COMPUTER-AIDED ENGINEERING METHODOLOGY FOR STRUCTURAL OPTIMIZATION AND CONTROL

By

YI-MEI MARIA CHOW

Master of Engineering in Civil and Environmental Engineering
at the
Massachusetts Institute of Technology
June 1998

Submitted to the Department of Civil and Environmental Engineering
In Partial Fulfillment of the Requirement for the Degree of

Civil Engineer's Degree
at the
Massachusetts Institute of Technology

February 2000

Signature of Author

Yi-Mei Maria Chow
September 15, 1999

Certified by

Jerome J. Connor
Professor, Department of Civil and Environmental Engineering
Thesis Supervisor

Approved by

Daniele Veneziano,
Chairman, Departmental Committee on Graduate Studies.

COMPUTER-AIDED ENGINEERING METHODOLOGY FOR STRUCTURAL OPTIMIZATION AND CONTROL

By

YI-MEI MARIA CHOW

Submitted to the Department of Civil and Environmental Engineering on
September 15 1999
In Partial Fulfillment of the Requirement for the Degree of
Civil Engineer's Degree

ABSTRACT

The evolution of computer-aided engineering methodology has played a major role in the advancement of traditional structural engineering. It has allowed engineers to obtain better solutions for design and analysis problems by means of diverse user-friendly software packages. However, as the precision and safety requirements have increased in the ever-changing world, civil and structural engineers face numberless unsolved issues with respect to structural optimization and control.

Matlab is a very useful CAE tool for numerical analysis. In this dissertation, the computational algorithms for several vibration control systems are implemented with Matlab and used for diverse simulation studies. Matlab has many embedded tools which simplify matrix operations encountered in the vibration control. Furthermore, newly developed approaches such as evolution strategy, genetic algorithm, finite element, and neural networks form the paradigm of software elaboration in structural engineering field. Those modern numerical CAE methodologies, in addition to analytical solutions, enable further structural optimization and topological modification accomplished by computer simulation.

With the progression in programming languages and software packages, computer-aided engineering methodology has significantly advanced the practice of structural engineering. In addition to providing the ability to model the structural behavior under static or dynamic load, newly developed web-interfaced CAE tools allow for easy access and transport information which eventually benefits structural engineering work.

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

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my gratitude to my advisor, Professor Jerome J. Connor, for his support, guidance, and inspiration. I enjoyed and learned a lot from every meeting and discussion we had been having during the Master of Engineering Program with the specialty of High Performance Structure. I definitely admire his rich knowledge and experience, as well as his broad viewpoint towards the innovative technology in both Structural Engineering and Information Technology.

Additionally, I would like to thank Professor John Williams and Professor Kevin Amaratunga for accepting me to the Information Technology Program for my second year graduate study at MIT. I enjoyed taking courses and working on group projects in the classes they offered regarding Computer-Aided Engineering and Software Developing. This valuable experience will surely build a bridge between my master study at MIT and future Ph.D. study at Carnegie Mellon University.

I am also grateful to all the people in my office, and friends who have been there for me for the past two years. Karen, Justin, Yo-Ming, Yi-san, Chun-Yi and Ching-Yu, thanks for your 'technical support'. Roxy, Bruce, Eric, Kathy, Pei-ting, Ching-yin. I will always remember how we sheared our thoughts, our minds and our hearts! Special thanks for my family friend George who constantly supports me to overcome all difficulties at school.

And, finally, Daddy and Mommy, thanks for being wonderfully supportive. I will keep going and try my best.

[TABLE OF CONTENT]

[LIST OF TABLES]

[LIST OF FIGURES]

Chapter I

INTRODUCTION

1.1 Motivation

In the past several decades, civil engineering in general and structural engineering in particular have achieved remarkable success in employing computational support technologies. Useful tools for specific process steps, notably preliminary design and simulation, have been developed for many advanced structural optimization and control cases under static or dynamic load [1]. However, the full integration of future computing capabilities into structural engineering is a challenge to civil and structural engineers who use computer technology to implement the design methodology.

Among all the topics related to civil or structural engineering, optimization on form and function, and control of structural system appear to be two of the most important topics in computer-aided engineering methodology. Owing to the mathematical complexity of structural analysis and control, engineers are seeking better ways to simplify the tedious calculations required to optimize the performance. With the advances in computer programming languages and the concurrent development in software product for design and analysis, structural engineers can obtain numerical results from compiling coding syntax or applying sophisticated simulation package tools.

1.2 Innovation

Several innovative methodologies are widely applied to engineering simulation. In recent years, topology has been incorporated in structural optimization. Topology is concerned the pattern of connectivity or spatial sequence of members and elements in the entire structure [2]. Optimization of the structure is usually obtained with computer simulation. Table 1 illustrated the differences between exact analytical formulation and approximate computer simulation in structural design:

Table 1 : Comparing Exact-Analytical and Approximate-Discretized Formulation

Formulation	***Exact***	***Approximation***
Computational Method	Analytical	Numerical
Structural Model	Continuum	Discretized (FEM)
Solution Procedure	Simultaneous Solution of All Equations	Interactive Solution
Initial Structure	Structure Universe (Infinite No. of Members)	Ground Structure (Finite but Large Number of Members)
Prescribed Minimum Cross-sectional Area	Zero	Small (or Zero)
Usual Means of Computation	By Hand	By Computer

Figures 1 and 2 illustrate a particular coupling of topology and structural shape optimization, called the "*Bubble-Method*" [3]. Its basic idea is the interactive positioning of modified components into a given design domain. Resulting from the increasing demands on the efficiency, reliability and shortened development cycle of bridge or building structure, it has become inevitable to solve problems by computer-based procedures. Substantial progress has been achieved in the structural computation of components, especially in view of the versatile Finite-Element-Method [4].

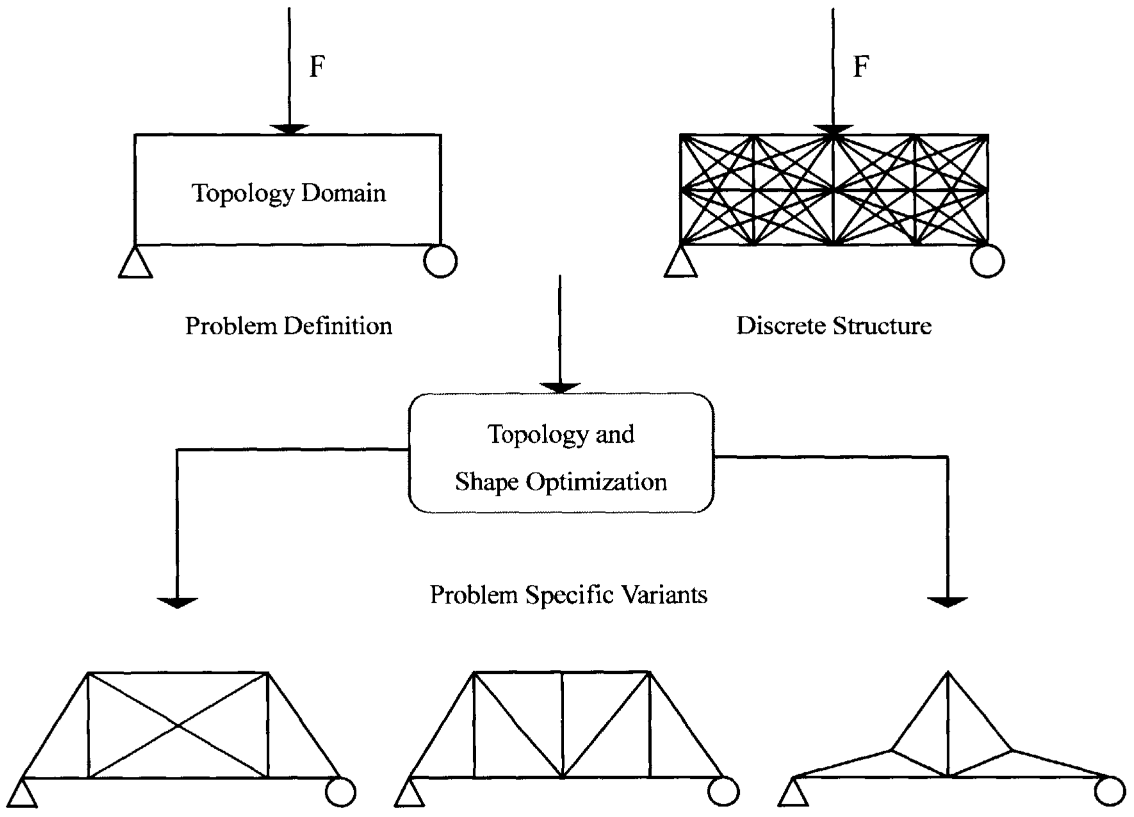


Figure 1: Use of Topology and Shape Optimization in the Design Process

1.3 Application

Some programming languages are very suitable to apply on the computer-aided engineering on structural optimization and control. For instance, Matlab has very powerful functions for dynamic analysis and graphical representation of the output. In Figure 3a-3l, twelve modes of a bridge under vibration are illustrated with the deformation of the truss structure. At the same time, Matlab code, which generates the graphs, is presented in Appendix [1] at the end of this thesis.

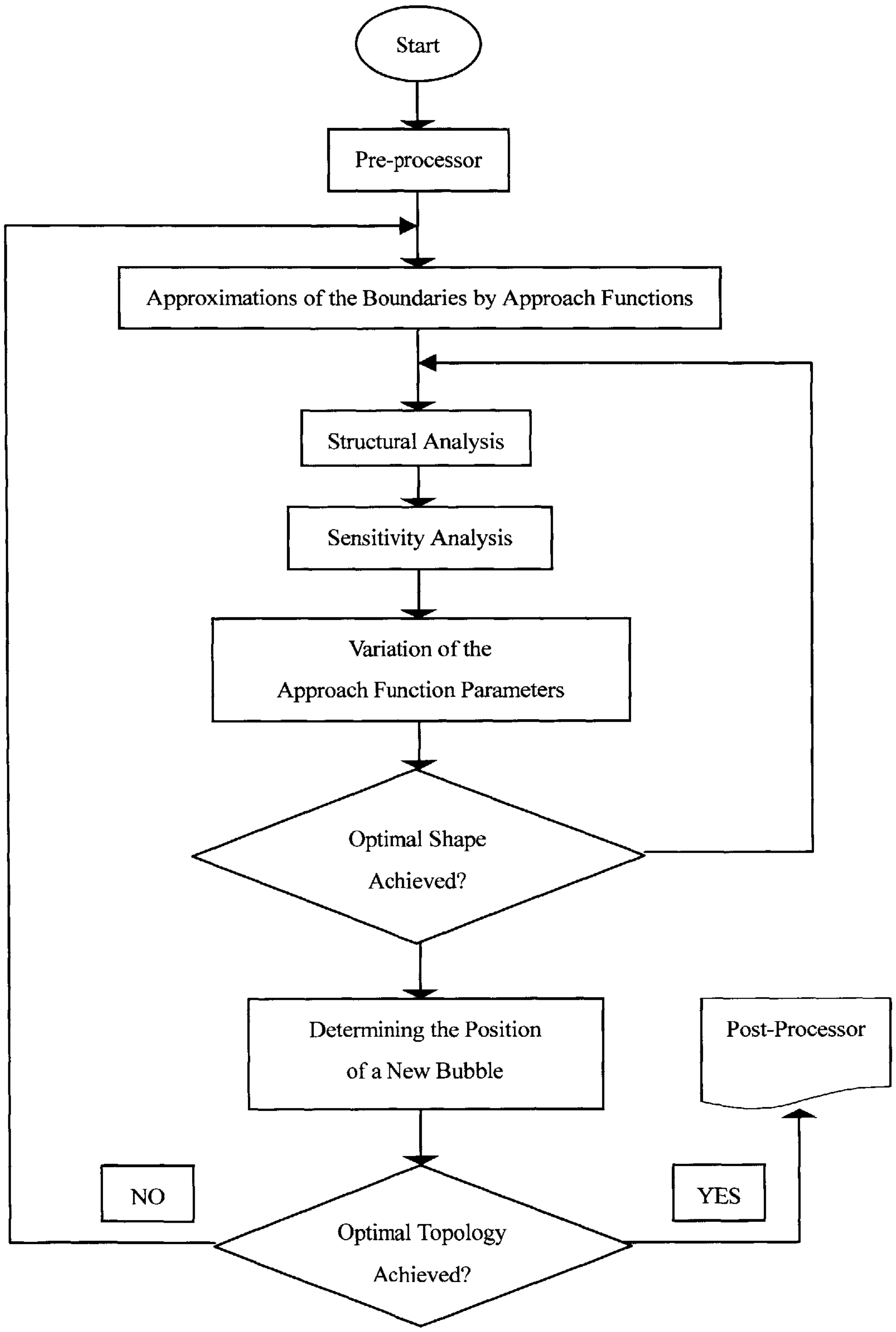

Start
Pre-processor
Approximations of the Boundaries by Approach Functions
Structural Analysis
Sensitivity Analysis
Variation of the Approach Function Parameters
Optimal Shape Achieved?
Determining the Position of a New Bubble
Post-Processor
NO
YES
Optimal Topology Achieved?

Figure 2: Flow Chart of the Bubble Method

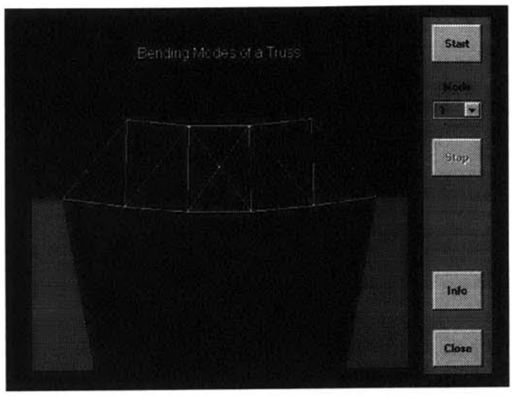


Fig. (3a) First Mode Vibration Displacement

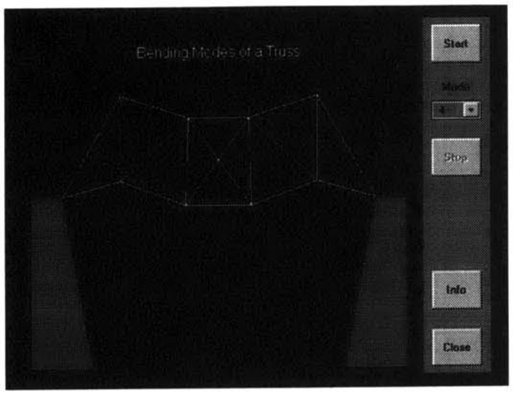


Fig. (3d) Fourth Mode Vibration Displacement

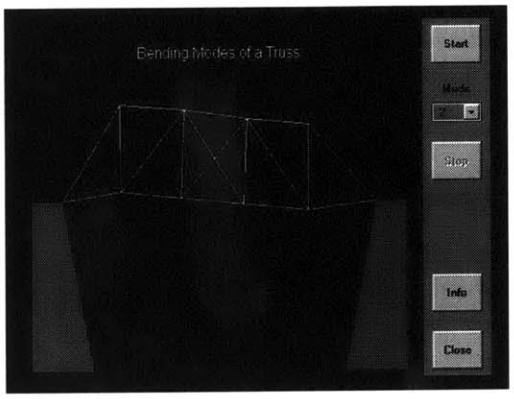


Fig. (3b) Second Mode Vibration Displacement.

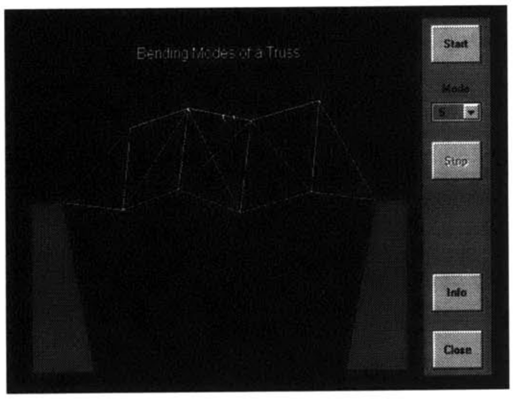


Fig. (3e) Fifth Mode Vibration Displacement

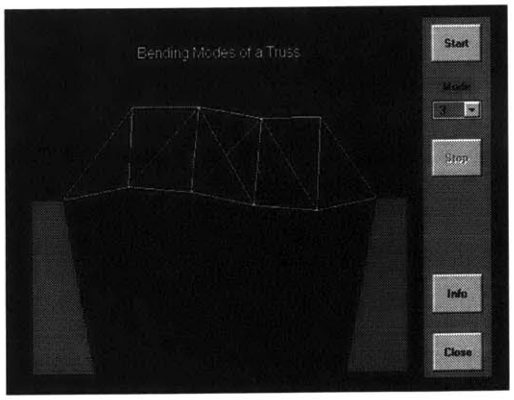


Fig. (3c) Third Mode Vibration Displacement.

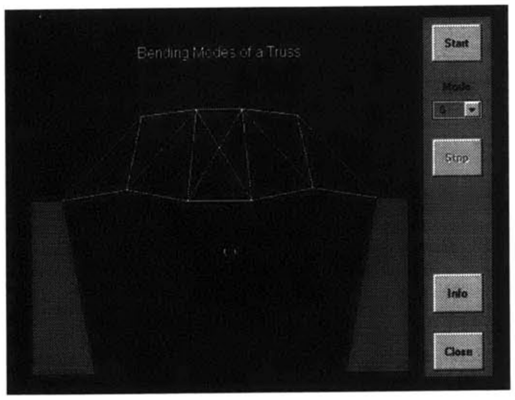


Fig. (3f) Sixth Mode Vibration Displacement

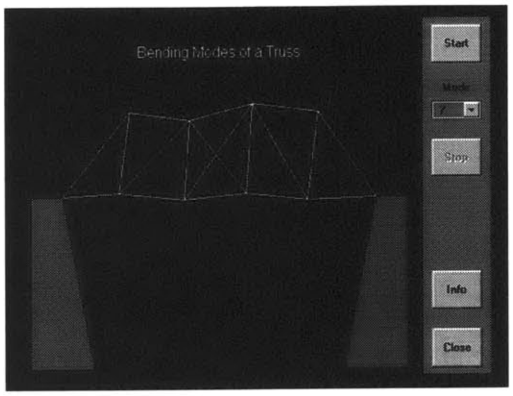


Fig. (3g) Seventh Mode Vibration Displacement

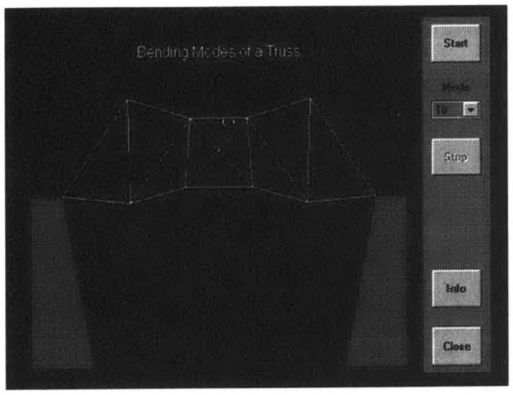


Fig. (3j) Tenth Mode Vibration Displacement

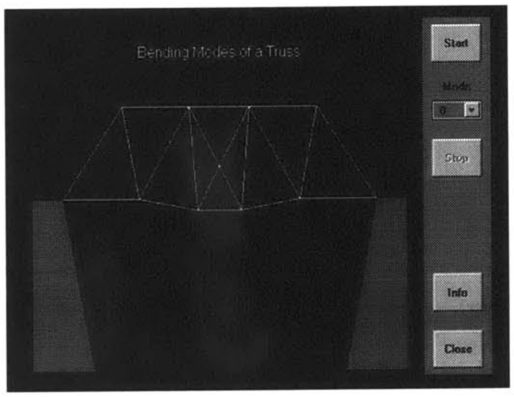


Fig. (3h) Eighth Mode Vibration Displacement

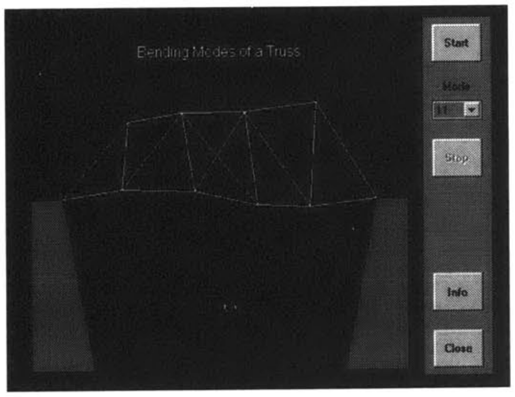


Fig. (3k) Eleventh Mode Vibration Displacement

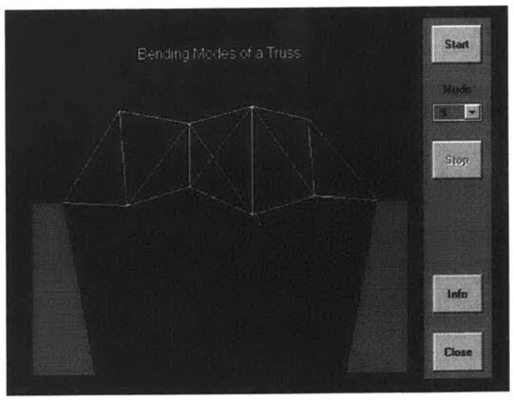


Fig. (3i) Ninth Mode Vibration Displacement

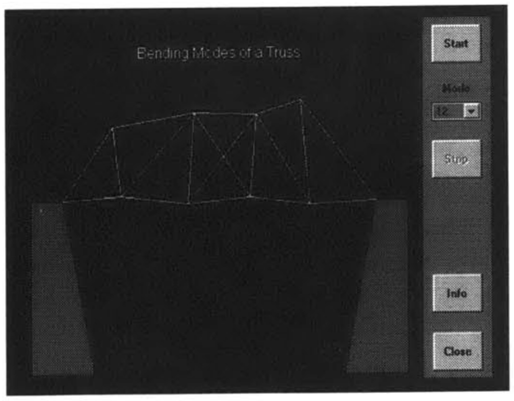


Fig. (3l) Twelfth Mode Vibration Displacement

Figure 3: Matlab Output of Bridge Structural Vibration--12 Modes Presented

Chapter II

CAE METHODOLOGY ON MATLAB

Among civil engineering students, structural engineers and academic faculty, Matlab is becoming increasingly popular because of its features such as interactive mode of work, immediate graphing facilities, built-in functions, the possibility of adding user-written syntax and simple programming. Also, data sets as well as options for keeping records of calculation result can be later transformed into technical reports [5]. For example, in some case studies of control theories presented in the second and the fourth chapter of this dissertation, the numerical data and output graphics can be generated into different window objects to be inserted and linked to another document. Besides, Matlab operates simple coding syntax which is very similar to scientific or engineering declarations. As a result, structural engineers will benefit from this handy tool for logic operation of preliminary simulation.

