Chapter 7

An Adaptive Gradient Training Approach for a Neural-Network Controller: A Civil Engineering Application

The following papers are based on the contents of the this Chapter:

- **"An Efficient Training Technique for a Neural-Network Controller for Seismically Excited Structures," (reference [54]) by Liut *et al*.**
- **"An overview of Some Non-Traditional Neural-Network Training Strategies for Seismic Response Suppression of Building Structures," (reference [56]) by Liut *et al.***
- **"A Modified Gradient-Search Training Technique for Neural-Network Structural Control," (reference [59]) by Liut *et al.***
- **"Seismic Response Mitigation for Hysteretic Building Structures Using a Neural-Network Controller," (reference [64]), by Matheu *et al.***

7.1 Introduction

The primary emphasis of this chapter is to show a new training technique capable of generating an effective control law for a neural-network controller. To illustrate this method we use a civil engineering application in which we seek to reduce the vibrations induced in a building by seismic excitations. Many devices and control techniques have been explored for such purpose. In this chapter, we use an active tuned mass damper similar to that we

discussed in Chapters 3 and 6. The TMD is located on the top floor of a building, and is tuned so that it yields a good passive response.

The basis of the present training technique is to determine an efficient control law in terms of the adjustments introduced to the parameters of the controller. These parameters are the weights of the neural-network controller. Nevertheless, the same technique could be applied to other controllers, such as fuzzy logic controllers. Part of our ongoing research is related to this.

Like in the load-matching procedure discussed in Chapters 5 and 6, the training technique we are about to discuss does not need a system-simulator neural network, as typically required in system identification strategies (references, [1, 10, 21, 28, 37]).

The training method adjusts the parameters of the controller (the weights, for a neural-network controller) by means of a gradient-search strategy carried out according to the outcome of a cost function. During the training process, the backpropagation technique is used as an auxiliary computational tool.

To train the neural-network controller, a set of ground-acceleration time histories is used for the training data. These time histories are artificially generated such that their combined average spectrum is comparable to the spectral characteristics of a particular site. We follow the same training-datum procedure described in Chapter 6.

The following two sections describe the structural model, the neural-network controller, and the training scheme used to obtain the neural-network weights. This is followed by the section on numerical results, where several sets of uncontrolled and controlled responses, obtained for ground acceleration inputs other than those used in the training process, are presented and compared to demonstrate the effectiveness of the controller.

7.2 The Structural Model and the Neural-Network Controller

In this chapter, we briefly describe both the linear and hysteretic structural models described in Chapter 3. The equations of the linear model are given by:

$$\mathbf{M}\,\ddot{\mathbf{z}}(t) + \mathbf{C}\,\dot{\mathbf{z}}(t) + \mathbf{K}\,\mathbf{z}(t) = \mathbf{M}\,\mathbf{r}\,\ddot{x}_g(t) + \mathbf{h}\,u(t) \tag{7.1}$$

where **z** is the $N_d \times 1$ vector of relative displacements of each degree of freedom with respect to the base, N_d is the number of degrees of freedom of the building-TMD system $(N_d = number\ of\ floors + 1)$, the acceleration $\ddot{x}_g$ is the seismic excitation, the $N_d \times N_d$ matrices **M**, **C** and **K** are the mass, damping and stiffness matrices of the system, respectively, the $N_d \times 1$ vector **r** is the influence vector of the ground motion, and **h** is the $N_d \times 1$ location vector, which specifies the position of the actuator.

To take into account a hysteretic behavior of the building we use the Bouc-Wen model. The corresponding equations of motion can be summarized by

$$\mathbf{M}_h\,\ddot{\mathbf{d}} + \mathbf{C}_h\,\dot{\mathbf{d}} + \mathbf{f}(\mathbf{d}, \boldsymbol{\zeta}) = -\,\mathbf{r}_h\,\ddot{x}_g + \mathbf{h}_h\,u_c(\mathbf{y}) \tag{7.2}$$

where **d** is the vector of interstory drifts, and $\ddot{x}_g$ is the ground excitation. The modified mass and damping matrices, $\mathbf{M}_h$, $\mathbf{C}_h$, the modified influence and location vectors, $\mathbf{r}_h$, $\mathbf{h}_h$, and the restoring force **f** are described Chapter 3, section 3.1. Also in Chapter 3 we describe the tuned mass damper used for both models.

Regarding the controller, we use the same two-layer, neural network controller discussed in Chapter 4. A schematic of it is shown in Figure 7-1. The output of the second layer is the control action, which in our case is the control force u_c applied to the tuned mass damper.

The input data to the network consists of a set of N^0 structural response variables. We arrange the input data in a vector **p** of dimension N^0. We summarize the equations that represent this neural network as follows:

Layer *I*:

$$v_j^I(t) = \sum_{i=1}^{N^0} w_{ji}^I \; p_i(t) \qquad \text{for} \quad j = 1,\ldots, N^I \tag{7.3}$$

$$y_j^I(t) = A^I \tanh\left(\beta^I \; v_j^I(t)\right) \qquad \text{for} \quad j = 1,\ldots, N^I \tag{7.4}$$

Layer *II*:

$$v^{II}(t) = \sum_{j=1}^{N^I} w_j^{II} \; y_j^I(t) \qquad \text{for} \quad j = 1,\ldots, N^I \tag{7.5}$$

$$u_c(t) = A^{II} \tanh\left(\beta^{II} \; v^{II}(t)\right) \tag{7.6}$$

where N^I is the number of neurons of the first layer, w_{ji}^I are the weights of the first layer, the hyperbolic tangents represent the squashing functions, w_j^{II} is the single set of weight coefficients for to the second layer, and u_c is the output of the neural network, which is the control command. The constants A^I, β^I, A^{II}, and β^{II} are neural-network parameters, which are determined heuristically.

7.3 Weight Adjustment Procedure

To simplify the discussion of the proposed training method, we consider the whole set of weights as an array of $r = 1,\ldots N_w$ weights, w_r, where N_w is the total number of weights, without differentiating which layer they belong to. Thus, if we compare with the notation used in the previous section for our two-layer neural network, each subscript *r* is associated

with a set of two subscripts *i* and *j* and a superscript *I*, for a first-layer weight ($i.e., w_r = w_{ji}^{I}$), or with a subscript *j* and a superscript *II*, for a second-layer weight ($i.e., w_r = w_{j}^{II}$).

The objective of the training process is to find a set of weights that minimizes a cost function *J* related to the response of the structure. The procedure starts by initializing the weights with small random numbers represented by $w_1\big|_0$, where the zero subscript denotes initial values for the weights, w_r. The set of initial weights is updated in subsequent epochs. An epoch is a cycle of analysis during which the system is exposed to a set of training ground motions, similar to what we explained in Chapter 6. At the end of each epoch, we evaluate the performance of the controller in terms of the cost function *J*. Since the training data are the same for every epoch, any variation ΔJ of the cost function *J* is only triggered by the changes introduced to the weight set. An epoch is considered "successful" if by the end of it the value of the cost function *J* is smaller than the cost function of the previous successful epoch. Each successful epoch is identified by a sequence order denoted by *n*; index *n* ranges from 1 to the total number of successful epochs, N_s.

During every successful epoch *n* we perform three tasks. First, we add a weight adjustment $\Delta w_q\big|_n$ to weight w_q at the beginning of the current n^{th} epoch according to:

$$w_q\big|_n = w_q\big|_{n-1} + \Delta w_q\big|_n \tag{7.7}$$

Second, we compute a new weight adjustment, $\Delta w_q\big|_{n+1}$, in terms of the impact that $\Delta w_q\big|_n$ has on the cost function *J* according to:

$$\Delta w_q\big|_{n+1} = \alpha_q \left.\frac{\partial J}{w_q}\right|_n \tag{7.8}$$

where the parameters α_q are constants defined for each weight (in actual practice a constant α^L was used for each layer L). Third, we update the previously computed weight adjustments $\Delta w_r\big|_n$, for $r \neq q$. Since the change in the weight w_q is in general associated with a change in the gradient of J with respect to the weight set, secondary adjustments, $\Delta \hat{w}_r^q$, are required in order to update all the other precomputed weight adjustments. We express this as

$$\Delta w_r\big|_{n+1} = \Delta w_r\big|_n + \Delta \hat{w}_r^q \qquad \text{for} \quad r \neq q \tag{7.9}$$

where $r = 1, \ldots, N_w$. In the following we describe the procedure to evaluate numerically equations (7.7) to (7.9).

After initializing the weight set with small random numbers, the first epoch is run. This initial epoch is referred to as the zero epoch and is identified by a zero subscript. During this cycle, we compute the average of the absolute values of each input variable. We use this information to normalize each input variable. This helps the "tuning" of the neural network and the training scheme. At the end of epoch zero, the first baseline value of the cost function, J_0, is also determined.

Before epoch one starts, an adjustment $\Delta w_1\big|_1$ is introduced to the first weight of the array w_r according to:

$$w_1\big|_1 = w_1\big|_0 + \Delta w_1\big|_1 \tag{7.10}$$

The initial value of the weight adjustment $\Delta w_1\big|_1$ is randomly set with a small value. If by the end of epoch one $J_1 < J_0$, then the epoch was successful and the adjustment performed to w_1 is kept. If this epoch is not successful, the previous adjustment is discarded, and a new epoch is explored for $w_1\big|_1 = w_1\big|_0 - \Delta w_1\big|_1$. If this new epoch is still unsuccessful no correction is applied to weight one $\left(w_1\big|_1 = w_1\big|_0\right)$. The same criterion is used to update each weight, that

is, if by the end of any epoch n, $J_n < J_{n-1}$, then n was a successful epoch, and the corresponding weight adjustment Δw_q performed on the weight w_q is kept (see equation (7.7)). Otherwise, the adjustment Δw_q is discarded, the subscript n is not updated to $n+1$, and a new epoch is explored for $w_q\big|_n = w_q\big|_{n-1} - \Delta w_q\big|_n$. If this new epoch is also unsuccessful no correction is applied to weight q $\left(w_q\big|_n = w_q\big|_{n-1}\right)$, the subscript n is not updated to $n+1$, and some corrections, which we describe later, are introduced to the weight adjustment Δw_q.

If the current epoch is successful, *i.e.*, $J_n < J_{n-1}$, a new adjustment Δw_q is computed at the end of the epoch according to equation (7.8). The partial derivative in this equation is approximated numerically by:

$$\Delta w_q\big|_{n+1} = \alpha_q \frac{J_n - J_{n-1}}{\Delta w_q\big|_n} \tag{7.11}$$

where $J_n - J_{n-1}$ is the difference between the cost function obtained at the end of the current epoch (J_n) and the cost function obtained at the end of the last successful epoch (J_{n-1}).

In equation (7.9) we consider the updates, $\Delta \hat{w}_r^q$, to the weight adjustments, Δw_r, required by the change, Δw_q, introduced in weight q, for $r \neq q$. Although at the beginning of the current epoch only weight w_q is changed, the gradient direction in the weight space of the cost function J at point $\left(w_1, \ldots, w_q \ldots, w_{N_w}\right)$ very likely differs from its direction at the new point $\left(w_1, \ldots, w_q + \Delta w_q, \ldots, w_{N_w}\right)$. The factor $\Delta \hat{w}_r^q$ in equation (7.9) accounts for this change in the direction of the gradient. To estimate $\Delta \hat{w}_r^q$, the following approach, which requires the backpropagation technique, is applied.

We start by considering the first-order terms in a Taylor series expansion of Δw_r in terms of the change in Δw_q, as follows:

$$\Delta w_r\big|_{n+1} = \Delta w_r\big|_n + \left.\frac{\partial(\Delta w_r)}{\partial(\Delta w_q)}\right|_n \delta w_q + \mathrm{O}\left[(\delta w_q)^2\right] \quad \text{for} \quad r \neq q \tag{7.12}$$

where δw_q is approximated by:

$$\delta w_q = \Delta w_q\big|_{n+1} - \Delta w_q\big|_n \tag{7.13}$$

In the present algorithm, the second term of the right-hand side of equation (7.12) represents the correction $\Delta \hat{w}_r^q$ of equation (7.9) performed at the end of each successful epoch; that is,

$$\Delta \hat{w}_r^q = \left.\frac{\partial(\Delta w_r)}{\partial(\Delta w_q)}\right|_n \delta w_q \tag{7.14}$$

The factor $\partial(\Delta w_r)/\partial(\Delta w_q)$ represents the sensitivity of the r^{th} component of the gradient with respect to the adjustment introduced to the q^{th} weight. An approximate procedure to estimate this factor based on the backpropagation technique is described next.

