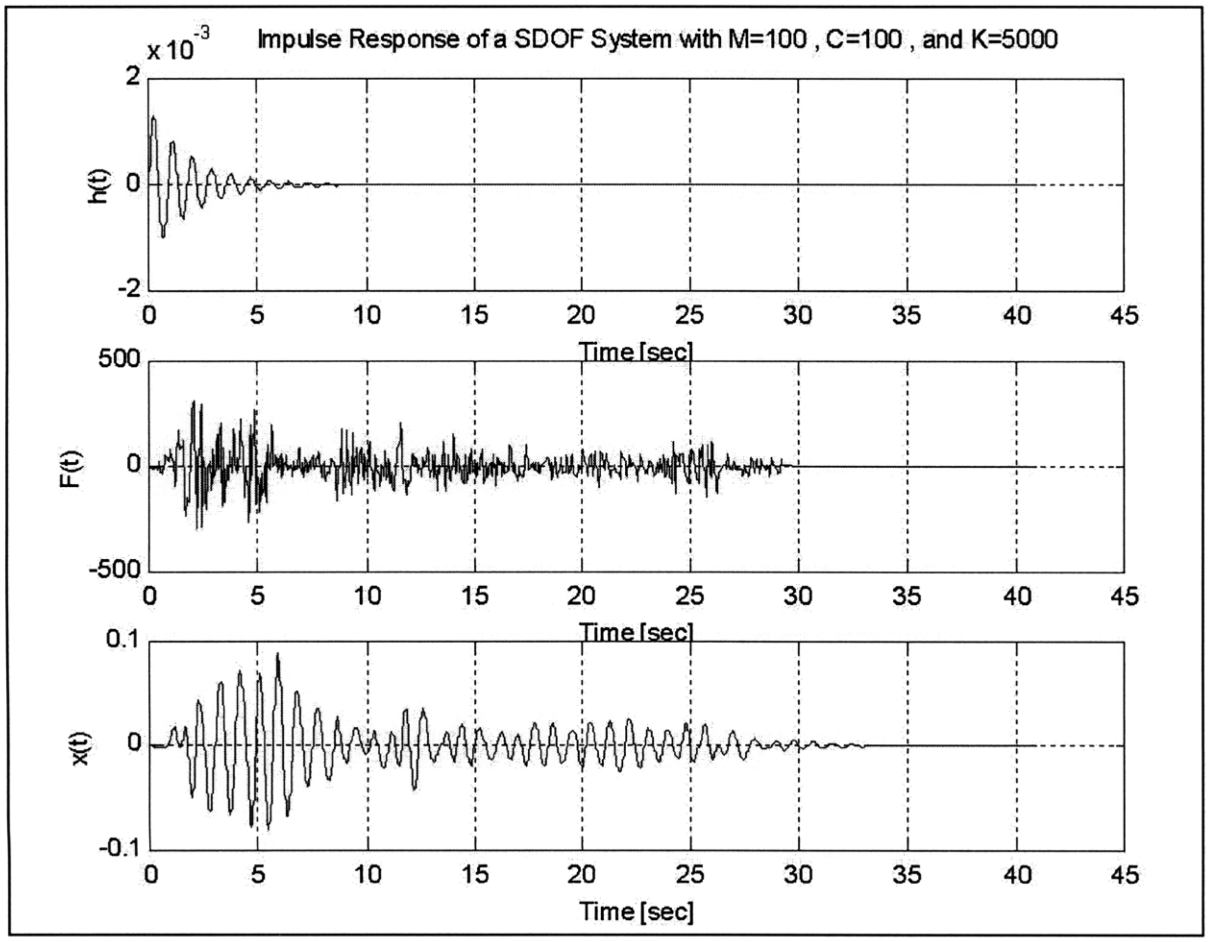


**Figure 5.3 Response plot of El Centro earthquake**

Next, do test on the Motion Lab using the same structure properties and setting. The input process follows the procedure provided in Section 4.3. First, setup the degrees of freedom of the structure. The response display frame comes out the structure shown in Figure 5.4. Then, select all the structure components and setup using the same mass, damping and stiffness as in the above example. Then, setup the system attributes as shown in Figure 5.5. Select El Centro earthquake as the excitation in Figure 5.6. Then, press the "Response to General Forces" and the demo comes out immediately.

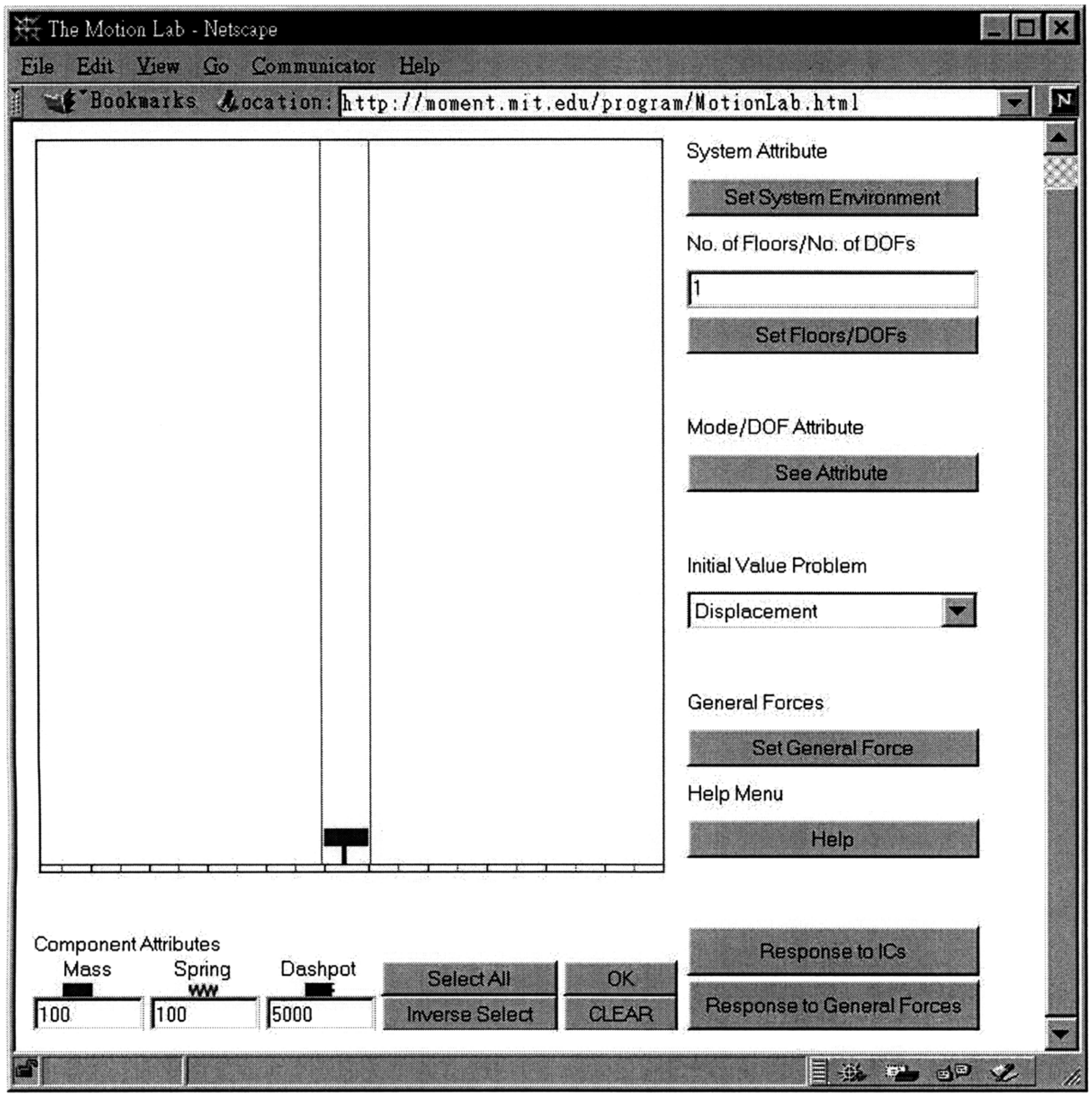
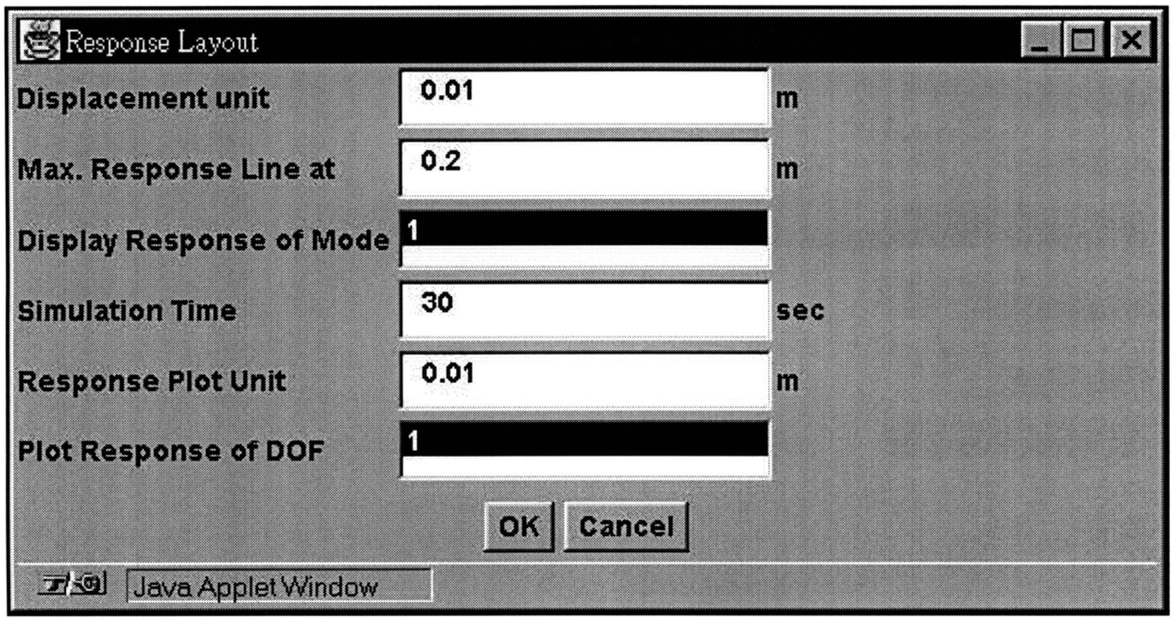