2.1 From Mathematical Model to Structural Control

The design and analysis of structural control systems is based on mathematical models of complex physical systems. The mathematical models, which derive from the physical or mechanical lows of the process, are generally highly coupled with advanced engineering mathematics such as nonlinear differential equations [6]. Fortunately, with the modern computer languages, many abstract physical or mechanical systems can be modeled to behave linearly around an operating point within some range of the variables.

The Laplace Transform method is one way to compute the solution of differential equation [7]. It is also possible to develop linear approximation to the structural or mechanical system. Furthermore, it can be used to obtain an input-output description of the linear, time-invariant (LTI) system in the form of a transfer function in structural vibration control. The application of many classical and modern control system design and analysis tools is based on LTI mathematical models. Particularly, Matlab can be utilized with LTI systems given in the form of transfer function descriptions or state-space description in advanced structural dynamics or seismic engineering [5].

In the following paragraphs, several case studies together with Matlab program demonstrate that Computer-Aided Engineering Methodology can be used for simulation and visualization of the results of preliminary structural optimal control of a dynamic system.

2.2 Spring-Mass-Damper System

A spring-mass-damper mechanical system is shown in Figure 4 as a simplified structural component. The motion of the mass, denoted by *y(t)*, is described by the differential equation:

$$M\ddot{y}(t) + f\dot{y}(t) + Ky(t) = r(t) \qquad \textbf{Equation (1)}$$

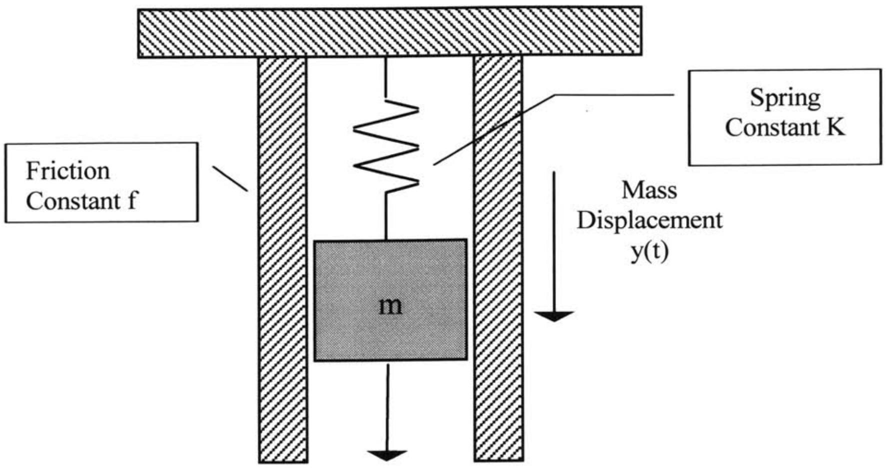


Figure 4: 1-D Spring-Mass-Damper Mechanical System

The solution, *y(t)*, of the differential equation describes the displacement of the mass as a function of time. The forcing function is represented by *r(t)*. These ideal models for the spring and damper are based on lumped, linear, dynamic elements and only approximate the actual elements.

In order to illustrate the usefulness of the Laplace transformation and the steps involved in the system analysis, rewrite Equation (1) to:

$$M\frac{d^2y}{dt^2}+f\frac{dy}{dt}+Ky=r(t) \qquad \textbf{Equation (2)}$$

We wish to obtain the response, *y*, as a function of time. The Laplace transform of Equation (2) is:

$$M\left(s^2Y(s)-sy(0^+)-\frac{dy(0)}{dt}\right)+f\left(sY(s)-y(0^+)\right)+KY(s)=R(s) \qquad \textbf{Equation (3)}$$

when

$$r(t)=0, \qquad \mathrm{y}(0^+)=y_0, \qquad \frac{dy}{dt}\Big|_{t=0}=0,$$

we have

$$Ms^2Y(s) - Msy_0 + fsY(s) - fy_0 + KY(s) = 0$$ **Equation (4)**

Solving for Y(s), we obtain

$$Y(s) = \frac{(Ms+f)y_0}{Ms^2+fs+K} = \frac{(s+f/M)(y_0)}{(s^2+(f/M)s+K/M)} = \frac{(s+2\xi\omega_n)(y_0)}{s^2+2\xi\omega_n s+\omega_n^2}$$ **Equation (5)**

Our objective is to design an active control system to make the structural more stable under dynamic loading. In Equation (5), ω_n *=sqrt(K/M)* is the natural frequency of the system and ξ*=f/2*[sqrt(KM)]* is the dimensionless damping ratio. In the following paragraphs, the control design and subsequent analysis would be based on the model described before. We will soon see how to use Matlab to enhance our structural control design and analysis capability.

The unforced dynamic response, y(t), of the spring-mass-damper mechanical system is:

$$y(t) = \frac{y(0)}{\sqrt{1-\zeta^2}} e^{-\xi\omega_n t} \sin\left(\omega_n\sqrt{1-\xi^2}t+\theta\right)$$ **Equation (6)**

where $\theta=\cos^{-1}\xi$. The initial displacement is *y(0)*. The transient system response is underdamped when $\xi<1$, overdamped when $\xi>1$, and criticallydamped when $\xi=1$. We can use Matlab to visualized the unforced time response of the mass displacement following an initial displacement of *y(0)*. Consider the overdamped and underdamped cases:

- Case 1: y(0)=0.15m, ω_n=sqrt2 (rad/sec), ξ_1=3/[2sqrt(2)] [(K/M)=2, (f/M)=3]
- Case 2: y(0)=0.15m, ω_n=sqrt2 (rad/sec), ξ_2=1/[2sqrt(2)] [(K/M)=2, (f/M)=3]

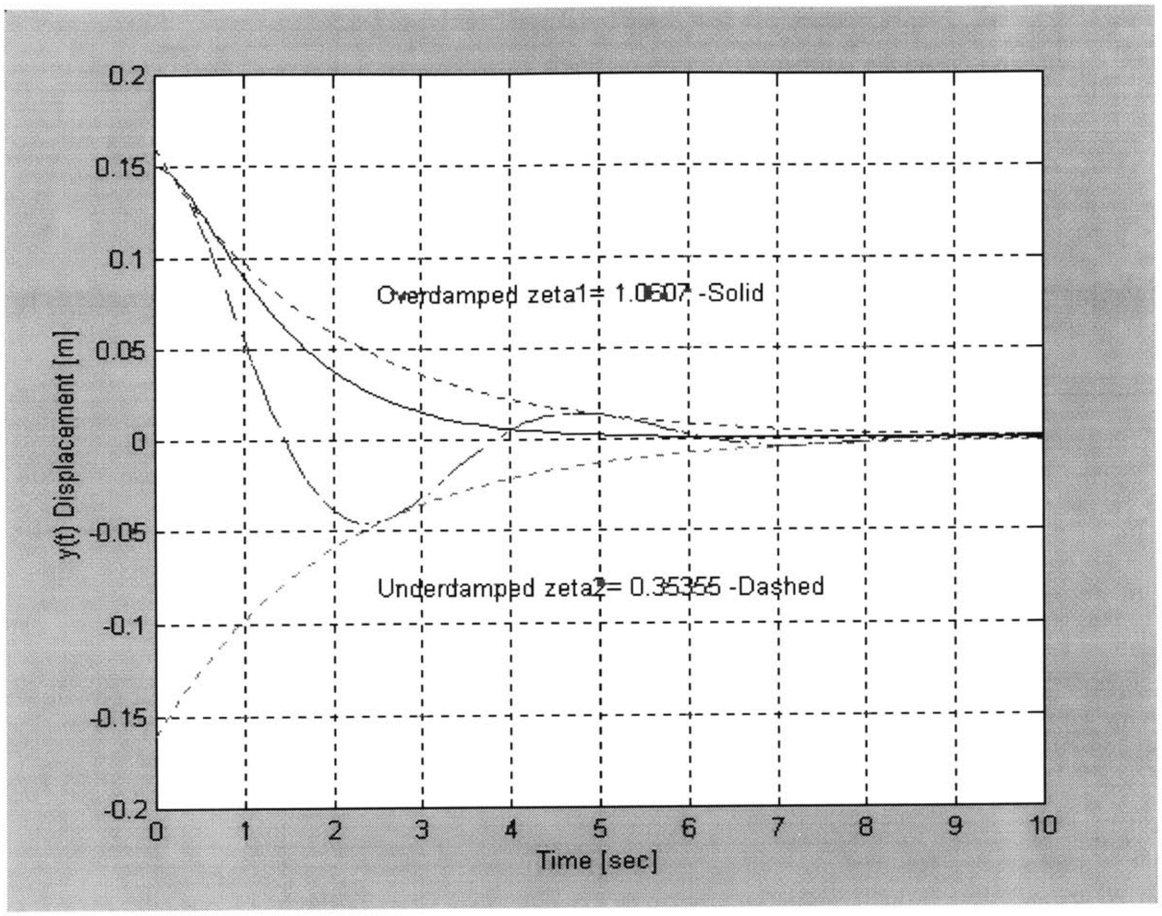


Figure 5: Response of a Spring-Mass-Damper System

The Matlab commands to generate the plot of unforced response are shown in Appendix [2]. In the Matlab setup, the variables *y(0)*, ω_n, ξ_1 and ξ_2 are inputs to the workplace at the command level. Then the script <unforced.m> is executed to generate the desired plots. This creates an interactive analysis capability to analyze the effects of natural frequency and damping on the unforced response of the mass displacement.

Figure 5 shows the plotting result generated by the Matlab function. One can investigate the effect of varying the natural frequency and the damping on the time response by simply changing the values of ω_n, ξ_1 or ξ_2 in the command syntax and executing the <unforced.m> program again. Notice that the script automatically labels the plot with the values of the damping coefficients. This avoids confusion when making many interactive simulations. The natural frequency value (zeta) could also be automatically labeled on the plot. Utilizing scripts is an important aspect of developing an effective interactive program in computer-aided design and analysis capability in Matlab. Since one can relate the natural frequency and damping to the spring constant, *K*, and friction *f*, one can also analyze the effects of *K* and *f* on the response.

In the spring-mass-damper problem, the unforced solution to the differential equation was readily available. In general, when simulating close-loop feedback control systems subjected to a variety of inputs and initial conditions, it is not feasible to attempt to obtain the solution analytically. In these cases, it is very convenient to use Matlab to compute the solution numerically and to display the solution graphically [7]. Noted that Matlab is one of the many useful programs of doing computer-aided engineering in structural optimization and control. Users can always select their favorite CAE tools.

2.3 Structural Control System Performance

Primary concerns in structural control system design are stability and performance. In order to design and analyze structural control systems, we must first establish adequate performance specifications to be presented in the time domain or the frequency domain. Time-domain specifications generally take the form of setting time, percent overshoot, rise time and steady-state error specification. In this section, a Matlab function called "`impulse`" is introduced to simulate linear control engineering system.

Time-domain performance specifications are generally given in terms of the transient response of a system to a given input signal, for example, seismic wave or dynamic loading histogram [8]. Since the actual input signals are usually unknown, a standard test input signal is used in the following case study and coding example can be found in Appendix [3]. The test signal is of the general form:

$$r(t) = t^n \qquad \textbf{Equation (7)}$$

and the corresponding Laplace transform is:

$$R(s) = \frac{n!}{s^{n+1}} \qquad \textbf{Equation (8)}$$

when n=2, *R(s)* is defined as a step function, which is shown in the block diagram in Figure 6 below:

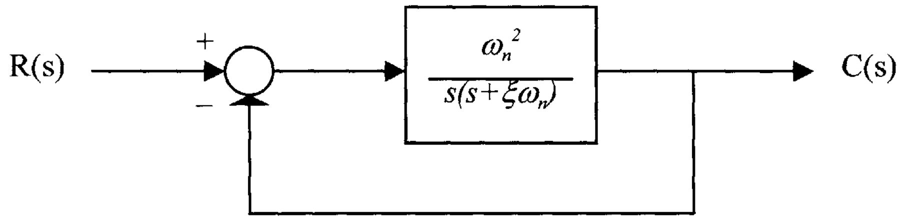


Figure 6: Single-Loop Second-Order Feedback System

Consider this second-order system, the close-loop output is:

$$C(s) = \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} R(s) \qquad \textbf{Equation (9)}$$

From the above equation, one can tell that this output is similar to the step response function, which is plotted in Figure 7. Besides, the feedback control system has the following characteristics:

1. decrease the sensibility of the system to plant variation
2. enable adjustment of the system transient response
3. reject disturbances and
4. reduce steady-state tracking errors

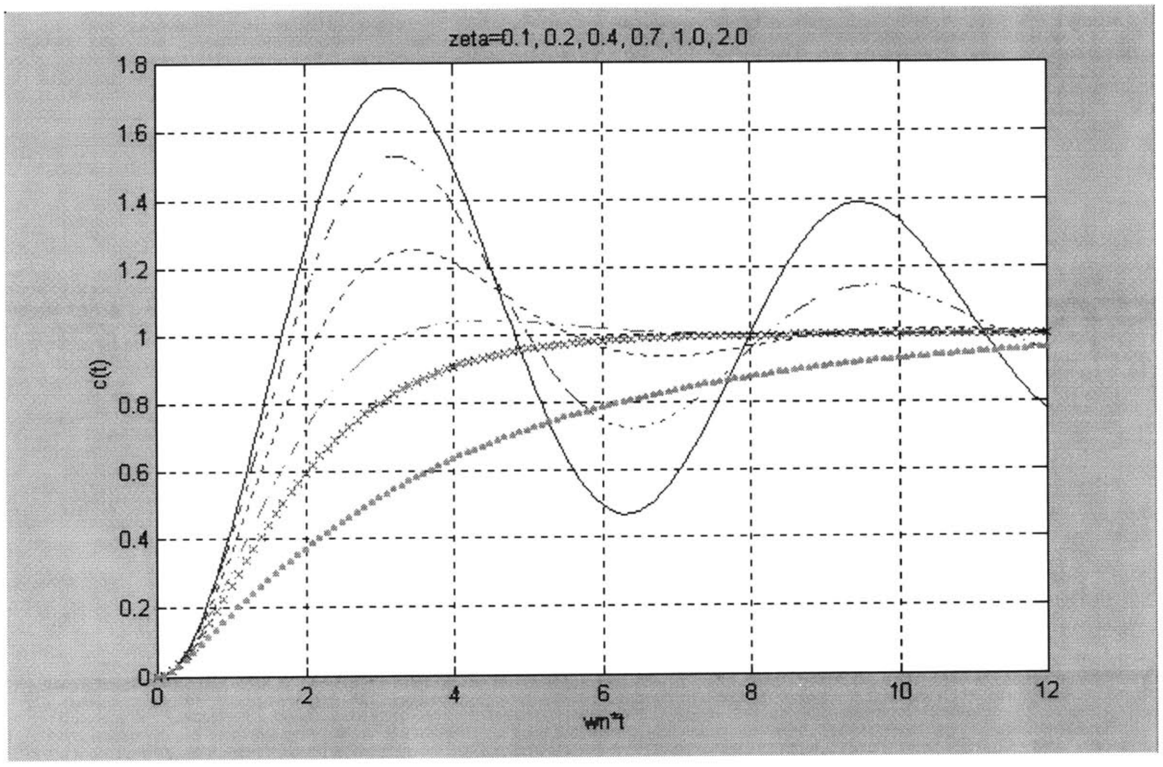


Figure 7: Step Response of the Second-Order System

2.4 Mechanical Vibration on Structural Dynamics

In this case study, we are going to use Matlab to compute the response of a single degree of freedom system to an arbitrary forcing function. The intent is to take advantage of Matlab function such as convolution operation for the analysis of dynamic response of a simple structural system.

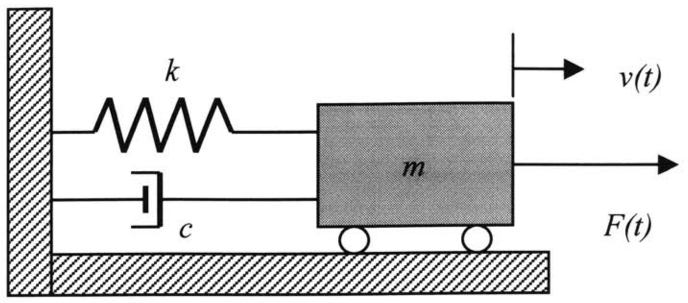


Figure 8: One Degree-of-freedom Structural System Subjected to Dynamic Load

For the system shown in Figure 8, the equation of motion is:

$$m\ddot{v} + c\dot{v} + kv = F(t)$$ **Equation (10)**

The impulse response of the above system is defined as the solution of Equation (10) when *F(t)=δ(t)*. The impulse response function *h(t)* is given by:

$$h(t) = \begin{cases} \dfrac{1}{m\omega_d} \times \sin(\omega_d t) \times e^{-\xi w_n t} & \text{when t} \geq 0 \\ \\ 0 & \text{when t} < 0 \end{cases}$$ **Equation (11)**

where ω_n= *(k/m)* is the undamped natural frequency, ω_n= ω_d[sqrt*(1-ξ^2)*] is the damped frequency, and *ξ=c/2*[sqrt*(k*m)*] is the damped ratio. The response of the above system to any arbitrary forcing function *F(t)* can then be computed as the convolution of this forcing function with the system's impulse response h(t):

$$v(t) = \int_{-\infty}^{\infty} h(\tau)F(t-\tau)d\tau$$ **Equation (12)**

Because of causality, the above expression becomes:

$$v(t) = \int_0^\infty h(\tau)F(t-\tau)d\tau \quad \text{or} \quad \int_0^\infty h(t-\tau)F(\tau)d\tau \qquad \textbf{Equation (13)}$$

In the Matlab program listed in Appendix [4], one can compute the response function *v(t)* of a single degree of freedom system when the excitation *F(t)* is an arbitrary function that a structural or mechanical engineer might define.

The program prompts the user to input the M, C, and K values (with any consistent system of units) and then calculates and displays the impulse response function. For example, one might try M= 1 kg, C= 0.05N/ (m/s), K= 4 N/m. It also displays the undamped natural frequency in rad/s and Hz, as well as the damping ratio *Zt*. Additionally, it computes the value of $1/(M*\omega_d)$, the coefficient of the impulse response function *h(t=0)*. For a lightly damped system this should be about the peak value of *h(t)*.

The response to any input can be obtained by convoluting the impulse response *h(t)*, with that of an arbitrary excitation vector. To do this, one needs to edit this file <convol.m> and modify the excitation vector, e.g., let $f=sin(\omega_n*t)$.

Because we want our response to die out completely over our time axis, we will make the length of the time axis four times the time constant *T*, which is defined as, $T=1/(Zt*\omega_n)$. This is the length of time required for the impulse response function to decrease by the factor *1/e*.

While executing this computer program, one has to enter numerical values for the system's parameters. Once all three numbers (M, C, K) are entered, the impulse response *h(t)* and the excitation force *F(t)* will be plotted, and the response function *v(t)* will also be showed along with the "*Matlab Graph Window*" [5].

The forcing function we have used in the <convol.m> program is a delayed impulse. By modifying the Matlab syntax and re-running the program, the corresponding result will be displayed in Figure 9 to Figure 13. It is very convenient to compare the three diagrams of impulse response, time history, and response to the excitation at the same time from the Matlab plotting. In the first case, the forcing function is a single pulse of height 1.0 and width *DT*=0.16 seconds. This value was delayed by 167=(500/3) timesteps. The response *x(t)* is plotted in the third graphs of the output figure.

The Matlab convolution is a numerical integration. The discrete time step, *DT*, in the <convol.m> program is a function of damping. However, this is only a result of programming choice not physics [9]. In fact, *DT*=4**T*/1000, where *T=1/(Zt*Wn)*, and *Zt* is defined as the damping ration in the program. If the selected damping ratio is too small, the number of time steps in one cycle of motion is too few and the numerical integration is inaccurate.

Figure 10 is the output after editing the program and substituting $F(t)=sin(\omega_n*t)$. The damping value is assigned to be 2% of critical damping. Since the excitation frequency equals the natural frequency, this is resonant excitation with a transient buildup to steady state.

