REAL-TIME INTEGRAL BASED STRUCTURAL HEALTH MONITORING

A thesis

submitted in partial fulfilment

of the requirements for the Degree

of

Master of Mechanical Engineering

in the

University of Canterbury

By

Ishan Singh-Levett

University of Canterbury

2006

ABSTRACT

Structural Health Monitoring (SHM) is a means of identifying damage from the structural response to environmental loads. Real-time SHM offers rapid assessment of structural safety by owners and civil defense authorities enabling more optimal response to major events. This research presents an real-time, convex, integral-based SHM methods for seismic events that use only acceleration measurements and infrequently measured displacements, and a non-linear baseline model including hysteretic dynamics and permanent deformation. The method thus identifies time-varying pre-yield and post-yield stiffness, elastic and plastic components of displacement and final residual displacement. For a linear baseline model it identifies only time-varying stiffness. Thus, the algorithm identifies all key measures of structural damage affecting the immediate safety or use of the structure, and the long-term cost of repair and retrofit.

The algorithm is tested with simulated and measured El Centro earthquake response data from a four storey non-linear steel frame structure and simulated data from a two storey non-linear hybrid rocking structure. The steel frame and rocking structures exhibit contrasting dynamic response and are thus used to highlight the impact of baseline model selection in SHM. In simulation, the algorithm identifies stiffness to within 3.5% with 90% confidence, and permanent displacement to within 7.5% with 90% confidence. Using measured data for the frame structure, the algorithm identifies final residual deformation to within 1.5% and identifies realistic stiffness values in comparison to values predicted from pushover analysis. For the rocking structure, the algorithm accurately identifies the different regimes of motion and linear stiffness comparable to estimates from previous research. Overall, the method is seen to be accurate, effective and real-time capable, with the non-linear baseline model more accurately identifying damage in both of the disparate structures examined.

ACKNOWLEDGEMENTS

Firstly, I would like to thank Associate Professor J. Geoffrey Chase. Having been my supervisor since my final undergraduate year, his advice and guidance has been invaluable and I simply would not have got where I am without him. A huge thanks must also go to Dr Chris Hann whose mathematical expertise has been a vital part of any progress made in my research. From the Civil Engineering Department I would like to thank my associate supervisor Dr Bruce Deam and also Professor John Mander. They have both provided me with a greatly improved understanding of structural engineering and helped me to maintain perspective on the overall direction of the project.

Additional thanks must go to:

- Dr Athol Carr for his help with modeling and time-history analysis
- Dr Rajesh Dhakal for his advice and willingness to always help
- My friend and colleague Samuel J. Houghton with whom I went through engineering school right from the first day
- My friends, who have made my time in Christchurch something to cherish
- My parents and sister, who have always been supportive of me in whatever I have chosen

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

PART I

INTRODUCTION

CHAPTER 1 - INTRODUCTION

1.1 Motivation

Structural Health Monitoring (SHM) is the process of comparing the current state of a structure's condition relative to a baseline state using measured response data, and determining the existence, location, and degree of damage that may exist, particularly after a damaging input, such as an earthquake or other large environmental load. SHM can simplify typical procedures of visual or localized experimental methods, such as acoustic or ultrasonic methods, magnetic field methods, radiography, eddy-current methods or thermal field methods (Doherty, 1987), as it does not require visual inspection of the structure and its connections or components.

The ability to assess damage in real-time or immediately after a catastrophic event, such as an earthquake or bomb blast, would allow Civil Defence authorities to determine which instrumented structures were safe. In addition, the damage state of surrounding non-instrumented office buildings, bridges and houses could be inferred from the damage state of instrumented structure determined using this data in conjunction with pre-determined fragility relationships. Hence, SHM in real-time would allow the optimization of response and recovery after an event, thereby reducing economic and social costs caused by disabled or unusable infrastructure.

In 1999, the International Association for Structural Control (IASC) and the Dynamics committee of the American Society of Civil Engineers (ASCE) Engineering Mechanics Division formed the SHM Task Group to study the efficacy of various SHM methods. The IASC-ASCE SHM Task Group developed a series of Benchmark SHM problems and established a set of

specific Benchmark results for a specially designed test structure in the Earthquake Engineering Research Laboratory at the University of British Columbia (Johnson et al, 2000). After the Benchmark problem was established, SHM research for civil structure concentrated on applying different techniques to the Benchmark problem to examine the relative and absolute effectiveness of different algorithms.

1.2 Existing SHM Techniques

Many current vibration-based SHM methods, particularly for large civil structures, are based on modal parameter damage detection in both the time series and frequency domain (Doebling et al, 1996). These modal methods are typically more applicable to steel-frame and bridge structures where vibration response is highly linear (Chase et al, 2004; Doebling et al, 1996; Doherty, 1987). Additionally, many existing algorithms require detailed knowledge of the structure and assume that response data from the undamaged structure is available (Farrar and Doebling, 1997). Doebling et al (1996) has an excellent review of the numerous different approaches for vibration-based damage detection methods. However, the various studies apply different methods to different structures, rendering side-by-side comparison difficult.

Modal-based methods can be insensitive to localised damage and non-robust in the presence of noise. Also, many current methods are unable to be implemented in real-time, as the event occurs. For example, current wavelet and ERA (Eigensystem Realisation Algorithm) methods (Lus and Betti, 2000; Caicedo et al, 2000) require the entire measured response to process and identify damage. Hence, they cannot provide and update structural damage measures until some time afterwards. It should be noted that modal parameter based methods have been shown to work well in a variety of other application areas (Farrar and Doebling, 1997).

Other identification methods with the potential to provide near real-time SHM have been employed to identify modal parameters by using the adaptive fading Kalman filter technique (Loh et al, 2000), and an Adaptive H_∞ Filter (Sato and Qi, 1998). However these methods involve significant computational modeling complexity, rendering them unlikely for easy real-time implementation. Others have mixed Bayesian and modal parameter identification methods to better manage sensor noise and baseline model uncertainty (Ching and Beck, 2004).

The ASCE Journal of Engineering Mechanics ran a recent special edition on the SHM Benchmark Problem. The methods presented included Bayesian statistical approaches using one or two stages to identify modal parameters and then damage (Yuen et al, 2004; Lam et al, 2004). Bernal and Gunes (2004) presented a flexibility based method that involved sub-matrix inverses and the full data record to perform modeless identification. Lus et al (2004) and Caicedo et al (2004) presented ERA based methods with the former using a Kalman filter estimator to identify a baseline model and the ERA method for modal parameters, and the latter identified modal parameters before using least squares optimisation to locate and identify damage. Finally, Yang et al (2004) used Hilbert-Huang linear transforms to identify model changes over the full record similar to the work by Allison and Abbott (2001). Finally, Hera and Hou (2004) used wavelet methods to identify damage. None of these methods focused on sample-to-sample real-time identification using minimal or simplest-possible methods. However, in summary, these works span the breadth of most current fundamental approaches.

To qualify as an early warning system for real-time civil defence application, an SHM method must be capable of identifying localized damage, be robust in the presence of noise and evaluate structural health rapidly or in real-time (Buyukozturk and Yu, 2003). Though many existing

methods provide structural health monitoring with significant post-event processing, real-time methods are a recent advent in the field.

Chase et al (2004a) identified changes in structural stiffness in real-time for the benchmark problem using LMS-based adaptive filtering. The same real-time LMS-based method was applied to a highly non-linear hybrid rocking structure (Chase et al, 2005), successfully identifying changes in stiffness which are representative of the structures various response regimes. However, this method requires measurement and or estimation of structural acceleration, velocity and displacement, which is impractical in many realistic cases due to excessive sensor requirements. Also, the LMS-based method uses a linear or bi-linear elastic baseline model to identify changes in stiffness, rendering it unable to capture permanent structural deformations due to ductile behaviour.

1.3 Proposed approach to the SHM problem

The approach presented in this research uses an integral-based linear least squares method to identify structural stiffness and permanent displacement using only measured accelerations with low frequency displacement measurement for correction. Thus, it reduces the sensor requirements and identifies relevant and important structural damage metrics, such as permanent deformation that other methods cannot identify. This is achieved by using a non-linear Bouc-Wen Hysteresis model (Bouc, 1967 and Wen, 1976) or a linear baseline model and measured ground acceleration and structural response data. The resulting method can be readily implemented in real time as it only requires minimal computation, is robust in the presence of noise, and can accurately identify localized damage, thereby fulfilling all major requirements of an SHM system.

In this research, the algorithm is first tested using data simulated in MATLAB using a standard shear building arrangement and the Bouc-Wen Hysteresis model. Next, the algorithm is applied to measured data from shaking table tests of a four storey, non-linear steel frame structure. Finally, a highly non-linear, two storey hybrid rocking structure is used to test the algorithm's ability to identify damage in a structure radically different from the shear building arrangement for which it was originally intended. Acceleration and displacement data for the rocking structure was generated by Chase et al (2005), using the time history analysis package RUAUMOKO (Carr, 2004).

Each structure is tested using both the linear and non-linear Bouc-Wen baseline models. Like the LMS-based approach of Chase et al (2004, 2005) and many other methods that identify damage using a baseline model, the selection of baseline model may have significant impact on the results. This research investigates the abilities of each baseline model to identify the existence, location and degree of damage when applied to each of the structures. Additionally, this research describes the necessary interpretation of identified structural parameters required when considering highly non-linear structural behaviour in conjunction with the application of potentially inappropriate baseline models.

PART II
METHODOLOGY

CHAPTER 2 - SINGLE DEGREE OF FREEDOM SYSTEM IDENTIFICATION

2.1 Stiffness Identification for Linear System

The motion of a structure undergoing seismic motion which has no hysteresis or permanent displacement can be modelled as follows:

$$\mathbf{M\ddot{x}(t) + C\dot{x}(t) + K(t)x(t) = -M\ddot{x}_g(t)} \tag{2.1}$$

where **M** is the mass matrix, **C** is the viscous damping matrix, **K(t)** is the time-varying stiffness matrix, $\mathbf{x(t)}$ is the displacement vector, $\mathbf{\dot{x}(t)}$ is the velocity vector, $\mathbf{\ddot{x}(t)}$ is the acceleration vector and $\mathbf{\ddot{x}_g(t)}$ is the ground acceleration vector. In this special case, it is possible to identify **K(t)** using only knowledge of the structural acceleration $\mathbf{\ddot{x}}$ and ground acceleration $\mathbf{\ddot{x}_g}$.

For a single degree of freedom system, Equation (2.1) reduces to the form:

$$m\ddot{x} + c\dot{x} + k(t)x = -m\ddot{x}_g \tag{2.2}$$

where *m*, *c*, and *k(t)* are scalar system quantities, and $\ddot{x}$, $\dot{x}$ and x are scalar model outputs and $\ddot{x}_g$ is as defined in Equation (2.1).

To identify the time-varying stiffness *k(t)* using the measured structural and ground acceleration data, the displacement and velocity terms in Equation (2.2) are replaced by integral approximations defined:

$$\dot{x} = \int_0^t \ddot{x} dt + \delta \tag{2.3}$$

$$x = \int_0^t \int_0^t \ddot{x} dt + \delta t + \sigma \tag{2.4}$$

where δ and σ are constants defined to take account of non-zero initial conditions and integration errors, or drift, due to noise on the measured acceleration signal.

Substituting Equations (2.3) and (2.4) into Equation (2.2), a revised equation of motion is developed:

$$m\ddot{x} + c\left[\int_0^t \ddot{x} dt + \delta\right] + k(t)\left[\int_0^t \int_0^t \ddot{x} dt + \delta t + \sigma\right] = -m\ddot{x}_g \tag{2.5}$$

Assuming that *k(t)* is a piecewise constant function over *N* fixed time intervals of length Δt, *k(t)* can be defined:

$$k(t) = k_i \quad , \quad (i-1)\Delta t \le t \le i\Delta t \quad , \quad i = 1,.., N \tag{2.6}$$

Equation (2.5) can then be reformulated:

$$k_i \int_0^t \int_0^t \ddot{x} dt dt + \bar{\delta}_i t + \bar{\sigma}_i = -m\ddot{x}_g - m\ddot{x} - c\int_0^t \ddot{x} dt \quad , \quad (i-1)\Delta t \le t \le i\Delta t \quad , \quad i = 1,..,N \tag{2.7}$$

where

$$\bar{\delta}_i = \delta_i k_i \tag{2.8}$$

$$\bar{\sigma}_i = k_i \sigma + \delta_i c \tag{2.9}$$

To identify each k_i , $i = 1,..,N$ it is necessary to construct a system of linear equations from the equation of motion at various evenly spaced times. For example, if five values of *t* are chosen in each time interval $t_i \le t \le t_i + \Delta t$, $i = 1,..,N$, there will be a total of 5*N* linear equations. Each linear equation will be in the form of Equation (2.7) and this leads to 3*N* unknowns k_i , $\bar{\delta}_i$ and $\bar{\sigma}_i$.

These linear equations can be represented as N matrix equations each of the form:

$$\mathbf{A_i} \begin{Bmatrix} k_i \\ \bar{\delta}_i \\ \bar{\sigma}_i \end{Bmatrix} = \mathbf{b_i} \quad , \quad i = 1,..,N \tag{2.10}$$

where $\mathbf{A_i}$ is a 5*x*3 matrix for this example of the form:

$$\mathbf{A_i} = \begin{bmatrix} \int_0^{t_i1}\int_0^{t_i1} \ddot{x}(t)dtdt & t_{i1} & 1 \\ \int_0^{t_i2}\int_0^{t_i2} \ddot{x}(t)dtdt & t_{i2} & 1 \\ \int_0^{t_i3}\int_0^{t_i3} \ddot{x}(t)dtdt & t_{i3} & 1 \\ \int_0^{t_i4}\int_0^{t_i4} \ddot{x}(t)dtdt & t_{i4} & 1 \\ \int_0^{t_i5}\int_0^{t_i5} \ddot{x}(t)dtdt & t_{i5} & 1 \end{bmatrix} \tag{2.11}$$

and $\mathbf{b_i}$ is a 5*x*1 vector for this example of the form:

$$\mathbf{b_i} = \begin{Bmatrix} -m\ddot{x}_g(t_{i1}) - m\ddot{x}(t_{i1}) - c\int_0^{t_i1} \ddot{x}(t)dt \\ -m\ddot{x}_g(t_{i2}) - m\ddot{x}(t_{i2}) - c\int_0^{t_i2} \ddot{x}(t)dt \\ -m\ddot{x}_g(t_{i3}) - m\ddot{x}(t_{i3}) - c\int_0^{t_i3} \ddot{x}(t)dt \\ -m\ddot{x}_g(t_{i4}) - m\ddot{x}(t_{i4}) - c\int_0^{t_i4} \ddot{x}(t)dt \\ -m\ddot{x}_g(t_{i5}) - m\ddot{x}(t_{i5}) - c\int_0^{t_i5} \ddot{x}(t)dt \end{Bmatrix} \tag{2.12}$$

The linear system of Equation (2.10) is over determined. Thus the required unknowns k_i, $\overline{\delta}_i$, and $\overline{\sigma}_i$, $i = 1,..,N$ can be found by linear least squares.

2.2 Bouc-Wen Hysteresis Model and Identification

The motion of a structure undergoing earthquake acceleration with Bouc-Wen hysteresis is given by the matrix equation defined: (Bouc, 1967 and Wen, 1976)

$$\mathbf{M\ddot{x}(t) + C\dot{x}(t) + K_e x(t) + K_h z(t) = -M\ddot{x}_g(t)} \tag{2.13}$$

where $\mathbf{K_e}$ is the pre-yield, linear-elastic stiffness matrix and $\mathbf{K_h}$ is the post-yield stiffness matrix. $\mathbf{K_e}$ and $\mathbf{K_h}$ can be expressed in terms of the pre-yield stiffness $\mathbf{K_p}$:

$$\mathbf{K_e} = \alpha \mathbf{K_p} \tag{2.14}$$

$$\mathbf{K_h} = (1-\alpha)\mathbf{K_p} \tag{2.15}$$

where α is the bi-linear factor that determines the change in slope between elastic and plastic regimes and typically has a value of $\alpha = 0.02 - 0.1$.

The vector **z(t)** represents hysteretic displacement and is governed by the equation (Wen, 1976):

$$\dot{z}_i(t) = \dot{r}_i(t)\left[1 - 0.5[1 + \mathrm{sgn}(\dot{r}_i(t) z_i(t))]\left|\frac{z_i(t)}{Y_i}\right|^{ni}\right] \quad , \quad i = 1,..,\overline{N} \tag{2.16}$$

where $\dot{r}_i(t)$ is the velocity of storey i relative to storey $i-1, \overline{N}$ is the number of degrees of freedom, Y is the yield displacement and n_i is a shaping parameter determining the curve from elastic to plastic force-deflection behaviour. Permanent displacement of an element undergoing Bouc-Wen Hysteresis is defined:

$$D_i(t) = \frac{r_i(t) - z_i(t)}{\left[1 + \dfrac{\alpha_i}{1-\alpha_i}\right]} \tag{2.17}$$

where $D_i(t)$ is the permanent deformation of storey i and $r_i(t)$ is the displacement of storey i relative to storey $i-1$ found by:

$$r_i(t) = x_i(t) - x_{i-1}(t) \tag{2.18}$$

In the identification method $\mathbf{M}$ and $\mathbf{C}$ are assumed to be reasonably well known and constant. Therefore, the variables to be identified are pre-yield stiffness $\mathbf{K_p}$, which yields $\mathbf{K_e}$ and $\mathbf{K_h}$ and the permanent displacement $D(t)$. All other values are estimated or known.

2.2.1 Displacement and Velocity Estimation

Identification of $\mathbf{K_p}$ and $D(t)$ from Equation (2.13) and (2.17) requires knowledge of the ground acceleration and the structural displacement, velocity and acceleration. Due to a variety of practical constraints, direct high frequency measurement of displacement and velocity is not typically possible for a real structure whereas acceleration measurements might be readily instrumented.

Displacement and velocity are therefore estimated by integration of measured acceleration. Integrated estimates are naturally subject to drift and numerical error. However, this error can be corrected using displacement data measured at relatively very low frequency. This low frequency displacement can be reasonably obtained via a variety of sensors, such as ground-based GPS, fibreobtics or story drift extensometers (Kejewski-Correa and Kochly, 2006; Brownjohn et al, 2003 and Leng et al, 2004).

For the results presented in this research, the displacement measurement was taken at 10Hz and assumed to be a 100pt backward moving average of 1kHz sampled displacement data which was available from both simulation and experimental testing. The backward moving average was calculated using the following equation:

$$x_{meas}(t) = \frac{\sum_{k=0}^{99}\left[x(t-0.001k+y+1)\bar{f}^{k}\right]}{\sum_{k=0}^{99}\bar{f}^{k}} \tag{2.19}$$

where $\bar{f}$ is a weighting factor typically valued between 1 and 1.1 and $x_{meas}(t)$ is the 10Hz measured displacement at time t.

The integrated displacement $x(t)$ and velocity $\dot{x}(t)$ are defined as follows:

$$x(t) = \int_0^t \left[\int_0^t \ddot{x} dt \right] dt \tag{2.20}$$

$$\dot{x}(t) = \int_0^t \ddot{x}(t) dt \tag{2.21}$$

where $\ddot{x}(t)$ is the measured 1kHz sampled acceleration data and the integrals refer to numerical integration for more compact notation.

The integration displacement error to be corrected by measured 10Hz displacement is defined by Equation (2.22):

$$e_i = x_{meas}(hi) - x(hi) \quad , \quad i = 1,..,N \tag{2.22}$$

where e_i is the error at the time $t = hi$ and h is the time interval at which displacement measurements are reported, $h = 0.1\,\text{s}$ for the case where displacement is measured at 10Hz.

The measured and integrated displacements x_{meas} and x are given by Equations (2.19) and (2.20) respectively.

Similarly, the error, c_i, in the integrated velocity given by Equation (2.21) can thus be calculated by simple numerical differentiation of e_i:

$$c_i = (e_i - e_{i-1})/h \quad , \quad i = 1,..,N \tag{2.23}$$

Note that e_0 and c_0 are both assumed to be zero.

The displacement error e_i , $i = 1,..,N$ of Equation (2.22) can be linearly interpolated to define a continuous corrector function:

$$f(t) = e_i + c_i(t - h(i-1)) \quad , \quad h(i-1) \le t \le hi \quad , \quad i = 1,..,N \tag{2.24}$$

where

$$f(hi) = e_{i+1} \tag{2.25}$$

Therefore, one possible estimate for the true displacement $d(t)$ is as follows:

$$d(t) = x(t) + f(t) \tag{2.26}$$

However, numerically differentiating $d(t)$ in Equation (2.26) to estimate the velocity will cause discontinuities since $d(t)$ is not continuous. Furthermore, double differentiating $d(t)$ may cause spikes to occur so that $d''(t)$ is potentially significantly different from the measured acceleration $\ddot{x}(t)$. The end result is that the estimates to $x(t)$, $\dot{x}(t)$ and $\ddot{x}(t)$ in Equation (2.13) may not be mathematically related by differentiation alone with the resulting error potentially corrupting the fitted parameters of K_e, K_h and $\Delta Z(t)$. To ensure the estimates to $x(t)$, $\dot{x}(t)$ and $\ddot{x}(t)$ are compatible a C^1 continuous displacement estimate is constructed for the distance $d(t)$. C^1 continuity was found to provide adequately smooth velocity and acceleration estimates, with $d''(t)$ sufficiently close to $\ddot{x}(t)$.

A C^1 continuous displacement estimate is constructed by fitting a series of piecewise cubic curves to the displacement corrector function $\bar{f}(t)$ defined in Equation (2.13) with C^1 continuity at the joins as illustrated in Figure 2.1.

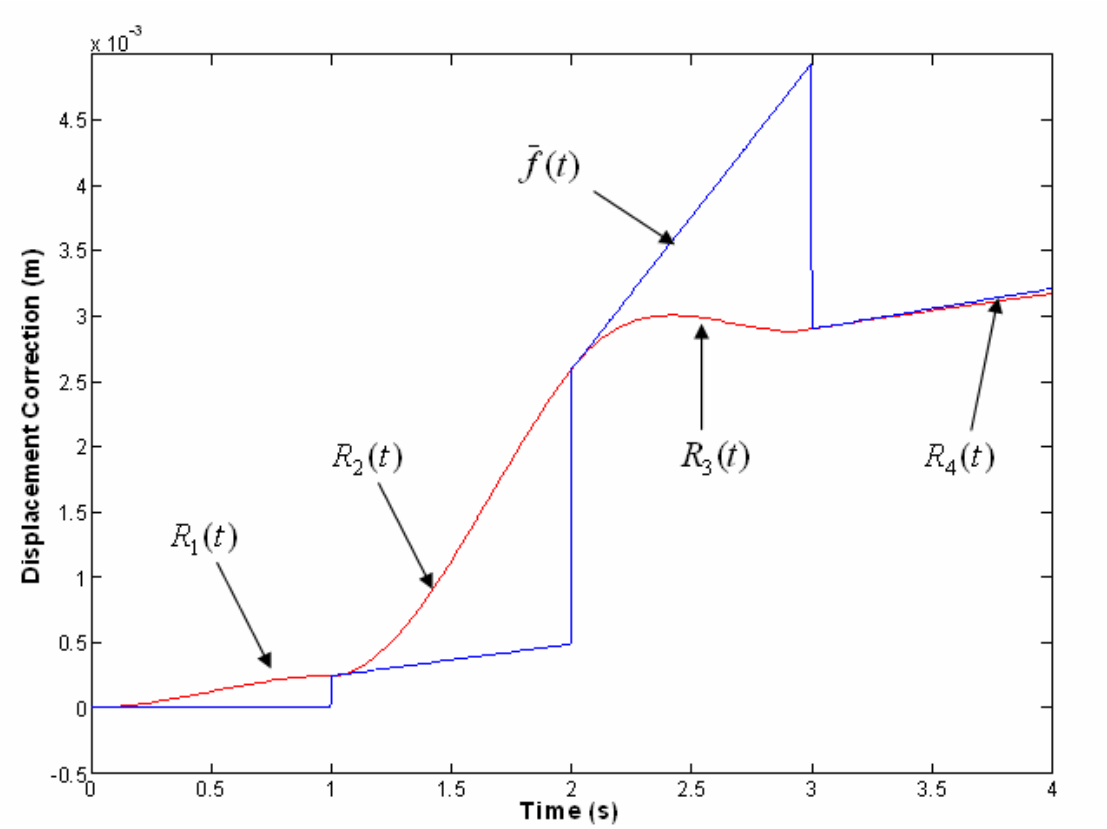

Figure 2.1: Constructing a C^1 continuous displacement corrector from the discontinuous function $\bar{f}(t)$ in Equation (2.24)

The C^1 continuous displacement corrector function is defined as follows:

$$\bar{f}(t) = R_i(t) \quad , \quad h(i-1) < t < hi \quad , \quad i = 1,..,N \tag{2.27}$$

$$R_i(t) = a_{i1}[t - h(i-1)]^3 + a_{i2}[t - h(i-1)]^2 + a_{i3}[t - h(i-1)] + a_{i4} \quad , \quad h(i-1)t < t < hi \tag{2.28}$$

where

$$\bar{f}(hi) = R_i(hi) = e_{i+1} \tag{2.29}$$

$$\bar{f}'(hi) = R_i'(hi) = c_i \tag{2.30}$$

where constraints are placed on the cubics by the following equations:

$$R_i(h(i-1)) = R_{i-1}(h(i-1)) \quad , \quad i = 1,..,N \tag{2.31}$$

$$R_i'(h(i-1)) = R_{i-1}'(h(i-1)) \quad , \quad i = 1,..,N \tag{2.32}$$

where

$$R_i'(t) = 3a_{i1}[t-h(i-1)]^2 + 2a_{i2}[t-h(i-1)] + a_{i3} \quad , \quad h(i-1)t < t < hi \tag{2.33}$$

Combining Equations (2.27)-(2.32), the conditions on the polynomial coefficients $\left[a_{i1}\ a_{i2}\ a_{i3}\ a_{i4}\right], i = 1,..,N$ can be expressed in matrix form by the following recursive equations:

$$\begin{bmatrix} h^3 & h^2 & h & 1 \\ 3h^2 & 2h & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} a_{i1} \\ a_{i2} \\ a_{i3} \\ a_{i4} \end{Bmatrix} = \begin{Bmatrix} e_i \\ c_i \\ 3a_{i-1,1}(hi)^2 + 2a_{i-1,2}(hi) + a_{i-1,3} \\ a_{i-1,1}(hi)^3 + a_{i-1,2}(hi)^2 + a_{i-1,3}(hi) + a_{i-1,4} \end{Bmatrix} \tag{2.34}$$

The time-varying corrected displacement may now be estimated by adding the integrated displacement and the continuous corrector function $\bar{f}(t)$ of Equation (2.27):

$$d(t) = \int_0^t \left[\int_0^t \ddot{x} dt \right] dt + \bar{f}(t) \quad , \quad 0.1(i-1) < t < 0.1(i) \quad , \quad i = 1,..,N \tag{2.35}$$

The time-varying corrected velocity and acceleration are then calculated by numerical differentiation:

$$v(t_i) = \frac{d(t_{i+1}) - d(t_{i-1})}{2\Delta t} \tag{2.36}$$

$$a(t_i) = \frac{v(t_{i+1}) - v(t_{i-1})}{2\Delta t} \tag{2.37}$$

Note that recalculating the already measured acceleration in Equation (2.37) ensures mathematical continuity via differentiation and integration of acceleration, velocity and displacement. Hence, no added error will be introduced into Equation (2.13) using these variables.

Figures 2.2-2.4 show an example where the true, simulated displacement, velocity and acceleration of the top storey for a four degree of freedom shear building subject to the El Centro earthquake record and sampled at 1kHz is compared with the values estimated using Equations (2.35)-(2.37). Note that only a portion of the strong motion is shown. The 'measured' acceleration data had 10% uniformly distributed noise applied and the less frequently sampled displacement correction data had 5% uniformly distributed noise applied. These errors are typical of extensometers for displacement and extreme for a reasonable accelerometer (Analog Instruments). Thus, they were chosen to test the method conservatively. The estimated displacement had less than 1.3% error at all points over the entire 30 second record, the estimated velocity less than 0.3% error and the estimated acceleration less than 5.3% error. These errors are well within the simulated noise levels.

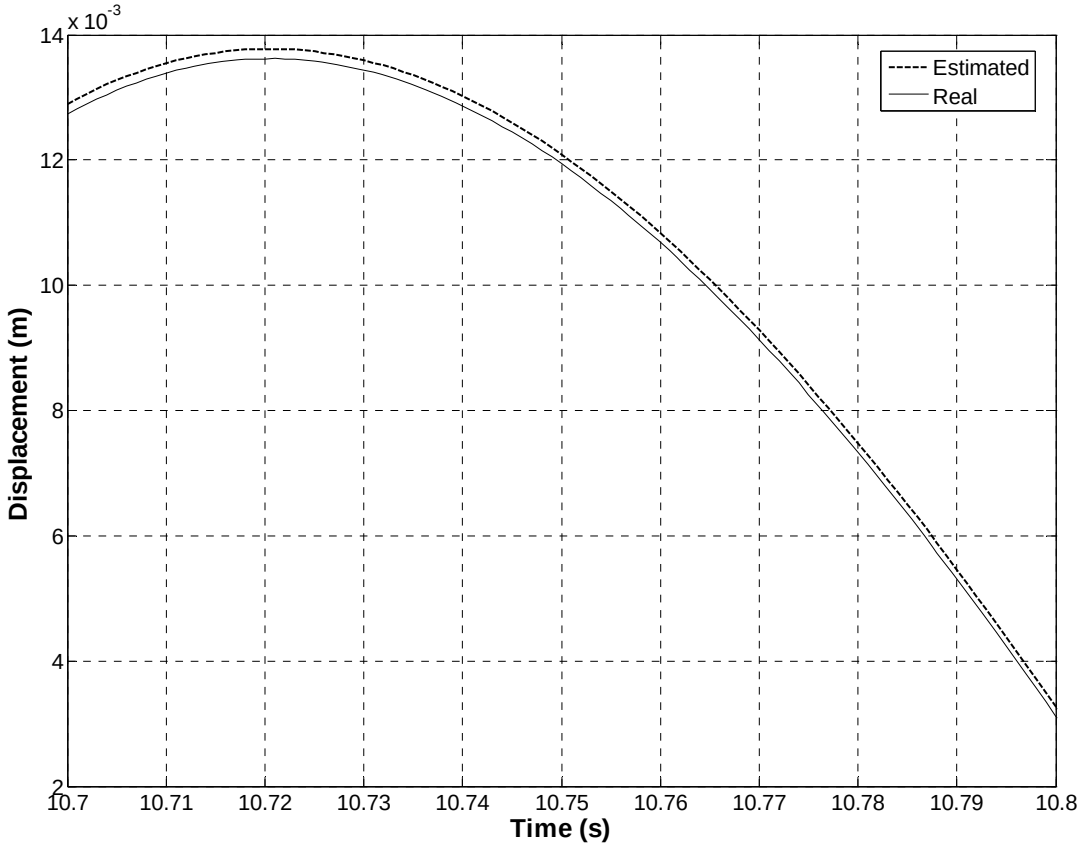

Figure 2.2: Comparison of Real and Estimated Displacements

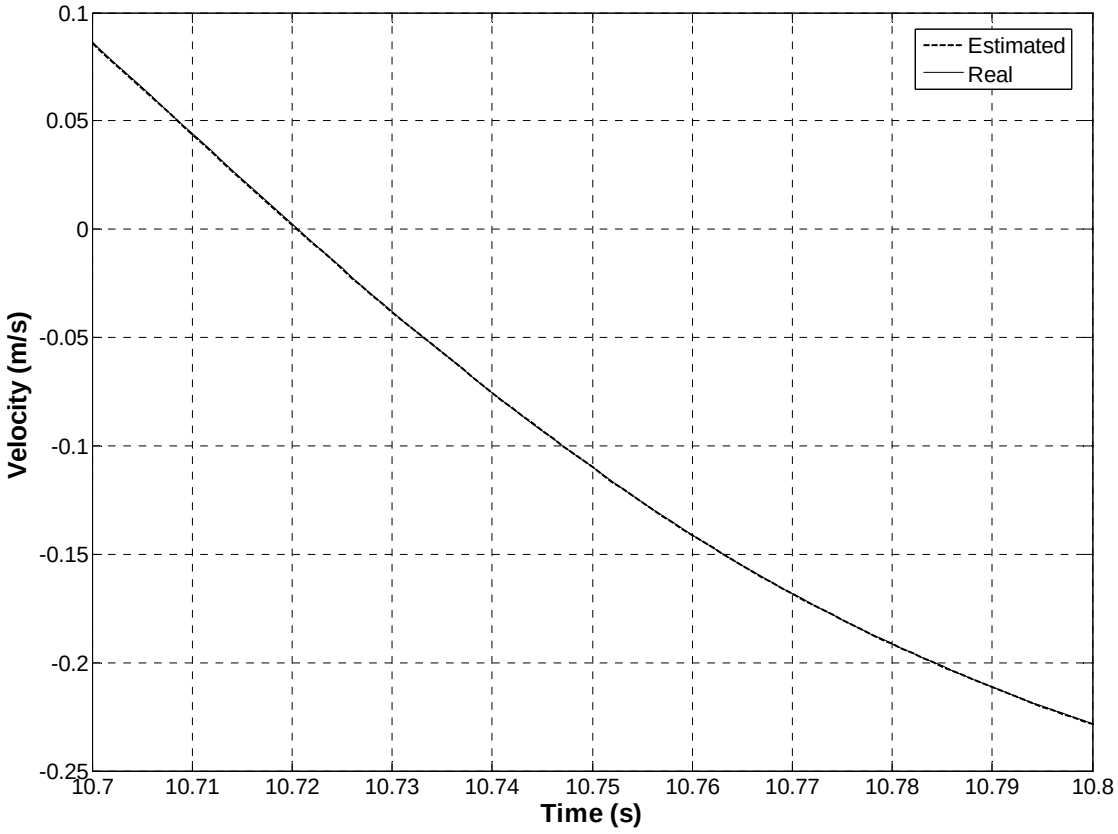

Figure 2.3: Comparison of Real and Estimated Velocities

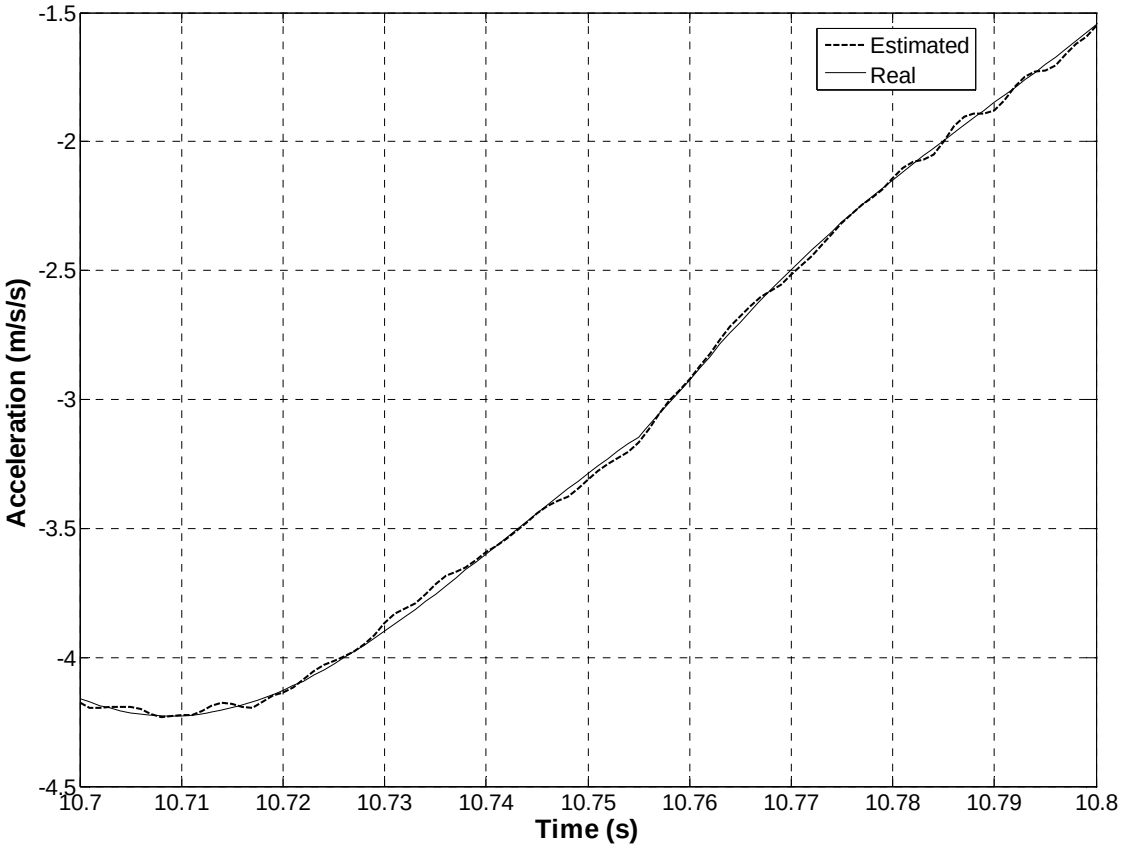

Figure 2.4: Comparison of Real and Estimated Accelerations

2.3 Identification of Stiffness and Permanent Displacement

For a single degree of freedom (SDOF) case, Equation (2.13) reduces to:

$$m\ddot{x}(t) + c\dot{x}(t) + k_e x(t) + k_h z(t) = -m\ddot{x}_g(t) \tag{2.38}$$

The aim is to identify the stiffnesses k_e and k_h, and the permanent displacement $D(t)$ using Equation (2.38), where $D(t)$ is defined in Equation (2.17).