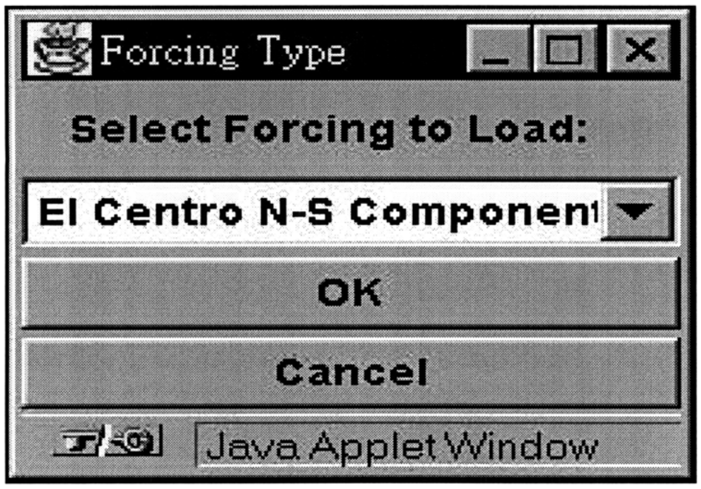
**Figure 5.4 Setting up degrees of freedom of the structure**



Response Layout

| | | |
|---|---|---|
| Displacement unit | 0.01 | m |
| Max. Response Line at | 0.2 | m |
| Display Response of Mode | 1 | |
| Simulation Time | 30 | sec |
| Response Plot Unit | 0.01 | m |
| Plot Response of DOF | 1 | |

**Figure 5.5 System attribute setting**



Forcing Type

Select Forcing to Load: El Centro N-S Componen

**Figure 5.6 Select El Centro as the Excitation**

The response plot is in Figure 5.7 and a time step of the demo is in Figure 5.8. Press "See Mode/DOF attribute shown in Figure 5.9 to dump the numerical result, and verify with the result obtained from convolution integral.