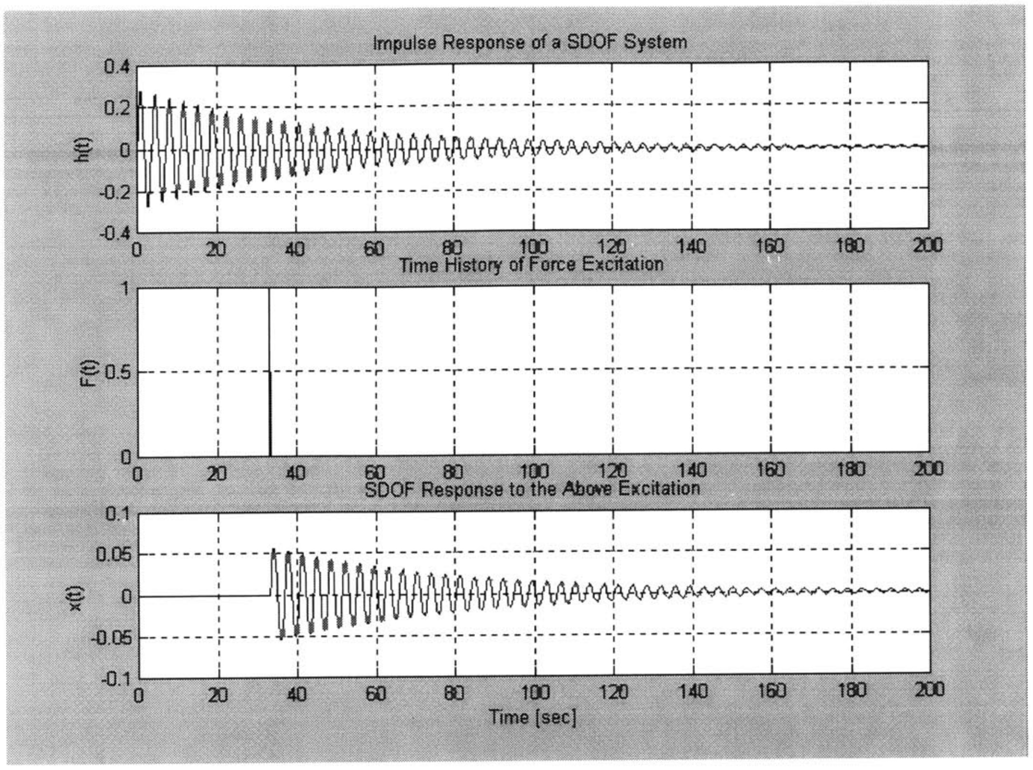


Figure 9: f(1,167)=1

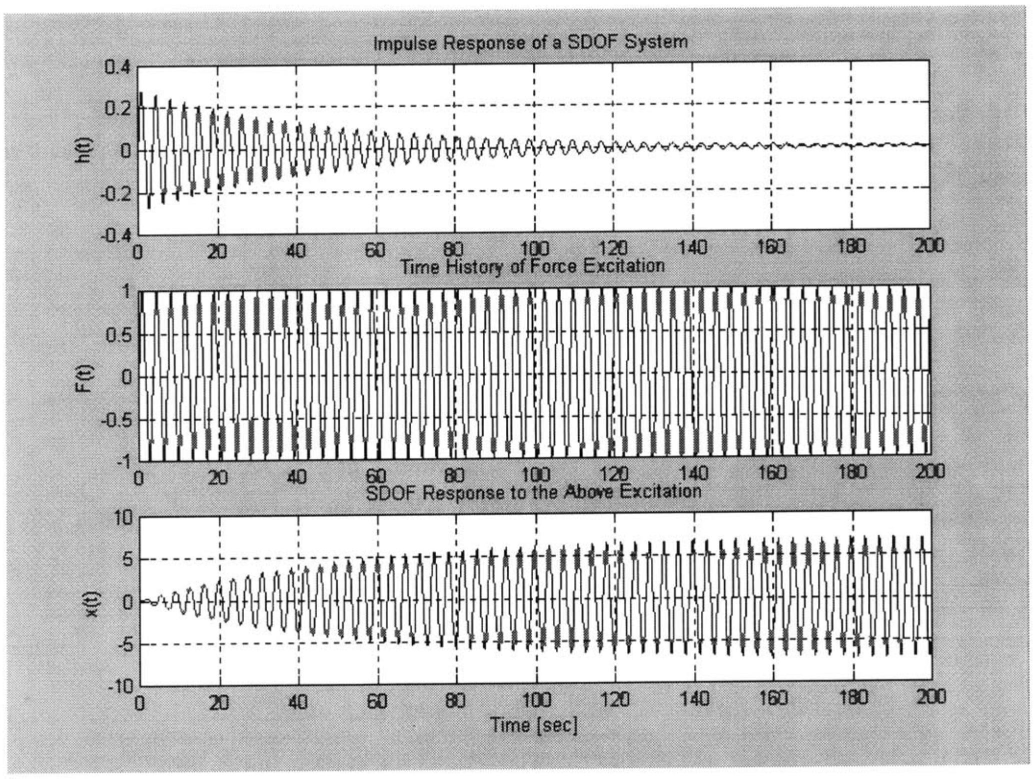


Figure 10: f=sin(ω_n*t)

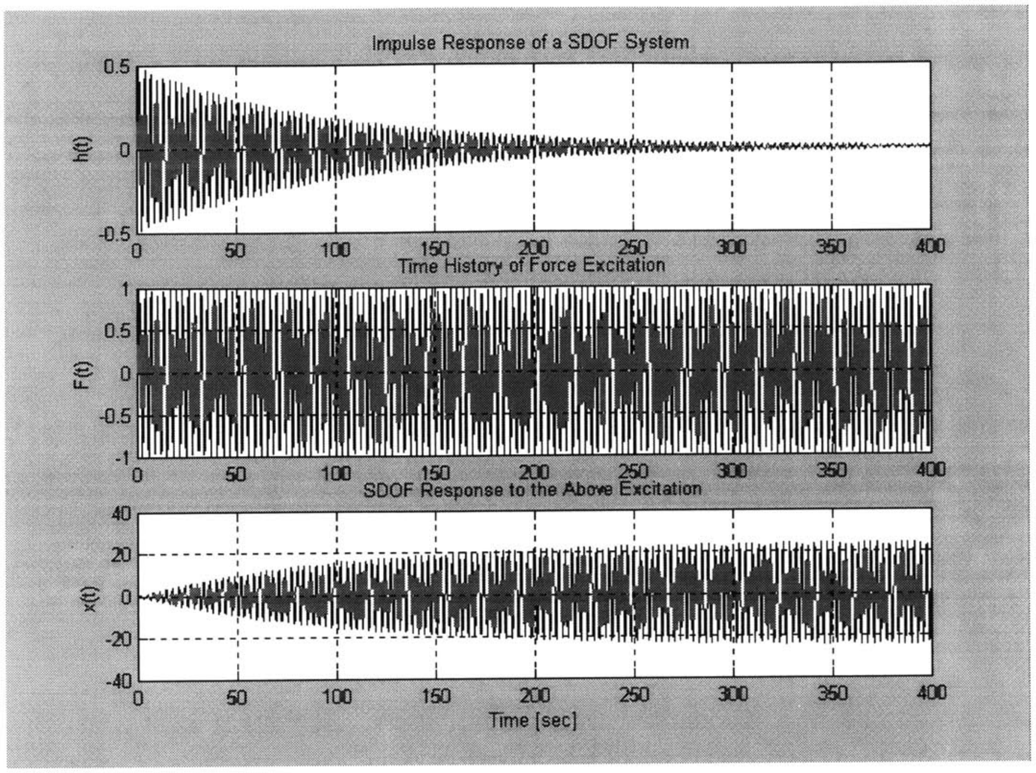


Figure 11: Critical Damping c=2%

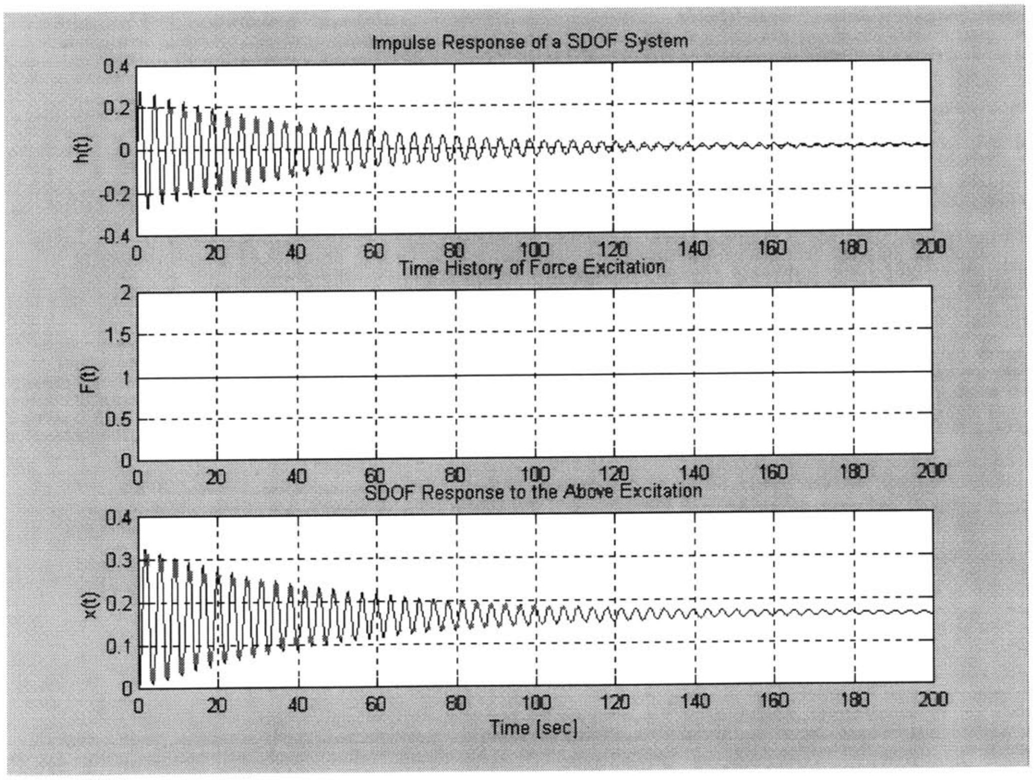


Figure 12: Step Function Starting at t=0

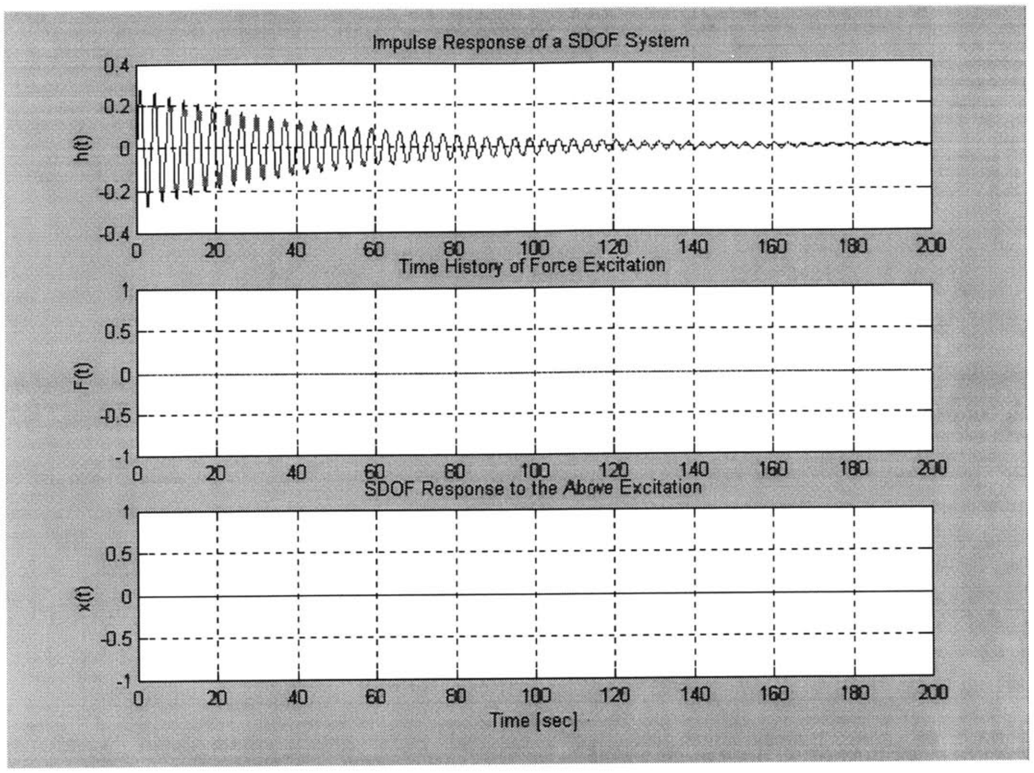


Figure 13: f=zeros(1,1000) ; No Step Function

By modifying the Matlab code, try two other frequencies, such as 0.4 and 3 times the natural frequency, i.e. let $F(t)=sin(0.4*\omega_n*t)$ and $F(t)=sin(3*\omega_n*t)$. In other words, one can see if the stiffness controlled or mass controlled steady state response behavior according to $H_{VH}(\omega)$ transfer action.

Chapter III

Structural Optimization And Control Using CAE

In this chapter, certain mathematical or mechanical CAE models for solving special structural design problems will be presented. To develop a sophisticated CAE tool particularly for structural optimization and control, the algorithm and methodology are the most important parts of the prototype. Additionally, new concepts have been brought to civil and structural engineering from computer science and operation research to improve the performance to the ultimate structural design.

3.1 Discrete Structural Optimization Using Evolution Strategies

3.1.1 The Definition and Concept of Evolution Strategies

In recent years, several new methods to handle discrete structural optimization have been developed. For example, the evolution strategies (ESs), which imitate biological evolution in nature, have been converted to the computer algorithm applied to continuous optimization problem [10]. Instead of solving a single design point in the conventional structural engineering task, the ESs work simultaneously with a group of design points in the space of variables. The basic concept of ESs presents a method of sequential linear discrete programming and converts the nonlinear discrete problem into a sequence of [1,0]-problem [11]. Engineers can apply the methodology of sequential approximation to the mixed-discrete nonlinear programming problem by combining a sequential linear one with a branch and bound algorithm.

Once the continuous problem has been transformed into a discrete task to be executed iteratively by computer, the structural design can be completed with less error and time. In the following section, two examples illustrating the use of evolution strategies (two-member ESs and multi-member ESs) will be presented. With of this procedure, a nonlinear discrete structural optimization problem can be solved.

3.1.2 Mathematical Formulation of the Evolution Strategies

In general, a structural optimization problem can be represented as a nonlinear mathematical computer-programming problem as the following form [10]:

$$\begin{array}{lll} \text{minimize} & f(X) & X \in E^n \\ \text{subject to} & g_i(X) \geq 0, & i = 1,2,\dots,m. \end{array}$$ **Equation (14)**

First of all, to solve the above equation, the two-membered evolution strategies work in two steps:

<Step 1: Mutation>

In the gth generation a new design-vector is computed from where $X_O^{(g)}$ *and* $X_P^{(g)}$ are the offspring and parent vectors, respectively:

$$X_O^{(g)} = X_P^{(g)} + Z^{(g)}$$ **Equation (15)**

$Z^{(g)}=[z_1^{(g)}, z_2^{(g)}, \dots, z_n^{(g)}]^T$ is a normally distributed random vector whose components $z_1^{(g)}$ has a probability density function:

$$p(z_i)=\frac{1}{\sqrt{(2\pi)}\sigma_i}\exp\left[-\frac{(z_i-\xi_i)^2}{2\sigma_i^2}\right]$$ **Equation (16)**

The design purpose of the computer program should fulfill the requirement that the expectation ξ_i should be zero and the variation σ_i^2 should be small.

<Step 2: Selection>

The selection chooses the best individual between the parent and offspring to match the conditions described:

$$X_P^{(g+1)}=\begin{cases} X_O^{(g)} & \text{if} \quad g_i(X_O^{(g)})\geq 0, \quad i=1,\dots,m \quad \text{and } f(X_O^{(g)})\leq f(X_P^{(g)}); \\ X_P^{(g)} & \text{otherwise,} \end{cases}$$

Equation (17)

Usually, $Z^{(g)}$ has the role of mutation. The standard deviation σ can be considered as a step length which is adjusted during the search success of the computer algorithm.

Secondly, in the case of multi-membered ESs, a population of parent vector and offspring vector will be handled simultaneously, and the operator recombination as well as the idea of self-adaptation of strategy parameters [10] is incorporated:

<Step1: Recombination and Mutation>

In the gth generation, λ offspring vectors are created by means of recombination and mutation.

<Step2: Selection>

There are two variants of multi-membered **ES:** *$(\mu+\lambda)$*-**ES** and *(μ,λ)*-**ES**.

(μ+λ)–**ES**: The best μ individuals are selected out of parents and offsprings to form the parents of the next generation. This is so-called the “genetic algorithm” for the iteration of the computer program.

(μ,λ)–**ES**: The best μ individuals are selected only from the λ offsprings. The *m* parents in the g^{th} generation die out. This is defined as the process of “optimization”.

3.1.3 Ten-Bar Truss: An Example

The well-known 10-bar truss problem shown in Figure 14 will be analyzed by the evolution strategy [12]. The objective function $f(X)$ of this problem is the weight of the structure itself. The design variables $g_i(X_O^{(g)})$ are the cross-sectional areas of the 10 members. The constraints $f(X_O^{(g)})$ and $f(X_P^{(g)})$ are the member stresses and the vertical displacement of the node 2 and 4. The allowable displacement is limited to 2 inches and the allowable stress is limited to ± 25 ksi. The design parameters are:

E=104 ksi	E states as Young's Modulus of the structure component.
F=100 kips	F states the applying force.
ρ=0.1 lb/in^3	ρ states the density of the material.
a=360 in	a states the length of the bar.

$$S = \begin{bmatrix} 1.62, & 1.80, & 1.99, & 2.13, & 2.38, & 2.62, \\ 2.63, & 2.88, & 2.93, & 3.09, & 3.13, & 3.38, \\ 3.47, & 3.55, & 3.63, & 3.84, & 3.87, & 3.88, \\ 4.18, & 4.22, & 4.49, & 4.59, & 4.80, & 4.97, \\ 5.12, & 5.74, & 7.22, & 7.97, & 11.50, & 13.50, \\ 13.90, & 14.20, & 15.50, & 16.00, & 16.90, & 18.80, \\ 19.90, & 22.00, & 22.90, & 26.50, & 30.00, & 33.50 \end{bmatrix} \text{in}^2$$

In this example the discretization of the design variables is considered and presented in matrix "*S*", which denotes the allowable area of the entire truss structure, according to the American Institute of Steel Construction Manual [13].

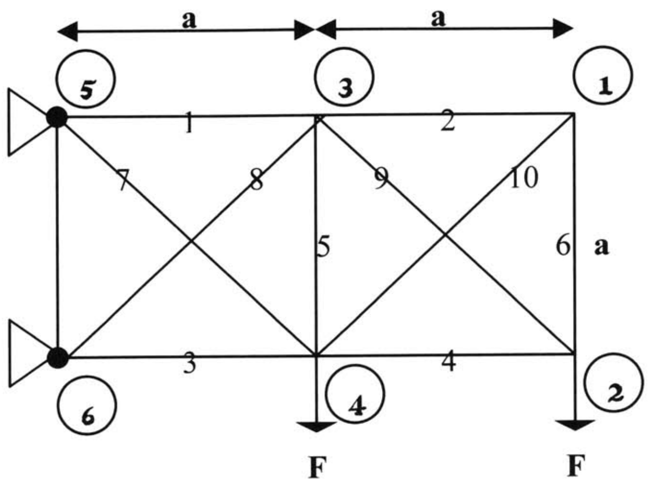


Figure 14: Ten-Bar Truss Using Evolution Strategies

This example is attempted to be solved with the improved multi-membered evolution strategies (M-ES) [11]. Several tests with different combination of μ and λ have been made and the results are given in Table 2. Also, the sample structural layout can be examined by using genetic algorithms (GAs shown in the Table).

Table 2: Solution of the Ten-Bar Truss

Approach	GAs	M-ES	M-ES	M-ES	M-ES
$\lambda+\mu$ or (λ,μ)	30,30	2+2	20+20	4,16	5+10
f (lb)	5613.84	5490.71	5490.71	5534.20	5490.71
X1	33.5	33.5	33.5	33.5	33.5
X2	1.62	1.62	1.62	1.99	1.62
X3	22.0	22.9	22.9	22.0	22.9
X4	15.5	14.2	14.2	13.9	14.2
X5	1.62	1.62	1.62	1.62	1.62
X6	1.62	1.62	1.62	2.13	1.62
X7	14.2	7.97	7.97	7.97	7.97
X8	19.9	22.9	22.9	22.9	22.9
X9	19.9	22.0	22.0	22.9	22.0
X10	2.62	1.62	1.62	1.80	1.62

This design example can be extended to a two-hundred-bar truss system. Since this ESs method applies to all discrete structural optimization problems [14], it is very convenient and reliable to solve the structural engineering program with this methodology. As a result, the multi-membered ESs work on a population of individuals – μ parent vectors and λ offspring vectors. This inherent parallelism allows an implementation in the parallel-computing environment such as workstation or super-computer, which is particularly for intensive computation for CAE on complex structural design and analysis [1].

3.2 Topological Design of Truss Structure

Topology means the pattern of connectivity or spatial sequence of members or elements in a structure. Optimization of the topology is involved in two fundamental classes of problems: (1) layout optimization; (2) generalized (variable topology) shape optimization [2].

In the following paragraphs, an extended study of the optimization technique will be presented. This topological concept is to minimize the weight of plane truss subjected to a multiple loading. The main tool for this study can be any modular program developed in C/C++ programming languages. The truss structure may be optimized over a N-dimensional space of continuous values for the member cross sectional areas or a discrete set of chosen values [11]. Furthermore, changes in the topology of the structure will be performed based on the theory of minimizing the member cross sectional area. Finally, the redundant members will be removed for generating the reduced structures.

3.2.1 Problem Formulation and Solution Strategy

Several other discrete and continuous optimization strategies have been studied and applied to truss design over many years. The main objective in truss optimization is to select cross sectional areas that minimize the truss weight and satisfy all strength and serviceability requirements. In addition, the optimization problem may be enlarged for the search of improved design topology.

The problem developed in this work is an example of simulated annealing applied to the minimum weight design of trusses for either fixed or variable topology constraints. With fixed topology optimization it is assumed that better designs may frequently be obtained by changing both the topology and the member cross sectional areas [15]. The optimization problem is formulated as to minimize the weight by minimizing the non-dimensional function:

$$f = \sum_{k=1}^{n} \frac{A_k l_k}{A_{\max} l_{\max} n}$$ **Equation (18)**

and subjected to stress constraints (g_k) for compressive and tensile members:

$$g_k = \begin{cases} 1 - \dfrac{F_{c_k}}{\sigma_{pc_k} A_k} & \text{for } F_{c_k} < 0 \\ \\ 1 - \dfrac{F_{c_k}}{\sigma_{pc_k} A_k} & \text{for } F_{c_k} \geq 0 \end{cases}$$ **Equation (19)**

where k is a generic member, A_k is the value of its cross sectional area, l_k is the value of its length, A_{max} is the maximum admissible value for the member cross sectional area, l_{max} is the sum of the member lengths, F_{tk} and F_{ck} are the tension and compression member force respectively, and σ_{ptk} and σ_{pck} are the permissible stresses.

The simulation of this structural optimization problem can be implemented by any programming language, such as C, C++ or Java which provide the concept of method and module to be called by the function. Usually in the coding procedure, the program can be divided into three main modules [10]. The first is a simple program for the analysis of plane trusses subject to multiple loading cases, for obtaining the member forces after each alteration. The second is a program to record the structure data when the current topology is different from the initial layout. It also performs slight changes to the topology when the structure becomes unstable. The third, i.e. the main program, implements the simulation for the minimum weight design of plane trusses. The last two parts are the heart of the solution strategy.