First, the estimated acceleration $a(t)$, velocity $v(t)$ and displacement $d(t)$ are calculated using Equations (2.35-2.37) to approximate $\ddot{x}(t)$, $\dot{x}(t)$ and $x(t)$ in Equation (2.38) respectively.

Upon substitution of the estimated properties, Equation (2.38) can be rewritten:

$$k_p d(t) + \Delta Z(t) = -m\ddot{x}_g(t) - ma(t) - cv(t) \tag{2.39}$$

where

$$k_p = k_e + k_h \tag{2.40}$$

$$\Delta Z(t) = k_h[z(t) - x(t)] \tag{2.41}$$

The quantity $[z(t) - x(t)]$ is proportional to the permanent displacement of this structure with proportionality constant defined in Equation (2.17). Thus, $\Delta Z(t)$ is considered to be a permanent applied force due to deformation.

The parameters k_p and $\Delta Z(t)$ can now be found in a piecewise sense from Equation (2.39) where the term $d(t)$ on the left hand side and all terms on the right hand side are either measured or estimated.

To be as general as possible and potentially account for unmodelled dynamics, the parameter k_p is made time-varying as follows:

$$k_p(t) = k_{p,i} \quad , \quad (i-1)\Delta t_{kp} \le t \le i\Delta t_{kp} \quad , \quad i = 1,..,N \tag{2.42}$$

where Δt_{kp} is a user-selected interval over which piecewise constant behaviour is a reasonable approximation for the structure considered. The time-varying term $\Delta Z(t)$ can also be represented by a piecewise constant function over M fixed time intervals Δt_Z:

$$\Delta Z(t) = \Delta Z_j \quad , \quad (j-1)\Delta t_Z \le t \le j\Delta t_Z \quad , \quad j = 1,..,M \tag{2.43}$$

For ease of fitting Δt_{kp} is chosen to be an integer multiple l of Δt_Z:

$$\Delta t_{kp} = l\Delta t_Z \tag{2.44}$$

Thus, l values of ΔZ_j are fitted alongside every single value of $k_{p,i}$ as shown in Figure 2.5 for the case $l = 3$.

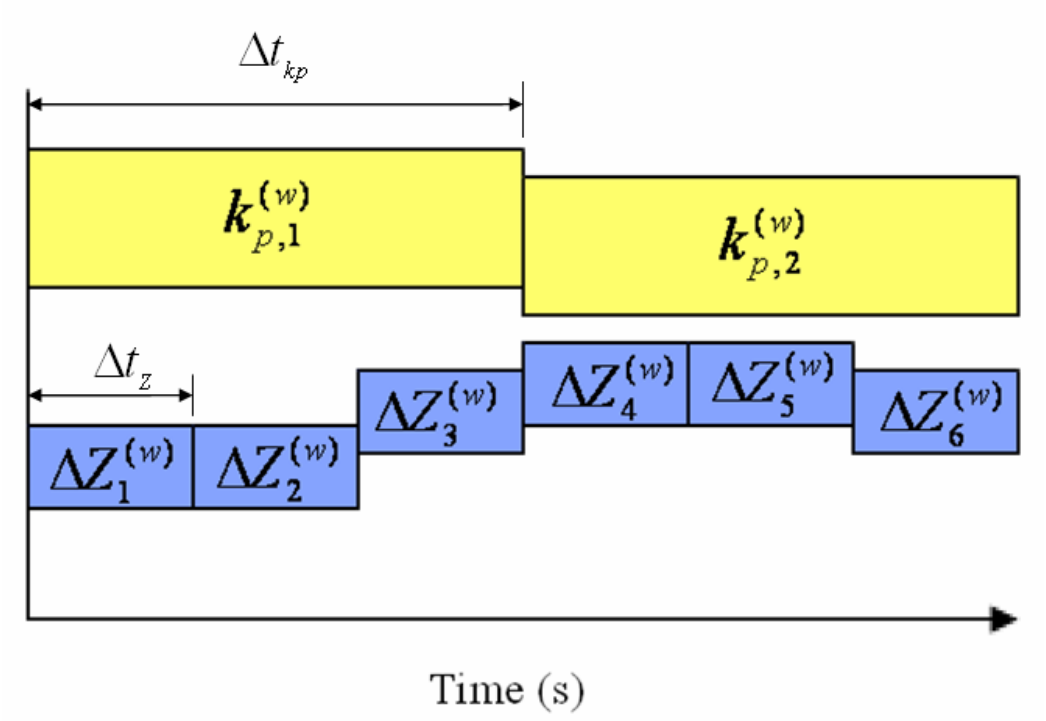

Figure 2.5: Time Variation of Fitted Parameters for $l = 3$

Equation (2.39) can now be rewritten as l equations for each interval Δt_{kp} corresponding to $i = 1,..,N$, as illustrated in Figure 2.5:

$$k_{p,i} d(t) + \Delta Z_j = -m\ddot{x}_g(t) - ma(t) - cv(t)\,,\ (j-1)\Delta t_Z \le t \le j\Delta t_Z\ ,\ j = 1,..,M\ ,\ i = 1,..,N \qquad (2.45)$$

Identification of k_p and $\Delta Z(t)$ requires the construction of a system of linear equations, each in the form of Equation (2.38). For the example of Figure 2.5, 3 values of *t* could be chosen in each time interval $(j-1)\Delta t_Z \le t \le j\Delta t_Z$ of Equation (2.45). This choice will give 9 equations for each time interval Δt_{kp} in Figure 2.5 and thus in total, $9N$ linear equations in $4N$ unknown parameters defined as follows:

$$\underline{\alpha}_i = \begin{Bmatrix} k_{p,i} \\ \Delta Z_{li-2} \\ \Delta Z_{li-1} \\ \Delta Z_{li} \end{Bmatrix} \quad , \quad i = 1,..,N \qquad (2.46)$$

$$\underline{\alpha} = \left[\underline{\alpha_1},..,\underline{\alpha_N}\right]^T \qquad (2.47)$$

The linear equations can be represented in a matrix equation of the form:

$$\mathbf{A_i}\,\underline{\alpha_i} = \mathbf{b_i} \qquad (2.48)$$

where $\mathbf{A_i}$ for this example is a 9x4 matrix of the form:

$$\mathbf{A_i} = \begin{bmatrix} d(t_{i1}) & 1 & 0 & 0 \\ d(t_{i2}) & 1 & 0 & 0 \\ d(t_{i3}) & 1 & 0 & 0 \\ d(t_{i4}) & 0 & 1 & 0 \\ d(t_{i5}) & 0 & 1 & 0 \\ d(t_{i6}) & 0 & 1 & 0 \\ d(t_{i7}) & 0 & 0 & 1 \\ d(t_{i8}) & 0 & 0 & 1 \\ d(t_{i9}) & 0 & 0 & 1 \end{bmatrix} \tag{2.49}$$

and $\mathbf{b_i}$ is a 9x1 vector of the form:

$$\mathbf{b_i} = \begin{Bmatrix} -m\ddot{x}_g(t_{i1}) - ma(t_{i1}) - cv(t_{i1}) \\ -m\ddot{x}_g(t_{i2}) - ma(t_{i2}) - cv(t_{i2}) \\ -m\ddot{x}_g(t_{i3}) - ma(t_{i3}) - cv(t_{i3}) \\ -m\ddot{x}_g(t_{i4}) - ma(t_{i4}) - cv(t_{i4}) \\ -m\ddot{x}_g(t_{i5}) - ma(t_{i5}) - cv(t_{i5}) \\ -m\ddot{x}_g(t_{i6}) - ma(t_{i6}) - cv(t_{i6}) \\ -m\ddot{x}_g(t_{i7}) - ma(t_{i7}) - cv(t_{i7}) \\ -m\ddot{x}_g(t_{i8}) - ma(t_{i8}) - cv(t_{i8}) \\ -m\ddot{x}_g(t_{i9}) - ma(t_{i9}) - cv(t_{i9}) \end{Bmatrix} \tag{2.50}$$

The least squares solution of the matrix Equation (2.48) determines the vector $\underline{\alpha}$ of unknowns. Assuming that the bi-linear factor α in Equation (2.14) is known as it is typically $\alpha = 0.02 - 0.1$, the parameters $\{k_{e,i}\ , i = 1,..,N\}$, $\{k_{h,i}\ , i = 1,..,N\}$ and $\{D_j\ , j = 1,..,M\}$ can then be found as follows:

$$k_{e,i} = \alpha k_{p,i} \tag{2.51}$$

$$k_{h,i} = (1-\alpha) K_{p,i} \tag{2.52}$$

$$D_j = \frac{-\Delta Z_j}{k_{h,i}\left(1+\frac{\alpha}{1-\alpha}\right)} \tag{2.53}$$

where $j = l(i-1)+1,..,li$ and $i = 1,..,N$.

Figure 2.6 shows a graphical description of the fitting process for a single degree of freedom non-linear system.

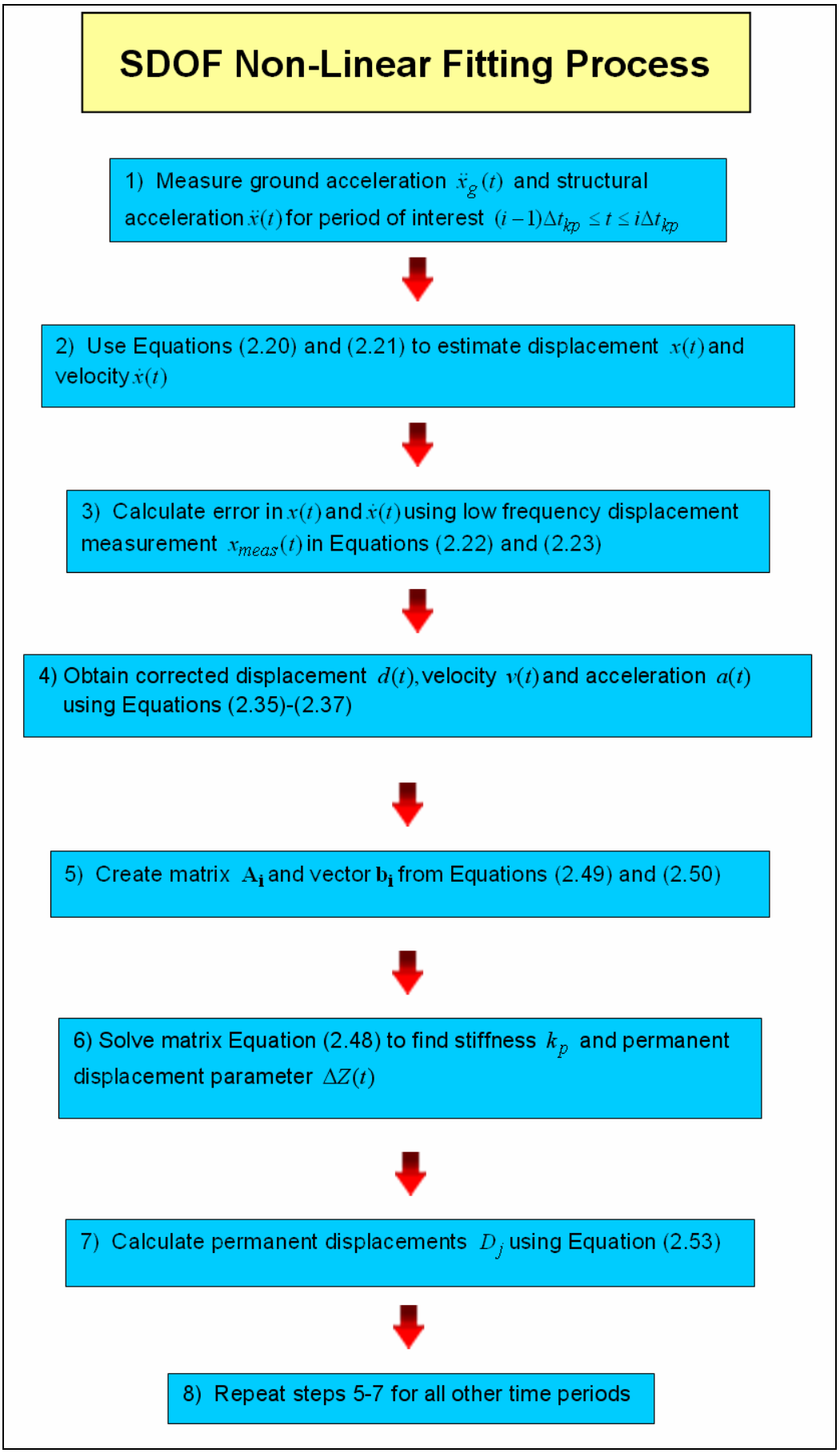

Figure 2.6: Flow chart depicting fitting process for single degree of freedom shear building

CHAPTER 3 - MULTIPLE DEGREE OF FREEDOM SYSTEM IDENTIFICATION

3.1 2-DOF System Identification

Writing Equation (2.13) as its constituent equations, a 2-DOF shear building model can be defined:

$$m_1\ddot{x}_1(t) + c_1\dot{x}_1(t) + k_e^{(2)}[x_1(t) - x_2(t)] + k_e^{(1)}x_1(t) - k_h^{(2)}z_2(t) + k_h^{(1)}z_1(t) = -m_1\ddot{x}_g(t) \qquad (3.1)$$

$$m_2\ddot{x}_2(t) + c_2\dot{x}_2(t) + k_h^{(2)}[x_2(t) - x_1(t)] + k_h^{(2)}z_2(t) = -m_2\ddot{x}_g(t) \qquad (3.2)$$

where subscript *1* denotes the bottom storey and subscript *2* the top storey.

The unknowns in Equation (3.1) are: $k_e^{(1)}$, $k_h^{(1)}$, k_e^2, k_h^2, $z_1(t)$ and $z_2(t)$.

Adding Equations (3.1) and (3.2) eliminates the unknowns k_e^2, k_h^2 and $z_2(t)$, and yields a single equation of the form:

$$m_1\ddot{x}_1(t) + m_2\ddot{x}_2(t) + c_1\dot{x}_1(t) + c_2\dot{x}_2(t) + (k_e^{(1)} + k_h^{(1)})x_1(t) + k_h^{(1)}[z_1(t) - x_1(t)] = -(m_1 + m_2)\ddot{x}_g(t) \qquad (3.3)$$

The estimated accelerations $a_1(t)$ and $a_2(t)$, velocities $v_1(t)$ and $v_2(t)$ and displacements $d_1(t)$ and $d_2(t)$ are then calculated using the procedure described in Chapter 2.2.1 approximate $\ddot{x}_1(t), \ddot{x}_2(t)$, $\dot{x}_1(t), \dot{x}_2(t)$ and $x_1(t), x_2(t)$ respectively.

Thus, Equation (3.3) is rewritten:

$$k_p^{(1)} d_1(t) + \Delta Z_1(t) = -(m_1 + m_2)\ddot{x}_g(t) - m_1 a_1(t) - m_2 a_2(t) - c_1 v_1(t) - c_2 v_2(t) \tag{3.4}$$

where

$$k_p^{(1)} = k_e^{(1)} + k_h^{(1)} \tag{3.5}$$

$$\Delta Z^{(1)}(t) = k_h^{(1)}[z_1(t) - x_1(t)] \tag{3.6}$$

Equation (3.4) is now in the same form as Equation (2.32) where all the terms on the right hand side are either measured or estimated. Thus, the same procedure of Equations (2.35)-(2.42) can be applied to find the parameters $\Delta Z^{(1)}(t)$ and $k_p^{(1)}(t)$ in a piecewise sense. In this case the matrix $\mathbf{A_i}$ and column vector $\mathbf{b_i}$ for Equation (2.40) are of the form:

$$\mathbf{A_i} = \begin{bmatrix} d(t_{i1}) & 1 & 0 & 0 \\ d(t_{i2}) & 1 & 0 & 0 \\ d(t_{i3}) & 1 & 0 & 0 \\ d(t_{i4}) & 0 & 1 & 0 \\ d(t_{i5}) & 0 & 1 & 0 \\ d(t_{i6}) & 0 & 1 & 0 \\ d(t_{i7}) & 0 & 0 & 1 \\ d(t_{i8}) & 0 & 0 & 1 \\ d(t_{i9}) & 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

$$\mathbf{b_i} = \begin{Bmatrix} -(m_1+m_2)\ddot{x}_g(t_{i1}) - m_1a_1(t_{i1}) - m_2a_2(t_{i1}) - c_1v_1(t_{i1}) - c_2v_2(t_{i1}) \\ -(m_1+m_2)\ddot{x}_g(t_{i2}) - m_1a_1(t_{i2}) - m_2a_2(t_{i2}) - c_1v_1(t_{i2}) - c_2v_2(t_{i2}) \\ -(m_1+m_2)\ddot{x}_g(t_{i3}) - m_1a_1(t_{i3}) - m_2a_2(t_{i3}) - c_1v_1(t_{i3}) - c_2v_2(t_{i3}) \\ -(m_1+m_2)\ddot{x}_g(t_{i4}) - m_1a_1(t_{i4}) - m_2a_2(t_{i4}) - c_1v_1(t_{i4}) - c_2v_2(t_{i4}) \\ -(m_1+m_2)\ddot{x}_g(t_{i5}) - m_1a_1(t_{i5}) - m_2a_2(t_{i5}) - c_1v_1(t_{i5}) - c_2v_2(t_{i5}) \\ -(m_1+m_2)\ddot{x}_g(t_{i6}) - m_1a_1(t_{i6}) - m_2a_2(t_{i6}) - c_1v_1(t_{i6}) - c_2v_2(t_{i6}) \\ -(m_1+m_2)\ddot{x}_g(t_{i7}) - m_1a_1(t_{i7}) - m_2a_2(t_{i7}) - c_1v_1(t_{i7}) - c_2v_2(t_{i7}) \\ -(m_1+m_2)\ddot{x}_g(t_{i8}) - m_1a_1(t_{i8}) - m_2a_2(t_{i8}) - c_1v_1(t_{i8}) - c_2v_2(t_{i8}) \\ -(m_1+m_2)\ddot{x}_g(t_{i9}) - m_1a_1(t_{i9}) - m_2a_2(t_{i9}) - c_1v_1(t_{i9}) - c_2v_2(t_{i9}) \end{Bmatrix} \tag{3.8}$$

and the vector of unknown parameters is denoted

$$\alpha^{(1)} = \left\{ \underline{\alpha}_1^{(1)}, .., \underline{\alpha}_N^{(1)} \right\}^T \tag{3.9}$$

Least squares solution of Equation (2.40) with $\mathbf{A_i}$ and $\mathbf{b_i}$ given by Equation (3.7) and (3.8) will determine $\underline{\alpha}^{(1)}$ of Equation (3.9). The parameters $\left\{ k_{e,i}^{(1)}, i = 1,..,N \right\}$, $\left\{ k_{h,i}^{(1)}, i = 1,..,N \right\}$ and $\left\{ D_j^{(1)}, j = 1,..,M \right\}$ are then found from Equations (2.51), (2.52) and (2.53) respectively.

To find the stiffness and permanent displacement of storey 2, Equation (3.2) is rewritten:

$$k_p^{(2)} r_2(t) + \Delta Z^{(2)}(t) = -m_2 \ddot{x}_g(t) - m_2 a_2(t) - c_2 v_2(t) \tag{3.10}$$

where

$$k_p^{(2)} = k_e^{(2)} + k_h^{(2)} \tag{3.11}$$

$$\Delta Z^{(2)}(t) = k_h^{(2)}[z_2(t) - r_2(t)] \tag{3.12}$$

Equation (3.10) is in the same form as Equation (2.39), where all the terms on the right hand side are either measured or estimated. Thus, the same procedure of Equations (2.42)-(2.50) can be used again to find the parameters $k_p^{(2)}$ and $\Delta Z^{(2)}(t)$ in a piecewise sense. In this case however the matrix $\mathbf{A_i}$ of Equation (2.48) is of the form:

$$\mathbf{A_i} = \begin{bmatrix} r_2(t_{i1}) & 1 & 0 & 0 \\ r_2(t_{i2}) & 1 & 0 & 0 \\ r_2(t_{i3}) & 1 & 0 & 0 \\ r_2(t_{i4}) & 0 & 1 & 0 \\ r_2(t_{i5}) & 0 & 1 & 0 \\ r_2(t_{i6}) & 0 & 1 & 0 \\ r_2(t_{i7}) & 0 & 0 & 1 \\ r_2(t_{i8}) & 0 & 0 & 1 \\ r_2(t_{i9}) & 0 & 0 & 1 \end{bmatrix} \tag{3.13}$$

and the vector $\mathbf{b_i}$ is of the form:

$$\mathbf{b_i} = \begin{Bmatrix} -m_2\ddot{x}_g(t_{i1}) - m_2 a_2(t_{i1}) - c_2 v_2(t_{i1}) \\ -m_2\ddot{x}_g(t_{i2}) - m_2 a_2(t_{i2}) - c_2 v_2(t_{i2}) \\ -m_2\ddot{x}_g(t_{i3}) - m_2 a_2(t_{i3}) - c_2 v_2(t_{i3}) \\ -m_2\ddot{x}_g(t_{i4}) - m_2 a_2(t_{i4}) - c_2 v_2(t_{i4}) \\ -m_2\ddot{x}_g(t_{i5}) - m_2 a_2(t_{i5}) - c_2 v_2(t_{i5}) \\ -m_2\ddot{x}_g(t_{i6}) - m_2 a_2(t_{i6}) - c_2 v_2(t_{i6}) \\ -m_2\ddot{x}_g(t_{i7}) - m_2 a_2(t_{i7}) - c_2 v_2(t_{i7}) \\ -m_2\ddot{x}_g(t_{i8}) - m_2 a_2(t_{i8}) - c_2 v_2(t_{i8}) \\ -m_2\ddot{x}_g(t_{i9}) - m_2 a_2(t_{i9}) - c_2 v_2(t_{i9}) \end{Bmatrix} \tag{3.14}$$

The vector of unknown parameters $\underline{\alpha}^{(2)}$ for this example is defined by replacing (1) by (2) in Equation (3.9).

The least squares solution of Equation (2.48) with $\mathbf{A_i}$ and $\mathbf{b_i}$ given by Equations (3.13) and (3.14) determines $\underline{\alpha}^{(2)}$. The parameters $\left\{k_{e,i}^{(2)}, i = 1,..,N\right\}$, $\left\{k_{h,i}^{(2)}, i = 1,..,N\right\}$ and $\left\{D_j^{(2)}, j = 1,..,M\right\}$ are then found from Equations (2.51), (2.52) and (2.53) respectively.

3.2 N-DOF System Identification

The procedure of Equations (3.1)-(3.14) can easily be generalised for an $\overline{N}$-DOF shear building model.

First, the constituent equations of the matrix Equation (2.13), which describe the motion of an $\overline{N}$-DOF shear building with Bouc-Wen hysteresis, are written as:

$$m_1\ddot{x}_1(t)+\sum_{b=1}^{\overline{N}}c_{1,b}\dot{x}_b(t)+k_e^{(2)}[x_1(t)-x_2(t)]+k_e^{(1)}x_1(t)-k_h^{(2)}z_2(t)+k_h^{(1)}z_1(t)=-m_1\ddot{x}_g(t) \tag{3.15}$$

$$\begin{aligned} &m_w\ddot{x}_w(t)+\sum_{b=1}^{\overline{N}}c_{w,b}\dot{x}_b(t)+k_e^{(w)}[x_w(t)-x_{w-1}(t)]+k_e^{(w+1)}[x_w(t)-x_{w+1}(t)] \\ &+k_h^{(w)}z_w(t)-k_h^{(w+1)}z_{w+1}(t)=-m_w\ddot{x}_g(t) \quad , \quad w=2,..,\overline{N}-1 \end{aligned} \tag{3.16}$$

$$m_{\overline{N}}\ddot{x}_{\overline{N}}(t)+\sum_{b=1}^{\overline{N}}c_{\overline{N},b}\dot{x}_b(t)+k_e^{(\overline{N})}[x_{\overline{N}}(t)-x_{\overline{N}-1}(t)]+k_h^{(\overline{N})}z_{\overline{N}}(t)=-m_{\overline{N}}\ddot{x}_g(t) \tag{3.17}$$

Note that superscripts '(w)' are used on the parameters k_e and k_h and subscripts 'w' are used on the rest of the parameters to denote the storey number.

Therefore, Equation (3.15) defines the motion of the bottom storey, *1*, Equation (3.16) defines the motion of any intermediate storey, $w=2,..,\overline{N}-1,$ and Equation (3.17) defines the motion of the top storey. Note that for some damping models many of the damping coefficient terms $c_{w,b}$ may be zero however the damping force here is calculated using the entire matrix for completeness. In practice, zeros in a known matrix structure could be ignored to save computation.

Next, a revised equation of motion for the bottom storey, *1*, can be written by the summation of all the Equations (3.15)-(3.17). The summation of all the left hand sides of Equations (3.15)-(3.17) gives:

$$\sum_{k=1}^{\overline{N}} m_k \ddot{x}_k(t) + \sum_{k=1}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} \dot{x}_b(t) + \sum_{k=1}^{\overline{N}} k_e^{(k)} [x_k(t) - x_{k-1}(t)] + \sum_{k=1}^{\overline{N}} k_h^{(k)} z_k(t) - \sum_{k=2}^{\overline{N}} k_e^{(k)} [x_k(t) - x_{k-1}(t)] - \sum_{k=2}^{\overline{N}} k_h^{(k)} z_k(t) \tag{3.18}$$

where the restoring force terms are:

$$\sum_{k=1}^{\overline{N}} k_e^{(k)} [x_k(t) - x_{k-1}(t)] + \sum_{k=1}^{\overline{N}} k_h^{(k)} z_k(t) \tag{3.19}$$

and

$$- \sum_{k=2}^{\overline{N}} k_e^{(k)} [x_k(t) - x_{k-1}(t)] - \sum_{k=2}^{\overline{N}} k_h^{(k)} z_k(t) \tag{3.20}$$

The terms shown in (3.19) and (3.20) form a telescoping sum, canceling out from $k = 2,..,\overline{N}$ and leaving only $(k_e^{(1)})[x_2(t) - x_1(t)] + k_h^{(1)}[z_1(t)]$. Thus, the revised equation of motion for the bottom storey, *1*, is written as follows:

$$\sum_{k=1}^{\overline{N}} m_k \ddot{x}_k(t) + \sum_{k=1}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} \dot{x}_b(t) + (k_e^{(1)} + k_h^{(1)}) r_1(t) + k_h^{(1)}[z_1(t) - r_1(t)] = - \sum_{k=1}^{\overline{N}} m_k \ddot{x}_g(t) \tag{3.21}$$

and similarly for the storey w, $w = 2,..,\overline{N} - 1$:

$$\sum_{k=w}^{\overline{N}} m_k \ddot{x}_k(t) + \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} \dot{x}_b(t) + (k_e^{(w)} + k_h^{(w)}) r_w(t) + k_h^{(w)}[z_w(t) - r_w(t)] = - \sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t) \tag{3.22}$$

The top storey $\overline{N}$ is kept in the form given by Equation (3.17). The summation process eliminates the unknowns k_{e2}, k_{h2} and $z_2(t)$ from Equation (3.15) and for each w in Equation (3.16) eliminates the unknowns $k_e^{(w+1)}, k_h^{(w+1)}$ and $z_{w+1}(t)$. Also, the only unknowns in Equation (3.17) are $k_e^{(\overline{N})}, k_h^{(\overline{N})}$ and $z_{\overline{N}}(t)$. This process effectively decouples the Equations (3.15)-(3.17), so that all the unknowns are specific to each storey. The estimated displacement $d_w(t)$, velocity $v_w(t)$ and acceleration $a_w(t)$ are then calculated using Equations (2.35)-(2.37) to approximate $\ddot{x}_w(t)$, $\dot{x}_w(t)$ and $x_w(t)$ respectively for Equations (3.17), (3.21) and (3.22). Equations (3.17), (3.21) and (3.22) can then be rewritten as one equation for any storey *w*, $w = 1,..,\overline{N}$:

$$k_p^{(w)} r_w(t) + \Delta Z^{(w)}(t) = -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t) - \sum_{k=w}^{\overline{N}} m_k a_k(t) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t) \tag{3.23}$$

where:

$$k_p^{(w)} = k_e^{(w)} + k_h^{(w)} \quad , \quad w = 1,..,\overline{N} \tag{3.24}$$

$$\Delta Z^{(w)}(t) = k_h^{(w)} [z_w(t) - r_w(t)] \quad , \quad w = 1,..,\overline{N} \tag{3.25}$$

Equation (3.23) is in the same form as Equation (2.39), thus the same procedure of Equations (2.42)-(2.50) in Section 2.3 can be applied to find the parameters $\Delta Z^{(w)}(t)$ and $k_p^{(w)}$ in a piecewise sense. In this case for each $w = 1,..,\overline{N}$ the matrix $\mathbf{A_{w,i}}$ for Equation (2.48) is of the form:

$$\mathbf{A_{w,i}} = \begin{bmatrix} r_w(t_{i1}) & 1 & 0 & 0 \\ r_w(t_{i2}) & 1 & 0 & 0 \\ r_w(t_{i3}) & 1 & 0 & 0 \\ r_w(t_{i4}) & 0 & 1 & 0 \\ r_w(t_{i5}) & 0 & 1 & 0 \\ r_w(t_{i6}) & 0 & 1 & 0 \\ r_w(t_{i7}) & 0 & 0 & 1 \\ r_w(t_{i8}) & 0 & 0 & 1 \\ r_w(t_{i9}) & 0 & 0 & 1 \end{bmatrix} \tag{3.26}$$

and the right hand side vector $\mathbf{b_{w,i}}$ is of the form:

$$\mathbf{b_{w,i}} = \begin{Bmatrix} -\sum_{k=q}^{\overline{N}} m_k \ddot{x}_g(t_{i1}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i1}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i1}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{i2}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i2}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i2}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{i3}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i3}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i3}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{i4}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i4}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i4}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{i5}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i5}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i5}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{i6}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i6}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i6}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{i7}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i7}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i7}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{8i}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i8}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i8}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{9i}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i9}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i9}) \end{Bmatrix} \tag{3.27}$$

The vector of unknowns for each $w = 1,..,\overline{N}$ is defined:

$$\underline{\alpha}^{(w)} = \left\{\underline{\alpha}_1^{(w)},..,\underline{\alpha}_N^{(w)}\right\}^T \tag{3.28}$$

where

$$\underline{\alpha}_i^{(w)} = \left\{k_p^{(w)}, \Delta Z_{3i-2}^{(w)}, \Delta Z_{3i-1}^{(w)}, \Delta Z_{3i}^{(w)}\right\}^T \quad , \quad i = 1,..,N \tag{3.29}$$

The matrix equations for each $w = 1,..,\overline{N}$ are therefore defined:

$$\mathbf{A_{w,i}}\,\underline{\alpha}_i^{(w)} = \mathbf{b_{w,i}} \quad , \quad i = 1,..,N \tag{3.30}$$

Least squares solution of the matrix Equation (3.26) determines the vector $\underline{\alpha}^{(w)}$ of unknowns for each $w = 1,..,\overline{N}$. The parameters $\left\{k_{e,i}^{(w)}\,, i = 1,..,N\right\}$, $\left\{k_{w,i}^{(w)}\,, i = 1,..,N\right\}$and $\left\{D_j^{(w)}\,, j = 1,..,M\right\}$ are then found from Equations (2.51), (2.52) and (2.53) respectively.

Note that $D_j^{(w)}$ is the permanent displacement of storey w relative to storey $w-1$, the absolute permanent displacement, $DA_j^{(w)}$ of storey w can be calculated as follows:

$$DA_j^{(w)} = \sum_{k=1}^{w} D_j^{(k)} \tag{3.31}$$

This process is performed for each storey w and each time interval $(i-1)\Delta t_{kp} \le t \le i\Delta t_{kp}$ to identify the stiffness and permanent displacement of each storey over the entire record. A graphical depiction of the process is shown in Figure 3.1.