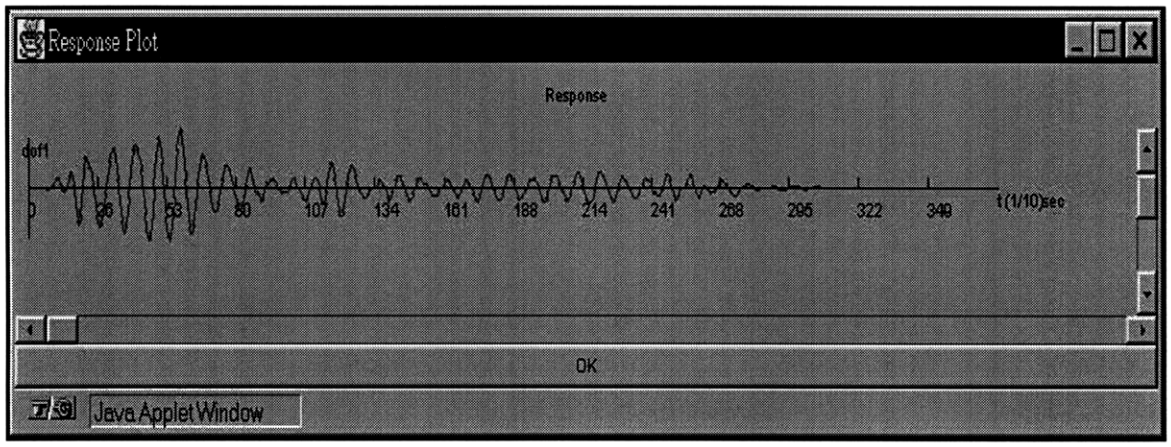


Figure 5.7 Response plot from Motion Lab

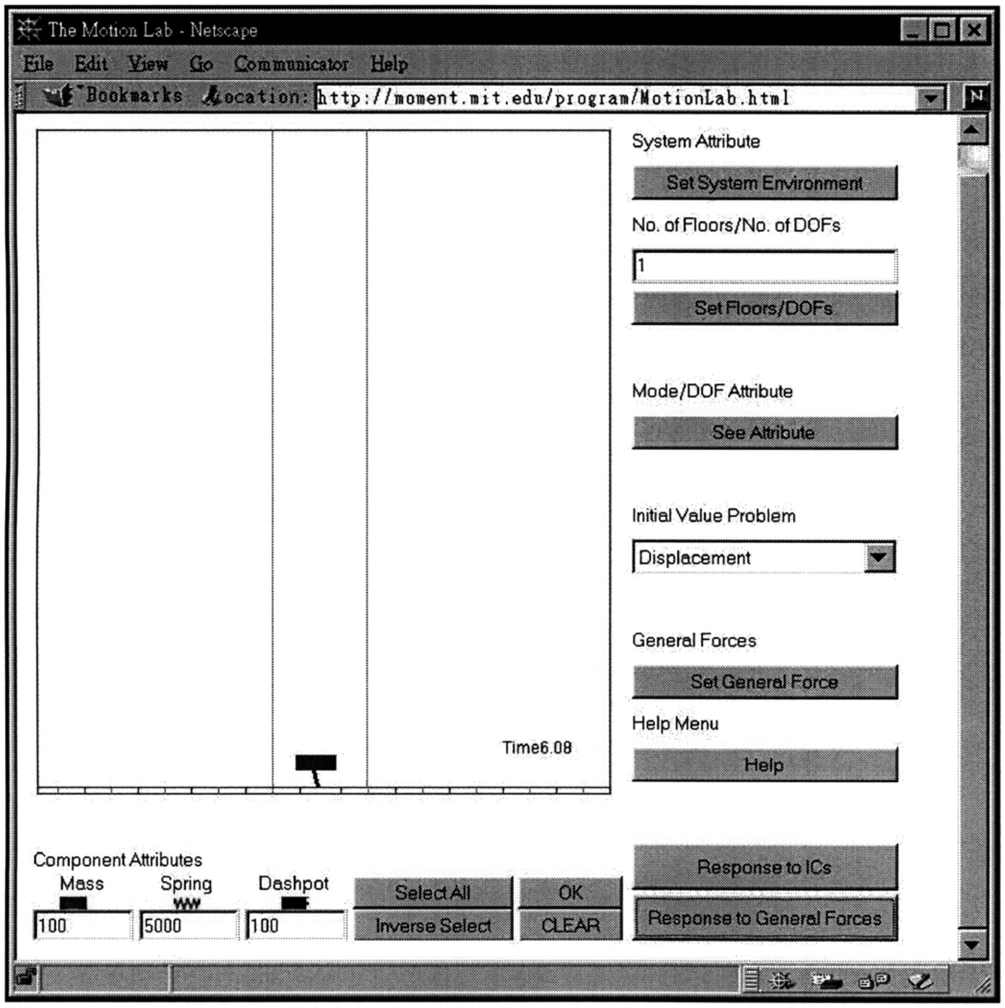
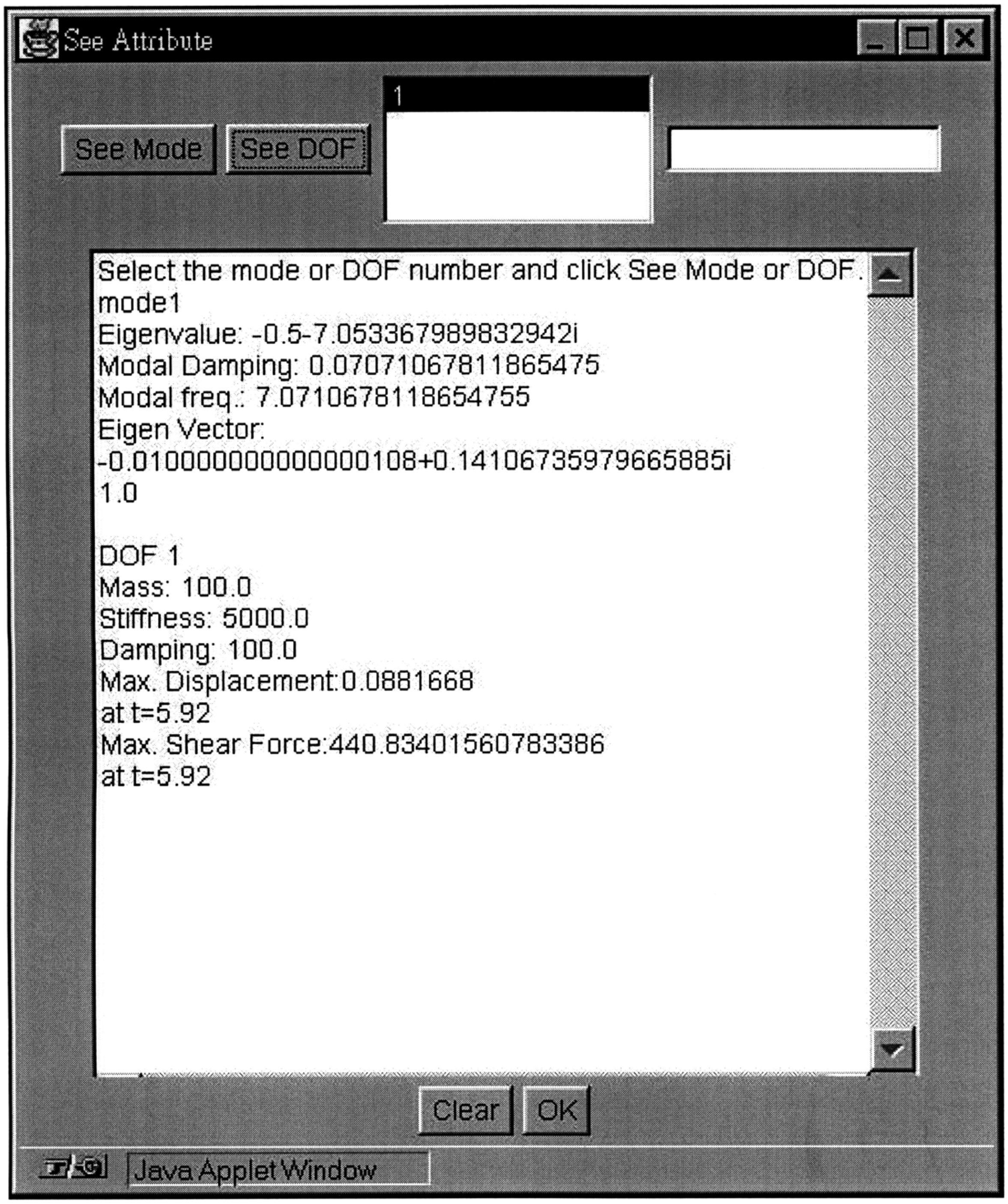


Figure 5.8 Response demo at t=6.08sec



```
Select the mode or DOF number and click See Mode or DOF.
mode1
Eigenvalue: -0.5-7.053367989832942i
Modal Damping: 0.07071067811865475
Modal freq.: 7.0710678118654755
Eigen Vector:
-0.01000000000000108+0.14106735979665885i
1.0

DOF 1
Mass: 100.0
Stiffness: 5000.0
Damping: 100.0
Max. Displacement:0.0881668
at t=5.92
Max. Shear Force:440.83401560783386
at t=5.92
```

**Figure 5.9 Numerical Mode/DOF attribute form Motion Lab**

The comparison is made for the convolution integral and the result form Motion Lab. As shown in Table 5.1, except for 0.5% difference in maximum response, the other results are the same.

| | $\omega$ | $\xi$ | Eigenvalue | Eigenvector | Max. Displacement Dmax | t at Dmax |
|---|---|---|---|---|---|---|
| Result from con-volution | 7.071 rad/s | 7.07% | -0.5-0.70534i | [0.1397+ 0.0099i; -0.9901i ] | 0.0887m | 5.92sec |
| Result form Motion Lab | 7.071 rad/s | 7.07% | -0.5-0.70534i | [-0.01+ 0.14107i; 1.0] | 0.0882m | 5.92sec |
| Error (%) | 0 | 0 | 0 | 0 | 0.5% | 0 |

**Table 5.1 Comparison between the results from convolution integral and Motion Lab**

Notice the normalization of the two eigenvectors in the table are different. Same result is obtained if the first eigenvector is normalized by dividing its second row.

## 5.2 Testing Example for Five-Degree-of-Freedom Structure

The following testing example is a five-degree-of-freedom shear structure. The properties of the components are divided into two layers thus make the structure damping non-proportional.

Figure 5.10 shows the mass, stiffness and damping properties of the structure. The five lumped masses of the structure are 200kg. The lateral damping and stiffness parameters are formed by two layers. In the lower two degrees of freedom, the lateral stiffness and damping is 8000N/m and 100N·s/m. In the upper three degrees of freedom, the lateral stiffness and damping is 10000N/m and 300N·s/m.

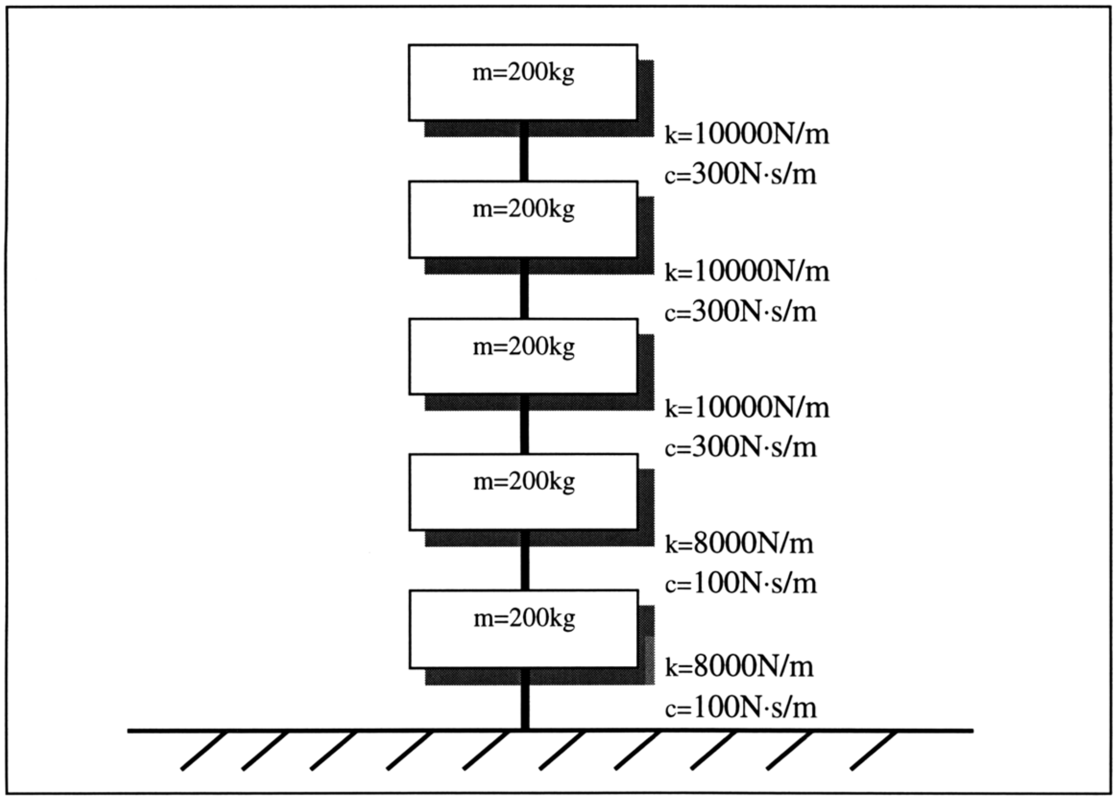


**Figure 5.10 Properties of the five-degree-of-freedom structure**

The mass, damping and stiffness matrices are presented in Equation 5.4a, b and c. The eigenvalue and eigenvector matrices of the corresponding state-space system are presented in Equation 5.5a to 5.5f.