3.2.2 Structural Optimization and Topology Modification

If the requirement for a specific topology is selected, the main program may eliminate the effect of any structural member by making its cross sectional area equal to zero [12]. This means that the geometric and mechanical property of this member will be ignored. For example, from Figure 15(a2) to 15(b2), member 2 and member 10 are eliminated and the rest of the structure elements will be re-numbered automatically by the computer program. Hence, in order to carry out the structural analysis, the member with cross sectional areas equal to zero must be physically removed from the structure, and the structural layout has to be recorded for the new topology. A record of the original topology is kept during the iteration of analysis [2]. On the contrary, the removed members might eventually be re-inserted to form a new layout by making their cross sectional areas greater than zero defined by the programming algorithm.

The simulation algorithm of computer program works with the original topology in Figure 15(a1). Hence the analysis done for the modified topology from Figure 15(b1) to 15(b3) has to be translated back to the format and the numbering sequence used for the original topology. If by removing or re-inserting a member, the structure becomes unstable [12], then the program makes slight changes in the topology by removing or re-inserting one of the two nodal joint connectivities of this member in Figure 15(a3) to 15(b3). In addition, the other members connected to this node will be removed. If after this, the structure still remains unstable, then the topology modification is rejected and the computer program will not be executed afterwards [15].

Since there is several alternatives to manage the task of problem formulation and structural modification, minimum weight or allowable area of the entire layout is not necessarily the optimum solution of the design. Engineers have to carefully evaluate the detailed requirement before starting the algorithm or exact coding.

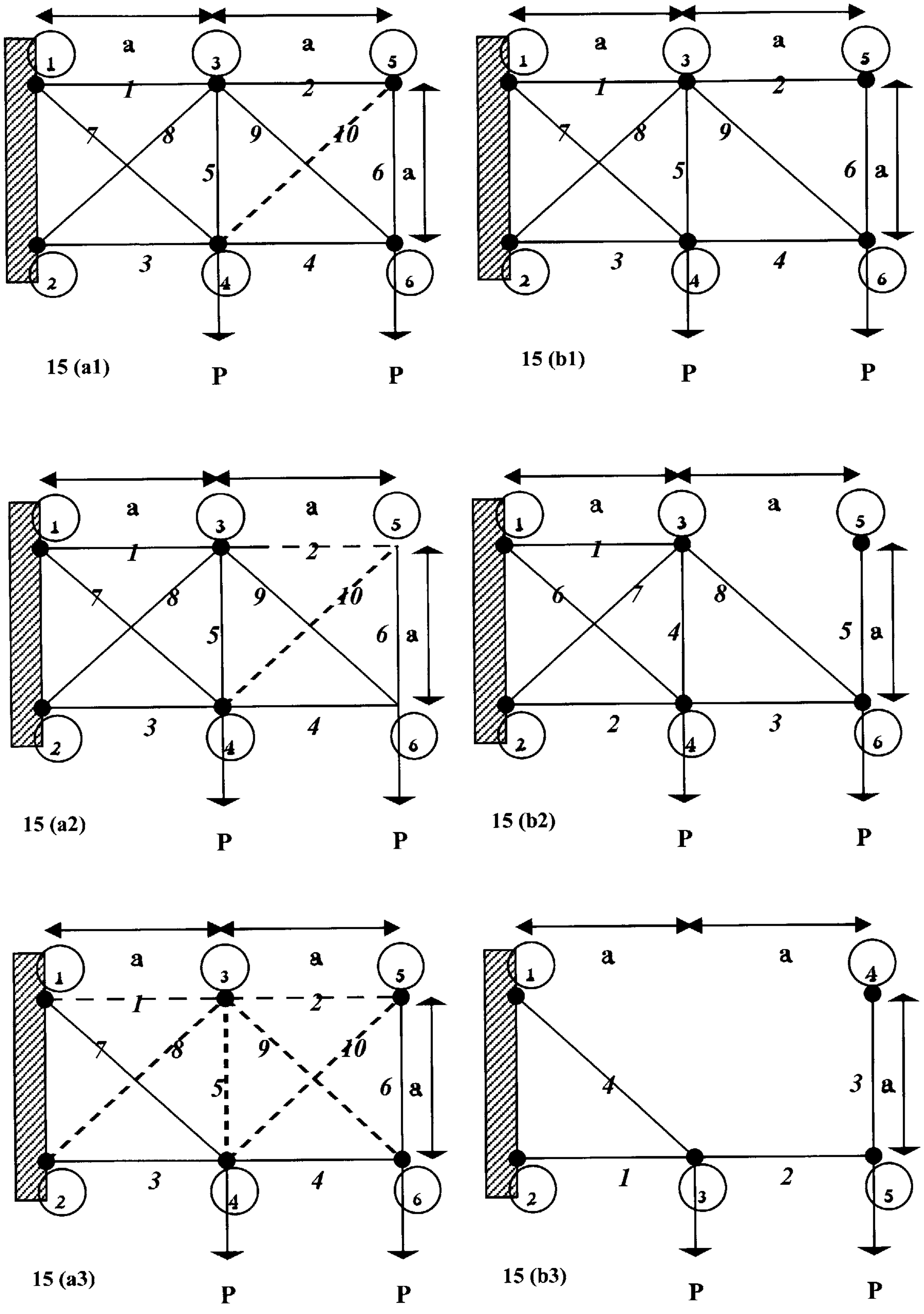


Figure 15: Example of a Path to Perform a Modification in the Topology

3.2.3 Generic Algorithm and FEM Applied on Aesthetic Design of Structures

Making master plan and evaluating configuration are becoming more and more important for aesthetic design of structure. Recently, an attempt is made to develop a decision-making supporting system for structural optimization and control based on generic algorithm and computer graphics [11]. Generic algorithm is applicable to produce many design alternatives, whereas computer graphics is useful to prepare their view simulations. In order to evaluate the alternative cases of structural design and optimization, Finite Element Method can be used to reflect the preferences of designer [4]. Meanwhile, the numerical simulation result from a structural layout should be presented to demonstrate the applicability of the system developed by this methodology.

In the past decade, the design guideline for structure follows the rule of mechanical property and performance requirement [16]. Recently, the aesthetic design of structures has been focused since the need to consider the view of the structure in the whole environment has becoming one of the very important design conditions. Due to the up-to-date developments in computer technology and data processing, it is possible to perform iterative or recursive calculation of load applied on the structure and it is effective to prepare several alternatives in the design process [3]. Moreover, the FEM tool can be applied to the previous case study and illustrate the structure deformation under different loading combinations. As a result, engineers can examine the effect on or harmony to the environment surrounding the structure from the viewpoint of aesthetics.

3.3 The Application of Neural Network on Vibration Control of Structures

In addition to Computer-Aided Engineering technology for structural design and optimization, advanced Neural Network Methodology has also been introduced in the field of Civil and Structural Engineering through mathematical concepts. For instance, Multi-layer Perception (MLP) network trained with the Back-Propagation Algorithm can be applied to damage assessment of a free-free cracked straight steel beam based on vibration measurement [17]. The problem of vibration damage assessment includes detecting, locating and quantifying a damage. Since artificial neural networks are providing to be an effective tool for pattern recognition, the basic idea is to train a neural network in order to recognize the behavior of the damaged/undamaged structure [18]. Particularly, this trained neural network should be subjected to the information regarding damaged states, locations and sizes of the vibration tests.

In this steel-beam case, the inputs to the network are usually estimated values of the relative changes of the lowest five bending natural frequencies due to structural damage. During the training process, these values can be obtained by a Finite Element Model (FEM) of the steel beam [4]. The basic idea of this model is to simulate the crack zone of a beam by means of a local flexibility matrix found from fracture mechanical prototype. The utility of the neural network approach is usually demonstrated by a simulation study as well as laboratory test. Finally, the neural network trained by simulated data will be capable for detecting location and size of a damage in a free-free beam when the network is subjected to an experimental data.

3.3.1 Introduction to the Neural Network

An artificial neural network is an assembly of a large number of highly connected processing units, which are so-called nodes or neurons [18]. The neurons are connected by unidirectional communication channels or connections. The strength of the connections between the nodes is represented by numerical values that normally are called "weights". Knowledge and information are stored in the form of a collection of weights. Each node has an activation value that is a function of the sum of inputs received from other nodes through the weighted connections. The neural networks are capable of self-organization and knowledge acquisition, i.e. learning, especially for adaptive control theorem of the structural vibration [19].

One of the characteristics of neural networks is the capability of producing correct, or nearly correct, outputs when presented with partially incorrect or incomplete inputs. Furthermore, neural networks are capable of performing an amount of generalization from the patterns on which they are trained [11]. Most neural networks have some sort of "training" rule whereby the weight of connections is adjusted on the basis of presented patterns. In other words, neural network "learns" from examples being executed by the program or mechanism, and exhibits some structural capability for generalization. Training consists of providing a set of known input-output pairs to the network. Eventually, the network iteratively adjusts the desired outputs for each input sets within a requested level of accuracy [10]. Error is defined as a measure of the difference between the computed pattern and the expected output pattern.

3.3.2 The Modeling and Thinking Methodology

Structural monitoring and control by measuring and analyzing vibrational signals of civil engineering structures is a subject of research investigated with increasing interests in the past decades [17]. Traditional structural inspection and control might be costly, risky and difficult to access. The basic approach for vibration control in structure is to detect the changes in the dynamic behavior of the structure that may be characterized by the natural frequencies and corresponding mode shapes [8]. For instance, one of the consequences of the development of a structural damage is a decrease in local stiffness, which in turn results in a change in some of the natural frequencies. Since natural frequencies can be obtained from the measurements at a single point of the structure, the computed changes of the response spectrum or mode shape can be used for structural control.

In this application, training of the network is performed with patterns of the relative changes of natural frequencies that occur due to structural vibration. This implies that each pattern represents the changed value of natural frequencies due to vibration behavior in the civil engineering structure. This value can be estimated by using finite-element model [4]. With all the values estimated by FEM and later brought to the input vector of the neural network, structural behavior can be predicted by the learning model after several computer iteration process. As a result, all CAE tools such as programming neural network and FEM will be very beneficial to the development of structural optimal control.

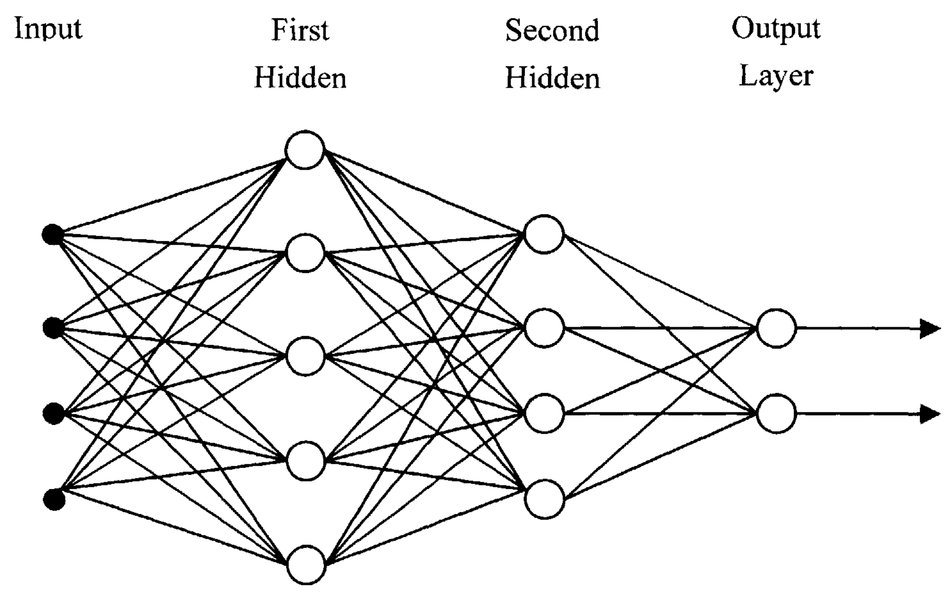


Figure 16: Multi-Layer Perception (MLP) Neural Network

3.3.3 Multi-Layer Perception

Artificial networks are computational models loosely inspired by the neuron architecture and operation of the human brain. The pioneering work in this field is usually attributed to McCulloch and Pitts in 1943 [20]. Since then there have been many studies of mathematical models of neural networks. The MLP network trained by the Back-Propagation Algorithm is currently given the greatest attention by application developers in the field of civil and structural engineering.

The multi-layered perception network belongs to the class of layered feed-forward (on the opposite of feed-back) nets with supervised learning [18]. A multi-layered neural network is made up of one or more hidden-layers placed between the input and output layers. Each layer consists of a number of nodes connected in the structure of a layered network. The typical architecture is fully interconnected, as shown in Figure 16. Each node in the lower level is connected to every node in the higher level. Output units cannot receive signals directly from the input layer.

Associated with each connection between node i and node j in the preceding layer l-1 and the following layer l is a numerical value $w_{lj,i}$, which is the strength or the weight of the connection. At the start of the training process these weights are initialized by random values. But on the application level, these input values would be replaced by the date obtained from the monitoring system of the real structure. Signal passes through the network and the jth node in layer l computed its output according to:

$$x_{lj} = f\left(\sum_{i=1}^{N_{l-1}} w_{lj,i} x_{l-1,i} + \theta_{lj}\right) \qquad \textbf{Equation (20)}$$

for j=1,...,N_l and l=1,...,k, where x_{lj} is the output of the jth node in the lth layer. θ_{lj} is a bias term or a threshold of the jth neuron in the lth layer. The kth layer is the output layer and the input layer must be labeled as layer zero. Thus N_0 and N_k refer to the number of network inputs and outputs, respectively. The function f is called the node activation function. For the nodes in the hidden layer, the activation function is often chosen to be a sigmoidal function [10]:

$$f(\beta) = \frac{1}{1+e^{-\beta}} \qquad \beta > 0 \qquad \textbf{Equation (21)}$$

For the neural network associated with each node on the hidden layer, node j, and each output node, node k, are coefficients or weights, θ_j and θ_k respectively. These weights are referred as biases [18]. Associated with each path, from an input node i to node j on the hidden layer, is an associated weight, w_{ij} and from node j on the hidden layer to output node k is an associated weight w_{jk}. Let q_i be input entered at node i. Node j on the hidden layer receives weighted inputs, $w_{ij}q_i$. It sums these inputs and uses an activation function to yield an output r_j. Thus, equation (21) can be written as:

$$r_i = \frac{1}{1+e^{-\sum w_{ij}q_i - \theta_j}} \qquad \textbf{Equation (22)}$$

Output node k then receives inputs $w_{jk}r_j$ which are summed and used with an activation function to yield an output s_k. This is to adjust the weights on each learning trial so as to reduce the difference between the predicted and desired output [11]. Training of this Neural Network is formulated as an unconstrained minimization problem. This algorithm performs a series of one-dimensional examinations along searching direction.

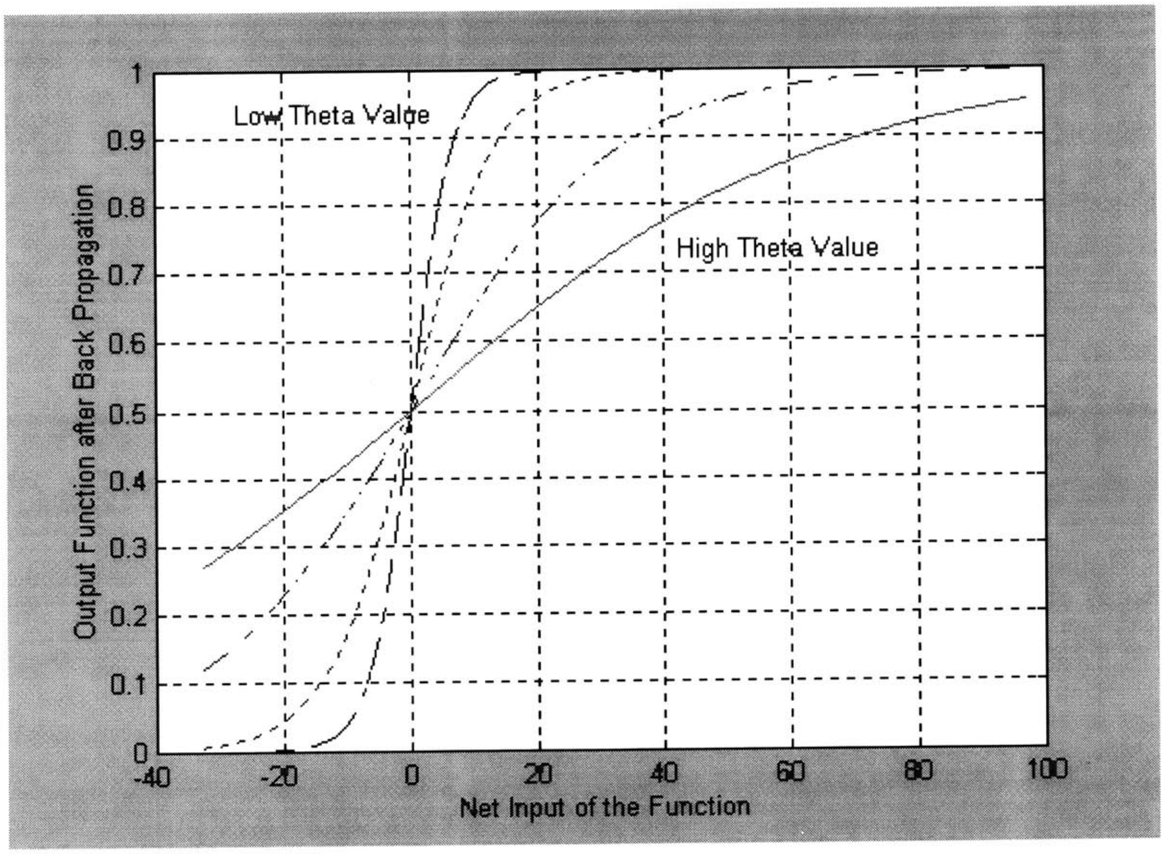


Figure 17: The Sigmoidal Activation Function.

If the error is considered small enough, the weights and thresholds are not adjusted. If however, a significant error is obtained hereby, the weights and thresholds adjusted in the negative gradient direction. A typical weight $w_{lj,i}$, which could belong to any layer, is adjusted from its old value $w_{lj,i}^{old}$ to its new value $w_{lj,i}^{new}$, according to:

$$w_{lj,i}^{new} = w_{lj,i}^{old} + \Delta w_{lj,i} \qquad \textbf{Equation (23)}$$

where

$$\Delta w_{lj,i} = \eta \delta_{li} x_{l-1,i} \qquad \textbf{Equation (24)}$$

δ_{li} is the error in the output of the i^{th} node in layer l and η is defined as the "learning rate". The error δ_{li} is not yet known but it must be constructed from the known error δ_{ki} at the output layer. The errors are passed backwards through the net and a training algorithm uses the error to adjust the connection weights moving backwards from the output layer, and then layer by layer. To overcome the intolerable oscillation problem, a momentum term is usually introduced into the update rule:

$$w_{lj,i}^{new} = w_{lj,i}^{old} + \eta \delta_{lj,i} x_{l-1,i} + \alpha \Delta w_{lj,i} \qquad \textbf{Equation (25)}$$

The variable w can be defined as the displacement, acceleration or other quantitative value on an arbitrary node of the structural component.

3.3.4 Use of Neural Network for Structural Optimization and Control

This neural network concept can be applied and gradually improve the design algorithm of the CAE methodology for structural optimization and control [21]. For example, with the learning results obtained by the MLP model, engineers can avoid trial and error on the redundant computer programming codes and figure out the optimal solution of a specific program regarding structural engineering.

Since artificial neural networks are providing to be an effective tool for pattern or data recognition, the basic idea in the neural based approach is to train a network with changes in quantities describing the dynamic behavior of structure. This implies that each pattern represents the computed changes of the response spectrum, natural frequencies, and mode shapes due to dynamic loading [8]. To train the neural network, the pattern values are usually used as inputs and the damage location and size of the structure are usually used as the outputs. In conclusion, the training of a neural network with appropriate data containing the information about the cause and effects is a key requirement in this methodology.

3.4 Neural Networks and Structural Design

A current trend in scientific and engineering computing is to use neural network approximations instead of polynomial approximation or other types of approximation involving mathematical function [10]. The goal of this chapter is to examine the effects of using under-determined neural network approximations and investigate the effects of design selection on the quality of neural network approximations. Moreover, it is our intention to compare the computing time required training neural networks and the time to develop polynomial approximation [18].

3.4.1 Neural Networks as Approximation

Approximations have been used in engineering since the profession's inception. Very recently, the use of neural networks as approximations has been popular [17]. Since neural networks mimic in some sense the working of the brain, which is with incredible conceptual and computational capacity, neural networks as approximators have been attributed with numerous supposed advantages over other types of approximators.

Consider a structural engineering problem with n design variables, such as displacement, force, moment, stress, strain and acceleration of dynamic response, the components of vector $\{x\}=\{x_1, x_2,\ldots x_n\}^t$. A total of N design points will be considered: $\{x\}_j$, j=1,... N. At the design point {x}j, let yj be the value of the function to be approximated. The pairing of $\{x\}_j$ and y_j is referred as the jth training pair. Let $y^\wedge_j$ be the value of the approximating function. The approximating function $y^\wedge$ should closely match the function, y, not only at the designs, $\{x\}_j$, but over the entire region of interest. At the design points when δ^2= the sum of the squares of the residuals is small where:

$$\delta^2 = \sum_1^N (y_i - \hat{y}_i)^2 \qquad \textbf{Equation (26)}$$

Let y_bar be the average value of y at the design points. In this study, one measure of the closeness of fit to be considered is the non-dimensional value v where:

$$v = \frac{\sqrt{\dfrac{\sum_1^N (y_i - \hat{y})^2}{N}}}{\bar{y}} \times 100 \qquad \textbf{Equation (27)}$$

The coefficient v is the non-dimensional root mean square (RMS) error at the design points. Thus, v=0 is a necessary and sufficient condition that the approximating function fits the actual function at the N design points.

While the initial motivation for developing artificial neural network was to derive computer models that could imitate certain brain function of making decision in sophisticated structural design, it can be thought of as another way of developing approximations, such as response surfaces [22]. Typical neural network is with only one hidden layer which has been used previously to develop response surfaces. And with enough nodes on the hidden layer, it is possible to obtain the approximation of any continuous function.