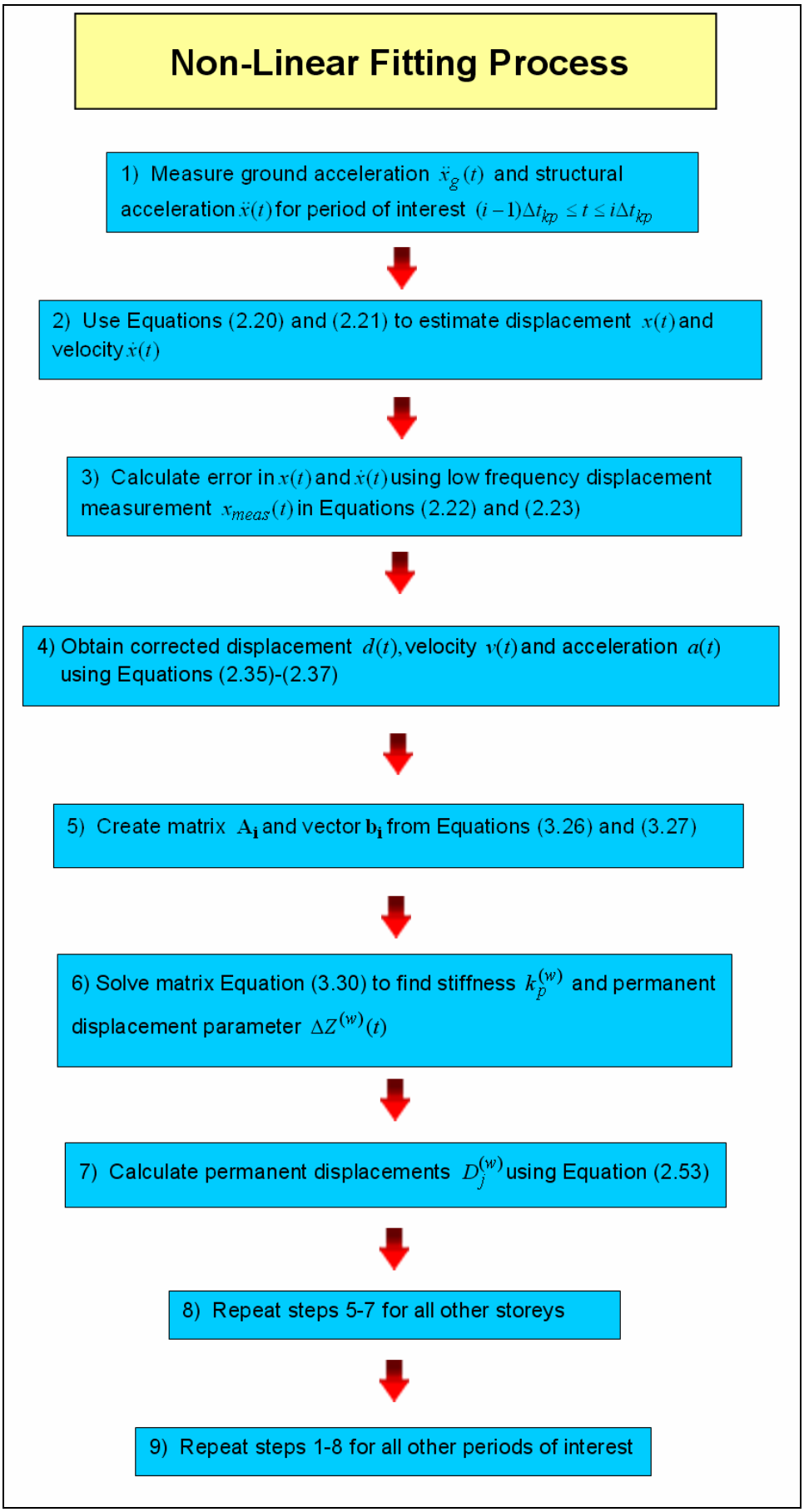

Figure 3.1: Flow chart depicting fitting process for multi-degree of freedom shear building

3.3 Linear Model Identification for Non-Linear Systems

For hybrid rocking structures it is useful to identify a time-varying linear stiffness without a permanent displacement since the method described in Chapter 2.1 can misinterpret rocking behaviour as an oscillating permanent displacement. Fitting a time-varying linear stiffness can capture the reduction in stiffness exhibited by a hybrid structure when operating in its' rocking regime. The linear stiffness is identified using a similar method to that outlined in Chapter 2.1 however the structural velocity and displacement are estimated using the procedure outlined in Chapter 2.2.1, thus eliminating the need for the error constants δ and σ seen in Equations (2.3) and (2.4).

First, Equation (2.1) is firstly written as its constituent equations:

$$m_1\ddot{x}_1(t)+\sum_{b=1}^{\overline{N}}c_{1,b}\dot{x}_b(t)+k_l^{(2)}[x_1(t)-x_2(t)]+k_l^{(1)}x_1(t)=-m_1\ddot{x}_g(t) \tag{3.32}$$

$$\begin{aligned} &m_w\ddot{x}_w(t)+\sum_{b=1}^{\overline{N}}c_{w,b}\dot{x}_b(t)+k_l^{(w)}[x_w(t)-x_{w-1}(t)] \\ &+k_l^{(w+1)}[x_w(t)-x_{w+1}(t)]=-m_w\ddot{x}_g(t) \quad , \quad w=2,..,\overline{N}-1 \end{aligned} \tag{3.33}$$

$$m_{\overline{N}}\ddot{x}_{\overline{N}}(t)+\sum_{b=1}^{\overline{N}}c_{\overline{N},b}\dot{x}_b(t)+k_l^{(\overline{N})}[x_{\overline{N}}(t)-x_{\overline{N}-1}(t)]=-m_{\overline{N}}\ddot{x}_g(t) \tag{3.34}$$

where Equation (3.32) defines the motion of the bottom storey, *1*, Equation (3.33) defines the motion of any intermediate storey, $w = 2,..,\overline{N}-1,$ and Equation (3.34) defines the motion of the top storey $\overline{N}$. The parameter k_l is the linear stiffness.

In a similar manner to the procedure of Equations (3.18)-(3.20), the unknown $k_l^{(2)}$ and can be eliminated from Equation (3.32) by summation of all the Equations (3.32)-(3.34). Thus a revised equation of motion for the bottom storey, *1*, can be written:

$$\sum_{k=1}^{\overline{N}} m_k \ddot{x}_k(t) + \sum_{k=1}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} \dot{x}_b(t) + k_l^{(1)} r_1(t) = -\sum_{k=1}^{\overline{N}} m_k \ddot{x}_g(t) \tag{3.35}$$

Similarly, for any storey *w*, $w = 2,..,\overline{N}-1$ the unknown $k_l^{(w+1)}$ can be eliminated by telescoping summation, yielding:

$$\sum_{k=w}^{\overline{N}} m_k \ddot{x}_k(t) + \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} \dot{x}_b(t) + k_l^{(w)} r_w(t) = -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t) \tag{3.36}$$

The top storey $\overline{N}$ is kept in the form given by Equation (3.30).

Now the only unknown in Equation (3.35) is $k_l^{(1)}$, while the only unknown in Equation (3.36) is $k_l^{(w)}$. Finally, the only unknown in Equation (3.34) is $k_l^{(\overline{N})}$. Equations (3.34), (3.35) and (3.36) can therefore be used to find the linear stiffness of any storey *w*, $w = 1,..,\overline{N}$.

The estimated displacement $d_w(t)$, velocity $v_w(t)$ and acceleration $a_w(t)$ are then calculated using Equations (2.35)-(2.42) to approximate $\ddot{x}_w(t)$, $\dot{x}_w(t)$ and $x_w(t)$ respectively. Upon substitution of the estimated properties, Equations (3.34), (3.35) and (3.36) are rewritten as one equation for any storey w, $w = 1,..,\overline{N}$:

$$k_l^{(w)} r_w(t) = -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t) - \sum_{k=w}^{\overline{N}} m_k a_k(t) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t) \tag{3.37}$$

Now, assuming that $k_l^{(w)}$ is a piecewise constant function over N fixed time intervals of length Δt, $k_l^{(w)}$ can be defined:

$$k_l^{(w)} = k_{l,i}^{(w)} \quad , \quad (i-1)\Delta t \le t \le i\Delta t \quad , \quad i = 1,..,N \tag{3.38}$$

Equation (3.37) can then be reformulated:

$$k_{l,i}^{(w)} r_w(t) = -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t) - \sum_{k=w}^{\overline{N}} m_k a_k(t) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t) \quad , \quad (i-1)\Delta t \le t \le i\Delta t \quad , \quad i = 1,..,N \tag{3.39}$$

Each $k_{l,i}^{(w)}$, $i = 1,..,N$ can now be identified by constructing a system of linear equations. For example, if five values of t are chosen in each time interval $t_i \le t \le t_i + \Delta t$, $i = 1,..,N$, there will be a total of $5N$ linear equations. Each linear equation will be in the form of Equation (3.39) and this leads to N unknowns k_i, $i = 1,..,N$.

These linear equations can be represented as N matrix equations each of the form:

$$\mathbf{a_{w,i}} k_{l,i}^{(w)} = \mathbf{b_{w,i}} \quad , \quad i = 1,..,N \tag{3.40}$$

where $\mathbf{a_{w,i}}$ is a 5x1 vector for this example of the form:

$$\mathbf{a_{w,i}} = \begin{Bmatrix} r_w(t_{i1}) \\ r_w(t_{i2}) \\ r_w(t_{i3}) \\ r_w(t_{i4}) \\ r_w(t_{i5}) \end{Bmatrix} \tag{3.41}$$

and $\mathbf{b_{w.i}}$ is a 5x1 vector for this example of the form:

$$\mathbf{b_{w,i}} = \begin{Bmatrix} -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{i1}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i1}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i1}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{i2}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i2}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i2}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{i3}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i3}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i3}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{i4}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i4}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i4}) \\ -\sum_{k=w}^{\overline{N}} m_k \ddot{x}_g(t_{i5}) - \sum_{k=w}^{\overline{N}} m_k a_k(t_{i5}) - \sum_{k=w}^{\overline{N}} \sum_{b=1}^{\overline{N}} c_{k,b} v_b(t_{i5}) \end{Bmatrix} \tag{3.42}$$

The linear system of Equation (3.40) is over determined. Thus the required unknowns $k_{l,i}^{(w)}$, $i = 1,..,N$ can be found by linear least squares.

This process is performed for each storey w and each time interval $(i-1)\Delta t_{kp} \le t \le i\Delta t_{kp}$ to identify the stiffness of each storey over the entire record. Figure 3.2 shows the flow diagram for the linear stiffness fitting process.

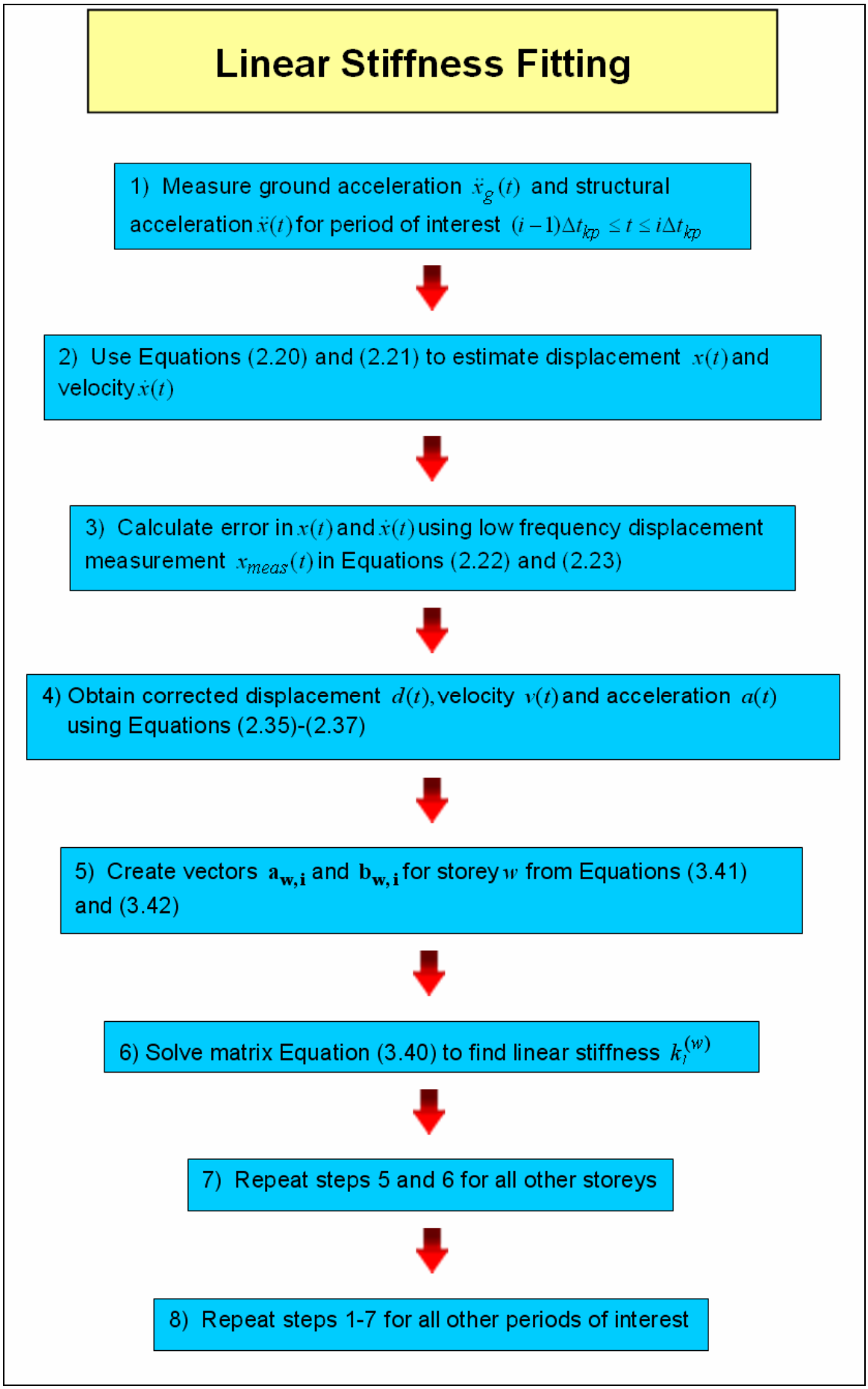

Figure 3.2: Flow chart depicting fitting process for linear stiffness method

CHAPTER 4 - TEST STRUCTURES

Three different test structures were used to test the algorithm as well as to highlight the impact of baseline model selection on the identified results. First, a four-storey, four degree of freedom structural model is used in simulation. Second, experimental data for a similar structure is employed. Finally, simulated data from a two-storey, six degree of freedom rocking structure us used. All cases employ the El Centro ground motion although the method is completely general.

4.1 Simulation Structure

The algorithm was first tested using simulated data in order to provide proof of concept and quantify the effects of noise in measured data on the accuracy of the fitted parameters k_p and $\Delta Z(t)$. MATLAB was used to simulate the response of the structure shown in Figure 4.1. Each storey had a pre-yield stiffness of 1610 N/m and mass of 1kg, resulting in an undamped fundamental natural period of 0.45s for the structure. This period was chosen to closely match the natural period of the laboratory structure, which has been calculated at 0.47s (Kao, 1998). A diagonal mass matrix was used in simulation.

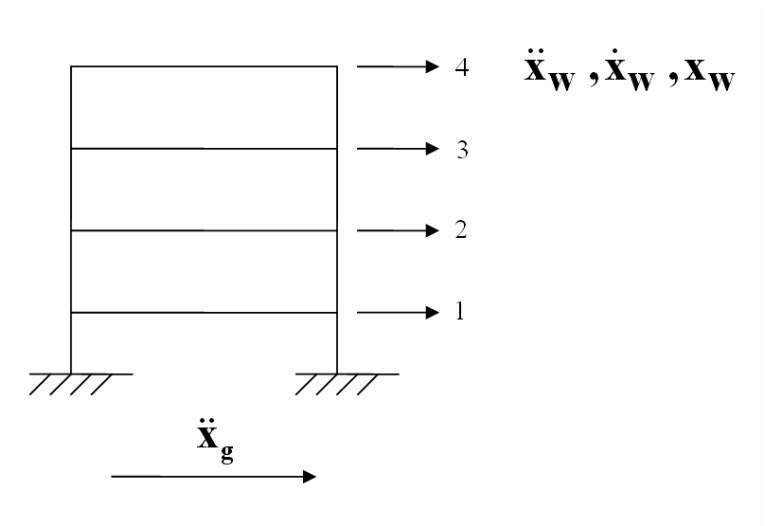

Figure 4.1: Simulated shear building arrangement

The laboratory structure has been calculated to have between 0.9% and 2.9% damping, depending on the magnitude of response (Kao, 1998). Thus, the following damping matrix was constructed for the simulated structure using the equivalent viscous damping approach assuming 2% damping in each mode:

$$C = \begin{bmatrix} 2.18 & -0.63 & -0.11 & -0.05 \\ -0.63 & 2.07 & -0.68 & 0.17 \\ -0.11 & -0.68 & 2.01 & -0.79 \\ -0.05 & -0.17 & -0.79 & 1.39 \end{bmatrix} \tag{4.1}$$

Each storey was given a yield displacement, Y, of 0.04m, shaping parameter, n, of 2 and a bi-linear factor, $\alpha = 0.1$. These parameters were chosen to provide realistic non-linear structural behaviour for the fitting algorithm to identify.

The simulated structure was subjected to the El Centro earthquake record, data was recorded at 1 kHz with noise applied after simulation calculated using the following equations:

$$\ddot{x}_{noise}(t) = \ddot{x}_{true}(t) + pf_a\sigma \text{ , } 0 < t < T \tag{4.2}$$

$$x_{noise}(t) = x_{true}(t) + pf_d\sigma \text{ , } 0 < t < T \tag{4.3}$$

where $x_{noise}(t)$ is the noisy displacement, $x_{true}(t)$ is the 'true' displacement from simulation, $\ddot{x}_{true}(t)$ is the 'true' acceleration, T is the total time span of the earthquake, p is the percentage of noise to be applied, f_a is the mean absolute value of $\ddot{x}(t)$, $0 < t < T$, f_d is the mean absolute value of $x(t)$, $0 < t < T$, and σ is a uniformly distributed random variable lying between -1 and 1. Note this uniform distribution is a conservative choice where outliers are more likely to occur.

4.2 Four Storey Non-Linear Steel Frame Structure

The shear building structure used to test the algorithm was a one-fifth scale four-storey steel frame structure originally designed to demonstrate the dynamic response of multiple degree of freedom systems (Kao, 1998). The structure, shown in Figure 4.2, features a two-bay frame in the longitudinal direction with one long span and one short span. The frame has overall dimensions of 2.1m x 2.1m with a short span length of 0.7m and a long span length of 1.4m. The first storey has a height of 0.6m while the second to fourth storeys have a height of 0.5m. Replaceable plastic hinges are located at beam column joints and at the mid-points of the long span beams.

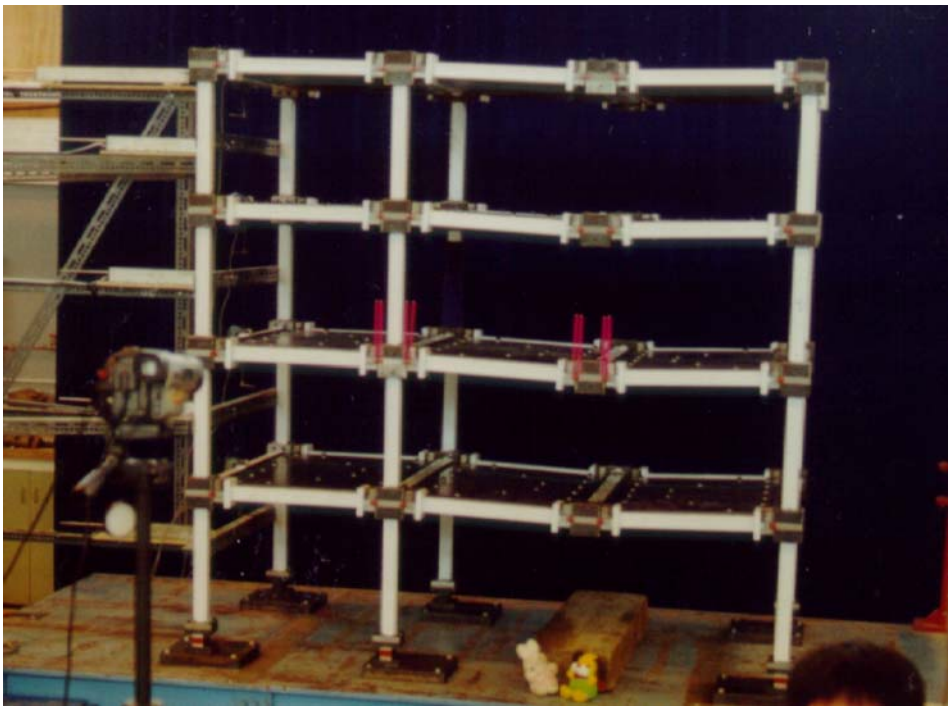

Figure 4.2: Four storey non-linear steel frame structure

A realistic computational model of the structure was created (Kao, 1998) for use with the time-history analysis package RUAUMOKO (Carr, 2004). This computational model was used to estimate simplified four degree of freedom mass and damping matrices for a baseline model to identify the structure, as well as to estimate the stiffness of each storey for comparison to fitted results by pushover analysis.

The following four degree of freedom diagonal mass matrix was calculated from the RUAUMOKO model:

$$\mathbf{M} = \begin{bmatrix} 464 & 0 & 0 & 0 \\ 0 & 464 & 0 & 0 \\ 0 & 0 & 464 & 0 \\ 0 & 0 & 0 & 461 \end{bmatrix} \tag{4.4}$$

Table 4.1 shows the storey stiffnesses that were calculated by pushover analysis from the RUAUMOKO model.

Table 4.1: Pushover analysis estimated storey stiffnesses

Storey	Stiffness (kN/m)
1	672
2	640
3	601
4	544

The damping matrix used in fitting the frame structure was estimated to provide 2% damping in each structural mode, in keeping with estimates of the structures damping provided by Kao (1998).

$$\mathbf{C_S} = 1\times10^{3}\begin{bmatrix} 1.01 & -0.29 & -0.05 & -0.03 \\ -0.29 & 0.96 & -0.32 & -0.08 \\ -0.05 & -0.32 & 0.93 & -0.37 \\ -0.03 & -0.08 & -0.37 & 0.64 \end{bmatrix} \tag{4.5}$$

The structure was subjected to the El Centro ground motion record and structural acceleration and displacement data was measured at 1kHz using a dSpace^{TM} system.

4.3 Two Storey Hybrid Rocking Structure

The hybrid rocking structure used to test the algorithm is shown in Figure 4.3. This structure is a quarter scale, two storey, one bay post-tensioned, pre-cast reinforced concrete frame structure (Murahidy et al, 2004). Previous experimental work (Priestley and MacRae (1996); Cheok and Lew (1993); Stone et al (1995); Nakaki et al (1999) and Priestley et al (1999)), has shown that pre-cast concrete frames constructed from post-tensioned segmental elements perform notably better than conventional frames, when subjected to lateral loading. Structural deformations are accommodated by gap openings at the pre-cast component interfaces. Similar strength and inter-storey drift capacities are achievable, but damage to the structural elements is significantly

reduced and the frames possess a re-centering capacity that results in negligible residual displacements. Such a structure is potentially better described by a linear base-line model rather than the Bouc-Wen based structural system model of Equation (2.13) since there would be very little permanent displacement.

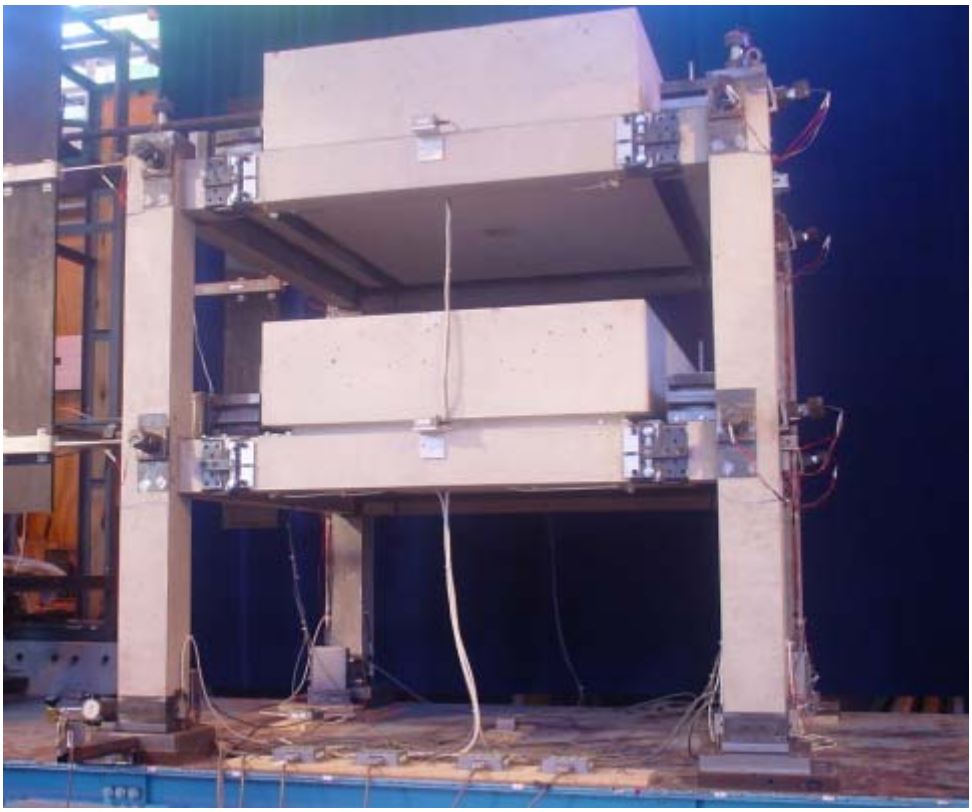

Figure 4.3: Two storey hybrid rocking structure

A fifty degree of freedom finite element model (Spieth et al. 2004, 2004a) was used to create simplified 6 degree of freedom mass and damping matrices by Chase et al (2004b). This simplified model is used as a baseline model to identify the structure and provide stiffness values for comparison to fitted results.

The following two degree of freedom mass and damping matrices were used in identification:

$$\mathbf{M} = \begin{bmatrix} 2400 & 0 \\ 0 & 2400 \end{bmatrix} \tag{4.6}$$

$$\mathbf{C_h} = 1 \times 10^5 \begin{bmatrix} 2.83 & -1.36 \\ -1.36 & 0.83 \end{bmatrix} \tag{4.7}$$

Linear and rocking response stiffnesses of the structure were identified by Chase et al (2004b). These values were used for comparison to the stiffnesses identified in this paper, and are shown in Table 4.2. The structure was subjected to the El Centro ground motion record and structural acceleration and displacement data was measured at 200Hz.

Table 4.2: Estimated pre-yield and rocking stiffnesses

Storey	Pre-Yield Stiffness (kN/m)	Rocking Stiffness (kN/m)
1	18500	76
2	12600	76

PART III

RESULTS

CHAPTER 5 - PROOF OF CONCEPT USING SIMULATED DATA

5.1 Single Degree of Freedom

5.1.1 Bi-Linear Stiffness Identification

To provide initial proof of concept for the integral-based approach and linear least squares optimization, the algorithm described in Chapter 3.1 was tested using simulated data from a single degree of freedom bi-linear elastic system. The structure was subjected to the El Centro excitation and 10% uniformly distributed noise was applied to the 1kHz sampled acceleration measurements. The system had a mass of 1kg with a pre-yield stiffness of 39.58N/m, resulting in an undamped fundamental period of 1s. The bi-linear factor, α, was set to 0.1. Note that in this research, all the mass and damping properties used in fitting to simulated data were assumed to be known.

Figure 5.1 shows a comparison of the true simulated stiffness (39.58N/m) and the identified stiffness for a linear elastic system using the identification interval $\Delta t = 0.25\,\text{s}$. Figure 5.2 shows the same comparison for a bi-linear elastic system with a yield point of 45mm. These results demonstrate that the algorithm accurately identifies linear stiffness as well as the time and magnitude of drops in stiffness that can occur due to elastic non-linearity. Thus, the least squares fitting procedure described in Section 3.1 successfully accounts for error and drift in the integral approximations to structural displacement and velocity.

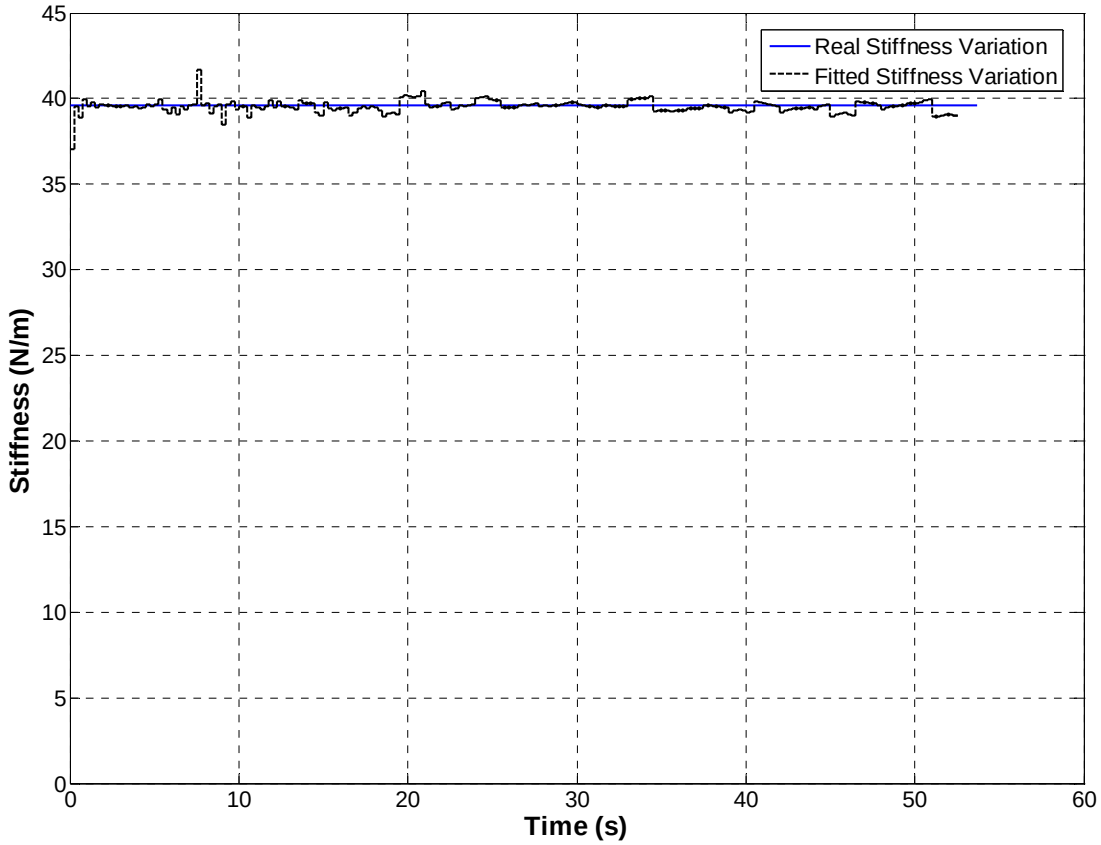

Figure 5.1: Identified stiffness comparison for linear elastic system

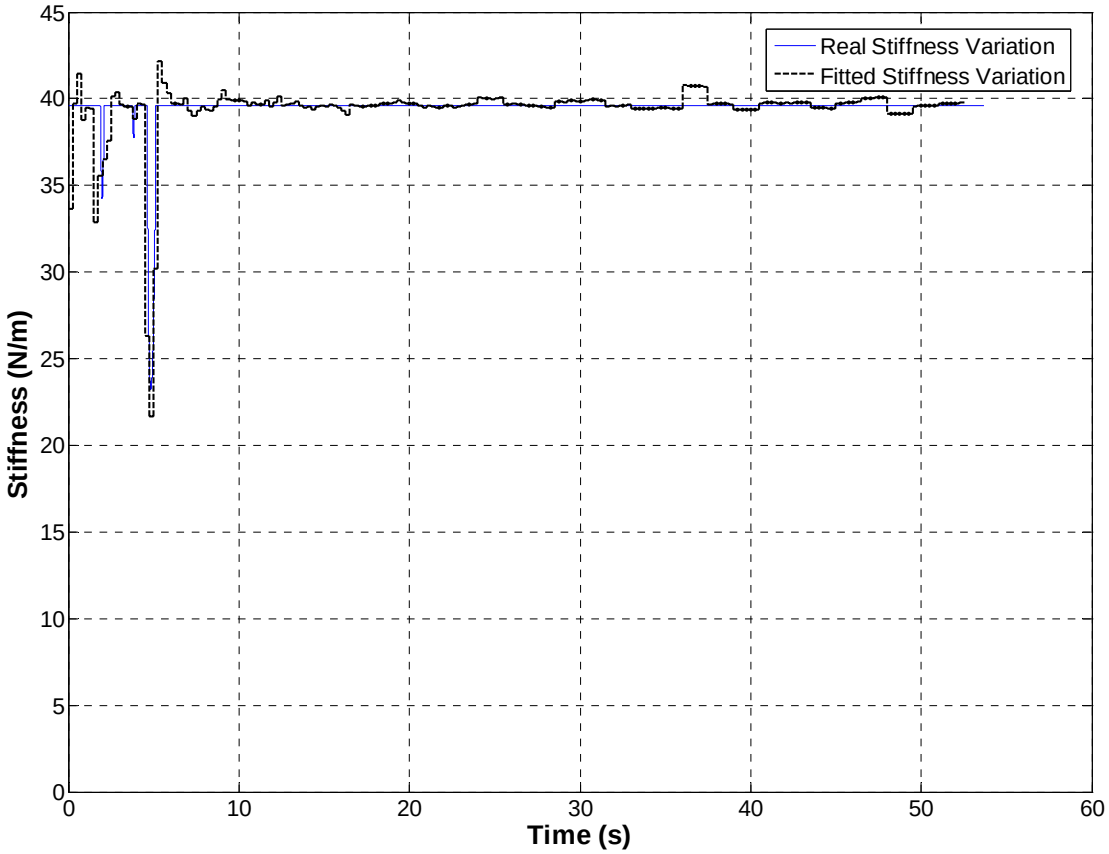

Figure 5.2: Stiffness identification for bi-linear elastic system

5.1.2 Non-Linear Baseline Model Identification

The identification procedure for a single degree of freedom non-linear system described in Chapter 2.3 was verified using simulated data from a structure subjected to the El Centro ground motion record. Acceleration data was sampled at 1kHz and displacement data for drift correction was sampled at 10Hz. All data was subject to 10% uniformly distributed noise.

The structure employed had an initial linear stiffness of 39.58N/m, bi-linear factor of 0.1, yield point of 100mm and a Bouc-Wen hysteresis shaping parameter n=2. Stiffness was fitted over 2s intervals while permanent displacement was fitted over 0.4s intervals. Note that the basic fitting approach of Section X is not used in this instance since the hysteretic behaviour results in permanent deformation of the structure.

Figure 5.3 compares the true stiffness (39.58N/m) to the identified stiffness while Figure 5.4 compares the true and identified permanent displacements. As seen in Figure 5.3, the algorithm accurately identifies the linear stiffness however there is a slight drop in identified stiffness while the structure is yielding rapidly near the start of the record. This drop occurs because the piecewise constant approximation to $\Delta Z(t)$ in Equation (3.6) may not always accurately represent the hysteretic behaviour. This effect is minor and only occurs during highly non-linear periods. Figure 5.4 shows that the algorithm accurately follows major trends in permanent displacement and captures the final residual deformation accurately.