$$\mathbf{M}=\begin{bmatrix} 200 & & & & \\ & 200 & & & \\ & & 200 & & \\ & & & 200 & \\ & & & & 200 \end{bmatrix} \qquad (5.4a)$$

$$\mathbf{C}=\begin{bmatrix} 200 & -100 & & & \\ -100 & 400 & -300 & & \\ & -300 & 600 & -300 & \\ & & -300 & 600 & -300 \\ & & & -300 & 300 \end{bmatrix} \qquad (5.4b)$$

$$\mathbf{K}=\begin{bmatrix} 16000 & -8000 & & & \\ -8000 & 18000 & -10000 & & \\ & -10000 & 20000 & -10000 & \\ & & -10000 & 20000 & -10000 \\ & & & -10000 & 20000 \end{bmatrix} \qquad (5.4c)$$

$$\mathbf{V}_1=\begin{bmatrix} -0.0002+0.0122i \\ 0.0123-0.0290i \\ -0.0270+0.0402i \\ 0.0263-0.0340i \\ -0.0108+0.0132i \\ -0.1584-0.0338i \\ 0.3470+0.2335i \\ -0.4568-0.4547i \\ 0.3774+0.4290i \\ -0.1452-0.1738i \end{bmatrix}; \quad \mathbf{V}_2=\tilde{\mathbf{V}}_1 \qquad (5.5a)$$

$$\mathbf{V}_3 = \begin{bmatrix} -0.0354 + 0.0237i \\ 0.0502 - 0.0152i \\ -0.0121 - 0.0098i \\ -0.0422 + 0.0175i \\ 0.0293 - 0.0076i \\ -0.2143 - 0.4294i \\ 0.0981 + 0.5829i \\ 0.1263 - 0.1211i \\ -0.1354 - 0.4969i \\ 0.0436 + 0.3386i \end{bmatrix}; \quad \mathbf{V}_4 = \tilde{\mathbf{V}}_3 \tag{5.5b}$$

$$\mathbf{V}_5 = \begin{bmatrix} -0.0748 + 0.0113i \\ 0.0015 + 0.0125i \\ 0.0573 - 0.0154i \\ 0.0246 - 0.0125i \\ -0.0454 + 0.0128i \\ -0.0353 - 0.6744i \\ -0.1125 + 0.0024i \\ 0.0868 + 0.5225i \\ 0.0900 + 0.2297i \\ -0.0746 - 0.4146i \end{bmatrix}; \quad \mathbf{V}_6 = \tilde{\mathbf{V}}_5 \tag{5.5c}$$

$$\mathbf{V}_7 = \begin{bmatrix} -0.0112 - 0.0838i \\ -0.0180 - 0.0995i \\ -0.0112 - 0.0480i \\ 0.0038 + 0.0342i \\ 0.0156 + 0.0944i \\ 0.4785 - 0.0299i \\ 0.5704 - 0.0624i \\ 0.2759 - 0.0442i \\ -0.1947 + 0.0077i \\ -0.5406 + 0.0509i \end{bmatrix}; \quad \mathbf{V}_8 = \tilde{\mathbf{V}}_7 \tag{5.5d}$$

$$\mathbf{V}_9 = \begin{bmatrix} 0.0781 - 0.0385i \\ 0.1493 - 0.0737i \\ 0.1951 - 0.0983i \\ 0.2275 - 0.1158i \\ 0.2442 - 0.1249i \\ 0.0694 + 0.1467i \\ 0.1328 + 0.2806i \\ 0.1773 + 0.3667i \\ 0.2089 + 0.4275i \\ 0.2253 + 0.4591i \end{bmatrix}; \quad \mathbf{V}_{10} = \tilde{\mathbf{V}}_9 \tag{5.5e}$$

$$\begin{aligned} \Lambda = diag\,[\, & -2.5273 + 13.0533i, \quad -2.5273 - 13.0533i, \quad -1.4266 + 11.1751i, \\ & -1.4266 - 11.1751i, \quad -0.8694 + 8.8851i, \quad -0.8694 - 8.8851i, \\ & -0.3963 + 5.6586i, \quad -0.3963 - 5.6586i, \quad -0.0304 + 1.8642i, \\ & -0.0304 - 1.8642i \,] \end{aligned} \tag{5.5f}$$

To verify the dynamic response to earthquakes, a MATLAB script based on direct integration method is written. El Centro earthquake is read in the program as an excitation to the five-degree-of freedom structure. The response plot is in Figure 5.11.

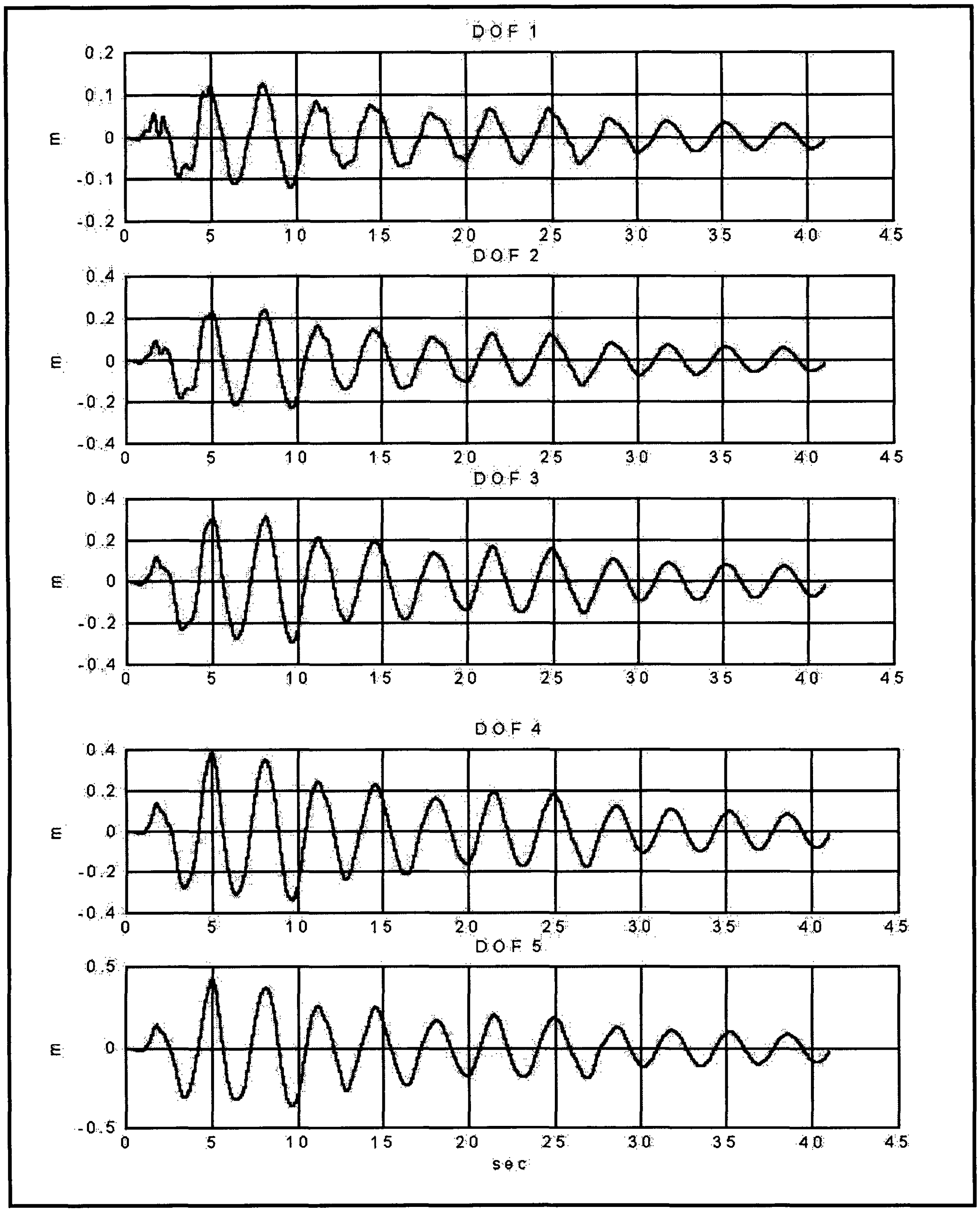
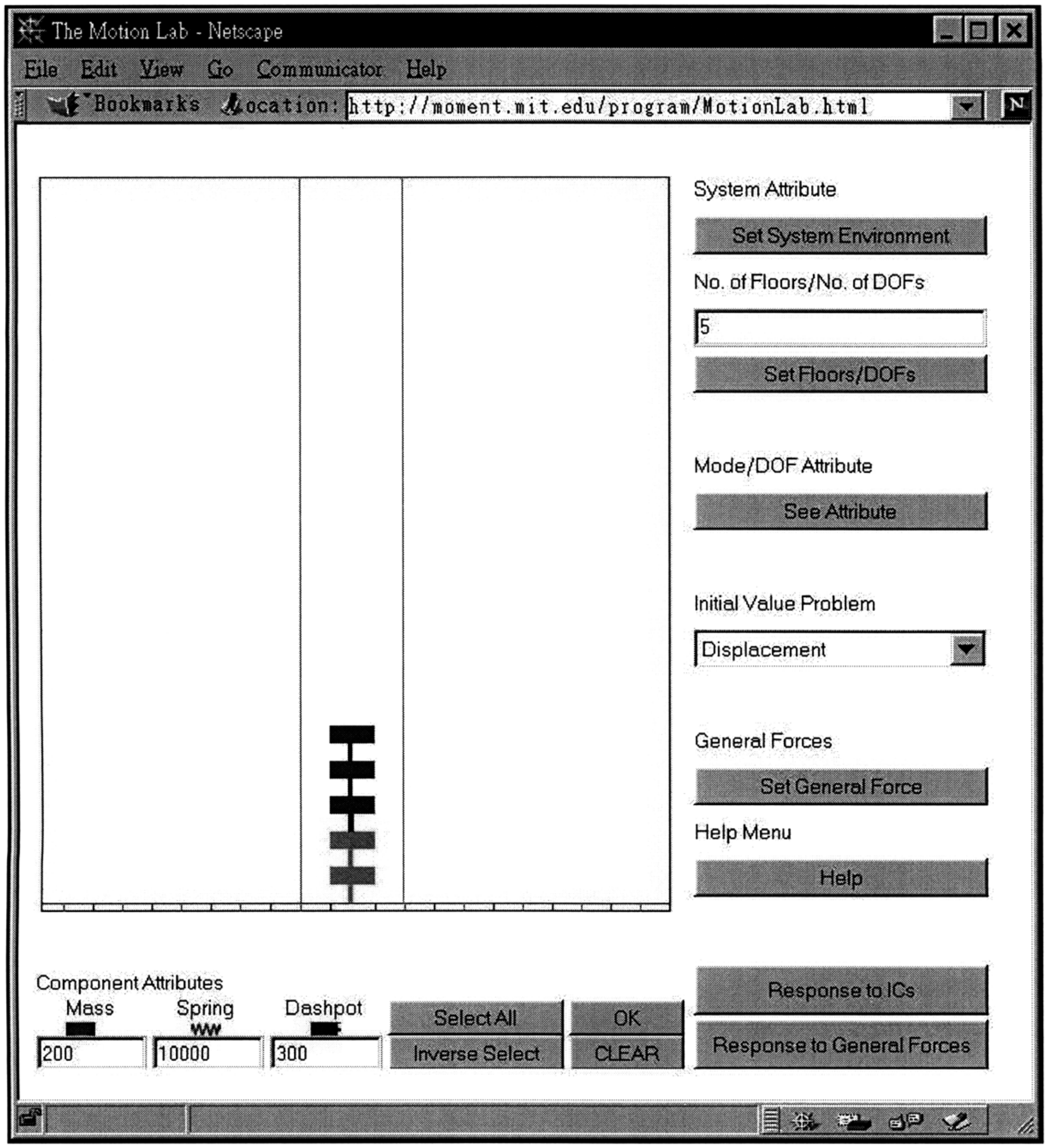