3.4.2 Polynomial Approximation

To ascertain the desirability of using neural network approximation, the criteria employed in comparing the quality of approximation is hereby discussed. This section is followed by a brief example of the methodology for making neural network and polynomial approximation [12]. Polynomial approximation can be made using an $m=k+1$ term polynomial expression:

$$\hat{y} = b_o + b_1 X_1 + \ldots b_k X_k$$ **Equation (28)**

where X_j is some expression involving the design variables, such as displacement, reaction force and moment at the joint. For example, a second order polynomial approximation in two variables could be of the form:

$$\hat{y} = b_0 + b_1 X_1 + b_2 X_2 + b_3 X_1^2 + b_4 X_1 X_2 + b_5 X_2^2$$ **Equation (29)**

Values of the function to be approximated at the *N* design points can be used to determine the *m=k+1* undetermined coefficients in the polynomial expression. For *N* design points, Equation (29) yields:

$$\begin{Bmatrix} y_1 \\ y_2 \\ \dots \\ y_N \end{Bmatrix} = \begin{bmatrix} 1 & X_{1_1} & \dots & X_{k_1} \\ 1 & X_{1_2} & \dots & X_{k_2} \\ \dots & \dots & \dots & \dots \\ 1 & X_{1_N} & \dots & X_{K_N} \end{bmatrix} \begin{Bmatrix} b_0 \\ b_1 \\ \dots \\ b_k \end{Bmatrix} \qquad \textbf{Equation (30)}$$

3.4.3 Neural Network Approximation is Superior to Polynomial Approximation

The numbers of undetermined parameters associated with a neural network are the weights $\{w\}$ and biases $\{\theta\}$ of the network. The numbers of undetermined parameters associated with a polynomial approximation are the coefficients $\{b\}$. In a general sense, the performance of an approximation depends upon the number of undetermined parameters associated with the approximation. The following example elucidates this point.

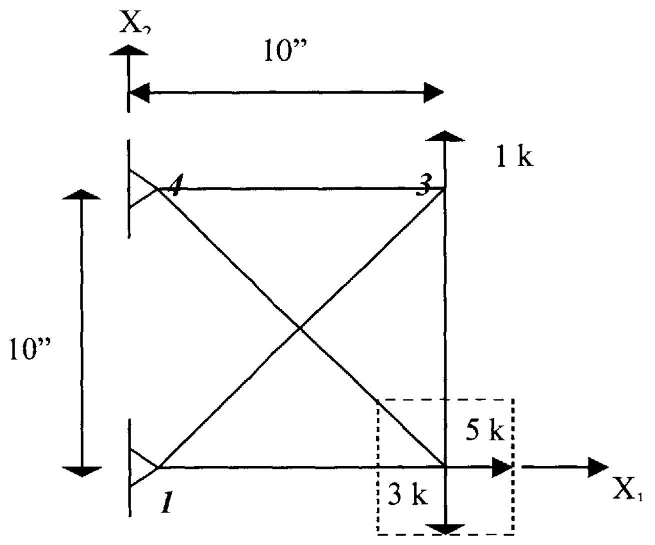


Figure 18: Five Bar Truss

Consider the five bar truss of Figure 18, let VOL be the minimum volume of this truss structure which satisfies the stress and stability constraints. We can use neural network or polynomial approximation to develop the functional relationship between VOL and x_1 and x_2 coordinate of node 2 of the truss [23]. At the mean time, the approximation can be compared to the exact function at meshing points by Finite Element Method. Referring to Figure 18, the approximation considered were:

1. Polynomial approximations of order 2,3,4, and 5.
2. Neural networks with 3,5, and 7 nodes on the hidden layer.

3.4.4 Parallel Training of Neural Networks for Finite Element Mesh Generation

A structural design problem concerned with finite element mesh generation can be solved using the parallel neural network software. The application can be involved in a parallel processing implementation for neural computing and its application to finite element mesh design. Finally, Artificial Neural Networks (ANN) can be introduced to structural control of complex systems [20].

The human brain's powerful thinking, remembering, and problem-solving capabilities have inspired many scientists and engineers to attempt computer modeling of its operations. In an artificial neural network (ANN), the artificial neurons or the processing unit may have several input paths, corresponding to those dendrites [24]. The combined value is then modified by a transfer function. This function may be for example as simple as a linear threshold function which only passes information if the summed activity level reaches a certain threshold, or it may be a continuous function such as a sigmoidal activation function [7]. The output value of the transfer function is generally passed directly to the output path of the unit. Each connection has a corresponding weight where the signals on the input lines to a unit are modified or weighted prior to being summed. The summation function has already shown in Figure 19 is a weighted summation:

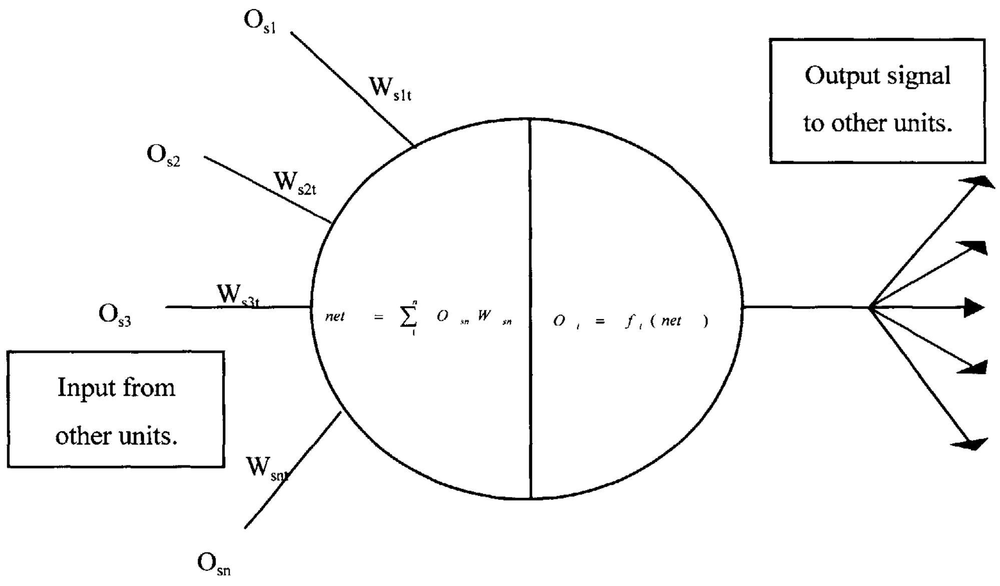


Figure 19: The Input and Output of Neural Network

3.4.5 The Back Propagation Learning Algorithm of the Neural Network

The neural network based on the Back Propagation (BP) theory, which follows the General Delta Rule [18], can be applied to structural control. It is carried out when a set of input training pattern is propagated through a network consisting of an input layer, a hidden layer and finally the output layer. According to the definition, the pattern is simply an input-output row vector in the entire matrix of I/O patterns which represent the training data structure.

Each layer has its corresponding units and weight connection. The weight values are originally initialized randomly for each connecting unit. During the forward propagation, each input data is multiplied by the weight value between the source unit and the subsequent target unit in the following layer. Thus the sum is defined as:

$$net_{pt} = \sum_{s} w_{st} o_{ps}$$ **Equation (31)**

Where w_{st} is the connecting weight from the source s layer to the target t layer and o_{ps} is the output produced for pattern p as result of the input $o_{p(s-1)}$. This net sum is then applied to an activation function (commonly a sigmoidal activation function), the product of which is an output signal [10]. This output signal is the input to the units in the subsequent layer. The output is defined by:

$$o_{pt}=f_t(net_{pt})$$ **Equation (32)**

where $f_t(net_{pt})$ is the activation function. When a sigmoidal activation function is used, the output may be defined by the following expression:

$$o_{pt} = \frac{1}{1+e^{\frac{-(net_{pt}+bias)}{\theta}}}$$ **Equation (33)**

where bias serves as an imaginary weight of a unit that is always firing (i.e. $o_{pt}=1$). The effect of a positive bias is to shift the activation function to the right along the horizontal axis and q modifies the shape of the sigmoid. The sigmoidal function has already illustrated in Figure 17 along with the Matlab code in Appendix [5].

Chapter IV

CASE STUDIES IN STRUCTURAL ENGINEERING USING MATLAB

Matlab is very useful in the study of structural dynamics and control. Especially, it is very easy to write the code and compile at the same moment to examine the output data and scientific plotting. Before the useful tools for Computer-Aided Engineering being introduced, it is very hard to have a comprehensive understanding of what the response curve looks like of a structure under dynamic loading or ground motion [6]. Unlike other professional programming languages such as Fortran, C/C++ and Java, Matlab is very powerful in matrix operation and control toolbox, which are particularly essential in the analysis of preliminary structural design [9].

4.1 Ground Motion Excitation

A building model has five equally spaced floors as shown in Figure 20. Assume the ends of columns to be clamped at each floor. Each column is made of aluminum with dimension give along with the building side-view plotting. The force per unit transverse displacement of one column is given by:

$$k = \frac{12EI}{L^3}$$ **Equation (34)**

$$I = \frac{wt^3}{12}$$ **Equation (35)**

A clamp is a device designed to bind or constrict or to press two or more parts of the structural elements together so as to hold them firmly under the dynamic loading [13]. With all the parameters defined, we can use Matlab to solve the following problems:

(i) Find the equation of the motion.

(ii) Use the Rayleigh quotient to estimate the lowest natural frequency.

(iii) Find the natural frequencies and mode shapes using Matlab program.

From the output of the Matlab code listed in Appendix [7], the answers of the previous three questions can be easily obtained. The equation of motion can be written as:

$M\ddot{u} + ku = P(t)$ where

$$M = 0.1740 \times \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$K = 2.7168 \times 10^3 \times \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}$$

Equation (36)

From the Theory of Rayleigh Quotient, one can easily obtain the lowest natural frequency ($37.6752Hz$) by doing simple matrix operation by Matlab [8]:

$$\omega_1 \cong \sqrt{\lambda_1} = \sqrt{\frac{v_i^T K v_i}{v_i^T M v_i}}$$

Equation (37)

Now we are going to discuss the application of model superposition to ground excitation of structure. Assume the absolute ground motion for this 5-story model building is specified as:

$$\ddot{u}_g = 5\left(\frac{m}{S^2}\right)\cos\left(\frac{\omega_1}{2}t\right) \qquad \textbf{Equation (38)}$$

starting at t=0. Assume the damping for all modes is 5%;

(a) Compute the peak relative displacement response of the top floor due to the contribution of the first mode only.

(b) There are two measured numbers of modes necessary to obtain good accuracy using model superposition for the relative displacement formulation:

$$\sum_{i=1}^{r}\phi_i\gamma_i \quad \text{and} \quad \frac{\sum_{i=1}^{r}\phi_i\gamma_i}{M_{total}} \qquad \textbf{Equation (39)}$$

How many modes must be included for this structure so that at the top floor this sum would be greater than 0.9? How many modes for relative displacement at the bottom floor? The Matlab code in Appendix [6] solves the above problems and the output graphs are shown in Figure 20.

From the output data and graphs of Matlab program <rayleigh.m> in Appendix [6], we can figure out that the peak response is 1.1cm and at least two modes are required to obtain the good accuracy number (>0.9) for this structure.

Table 3: Data of the 5-Story Building

Number of Modes	Sum_1 Top	Sum_5 Bottom	Mass_Sum
1	1.2517	0.3563	0.8795
2	0.8896	0.6572	0.9667
3	1.0481	0.8648	0.9909
4	0.9850	0.9711	0.9984
5	1.0000	1.0000	1.0000

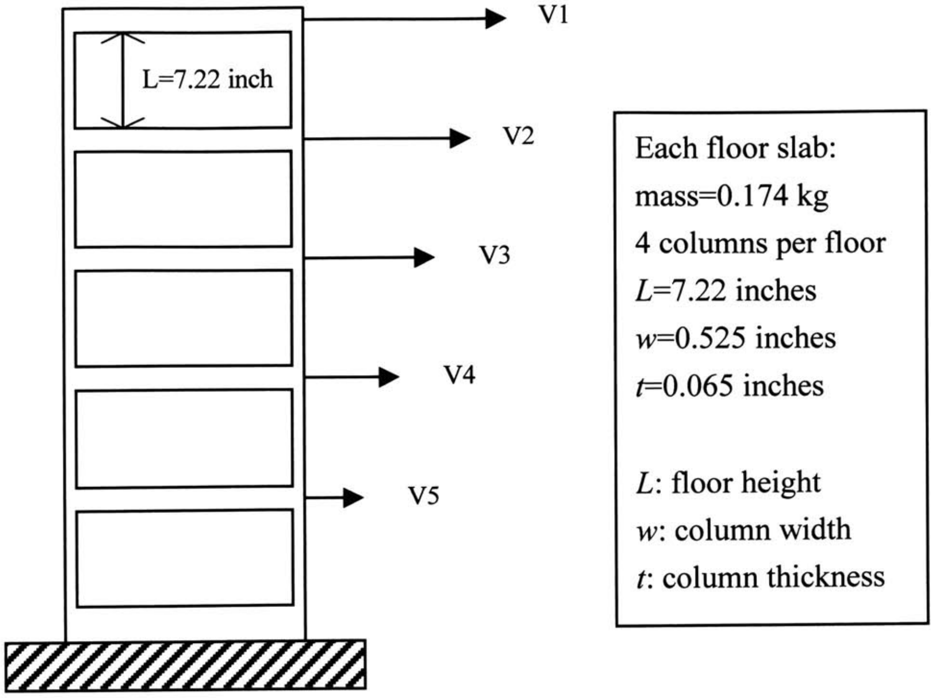


Figure 20: Building Layout

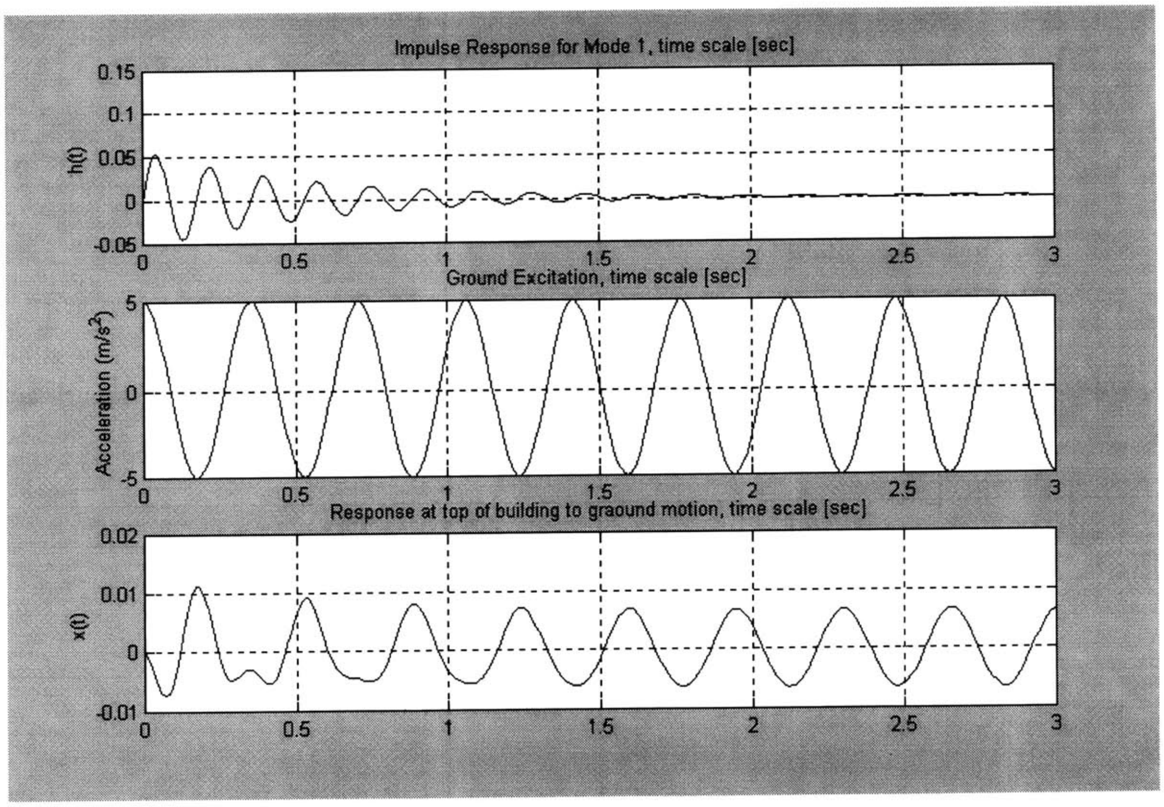


Figure 21: Matlab Response Graph

4.2 Animation of Bending Moment Diagram

It is very convenient to use Matlab to write a program that calculates and plots the bending moment produced by the force F as it moves along the simple-supported bridge shown in Figure 22. Assume that the force F is the weight of a man, namely 700N (approximately 70kg), and the span l=4m. In this program, it is very useful to apply the "`subplot`" function to show the several bending moment diagrams together [5]. Meanwhile, it is possible to write an animation program that shows the diagram of bending moment changes while the load moves along the bridge.

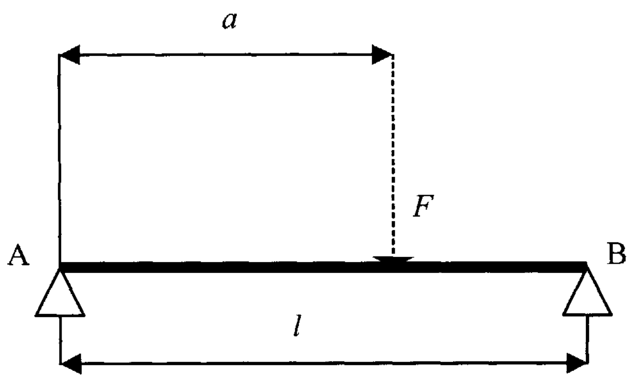


Figure 22: Simply-Supported Bridge

The algorithm of this Matlab program is relatively simple and can be completed by hand-calculation. First of all, one can determine the reaction forces on the end of the bridge supports, which are *F*a/l* and *F*(1-a/l)* respectively. Then by writing double loops in the main function of the computer program, one can easily obtain the magnitude of bending moment on each point and under different loading condition by moving *F* from left end to right end, namely *B=x*[1-a(m)/l]*F.*

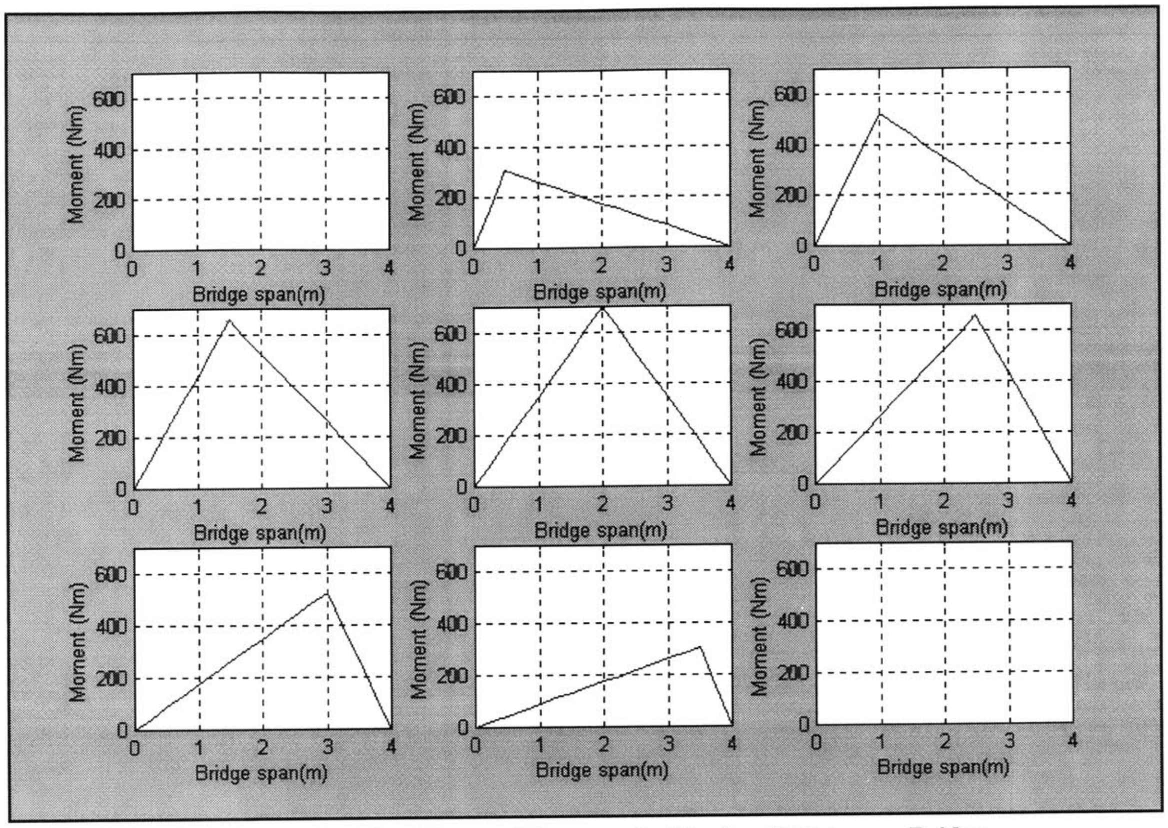


Figure 23: Bending Moment Diagram of a Moving Vehicle on a Bridge

From the above diagram, the bending moment of each loading location is presented by the Matlab output plotting. The maximum moment occurred when the loading is applied at the midpoint of the bridge span. And if the loading is applied on the right and left support, there will be no bending moment taking place. As a consequence, it is very useful to utilize Matlab on the optimization procedure of the structural design and loading control. Besides the fundamental 2-D plotting of the output moment diagram, Matlab also provides the animation function generating the above nine graphs into a series of animation picture, or so called a movie [6]. This effect enables engineering major students to have further enhancement of the moving load applied on a simply-supported structure. Appendix [8] presents the syntax of program called "movie.m" which shows the animation while Appendix [9] presents the nine bending moment diagrams.