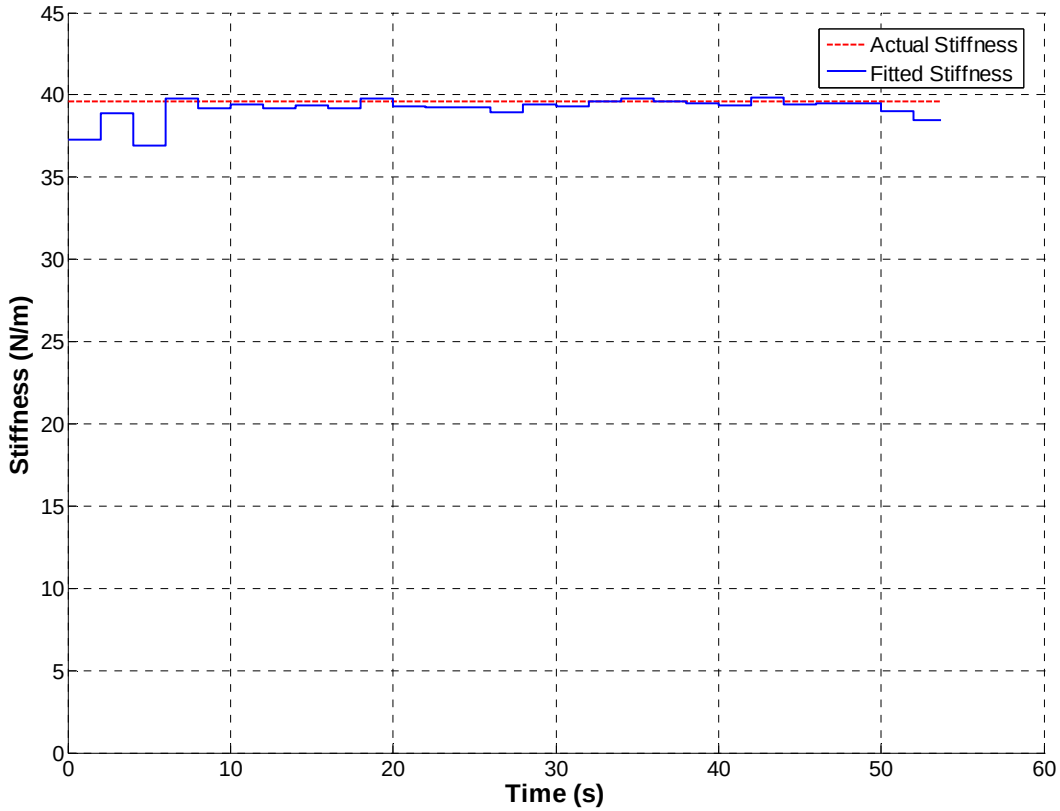

Figure 5.3: Stiffness identification of 1-DOF non-linear system

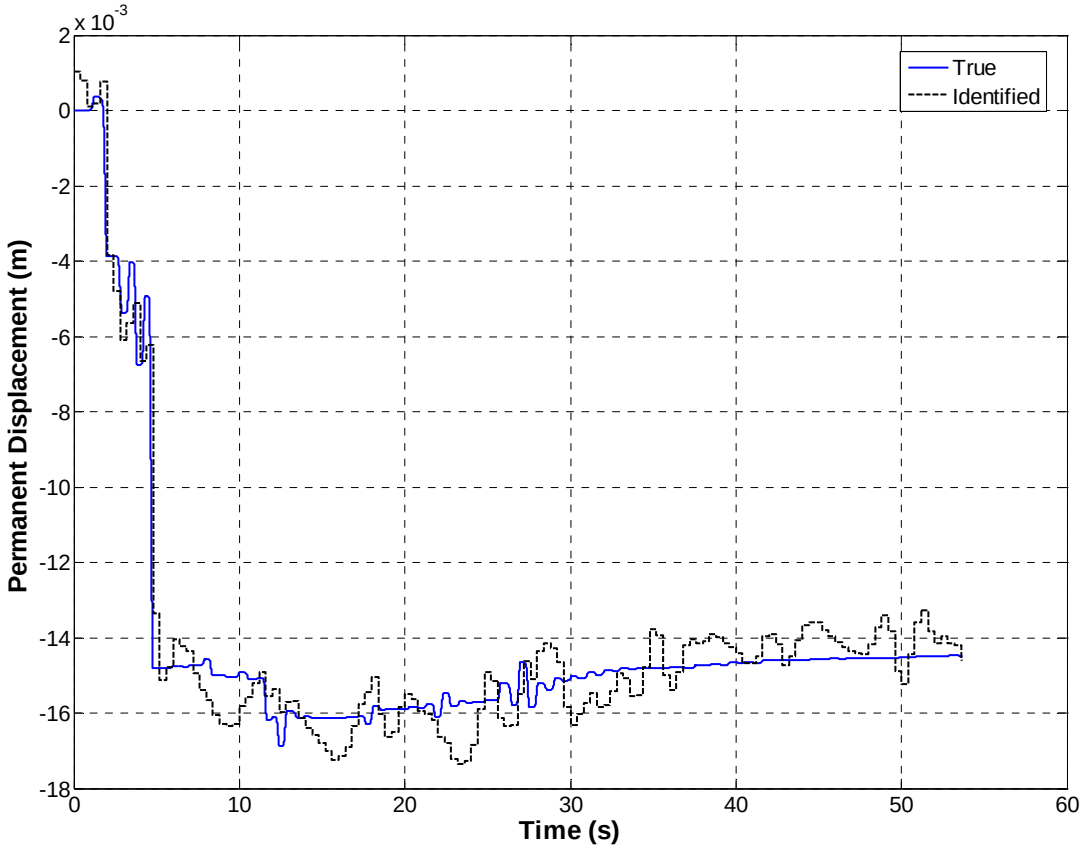

Figure 5.4: Permanent displacement identification of 1-DOF non-linear system

5.2 Two-DOF Model with Bouc-Wen Hysteresis

The algorithms ability to detect damage in multi-storey structures was first tested using two identical Bouc-Wen elements in a shear building arrangement. The elements each had stiffness of 102.64N/m, bi-linear factor of 0.1, yield point of 45mm and shaping parameter n=2, resulting in a fundamental natural period of 1s. The structure was subjected to the El Centro ground excitation and displacement and acceleration data was subject to 10% uniformly distributed noise. In keeping with the single degree of freedom example, acceleration data was sampled at 1kHz while displacement data was sampled at 10Hz.

Figures 5.5-5.8 show the identified stiffness and permanent displacement of both storeys where stiffness is fitted at 2s intervals and permanent displacement 0.4s intervals. The results are similar to those found for the single degree of freedom system. The linear stiffness and final residual deformations are matched accurately for both storeys, with major trends in permanent displacement also correctly identified. However, there is a noticeable decrease in accuracy of the identified stiffness of the top storey towards the end of the record. This loss of accuracy is due to the lack of significant response during this period. Where the measured and estimated properties are attenuating, the parameters identified by the least squares process become more susceptible to noise. During earthquake motion, any structural damage induced will occur during periods of significant response, thus the aforementioned decrease in accuracy will not reduce the algorithms ability to provide effective structural health monitoring.

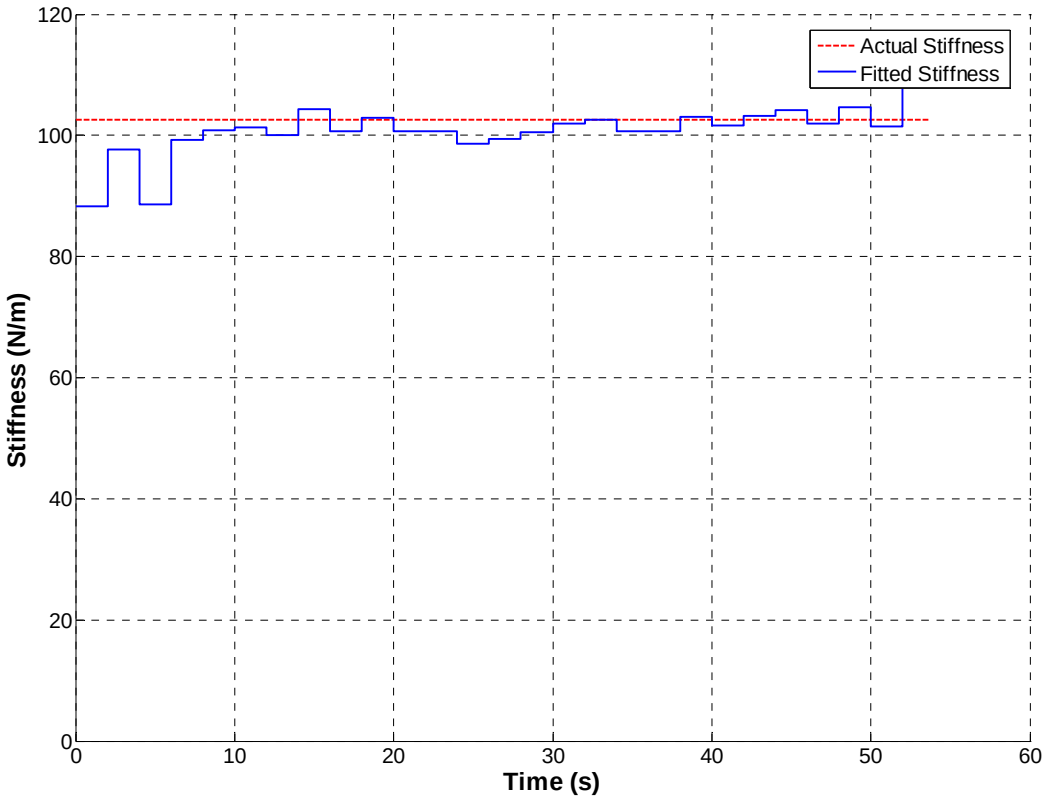

Figure 5.5: Identified stiffness of bottom storey of four storey simulated structure

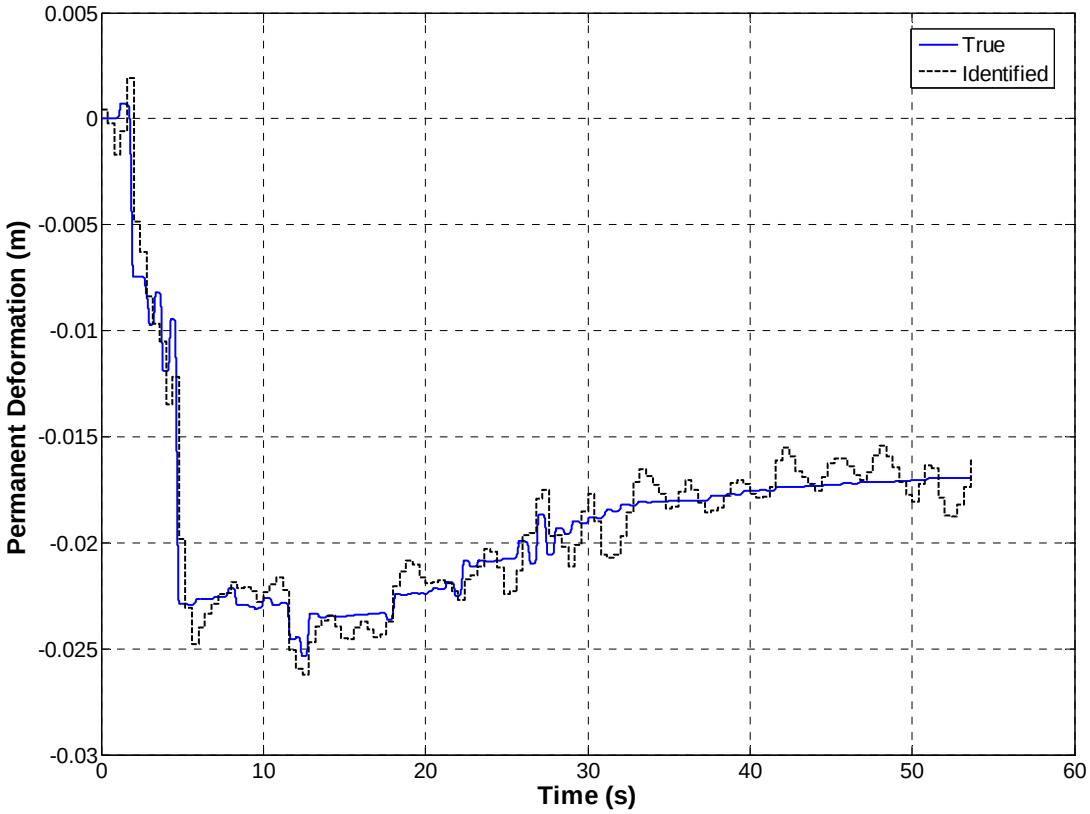

Figure 5.6: Identified permanent displacement of bottom storey of four storey simulated structure

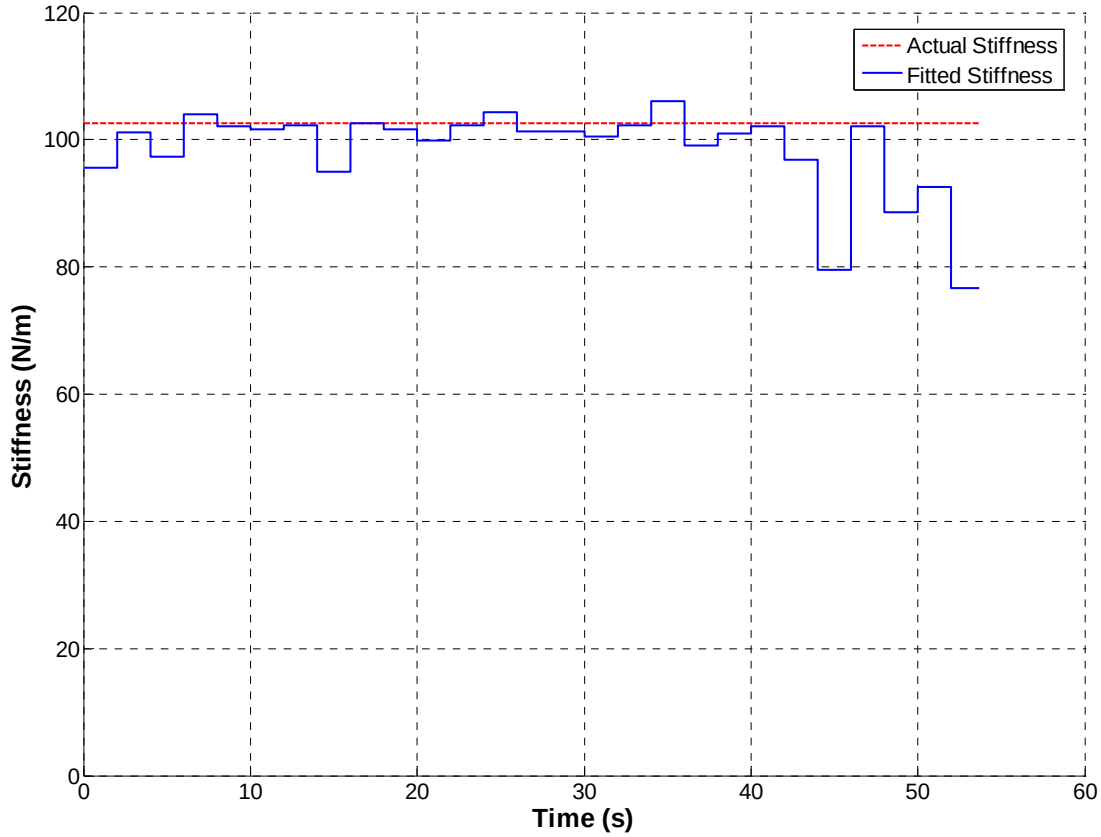

Figure 5.7: Identified stiffness of top storey of four storey simulated structure

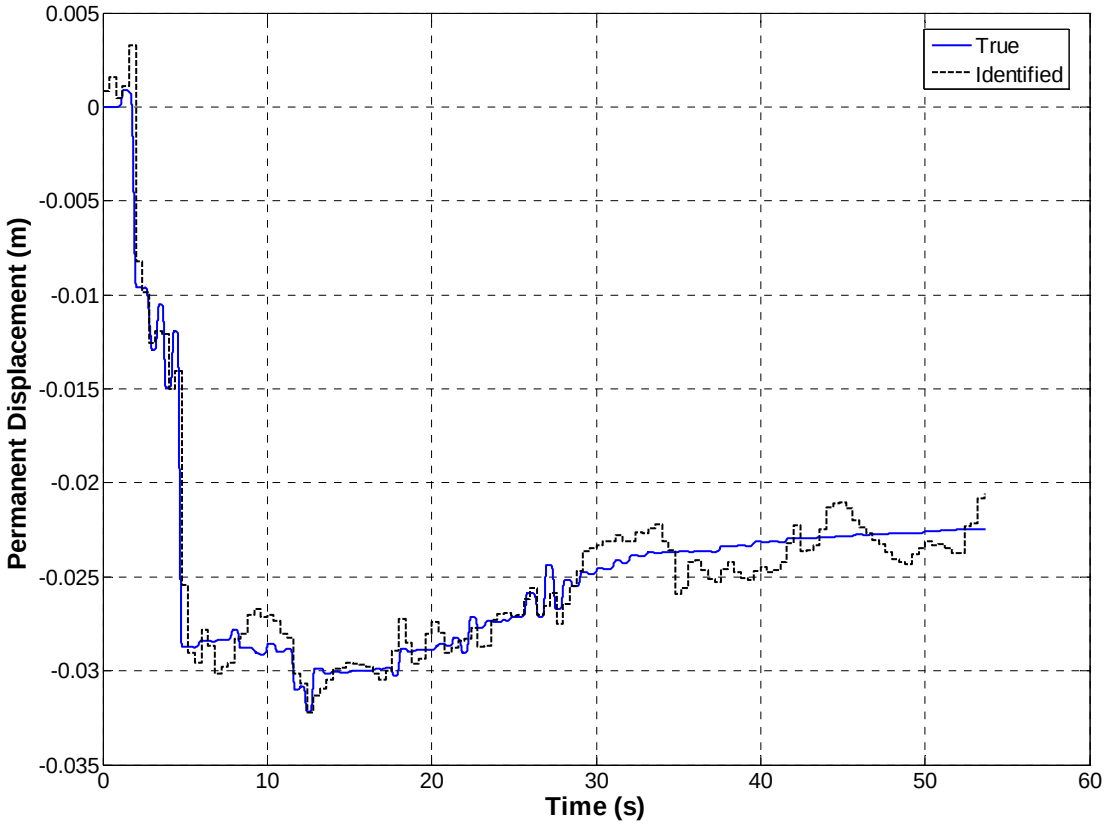

Figure 5.8: Identified permanent displacement of top storey of four storey simulated structure

The results presented in Sections 5.1 and 5.2 show that the algorithm is capable of accurately identifying the stiffness of elastic structures, and the stiffness and permanent displacement of inelastic structures. Also, the algorithm appears to be robust in the presence of noise and applicable to a two degree of freedom system without noticeable loss of accuracy.

5.3 Four Storey Shear Building with Bouc-Wen Hysteresis

5.3.1 Initial testing without noise

To assess the algorithms performance in a more realistic, multi-degree of freedom situation, it was applied to the four storey structure described in Chapter 4.1. Simulated acceleration data was sampled at 1 kHz and simulated displacement data was sampled infrequently at 10Hz. It was assumed that the damping matrix (4.1) was known. In reality this would have to be estimated however the results to follow in Chapter 5.3.3 show that a realistic choice of damping matrix has a lesser effect on the fitted results.

Figure 5.9 shows a comparison of the (simulated) real and fitted permanent displacements of the bottom storey with $\Delta t_{kp} = 2$ and $\Delta t_Z = 0.4$. Acceleration and displacement data were not subject to noise in this case.

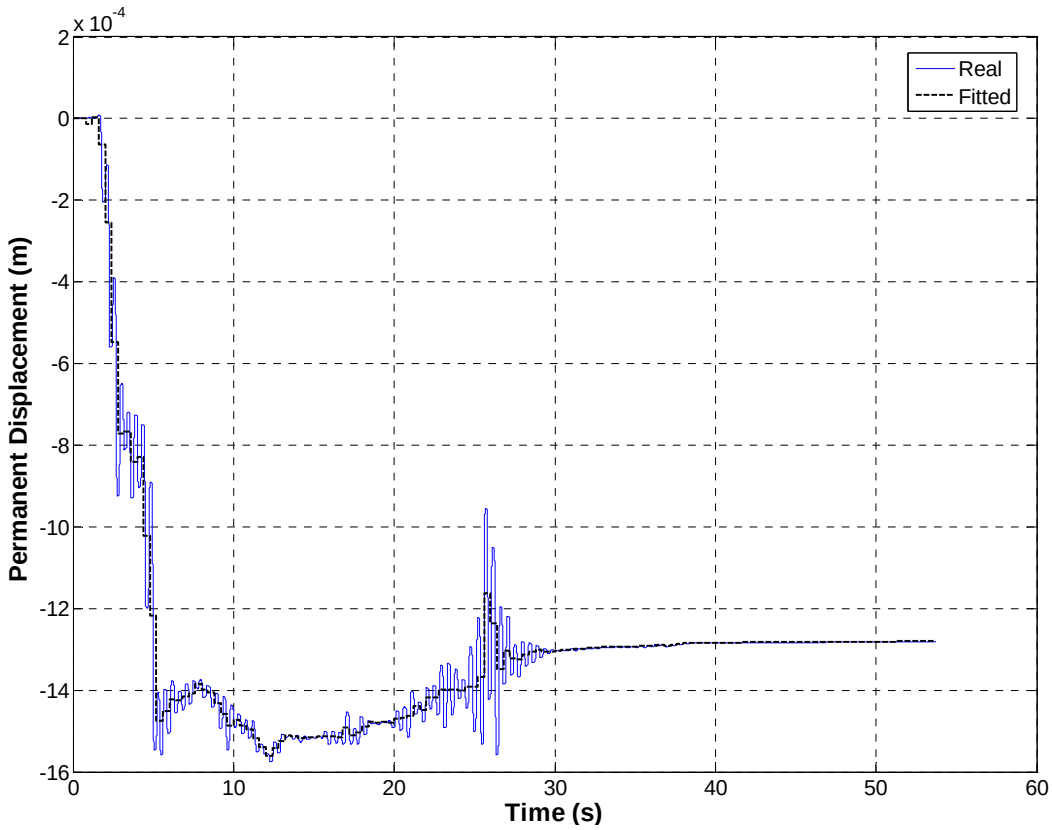

Figure 5.9: Permanent displacement comparison for four storey simulated structure without noise

Figure 5.10 shows the displacement of the bottom storey overlaid with the fitted permanent displacement:

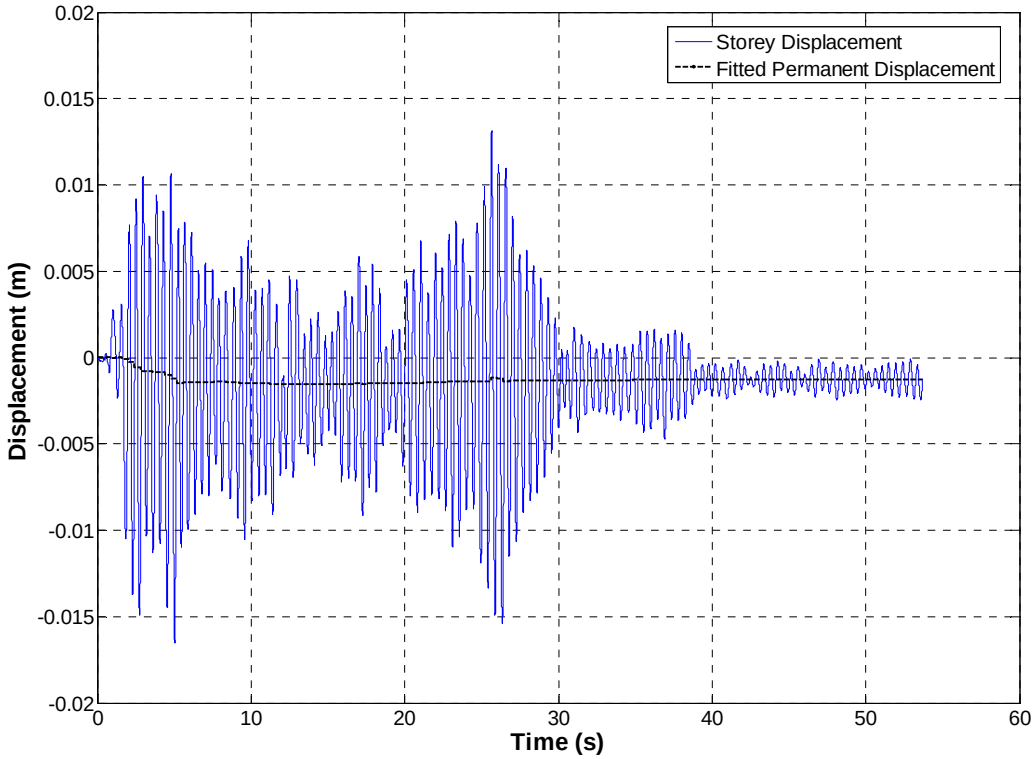

Figure 5.10: Permanent displacement overlay for four storey simulated structure without noise

In this case, the piecewise fitted stiffness for the bottom storey had a mean value of 1608 N/m, a maximum value of 1613 N/m and a minimum value of 1591 N/m corresponding to a maximum error of 1% from the true simulated value.

Figures 5.9 and 5.10 demonstrate that the algorithm accurately identifies permanent displacement and final residual displacement. The piecewise constant approximation to $\Delta Z(t)$ does not capture small-amplitude, high frequency oscillations in permanent displacement, as evident in Figure 5.9. However major trends and the more important final value are captured very accurately.

In the given example, the fundamental period of the structure is 0.45s and the maximum permanent displacement is 1.6mm. Real structures with typically longer periods will undergo larger permanent displacements occurring over longer periods of time without noticeable high frequency oscillation, such as is seen in Figure 5.9. Thus, permanent displacements in real structures should be accurately captured using a piecewise constant approximation. The small oscillations seen in Figure 5.9 could also be more accurately captured using a smaller fitting period, Δt_Z. However, the fitted parameters would then be more easily affected by noise since a smaller fitting period uses less data points to identify each parameter.

Figure 5.11 shows a comparison of the actual stiffness and fitted stiffness of the bottom storey. Comparison to Figures 5.9 and 5.10 reveals that the fitted stiffness $k_p^{(1)}$ is slightly less accurate during highly non-linear periods, that is, where the permanent displacement is changing rapidly. This result occurs because the piecewise constant approximation to $\Delta Z^{(1)}(t)$ in Equation (3.25) may no longer accurately model the hysteretic behaviour over the interval. However, this error does not reduce the algorithms ability to identify significant long-term reductions in structural

stiffness as it is within 1.2% of the true value and is only present during periods of substantial yielding. The error could be substantially reduced by fitting a piecewise linear permanent displacement parameter rather than piecewise constant. However this may cause larger oscillations in $\Delta Z^{(1)}(t)$ due to the greater freedom that is introduced into the fitting process. These oscillations may also be more susceptible to noise and thus less accurately reflect realistic permanent displacement behaviour.

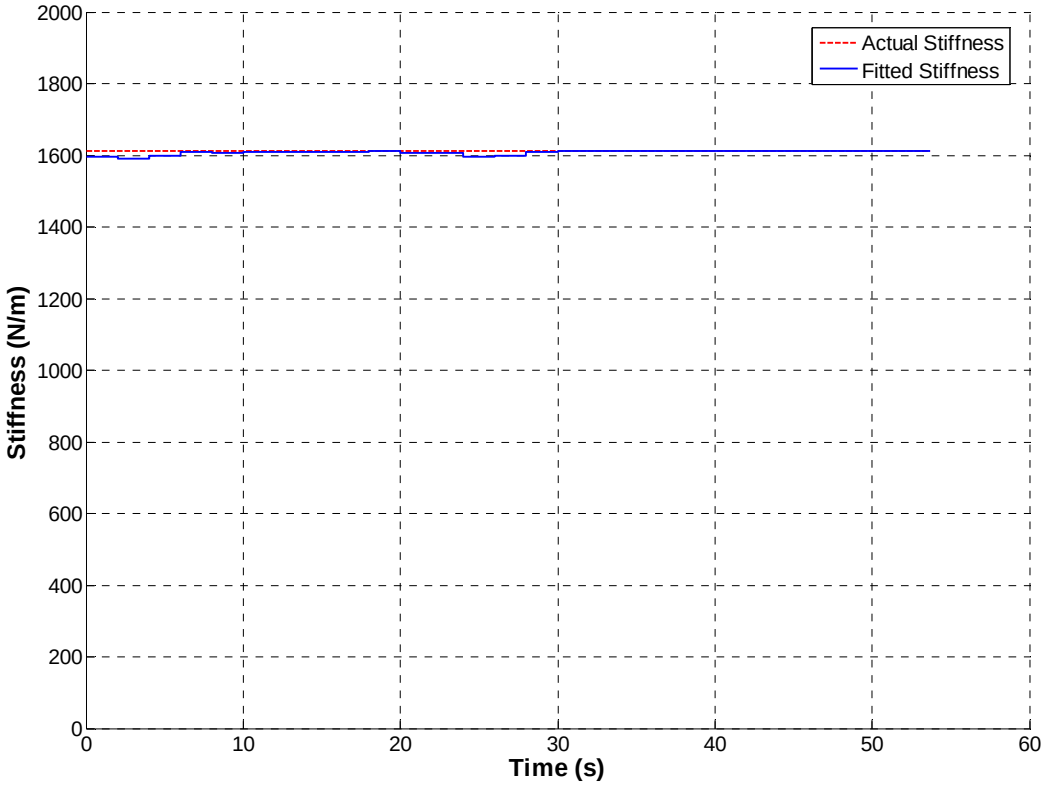

Figure 5.11: Fitted stiffness variation for four storey simulated structure without noise

5.3.2 Effect of Noise

Figure 5.12 compares the real and fitted permanent displacements of the bottom storey using acceleration and displacement data subject to 10% uniformly distributed noise. In Figure 5.13, the fitted permanent displacement is overlaid on the actual displacement of the bottom storey. Figures 5.12 and 5.13 show that even in the presence of 10% noise, the algorithm accurately

identifies the major trends in permanent displacement and the final residual displacement. In Figure 5.14, the actual stiffness and fitted stiffness of the bottom storey are compared with 10% noise applied to data.

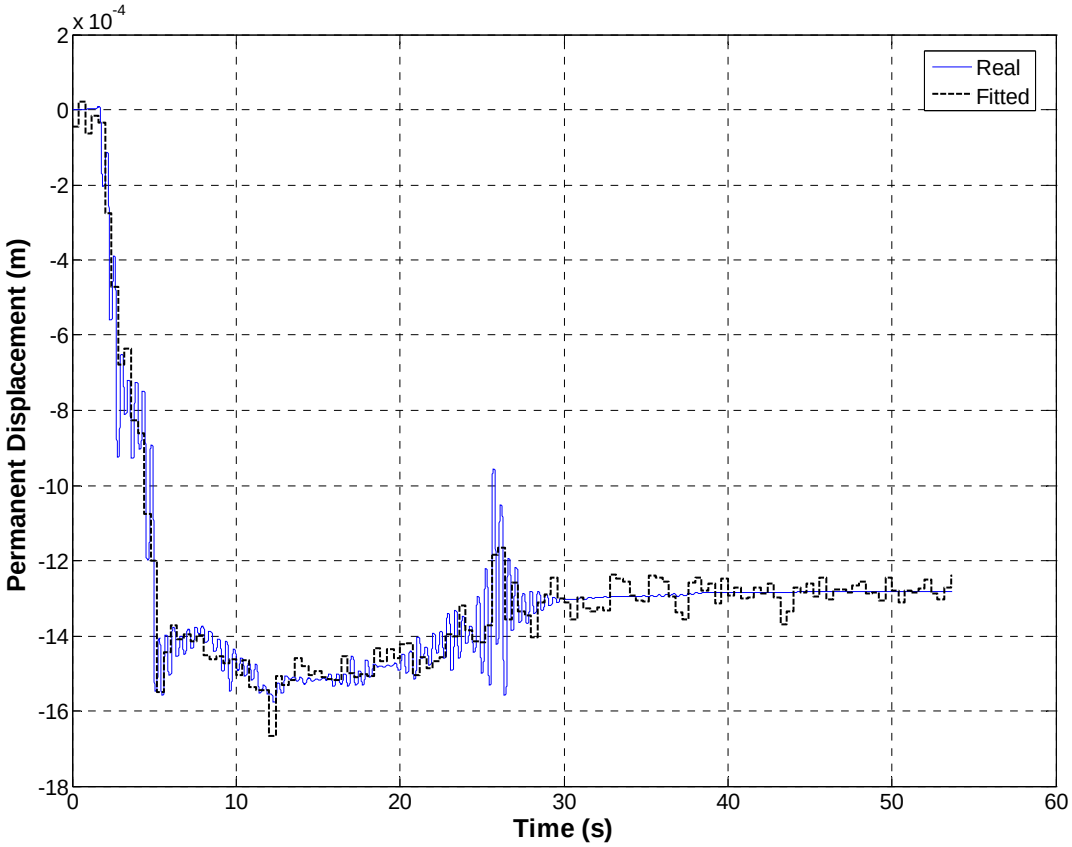

Figure 5.12: Permanent displacement comparison for four storey simulated structure with 10% noise

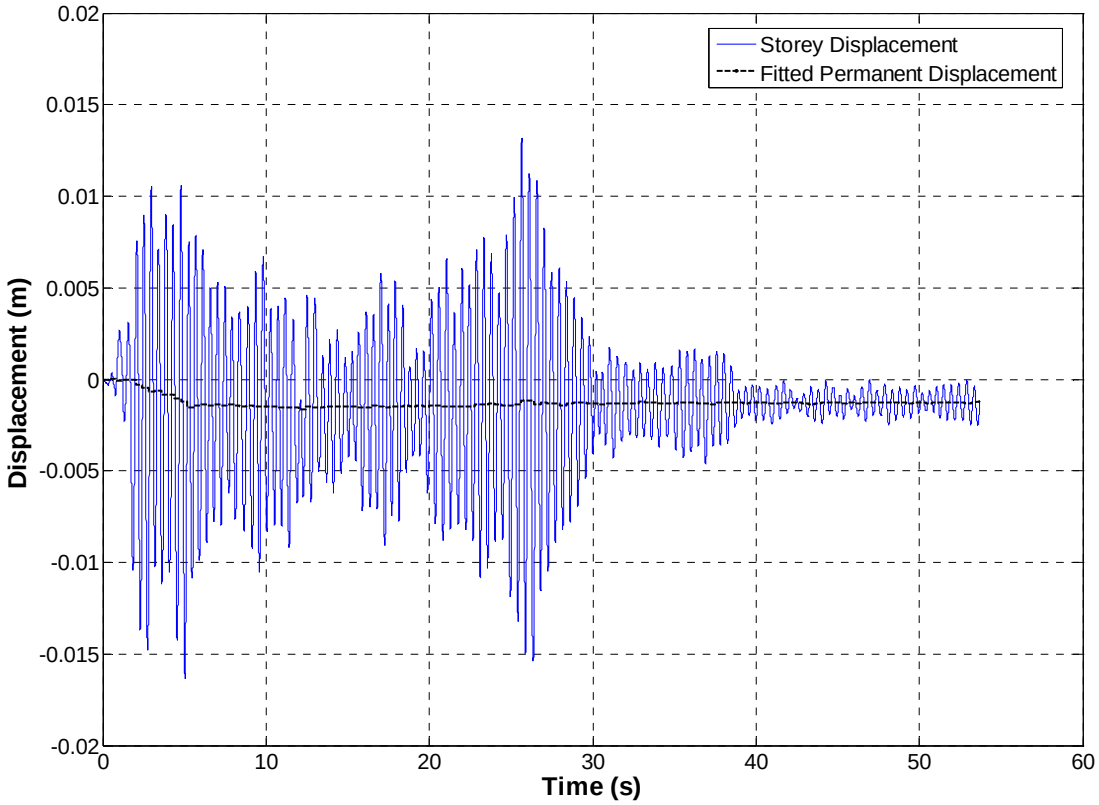

Figure 5.13: Permanent displacement overlay for four storey simulated structure with 10% noise

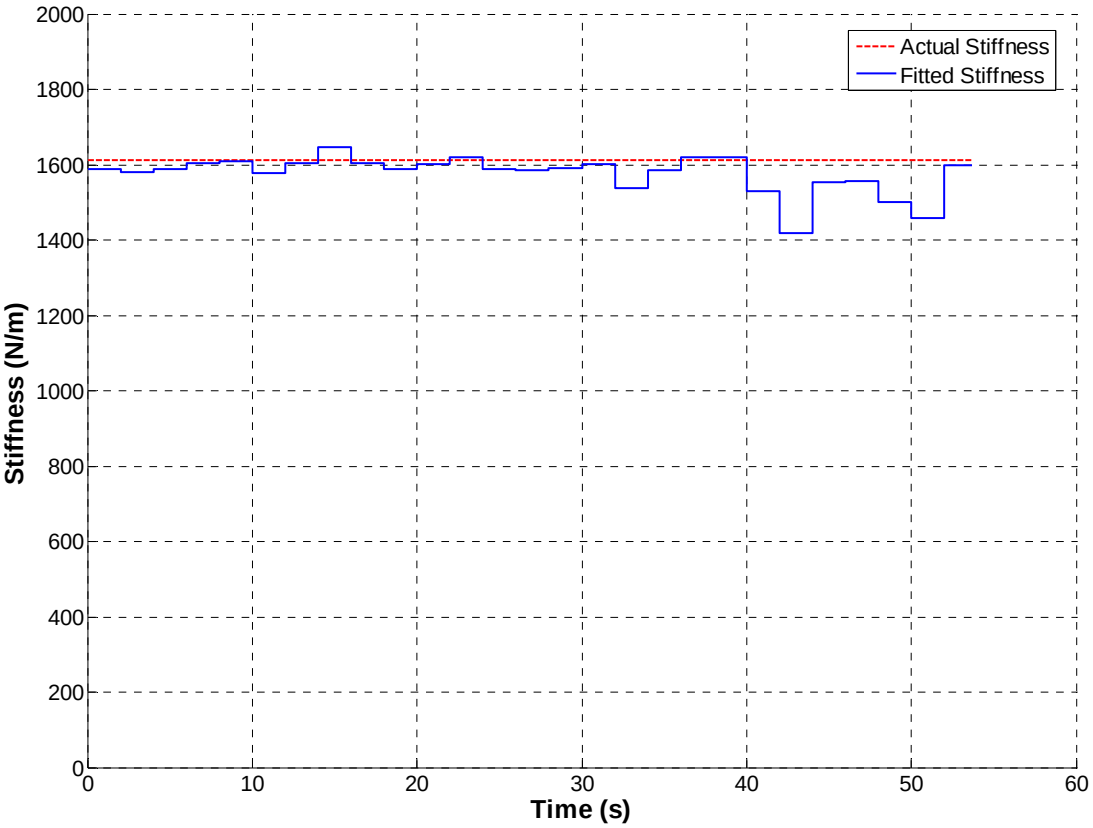

Figure 5.14: Fitted stiffness variation for four storey simulated structure with 10% noise

Figure 5.14 shows that stiffness is accurately identified over the first 30s or response. However, during the last portion there is a significant drop variation. This is because the ground motion and structural response, as shown in Figure 5.13, decreases significantly at around 30s. Therefore the relative magnitude of noise becomes much greater after this point. However as can be seen in Figure 5.12, structural damage only occurs during the first 30s of the earthquake where the magnitude of response is large. Thus, the accuracy of fitted results after 30s is largely inconsequential and typically such small responses or the identification results from them would be ignored.