**Figure 5.11 El Centro response for the five-degree-of-freedom structure**

Next, the structure is tested by Motion Lab. Setup the structure in two different property layers shown in Figure 5.12.



No. of Floors/No. of DOFs: 5

Initial Value Problem: Displacement

| Mass | Spring | Dashpot |
|---|---|---|
| 200 | 10000 | 300 |

**Figure 5.12 Setting up the five-degree-of-freedom structure**

After the structure properties is setup, select El Centro as the excitation. The response plot shown in Figure 5.13 is created after the demo starts. As can be observed from the demo, the response from the first mode dominates the response of the structure.

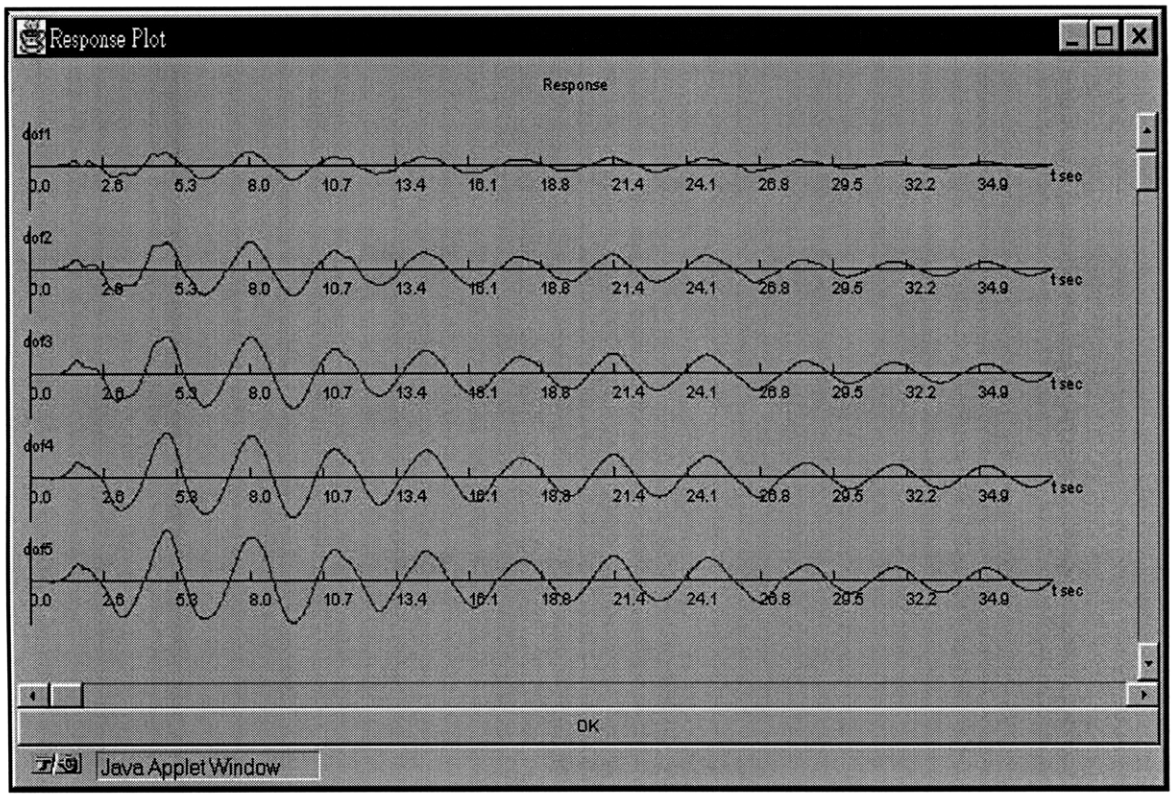
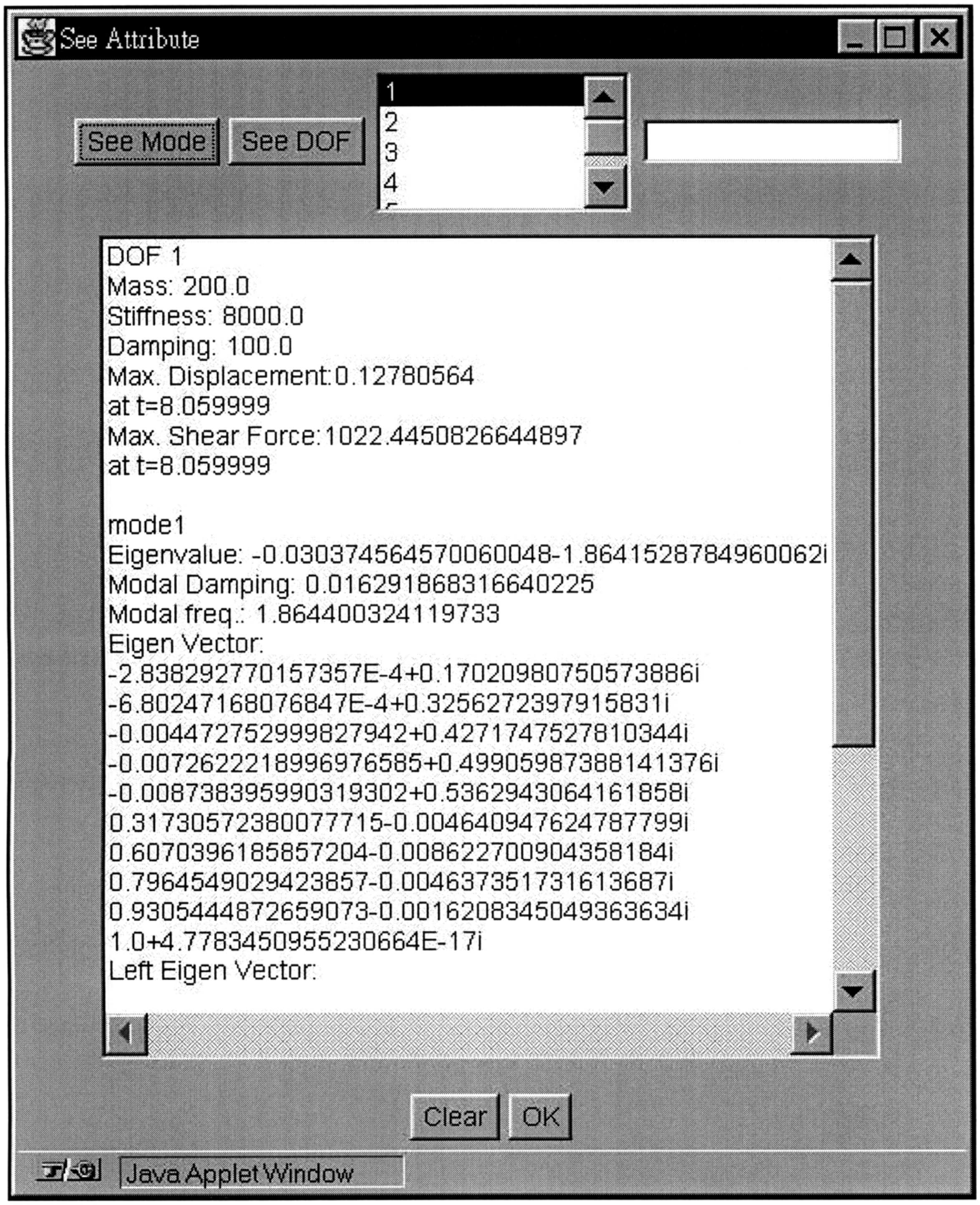