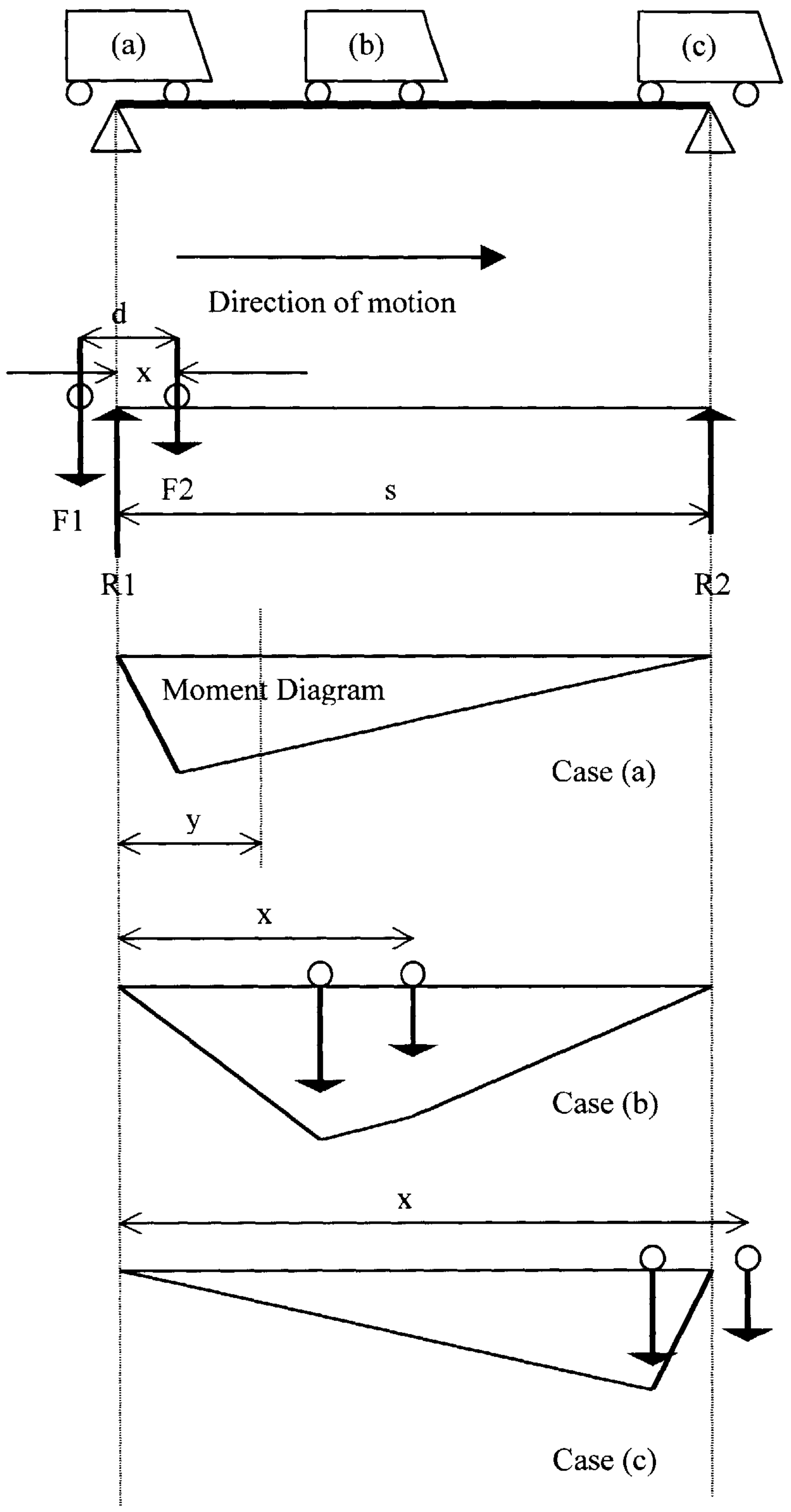


Figure 24: Bending Moment Produced by Vehicle along a Simply-Supported Beam

4.3 Bending Moment Caused by a Moving Vehicle

In recent Computer-Aided Engineering programming tools, better graphic facilities and economic storage of simulation data are two important features to present the result of structural optimization under special loading condition [25]. Besides the concentrated load case discussed before, the bending moment of a simply supported beam, under the influence of a moving twin-axle vehicle, will also be presented in a three-dimensional plot produced with new Matlab graphics.

The simply-supported beam is shown in Figure 24, and let the span be *s*. The distance between the two vehicle axles and the distance of the forward axle from the left-hand support are defined as *d* and *x* respectively. Basically the goal of writing this Matlab program is to write a function that calculates the bending moment acting on the beam as the vehicle moves from left to right. Most importantly, the recent development in Matlab function makes it possible to obtain the 3-D data plotting of moment diagram in conjunction with other parameters which might reinforce engineers' comprehension to the design problem of structural engineering [9].

Before starting the algorithm, it is essential to distinguish three cases as shown in the figure on the previous page. In case (a), only the forward axles of the vehicle entered the beam, while the rear axle is still moving on another beam segment, not concentrated on the one for which calculations are performed. In other words, if a bridge with several spans is under moving traffic load, we assume that each span can be analyzed one by one for the purpose of structural optimization and control [24].

In the second case (b), both axles are within the beam span. However, in the third case (c), the forward wheel moves on another beam, so that the bending moments are entirely due to the rear-axle load. Different equations must be used in the three cases for the computer-programming algorithm. The Matlab code for calculating bending moment is called <bending1.m> in Appendix [10], which was written with clear statements whose significance can be easily read by the users. At the same time, Table 4 helps to explain the basic algorithm and decision making process of this code. Similarly, for other complicated cases, engineers can modify the program to fit specific requirement and obtain preliminary simulation result of the optimal design of structure **[16]**.

From the output data of Matlab program <bending1.m>, once can evaluate the maximum moment to be 7636 Nt-m, which occurs when the vehicle moves very close to the midspan of the simply-supported bridge. With the 3-D surface plot (Figure 26) and contour plot (Figure 27) by Matlab function, structural engineer can have better interpretation of the numerical result.

Table 4: Fundamental Coding Algorithm of the Matlab Program

Loading Cases	***Reaction Force***	***Moment Calculation***
Case (a)	`R1=[1-x(k)/s]*F1`	`If y(l)>=x(k)`
x(k)≤d	`M(l,k)=y(l)*R1`	`M(l,k)=M(l,k)-[y(l)-x(k)]*F1`
Case (b)	`R1=[1-x(k)/s]*F1+`	`If y(l)>=x(k)-d`
	`[1+(d-x(k))/s]*F2`	
x(k)≤s	`M(l,k)=y(l)*R1`	`M(l,k)=M(l,k)-[y(l)-(x(k)-d)]*F2`
Case (c)	`R1=[1+(d-x(k))/s]*F2`	`If y(l)>x(k)-d`
	`M(l,k)=y(l)*R1`	`M(l,k)=M(l,k)-[y(l)-x(k)+d]*F2`

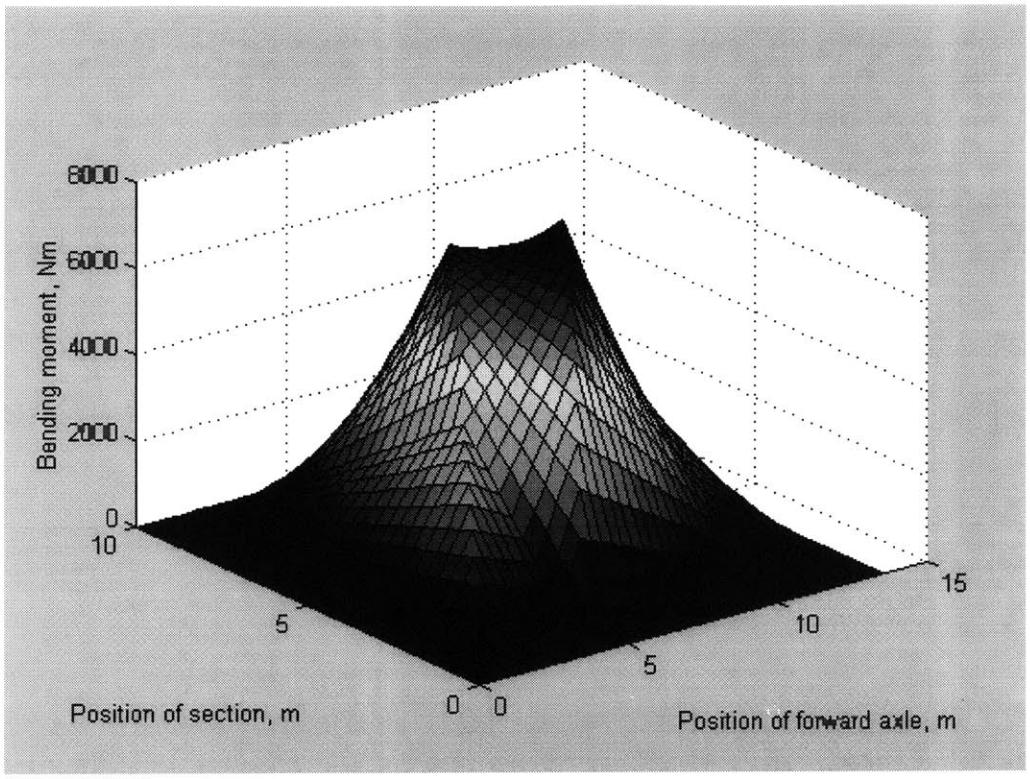


Figure 25: 3-D Contour of the Simulation Data

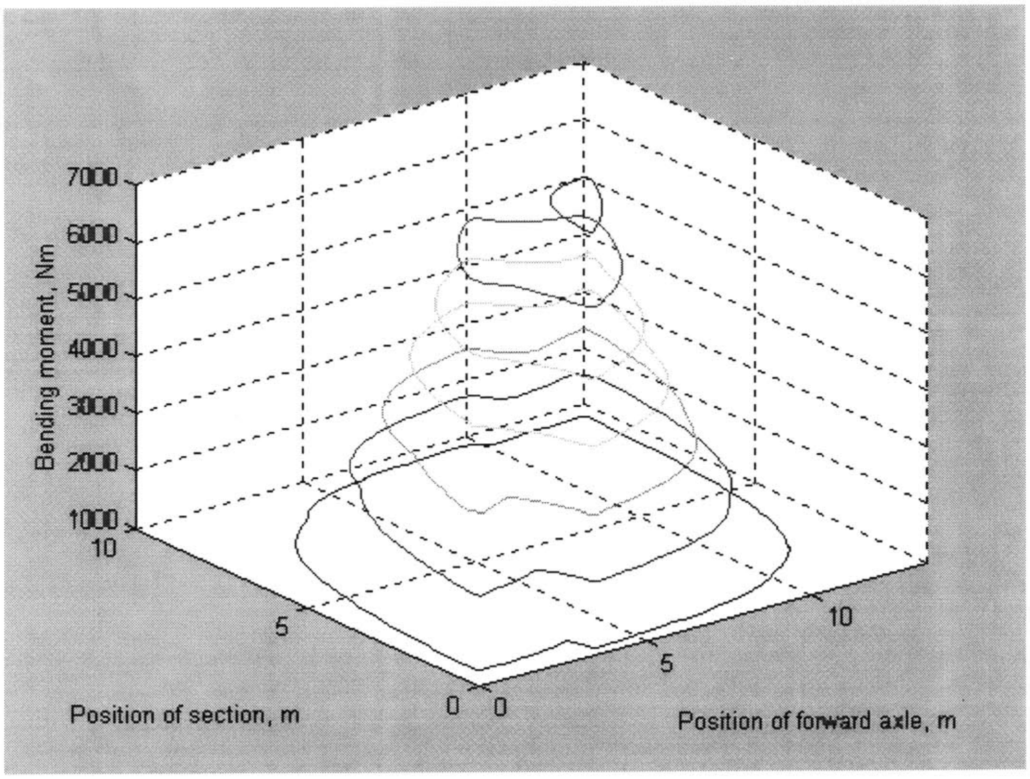


Figure 26: 3-D Spiral Data Plotting

Chapter V

Computer-Aided Engineering To-The-Date

In the past several decades, CAD technology has been improved a lot to simplify drafting and design tasks in Civil and Structural Engineering [26]. Nowadays, with the ever-changing CAE technology, structural engineers and construction managers can work on a project with better coordination. Currently, some easy-to-go graphic user interface (GUI) software packages has been brought to the real engineering world to strengthen the design capability for sophisticated structural engineers. To avoid the tedious calculation and code checking, building and bridge design work can be accomplished by CAE tools from shop-drawing to detail design and layout.

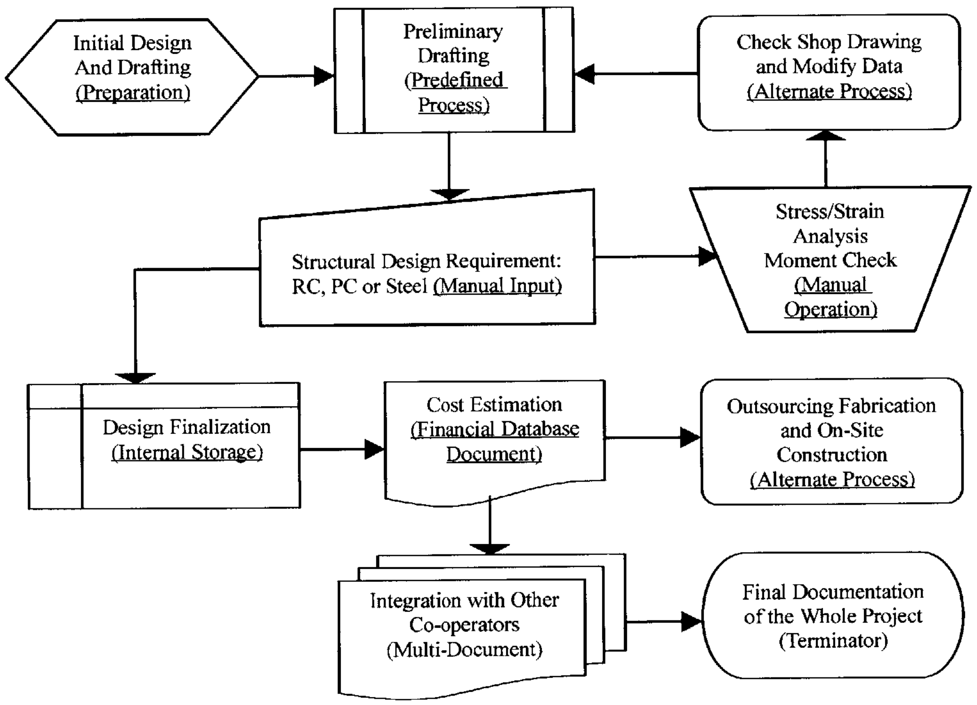
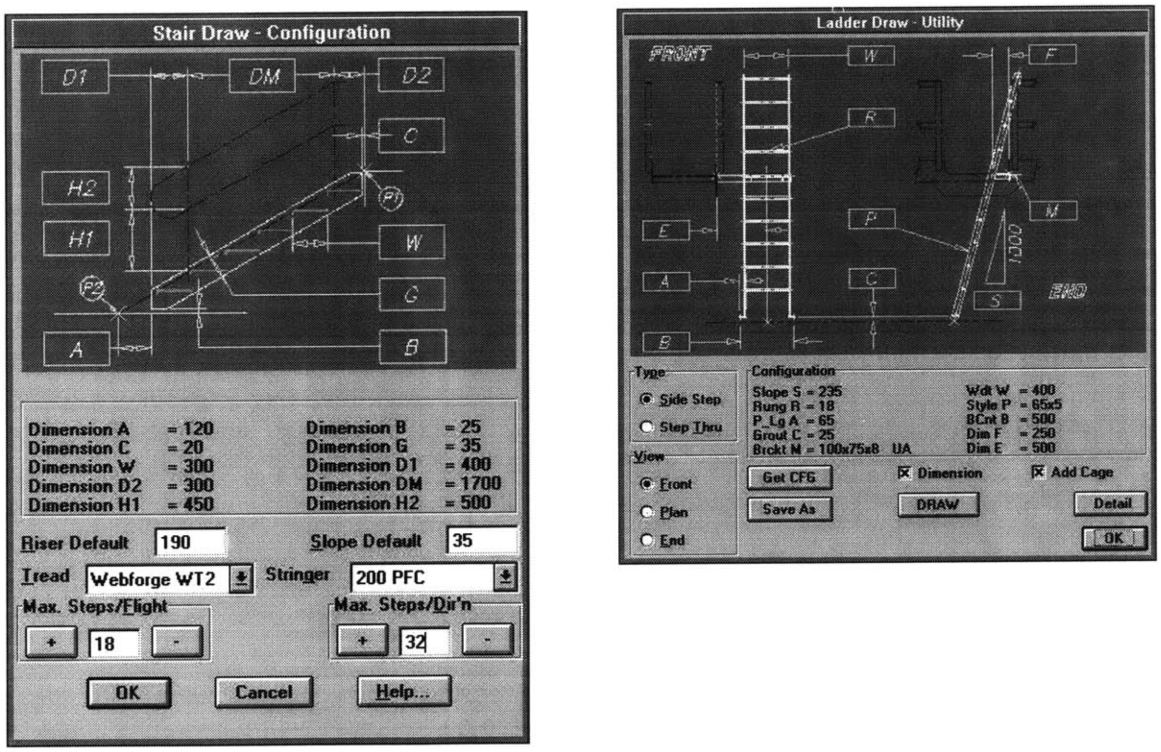

Initial Design And Drafting (Preparation)
Preliminary Drafting (Predefined Process)
Check Shop Drawing and Modify Data (Alternate Process)
Structural Design Requirement: RC, PC or Steel (Manual Input)
Stress/Strain Analysis Moment Check (Manual Operation)
Design Finalization (Internal Storage)
Cost Estimation (Financial Database Document)
Outsourcing Fabrication and On-Site Construction (Alternate Process)
Integration with Other Co-operators (Multi-Document)
Final Documentation of the Whole Project (Terminator)

Figure 27: Flow Chart of the CAE Methodology in Structural Design and Analysis

5.1 Software Development in Civil Engineering Industry

The enhancement of the CAE technology extends its functionality from design to loads estimation. For example, senior engineers are able to download the CAD drawing files from the draft technicians and upload the file on to the structural analysis package, such as SAP2000 and STADD-III [27]. Extra time to redo the geometric mapping and load modeling will be saved and the structural/mechanical simulation related to static and dynamic load can be easily and quickly obtained. Moreover, the data file provided from the CAD system can be implemented into the code checking and selection of construction material.



Dimension A	= 120	Dimension B	= 25
Dimension C	= 20	Dimension G	= 35
Dimension W	= 300	Dimension D1	= 400
Dimension D2	= 300	Dimension DM	= 1700
Dimension H1	= 450	Dimension H2	= 500

Riser Default 190 Slope Default 35

Tread Webforge WT2 Stringer 200 PFC

Max. Steps/Flight 18 Max. Steps/Dir'n 32

Type: [✓ Side Step] Step Thru

View: [✓ Front] Plan End

Configuration

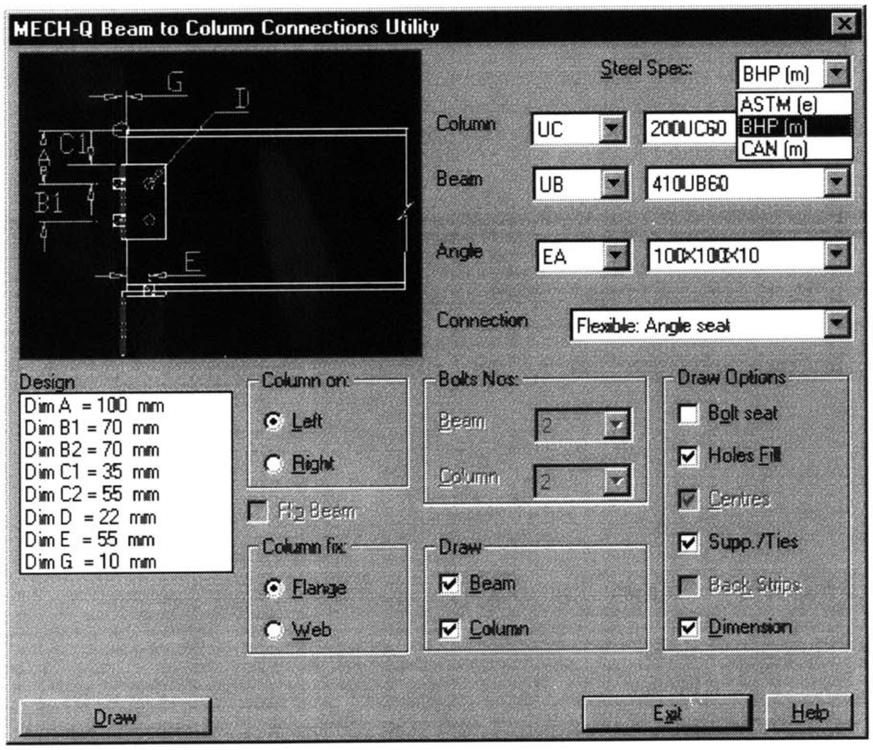
Slope S	= 235	Wdt W	= 400
Rung R	= 18	Style P	= 65x5
P_Lg A	= 65	BCnt B	= 500
Grout C	= 25	Dim F	= 250
Brckt M	= 100x75x8 UA	Dim E	= 500

[✓ Dimension] [✓ Add Cage]

Figure 28: (a) Stair Draw Configuration (b) Ladder Draw Utility

Figure 28 shows the detail design interface of the stair and ladder structure. With auto dimensioning for fabrication, engineers can use the programming design feature to draw the stairs. Then by quickly picking the stair stringers, it will be auto-detailed for the construction managers and site superintendents.

This powerful utility will enable engineers to draw stairs with the configuration set. The routine will draw single-flight and multi-flight stairs in front, plan, and end views and then add the main dimensions and stair data as required [28]. Handrail outline can also be included. To change the drawing configuration, the user could simply pick up (with pointing device) the relevant dimension box on the ladder image and enter the new value in the edit box displayed. The new configuration is then automatically saved on disks.