The effect of noise on the accuracy of fitted stiffness and permanent displacement was quantified using two sets of twenty simulations, with 5% uniformly distributed noise applied to the first set and 10% uniformly distributed noise applied to the second set. Tables 5.1 and 5.2 provide a summary of the mean and variation of the fitted stiffness across the data sets.

Table 5.1: Fitted stiffness confidence intervals for 5% noise data

Storey	Mean Stiffness (N/m)	90% Confidence Maximum	90% Confidence Minimum
1	1604	1655	1561
2	1606	1695	1553
3	1607	1707	1530
4	1603	1719	1470

Table 5.2: Fitted stiffness confidence intervals for 10% noise data

Storey	Mean Stiffness (N/m)	90% Confidence Maximum	90% Confidence Minimum
1	1602	1671	1531
2	1604	1717	1499
3	1598	1710	1433
4	1579	1832	1272

Tables 5.1 and 5.2 show that for 5% noise, the error of the mean fitted stiffness is within 0.5% of the true value 1610 N/m for every storey. For 10% noise the mean fitted stiffness is within 2% for every storey. Stiffness degradation was not modeled in this example therefore the mean

stiffness values shown in Tables 5.1 and 5.2 accurately represent the structural stiffness, however if stiffness degradation was present it would be necessary to accurately identify stiffness over each time period Δt_{kp} rather than the entire record. Additionally, the mean identified stiffness would be less meaningful for proof of concept. Tables 5.3 and 5.4 provide a summary of the fitted stiffness data sets over the first 30s of earthquake response. In this case, the significant and useful response occurs in this time period. The results show tighter identified results, as expected from the results in Figure 5.14.

Table 5.3: Fitted stiffness confidence intervals for 5% noise data over first 30 seconds

Storey	Mean Stiffness (N/m)	90% Confidence Maximum	90% Confidence Minimum
1	1600	1622	1564
2	1603	1626	1570
3	1607	1632	1573
4	1612	1650	1574

Table 5.4: Fitted stiffness confidence intervals for 10% noise data over first 30 seconds

Storey	Mean Stiffness (N/m)	90% Confidence Maximum	90% Confidence Minimum
1	1601	1628	1556
2	1602	1635	1561
3	1607	1645	1565
4	1608	1659	1556

To quantify the effects of noise on the identified permanent displacement, the error between each fitted permanent displacement and the mean of the real permanent displacement over that time period is given as a percentage. Tables 5.5 and 5.6 show the mean and 90% confidence intervals of the error for 5% error and 10% uniform noise. The permanent displacement of any storey is calculated to within 10% of the final residual displacement with 90% confidence for a 5% noise level, and to within 13% of the final residual displacement with 90% confidence for a 10% noise level.

Table 5.5: Fitted permanent displacement percentage error for 5% noise

Storey	Mean Error	90% Confidence Maximum	90% Confidence Minimum
1	0.17	9.70	-7.75
2	0.18	8.32	-5.73
3	0.19	8.86	-6.39
4	0.18	9.84	-6.74

Table 5.6: Fitted permanent displacement percentage error for 10% noise

Storey	Mean Error	90% Confidence Maximum	90% Confidence Minimum
1	0.05	12.63	-10.45
2	0.18	9.38	-6.75
3	0.21	9.86	-7.28
4	0.16	10.23	-7.42

The results in Tables 5.5 and 5.6 do not fully reflect the fact that the overall trend of the fitted permanent displacement accurately tracks the trend of the 'true' permanent displacement. Thus, a more suitable way of testing the accuracy of the fitted permanent displacement would be to use a short term moving average on both pieces of data before computing the mean error. The results of taking a 3 point moving average are detailed in Tables 5.7 and 5.8, as expected the fitted errors are dramatically reduced.

Table 5.7: Fitted permanent displacement percentage error for 5% noise with 3 point moving average

Storey	Mean Error	90% Confidence Maximum	90% Confidence Minimum
1	0.16	6.07	-4.68
2	0.17	4.78	-4.10
3	0.19	5.02	-4.46
4	0.17	6.22	-5.21

Table 5.8: Fitted permanent displacement percentage error for 10% noise with 3 point moving average

Storey	Mean Error	90% Confidence Maximum	90% Confidence Minimum
1	0.05	7.23	-5.90
2	0.18	6.05	-4.82
3	0.21	5.53	-4.85
4	0.15	6.41	-5.58

Identification accuracy could also be improved by measuring ground acceleration and structural response at a higher frequency than 1kHz, thus providing more measurements to fit each parameter and reducing the effect of noise.

5.3.3 Effect of Damping Matrix Model

A damping matrix is required for the identification procedure. However, all such models are approximations at best. During large earthquakes, the majority of damping is provided by hysteretic damping of structural members, rather than viscous damping. Thus, inaccuracy in the estimation of the damping matrix should have little effect on the accuracy of fitted parameters. Figure 5.15 shows a comparison between the real permanent displacement of the first storey of the simulated structure and the permanent displacement fitted assuming no damping $(\mathbf{C}=0)$ and using data with 10% noise. Similarly, Figure 5.16 compares the real and fitted stiffness. Finally, Figures 5.17 and 5.18 show the comparisons where the coefficients of the damping matrix have been doubled.

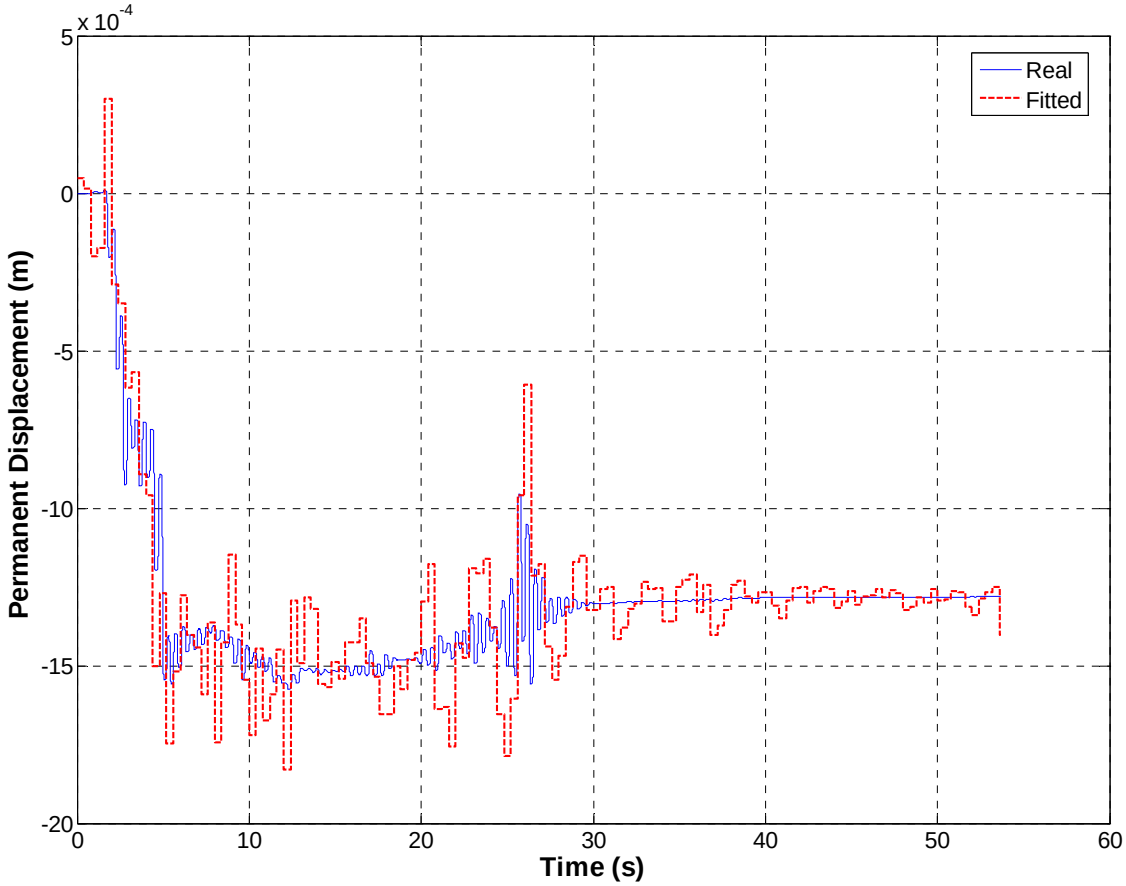

Figure 5.15: Permanent displacement comparison for four storey simulated structure with 10% noise and damping neglected

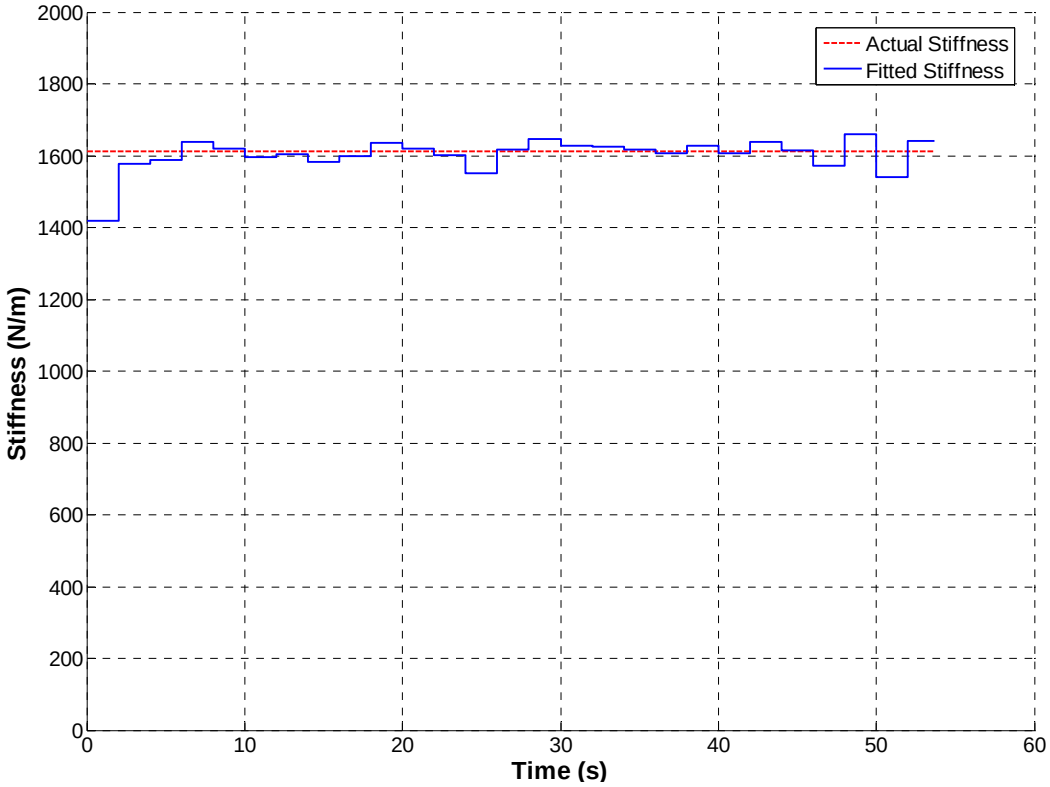

Figure 5.16: Fitted stiffness variation for four storey simulated structure with 10% noise and damping neglected

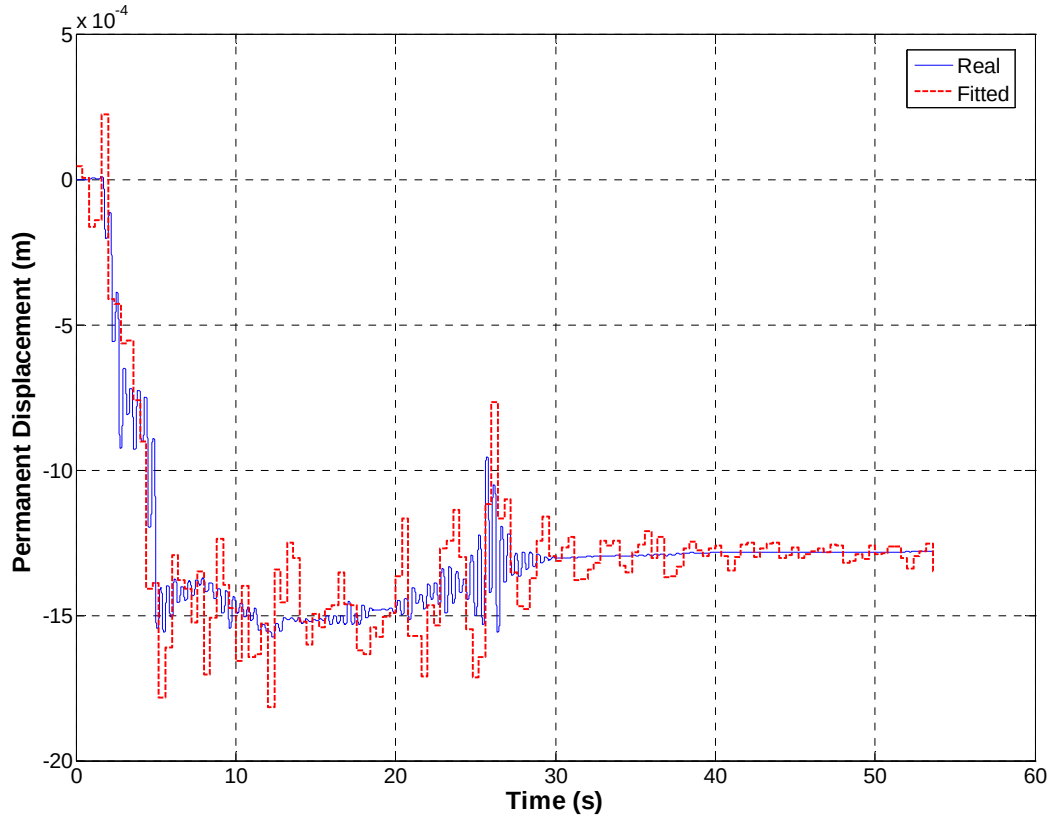

Figure 5.17: Permanent displacement comparison for four storey simulated structure with 10% noise and over-estimated damping matrix

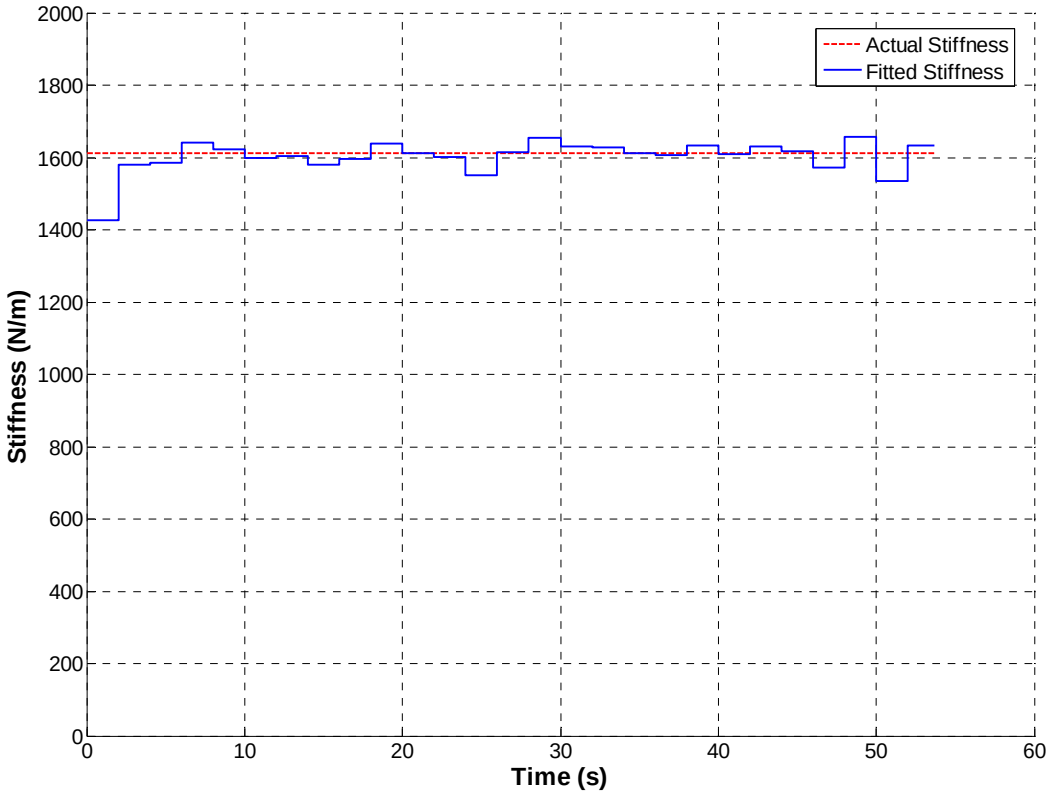

Figure 5.18: Fitted stiffness variation for four storey simulated structure with 10% noise and over-estimated damping matrix

The identified parameters in Figures 5.15-5.18 match the real values well. Major trends in permanent displacement are captured accurately, although they are accompanied by small oscillation (when $\mathbf{C} = 0$). The fitted stiffness is very close to the actual stiffness, with a reduction in accuracy during periods of very low response. These results show that even in the presence of 10% noise, an extreme error in estimation of damping matrix properties results in only minimal oscillation and error in the fitted results for these ground motions.

5.3.4 Effect of Displacement Measurement Frequency

The results shown in Sections 5.3.1-5.3.2 were found using a 10Hz measured displacement to correct the integral approximations to displacement, velocity and acceleration. In certain practical situations, displacement measurement may not be available at such high frequency. In

these cases, the estimated properties will be allowed to drift between displacement measurements more than with a 10Hz displacement measurement, potentially resulting in a reduction of identification accuracy. Figure 5.19 shows a comparison between the real permanent displacement of the structure and the fitted permanent displacement using a displacement sampling rate of 1Hz. Figure 5.20 compares the real and fitted stiffness. Acceleration and displacement data was subject to 10% uniformly distributed noise.

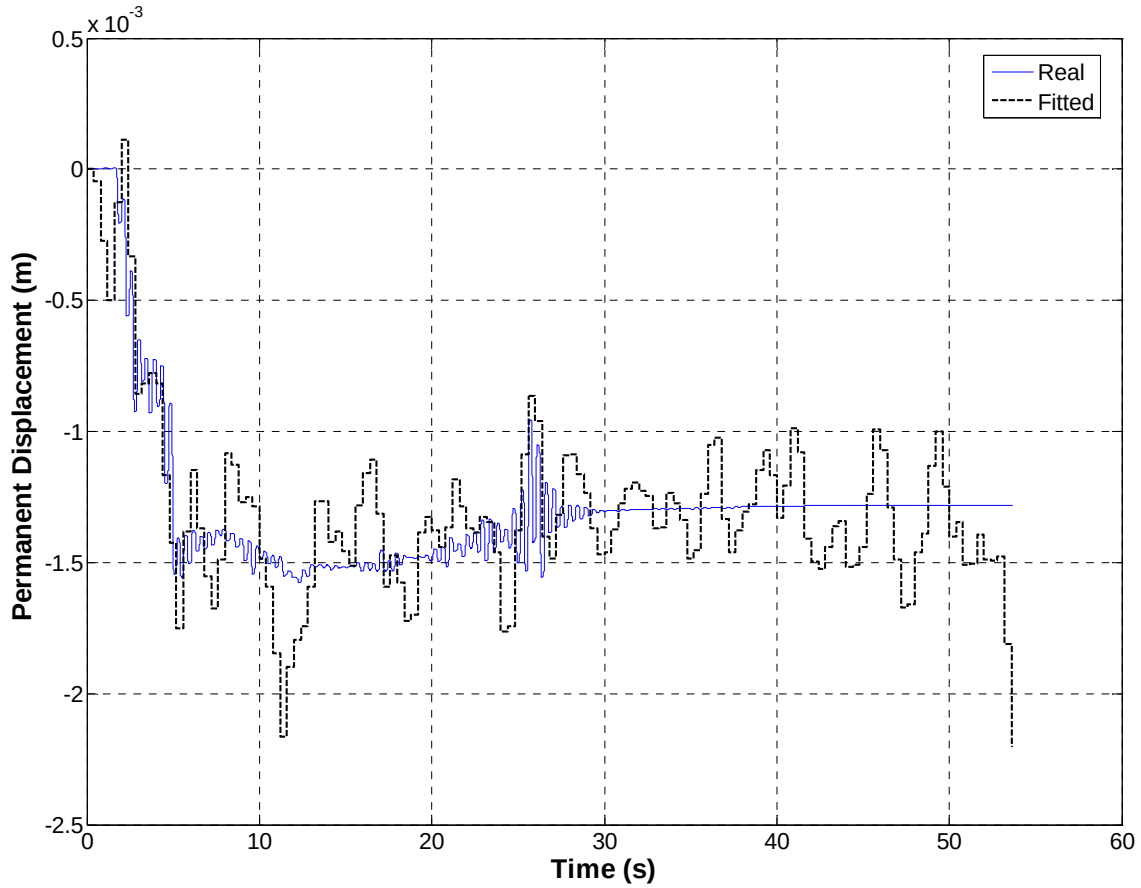

Figure 5.19: Comparison of real and fitted permanent displacement for four storey simulated structure with 1Hz displacement measurement

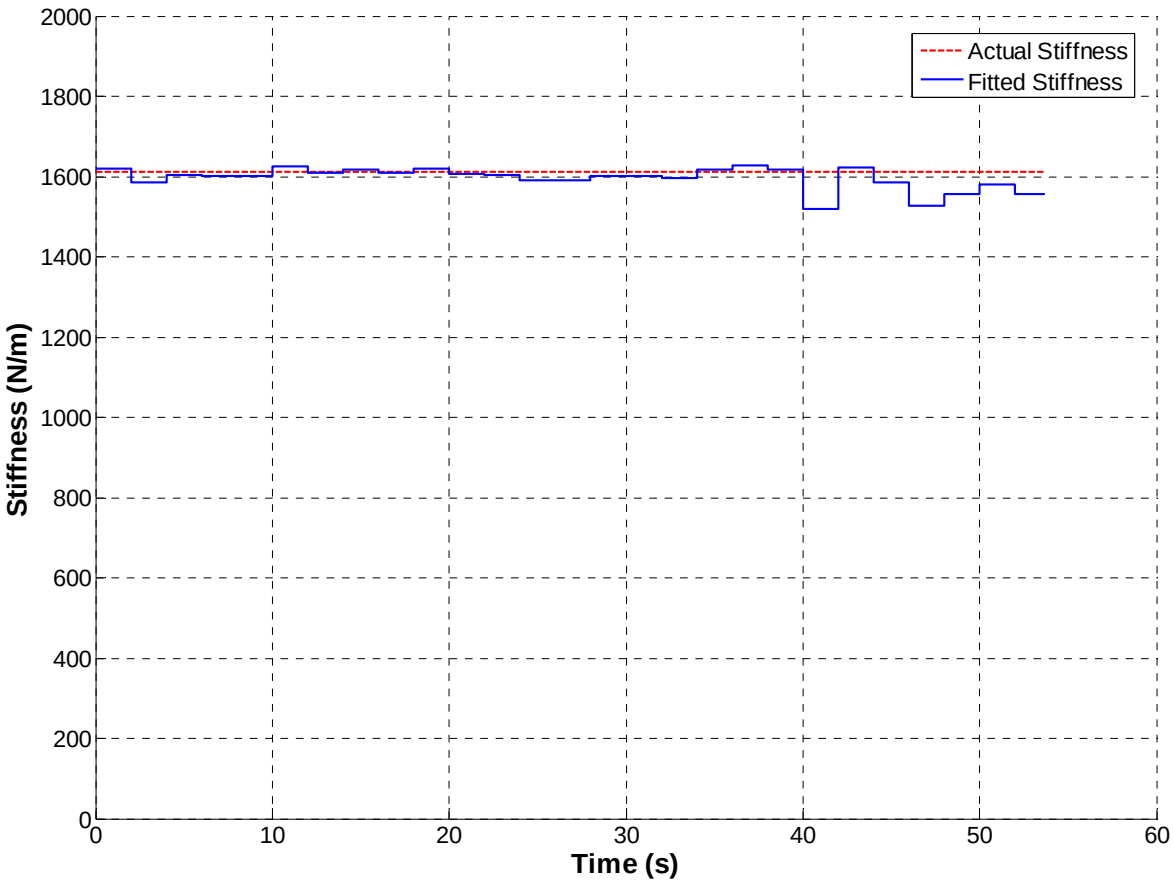

Figure 5.20: Comparison of real and fitted stiffness variation for four storey simulated structure with 1Hz displacement measurement

The fitted stiffness shown in Figure 5.20 is very close to the real value over the first 30s where significant response is observed. The identified permanent displacement shown in Figure 5.19 matches major trends of the real permanent displacement well. However, there is oscillation present in the fitted parameter. This oscillation could be largely eliminated using a moving average at the cost of ignoring some sharp peaks in permanent displacement. Figures 5.19 and 5.20 show that decreasing the frequency of displacement measurement does not significantly reduce the algorithms ability to identify stiffness and major trends in permanent displacement, even in the presence of 10% uniformly distributed noise.

CHAPTER 6 - STUCTURAL IDENTIFICATION USING SIMULATED AND EXPERIMENTAL DATA

6.1 Four Storey Non-Linear Frame Structure

6.1.1 Non-Linear Baseline Model

The algorithm described in Chapter 3.2 was tested for performance in a real situation using measured data from the frame structure described in Chapter 4.2. Figure 6.1 shows the fitted stiffness of the bottom storey compared to that predicted by pushover analysis of a computational model constructed by Kao (1998). The fitted permanent displacement of the bottom storey is overlaid with the measured displacement in Figure 6.2, showing a good overall match. The bottom storey is chosen since the structure is first mode dominant and the bottom storey therefore yields more than the higher storeys.

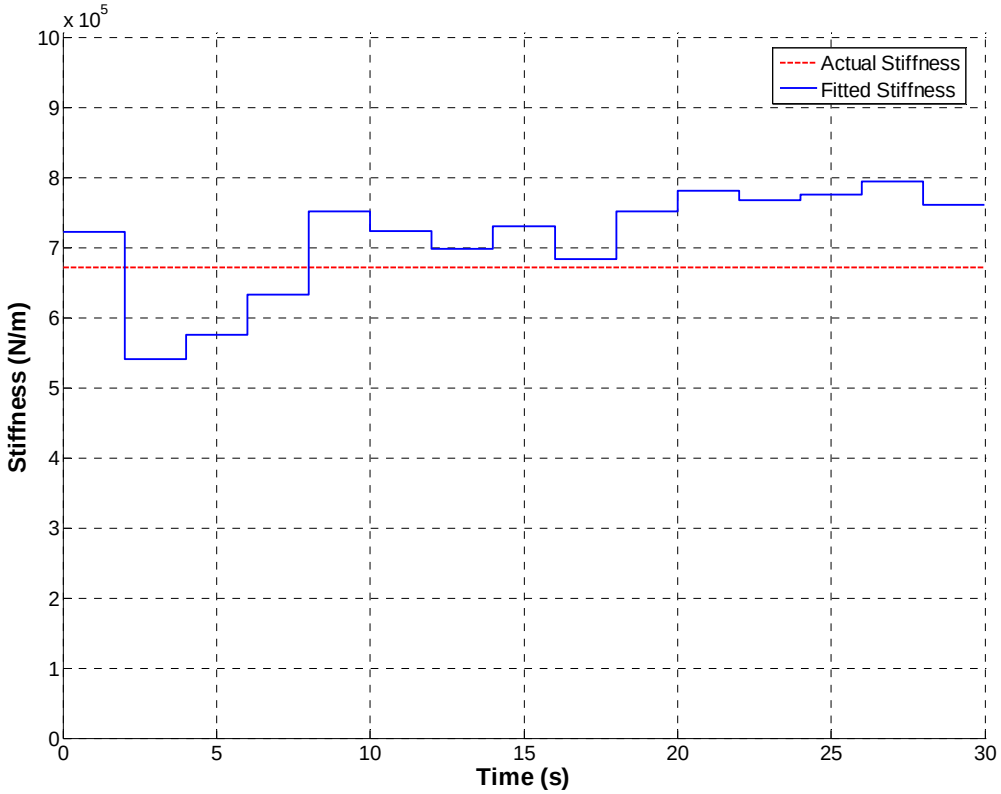

Figure 6.1: Bottom storey stiffness comparison for real frame structure

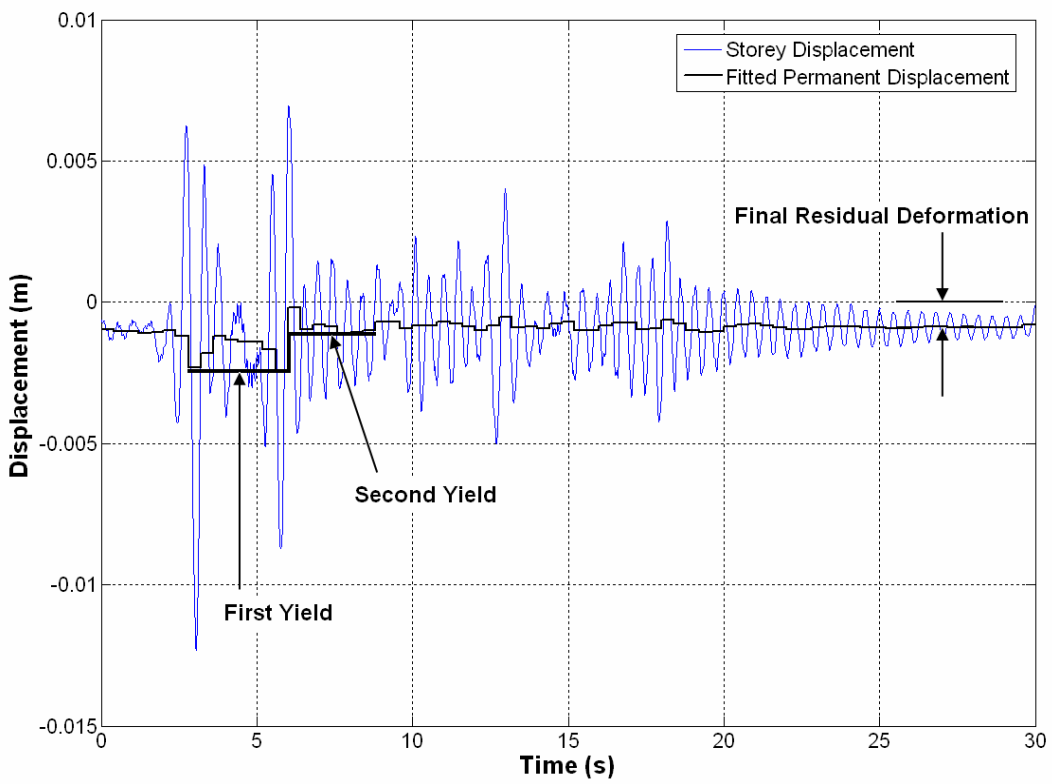

Figure 6.2: Bottom storey permanent displacement overlay for real frame structure

It can be seen in Figure 6.1 that, as with the simulated results, the value of the fitted stiffness drops between 2 and 8 seconds where significant yielding occurs. The drop in identified stiffness is greater in this real case than in the simulated case since the magnitude of yielding is greater. In addition, the experimental value of α is more likely $\alpha = 0.01 - 0.02$ instead of the $\alpha = 0.1$ used in simulation. Table 6.1 shows a summary of the fitted stiffnesses over the entire record where the predicted value is from the pushover analysis.

Table 6.1: Summary of fitted stiffness for real structure using non-linear baseline model

Storey	Mean (kN/m)	Max (kN/m)	Min (kN/m)	Predicted (kN/m)
1	713	795	541	672
2	765	836	614	640
3	662	724	548	601
4	490	579	370	544

As expected, the most significant yielding occurs at the same time as the largest peaks in displacement response. The structure also yields effectively twice, as shown in Figure 6.2, indicating that total absolute permanent deflection was reasonably captured. The fitted permanent displacement is therefore reasonable in both behaviour and magnitude. Table 6.2 shows the percentage error between the fitted and actual final residual displacements for all storeys.