**Figure 5.13 Response plot of the five-degree-of-freedom structure**

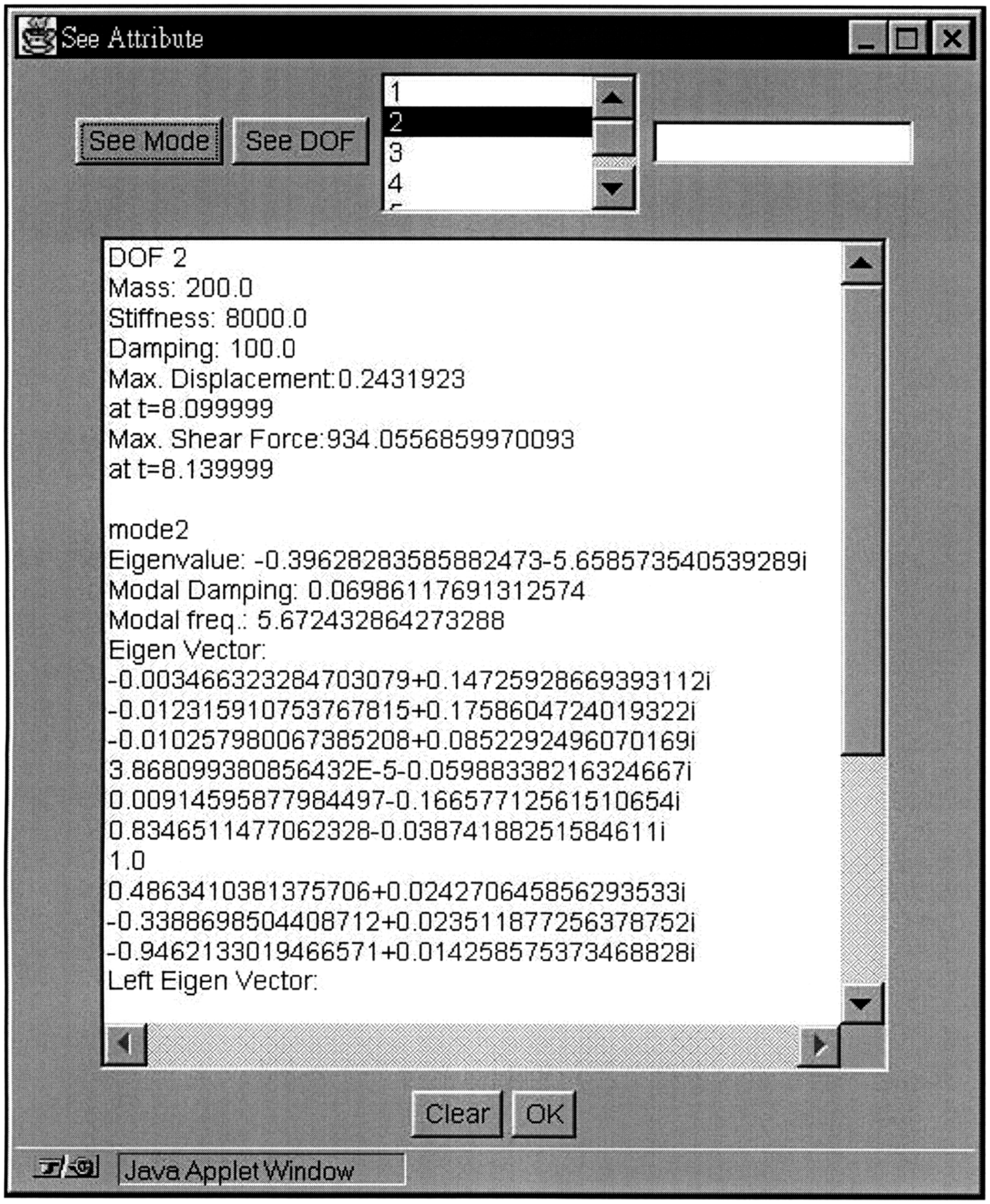
The numerical output of the structure is obtained from the Mode/DOF display dialog box shown in Figure 5.14 to Figure 5.18.



```
DOF 1
Mass: 200.0
Stiffness: 8000.0
Damping: 100.0
Max. Displacement:0.12780564
at t=8.059999
Max. Shear Force:1022.4450826644897
at t=8.059999

mode1
Eigenvalue: -0.030374564570060048-1.8641528784960062i
Modal Damping: 0.016291868316640225
Modal freq.: 1.864400324119733
Eigen Vector:
-2.838292770157357E-4+0.17020980750573886i
-6.80247168076847E-4+0.3256272397915831i
-0.004472752999827942+0.4271747527810344i
-0.0072622218996976585+0.49905987388141376i
-0.008738395990319302+0.5362943064161858i
0.31730572380077715-0.004640947624787799i
0.6070396185857204-0.008622700904358184i
0.7964549029423857-0.004637351731613687i
0.9305444872659073-0.0016208345049363634i
1.0+4.7783450955230664E-17i
Left Eigen Vector:
```