MECH-Q Beam to Column Connections Utility

Steel Spec: BHP (m) — ASTM (e) / BHP (m) / CAN (m)

Column: UC 200UC60

Beam: UB 410UB60

Angle: EA 100X100X10

Connection: Flexible: Angle seat

Design
Dim A = 100 mm
Dim B1 = 70 mm
Dim B2 = 70 mm
Dim C1 = 35 mm
Dim C2 = 55 mm
Dim D = 22 mm
Dim E = 55 mm
Dim G = 10 mm

Column on: [✓ Left] Right

☐ Flip Beam

Column fix: [✓ Flange] Web

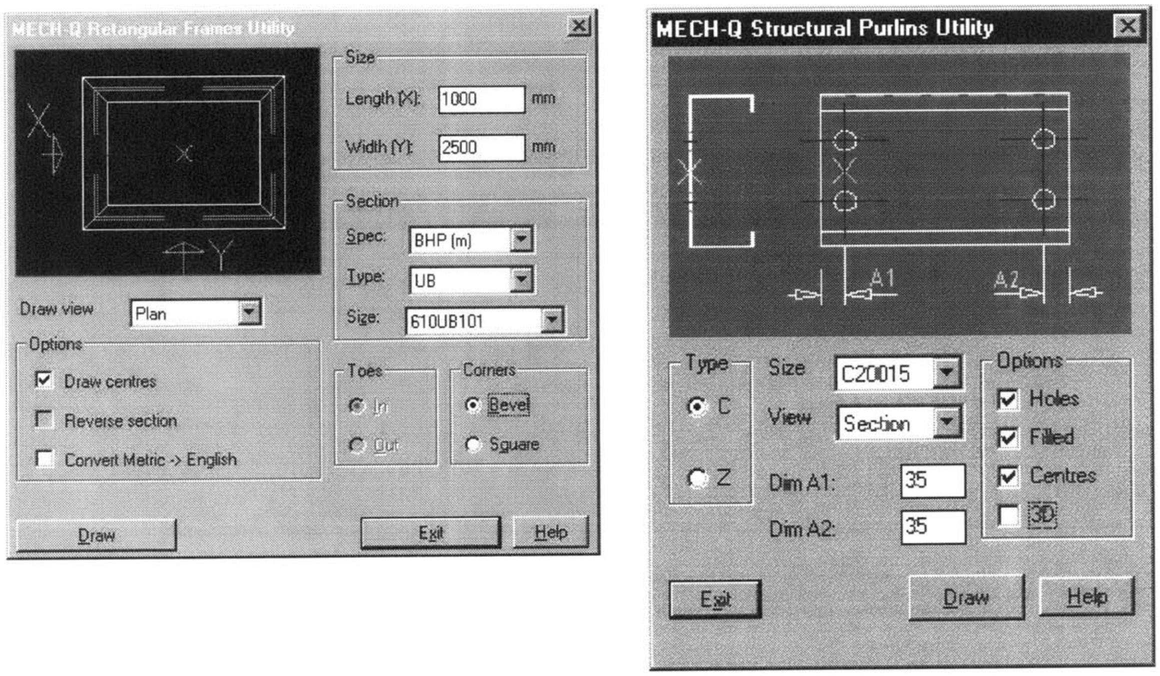
Bolts Nos: Beam 2, Column 2

Draw: ✓ Beam, ✓ Column

Draw Options: ☐ Bolt seat, ✓ Holes Fill, ✓ Centres, ✓ Supp./Ties, ☐ Back Strips, ✓ Dimension

Figure 29: Auto-Dimensioning of the Beam to Column Connection

For international construction projects cooperated by multinational engineering firm, it is very important to provide both Metric (Australian and Canadian) and English (US-ASTM) standard sections on the design platform [1]. The utility mentioned also provides the whole set of unit conversion template. As the result, structural engineers in different countries will have no difficulties sharing the same shop drawing and detail design and redundant calculation can be avoided.


MECH-Q Retangular Frames Utility

Size
Length (X): 1000 mm
Width (Y): 2500 mm

Section
Spec: BHP (m)
Type: UB
Size: 610UB101

Draw view: Plan

Options
☑ Draw centres
☐ Reverse section
☐ Convert Metric -> English

Toes: [✓ In] Out

Corners: [✓ Bevel] Square

MECH-Q Structural Purlins Utility

Type: [✓ C] Z
Size: C20015
View: Section
Dim A1: 35
Dim A2: 35

Options
☑ Holes
☑ Filled
☑ Centres
☐ 3D

Figure 30: (a) Structural Element Dimensions (b) Structural Layout Illustration

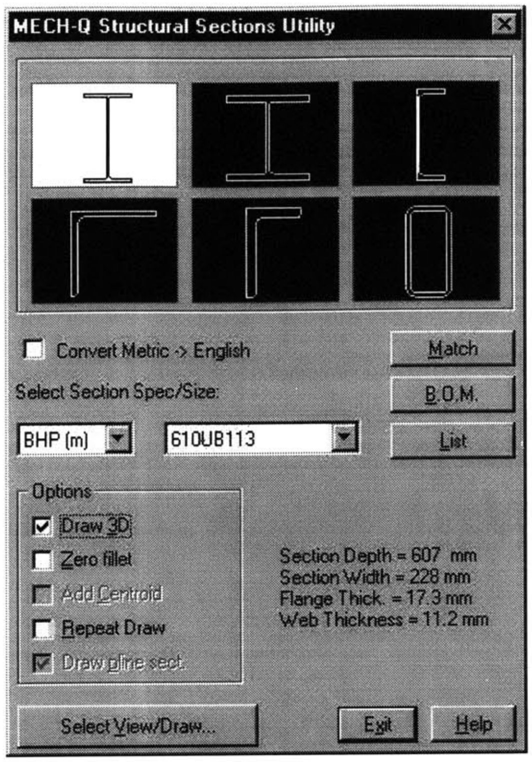
5.2 Impact of CAE to Structural Design Based on Optimization Concept

Software engineers attempt to improve the functionality and performance of the user-friendly CAD interface for structural engineers [29]. Since most of the detail design work deals with both geometric plotting and numerical simulation, it is very critical to embed all functional criteria in the software platform. But with the increase of computer hardware memory for data operation and advanced software for program execution, one-step drafting and simulation is no longer a daydream in the real world of structural design and analysis projects.

As shown in Figure 29-30, this CAD utility draw sections in plan and end views of all standard sizes both 2-D and 3-D. Meanwhile, section detail and property can also be stored as X-DATA. Section drawn can be listed for section properties including section size, unit mass and modulus. The plan and end views of structural welded frame have been presented. Section options such as C or Z shapes and bolt-holes can be chosen arbitrarily and spacing can also be defined [13]. Additionally, the utility also handles weighted bill of material (BOM) function in the entire construction project.

5.3 Overall Design Finalization

Structural engineers always seek alternative design methodology for the time-consuming and labor-consuming procedure. Especially, it is not simple to check the section property for each element while doing structural or mechanical simulation. With the Structural Section Utility presented in Figure 31, a complete basic range such as W-shapes, S-shapes, M-shapes, C-shapes, and L-shapes (equal/unequal legs) can all be applied to the design [27]. Furthermore, for special structural design, HP-shapes, MC-shapes and English Rectangular Hollow Sections would be supplementary choices. Moreover, options include checking section centroids and detail properties converted from metric size to English (imperial) and vise versa. With this CAE tool, it is unnecessary to check the design handbook and code in every minute.



Convert Metric -> English [ ]
Select Section Spec/Size: BHP (m) 610UB113

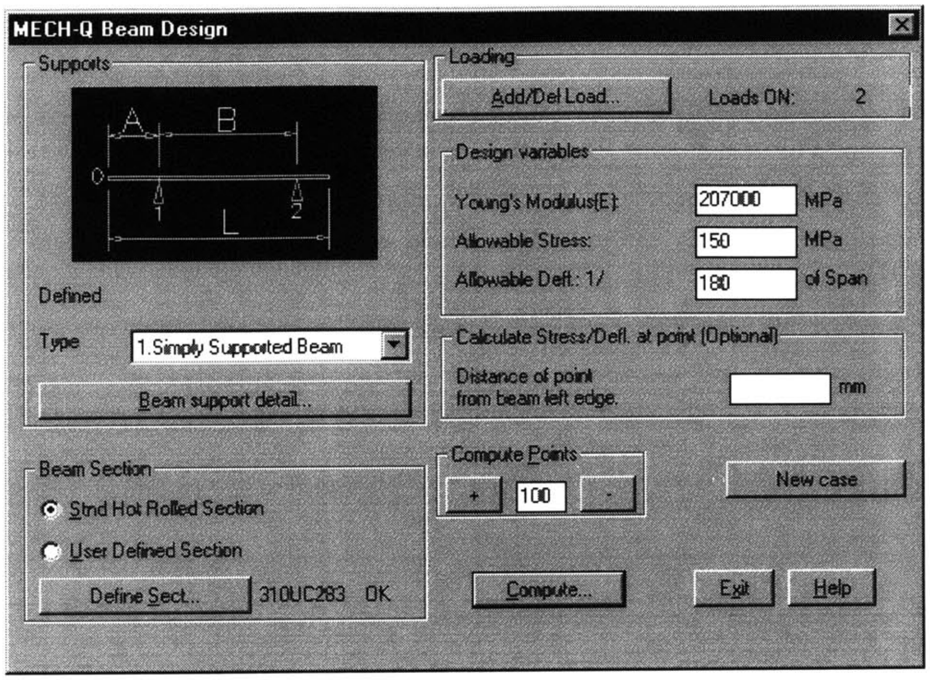
Options
[✓] Draw 3D
[ ] Zero fillet
[ ] Add Centroid
[ ] Repeat Draw
[✓] Draw pline sect.

Section Depth = 607 mm
Section Width = 228 mm
Flange Thick. = 17.3 mm
Web Thickness = 11.2 mm

Figure 31: Defining the Structural Section and Showing the Properties.

5.4 Loading Specification for Simulation

After defining the material and section properties for the structural element, auto-dimensioning to fabrication detail will be fully customized and saved on the disk. With the GUI template, beam member support types and beam to column connections can be chosen and confirmed according to the design requirement [28]. For example, flexible and rigid connection can be applied to the structure layout by simply adding the icon or arrow element. Support types for beam segments include simply supported, cantilever, propped-cantilever, built-in supported and so on. The output data will be reaction at support, deflection and strain/stress plot of the structural element under loading. For more detail, minimum steel section required for the given load cases are also obtained from the calculation results [30].



Defined Type: 1.Simply Supported Beam

Loads ON: 2

Young's Modulus(E): 207000 MPa

Allowable Stress: 150 MPa

Allowable Defl.: 1/ 180 of Span

Distance of point from beam left edge: mm

Compute Points: 100

[✓ Stnd Hot Rolled Section] User Defined Section

310UC283 OK

Figure 32: Mechanical Property and Loading Condition of the Structure.

5.5 Summary of CAE Software Design Guideline

Most CAE Tools represent the state-of-the-art in structural engineering. For instance, 3-D finite element technology usually applies on linear and nonlinear components, such as frame, trusses, shell or plate. The software programs support a wide range of load types, including prestress condition. Especially, the shell element is defined 3-D and includes in-plane rotational stiffness components [30]. The nonlinear type element may be used for dynamic analysis involving base isolation, dampers and discrete systems. Static loading options allow for gravity, pressure in addition to nodal loading with specified forces or displacements. Dynamic loading can be in the form of multiple base response spectrums, or multiple time varying loads such as base excitation [22]. Furthermore, vehicle life load generation for trucks or trains should also be available.

Chapter VI

Conclusion And Future Outlook

In the previous chapters, several CAE methodologies have been examined for their possibilities on structural optimization and control. For example, the concept of neural network improves the design and analysis of large-scale structural systems. Besides, Matlab programming language is an alternative user-friendly CAE Tool for preliminary design and analysis of structural or mechanical control systems.

Although there have been plenty of handful software packages for graphic drafting, structural design and analysis, project management tools as well as geographic information system (GIS), civil or structural engineers still encounter a lot of unsolved issues [29]. For example, CAE Tools for structural optimization and control have not been some of the shelf software products that can be customized to fit individual users.

6.1 Software Development and Information Technology in Civil Engineering

Over the course of history, civil engineers have made a significant contribution to improve the environment by designing and constructing major structures and facilities. The profession requires a lot of precision because most of the time the final result of any project will directly or indirectly affect people's lives and safety. As a result, software usage in this industry reduces the complexity of a lot of challenging projects.

Recently, Computer-Aided Engineering along with Information Technology gradually integrate with structural engineering industry. Software developers continuously produce user-friendly products to expedite the engineering work. For example, one of the most popular structural engineering software STADD-III, is comprehensive in model development, analysis, design, visualization and verification [29]. The entire data may be generated either graphically or by typing English language based commands.

Other structural analysis softwares are available in the market, namely SAP2000, a state-of-the-art three-dimensional finite element technology for structural engineering [31]. ETABS offers sophisticated 3-D analysis and design for multistory building structures. SAFE provides analysis and design for concrete slabs and basemat/fundation structures. Depending on what analysis the engineers are looking for, CAE softwares can be utilized and provide the most accurate results based on the primary design goals.

6.2 Building on the Web-- Facing the New Trend

The construction industry has been slow to adopt new technology, but as projects become more complex and out-of-state building partners become more commonplace, that is beginning to change. Today's projects can have 10 to 100 different partners, including designers, contractors, architects, lawyers, tenants, and insurers, all of whom need to share information on 10,000 or more details to complete a building.

While much of the information is generated on computers, it generally has to be shared by fax, mail, phone, or corporate e-mail systems. But in the past few years, Macomber's Collaborative Structure Inc. of Boston, and a dozen or so other companies, have created Internet/Intranet project management Web sites [32], on which all the partners can see, comment on, and amend the details of their project.

Traditionally, the lead project manager might have a schedule, a drawing, and a Performa (a spreadsheet with revenue and expenses) to be distributed to many people. Big binders are mailed to all parties for comment, and then the project manager manually compiles the comment, recopied the document, and resends it. Now that can all be done on-line. The advantages of Web project management include notifying workers of change more quickly, saving paper and mail costs, and making fewer mistakes. As a result, the structural design and construction firms do not need to deliver their blueprints via Federal Express or UPS.

6.3 High-Technology Revolution Comes to Structural Engineering

Some civil engineers like to look at the Web services as building a "new economy" business. Web project management following the building design indeed grows from the root of "old economy" business, construction. Nevertheless, construction really is an information business. One needs to know what point the schedule is at now, and what goes where. However, the architecture, engineering, and construction industries have lagged others in automation, because the products and services available haven't given builders the features they need to do their jobs better.

Ideally, CAE tools or solutions should be developed by people who came out of engineering and construction rather than the computer industry, so it is set up with a language used in the design and construction industry [29]. Since many construction projects last only12 to 24 months, it makes more sense for an independent party to handle project management, as opposed to the companies building their own computer networks.

For instances, FirstLine organizes all types of documents in a common way, and links related documents by a "thread map" that allows similar e-mails or drafts or blueprints to be pulled by topic, in chronological order from the most recent back [32]. It creates a permanent record of the decision-making process that can not be changed, and which is given to engineers of the sites on a CD-ROM once the project is completed. One of the most useful features is that it cuts down papers. For most users, the payback is efficiency and time, more than money.

Web project management also makes it easier for partners in a remote location across the country, or the world, to collaborate. Also, two or three years from now, technology will be a significant part of doing CAE and Web management on the structural design and civil engineering construction. In conclusion, future civil engineers should lead construction industry in technology revolution. In summary, it is very important for civil and structural engineers to face the new trend in the design and construction industry.

[REFERENCES]

1. B. H. T. Topping, *"Developments in Computer-Aided Design and Modeling for Civil Engineering"*, Civil-Comp Press, 1995.
2. B. Hassani and E. Hinton, *"Homogenization and Structural Topology Optimization"*, Springer, 1999.
3. M. P. Bendsoe, *"Optimization of Structural Topology, Shape and Material"*, Springer, 1995.
4. K. J. Bathe, *"Finite Element Procedures"*, Prentice Hall, 1994.
5. A. Biran, M. Breiner, *"Matlab for Engineers"*, Addison-Wesley Publishing Company, New York, 1995.
6. K. Ogaka, *"Solving Control Engineering Problems with Matlab"*, Prentice Hall, New York, 1994.
7. K. Ogaka, *"Designing Linear Control System with Matlab"*, Prentice Hall, 1994.
8. A. K. Chopra, "*Dynamics of Structures-- Theory and Application to Earthquake Engineering*", Prentice Hall, 1989.
9. R. H. Bishop, *"Modern Control System Analysis and Design Using Matlab"*, Addison-Wesley Publishing Compamy, 1993.
10. B. H. V. Topping, "*Developments in Computational Techniques for Structural Engineering*", Civil-Comp Press, 1994.
11. S. Rajeev and C. S. Krishnamoorthy, "*Discrete Optimization of Structure Using Generic Algorithms*", Journal of Structural Engineering, ASCE, Vol. 118, No. 5, 1992.
12. W. McGuire and R. H. Gallagher, *"Matrix Structural Mechanical System"*, JohnWiley & Sons, Inc., 1996.
13. C. G. Salmon and J. E. Johnson, *"Steel Structures, Design and Behavior"*, Harper Collins Publisher, 1990.
14. G. I. N. Rozvany, *"Optimization of Large Structural Systems"*, Kluwer Academic Publishers, 1991.
15. G. I. N. Rozvany, *"Topology Optimization in Structural Mechanics"*, SpringerWien, 1997.
16. *"Guideline to Structural Optimization"*, ASCE Manual and Reports on Engineering Practice No. 90, 1997
17. A. Guran, *"Recent Development in Stability, Vibration and Control of Structural Systems"*, The American Society of Mechanical Engineering, 1993.

18. B. H. V. Topping and A. I. Khan, *"Neural Network and Combinational Optimization in Civil and Structural Engineering"*, Civil-Comp Press, 1993.
19. R. S. Guttalu, *"Mechanics and Control"*, Plenum Press, 1992.
20. I. Smith, *"Artificial Intelligence in Structural Engineering--Information Technology for Design, Collaboration, Maintenance and Monitoring"*, Springer, 1998.
21. R. Bulirsch, D. Kraft, "Computational Optimal Control", Birkhauser Verlag, 1995.
22. P. J. Pahl, H. Werner, "Computing in Civil and Building Engineering", A. A. Balkema, 1995.
23. B. H. V. Topping and M. Papadrakakis, *"Advanced in Structural Engineering Computing"*, Civil-Comp Press, 1994.
24. A. Baratta and J. Rodellar, "Proceedings of the First European Conference on Structural Control", World Scientific, 1996.
25. D. M. Frangopol and F. Y. Cheng, *"Advanced in Structural Optimization"*, American Society of Civil Engineers, 1996.
26. http://www.autodesk.com/products/aectools/index.htm, AutoCAD official site.
27. http://www.csiberkeley.com/Tutorials/SAP2000AtoZ/SAP2000AtoZ.htm Computers and Structures, Inc. SAP2000: Structural Analysis Program Developer. 1997 Category: Structural Platform: Windows 95/NT Summary.
28. http://www.asvic.com.au/mq-str.htm, A/E/C/ SYSTEM in LA, developing "MECH-Q UTILITIES" for advanced structural design and analysis.
29. http://www.ite.poly.edu/mg/mg737/15049982/15049982enency737.html Software Development and Information Technology in Civil Engineering.
30. V. S. Hernandez, C. A. Brebbia, *"Computer Aided Optimum Design of Structure"*, Computational Mechanics Publications, 1997.
31. http://nisee.ce.berkeley.edu/software_and_data/eng_soft/software/sap2000.html National Information Service for Earthquake Engineering, University of California, Berkeley.
32. http://www.boston.com/dailyglobe2/167/business/Building_on_the_Web+.shtmlBoston Globe article: Building on the Web.

[APPENDIXES]

Appendix [1]

```
function truss(action)

%TRUSS  Animation of a bending bridge truss.

% Information regarding the play status will be held in
% the axis user data according to the following table:

play= 1;
stop=-1;

if nargin<1,
    action='initialize';
end;

if strcmp(action,'initialize'),
   oldFigNumber=watchon;

   figNumber=figure('Name','Bending Truss',...
                    'NumberTitle','off','Visible','off',...
                    'BackingStore','off','Colormap'[]);

   axes( 'Units','normalized', ...
        'Position',[0.05 0.05 0.75 0.90], ...
        'Visible','off', ...
        'NextPlot','add');

   text( 0,0,'Press the "Start" button to see the...
         Bending Truss demo','HorizontalAlignment','center');
   axis([-1 1 -1 1]);

%==================================================================
% Information for all buttons
      labelColor=[0.8 0.8 0.8];
      yInitPos=0.90;
      xPos=0.85;
      btnWid=0.10;
      btnHt=0.10;

% Spacing between the button and the next command's label
      spacing=0.05;

%==================================================================
% The CONSOLE frame
      frmBorder=0.02;
      yPos=0.05-frmBorder;
      frmPos=[xPos-frmBorder yPos btnWid+2*frmBorder 0.9+2*frmBorder];
      h=uicontrol('Style','frame','Units','normalized', ...
            'Position',frmPos,'BackgroundColor',[0.5 0.5 0.5]);
%==================================================================
% The START button
      btnNumber=1;
      yPos=0.90-(btnNumber-1)*(btnHt+spacing);
      labelStr='Start';
      cmdStr='start';
      callbackStr='truss(''start'');';

% Generic button information
      btnPos=[xPos yPos-spacing btnWid btnHt];
      startHndl=uicontrol('Style','pushbutton', ...
                   'Units','normalized', ...
                   'Position',btnPos, ...
                   'String',labelStr, ...
                   'Interruptible','yes', ...
                   'Callback',callbackStr);

%===================================================================
% The MODE popup button
      btnNumber=2;
    yPos=0.90-(btnNumber-1)*(btnHt+spacing);
    textStr='Mode';
    popupStr=reshape(' 1  2  3  4  5  6  7  8  9 10 11 12...
                 ',3,12)';

% Generic button information
    btnPos1=[xPos yPos-spacing+btnHt/2 btnWid btnHt/2];
    btnPos2=[xPos yPos-spacing btnWid btnHt/2];
    popupHndl=uicontrol('Style','text', ...
                         'Units','normalized', ...
                         'Position',btnPos1, ...
                         'String',textStr);

    btnPos=[xPos yPos-spacing btnWid btnHt/2];
    popupHndl=uicontrol( 'Style','popup', ...
                          'Units','normalized', ...
                   'Position',btnPos2, ...
                   'String',popupStr);

%===================================================================
% The STOP button
    btnNumber=3;
    yPos=0.90-(btnNumber-1)*(btnHt+spacing);
    labelStr='Stop';
% Setting userdata to -1 (=stop) will stop the demo.
    callbackStr='set(gca,''Userdata'',-1)';

% Generic button information
    btnPos=[xPos yPos-spacing btnWid btnHt];
    stopHndl=uicontrol('Style','pushbutton', ...
                         'Units','normalized', ...
                         'Position',btnPos, ...
                         'Enable','off', ...
                   'String',labelStr, ...
                   'Callback',callbackStr);

%===================================================================
% The INFO button
```

```
    labelStr='Info';
    callbackStr='truss(''info'')';
    infoHndl=uicontrol('Style','push', ...
                          'Units','normalized', ...
                          'Position',[xPos 0.20 btnWid 0.10], ...
                  'String',labelStr, ...
                  'Callback',callbackStr);


%=======================================================================
% The CLOSE button
    labelStr='Close';
    callbackStr='close(gcf)';
    closeHndl=uicontrol('Style','push', ...
                          'Units','normalized', ...
                          'Position',[xPos 0.05 btnWid 0.10], ...
                 'String',labelStr, ...
                          'Callback',callbackStr);

% Uncover the figure
    hndlList=[startHndl popupHndl stopHndl infoHndl closeHndl];
    set(figNumber,'Visible','on','UserData',hndlList);

    watchoff(oldFigNumber);
    figure(figNumber);

elseif strcmp(action,'start'),

      axHndl=gca;
    figNumber=gcf;
    hndlList=get(figNumber,'UserData');
    startHndl=hndlList(1);
    popupHndl=hndlList(2);
    stopHndl=hndlList(3);
    infoHndl=hndlList(4);
    closeHndl=hndlList(5);
    set([startHndl closeHndl infoHndl],'Enable','off');
    set(stopHndl,'Enable','on');

% ====== Start of Demo
    load truss
    n=get(popupHndl,'Value');
    set(axHndl,'Userdata',play);
    numframes=15;
    del= x0(:,n);
    del=reshape([del' zeros(1,4)] ,2,10)';
    [xd,yd]=gplot(a,xy);
    cla;
    axis([-0.5 5.5 -2 2]);
    h=plot(xd,yd,'EraseMode','background');

% Draw green embankment next to the bridge truss.
    patch([-0.5 0 0.5 -0.5],[0 0 -2 -2],[0 0.7 0]);
    patch([5 5.5 5.5 4.5],[0 0 -2 -2],[0 0.7 0]);
    text(2.5,1.7,'Bending Modes of a Truss', ...
    'HorizontalAlignment','center');
    axis off;
    set(gca,'Drawmode','Fast');

    t=linspace(0,2*pi,numframes+1);

    while get(axHndl,'Userdata')==play,

      for count=1:numframes,
          if get(axHndl,'Userdata')~=play, break; end;
            if n~=get(popupHndl,'Value'),
            n=get(popupHndl,'Value');
            del= x0(:,n);
            del=reshape([del' zeros(1,4)] ,2,10)';
        end;

        delnow=del*sin(t(count));
        [xd,yd]=gplot(a,xy+delnow);
        set(h,'xdata',xd,'ydata',yd);
        drawnow;
        end;    % for count=...
    end;    % while get(axHndl,...

% ====== End of Demo
    set([startHndl closeHndl infoHndl],'Enable','on');
   set(stopHndl,'Enable','off');

elseif strcmp(action,'info');
   ttlStr='Bending Modes Info';
   hlpStr= ...
       ['                                     '
        ' This demo animates 12 natural bending modes    '
        ' of a two-dimensional truss. These bending      '
        ' modes are the results of eigenvalue analysis.  '
        ' They have been ordered by natural frequency,   '
        ' with one being the slowest (and easiest to '
        ' excite) mode and 12 being the fastest.     '
        '                                     '
        ' Use the "Mode" popup menu to select among      '
        ' the various modes. The "Start" and "Stop"      '
        ' buttons control the animation.        '
        '                                     '
        ' File name: truss.m                  '];
   helpfun(ttlStr,hlpStr);

end;    % if strcmp(action, ...
```