Table 6.2: Percentage error in identified final residual displacement

Storey	Residual % Error
1	0.27
2	0.67
3	1.27
4	0.24

The smallest variation in stiffess occurs during the period from 20 to 30 seconds, where there is a reasonable magnitude of response, but little yielding. Thus, a more accurate identification of

stiffness occurs in this period. Table 6.3 shows a summary of the fitted stiffnesses over this period.

Table 6.3: Summary of fitted stiffness for real structure from 20-30s using non-linear baseline model

Storey	Mean (kN/m)	Max (kN/m)	Min (kN/m)	Predicted (kN/m)
1	777	795	762	672
2	810	836	791	640
3	674	686	664	601
4	532	576	458	544

For the first three storeys, the stiffness values in Table 6.3 are within 3.5% of the mean value for, but there is a greater maximum error for the fourth storey of 14%. However, as can be inferred from Equation (3.23), the parameters for the fourth storey are fitted using measured data from only one storey. Thus, it would be expected that with less data the identification process is more susceptible to noise or sensor error. Other storeys are fitted using measured data from multiple storeys. Overall, the errors are within reasonable tolerances given variations in, for example, construction and material properties.

The fitted stiffnesses shown in Tables 6.1 and 6.3 differ more significantly from those predicted by pushover analysis. There are many factors which could contribute to these differences. Primarily, strain hardening of the plastic hinges due to the structure being repeatedly subjected to earthquake excitation. Also, Kao (1998) observed increases in flexural stiffness of structural members due to strain rate effects and the bauschinger effect during high amplitude motion. Finally, the pushover analysis model may also have limitations or other errors as it is a static test.

Damping was estimated to be 2% at each mode, however the structural damping has experimentally been found to be amplitude dependent. Kao (1998) found that at an acceleration of 0.06g the damping was 0.9%, while at 0.11g the damping was 1.5%. During testing, the maximum structural acceleration induced was approximately 0.6g, indicating that the damping

may in fact have been greater than the 2% assumed. Both differences would result in small or moderate errors in the fitted stiffness values. Again, all the results were within reasonable tolerances for assessing damage.

Another contributing factor to the differences between the identified and predicted stiffnesses is the modeling error. During fitting, the structure is assumed to be a four-degree of freedom shear building. However, in reality, the structure contains 51 structural members, which results in considerable geometric non-linearity during high-amplitude motion. In addition, the structure undergoes appreciable flexural deformation during earthquake excitation, which is not accounted for in the fitted baseline shear building model.

In Figure 6.3, a reduced period of $\Delta t_Z = 0.1\text{s}$ is used to fit the piecewise constant permanent displacement of the first storey. Note that the major permanent displacement trend is still correctly identified. A smaller period however, allows more freedom in the movement of the permanent displacement so that the identified response is more susceptible to noise and modeling error, as can be seen in Figure 6.3.

Overall, the method presented accurately identifies the stiffness and permanent deflections in the non-linear structure. Furthermore, the method is reasonably robust to realistic sensor errors and noise, as well as baseline model errors.

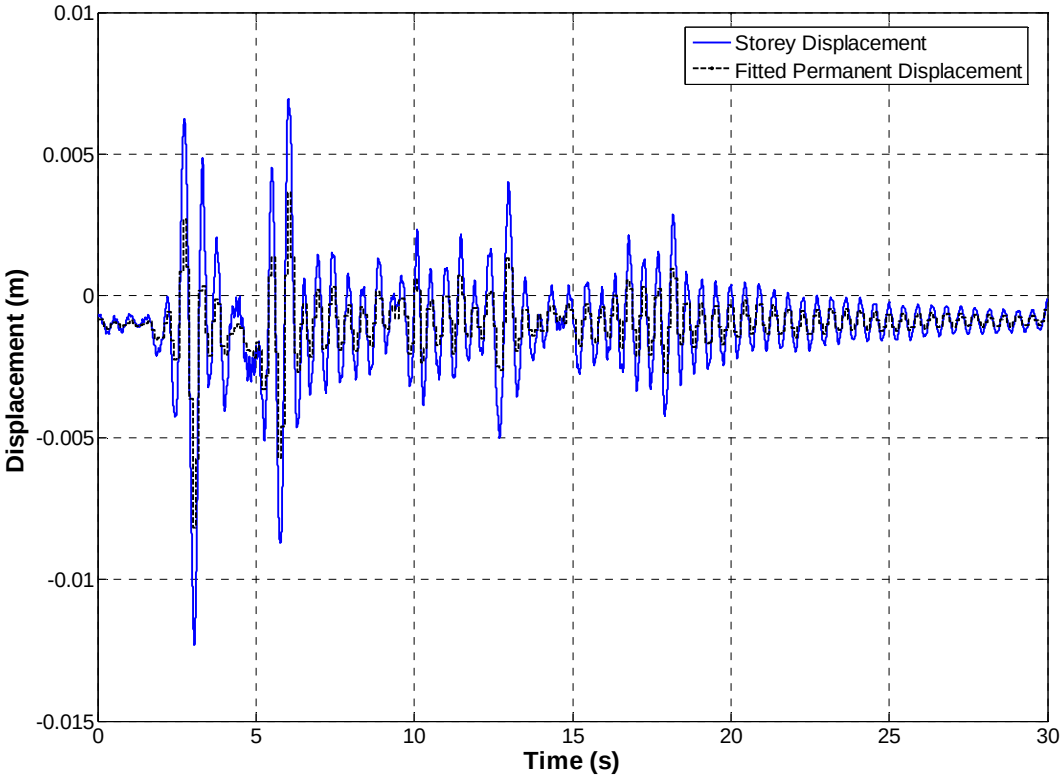

Figure 6.3: Overlaid permanent displacement for real frame structure with reduced fitting period

6.1.2 Linear Baseline Model

Figure 6.4 shows a comparison between the first storey stiffness fitted at 1.5s intervals using the linear baseline model and that estimated by pushover analysis. Figure 6.5 shows the same comparison for the fourth storey.

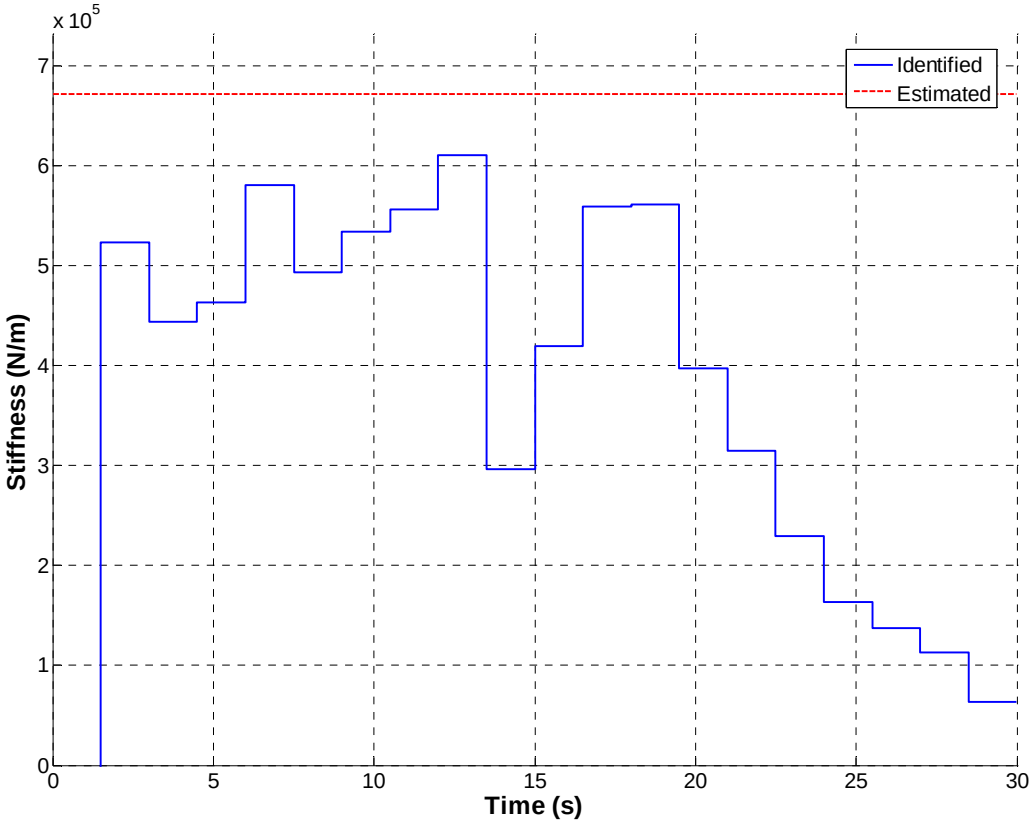

Figure 6.4: First storey stiffness comparison for real frame structure with linear baseline model

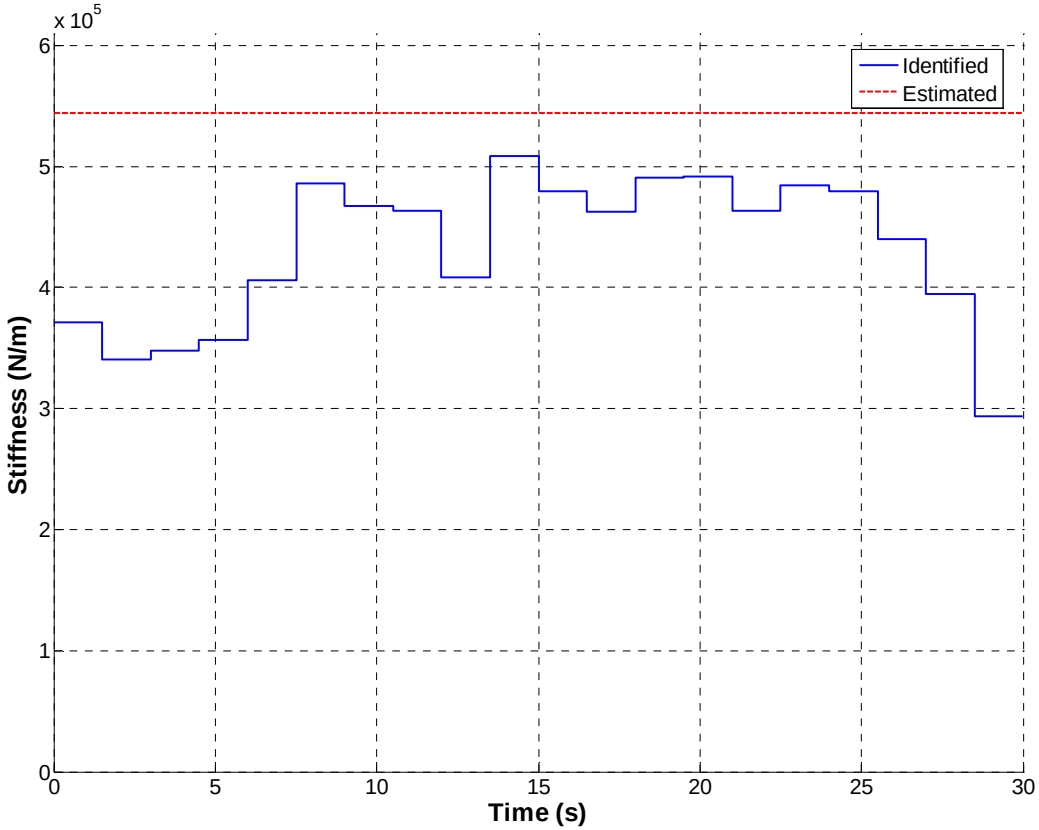

Figure 6.5: Fourth storey stiffness comparison for real frame structure with linear baseline model

Table 6.4 gives a summary of the fitted stiffnesses over the entire 30 second record where the predicted stiffness is from pushover analysis.

Table 6.4: Summary of fitted stiffnesses over entire period using linear baseline model

Storey	Mean (kN/m)	Max (kN/m)	Min (kN/m)	Predicted (kN/m)
1	373	611	-12	672
2	727	825	517	640
3	636	720	520	601
4	432	508	294	544

Unlike the results using the non-linear baseline model, Figures 6.4 and 6.5 reveal that stiffness is identified most accurately only certain periods from 10s to 20s where the magnitude of response is high. The accuracy falls away markedly after this period, in fact mean values over the record differ from the predicted values by up to ±45%. This reduction in accuracy is due to permanent deformation unaccounted for in the linear baseline model.

Examination of Equation (3.39) shows that where there is a significant displacement $r_w(t)$ without significant acceleration input $a_w(t)$, as would occur in the presence of a permanent deformation, the fitted stiffness value $k_l^{(w)}$ must compensate to ensure that the force balance is correctly satisfied. Permanent deformation causes inaccuracy in the identified stiffness value over the entire record as a result. Hence, the impact is greatest where the acceleration and dynamic structural response are minimal, in particular at the start of the record and during the last 10s of response. This effect is also most pronounced in the first storey where the most significant yielding occurs, as confirmed by Figure 6.4 and Table 6.4.

Table 6.5 gives a summary of the fitted stiffnesses over the period from 10s to 20s.

Table 6.5: Summary of fitted stiffness from 10s to 20s using linear baseline model

Storey	Mean (kN/m)	Max (kN/m)	Min (kN/m)	Predicted (kN/m)
1	506	611	297	672
2	772	825	718	640
3	675	720	638	601
4	468	508	408	544

The mean values in Table 6.5 match the predicted values to within ±25%, much closer than those in Table 6.4. However, there are still some discrepancies due to permanent deformations unaccounted for in the linear model. These further errors combine with material property differences and modeling error as discussed in Section 6.1.1 to give a significantly less accurate results as compared to using the Bouc-Wen model as the baseline.

Overall, Figures 6.4 and 6.5 demonstrate that although a linear baseline model is clearly an inappropriate description of this highly non-linear structure, it is still capable of approximately identifying the presence of structural damage. However, the degree of damage is only roughly approximated by examination of the identified stiffness. Large reductions in identified stiffness during periods of low magnitude response indicate the presence of a large permanent deformation, while a small reduction in stiffness indicates that there is very little or no permanent deformation.

In contrast, direct identification of permanent deformation provides a direct, easily understood link to structural damage. In particular, permanent deformation can be used to accurately estimate the cost and required time to repair damage. It also fits directly into the probabilistic fragility curves used in performance-based design and analysis.

6.1.3 Resimulation of Frame Structure Response

The structural response of the four-storey steel frame was resimulated using the estimated structural matrices and the parameters identified in Sections 6.1.1 and 6.1.2. Resimulation provides an overall validation of the fitting process by demonstrating that the identified parameters produce numerical solutions to Equations (2.1) and (2.13) that match the

displacement and acceleration responses reasonably accurately. It also provides another measure of the accuracy of the baseline model, by how well it reproduces the data.

In Figure 6.6, the resimulated displacement of the first storey is plotted using the parameters fitted from the non-linear model with $\Delta t_{kp} = 2$ and $\Delta t_Z = 0.4$.

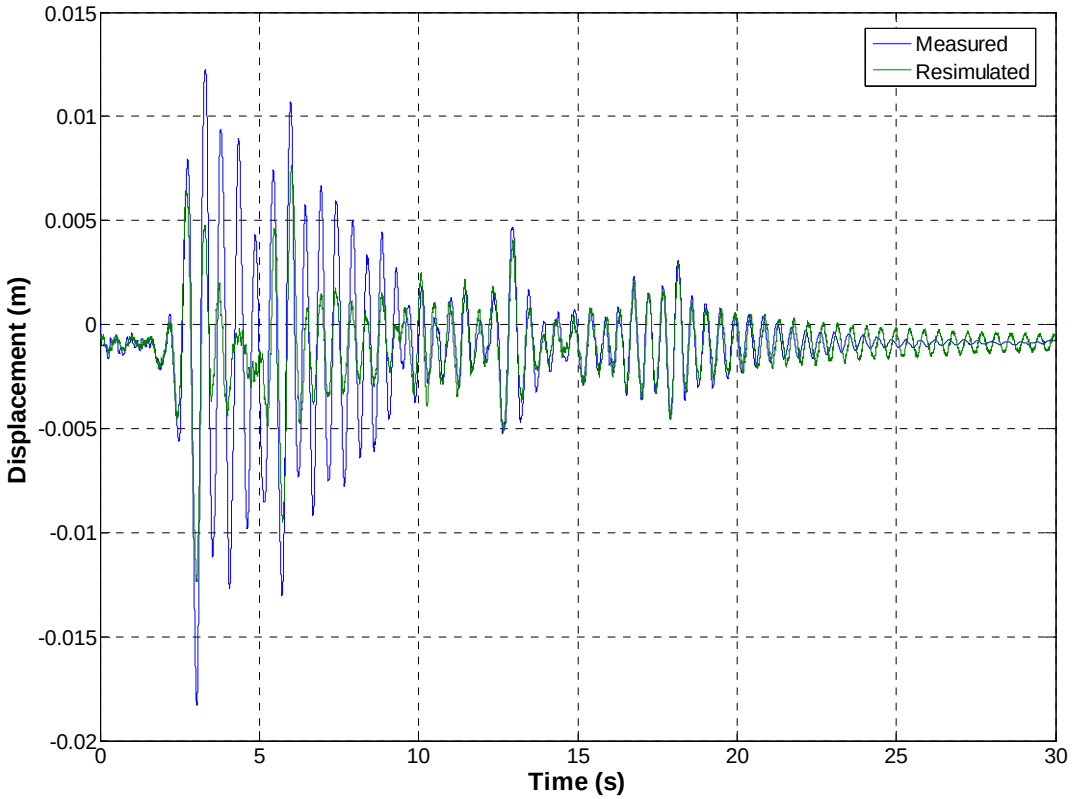

Figure 6.6: Resimulation for real frame structure with non-linear baseline model, $\Delta t_Z = 0.4$

The plot shows that the resimulated displacement matches the measured displacement quite closely except during periods of high yielding. This discrepancy is due to the piecewise constant constraint fitting period Δt_Z placed on the hysteretic parameter $\Delta Z^w(t)$. Reducing Δt_Z to 0.1s allows more freedom in $\Delta Z^w(t)$ and thus results in a much closer resimulation, see Figure 6.7. This is highlighted in Figure 6.8 where the error is plotted between the measured and resimulated results for both Figures 6.6 and 6.7. Allowing more freedom in $\Delta Z^w(t)$ however produces undesirable oscillations as can be seen in Figure 6.3. The longer fitting period of 0.4s is thus

more appropriate in this case, as it is the major trends in permanent displacement that are required to be identified.

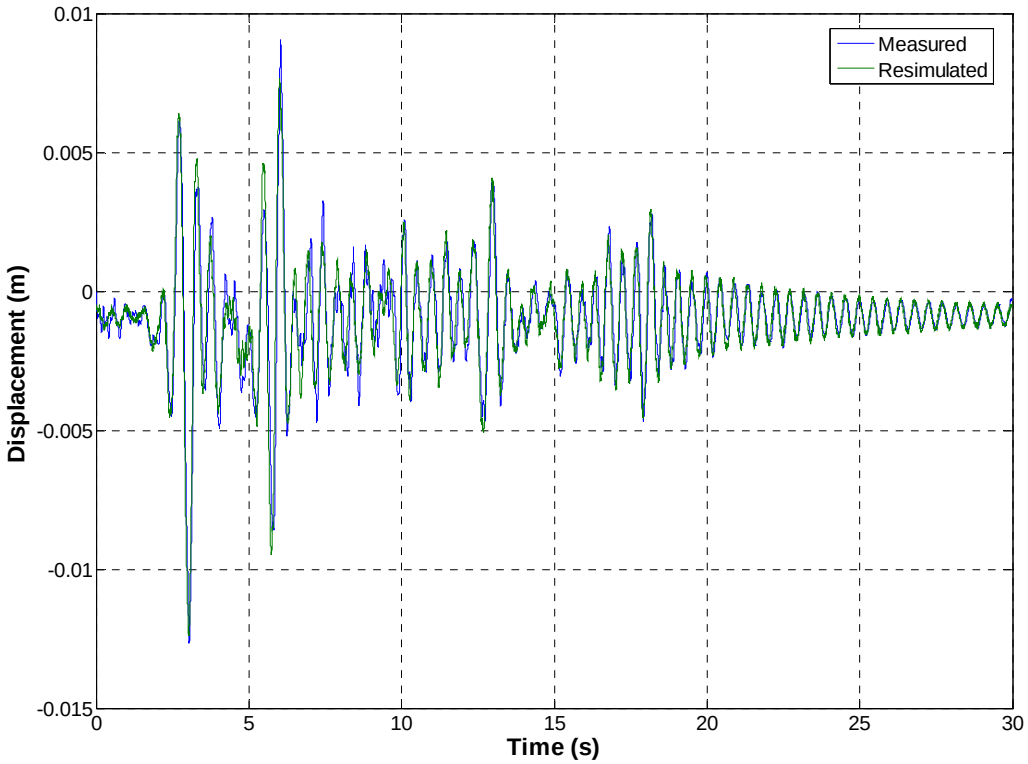

Figure 6.7: Resimulation for real frame structure with non-linear baseline model, $\Delta t_Z = 0.1$

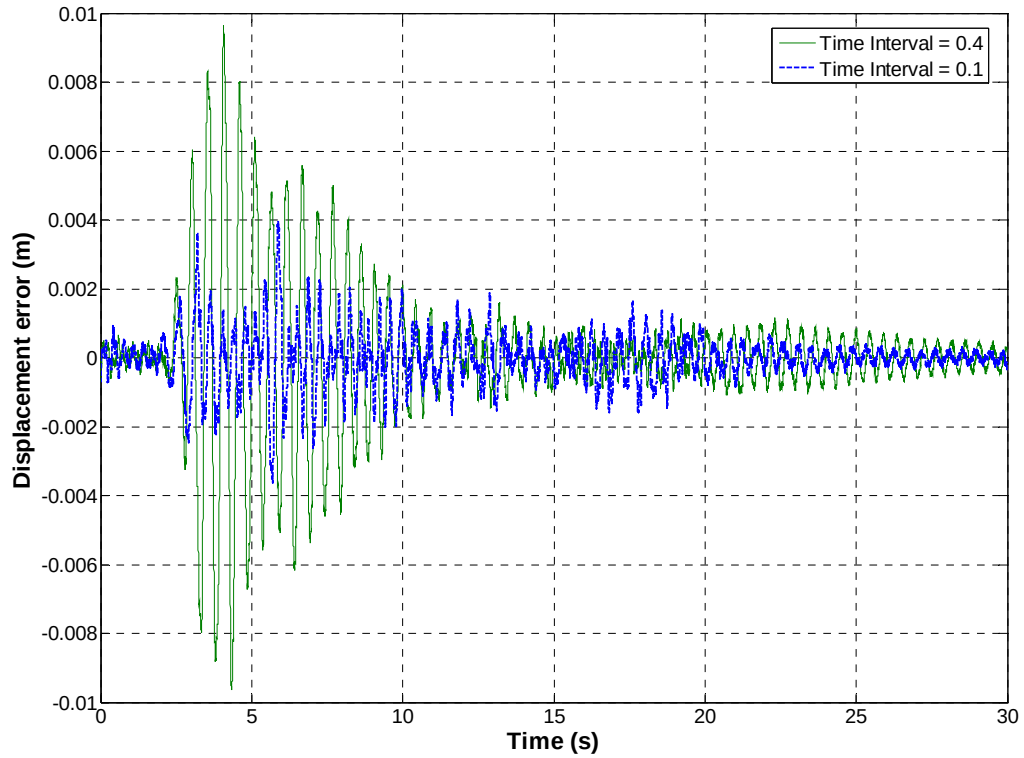

Figure 6.8: Comparison of errors in resimulated first storey displacements

The linear structural response of the four-storey steel frame was resimulated using the estimated structural matrices and the stiffnesses identified in Section 6.2.2. Figure 6.9 shows a comparison of the measured and resimulated displacements of the first storey using $\Delta t_{kp} = 0.5$ s. As expected, the model is incapable of reproducing non-linearity and permanent offset in the displacement response and the resimulated displacement is markedly different from the measured displacement. These limitations reinforce the notion that the linear baseline model is only appropriate for making general assessments of structural damage and highlight the advantage of using the non-linear Bouc-Wen baseline when identifying this kind of ductile structure.

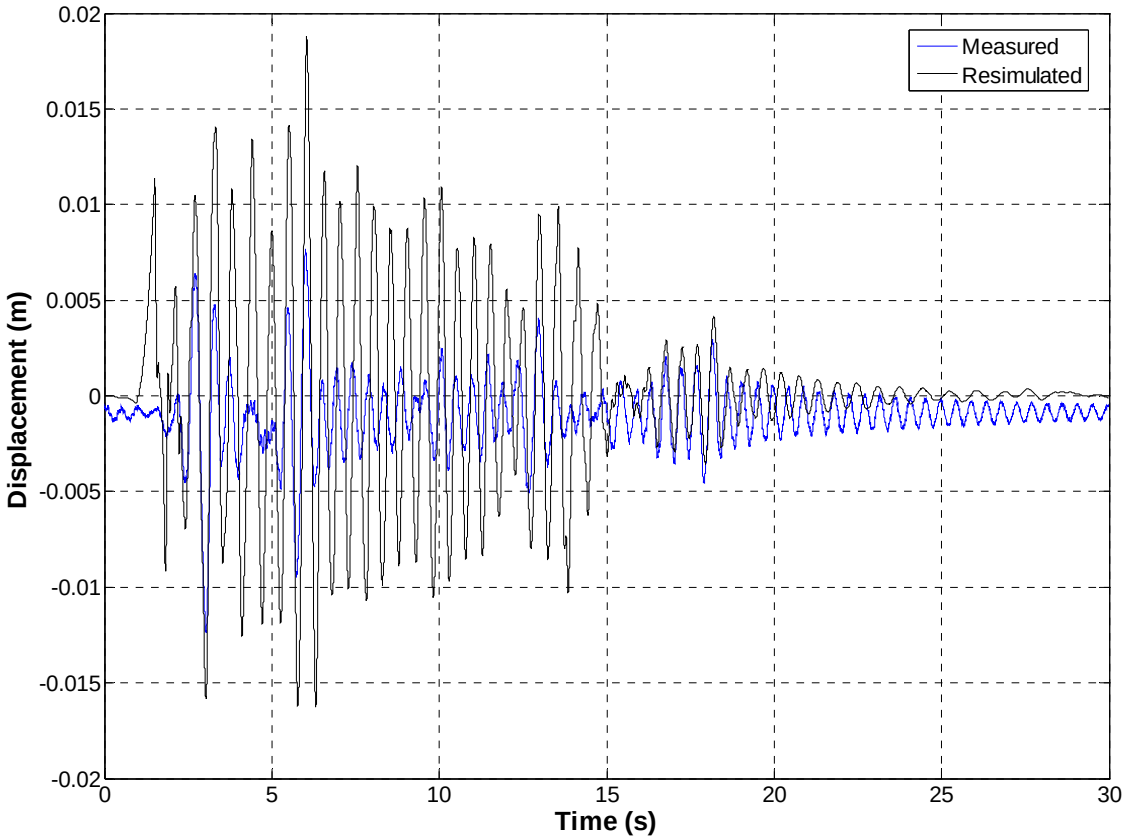

Figure 6.9: Resimulation for real frame using linear baseline model, $\Delta t_{kp} = 0.5$

6.2 Two Storey Hybrid Rocking Structure

6.2.1 Non-Linear Baseline Model

The stiffness and permanent displacement of the two-storey rocking structure were identified using simulated acceleration data sampled at 200 Hz and displacement data sampled infrequently at 10Hz. Both acceleration and displacement data had 10% uniformly distributed noise applied and the mass and damping matrices from Equations (4.6) and (4.7) were used in identification. Rocking structures behave much differently to frame structures during earthquake excitation. While a frame structure exhibits minimal changes in stiffness and significant permanent displacement, the re-centering behaviour of the rocking structure caused by post-tensioned members results in minimal permanent displacement accompanied by significant changes in stiffness as the structure switches between rocking, hybrid rocking-elastic and elastic regimes of motion. Potentially, the Bouc-Wen model is then a less suitable choice for a baseline model since it allows for permanent displacement when only a minimal amount occurs in reality.

Figures 6.10-6.13 show the identified stiffness and permanent displacement for both storeys where stiffness was fitted at 0.6s intervals and permanent displacement was fitted at 0.15s intervals. The changes in stiffness between response regimes are evident in Figures 6.10 and 6.12. However, at first look, it appears that the algorithm is identifying a false permanent displacement during periods of rocking, as can be seen in Figures 6.11 and 6.13.

The different response regimes exhibited by the structure are identified in Figures 6.10 and 6.11, where (1) indicates the linear regime, (2) indicates the rocking regime and (3) indicates the hybrid linear/rocking regime. The structure is responding elastically over the first 0.6s, as characterized by a high identified stiffness and no identified permanent displacement. The

structure is operating in a rocking regime from 0.6s to 3.6s which corresponds to a low identified stiffness and a large oscillating permanent displacement to account for the significant drop in stiffness. The hybrid rocking elastic/regime, which the structure exhibits from 3.6s to 10s, is characterized by a varying identified stiffness and an oscillating permanent displacement smaller than that identified during the pure rocking phase. After 10s the structure returns to a purely elastic regime. However, as the response is attenuated towards the end of the record, measured quantities are dominated by noise resulting in an increasing loss of accuracy in the identified parameters.

During the elastic response regime, the stiffnesses of both storeys are identified to within 10% of those estimated by Chase et al (2004b). However, identified stiffnesses during rocking and hybrid rocking/elastic periods differ significantly from the estimated rocking stiffness. This error occurs because the structure never enters a purely rocking regime. More specifically, a minimum linear displacement is required before the post-tensioned joints will allow rocking behaviour, meaning that the identified stiffness during these periods lies between the initial linear stiffness and the true rocking stiffness.

More importantly, the Bouc-Wen model sees the large drop in stiffness when rocking as oscillatory permanent displacements, rather than large changes in a (largely) linear system response. Hence it would appear that a purely linear time-varying model might do better in this situation.