The backpropagation technique is a well-known procedure that gradually adjusts the parameters of a system in order to emulate a desired or target system (see Rumelhart [81, 82]). If we adopt the current neural network as the desired target system, we can use the current control outputs at every time step, *k*, as the reference for the corresponding (stored) control outputs from the last successful epoch $n-1$. In this way we can define an error between the current actuations and the actuations from the previous successful epoch as:

$$\varepsilon(k) = u_n(k) - u_{n-1}(k) \tag{7.15}$$

We can associate a performance index Θ with the error ε as follows:

$$\Theta(k) = \frac{1}{2}\varepsilon^2(k) \tag{7.16}$$

We do not use the instantaneous index Θ to update the weights. Instead, we perform the sum of all the corresponding terms during the current epoch according to

$$\frac{\partial \overline{\Theta}}{\partial w_r} = \frac{1}{K}\sum_{k=1}^{K}\frac{\partial \Theta}{\partial w_r}(k) \tag{7.17}$$

where K is the total number of time steps in one epoch, and $\frac{\partial \overline{\Theta}}{\partial w_r}$ is an average of the gradient $\frac{\partial \Theta}{\partial w_r}(k)$ for the current epoch. The components of $\frac{\partial \overline{\Theta}}{\partial w_r}$ are computed by the backpropagation algorithm, as explained in Chapter 4. If we use layer notation, they can be expressed as:

Layer *II*:

$$\frac{\partial \overline{\Theta}}{\partial w_j^{II}} = \sum_{k=1}^{K}\lambda^{II}\ \vartheta^{II}(k)\ y_j^{I}(k) \tag{7.18}$$

$$\vartheta^{II}(k) = -\varepsilon(k)\frac{\partial y^{II}}{\partial v^{II}}(k) \tag{7.19}$$

Layer *I*:

$$\frac{\partial \overline{\Theta}}{\partial w_{ji}^{I}} = \sum_{k=1}^{K} \lambda^{I} \; \vartheta_{j}^{I}(k) \; y_{i}^{0}(k) \tag{7.20}$$

$$\vartheta_{j}^{I}(k) = \vartheta^{II}(k) \; w_{j}^{II}(k) \; \frac{\partial y_{j}^{I}}{\partial v_{j}^{I}}(k) \tag{7.21}$$

The learning parameters λ^{I} and λ^{II} are constants determined heuristically.

Equation (7.12) requires the computation of $\partial(\Delta w_r)/\partial(\Delta w_q)$. This partial derivative can be approximated by:

$$\left.\frac{\partial(\Delta w_r)}{\partial(\Delta w_q)}\right|_{n} \cong \frac{\Delta w_r\big|_{n+1} - \Delta w_r\big|_{n}}{\Delta w_q\big|_{n+1} - \Delta w_q\big|_{n}} \tag{7.22}$$

which can be estimated by

$$\left.\frac{\partial(\Delta w_r)}{\partial(\Delta w_q)}\right|_{n} \cong \xi_{rq} \frac{\partial \Delta J / \partial \Delta w_r}{\partial \Delta J / \partial \Delta w_q} \tag{7.23}$$

where ξ_{rq} are constants, and

$$\Delta J \cong J_n - J_{n-1} \tag{7.24}$$

From equations (7.17)-(7.21) we know how to compute $\partial \overline{\Theta} / w_r$, for $r = 1, \ldots, N_w$. Though ΔJ and $\overline{\Theta}$ are different functions, both measure changes in the neural-network controller between two successive successful epochs. Therefore, we can write:

$$\frac{\partial(\Delta J)}{\partial w_r} = \frac{\partial(\Delta J)}{\partial \overline{\Theta}} \frac{\partial \overline{\Theta}}{\partial w_r} \tag{7.25}$$

Substituting this equation into equation (7.23) and that result into equation (7.14), leads to the following form for $\Delta \hat{w}_r^q$:

$$\Delta \hat{w}_r^q \cong \xi_{rq} \frac{\partial \overline{\Theta} / \partial \Delta w_r}{\partial \overline{\Theta} / \partial \Delta w_q} \delta w_q \tag{7.26}$$

where the factor δw_q is given by equation (7.13). The constants ξ_{rq} are determined heuristically. In actual practice, one global constant was used and the learning parameters (λ^I and λ^{II}) were tuned appropriately. A better performance is obtained if very large values of $\Delta \hat{w}_r^q$ compared with $\Delta w_q\big|_{n+1}$ are filtered out.

We have also obtained very good results by simplifying equation (7.26) to

$$\Delta \hat{w}_r^q = \kappa^L \frac{\partial \overline{\Theta}}{\partial w_r} \tag{7.27}$$

where κ^L are constants defined for each layer that can be incorporated into the learning parameters.

In the last section we present a step-by-step example of the whole procedure for a simple three-weight neural network, when successful epochs are encountered.

If by the end of the current epoch, $J_n > J_{n-1}$, we ignore the updates computed for the weight adjustments by equations (7.8) and (7.9). But in this case, we compute a particular update for Δw_q. We adopted two criteria in relation to this: first, if by the end of the current epoch $J_n > J_{n-1}$, then the next epoch is explored for the same weight adjustment but with

opposite sign, $\Delta w_q\big|_n^{(new)} = -\Delta w_q\big|_n$. If by the end of this extra epoch $J_n^{(new)} > J_{n-1}$, then no change is performed to any of the weight adjustments and weight updates, except for Δw_q, whose size is reduced according to:

$$\Delta w_q\big|_n^{(new')} = \gamma^L \frac{J_{n-1}}{J_n} \Delta w_q\big|_n \tag{7.28}$$

where $0 < \gamma^L < 1$. That is, the current weight adjustment Δw_q is reduced proportionally to the increase in the cost function. This procedure guarantees that the algorithm always converges. Matter of fact, an indication that the system starts to converge occurs when predominantly unsuccessful epochs are obtained.

In the procedure described so far, one weight at a time is adjusted at the beginning of each epoch. We can further extend the same procedure to changes in sets of weights $\mathbf{s}_i$ performed at the beginning of each epoch, such that $\mathbf{s}_i \cap \mathbf{s}_j = \mathbf{0}$, for $i \neq j$. For example, all the weights associated with a neuron could be considered as such a weight set. Using this approach, equation (7.7) becomes

$$\mathbf{s}_q\big|_n = \mathbf{s}_q\big|_{n-1} + \Delta\mathbf{s}_q\big|_n \tag{7.29}$$

where for our example, $\mathbf{s}_q$ would be the current set of weights,

$$\mathbf{s}_q = \left\{ w_1^q, w_2^q, \ldots, w_{N_q}^q \right\} \tag{7.30}$$

The subscript N_q stands for the number of weights of weight-set q.

The generalization of equation (7.7) to compute the weight-set adjustments is given by

$$\Delta \mathbf{s}_q \big|_{n+1} = \alpha_q \, \nabla_{\mathbf{s}_q} (J) \big|_n \tag{7.31}$$

where the parameters α_q depend on each weight set q. The generalization of equation (7.8) to update the weight-set adjustments induced by the changes introduced to the weight set q is given by

$$\Delta \mathbf{s}_r \big|_{n+1} = \Delta \mathbf{s}_r \big|_n + \mathbf{\Phi}^{rq} \big|_n \cdot \delta \mathbf{s}_q \quad \text{for} \quad r \neq q \tag{7.32}$$

where $\mathbf{s}_r = \left\{ w_1^r, w_2^r, \ldots, w_{N_r}^r \right\}^T$, N_r stands for the number of weights of weight-set r, $\delta \mathbf{s}_q = \Delta \mathbf{s}_q \big|_{n+1} - \Delta \mathbf{s}_q \big|_n$, and the matrix $\mathbf{\Phi}^{rq}$ is given by

$$\Phi_{ij}^{rq} = \xi_{rq} \frac{\partial \overline{\Theta} / \partial w_i^r}{\partial \overline{\Theta} / \partial w_j^q} \quad \text{for} \quad \begin{matrix} i = 1, \ldots, N_r \\ j = 1, \ldots, N_q \end{matrix} \tag{7.33}$$

The objective of the algorithm we have described in this section is to minimize the cost function J. One of the advantages of this method is the flexibility to choose from a virtually unlimited variety of cost functions. These cost functions can be defined according to what we want to minimize and/or some restrictions we want to impose to the system, such as limited actuation power. This is explained in more detail in the next section.

7.4 Numerical Results

In this section, we discuss the performances of the neural-network controllers used to reduce the seismic response of the following two multi-story building models: 1) a 10-story linear model; 2) a 6-story hysteric model. A schematic of each model is given in Figure 7-1. The controllers are trained with the technique we have discussed in the previous section.

We start our discussion by describing the results obtained for the 10-story linear model. Each story has the same mass and stiffness parameters. Proportional damping is assumed for the system, with a 3% damping ratio assigned for all modes. The control actions are generated by means of a tuned mass damper (TMD) on top of the last story, as shown in Figure 7-1.*a*. The control force acts between the damper mass and the reaction wall (or support). The frequency of the TMD is 91% of the first modal frequency of the building. The mass of the damper is about 40% of the mass of one floor (about 4% of the floor of the entire building). The corresponding damping ratio is 11% of the critical value.

To train the neural-network controller (associated with the linear building) we use a cost function of the form:

$$J = \sum_{k=1}^{K} \left[\mathbf{z}'(k)^T \mathbf{Q}_1 \mathbf{z}'(k) + \ddot{\mathbf{z}}'_a(k)^T \mathbf{Q}_2 \ddot{\mathbf{z}}'_a(k) \right] \tag{7.34}$$

where the vector $\mathbf{z}'$ of dimension N_f is the same displacement vector $\mathbf{z}$ introduced in Chapter 3 without considering the tuned-mass damper, and N_f is the number of floors. The matrices $\mathbf{Q}_1$ and $\mathbf{Q}_2$ are weighting matrices of the form:

$$\mathbf{Q}_1 = a\,\mathbf{I} \quad \text{and} \quad \mathbf{Q}_2 = b\,\mathbf{I} \tag{7.35}$$

where *a* and *b* are positive design parameters, and **I** denotes the $N_f \times N_f$ identity matrix. If a different influence were to be assigned to each individual displacement or acceleration, more general weighting matrices could be used (*i.e.*, a diagonal matrix with different positive elements along its main diagonal). For the numerical results discussed in the following paragraphs, $a = 1$ and $b = 0.000606$.

We implemented a two-layer neural network controller with four neurons in the first layer $\left(N^I = 4\right)$ and a single neuron in the second layer (see Figure 7-2). It is assumed that the velocities and/or the absolute accelerations of each floor can be obtained from direct measurements. Based on this, we study two different control strategies: 1) an acceleration-feedback implementation, using the absolute accelerations of each floor; 2) a velocity-feedback implementation, using the velocities of each floor relative to the base. In both implementations, it is assumed that there is complete information regarding the relative motion of the TMD moving mass. This allows us to include the displacement and velocity of the TMD relative to the reaction wall, as well as its absolute acceleration, in the input vector to the controller, **p**. Therefore, the number of elements of the input vector N^0 is 13 for each case.

To train the controller, we use an ensemble of six artificially generated acceleration time histories. The set of training earthquakes is chosen to exhibit an average spectrum which is representative of the building site characteristics. An epoch consists of a sequence of training earthquakes. The sequence is randomly altered at the beginning of each epoch. Each sequence of training earthquakes defines an epoch. The earthquake excitations used for calculating the present numerical results do not belong to the time histories used for the training set.

The seismic inputs used to evaluate the controller performance and validate the training procedure are the ground-acceleration time histories corresponding to the El Centro (1941) and San Fernando (1971) earthquakes, both normalized to a maximum peak acceleration of 0.3*g*, where *g* is the acceleration of gravity. They are shown in Chapter 3 (Figures 3-2 and 3-

3). In Figure 7-3 (also discussed in 6-1) we compare the spectra of these two time histories with the average spectrum of the six earthquake histories used in the training set.

We first discuss the results for the acceleration-feedback controller. Figure 7-4 shows the top-floor displacement for El Centro and San Fernando motions, for the controlled and uncontrolled cases. The ratios of the controlled to uncontrolled peaks are 0.47 and 0.59, respectively, for the two earthquakes. The overall reduction is very good. In Figure 7-5 we show the uncontrolled and controlled base-shear responses. The base-shear values are normalized with the total weight of the building. Here the ratios of the controlled to uncontrolled base-shear responses are 0.43 and 0.54, respectively. Again, the overall reduction is very good. Figure 7-6 shows the control-force time histories used by El Centro and San Fernando motions, respectively. The forces are normalized with respect to the weight floor. The maximum values of the control forces for the two earthquakes are 15% and 18% of the floor weight, respectively. The corresponding peaks of the TMD strokes are $0.40m$ and $0.23m$. In general, all response quantities have been significantly reduced, and the required magnitude of the control force and TMD strokes are within practical limits.