Appendix [2]

```
% This program is for Civil Engineer's Thesis [Chapter 2.2].
% Matlab file saved as <unforced.m>
% Script to analyze the Spring-Mass-Damper system

y0=0.15; wn=sqrt(2);
% Define the natural frequncy and initial condition
zeta1=3/(2*sqrt(2));
zeta2=1/(2*sqrt(2));
```

```
% Define the value of dimensionless damping ratio
t=[0:0.1:10];

% Compute the Unforced Response to an Initial Condition
t1=acos(zeta1)*ones(1, length(t));
t2=acos(zeta2)*ones(1, length(t));
% Calculate the phase angle
c1=(y0/sqrt(1-zeta1^2));
c2=(y0/sqrt(1-zeta2^2));
y1=c1*exp(-zeta1*wn*t).*sin(wn*sqrt(1-zeta1^2)*t+t1);
% Response function of a overdamped system
y2=c2*exp(-zeta2*wn*t).*sin(wn*sqrt(1-zeta2^2)*t+t2);
% Response function of an undamped system

bu=c2*exp(-zeta2*wn*t);
bl=-bu;
% Define the envelope function of the response curve

plot(t,y1, '-', t, y2, '--', t, bu, ':', t, bl, ':'); grid
xlabel('Time [sec]'),
ylabel('y(t) Displacement [m]')
text(0.25, 0.70, ['Overdamped zeta1= ', num2str(zeta1), ' -Solid '], 'sc')
text(0.25, 0.30, ['Underdamped zeta2= ', num2str(zeta2), ' -Dashed '], 'sc')
```

Appendix [3]

```
% Compute step response for a second-order system
% This Matlab file is saved as <steprep.m>
% Civil Engineer's Thesis Chapter x.x.x

% Set up denominator polynomial
t=[0:0.1:12]; num=[1];
zeta1=0.1; den1=[1 2*zeta1 1];
zeta2=0.2; den2=[1 2*zeta2 1];
zeta3=0.4; den3=[1 2*zeta3 1];
zeta4=0.7; den4=[1 2*zeta4 1];
zeta5=1.0; den5=[1 2*zeta5 1];
zeta6=2.0; den6=[1 2*zeta6 1];

% Then compute step response
[y1,x,t]=step(num, den1, t);
[y2,x,t]=step(num, den2, t);
[y3,x,t]=step(num, den3, t);
[y4,x,t]=step(num, den4, t);
[y5,x,t]=step(num, den5, t);
[y6,x,t]=step(num, den6, t);

% Generate plot and labels
plot(t,y1, '-', t, y2, '-.', t, y3, ':', t,y4, '-.', t, y5, 'x', t, y6, '.')
xlabel(' wn*t');
ylabel('c(t)');
title('zeta=0.1, 0.2, 0.4, 0.7, 1.0, 2.0')
grid;
```

Appendix [4]

```
% Matlab program <convol.m> for Civil Engineer's Thesis [Chapter 2.4]
% Program to demonstrate use of convolution in getting the response of
% a Single Degree of Freedom (SDOF) to arbitrary inputs.
%
% The program prompts the user to input the M, C, and K values
% (with any consistent system of units) and then calculates and displays
% the impulse response function. For example, one might try:
% M=1 kg, C=0.05N/(m/s), K=4 N/m.
%
% It also displays the undamped natural frequency in rad/s and Hz,
% as well as the damping ratio Zt. It also computes the value of 1/(M*Wd)
% the coefficient of the impulse response function h(t=0).
% For a lightly damped system this should be about the peak value of h(t).
%
% The response to any input can be obtained by convolving the
% impulse response h(t), with that of an arbitrary excitation vector.
% To do this, one needs to edit this file convol.m and modify the
% excitation vector For example, let f=sin(Wn*t);

clc
clf
clear all

disp('  ');
disp('============================================================');
disp('  ');
disp('  Computation of the Response of a SDOF System by Convolution');
disp('  ');
disp('============================================================');
disp('  ');
disp('  ');

disp(' Note: To get meaningful results, use a consistent system of units')
disp('  ');
disp('  ');

M=input('    Enter Mass        M:  ');
C=input('     Enter Damping    C:   ');
K=input('    Enter Stiffness    K:   ');

% Compute the natural frequency, Wn or Fn, the damping ratio, Zt, and the
% damping natural frequency, Wd, from the parameters given above.

Wn=sqrt(K/M);
Fn=Wn/(2*pi);
Zt=C/(2*M*Wn);
Amplification=1/(2*Zt);
Wd=Wn*sqrt(1-Zt*Zt);

% Because we want our response to die out completely over our time axis,
% we will make the length of the time axis 4 times the time constant T,
% which is defined as, T=1/(Zt*Wn), This is the length of time required
% for the impulse response function to decrease by the factor 1/e.

T=1/(Zt*Wn);
DT=4*T/1000;

% Now define the time axis:
t=linspace(0, 4*T, 1000);

% And compute the impulse response h(t), which is a damped sinusoid.
h=(1/M)*(1/Wd)*sin(Wd*t).*exp(-Zt*Wn*t);

% Next, we need to specify the excitation vector.
% A delayed unit impulse will be chosen to illustrate the shifting property
% of the delta function. We first need to initialize the excitation vector.

f=zeros(1,1000);

% Set delayed impulse at the 167th time step.
% f(1,167)=1;

% Replace the above line with the following to get
% sin(Wn*t) excitation which starts at t=0
% f=sin(0.4*Wn*t);
% The above line can be activated by removing "%".

if t<0.5*Fn
   f=sin(0.4*Wn*t)
else
   f=0;
end


% The next line shows how to get a step function starting at t=0
% f=ones(1,1000);

% Compute response x(t) as the convolution of the impulse response h(t)
% with the excitation vector f(t). The next line is the Matlab convolution:
x=conv(h,f)*DT;

% Note the multiplication by DT. The Matlab conv function
% evidently assumes the time increment is 1.0 in the numerical integration
% and does not make allowance for it. Therefore, we have to put it in.

% Plot all results

subplot(311), plot(t,h), grid
title(['Impulse Response of a SDOF System']);
% xlabel(' Time [sec]')
ylabel(' h(t)')

subplot(312), plot(t,f), grid
title('Time History of Force Excitation')
% xlabel('Time [sec]')
ylabel('F(t)')

subplot(313), plot(t,x(1:length(t))), grid
```

```
title('SDOF Response to the Above Excitation')
xlabel('Time [sec]')
ylabel('x(t)')
% subplot(224)
% text(.10, 6, ' SDOF System Parameters ')
% text(.10, 5, ' ================= ')
% text(.15, 3, ['Mass             M= ' num2str(M)])
% text(.15, 2, ['Damping         C= ' num2str(C) ])
% text(.15, 1, ['Stiffness         K= 'num2str(K) ])
orient tall

Natfreq_Hz_Fn=Fn
Natural_period_secs=1/Fn
Circ_Nat_freq_Wn=Wn
Damped_Natfreq_Wd=Wd
Damping_ratio_Zt=Zt
Dyn_amplification=Amplification
Timestep_DT=DT
Timesteps_per_period=1/(Fn*DT)

% The timesteps per period should be greater than about 15
% so as to get a sufficiently smooth integration
% Compute the approximate peak value of h(t)
Max_H_at_zero=1/(M*Wd)
```

Appendix [5]

```
% This is the program for the Sigmoidal Activation Function
% File saved as <sigmoidal.m> for Civil Engineer's Thesis [Chapter 2.3].

R=input('Please enter the ratio of bias to input...
         function between 0 and 1:   ' );

i_fun=-50:1:150;
net_pt=(1-R)*i_fun;
bias=R*i_fun;

theta_1=5;      % Low theta value
theta_2=10;     % Medium theta value
theta_3=25;     % Higher theta value
theta_4=50;     % High theta value

beta_1=(net_pt+bias)/theta_1
beta_2=(net_pt+bias)/theta_2
beta_3=(net_pt+bias)/theta_3
beta_4=(net_pt+bias)/theta_4

o_pt_1=1./(1+exp(-1*beta_1))
o_pt_2=1./(1+exp(-1*beta_2))
o_pt_3=1./(1+exp(-1*beta_3))
o_pt_4=1./(1+exp(-1*beta_4))

plot(net_pt, o_pt_1, '--', net_pt, o_pt_2, ':',...
     net_pt, o_pt_3,'-.', net_pt, o_pt_4, '-')
```

```
grid;

xlabel(' Net Input of the Function')
ylabel(' Output Function after Back Propagation')
```

Appendix [6]

```
% This program is to solve the problem of model superpostion
% and Rayleigh quotient.
% The Matlab code is saved as the file <rayleigh.m>
% Determine frequencies and mode shapes first:

M=0.174*eye(5)
I=0.0133*0.001651^3/12;
k=4*12*70e9*I/0.1834^3;
K=k*[1 -1 0 0 0; -1 2 -1 0 0; 0 -1 2 -1 0; 0 0 -1 2 -1; 0 0 0 -1 2]

P1=[5 4 3 2 1]' % Estimate the mode shape for mode 1
W1_est_ral=sqrt((P1'*K*P1)/(P1'*M*P1))
P=zeros(5);
[P2, W2]=eig(K, M);
[W,pos]=sort(sqrt(diag(W2)));

for c=1:5;
   P(:,c)=P2(:,pos(c))/P2(1, pos(c));
end
W
P

% Modal properties
Mu=P'*M*P;
Kappa=P'*K*P;
Modal_Mass=Mu
Modal_Stiffness=Kappa

e=[1 1 1 1 1']';
for i=1:5;
   gamma(i)=P(:,i)'*M*e/Mu(i,i);
end
gamma

% Define time axis and build IRF and loading for mode 1
% IRF: Impulse Response Function
DT=0.01;
t=(0:DT:3);                % Time axis
Ug=5*cos(W(1,1)/2*t);      % Ground excitation
Zt=0.05;                   % Damping ratio
Wn=W(1,1);
Wd=W(1,1)*sqrt(1-Zt^2);    % Damping frequency

h1=(1/Mu(1,1))*(1/Wn)*sin(Wd*t).*exp(-Zt*Wn*t);
q1=-Mu(1,1)*gamma(1)*DT*conv(h1, Ug);
x=P(1,1)*q1;

subplot(311 ), plot(t, h1), grid
```

```
title(['Impulse Response for Mode 1, time scale [sec]']);
% xlabel('Time [sec]')
ylabel('h(t)')

subplot(312 ), plot(t, Ug), grid
title(['Ground Excitation, time scale [sec]']);
% xlabel('Time [sec]')
ylabel('Acceleration (m/s^2)')

subplot(313 ), plot(t, x(1:length(t))), grid
title(['Response at top of building to graound motion, time scale [sec]']);
% xlabel('Time [sec]')
ylabel('x(t)')


% Determine Number of Modes to be included:
sum=zeros(1,5);
disp('---Top Floor Sum---')
sum(1)=gamma(1)*P(1,1);
for i=2:5
   sum(i)=gamma(i)*P(1,i)+sum(i-1);
end;
sum1=sum

disp('---Bottom Floor---')
sum(1)=gamma(1)*P(5,1);
for i=2:5
   sum(i)=gamma(i)*P(5,i)+sum(i-1);
end
sum5=sum

% Using Modal Masses
disp('---Using Masses---')
sum(1)=gamma(1)^2*Mu(1,1)/0.87;
for i=2:5
   sum(i)=gamma(i)^2*Mu(i,i)/0.87+sum(i-1);
end
Mass_sum=sum
```

Appendix [7]

(The following text is the **output** of the Matlab program)

```
>rayleigh

M =

    0.1740         0         0         0         0
         0    0.1740         0         0         0
         0         0    0.1740         0         0
         0         0         0    0.1740         0
         0         0         0         0    0.1740
```

```
K =
  1.0e+003 *

    2.7168   -2.7168         0         0         0
   -2.7168    5.4335   -2.7168         0         0
         0   -2.7168    5.4335   -2.7168         0
         0         0   -2.7168    5.4335   -2.7168
         0         0         0   -2.7168    5.4335

P1 =
     5
     4
     3
     2
     1

W1_est_ral =

   37.6752

W =

   35.5658
  103.8159
  163.6556
  210.2368
  239.7859

P =

    1.0000    1.0000    1.0000    1.0000    1.0000
    0.9190    0.3097   -0.7154   -1.8308   -2.6825
    0.7635   -0.5944   -1.2036    0.5211    3.5133
    0.5462   -1.0882    0.3728    1.3979   -3.2287
    0.2846   -0.8308    1.3097   -1.6825    1.9190

Modal_Mass =

    0.4884    0.0000    0.0000    0.0000    0.0000
    0.0000    0.5783    0.0000    0.0000    0.0000
    0.0000    0.0000    0.8378    0.0000    0.0000
    0.0000    0.0000         0    1.6371    0.0000
    0.0000    0.0000    0.0000    0.0000    6.0285

Modal_Stiffness =
  1.0e+005 *

    0.0062    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0623    0.0000    0.0000    0.0000
    0.0000    0.0000    0.2244    0.0000    0.0000
    0.0000    0.0000         0    0.7236    0.0000
    0.0000    0.0000    0.0000         0    3.4662


gamma =

    1.2517   -0.3621    0.1586   -0.0632    0.0150
```

```
---Top Floor Sum---

sum1 =

   1.2517    0.8896    1.0481    0.9850    1.0000

---Bottom Floor---

sum5 =

   0.3563    0.6572    0.8648    0.9711    1.0000

---Using Masses---

Mass_sum =

   0.8795    0.9667    0.9909    0.9984    1.0000
```

Appendix [8]

```
% Movie of Bending Moment Produced by a Moving Load.
% File is saved as <movie.m>
% This program is for Civil Engineer's Degree Thesis
% Chapter 5, Section 5.1.1 <code for the case study>
% Maria Y.M. Chow, June 30, 1999

F=700;              % Travelling man's weight, Newton
l=4;                % Bridge span length, meter

% Generate a reference frame

a=1;                % Man's distance from left support, meter
x=0:l/8:l;             % Axis along bridge, meter
B=x.*(1-a/l)*F;        % Bending moment, N-m

for k=1:length(x)
   if x(k)>a
      moment=B(k)-F*(x(k)-a);
      b(k)=moment;
   end
end

plot(x,B)              % This is the reference frame plot of a=1
grid;
xlabel('Diatance from the left end of the bridge span')
ylabel('Bending Moment')
text(0.3, 0.8, 'Maximum bending moment', 'sc')

Mmax=F*l/4          % Maximum-maximorum moment, N-m
% Maximum maximorum means the maximum of all maxima
axis([0 l 0 Mmax])
pause
```

```
% Generate the dynamic animation movie

a=0:0.5:1;
M=moviein(length(a));
for m=1:length(a)
   B=x.*(1-a(m)/l)*F; %Bending moment
   for k=1:length(x)
      if x(k)>a(m)
         moment=B(k)-F*(x(k)-a(m));
         B(k)=moment;
      end
   end
   plot(x,B), axis ([0 1 0 Mmax])
   grid;
   xlabel('Diatance from the left end of the bridge span')
   ylabel('Bending Moment')
   text(0.3, 0.8, 'Maximum bending moment', 'sc')

   M(:,m)=getframe;
end
```

Appendix [9]

```
% Nine Bending Moment Diagram Produced by a Moving Load.
% File is saved as <bridge1.m>
% This program is for Civil Engineer's Degree Thesis
% Chapter 5, Section 5.1.1 <code for the case study>
% Maria Y.M. Chow, June 30, 1999

F=700;              % Travelling man's weight, Newton
l=4;                % Bridge span length, meter

Mmax=F*l/4
% Maximum-maximorum moment, N-m
% Maximum maximorum means the maximum of all maxima

% Generate individual plot of the bending moment of different cases

a=0:0.5:1;
x=0:1/16:1;
for m=1:length(a)
   B=x.*(1-a(m)/l)*F; % Bending moment
   for k=1:length(x)
      if x(k)>a(m)
         moment=B(k)-F*(x(k)-a(m));
         B(k)=moment;
      end
   end
   subplot(3,3,m), plot(x,B);
   grid;
   axis([0 1 0 Mmax]) % Unify the y-axis length of each graph
   xlabel('Bridge span(m)');
   ylabel('Moment (Nm)')
end
```

Appendix [10]

```
function M=bending1 (s,d,F1, F2)

% This file is saved as <bending1.m>
% Bending moment caused vehicle by moving on simply-supported beam.
% The input argument and their SI units are:
%     s - beam span  , m
%     d - car axles  , m
%     F1 - forward-axle load, N
%     F2 - rear axle load, N
%     The output is
%     M, m-by-n matrix of bending moments, Nm
%     Internal variables are:
%     x - position of forward axle relative to left-hand support, positive
forward, m
%     y - coordinate where M is calculated, measured from the left-hand
support, m
%     R1 - reaction in left-hand support, N
%     R2 - reaction in right-hand support, N


s=input(' Please enter the beam span in meter  ');
d=input(' Please enter the distance between car axles  ');
F1=input(' Please enter the forward-axle load in Newton  ');
F2=input(' Please enter the rear axle load in Newton  ');

s, d, F1, F2         % Display the input
disp(' Press ENTER to continue')
pause

step=d/10;
m=fix(s/step)         % number of y steps
n=fix((s+d)/step)     % number of x steps
x=[0: n]*step;        % equally spaced point
x=[x (s+d)]        % include last point
y=[0: m]*step;        % equally spaced points
y=[y s]            % include last point of span
M=zeros((m+2), (n+2)); % space for array of moments

for k=1:(n+1)

   % Case 1

   if x(k)<d
      disp('case 1')      % Show where we are
      R1=(1-x(k)/s)*F1;
      for l=1:(m+1)
         M(l,k)=y(l)*R1;
         if y(l)>=x(k)
            M(l,k)=M(l,k)-(y(l)-x(k))*F1;
         end
      end
   % Case 2

   elseif x(k)<s
      disp('case 2')      % Show where we are
      R1=(1-x(k)/s)*F1+(1+(d-x(k))/s)*F2;
      for l=1:(m+1)
         M(l,k)=y(l)*R1;
         if y(l)>=x(k)-d
            M(l,k)=M(l,k)-(y(l)-(x(k)-d))*F2;
         end
         if y(l)>=x(k)
            M(l,k)=M(l,k)-(y(l)-x(k))*F1;
         end
      end

   % Case 3

   else
      disp('case 3')      % Show where we are
      R1=(1+(d-x(k))/s)*F2;
      for l=1:(m+1)
         M(l,k)=y(l)*R1;
         if y(l)>x(k)-d
            Mdiff=(y(l)-x(k)+d)*F2;
            M(l,k)=M(l,k)-Mdiff;
         end
      end
   end
end

Mmax=max(M)     % Display maximum for each vehicle position
Mmaxmax=max(Mmax)% Maximum maximorum moment

% 3-D Matlab version 5.2 plot

surf(x,y,M)
xlabel('Position of forward axle, m')
ylabel('Position of section, m')
zlabel('Bending moment, Nm')
pause

contour3(x,y,M)
xlabel('Position of forward axle, m')
ylabel('Position of section, m')

zlabel('Bending moment, Nm')
```

This page is intended to leave blank.