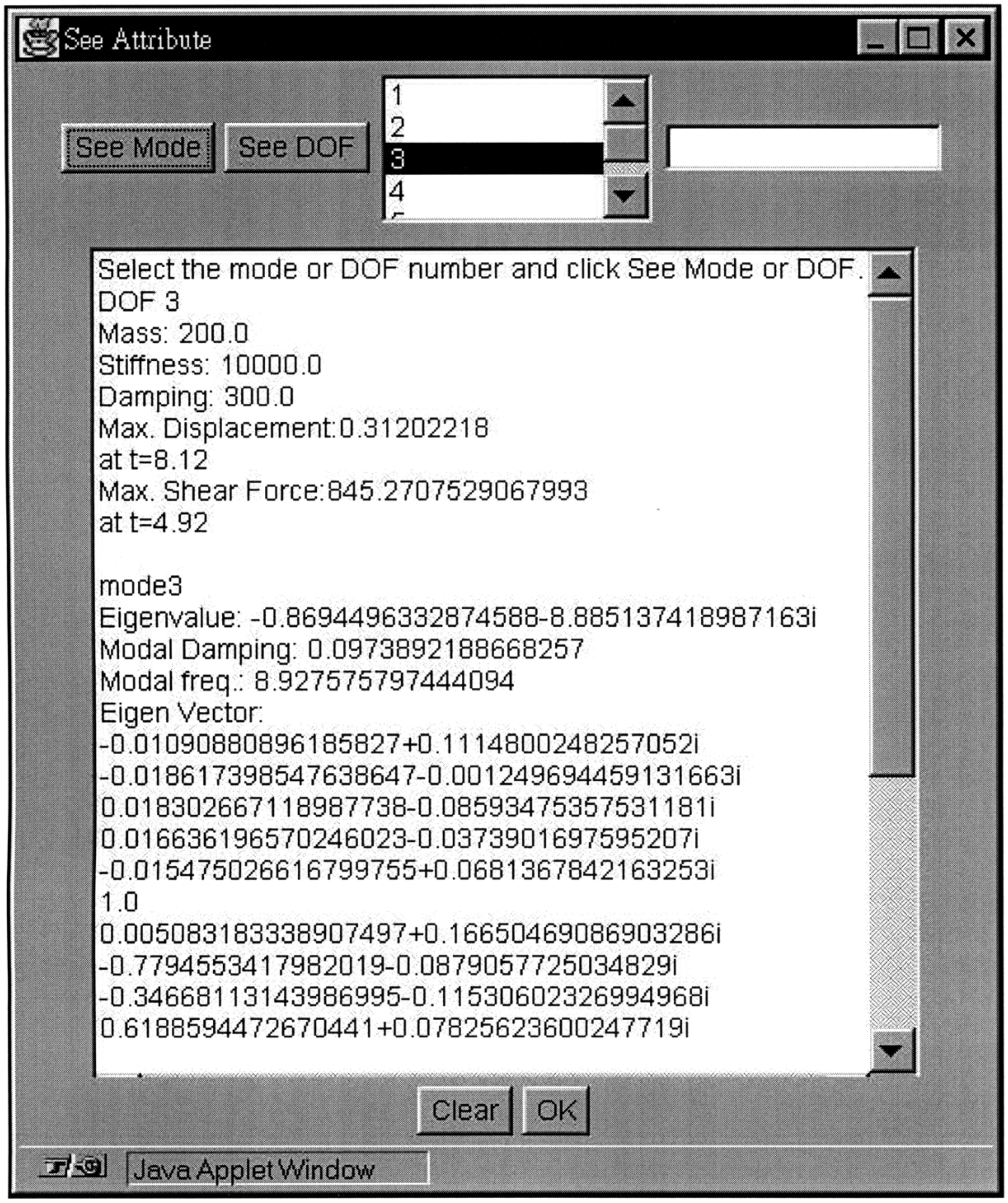
Figure 5.14 Numerical attribute of the first mode and DOF



```
DOF 2
Mass: 200.0
Stiffness: 8000.0
Damping: 100.0
Max. Displacement:0.2431923
at t=8.099999
Max. Shear Force:934.0556859970093
at t=8.139999

mode2
Eigenvalue: -0.3962828358588247З-5.658573540539289i
Modal Damping: 0.06986117691312574
Modal freq.: 5.672432864273288
Eigen Vector:
-0.003466323284703079+0.14725928669393112i
-0.012315910753767815+0.1758604724019322i
-0.010257980067385208+0.0852292496070169i
3.868099380856432E-5-0.05988338216324667i
0.00914595877984497-0.16657712561510654i
0.8346511477062328-0.03874188251584611i
1.0
0.4863410381375706+0.024270645856293533i
-0.3388698504408712+0.023511877256378752i
-0.9462133019466571+0.014258575373468828i
Left Eigen Vector:
```

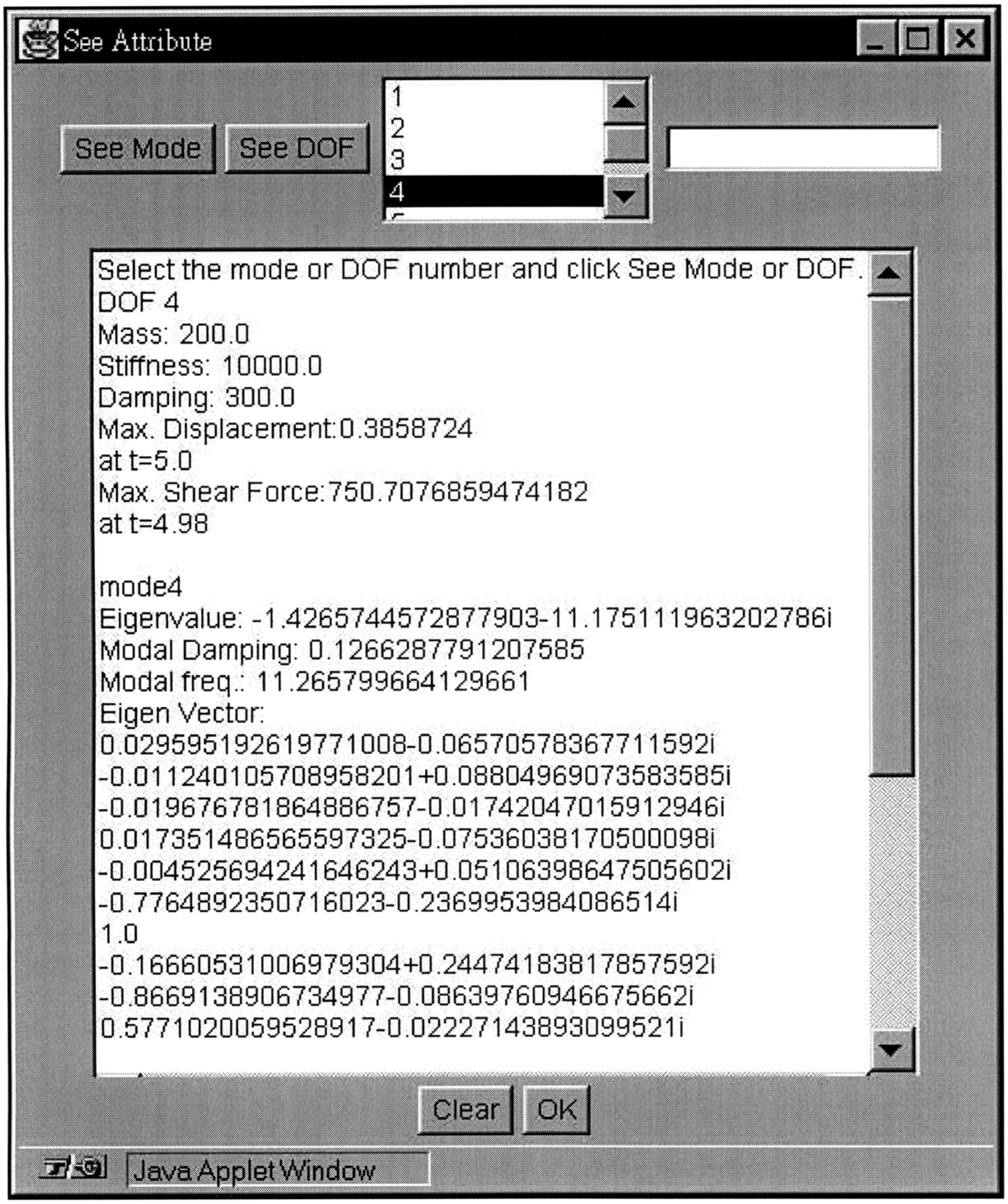
**Figure 5.15 Numerical attribute of the second mode and DOF**



```
Select the mode or DOF number and click See Mode or DOF.
DOF 3
Mass: 200.0
Stiffness: 10000.0
Damping: 300.0
Max. Displacement:0.31202218
at t=8.12
Max. Shear Force:845.2707529067993
at t=4.92

mode3
Eigenvalue: -0.8694496332874588-8.885137418987163i
Modal Damping: 0.0973892188668257
Modal freq.: 8.927575797444094
Eigen Vector:
-0.010908808961858 27+0.1114800248257052i
-0.018617398547638647-0.001249694459131663i
0.018302667118987738-0.085934753575311811i
0.016636196570246023-0.0373901697595207i
-0.015475026616799755+0.0681367842163253i
1.0
0.005083183338907497+0.16650469086903286i
-0.7794553417982019-0.0879057725034829i
-0.34668113143986995-0.11530602326994968i
0.6188594472670441+0.07825623600247719i
```