Next, we discuss the results for the velocity-feedback controller. The next three figures show the performance of this controller. We again use the two validating time histories already introduced. Figure 7-7 shows the controlled and uncontrolled top-floor displacements as functions of time. The ratios of the controlled to the uncontrolled peak responses are 0.42 and 0.59 for the El Centro and San Fernando earthquake motions, respectively. The uncontrolled and controlled base-shear responses are shown in Figure 7-8. Here the corresponding ratios for the same two earthquakes are 0.39 and 0.55, respectively. Figure 7-9 shows the control-force time histories corresponding to the El Centro and San Fernando motions, respectively. For these two earthquakes, the peaks force values are 23% and 22% of the floor weight, respectively. Their corresponding TMD strokes values are $0.35m$ and $0.22m$.

Figure 7-10 shows the top-floor acceleration response spectra of the two feedback strategies and the passive controller, for both the El Centro and San Fernando ground

motions. For both earthquakes, the two active feedback-controllers produce significant reductions of the maximum floor accelerations in the high frequency range. The velocity-feedback controller achieves significantly more reduction than the acceleration-feedback controller for most of the frequency range considered.

Velocity- and acceleration-feedback controllers are also compared in Figure 7-11 for the El Centro earthquake. The response ratios (ratio of the peak controlled to uncontrolled responses) for the peak relative displacements, peak interstory drifts and absolute accelerations of each floor are displayed. For all floors, the response ratios for the acceleration-feedback controller are higher than those for the velocity-feedback controller. This trend is more pronounced for the peak-acceleration results (part *c*), where we observe performance differences of up to 45% in favor of the velocity-feedback controller. We can also observe more interstory drift in the upper floors for the acceleration-feedback case. Since the building used in these examples has the same stiffness and geometrical distribution along its structure, it can be inferred that the relative increase in the interstory drift of the upper floors is directly related to the influence of the TMD. In columns (2), (5), and (8), Table 7.1 shows the response ratios for the passive case. These results are obtained when there is no active component in the TMD $(u = 0)$.

Figure 7-12 shows the results of two different cost functions. The first cost function is the same as in equation (7.34), for $a \neq 0$ and $b \neq 0$ (see equation (7.35)). The second cost function ignores the acceleration dependent term $(\mathbf{Q}_2 = \mathbf{0})$. In order to compare both cases, we used two sets of weights such that the final control laws required nearly the same predetermined peak values of mechanical actuation power. The results shown in Figure 7-12 are for the peak relative displacements and absolute accelerations of each floor for the San Fernando motion. In both instances, we observe a degradation in the control performance when the cost function does not take into account the influence of the absolute accelerations $(\mathbf{Q}_2 = \mathbf{0})$. This is due to the fact that the resulting cost function does not impose a penalty on the control forces. As expected, this effect is especially significant for acceleration response.

In Figure 7-13.*a* we show the base-shear response for a velocity-feedback controller for various levels of peak ground acceleration, for both the El Centro and San Fernando earthquakes. We observe some degradation in the control reduction as the level of excitation increases. In Figure 7-13.*b* the non-linear nature of the controller is made evident. The peak control forces for the same velocity-feedback controller are displayed when the structure is exposed to both El Centro and San Fernando motions. As higher levels of excitations are applied, the force generated by the controller does not increase linearly with the peak ground acceleration; rather it approaches a saturation limit.

We also evaluated the neural-network controller and the training method with the hysteretic building introduced in Chapter 3. In this case, we use a 6-story shear building model with uniform mass, stiffness and strength properties. The first natural period is about 0.61*sec*. As before, 3% modal-damping ratio is applied to all modes. On top of the building an active TMD is added, with a damper mass about 3% of the total building mass. A schematics of this model can be seen in Figure 7-1.*b*.

The input to the controller consists of set of all six interstory velocities, in addition to the actuator stroke and corresponding actuation velocity. Thus, the total number of input quantities is $N^0 = 8$. We used the same 5-neuron, two-layer architecture for the neural-network controller we discussed for the linear building, and the same training set of six artificially generated earthquakes with spectra compatible with the characteristic spectrum of the building site. To the El Centro and San Fernando validating records, we added the Loma Prieta (1989) ground motion. In Figure 7-14 part *a*, we show its time history, and in part *b* a plot of both Loma Prieta and the average training set spectra. As before, all earthquakes are normalized to a maximum peak acceleration of 0.3*g*.

The training cost function we used in this case was:

$$J = \sum_{k=1}^{K} \left(\mathbf{d}^T(k)\, \mathbf{Q}\, \mathbf{d}(k) + \eta\, \nu^2(k) \right) \tag{7.36}$$

where $\mathbf{d}(k)$ is the instantaneous interstory deformation vector, and $\nu(k)$ is the instantaneous velocity of the TMD mass. Like before

$$\mathbf{Q} = c\,\mathbf{I} \tag{7.37}$$

The constants c and η are weighting parameters similar to a and b, for the cost function previously discussed.

Figure 7-15 shows the response ratios (ratio of the peak controlled to uncontrolled responses) for the peak relative displacement and absolute acceleration of each story when the building is excited with the Loma Prieta earthquake. We show the same in Figure 7-16 for the El Centro ground motion excitation. In both cases, a good reduction is detected.

Table 7-2 shows the response reduction of the relative displacements and absolute accelerations for all story levels for the three validating ground motions. In addition to this, the response reductions obtained with a passive tuned mass damper are shown. Table 7-2 also shows the control requirements corresponding to each ground motion, in terms of the maximum excursion of the TMD mass and the peak value of the control force, normalized with respect to the weight of each floor. It is noted that the response factor values are reduced by the active controller for all three ground motions. The peak values of the control force in the three inputs are comparable to those obtained for the linear model.

If we compare the response reductions of Figures 7-15 and 7-16 with those obtained for the linear building model, it seems that the controller used for the hysteretic model is less efficient. The comparison is crude since we use different number of stories, and we used intesrtory velocity feedback for the hysteretic building controller, whereas for the linear-building controller we used velocity feedback. However from Tables 7-1 and 7-2 it follows that the passive TMD performance is less efficient in the case of the hysteretic building. For example, for El Centro earthquake we obtained an excellent average passive response reduction of about 0.6 for the ten floors of the linear model, whereas the corresponding

average reduction for the hysteretic model was only 0.9. We can say something similar of the acceleration response reduction; an average of 0.8 was obtained for the linear model whereas for the hysteretic model the corresponding average was 0.9. However good active response reductions compared with the corresponding passive response reductions were obtained in all cases, for both the linear and hysteretic models.

7.5 Final Remarks

In this chapter, we studied a neural-network-control application for the reduction of the seismic response of building structures. We discussed a new training strategy based on the optimization of a cost function. This strategy provides lots of flexibility in choosing the parameters to be minimized. To train both velocity- and acceleration-feedback controllers we used an ensemble of artificially generated earthquakes with an average spectrum comparable with the spectra of the actual earthquakes. We successfully applied the same training-datum strategy used for the force-matching training procedure described in Chapter 6. The performances of the controllers were evaluated for a 10-story shear building model with typical mass and frequency characteristics, and a 6-story hysteretic model with analogous characteristics. The controllers proved efficient in both cases. We explored two different feedback implementations for the linear model: velocity and acceleration feedback. Their respective performances were compared, and the velocity-feedback strategy proved more efficient. We used an interstory velocity-feedback controller for the hysteretic model. For both the linear and hysteretic models, the numerical results show the effectiveness of the proposed training approach in providing training for the neural-network controllers, which were capable of achieving significant response reductions with low control requirements. Three different training cost functions were applied. Although their performances were different, the training method proved effective in a finding control law in all cases.

7.6 Complementary Example: Step-By-Step Training Procedure for a Three-Weight Neural Network during the First Six Epochs.

All epochs used in this example are assumed to be successful epochs.

Epoch 0:

1) Weights and weight adjustments are initialized with initial guesses (zero subscript) as:

$$w_r = w_r\big|_0 \quad \text{for} \quad r = 1 \ldots N_w \tag{7.38}$$

$$\Delta w_1 = \Delta w_1\big|_0 \tag{7.39}$$

$$\Delta w_r = 0 \quad \text{for} \quad r > 1 \tag{7.40}$$

where N_w is the total number of weights, and the subscript r accounts for the weight number (all weights are reordered into a vector).

2) The initial cost function J_0 is computed with $w_1\big|_0$, $w_2\big|_0$ and $w_3\big|_0$.

Epoch 1:

1) Weight one is adjusted with its initial guess as follows:

$$w_1\Big|_1 = w_1\Big|_0 + \Delta w_1\Big|_1 \tag{7.41}$$

where $\Delta w_1\Big|_1 = \Delta w_1\Big|_0$.

2) A new cost function J_1 is computed with $w_1\big|_1$, $w_2\big|_1 = w_2\big|_0$ and $w_3\big|_1 = w_3\big|_0$.

3) A new weight adjustment is computed for weight one according to:

$$\Delta w_1\Big|_2 = \alpha_1 \frac{J_1 - J_0}{\Delta w_1\Big|_1} \tag{7.42}$$

where α_1 is a constant.

4) Weight adjustment two is initialized according to:

$$\Delta w_2\Big|_2 = \xi_{21} \frac{\dfrac{\partial \overline{\Theta}}{\partial w_2}\Big|_{BP_1}}{\dfrac{\partial \overline{\Theta}}{\partial w_1}\Big|_{BP_1}} \left(\Delta w_1\Big|_2 - \Delta w_1\Big|_1 \right) \tag{7.43}$$

where ξ_{21} is a constant and BP_1 denotes backpropagation calculations performed during epoch 1. This is done according to

$$\left.\frac{\partial\overline{\Theta}}{\partial w_1}\right|_{BP_1} = \frac{1}{K}\sum_{k=1}^{K}\frac{\partial\Theta}{\partial w_1}(k) \tag{7.44}$$

$$\left.\frac{\partial\overline{\Theta}}{\partial w_2}\right|_{BP_1} = \frac{1}{K}\sum_{k=1}^{K}\frac{\partial\Theta}{\partial w_2}(k) \tag{7.45}$$

$$\Theta(k) = \frac{1}{2}[u_1(k) - u_0(k)]^2 \tag{7.46}$$

where k is each time step within the current epoch, K is the total number of time steps of the current epoch (the same for all epochs), $u_1(k)$ are the control actions for each time step of epoch 1, and $u_0(k)$ are the corresponding control actions from the previous successful epoch (which must be stored). For the present example $u_0(k) = 0$. Weight adjustment $\Delta w_2\Big|_2$ is used in the next epoch.

5) Weight adjustment three is initialized according to:

$$\Delta w_3\Big|_2 = \xi_{31}\frac{\left.\frac{\partial\overline{\Theta}}{\partial w_3}\right|_{BP_1}}{\left.\frac{\partial\overline{\Theta}}{\partial w_1}\right|_{BP_1}}\left(\Delta w_1\Big|_2 - \Delta w_1\Big|_1\right) \tag{7.47}$$

where ξ_{31} is a constant. Weight adjustment $\Delta w_3\Big|_2$ is stored and used later.

Epoch 2:

1) Weight two is adjusted with its initial guess as follows:

$$w_2\Big|_2 = w_2\Big|_1 + \Delta w_2\Big|_2 \tag{7.48}$$

2) A new cost function J_2 is computed with $w_1\Big|_2 = w_1\Big|_1$, $w_2\Big|_2$ and $w_3\big|_2 = w_3\big|_1$.

3) A new weight adjustment is computed for weight two according to:

$$\Delta w_2\Big|_3 = \alpha_2 \frac{J_2 - J_1}{\Delta w_2\Big|_2} \tag{7.49}$$

where α_2 is a constant. Weight adjustment $\Delta w_2\Big|_3$ is stored and used later.

4) Weight adjustment one is updated according to:

$$\Delta w_1\Big|_3 = \Delta w_1\Big|_2 + \xi_{12} \frac{\left.\dfrac{\partial \overline{\Theta}}{\partial w_1}\right|_{BP_2}}{\left.\dfrac{\partial \overline{\Theta}}{\partial w_2}\right|_{BP_2}} \left(\Delta w_2\Big|_3 - \Delta w_2\Big|_2 \right) \tag{7.50}$$

where ξ_{12} is a constant, and $\dfrac{\partial \overline{\Theta}}{\partial w_1}$ and $\dfrac{\partial \overline{\Theta}}{\partial w_2}$ are computed as in equations (7.44) and (7.45) for $\Theta = \frac{1}{2}[u_2(k) - u_1(k)]^2$. Weight adjustment $\Delta w_1\Big|_3$ is stored and used later.