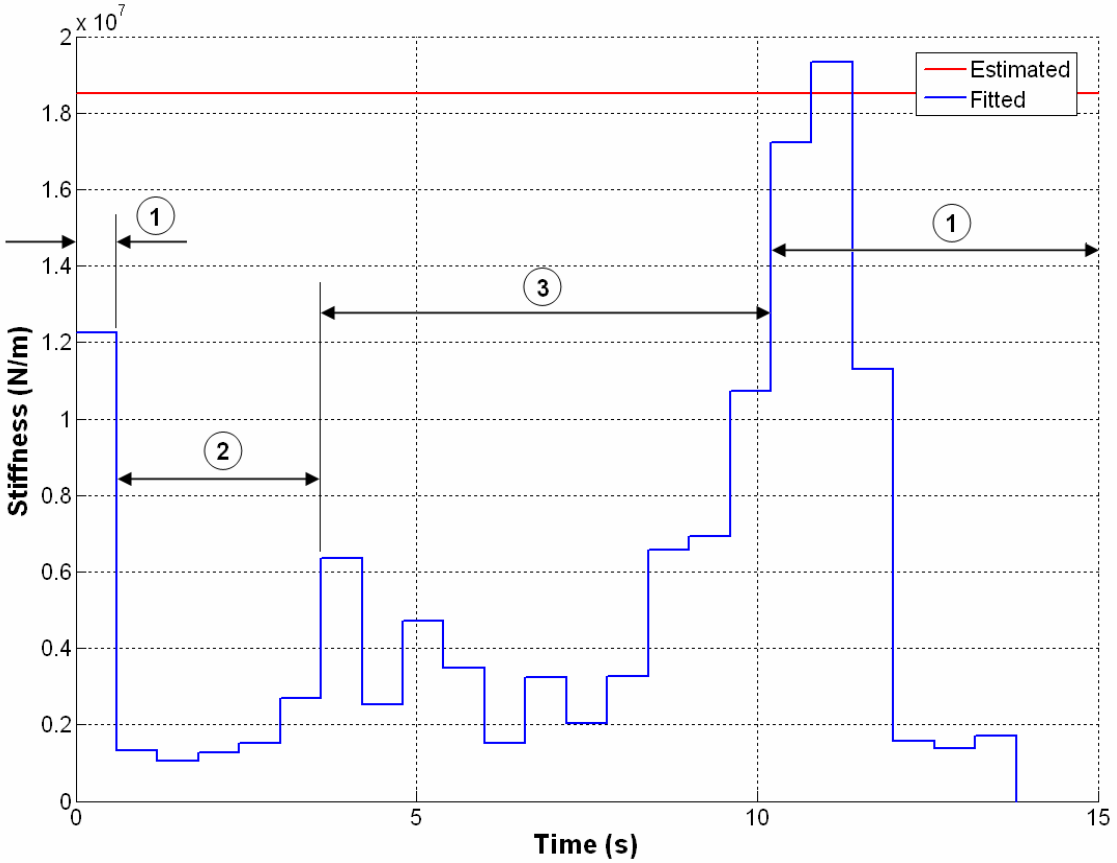

Figure 6.10: Bottom storey stiffness comparison for rocking structure using non-linear baseline model

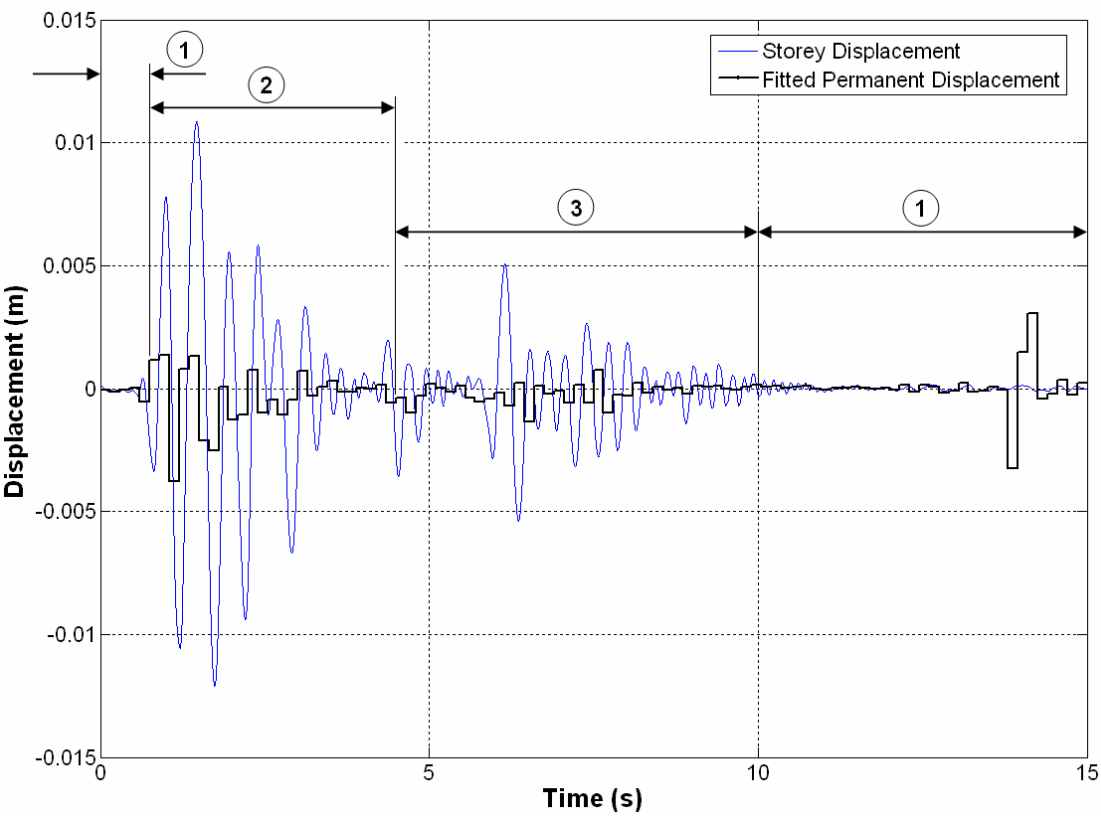

Figure 6.11: Bottom storey displacement overlay for rocking structure using non-linear baseline model

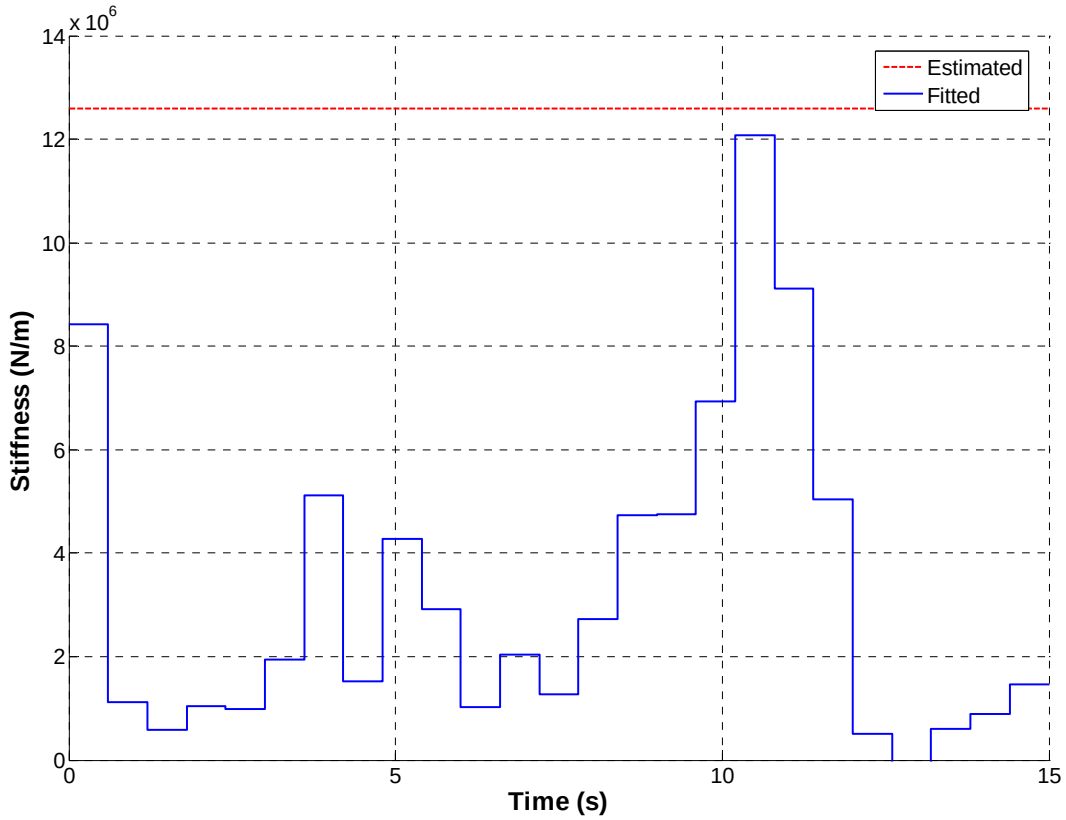

Figure 6.12: Top storey stiffness comparison for rocking structure using non-linear baseline model

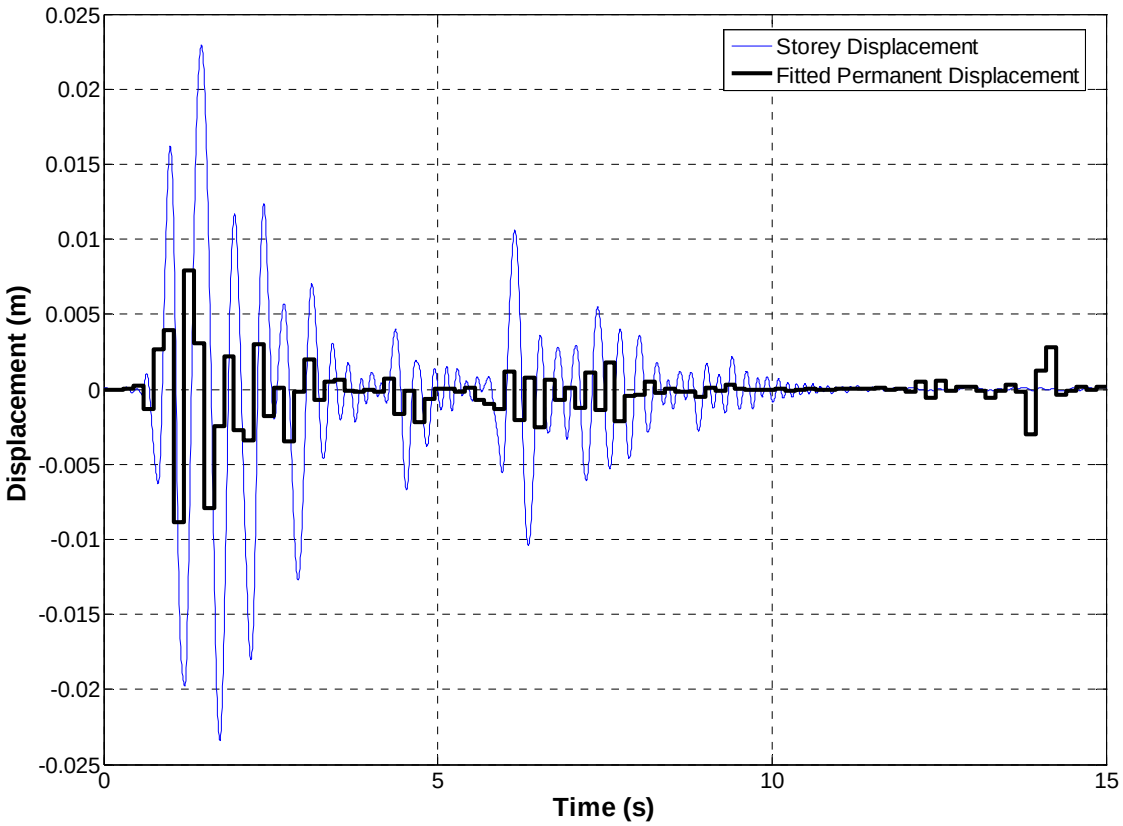

Figure 6.13: Bottom storey displacement overlay for rocking structure using non-linear baseline model

Table 6.6 shows the mean percentage difference when compared to the zero noise scenario for each response regime. Table 6.7 shows the same results where the fitting period Δt_Z is reduced to 0.02s. The results in Tables 6.6 and 6.7 were calculated using identified parameters from 20 simulations where both displacement and acceleration were subject to 10% uniformly distributed noise.

Table 6.6: Mean effect of noise on identified stiffness values where $\Delta t_Z = 0.15$

Storey	**1**	**2**
Mean Stiffness Change - Regime 1	-23.5642	-32.5497
Mean Stiffness Change - Regime 2	-0.2142	0.0093
Mean Stiffness Change - Regime 3	-0.5038	-1.4741

Table 6.7: Mean effect of noise on identified stiffness values where $\Delta t_Z = 0.02$

Storey	**1**	**2**
Mean Stiffness Change - Regime 1	-7.468	-12.0116
Mean Stiffness Change - Regime 2	0.4589	0.2959
Mean Stiffness Change - Regime 3	0.055	0.055

The results in Tables 6.6 and 6.7 show that noise-induced errors are not significant and do not affect the algorithm's ability to identify the existence, location or magnitude of structural damage during periods of appreciable response. The relatively high value shown in Table 6.6 for the second storey during regime 1 (the linear regime) is a result of noise dominating the insignificant response during these periods. This is avoided by reducing Δt_Z which, as demonstrated by Table 6.7, clearly decreases the algorithms' noise sensitivity dramatically. For this type of structure, the permanent displacement effectively acts as a filter, absorbing noise and non-linearity to allow a more accurate identification of stiffness. Reducing the period over which permanent displacement is identified allows the parameter $\Delta Z^{(w)}(t)$ more freedom, allowing more effective separation of the linear and non-linear components of response.

6.2.2 Linear Baseline Model

Figures 6.14 and 6.15 show comparisons of the identified rocking structure stiffnesses and the stiffnesses estimated by Chase et al, (2004b), where stiffness is fitted at 0.5s intervals. Figures 6.14 and 6.15 show that using a linear baseline model, the algorithm identifies the structures regimes of motion in a similar fashion to the non-linear baseline model as discussed in Section 6.2.1. However, the identified stiffnesses are much less accurate than those shown in Figures 6.12 and 6.14. This result is unexpected since the linear baseline model appears more appropriate to the rocking structure which exhibits minimal permanent displacement. But another interpretation of the $\Delta Z_w(t)$ parameter in Equation (3.23) is that it accounts for non-linear dynamics not captured by the linear model. Thus for the rocking structure, rather than calling $\Delta Z_w(t)$ permanent displacement, it should be interpreted as the deviation from the linear baseline model. This extra parameter can hence assist in removing part of the non-linear response from the linear stiffness parameter. While $\Delta Z_w(t)$ helps filter non-linear dynamics and noise from the linear stiffness, the linear baseline model lumps all non-linear dynamics and noise into the single time-varying stiffness parameter. This explains why there is significantly greater underestimation in Figures 6.14 and 6.15.

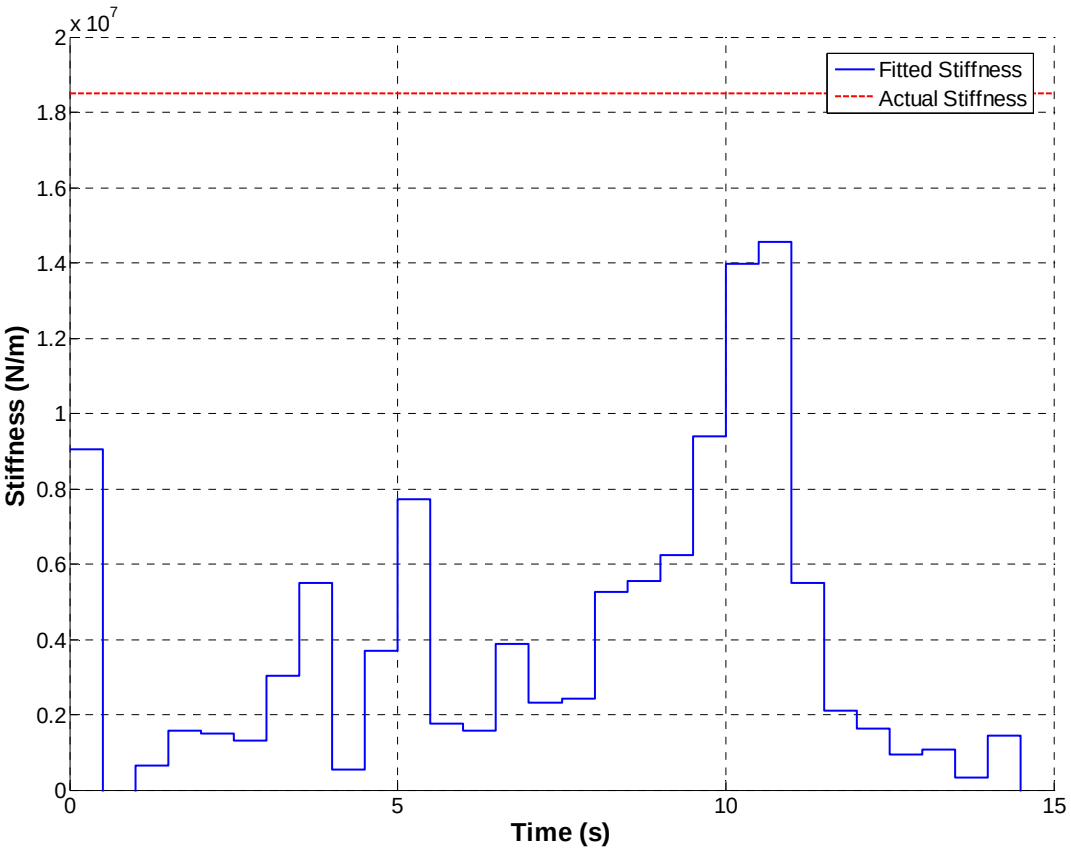

Figure 6.14: Stiffness comparison for first storey of rocking structure using linear baseline model

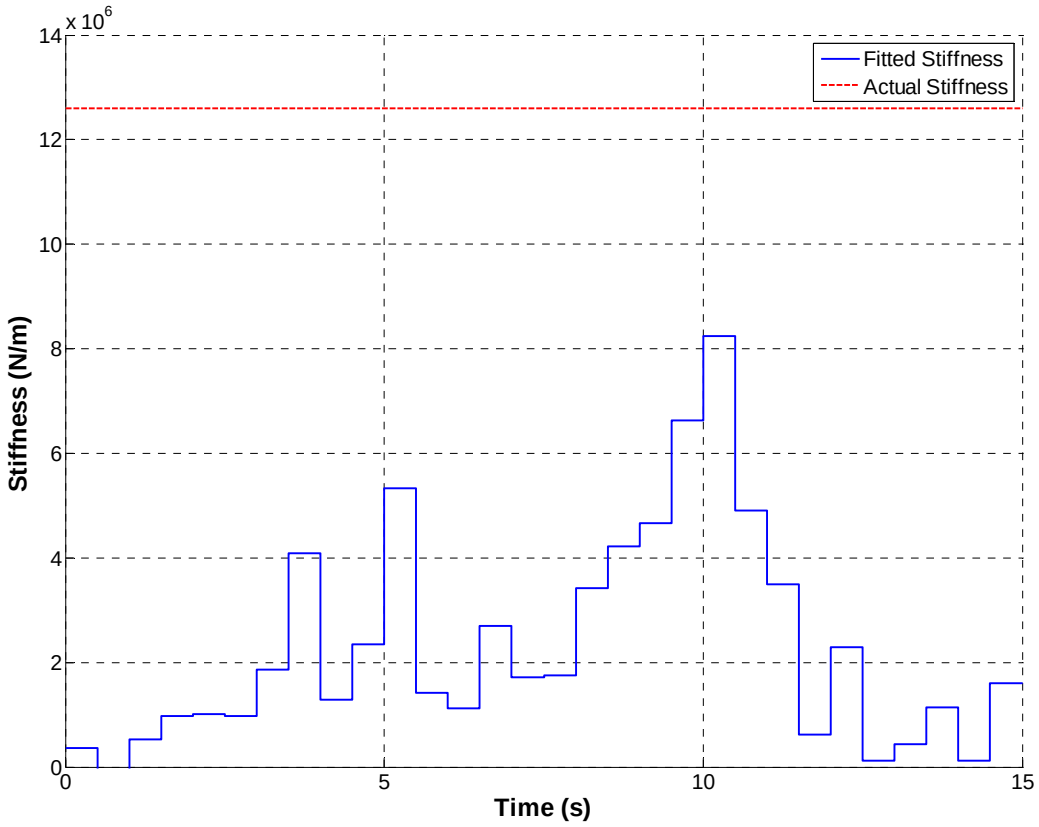

Figure 6.15: Stiffness comparison for second storey of rocking structure using linear baseline model

Table 6.8 shows the mean change in stiffness due to noise as a percentage of the zero noise identified stiffness for each regime. The results in Table 6.8 were calculated using identified parameters from 20 simulations where both displacement and acceleration were subject to 10% uniformly distributed noise.

Table 6.8: Mean effect of noise on identified stiffness values using linear baseline model

Storey	**1**	**2**
Mean Stiffness Change - Regime 1	-41.6133	-50.9734
Mean Stiffness Change - Regime 2	-0.1185	-0.0706
Mean Stiffness Change - Regime 3	-1.2135	-1.3088

Comparison of the results in Table 6.8 to the results in Tables 6.6 and 6.7 reinforces the relative inability of the linear baseline model to accurately identify stiffness in a robust manner. In particular, stiffness values identified using the linear model in the presence of noise exhibit more significant underestimation than values identified with the non-linear model.

6.2.3 Resimulations of Rocking Structure Response

Figure 6.16 shows a comparison of the measured and resimulated displacements for the first storey, using the parameters identified in Section 6.2.1 using the non-linear baseline model with $\Delta t_{kp} = 0.6\,\text{s}$ and $\Delta t_Z = 0.15\,\text{s}$. Figure 6.17 shows the same comparison for the parameters fitted in Section 6.2.2 using the linear baseline model with $\Delta t_{kp} = 0.5\,\text{s}$. In both cases, the resimulated displacement matches the measured displacement adequately throughout the record, with some discrepancies during highly non-linear periods. Thus, the identified parameters correctly satisfy the differential Equations (2.1) and (2.13), validating the accuracy of the fitting method presented.

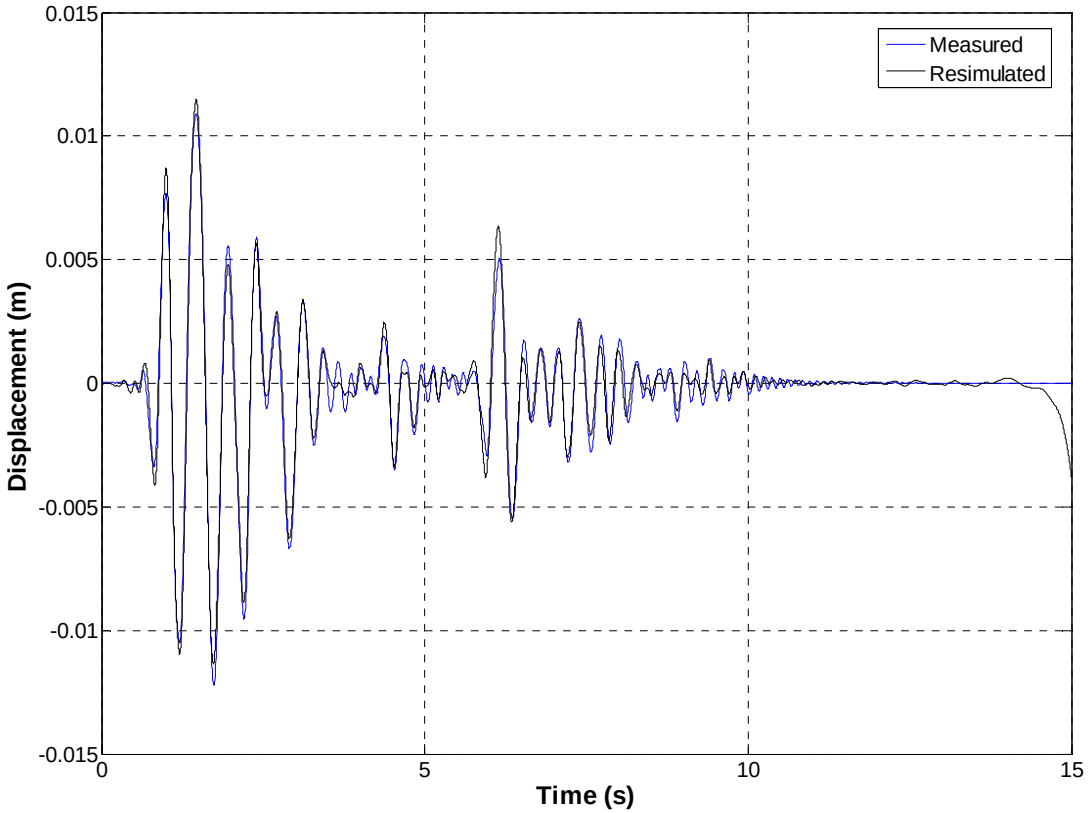

Figure 6.16: Resimulated displacement of first storey of rocking structure using non-linear baseline model

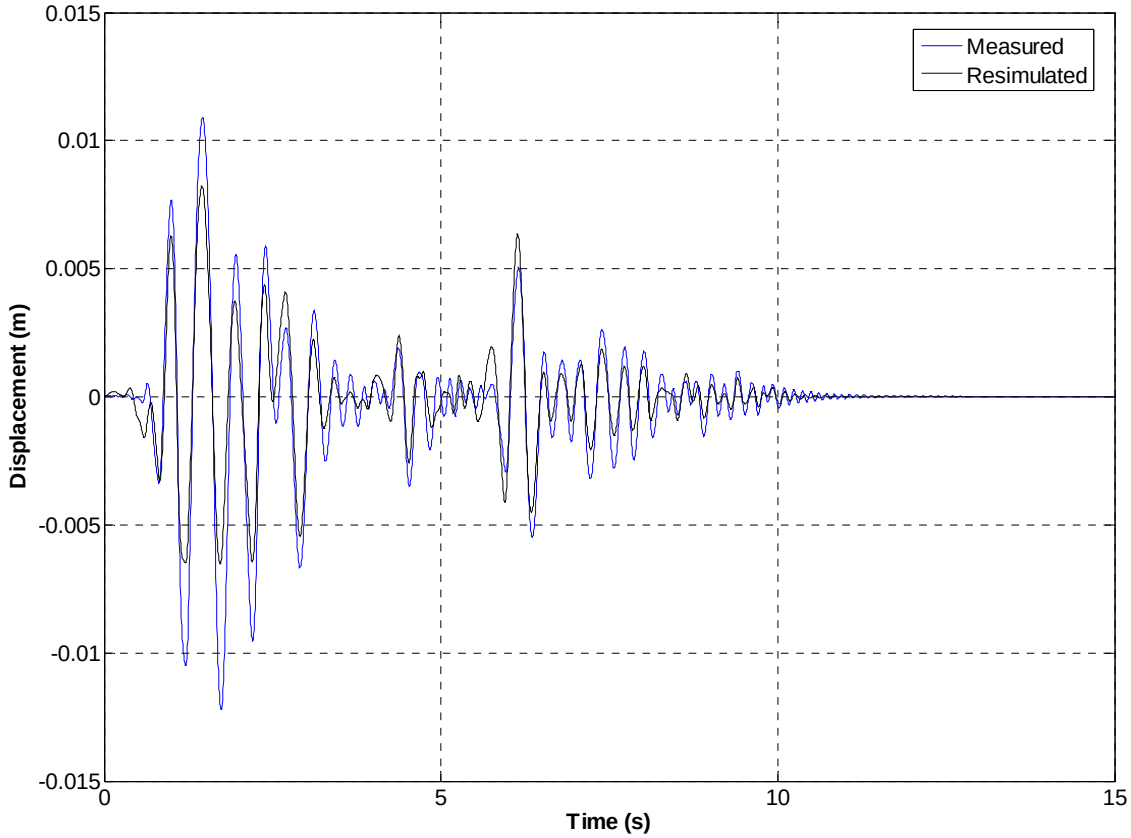

Figure 6.17: Resimulated displacements of first storey of rocking structure using linear baseline model

PART IV

CONCLUSIONS

CHAPTER 7 - DISCUSSION AND CONCLUSIONS

The algorithm accurately identifies time-varying profiles of stiffness and permanent displacement for each storey in multi degree of freedom structures. The adopted linear least squares approach ensures that the identification could be performed in real-time, therefore the algorithm would be particularly appropriate for real life structural health monitoring. Since the algorithm effectively decouples the fitting process into separate optimizations for each storey and time period, it is easily generalized to higher degrees of freedom and not susceptible to cumulative errors.

Using a non-linear baseline model, simulation shows that during periods of significant response, the stiffness of a particular storey of a shear building can be identified to within ±3.5% with 90% confidence using acceleration and displacement data subject to 10% uniformly distributed noise. Additionally, permanent displacement can be identified to within ±12.5% with 90% confidence using acceleration and displacement data subject to 10% uniformly distributed noise. However, the error is oscillatory in nature and the use of a three point moving average procedure improved the accuracy of permanent displacement identification to ±7.5% with 90% confidence

When applied to measured data from a four-storey steel frame structure, the algorithm identified a realistic permanent displacement and stiffness for each storey. Major yielding was identified to occur at the same times as peaks in structural response and final residual displacements were identified to within 1.3% of the true value for all storeys. During periods of significant response, stiffness values for the first three storeys were identified to within 3.5% of the mean values, while the stiffness of the fourth storey was identified to within 14% of the mean value. For a

first mode dominant structure such as this, the fourth storey is highly unlikely to suffer damage. Thus, the observed decrease in accuracy for the top storey is not highly significant when considering the damage state of the structure.

The identified mean stiffness values were within ±27% of values calculated by pushover analysis of a previously constructed computational model. This indicates that the identification process is yielding parameters of the correct order of magnitude, though any absolute assessment of accuracy is difficult. The discrepancies between fitted and predicted stiffness values are attributed to geometric non-linearity, flexural deformation not accounted for by the baseline model, and strain hardening of connections from repeated testing of the structure for class demonstrations.

Using the non-linear baseline model, the algorithm is shown to detect rocking, hybrid rocking-elastic and elastic regimes of motion for simulated data from a non-linear hybrid rocking structure whilst also identifying the initial linear stiffness to within 10% of that estimated by Chase et al (2004b). Acceleration and displacement data used from the rocking structure was subject to 10% uniformly distributed noise. This result highlights the ability of the algorithm to detect damage in structures which are radically different from the shear building arrangement for which it was originally designed, the robustness of the integral approach used and the versatility of the chosen non-linear Bouc-Wen baseline model.

Using a linear baseline model, structural damage is indicated purely by changes in stiffness. For the frame structure, the mean stiffness of each storey is identified to within 25% of the pushover analysis estimated values where significant response is observed. Also, the variation in stiffness accurately identifies each regime of motion for the rocking structure. These stiffness values are sufficient basis to make a judgment as to the presence and location of damage in the frame

structure and rocking structure. However, due to its inability to capture non-linear dynamics and its sensitivity to noise, the linear baseline model does not give an accurate indication as to the magnitude of permanent displacement induced in the frame structure by earthquake loading, nor does it accurately identify the linear stiffness of the rocking structure. In fact, the linear baseline model underestimates the top storey stiffness by 40%. Thus, the non-linear baseline model is more appropriate for use with both structures, despite the fact that the rocking structure undergoes minimal permanent deformation.

The algorithm's accuracy could be improved in the long term by sampling ground motion and structural response at higher frequency and by fitting a piecewise linear permanent displacement, rather than the piecewise constant approach presented which does not always accurately capture hysteretic response.

In summary this research provides a highly efficient and accurate method for identifying linear stiffness and permanent displacement in multi-story buildings under seismic loads. Identification of permanent deformation is novel in this field and can provide significant feedback to engineers as damage state and is also an ideal measure for the construction of probabilistic fragility functions. The overall method can be readily generalized to higher degrees of freedom and is computationally efficient. Specifically, using MATLAB, identification of all parameters over the entire record for the simulated four degree of freedom example given takes 2.55s on a 3GHz Pentium 4 machine. Thus, after coding in C, the method could be easily implemented in real-time.

CHAPTER 8 - FUTURE WORK

The algorithm presented in this research accurately identifies stiffness and permanent displacement in multi-storey shear type structures and identifies the various regimes of dynamic response exhibited by a hybrid rocking structure. However, the piecewise constant used in approximation of the permanent displacement parameter $\Delta Z(t)$, as defined in Chapter 2.3, does not always capture the entire hysteretic response. An improved algorithm which identified a linearly varying permanent displacement would potentially provide more accurate identification of stiffness and permanent deformation, particularly for highly non-linear structures. Additionally, an improved algorithm could use the fitted permanent displacement parameters to provide further information on the structures condition. Specifically, the yield point Y and shaping parameter n could be identified for each storey from Equation (2.16) using a least squares method.

In the long term, the general approach presented could be applied to different and more complex structural equations of motion in order to monitor structures such as bridges which differ significantly from the shear type format considered in this research. Also, identification of a more complex structural model would provide a more detailed description of a structures damage state.

REFERENCES

Analog Instruments, http://www.analog.com/

Allison, A. and Abbott, D. (2001). "Parameter Identification Using the Hilbert Transform," *SPIE Microelectronics and MEMS Conference*, Adelaide Australia, December 17-19.

Bernal, D. and Gunes, B. (2000) "Observer/Kalman and Subspace Identification of the UBC Benchmark Structural Model" *Proc. of the 14th ASCE Engineering Mechanics Conference*, Austin, Texas, May 21–24.

Bouc, R. 1967 "Forced Vibration of Mechanical Systems with Hysteresis" *Proceedings of the 4th Conference on Non-Linear Oscillation*. Prague, Czechoslovakia

Brownjohn, J. M. W., Moyo, P., Rizos, C. and Tjin, S. C. (2003) Practical issues in using novel sensors in SHM of civil infrastructure: problems and solutions in implementation of GPS and fibre optic sensors. *4th International Workshop on SHM*, pp. 499-506, Palo Alto, September.

Buyukozturk, O., Yu, T. (2003) "Structural Health Monitoring and Seismic Impact Assessment" *Proceedings of the 5th National Conference on Earthquake Engineering*, Istanbul, Turkey, May 26-30.

Caicedo, J. M., Dyke, S. J. and Johnson E. A. (2004). "Natural Excitation Technique and Eigensystem Realization Algorithm for Phase I of the IASC-ASCE Benchmark Problem: Simulated Data," *ASCE J. Engineering Mechanics*, Vol 130(1), pp. 49-60.

Caicedo, J. M., Dyke, S. J. and Johnson, E. A. (2000) "Health Monitoring Based on Component Transfer Functions" *Proceedings of the 2000 International Conference on Advances in Structural Dynamics*, Hong Kong, December 13-15.

Cheok, G.S., Lew, H.S. (1993). Model Precast Concrete Beam-to-Column Connections Subject to Cyclic Loading. *PCI Journal*, Vol. 38(4), pp 80-92.

Chase, J. G., Hwang, K. L., Barroso, L. R. and Mander, J. B. (2004) "A simple LMS-based approach to the structural health monitoring benchmark problem" *Journal of Earthquake Engineering and Structural Dynamics,* Vol. 34, No. 6, pp 575-594.

Chase, J.G., Spieth, H.A., Blome, C.F. and Mander, J.B. (2005) "LMS-based Structural Health Monitoring of a Non-linear Rocking Structure" *Journal of Earthquake Engineering and Structural Dynamics,* Vol. 34, No. 8, pp 909-930.

Ching, J. and Beck, J.L. (2004) "New Bayesian Model Updating Algorithm Applied to a Structural Health Monitoring Benchmark" *Journal of Structural Health Monitoring*, Vol. 3, No. 4, pp313-332

Doebling, S.W., Farrar, C.R., Prime, M.B., and Shevitz, D.W. (1996a) "Damage Identification and Health Monitoring of Structural and Mechanical Systems from Changes in Their Vibration Characteristics: a Literature Review" *Los Alamos National Laboratory*, Report LA-13070-MS.

Doherty, J. E. (1987) "Non-destructive Evaluation," Chapter 12 in *Handbook on Experimental Mechanics*, A. S. Kobayashi Edt., Society for Experimental Mechanics, Inc.

Farrar, C. R. and Doebling, S. W., "An Overview of Modal-Based Damage Identification Methods," EUROMECH 365 International Workshop: DAMAS 97, Structural Damage Assessment Using Advanced Signal Processing Procedures, Sheffield, U.K., June 1997

Johnson, E. A., Lam, H. F., Katafygiotis, L. S., and Beck, J. L. (2000) "A Benchmark Problem for Structural Health Monitoring and Damage Detection" *Proc. of the 14th ASCE Engineering Mechanics Conference*, Austin, Texas, May 21–24.

Kao, G. C. (1998) "Design and Shaking Table Tests of a Four-Storey Miniature Structure Built with Replaceable Plastic Hinges" *Masters of Civil Engineering Thesis,* University of Canterbury, Christchurch, New Zealand.

Kejewski-Correa, T. and Kochly, M. (2006) "Practical consideration for Global Positioning Systems in Urban Zones." *4th World Conference on Structural Control and Monitoring*, University of California, San Diego 11-13 July

Lam, H. F., Katafygiotis, L. S., and Mickleborough, N. C. (2004). "Application of a Statistical Model Updating Approach on Phase I of the IASC-ASCE Structural Health Monitoring Benchmark Study," *ASCE Journal of Engineering Mechanics*, Vol 130(1), pp. 34-48.

Leng, J.S., Winter, D., Barnes, R.A., Mays, C. and Fernando, G.F. (2004). "Structural health monitoring of smart civil structures using fibre optic sensors" *Proceedings of the 3rd International Conference on Experimental Mechanics*, Singapore, 29 November-1 December

Loh, C.-H., Lin, C.-Y., and Huang, C.-C. (2000) "Time Domain Identification of Frames under Earthquake Loadings" *ASCE Journal of Engineering Mechanics,* Vol.126, No.7, pp 693-703.

Lus, H. and Betti, R. (2000) "Damage Identification in Linear Structural Systems" *Proceedings of the 14th ASCE Engineering Mechanics Conference*, Austin, Texas, May 21–24.

Lus, H., Betti, R., Yu, J. and De Angelis, M. (2004). "Investigation of a System Identification Methodology in the Context of the ASCE Benchmark Problem," *ASCE J. Engineering Mechanics*, Vol 130(1), pp. 71-84.

Murahidy A.G., Carr A.J., Spieth H.A., Mander J.B., Bull D.K. (2004). "Design, Construction and Dynamic Testing of Post-tensioned Precast Reinforced Concrete Frame Building with Rocking Beam Column Connections and ADAS Elements", *Proc. NZSEE Conference*, Rotorua, New Zealand, March 19-21.

Nakaki S.D., Stanton J.F., Sritharan S. (1999). "An Overview of the PRESSS five-Story pre-cast test building," *PCI Journal*, Vol. 44(2), pp. 26-39.

Priestley, M.J.N. and MacRae, G.A. (1996). "Seismic Tests of Pre-cast Beam-to-Column Joint Sub-assemblages with Un-bonded Tendons", *PCI Journal*, Vol. 41(1), pp. 64-80.

Priestley, M.J.N., Sritharan, S., Conley, J.R., Pampanin, S. (1999). "Preliminary Results and Conclusions from the PRESSS Five-Storey Pre-cast Concrete Test Building," *PCI Journal*, Vol. 44(6), pp 43-67.

Sato, T. and Qi, K. (1998) "Adaptive H_∞ Filter: Its Application to Structural Identification" *ASCE Journal of Engineering Mechanics,* Vol.124, No.11, pp 1233-1240.

Stone, W.C., Cheok, G.S., Stanton, J.F. (1995). Performance of Hybrid Moment-Resisting Pre-cast Beam-Column Concrete Connections Subjected to Cyclic Loading*, ACI Journal,* Vol. 91(2), pp. 229.

Wen, Y.-K. (1976). "Method for Random Vibration of Hysteretic Systems." *Journal of Engineering Mechanics Division*, ASCE, Vol.102 No.2, pp 249-263.

Yang, J. N., Lei, Y. Lin, S. and Huang, N. (2004). "Hilbert-Huang Based Approach for Structural Damage Detection," *ASCE Journal of Engineering Mechanics*¸ Vol 130 No. 1, pp. 85-95.