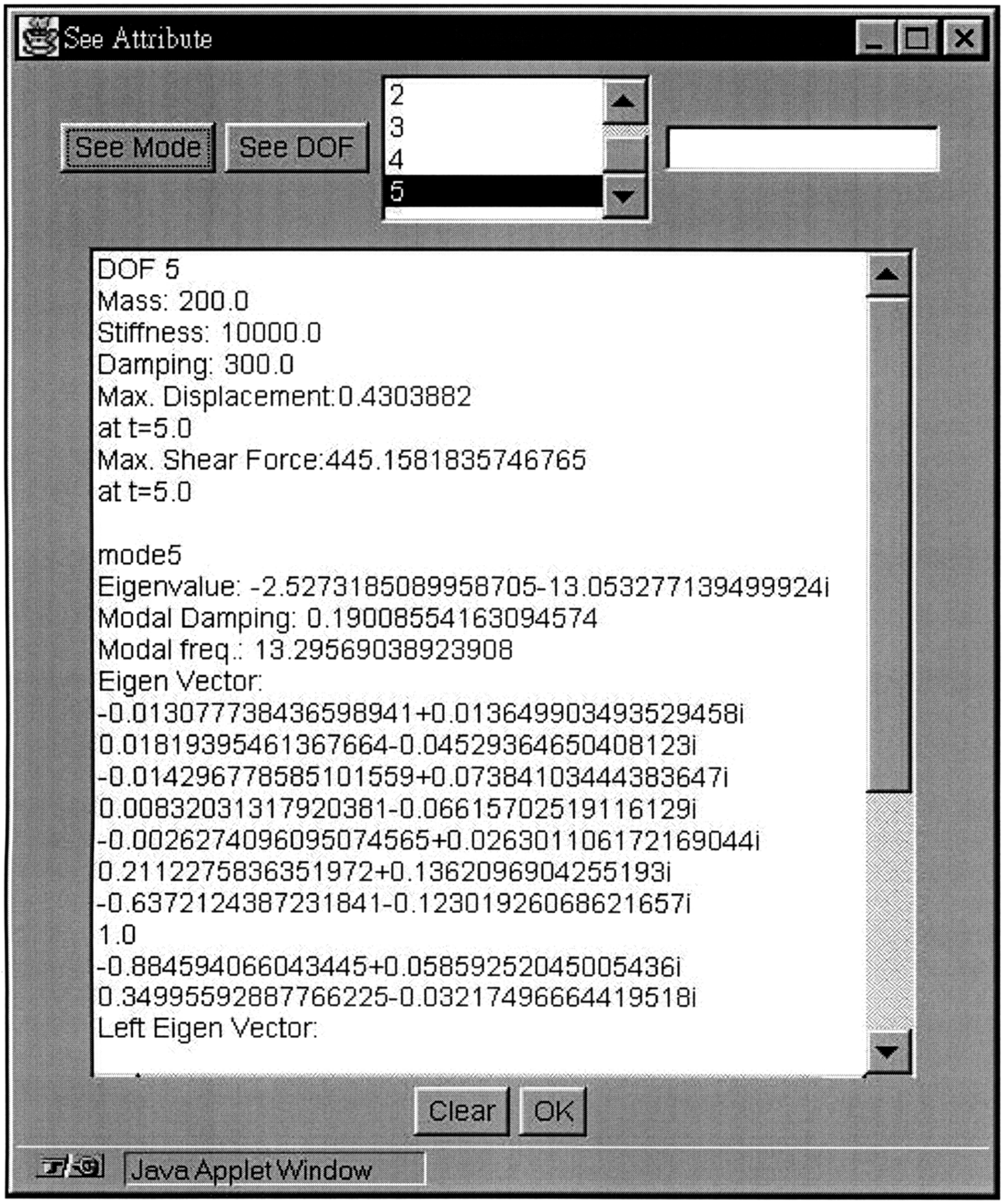
**Figure 5.16 Numerical attribute of the third mode and DOF**



```
Select the mode or DOF number and click See Mode or DOF.
DOF 4
Mass: 200.0
Stiffness: 10000.0
Damping: 300.0
Max. Displacement:0.3858724
at t=5.0
Max. Shear Force:750.7076859474182
at t=4.98

mode4
Eigenvalue: -1.4265744572877903-11.175111963202786i
Modal Damping: 0.1266287791207585
Modal freq.: 11.265799664129661
Eigen Vector:
0.029595192619771008-0.06570578367711592i
-0.011240105708958201+0.08804969073583585i
-0.019676781864886757-0.01742047015912946i
0.017351486565597325-0.07536038170500098i
-0.004525694241646243+0.05106398647505602i
-0.7764892350716023-0.2369953984086514i
1.0
-0.16660531006979304+0.24474183817857592i
-0.8669138906734977-0.08639760946675662i
0.5771020059528917-0.02227143893099521i
```

**Figure 5.17 Numerical attribute of the fourth mode and DOF**



```
DOF 5
Mass: 200.0
Stiffness: 10000.0
Damping: 300.0
Max. Displacement:0.4303882
at t=5.0
Max. Shear Force:445.1581835746765
at t=5.0

mode5
Eigenvalue: -2.5273185089958705-13.053277139499924i
Modal Damping: 0.19008554163094574
Modal freq.: 13.29569038923908
Eigen Vector:
-0.013077738436598941+0.013649903493529458i
0.01819395461367664-0.04529364650408123i
-0.014296778585101559+0.07384103444383647i
0.00832031317920381-0.06615702519116129i
-0.002627409609507456+0.026301106172169044i
0.2112275836351972+0.1362096904255193i
-0.6372124387231841-0.12301926068621657i
1.0
-0.884594066043445+0.05859252045005436i
0.34995592887766225-0.03217496664419518i
Left Eigen Vector:
```

**Figure 5.18 Numerical attribute of the fifth mode and DOF**

The modal properties from both MATLAB and Motion Lab are the same. Table 5.2 shows the comparison of the maximum displacements obtained from the two approaches.

| | DOF 1 | DOF 2 | DOF 3 | DOF 4 | DOF 5 |
|---|---|---|---|---|---|
| From Direct Integration | 0.1279m | 0.2434m | 0.3123m | 0.3860m | 0.4306m |
| From Motion Lab | 0.1278m | 0.2432m | 0.3120m | 0.3859m | 0.4304m |
| Error % | 0% | 0% | 0% | 0% | 0% |

**Table 5.2 Comparison between results from direct integration and Motion Lab**

The result shows the difference between the two approaches is within $10^{-3}$. Thus Motion Lab meets the accuracy requirement of the system.

# Chapter 6

# Conclusion and Future Work

## 6.1 Conclusion

With the great improvement in personal computing capability, the level of engineering computation tool has moved from hand calculation to high performance computing. However, the methodology for the engineering computation does not move on with the speed of evolution in computing capability. The computations perform on personal computers are similar to what engineers did on calculators because the computers are treated as calculators with better computing power.

On the other hand, this thesis presents a different approach to the engineering computation. Friendly user interface supported by optimized numerical procedures, the tool demonstrates a new standard for engineering computation. It shows the ability of immediate response with a good accuracy. In addition, it handles the seismic response of a structure with arbitrary properties by modal analysis. User can actually "feel" the modal properties provided by the dynamic simulation, instead of dealing with meaningless complex numbers.

## 6.2 Future Work

Future work can be proceeded in two areas. The first area is to extend the ability of the environment to the structure control problem. With the control forces built in the tool, active structural control can be simulated.

The second area is the automatic seismic design of structures. Motion based design methodology [12] can be introduced for design. Optimal design can be obtained by iterating on the response of the structure subject to constraints on several response parameters and/or material attributes.

# Bibliography

[1] Foss,K. A., *Coordinates which Uncouple the Equations of Motion of Damped Linear Dynamic Systems,* Technical Report, MIT Aeroelastic and Structures Research Laboratory, 1956.

[2] Mendoza Zabala, J. L., *State-space formulation for structure dynamics*, Master Thesis, MIT, 1996.

[3] Meirovitch, L., *Elements of Vibration Analysis*, McGraw Hill, 1986.

[4] Miller, G. and Cooper, S. C., *Visual Mechanics - Beams and Stress States*, PWS Publishing, 1998.

[5] Chow, T. Y., *Educational Structural Analysis Using cT*, Bachelor Thesis, MIT, 1994.

[6] Press, W. H., *Numerical Recipes in C: The Art of Scientific Computing* $2^{nd}$ *Ed*, Cambridge University Press, 1993.

[7] Clough, R. W., *Dynamics of Structures*, McGraw Hill, 1993.

[8] Wilkinson, J. H., *Linear Algebra,vol.II of Handbook for Automatic Computation*, Springer-Verlag, 1971.

[9] Oppenheim, A. V., *Discrete-time Signal Processing*, Prentice Hall, 1989.

[10] Rumbaugh, J., *Object Oriented Modeling and Design*, Prentice Hall, 1991.

[11] Smith, B.T., et al., *Matrix Eigensystem Routines — EISPACK Guide, 2nd ed., vol. 6 of Lecture Notes in Computer Science*, Springer-Verlag, 1976.

[12] Connor, J. J. and Klink, B. S., *Introduction to Motion Based Design*, Computational Mechanics Publications, 1996.