5) Weight adjustment three is updated according to:

$$\Delta w_3\Big|_3 = \Delta w_3\Big|_2 + \xi_{32}\frac{\left.\dfrac{\partial\overline{\Theta}}{\partial w_3}\right|_{BP_2}}{\left.\dfrac{\partial\overline{\Theta}}{\partial w_2}\right|_{BP_2}}\left(\Delta w_2\Big|_3 - \Delta w_2\Big|_2\right) \tag{7.51}$$

where ξ_{32} is a constant. Weight adjustment $\Delta w_3\Big|_3$ is used in the next epoch.

Epoch 3:

1) Weight three is adjusted with its initial guess as follows:

$$w_3\Big|_3 = w_3\Big|_2 + \Delta w_3\Big|_3 \tag{7.52}$$

2) A new cost function J_3 is computed with $w_1\Big|_3 = w_1\Big|_2$, $w_2\Big|_3 = w_2\Big|_2$ and $w_3\Big|_3$.

3) A new weight adjustment is computed for weight three according to:

$$\Delta w_3\Big|_4 = \alpha_3 \frac{J_3 - J_2}{\Delta w_3\Big|_3} \tag{7.53}$$

where α_3 is a constant. Weight adjustment $\Delta w_3\Big|_4$ is stored and used later.

4) Weight adjustment one is updated according to:

$$\Delta w_1\Big|_4 = \Delta w_1\Big|_3 + \xi_{13} \frac{\left.\frac{\partial \overline{\Theta}}{\partial w_1}\right|_{BP_3}}{\left.\frac{\partial \overline{\Theta}}{\partial w_3}\right|_{BP_3}} \left(\Delta w_3\Big|_4 - \Delta w_3\Big|_3 \right) \tag{7.54}$$

where ξ_{13} is a constant, and $\frac{\partial \overline{\Theta}}{\partial w_1}$ and $\frac{\partial \overline{\Theta}}{\partial w_3}$ are computed as in equations (7.44) and (7.45) for $\Theta = \frac{1}{2}[u_3(k) - u_2(k)]^2$. Weight adjustment $\Delta w_1\Big|_4$ is used in the next epoch.

6) Weight adjustment two is updated according to:

$$\Delta w_2\Big|_4 = \Delta w_2\Big|_3 + \xi_{23} \frac{\left.\frac{\partial \overline{\Theta}}{\partial w_2}\right|_{BP_3}}{\left.\frac{\partial \overline{\Theta}}{\partial w_3}\right|_{BP_3}} \left(\Delta w_3\Big|_4 - \Delta w_3\Big|_3 \right) \tag{7.55}$$

where ξ_{23} is a constant. Weight adjustment $\Delta w_2\Big|_4$ is stored and used later.

Epoch 4:

1) Weight one is readjusted according to:

$$w_1\Big|_4 = w_1\Big|_3 + \Delta w_1\Big|_4 \tag{7.56}$$

2) A new cost function J_4 is computed with $w_1\Big|_4$, $w_2\Big|_4 = w_2\Big|_3$ and $w_3\Big|_4 = w_3\Big|_3$.

3) A new weight adjustment is computed for weight one as follows:

$$\Delta w_1\Big|_5 = \alpha_1 \frac{J_4 - J_3}{\Delta w_1\Big|_4} \tag{7.57}$$

where α_1 is a constant.

4) Weight adjustment two is updated according to:

$$\Delta w_2\Big|_5 = \Delta w_2\Big|_4 + \xi_{21} \frac{\dfrac{\partial \overline{\Theta}}{\partial w_2}\Big|_{BP_4}}{\dfrac{\partial \overline{\Theta}}{\partial w_1}\Big|_{BP_4}} \left(\Delta w_1\Big|_5 - \Delta w_1\Big|_4 \right) \tag{7.58}$$

where ξ_{21} is a constant, and $\dfrac{\partial \overline{\Theta}}{\partial w_2}$ and $\dfrac{\partial \overline{\Theta}}{\partial w_1}$ are computed as in equations (7.44) and (7.45) for $\Theta = \dfrac{1}{2}[u_4(k) - u_3(k)]^2$.

5) Weight adjustment three is updated according to:

$$\Delta w_3\Big|_5 = \Delta w_3\Big|_4 + \xi_{31}\frac{\left.\frac{\partial\overline{\Theta}}{\partial w_3}\right|_{BP_4}}{\left.\frac{\partial\overline{\Theta}}{\partial w_1}\right|_{BP_4}}\left(\Delta w_1\Big|_5 - \Delta w_1\Big|_4\right) \tag{7.59}$$

where ξ_{31} is a constant.

Epoch 5:

1) Weight two is readjusted according to:

$$w_2\Big|_5 = w_2\Big|_4 + \Delta w_2\Big|_5 \tag{7.60}$$

2) A new cost function J_5 is computed with $w_1\Big|_5 = w_1\Big|_4$, $w_2\Big|_5$ and $w_3\Big|_5 = w_3\Big|_4$.

3) A new weight adjustment is computed for weight two as follows:

$$\Delta w_2\Big|_6 = \alpha_2 \frac{J_5 - J_4}{\Delta w_2\Big|_5} \tag{7.61}$$

where α_2 is a constant.

4) Weight adjustment one is updated according to:

$$\Delta w_1\Big|_6 = \Delta w_1\Big|_5 + \xi_{12} \frac{\dfrac{\partial \overline{\Theta}}{\partial w_1}\Big|_{BP_5}}{\dfrac{\partial \overline{\Theta}}{\partial w_2}\Big|_{BP_5}} \left(\Delta w_2\Big|_6 - \Delta w_2\Big|_5 \right) \tag{7.62}$$

where ξ_{12} is a constant, and $\dfrac{\partial \overline{\Theta}}{\partial w_1}$ and $\dfrac{\partial \overline{\Theta}}{\partial w_2}$ are computed as in equations (7.44) and (7.45) for $\Theta = \frac{1}{2}[u_5(k) - u_4(k)]^2$.

5) Weight adjustment three is updated according to:

$$\Delta w_3\Big|_6 = \Delta w_3\Big|_5 + \xi_{32}\frac{\left.\frac{\partial\overline{\Theta}}{\partial w_3}\right|_{BP_5}}{\left.\frac{\partial\overline{\Theta}}{\partial w_2}\right|_{BP_5}}\left(\Delta w_2\Big|_6 - \Delta w_2\Big|_5\right) \tag{7.63}$$

where ξ_{32} is a constant.

Epoch 6:

1) Weight three is readjusted according to:

$$w_3\Big|_6 = w_3\Big|_5 + \Delta w_3\Big|_6 \tag{7.64}$$

2) A new cost function J_6 is computed with $w_1\Big|_6 = w_1\Big|_5$, $w_2\Big|_6 = w_2\Big|_5$ and $w_3\Big|_6$.

3) A new weight adjustment is computed for weight three as follows:

$$\Delta w_3\Big|_7 = \alpha_3 \frac{J_6 - J_5}{\Delta w_3\Big|_6} \tag{7.65}$$

where α_3 is a constant.

4) Weight adjustment one is updated according to:

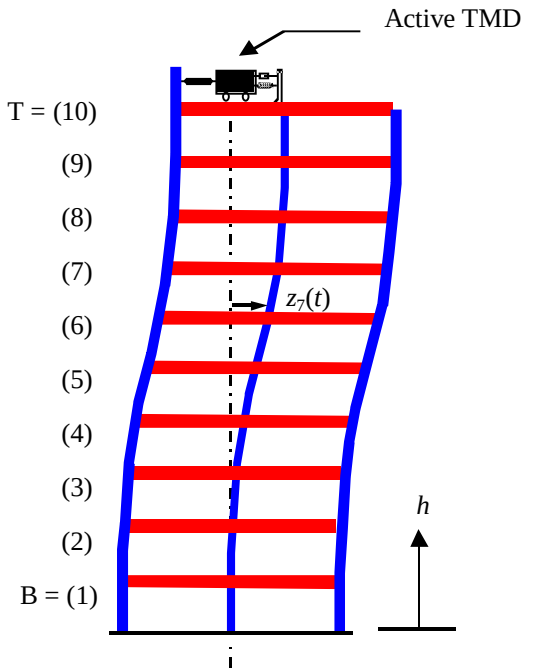
$$\Delta w_1\Big|_7 = \Delta w_1\Big|_6 + \xi_{13} \frac{\left.\frac{\partial \Theta}{\partial w_1}\right|_{BP_6}}{\left.\frac{\partial \Theta}{\partial w_3}\right|_{BP_6}} \left(\Delta w_3\Big|_7 - \Delta w_3\Big|_6 \right) \tag{7.66}$$

where ξ_{13} is a constant, and $\frac{\partial \overline{\Theta}}{\partial w_1}$ and $\frac{\partial \overline{\Theta}}{\partial w_3}$ are computed as in equations (7.44) and (7.45) for $\Theta = \frac{1}{2}[u_6(k) - u_5(k)]^2$.

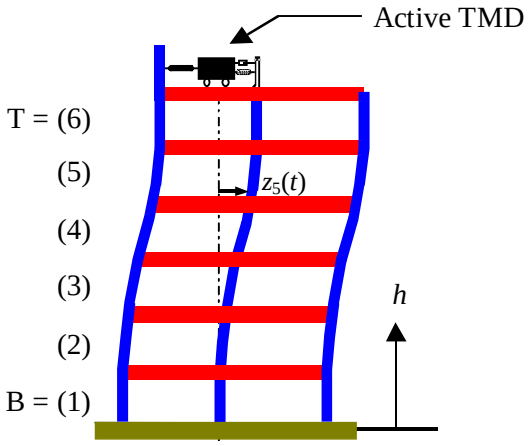
5) Weight adjustment two is updated according to:

$$\Delta w_2\Big|_7 = \Delta w_2\Big|_6 + \xi_{23}\frac{\left.\frac{\partial\overline{\Theta}}{\partial w_2}\right|_{BP_6}}{\left.\frac{\partial\overline{\Theta}}{\partial w_3}\right|_{BP_6}}\left(\Delta w_3\Big|_7 - \Delta w_3\Big|_6\right) \tag{7.67}$$

where ξ_{23} is a constant.



(a)



(b)

Figure 7-1: (a) Ten-story and (b) six-story schematic representations of two buildings equipped with active tuned mass dampers.

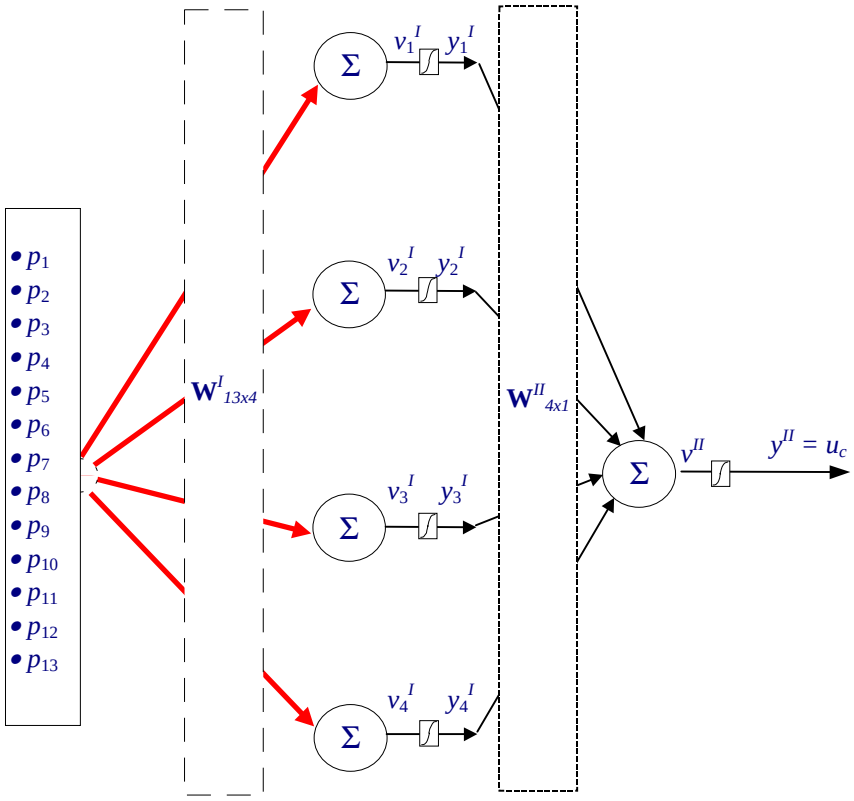


Figure 7-2: Two-layer, five-neuron neural-network controller. Thick arrows denote different sets of weighted input data.

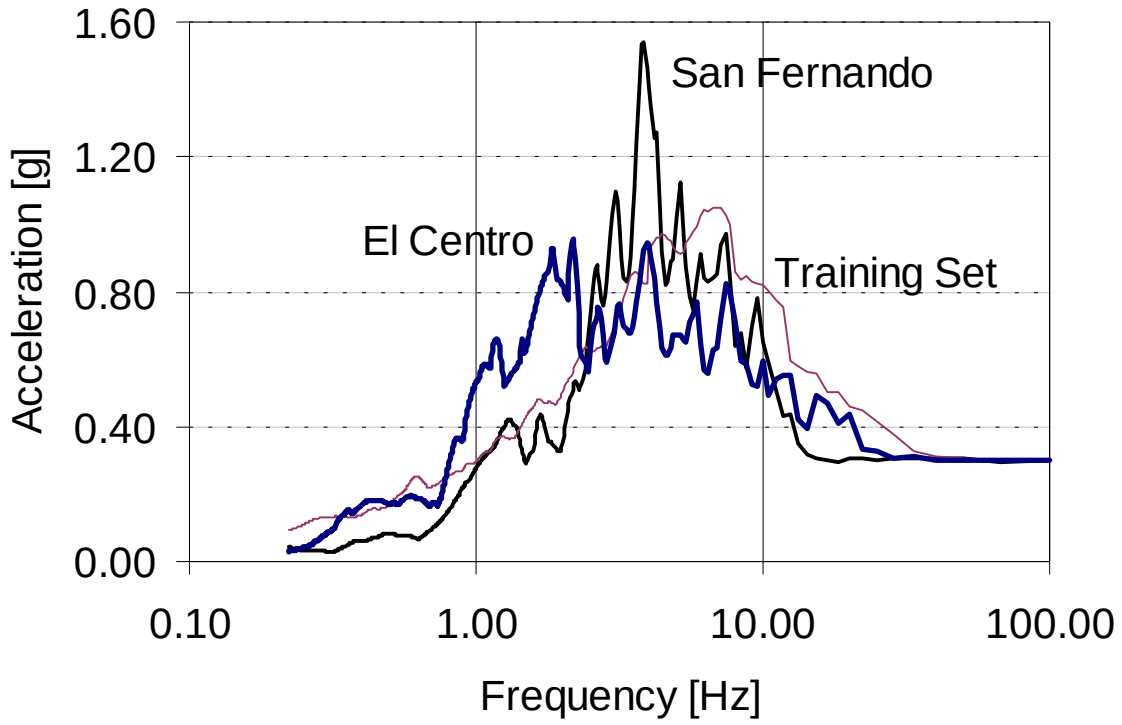
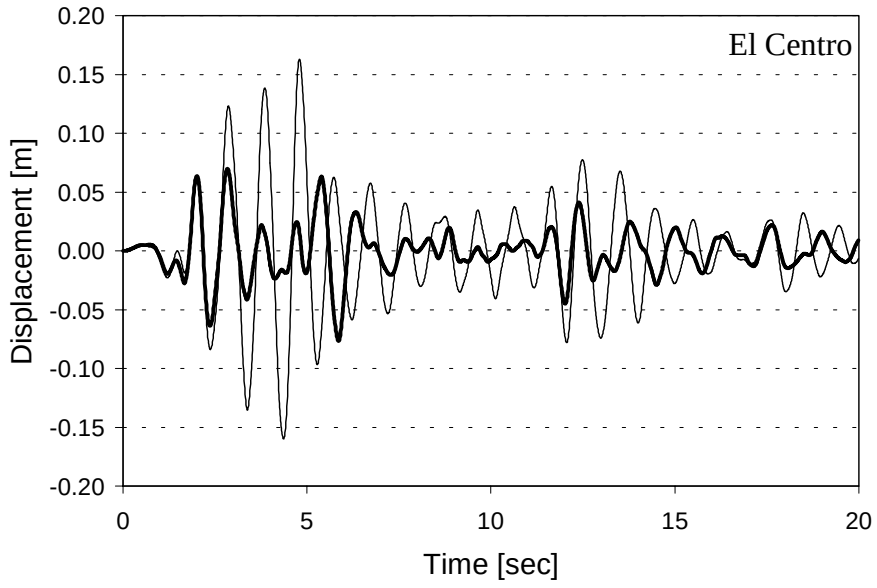
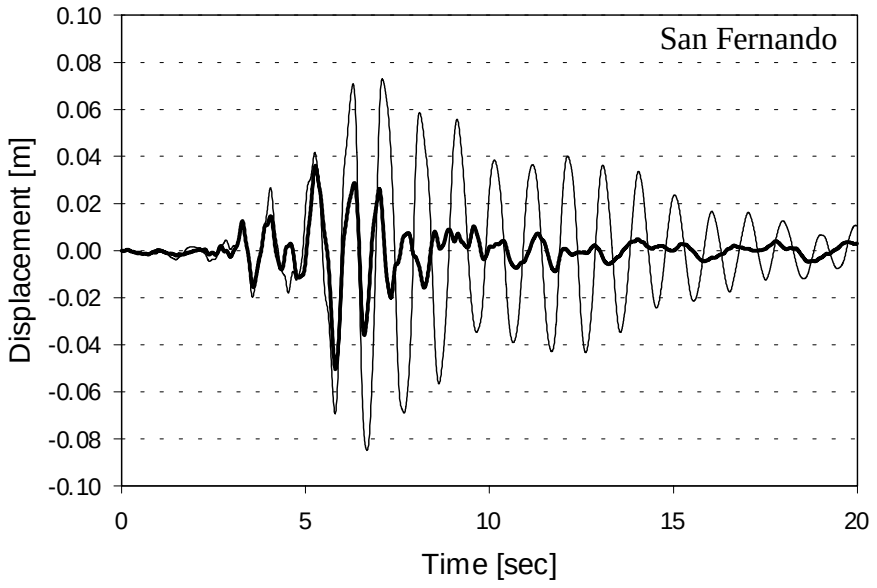


Figure 7-3**:** Ground-response spectra corresponding to the set of ground motions used for training (average) and validating earthquakes.

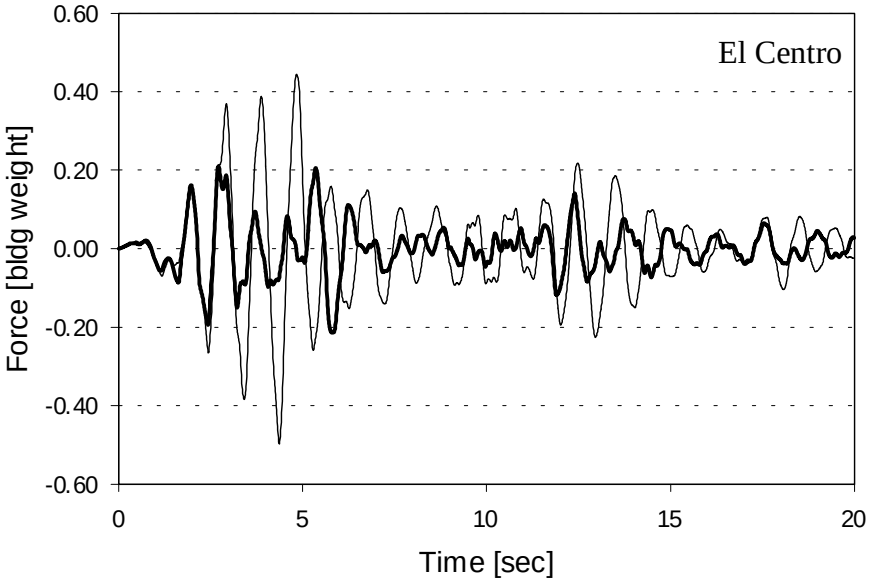


(a)

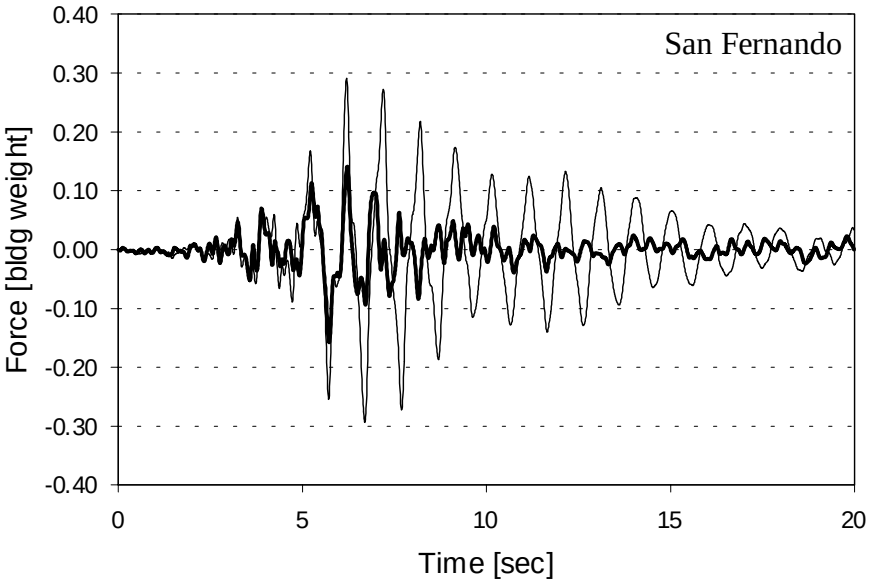


(b)

Figure 7-4: Uncontrolled and controlled top-floor displacements for (a) El Centro and (b) San Fernando ground motions. Acceleration feedback.

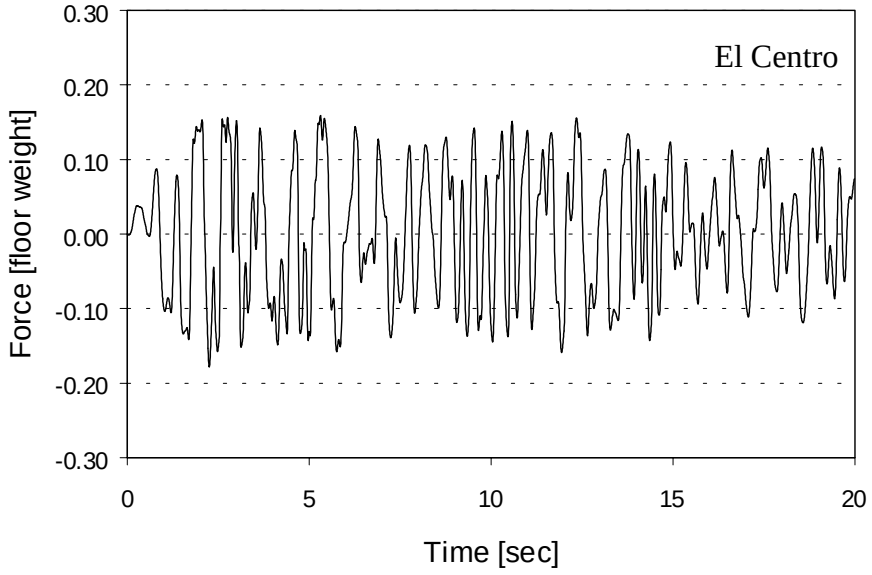


(a)

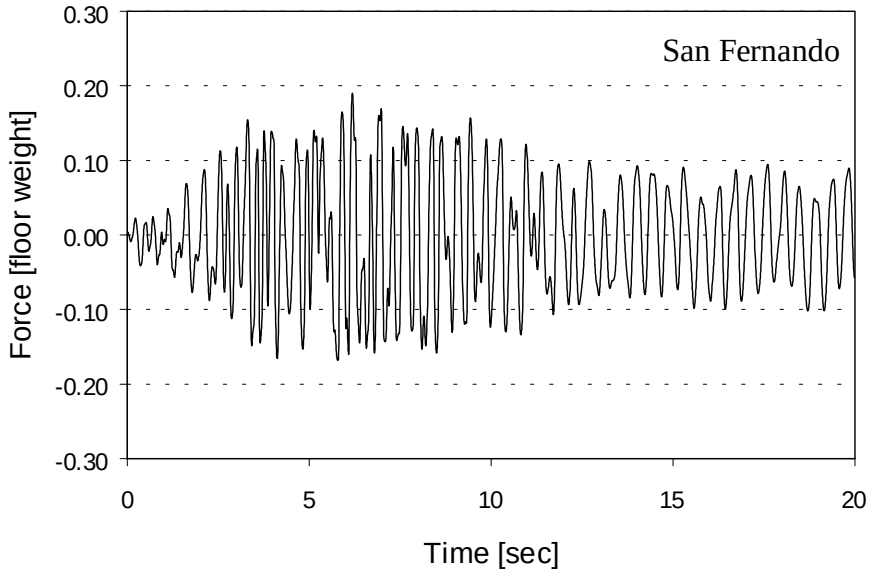


(b)

Figure 7-5: Uncontrolled and controlled base-shear responses for (a) El Centro and (b) San Fernando ground motions. Acceleration feedback.

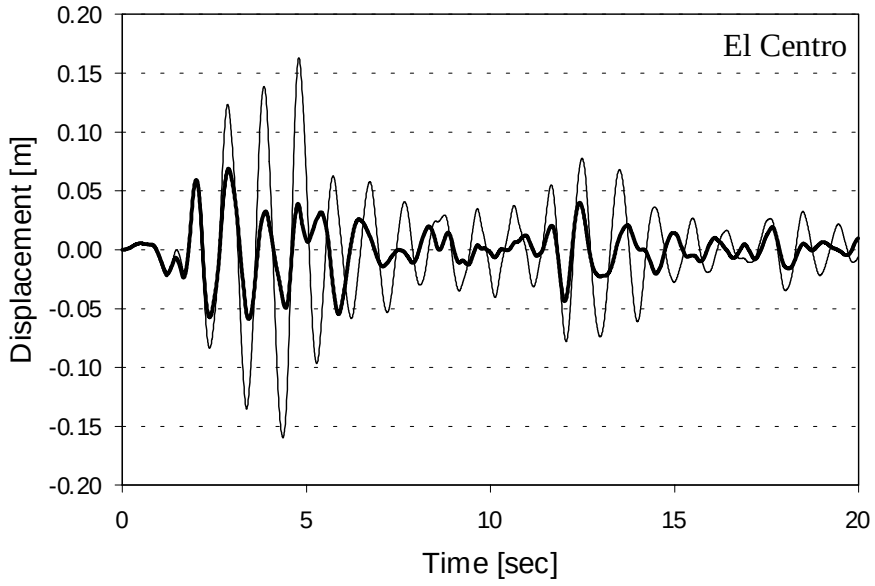


(a)

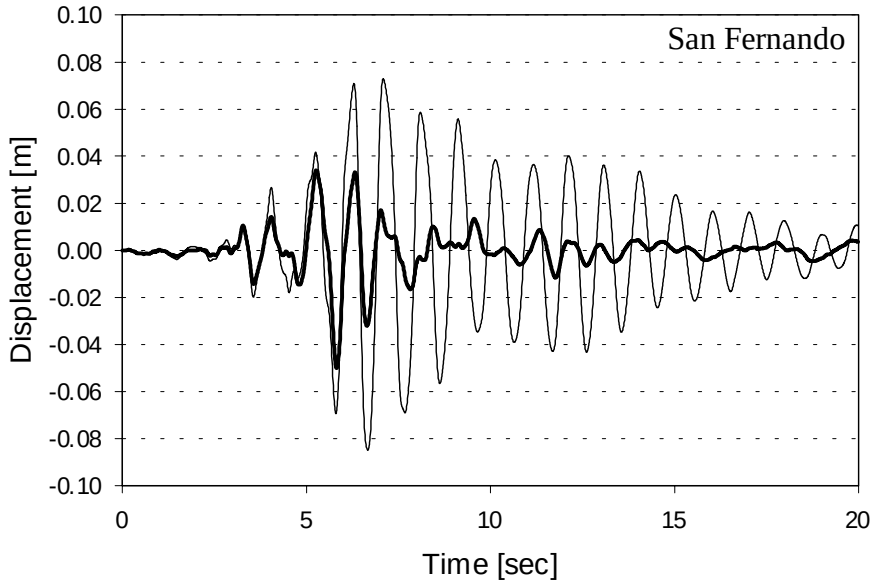


(b)

Figure 7-6: Control-force time histories for (a) El Centro and (b) San Fernando ground motions. Acceleration feedback.

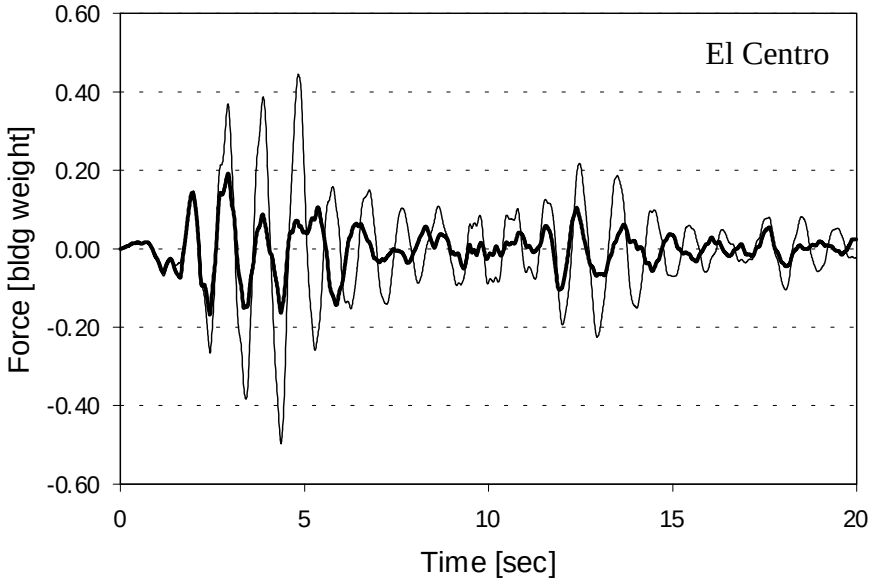


(a)

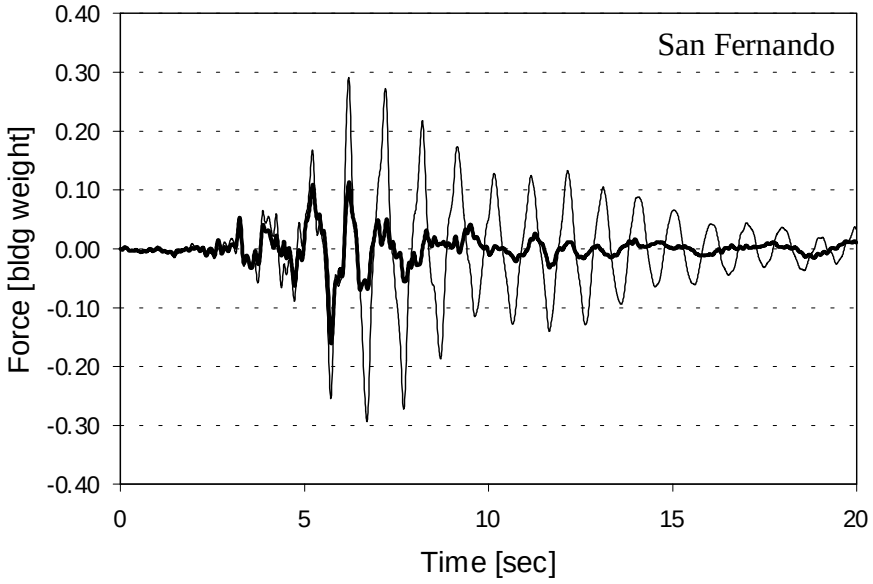


(b)

Figure 7-7: Uncontrolled and controlled top-floor displacements for (a) El Centro and (b) San Fernando ground motions. Velocity feedback.

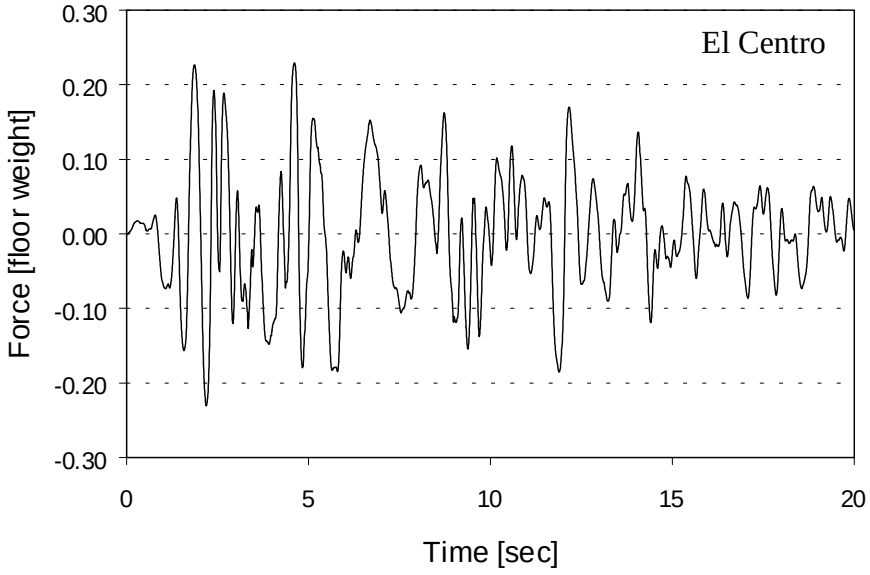


(a)

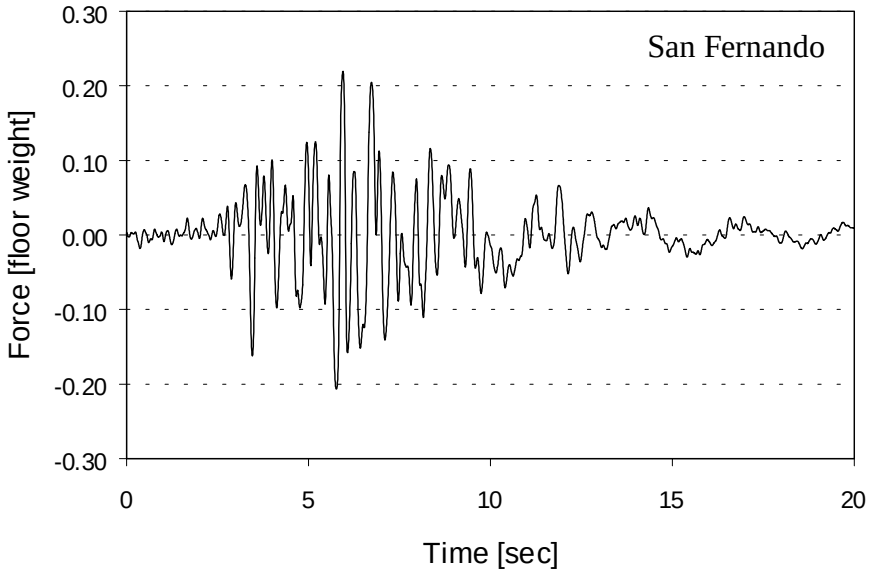


(b)

Figure 7-8: Uncontrolled and controlled base-shear responses for (a) El Centro and (b) San Fernando ground motions. Velocity feedback.

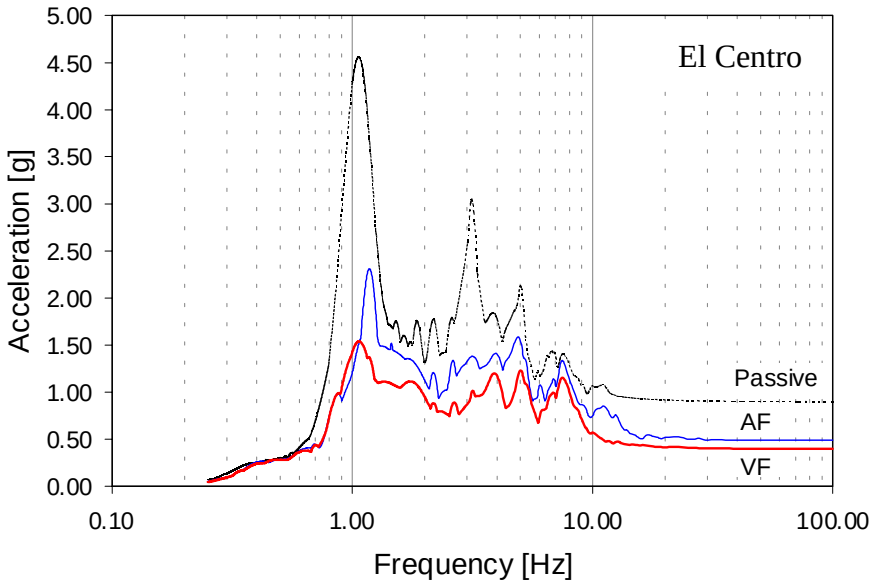


(a)

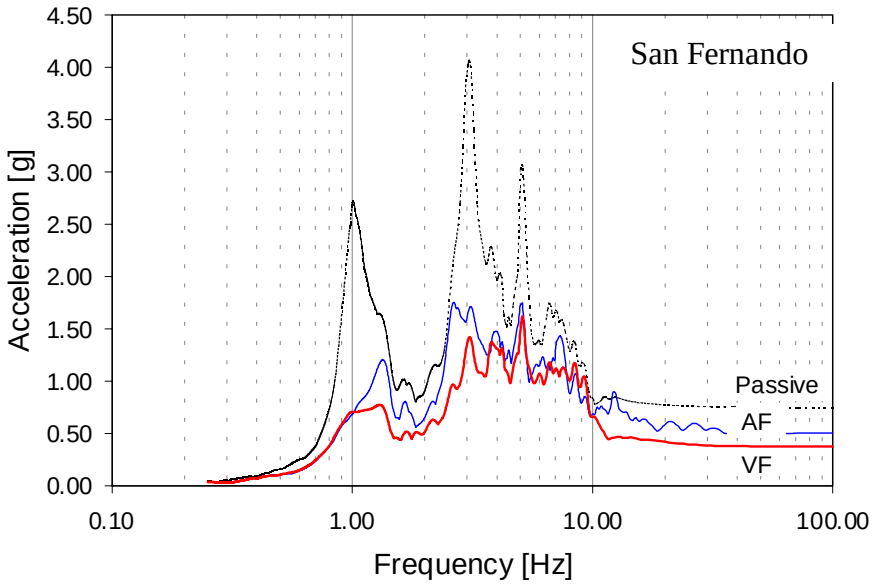


(b)

Figure 7-9: Control-force time histories for (a) El Centro and (b) San Fernando ground motions. Velocity feedback.



(a)



(b)

Figure 7-10**:** Top-floor response spectra for a passive controller (Passive), velocity-feedback controller (VF), and acceleration-feedback controller (AF). El Centro (a) and San Fernando (b) ground motions.

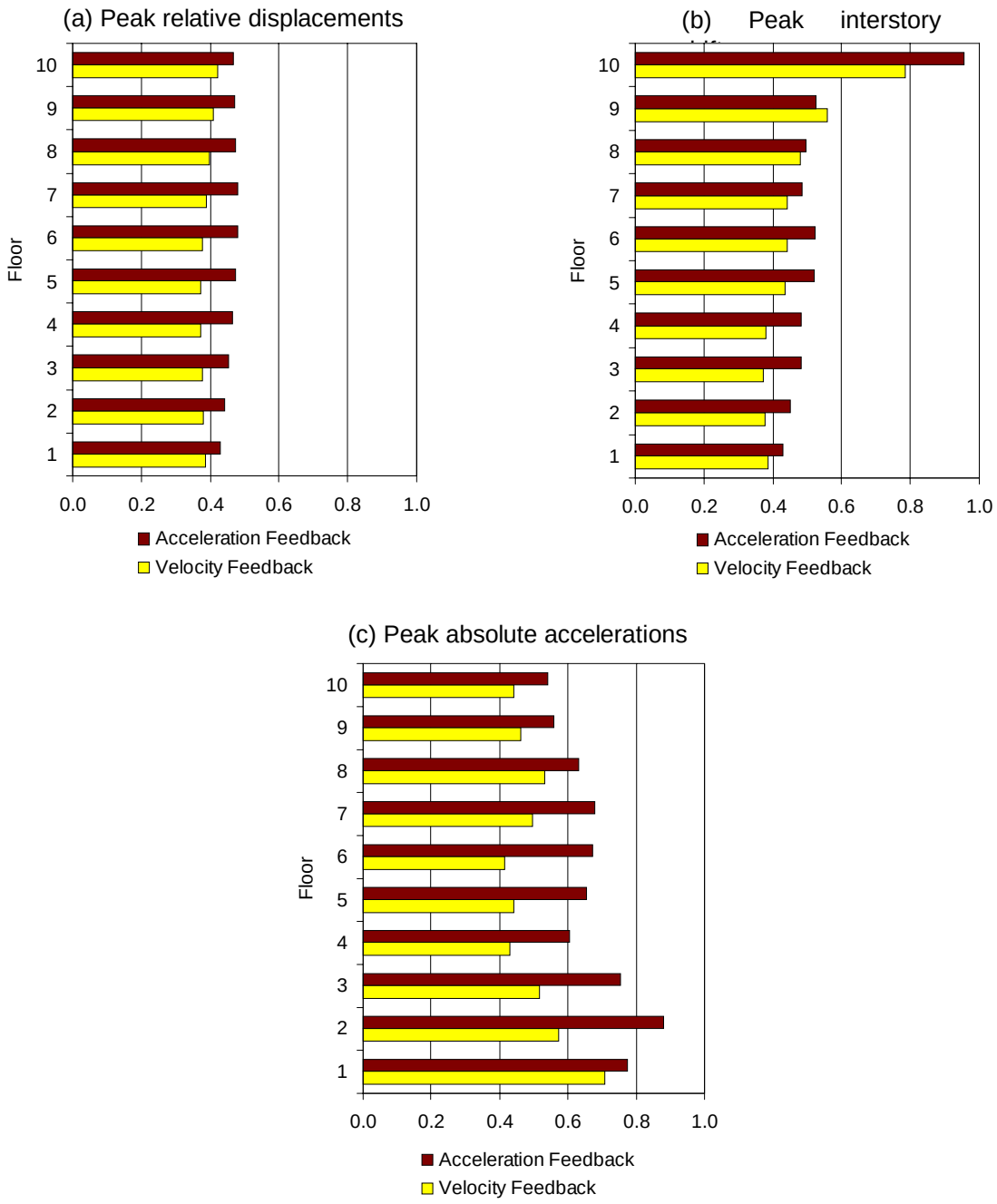


Figure 7-11**:** Response-reduction factors for acceleration and velocity-feedback for (a) peak relative displacements, (b) peak interstory drift, and (c) peak absolute accelerations. El Centro earthquake.

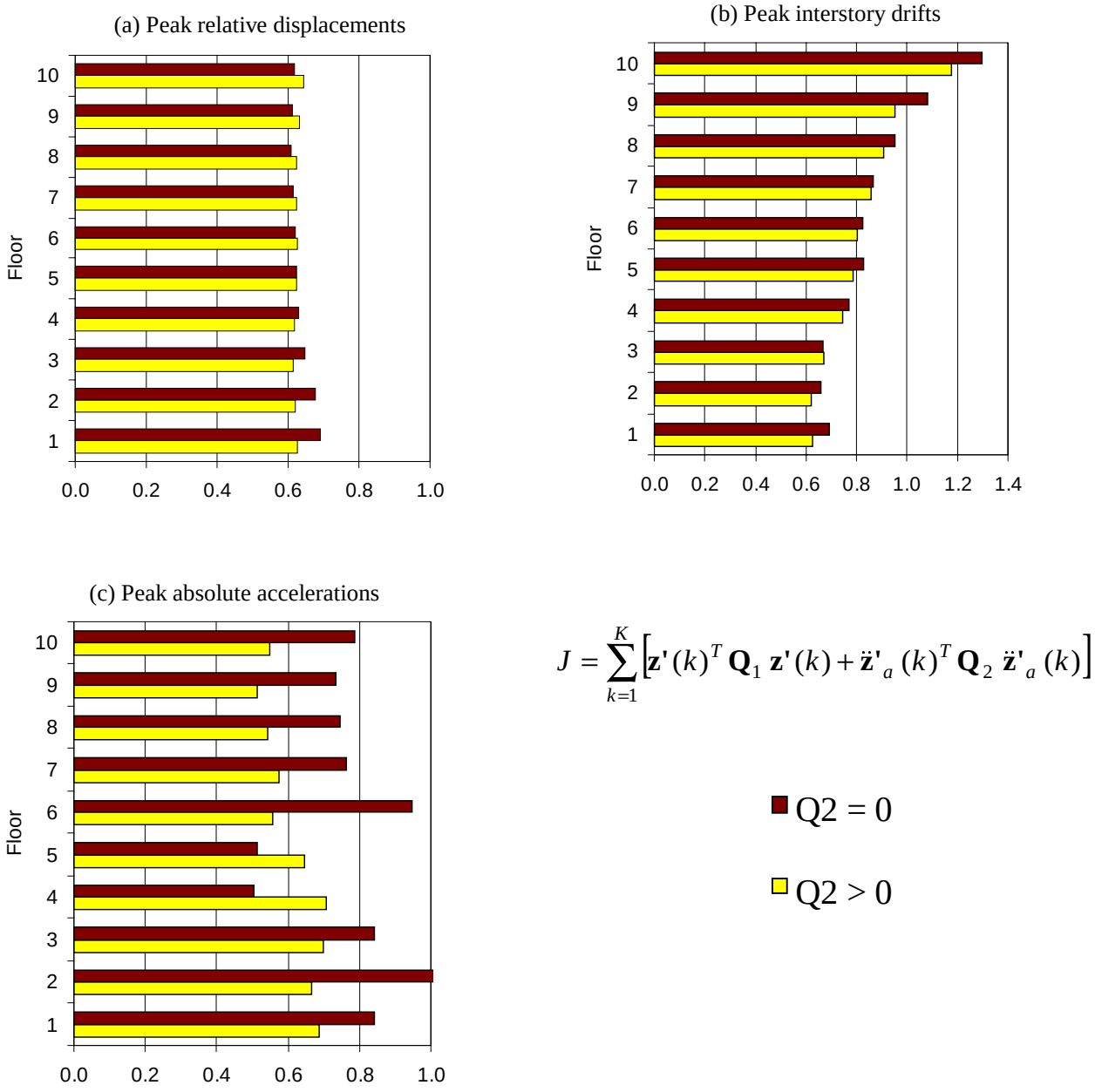
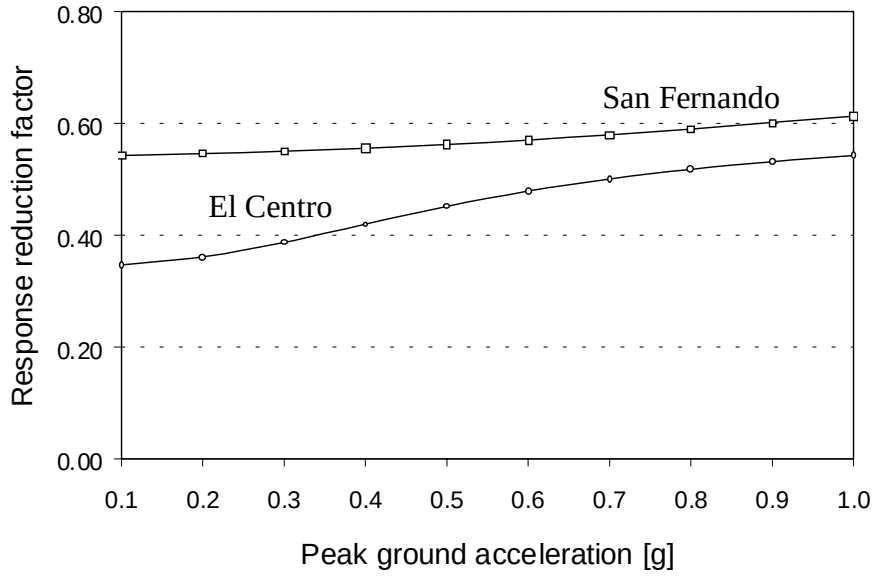
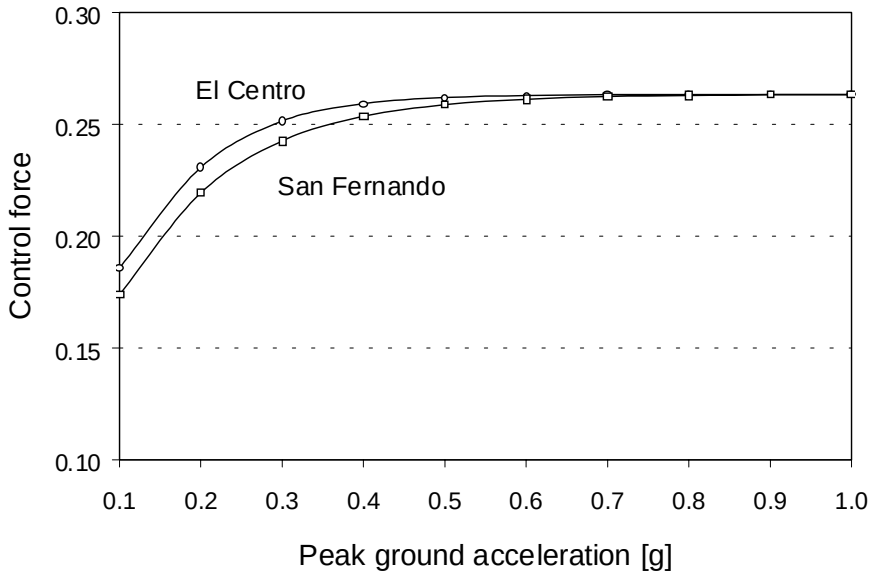


Figure 7-12: Response-reduction factors obtained for two different training cost functions, *J*, for (a) peak relative displacements, (b) peak interstory drifts, and (c) peak absolute accelerations. San Fernando earthquake.

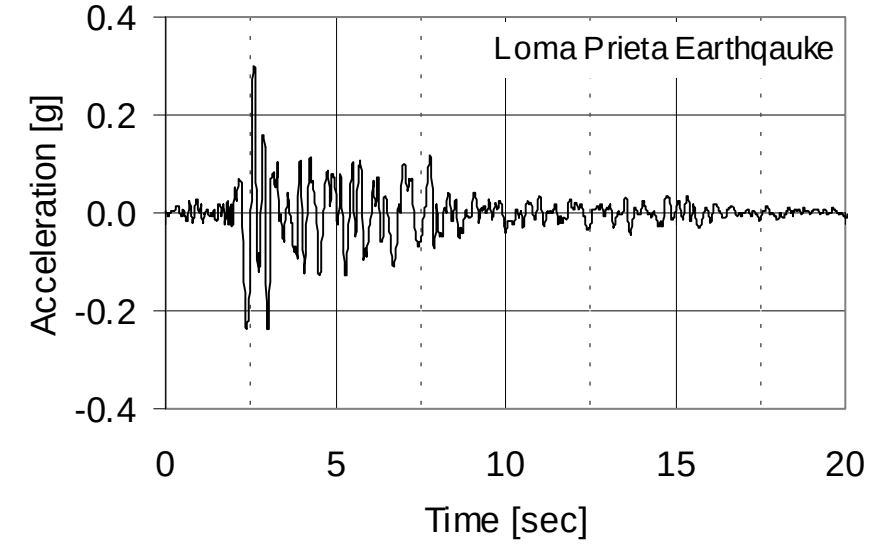


(a)

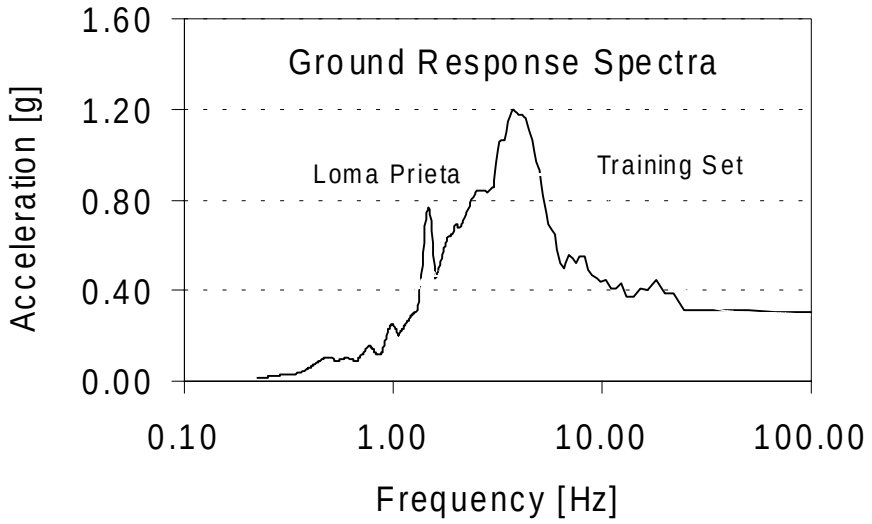


(b)

Figure 7-13: (a) Peak base shear reduction factor and (b) peak control force as functions of peak ground acceleration, for a velocity-feedback controller.



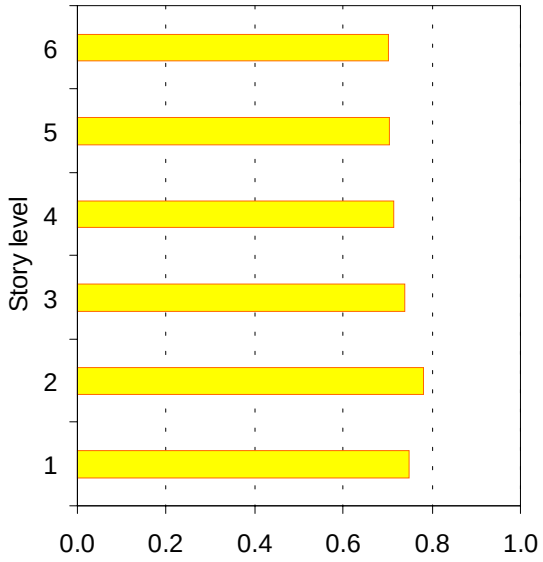
(a)



(b)

Figure 7-14: (a) Loma Prieta ground-acceleration time history. (b) Ground response spectra of the training set of earthquakes (average) and Loma Prieta earthquake.

(a) Peak relative



(b) Peak absolute accelerations

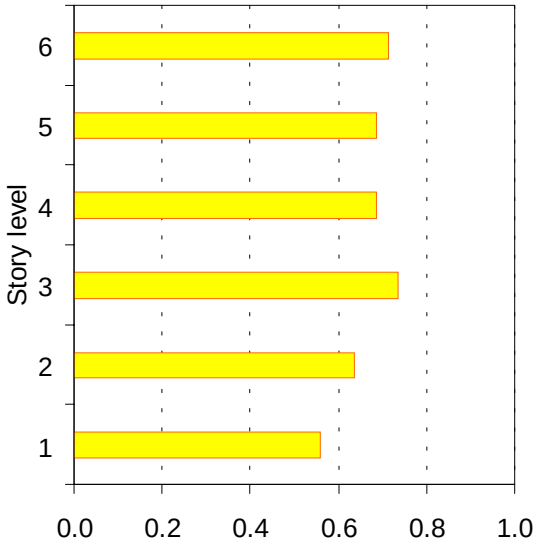
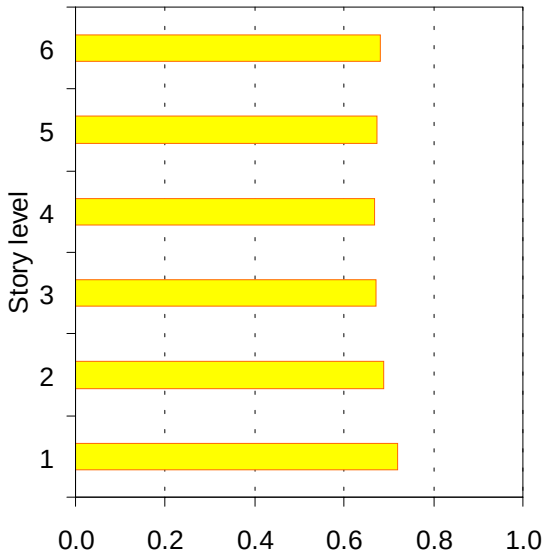


Figure 7-15: (a) Peak relative-displacement and (b) peak absolute-acceleration reduction factors for Loma Prieta earthquake. Six-story hysteretic building model. Velocity-feedback control (interstory velocities).

(a) Peak relative



(b) Peak absolute accelerations

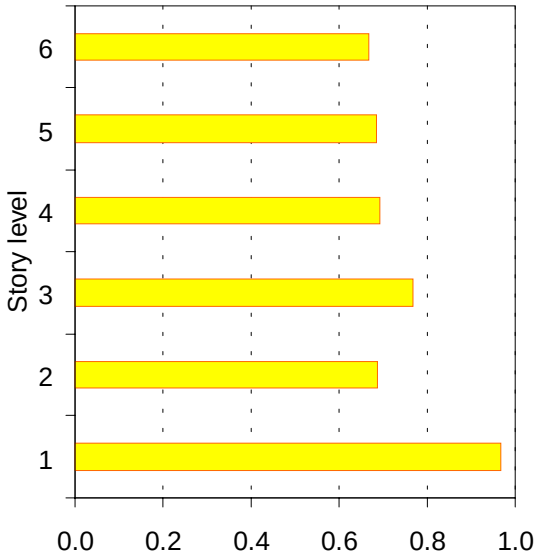


Figure 7-16: (a) Peak relative-displacement and (b) peak absolute-acceleration reduction factors for El Centro earthquake. Six-story hysteretic building model. Velocity-feedback control (interstory velocities).

Table 7-1: Response-reduction factors for different control schemes. Linear building model.

(a) El Centro

	Relative Displacements			Interstory Drifts			Absolute Accelerations		
Floor	**Passive**	**Accel. F.**	**Vel. F.**	**Passive**	**Accel. F.**	**Vel. F.**	**Passive**	**Accel. F.**	**Vel. F.**
1	0.603	0.431	0.387	0.603	0.431	0.387	0.893	0.776	0.708
2	0.595	0.441	0.382	0.586	0.451	0.377	1.015	0.879	0.572
3	0.587	0.454	0.377	0.569	0.481	0.371	0.907	0.753	0.518
4	0.577	0.465	0.373	0.591	0.483	0.381	0.675	0.605	0.430
5	0.571	0.474	0.371	0.672	0.519	0.435	0.775	0.656	0.441
6	0.580	0.479	0.378	0.695	0.524	0.442	0.652	0.673	0.415
7	0.598	0.478	0.390	0.675	0.484	0.442	0.830	0.677	0.498
8	0.609	0.473	0.399	0.649	0.497	0.479	0.875	0.632	0.532
9	0.616	0.471	0.410	0.669	0.525	0.558	0.736	0.558	0.463
10	0.619	0.469	0.421	0.644	0.956	0.786	0.729	0.540	0.441

(b) San Fernando

	Relative Displacements			Interstory Drifts			Absolute Accelerations		
Floor	**Passive**	**Accel. F.**	**Vel. F.**	**Passive**	**Accel. F.**	**Vel. F.**	**Passive**	**Accel. F.**	**Vel. F.**
1	0.770	0.537	0.550	0.770	0.537	0.550	1.002	0.704	0.708
2	0.774	0.553	0.560	0.773	0.572	0.579	0.952	0.609	0.616
3	0.768	0.565	0.574	0.751	0.641	0.661	0.923	0.458	0.510
4	0.749	0.585	0.598	0.814	0.664	0.697	0.912	0.371	0.421
5	0.695	0.592	0.609	0.889	0.675	0.703	0.885	0.431	0.486
6	0.694	0.589	0.608	0.930	0.704	0.683	0.884	0.683	0.835
7	0.697	0.583	0.598	0.954	0.726	0.750	0.845	0.681	0.656
8	0.703	0.583	0.586	0.975	0.756	0.797	0.924	0.790	0.683
9	0.716	0.587	0.586	0.929	0.837	0.855	0.937	0.593	0.683
10	0.727	0.594	0.591	0.913	1.111	1.208	0.923	0.493	0.503

Table 7-2: Response-reduction factors for different control schemes. Hysteretic building model.

Floor	El Centro		Loma Prieta		San Fernando	
	Passive	Active	Passive	Active	Passive	Active
	a) Relative displacements					
1	0.912	0.721	0.860	0.749	0.904	0.716
2	0.893	0.688	0.915	0.781	0.898	0.763
3	0.880	0.671	0.874	0.739	0.819	0.760
4	0.873	0.669	0.842	0.715	0.793	0.770
5	0.868	0.673	0.824	0.704	0.802	0.781
6	0.865	0.681	0.814	0.701	0.813	0.786
	b) Absolute accelerations					
1	1.080	0.967	0.850	0.559	0.903	0.900
2	1.027	0.688	0.846	0.635	0.871	0.586
3	0.975	0.768	0.861	0.736	0.882	0.598
4	0.924	0.693	0.881	0.686	0.970	0.805
5	0.893	0.685	0.942	0.686	0.909	0.827
6	0.910	0.668	0.989	0.713	0.955	0.789
	c) Control requirements					
TMD Stroke	0.153	0.293	0.131	0.184	0.084	0.146
Control Force	-	0.202	-	0.200	-	0.224