Yuen, K-V, Siu, K. A. and Beck, J. L. (2004). "Two-Stage Structural Health Monitoring Approach for Phase I Benchmark Studies," *ASCE J. Engineering Mechanics*¸ Vol 130(1), pp. 16-33.

APPENDICES A MATLAB CODE

A1: Functions for Simulating Response of Bi-Linear Elastic System

Simulator.m

```
%Determine bi-linear 1DOF system response for given earthquake record -
%Ishan Singh-Levett

clear all
clc
close all

mass=1;             %Structural mass (kg)
alpha=0.31455;      %alpha = zi * wn to provide 5% damping at 1Hz
c=2*alpha*mass;     %damping coefficient

qsmooth=0;
%0 smooth acceleration before simulation, 1 for no smoothing
quake=2;            %1 For Pacoima, 2 For El Centro, 3 For Kobe

[accel,name,inte]=select(qsmooth,quake);
%Genererate Acceleration Vector From Record

vary=menu('Pre-Determined Stiffness Variation or Standard Yielding?','Pre-
Determined','Standard');

%Choosing pre-determined allows resimulation from previously fitted
%parameters

if vary==1
    RK=csvread('srec.txt');
    KK=RK(2:length(RK));
    fp=RK(1);
    quakespan=fp*length(KK);
    clear RK
elseif vary==2
    yieldpoint=0.1;
    k=39.577*mass;                          %omega squared is 39.577
    K=[k 0.1*k];                            %Bilinear stiffness
characteristics
    quakespan=inte*(length(accel)-1);       %length of earthquake
end

tspan=[0 quakespan];
%timespan of earthquake
Conds0=[0 0];
%initial conditions of displacement and velocity

if vary==1
%Simulate response using newmark beta method and known stiffness profile
[T,Conds,bodyacc,P,divisor,noise,ntype]=PreBeta(Conds0,tspan,mass,accel,c,int
e,fp,KK);
elseif vary==2
%Simulate response using newmark beta with bi-linear yield characteristics
[T,Conds,bodyacc,P,divisor,noise,ntype]=NBeta(Conds0,tspan,mass,yieldpoint,ac
cel,K,c,inte);
end
```

```
if divisor~=1
%This loop extends the ground acceleration vector for plotting
    accel2=zeros(length(T),1);
    for count=1:length(accel2)
        accel2(count)=accel(ceil(count/divisor));
    end
    accel=accel2;
elseif divisor==1
    accel=accel';
end

if noise==1
%Determine noise magnitude from chosen value
    n=0;
elseif noise==2
    n=0.05;
elseif noise==3
    n=0.1;
elseif noise==4
    n=0.2;
end

for i=1:length(accel)
%This loop adds chosen type and magnitude noise to the accel vector
    if ntype==1
    accel(i)=accel(i)+randn*accel(i)*n/3;
    elseif ntype==2
    accel(i)=accel(i)+2*(rand-0.5)*accel(i)*n;
    elseif ntype==3
    accel(i)=accel(i)+mean(abs(accel))*n*2*(rand-0.5);
    end
end

%PLOT STUCTURAL RESPONSE=================================================
figure(1);

if vary==1
title1=strcat('\bfDisplacement');
title3=strcat('\bfGround Acceleration');
title2=strcat('\bfVelocity');
title4=strcat('\bfBody Acceleration');
elseif vary==2
title1=strcat('\bfDisplacement, Yieldpoint= ',num2str(yieldpoint));
title3=strcat('\bfGround Acceleration, Yieldpoint= ',num2str(yieldpoint));
title2=strcat('\bfVelocity, Yieldpoint= ',num2str(yieldpoint));
title4=strcat('\bfBody Acceleration, Yieldpoint= ',num2str(yieldpoint));
end

subplot(2,1,1); plot(T,Conds(:,1)); title(title1); grid on; xlim([0
quakespan]);
subplot(2,1,2); plot(T,Conds(:,2)); title(title2); grid on;  xlim([0
quakespan]);

figure(2);

subplot(2,1,1); plot(T,bodyacc); title(title4); grid on; xlim([0 quakespan]);
subplot(2,1,2); plot(T,accel); title(title3); grid on;  xlim([0 quakespan]);
```

```
if vary==2
%Save data in ASCII format for fitting
number=num2str(yieldpoint*1000);
fact=strcat('Body',name,number);
save(fact,'bodyacc','-ASCII');
fact=strcat('Ground',name,number);
save(fact,'accel','-ASCII');
end


if vary==2
%Calculate real stiffnes variation for comparison to identified parameters
DPP=Conds(:,1);
KK=[];
for i=1:length(DPP)
    ds=abs(DPP(i));
if ds>yieldpoint
KK(i)=(yieldpoint*K(1)+(ds-yieldpoint)*K(2))/ds;
else
KK(i)=K(1);
end
end

figure(4)
plot(T,KK);
title('\bfReal Stiffness Variation');
ylim([0 max(KK)*1.3]);
xlabel('\bfTime');
ylabel('\bfStiffness');
grid on
end
```

select.m

```
%Select quake record - Ishan Singh-Levett 2005

function [accel,name,inte]=select(qsmooth,quake)

if quake==1         %if the earthquake is pacoima dam

range=[17 1 800 10];
%coordinates in earthquake file to read this will have to be changed with
changing format
inte=0.005;
%timestep of earthquake record, also dependent on format etc.
scaler=9810;
%earthquake scale in terms of m/s^2, dependent on format etc.
xddg=dlmread('PACOM71A.eqc','',range);
name='pacoima';
long=length(xddg);
width=min(size(xddg));
X=[];
for i=1:long
    for j=1:width
        X=[X xddg(i,j)];
    end
end

X=X/scaler;
```

```
%=========================================================================
elseif quake==2             %if the earthquake is el centro
%=========================================================================

range=[17 1 1090 10];
%coordinates in earthquake file to read this will have to be changed with
changing format
inte=0.005;
%timestep of earthquake record, also dependent on format etc.
scaler=9810;
%earthquake scale in terms of m/s^2, dependent on format etc.

xddg=dlmread('ELCEN40A.eqc','',range);

name='elcentro';

long=length(xddg);
width=min(size(xddg));

X=[];
for i=1:long
    for j=1:width
        X=[X xddg(i,j)];
    end
end

X=X/scaler;


%=========================================================================
elseif quake==3             %if the earthquake is kobe
%=========================================================================

range=[17 0 220 9];
%coordinates in earthquake file to read this will have to be changed with
changing format
inte=0.02;
%timestep of earthquake record, also dependent on format etc.
scaler=100;
%earthquake scale in terms of m/s^2, dependent on format etc.

xddg=dlmread('KOBE95NS.eqn','',range);

name='kobe';

long=length(xddg);
width=min(size(xddg));

X=[];
for i=1:long
    for j=1:width
        X=[X xddg(i,j)];
    end
end

X=X/scaler;


end
```

```
%-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

if qsmooth==0

accel=movav(X,3);
%Smooth acceleration input before simulation
accel=movav(accel,3);
accel=movav(accel,3);
accel=movav(accel,3);
accel=movav(accel,3);
accel=movav(accel,3);
accel=movav(accel,3);
accel=movav(accel,3);
accel=movav(accel,3);
accel=movav(accel,3);
accel=movav(accel,3);
accel=movav(accel,3);
else
accel=X;
%No Smoothing
end
```

PreBeta.m

```
%Newmark Beta Integration with pre-determined stiffness variation
%Ishan Singh-Levett

function [T,Conds,bodyacc,P,divisor,noise,ntype] =
PreBeta(Conds0,tspan,mass,accel,c,inte,fp,KK)

h=0.001;
%Give data at 1KHz
divisor=inte/h;
%divisor between integration time step and desired sampling rate
P=[];
%right hand side vector for newmark beta
P(1)=-accel(1)*mass;
Conds(1,1)=Conds0(1);
Conds(1,2)=Conds0(2);
%Initial conditions of displacement and velocity
uddnext=-accel(1);
%Initial acceleration
T=0:h:tspan(2);
%Time steps for integration
bodyacc=zeros(length(T),1);
%Structural acceleration vector
bodyacc(1)=-accel(1);
Conds=zeros(length(T),2);
%Matrix to store displacement and velocity

ntype=menu('What Type of Noise?','Normal','Even','Even constant magnitude');

noise=menu('How much noise?','None','5%','10%','20%');

for count=1:(tspan(2)/h)

        uddn=uddnext;
```

```
        tt=T(count);
        index=floor(tt/fp)+1;
        %Calculate index for pre-determined stiffness vector
        stiffness=KK(index);

        P(count+1)=-accel(ceil((count+1)/divisor))*mass;
        %Calculate right hand side force vector

        D=Conds(count,1);
        V=Conds(count,2);


Conds(count+1,1)=(P(count+1)+mass*(4/h^2*D+4/h*V+uddn)+c*(2/h*D+V))/(4*mass/h
^2+2*c/h+stiffness);
        %Calculate displacement at next time step
        Conds(count+1,2)=-V+2/h*(Conds(count+1,1)-D);
        %Calculate velocity at next time step
        uddnext=4/h^2*(Conds(count+1,1)-D-h*V)-uddn;
        %Calculate acceleration at next time step

        if ntype==1
        %Add noise to simulated structural and ground acceleration data
        if noise==1
        bodyacc(count+1)=uddnext;
        elseif noise==2
            bodyacc(count+1)=uddnext+randn*uddnext*0.05/3;
        elseif noise==3
            bodyacc(count+1)=uddnext+randn*uddnext*0.1/3;
        elseif noise==4
            bodyacc(count+1)=uddnext+randn*uddnext*0.2/3;
        end

        elseif ntype==2

        if noise==1
                bodyacc(count+1)=uddnext;
        elseif noise==2
                bodyacc(count+1)=uddnext+2*(rand-0.5)*uddnext*0.05;
        elseif noise==3
                bodyacc(count+1)=uddnext+2*(rand-0.5)*uddnext*0.1;
        elseif noise==4
                bodyacc(count+1)=uddnext+2*(rand-0.5)*uddnext*0.2;
        end

        elseif ntype==3
        bodyacc(count+1)=uddnext;
        end

end

if ntype==3
    %Add noise to data from final time step
    fac=max(bodyacc);
    for i=1:length(bodyacc);

        if noise==1
            bodyacc(i)=bodyacc(i)+2*(rand-0.5)*fac*0.05;
        elseif noise==2
            bodyacc(i)=bodyacc(i)+2*(rand-0.5)*fac*0.1;
        elseif noise==3
```

```
            bodyacc(i)=bodyacc(i)+2*(rand-0.5)*fac*0.2;
        end

    end
end
```

NBeta.m

```
%Newmark Beta Integrator
%Ishan Singh-Levett 2005

function [T,Conds,bodyacc,P,divisor,noise,ntype] =
NBeta(Conds0,tspan,mass,yieldpoint,accel,K,c,inte)
h=0.001;
%Give data at 1KHz
divisor=inte/h;
%divisor between integration time step and desired sampling rate
P=[];
%right hand side vector for newmark beta
P(1)=-accel(1)*mass;
Conds(1,1)=Conds0(1);
Conds(1,2)=Conds0(2);
%Initial conditions of displacement and velocity
uddnext=-accel(1);
%Initial acceleration
T=0:h:tspan(2);
%Time steps for integration
bodyacc=zeros(length(T),1);
%Structural acceleration vector
bodyacc(1)=-accel(1);
Conds=zeros(length(T),2);
%Matrix to store displacement and velocity

ntype=menu('What Type of Noise?','Normal','Even','Even constant magnitude');

noise=menu('How much noise?','None','5%','10%','20%');

for count=1:(tspan(2)/h)

    %First determine whether the displacement has exceeded the yield point
    yieldornot=abs(Conds(count,1));
    if yieldornot<yieldpoint
        stiffness=K(1);
        quan=0;
    else
        %Calculate effective linear stiffness during yield periods
        stiffness=K(2);
        if Conds(count,1)>0
            quan=yieldpoint*(K(2)-K(1));
        else
            quan=yieldpoint*(K(1)-K(2));
        end
    end

    uddn=uddnext;

    P(count+1)=-accel(ceil((count+1)/divisor))*mass;
    %Create right hand side force vector

    D=Conds(count,1);
```

```
    V=Conds(count,2);


Conds(count+1,1)=(P(count+1)+quan+mass*(4/h^2*D+4/h*V+uddn)+c*(2/h*D+V))/(4*m
ass/h^2+2*c/h+stiffness);
    %Calculate displacement at next time step
    Conds(count+1,2)=-V+2/h*(Conds(count+1,1)-D);
    %Calculate velocity at next time step
    uddnext=4/h^2*(Conds(count+1,1)-D-h*V)-uddn;
    %Calculate acceleration at next time step


    %Next add noise to acceleration data
    if ntype==1
        if noise==1
            bodyacc(count+1)=uddnext;
        elseif noise==2
            bodyacc(count+1)=uddnext+randn*uddnext*0.05/3;
        elseif noise==3
            bodyacc(count+1)=uddnext+randn*uddnext*0.1/3;
        elseif noise==4
            bodyacc(count+1)=uddnext+randn*uddnext*0.2/3;
        end

    elseif ntype==2

        if noise==1
            bodyacc(count+1)=uddnext;
        elseif noise==2
            bodyacc(count+1)=uddnext+2*(rand-0.5)*uddnext*0.05;
        elseif noise==3
            bodyacc(count+1)=uddnext+2*(rand-0.5)*uddnext*0.1;
        elseif noise==4
            bodyacc(count+1)=uddnext+2*(rand-0.5)*uddnext*0.2;
        end

    elseif ntype==3
        bodyacc(count+1)=uddnext;
    end

end

%Finally, add noise to data from final time step

if ntype==3
    fac=mean(abs(bodyacc));
    fac2=0.003*9.81;

    for i=1:length(bodyacc);

        if noise==2
            bodyacc(i)=bodyacc(i)+2*(rand-0.5)*fac*0.05;
        elseif noise==3
            bodyacc(i)=bodyacc(i)+2*(rand-0.5)*fac*0.1;
        elseif noise==4
            bodyacc(i)=bodyacc(i)+2*(rand-0.5)*fac*0.2;
        end
    end
end
```

ConstrainFit.m

```
%Identify stiffness for bi-linear elastic system
%Ishan Singh-Levett

clear all;clc;close all;

%First load simulated data
accel=load('Groundelcentro100')';
%Ground acceleration
accelerationfile='Bodyelcentro100';
%Structural acceleration
fprintf('\n')

inte=0.001;
%Sampling rate

bodyacc=load(accelerationfile)';
quakespan=inte*(length(accel)-1);
t=0:inte:quakespan;

%Known structural properties
mass=1;
alpha=0.31455;
c=2*alpha*mass;

%Matrices to store integrated parameters
v=[]; d=[];

fp=0.25;
%Stiffness fit Period, tune this to ensure accuracy
cp=1;
%Fit errors every n stiffness fit period

smoothkk=1;
%if 0 then it is smoothed after fitting, else it is not

ced=1;
%if this is 0, estimated velocities and displacements are calculated across
entire record, else not

oldv=0;oldd=0;
%Initial conditions of displacement and velocity
KK=[]; ER1=[]; ER2=[];
%Matrices to record stiffness and fitted error

mode='unset';

[KK,T]=retfit(mass,c,accel,bodyacc,inte,fp,oldd,oldv,smoothkk,KK,ER1,ER2,t,qu
akespan,cp,mode);
%Fit stiffnesses and errors using integral-based approach

%Now plot identified stiffness profile
figure(1)
for i=1:length(KK)
    a=[T(i) T(i)-fp];
    b=[KK(i) KK(i)];
    line(a,b)
    if i<length(KK)
```

```
    a=[T(i) T(i)];
    b=[KK(i) KK(i+1)];
    line(a,b,'LineWidth',2)
    end
end
title('\bfPiecewise Stiffness Variation');
ylim([0 max(KK)*1.5]);
xlabel('\bfTime (s)');
ylabel('\bfStiffness (N/m)');
grid on

csvwrite('srec.txt',[fp KK]);
%Save stiffnesses for resimulation
end
```

retfit.m

```
%Function to fit stiffness and error over entire given record
%Ishan Singh-Levett
function
[KK,T]=retfit2(mass,c,accel,bodyacc,inte,fp,oldd,oldv,smoothkk,KK,ER1,ER2,t,q
uakespan,cp,mode)

v=cumtrapz(t,bodyacc)+oldv;
%integrate accelerations using trapezium rule
d=cumtrapz(t,v)+oldd;
%integrate velocities with trapezium rule

T=[];
%Vector to hold time values

period=cp*fp;
%Fitting period

dsint=int32(period/inte);
%TIme length of individual fitting matrices and vectors

oldtime=0;

for i=1:quakespan/period

    ds=zeros(dsint,cp);
    %Create vector for left hand side of equation

    for j=1:cp
        time1=(i-1)*period+(j-1)*fp;
        time2=(i-1)*period+j*fp;
        ind1=int32(time1/inte+2);
        ind2=int32(time2/inte+1);
        ind3=int32(fp/inte)*(j-1)+1;
        ind4=int32(fp/inte)*j;
        ds(ind3:ind4,j)=d(ind1:ind2);
        %Fill LHS vector
        T=[T time2];
    end

    time1=period*i;
    ind1=int32(oldtime/inte+2);
    ind2=int32(time1/inte+1);
```

```
    oldtime=time1;

    kx=-mass*accel(ind1:ind2)-mass*bodyacc(ind1:ind2)-c*v(ind1:ind2);
    %Generate right hand side vector

    C=[ds,(t(ind1:ind2)'+c*d(ind1:ind2).^0'),d(ind1:ind2).^0'];dd=kx';
    %Assemble matrices for least squares solution

    XX=lsqlin(C,dd);
    %Perform least squares solution

    for k=1:cp
        KK=[KK XX(k)];
        %Record identfied stiffness value
    end

end

if smoothkk==0
    %Smooth identified stiffness
    KK=movav(KK,3);
end
```

movav.m

```
%Moving Average of any vector - Ishan Singh-Levett 2005

function [A]=movav(A,b)

if rem(b,2)==0
    error('Smoothing requires odd number of points');
end

c=int32((b-1)/2);

B=[];

for i=[1:c,length(A)-c+1:length(A)];
    B(i)=A(i);
end

for count=c+1:length(A)-c
    for loop=-c:c
    B(count)=B(count)+A(count+loop)/b;
    end
end

A=B;
```

Published with MATLAB®7.0

A2: Functions for Stiffness Identification of Elastic Systems

fitter.m

```
%N-DOF Linear Stiffness Fitting Algorithm - Ishan Singh-Levett May 2006

clear all
clc
close all

m=4;
%no of DOF
quake=2;
%1 For Pacoima, 2 For El Centro, 3 for Kobe, 4 for table,5 for simple model,
6 for complex model
y=0.04;
%yield point (only for file numbering purposes)
yield=num2str(y*1000);
dof=num2str(m);

%First, load ground and structural data

if quake==1
    name=strcat('pacoima',yield,'-',dof);
elseif quake==2
    name=strcat('elcentro',yield,'-',dof);
elseif quake==3
    name=strcat('kobe',yield,'-',dof);
end

mass=1;
%mass of each DOF
M=diag(mass*diag(ones(m,m)));
%Mass Matrix

PK=1.6131e+003*ones(4,1);
%Previously estimated stiffness for comparison

dname=strcat('Disp',name);
bname=strcat('Body',name);

Disp=load(dname);
%Generate realistic displacement measurement from this
bodyacc=load(bname);
%Structural acceleration

aname=strcat('Ground',name);
accel=load(aname);
%Ground acceleration

t1=cputime;
inte=0.001;
%Sampling rate

quakespan=inte*(length(bodyacc)-1);
%Time length of earthquake
t=0:inte:quakespan;
```

```
rt=10;
%displacement reporting interval


 C=2*[2.1799   -0.6279   -0.1125   -0.0545
   -0.6279    2.0674   -0.6824   -0.1670
   -0.1125   -0.6824    2.0128   -0.7949
   -0.0545   -0.1670   -0.7949    1.3850];
%Estimated damping matrix

[newv,newd,v,d,d2,md,smoothd]=adjust2(bodyacc,rt,quakespan,t,inte,Disp,m);
%Calculate estimated displacement response

[smoothv]=cdiff(smoothd,inte);
[smoothb]=cdiff(smoothv,inte);
%Numerical differentiation to find estimated velocity and acceleration

fp=2;
%Stiffness fitting period

alpha=0.1;
%Bi-linear factor estimate

D=diag(ones(m,1));
for j=2:m
    D(j,j-1)=-1;
end
%Co-ordinate transformation matrix from absolute to relative

RD=smoothd*D';
RV=smoothv*D';
RB=smoothb*D';
%Calculate relative displacement, velocity and accelerations

[KK,Ktime]=linear(fp,quakespan,accel,smoothb,smoothv,smoothd,inte,M,C,alpha,m
,RD);
%Use least squares procedure to identify stiffness
%Save parameters to ASCII file for resimulation
O=[fp];

t2=cputime;
fact='FStiff';
save(fact,'KK','-ASCII');
fact='Op';
save(fact,'O','-ASCII');

%Plot fitted parameters
KPlot(KK,Ktime,fp,m,PK,t,smoothd);
```

linear.m

```
% Least Squares fitting for N-DOF linear system - Ishan Singh-Levett

function[KK,Ktime]=boucwen(fp,quakespan,accel,bodyacc,newv,newd,inte,M,C,alph
a,m,RD,div);



T=0:fp:quakespan;
```

```
if max(T)<quakespan
    T=[T quakespan];
end
%Times at which stiffness is fitted

Time=0:inte:quakespan;
if max(Time)<quakespan
    Time=[Time quakespan];
end

DM=diag(M);
DC=diag(C);
%Separate coefficients of mass and damping matrices for fitting

KK=zeros(length(T)-1,m);

Ktime=[];

for i=1:length(T)-1

    time1=T(i);
    time2=T(i+1);
    ind1=int32(time1/inte+1);
    ind2=int32(time2/inte);

    %Generate indices for construction of least squares matrices

    Ktime=[Ktime time2];

    for k=1:m

        ls=zeros(ind2-ind1+1,1);
        rs=zeros(ind2-ind1+1,1);

        ls(1:ind2-ind1+1,1)=RD(ind1:ind2,k);
        %Generate LHS vector

        %Perform summation to create RHS vector

        mterm=-accel(ind1:ind2)*sum(DM(k:m));
        aterm=-bodyacc(ind1:ind2,k:m)*DM(k:m);
        dterm=-newv(ind1:ind2,k:m)*DC(k:m);

        rs=mterm+aterm+dterm;

        %Solve system using linear least squares
        b=lsqlin(ls,rs);
        KK(i,k)=b;
    end
end
```

Kplot.m

```
%Function to plot identified stiffness - Ishan Singh-Levett May 2006

function KPlot(KK,Ktime,fp,m,t,smoothd,PK);
```

```
for k=1:m

figure(k)

Ktimeforplot=[0 Ktime];

Ktp=zeros(length(Ktime)*2,1);
Krecord=zeros(length(Ktp),1);

Ktp(1)=0;
Krecord(1)=KK(1,k);

Ktp(end)=Ktime(end);
Krecord(end)=KK(end,k);

for z=1:length(Ktime)-1
    Ktp(2*z)=Ktime(z);
    Ktp(2*z+1)=Ktime(z);
    Krecord(2*z)=KK(z,k);
    Krecord(2*z+1)=KK(z+1,k);
end

hold on
plot(Ktp,Krecord,'Color','b','LineWidth',2);
plot(Ktimeforplot,PK(k)*ones(length(Ktime)+1,1),'LineStyle','--
','Color','r','LineWidth',2)

index=num2str(k);
title(strcat('\bfFitted Stiffness Variation - Story  ',index),'FontSize',20);
% ylim([min(Krecord)-0.5*max(Krecord) max(Krecord)*1.5]);
ylim([0 max(Krecord)*1.2]);
% legend('Actual Stiffness','Fitted Stiffness')
xlabel('\bfTime (s)','FontSize',20);
ylabel('\bfStiffness (N/m)','FontSize',20);
grid on
end
```

Published with MATLAB®7.0

A3: Functions for Simulating Response of Non-Linear System

Simulator.m

```
%Multi Degree of Freedom Earthquake Response Simulator - Ishan Singh-Levett
%NOTE 1 is the bottom floor

%Shear building model (flexural effects etc ignored)
%It is assumed that all members have 5% damping

clear all
clc
close all

%OPTIONS=============================================================
m=4;            %Number of degrees of freedom
eqfactor=1;     %Earthquake acceleration multiplication factor
qsmooth=1;      %0 to smooth acceleration, 1 for no smoothing
quake=2;        %1 For Pacoima, 2 For El Centro, 3 for Kobe
samem=0;        %0 for all masses equal, 1 for user input masses
samek=0;        %0 for all stiffnesses equal, 1 for user input stiffness
fund=1;         %intended fundamental natural frequency
dtype=0;        %0 for equivalent viscous damping, 1 for rayleigh damping, 2
for pre-determined coefficient
%====================================================================

[accel,name,inte]=hselect(qsmooth,quake,eqfactor);
%Generate Acceleration Vector From Record

clear eqfactor qsmooth quake

%Now choose type and magnitude of noise for acceleration and displacement
%data
ntype=menu('Which Type of Acceleration Noise?','Normal Distribution','Flat
Distribution','Mean Absolute Acceleration','None');

if ntype<4
    nmag=menu('What Magnitude of Noise?','3%','5%','10%','20%','50%');
else
    nmag=0;
end

dntype=menu('Which Type of Displacement Noise?','Normal Distribution','Flat
Distribution','Mean Absolute Acceleration','None');

if dntype<4
    dmag=menu('What Magnitude of Noise?','3%','5%','10%','20%','50%');
else
    dmag=0;
end

quakespan=inte*(length(accel)-1);
%Time length of earthquake

IC=zeros(m*3,1);
%Initial conditions of displacement, velocity and hysteretic displacement z
```

```
tspan=[0 quakespan];

%Firstly create necessary structural matrices
alpha=0.1;      %Bi-Linear Factor of Elements
p=2;            %shape parameter for bouc-wen loop
fac=188;        %Adjustment factor to ensure correct fundamental frequency
a=0.1;          %Parameters for rayleigh damping if used
b=0.1;
lamda=0.02;     %5% damping for each mode
y=0.04;         %default yieldpoint

% [M,Kt,Ke,Kh,ST]=hmandk(samem,samek,fund,fac,m,y,alpha,p); (Multi DOF)
[M,Kt,Ke,Kh,ST]=hmandk2(samem,samek,fund,fac,m,y,alpha,p);  %(Single DOF)
%Generate Mass and Stiffness Matrices

clear alpha p fund samem samek

[C,freqs,v]=hcreatec(Kt,M,m,dtype,a,b,lamda);
%Generate damping matrix

clear dtype a b lamda

[T,Disp,Vel,bodyacc,accel2,divisor,Z,RD,RV,RB,RFORCE,CDisp]=hEuler(IC,tspan,M
,ST,accel,C,inte,Kt,Ke,Kh,ntype,nmag,dntype,dmag,m);
%Simulate non-linear structural response using modified Eulers method

%Finally, output data as text for fitting program

number=num2str(y*1000);
dof=num2str(m);

fact=strcat('Body',name,number,'-',dof);
save(fact,'bodyacc','-ASCII');
fact=strcat('Ground',name,number,'-',dof);
save(fact,'accel2','-ASCII');
fact=strcat('Disp',name,number,'-',dof);
save(fact,'Disp','-ASCII');
```

hselect.m

```
%Select quake record - Ishan Singh-Levett 2005
function [accel,name,inte]=hselect(qsmooth,quake,eqfactor)

if quake==1         %if the earthquake is pacoima dam

    range=[17 1 800 10];
    %coordinates in earthquake file to read this will have to be changed with
changing format
    inte=0.005;
    %timestep of earthquake record, also dependent on format etc.
    scaler=9810;
    %earthquake scale in terms of m/s^2, dependent on format etc.
    xddg=dlmread('PACOM71A.eqc','',range);
    name='pacoima';
    long=length(xddg);
    width=min(size(xddg));
    X=[];
    for i=1:long
```

```
        for j=1:width
            X=[X xddg(i,j)];
        end
    end

    X=X/scaler;


%=========================================================================
elseif quake==2             %if the earthquake is el centro

%=========================================================================

    range=[17 1 1090 10];
    %coordinates in earthquake file to read this will have to be changed with
changing format
    inte=0.005;
    %timestep of earthquake record, also dependent on format etc.
    scaler=9810;
    %earthquake scale in terms of m/s^2, dependent on format etc.

    xddg=dlmread('ELCEN40A.eqc','',range);

    name='elcentro';

    long=length(xddg);
    width=min(size(xddg));

    X=[];
    for i=1:long
        for j=1:width
            X=[X xddg(i,j)];
        end
    end

    X=X/scaler;


    %=========================================================================
elseif quake==3             %if the earthquake is kobe
    %=========================================================================

    range=[17 0 220 9];
    %coordinates in earthquake file to read this will have to be changed with
changing format
    inte=0.02;
    %timestep of earthquake record, also dependent on format etc.
    scaler=100;
    %earthquake scale in terms of m/s^2, dependent on format etc.

    xddg=dlmread('KOBE95NS.eqn','',range);

    name='kobe';

    long=length(xddg);
    width=min(size(xddg));

    X=[];
    for i=1:long
```

```
        for j=1:width
            X=[X xddg(i,j)];
        end
    end

    X=X/scaler;


elseif quake==4     %if the earthquake is 2005 shake table el centro

    range=[4 1 29000 1];
    inte=0.001;
    scaler=1/9.81;

    xddg=dlmread('Table2005.eqe','',range);

    name='table';

    long=length(xddg);
    width=min(size(xddg));

    X=[];
    for i=1:long
        for j=1:width
            X=[X xddg(i,j)];
        end
    end

    X=X/scaler;

end

if qsmooth==0
    accel=movav(X,3)*eqfactor;
    %Smooth acceleration input before simulation
else
    accel=X*eqfactor;                    %No Smoothing
end
```

hmandk.m

```
%Create Mass and Stiffness Matrix
function [M,Kt,Ke,Kh,ST]=hmandk(samem,samek,fund,fac,m,y,alpha,p)

M=[];
%Mass matrix formulation
if samem==1
    for i=1:m
        fprintf('Please enter mass of degree of freedom %g : ',i);
        thism=input('');
        row=zeros(1,m);
        row(1,i)=thism;
        M(i,:)=row;
    end
else
    for i=1:m
        row=zeros(1,m);
        row(1,i)=1;
        M(i,:)=row;
```

```
    end
end

fprintf('\n');

ST=[];
%Matrix storing physical properties of each storey
if samek==1
    for i=1:m
        fprintf('Please enter pre-yield stiffness of floor %g : ',i);
        thisk=input('');
        fprintf('Please enter bi-linear factor of floor %g : ',i);
        thisr=input('');
        fprintf('Please enter yield point of floor %g : ',i);
        thisy=input('');
        fprintf('Please enter bouc-wen shape of floor %g : ',i);
        thisp=input('');
        row=[thisk thisr thisy thisp];
        ST(i,:)=row;
    end
else
    for i=1:m
        meff=sum(sum(M));
        keff=(fund*2*pi)^2*meff;
        if m==1
            eachk=keff;
        else
            eachk=exp(log(m*keff)/(m-1))*fac;
            %calculates overall stiffness to achieve desired natural
frequency
        end
        row=[eachk alpha y p];
        ST(i,:)=row;
    end
end

%Generate tangent, hysteretic and elastic stiffness matrices

Kt=zeros(m,m);
Kh=zeros(m,m);
Ke=zeros(m,m);

Kt(1,1)=ST(1,1)+ST(2,1);
Ke(1,1)=ST(1,1)*ST(1,2)+ST(2,1)*ST(2,2);
Kh(1,1)=ST(1,1)*(1-ST(1,2))+ST(2,1)*(1-ST(2,2));

Kt(1,2)=-ST(2,1);
Ke(1,2)=-ST(2,1)*ST(2,2);
Kh(1,2)=-ST(2,1)*(1-ST(2,2));

Kt(m,m)=ST(m,1);
Ke(m,m)=ST(m,1)*ST(m,2);
Kh(m,m)=ST(m,1)*(1-ST(m,2));

Kt(m,m-1)=-ST(m,1);
Ke(m,m-1)=-ST(m,1)*ST(m,2);
Kh(m,m-1)=-ST(m,1)*(1-ST(m,2));

for i=2:m-1
    Kt(i,i-1)=-ST(i,1);
```

```
    Ke(i,i-1)=-ST(i,1)*ST(i,2);
    Kh(i,i-1)=-ST(i,1)*(1-ST(i,2));

    Kt(i,i)=ST(i,1)+ST(i+1,1);
    Ke(i,i)=ST(i,1)*ST(i,2)+ST(i+1,1)*ST(i+1,2);
    Kh(i,i)=ST(i,1)*(1-ST(i,2))+ST(i+1,1)*(1-ST(i+1,2));

    Kt(i,i+1)=-ST(i+1,1);
    Ke(i,i+1)=-ST(i+1,1)*ST(i+1,2);
    Kh(i,i+1)=-ST(i+1,1)*(1-ST(i+1,2));
end
```

hmandk2.m

```
%Revised Stiffness Creator - Ishan Singh-Levett

%Create Mass and Stiffness Matrix
function [M,Kt,Ke,Kh,ST]=hmandk2(samem,samek,fund,fac,m,y,alpha,p)

M=[];
%Mass matrix formulation
if samem==1
    for i=1:m
        fprintf('Please enter mass of degree of freedom %g : ',i);
        thism=input('');
        row=zeros(1,m);
        row(1,i)=thism;
        M(i,:)=row;
    end
else
    for i=1:m
        row=zeros(1,m);
        row(1,i)=1;
        M(i,:)=row;
    end
end

fprintf('\n');

ST=[];
%Matrix storing physical properties of each storey
if samek==1
    for i=1:m
        fprintf('Please enter pre-yield stiffness of floor %g : ',i);
        thisk=input('');
        fprintf('Please enter bi-linear factor of floor %g : ',i);
        thisr=input('');
        fprintf('Please enter yield point of floor %g : ',i);
        thisy=input('');
        fprintf('Please enter bouc-wen shape of floor %g : ',i);
        thisp=input('');
        row=[thisk thisr thisy thisp];
        ST(i,:)=row;
    end
else
    for i=1:m
        meff=sum(sum(M));
        keff=(fund*2*pi)^2*meff;
        if m==1
            eachk=keff;
```

```
        else
            eachk=exp(log(m*keff)/(m-1))*fac;
            %calculates overall stiffness to achieve desired natural
frequency
        end
        row=[eachk alpha y p];
        ST(i,:)=row;
    end
end

%Generate tangent, hysteretic and elastic stiffness matrices

Kt=zeros(m,m);
Kh=zeros(m,m);
Ke=zeros(m,m);

Kt(1,1)=ST(1,1)+ST(2,1);
Ke(1,1)=ST(1,1)*ST(1,2)+ST(2,1)*ST(2,2);
Kh(1,1)=ST(1,1)*(1-ST(1,2));


Kt(1,2)=-ST(2,1);
Ke(1,2)=-ST(2,1)*ST(2,2);
Kh(1,2)=-ST(2,1)*(1-ST(2,2));

Kt(m,m)=ST(m,1);
Ke(m,m)=ST(m,1)*ST(m,2);
Kh(m,m)=ST(m,1)*(1-ST(m,2));

Kt(m,m-1)=-ST(m,1);
Ke(m,m-1)=-ST(m,1)*ST(m,2);

for i=2:m-1
    Kt(i,i-1)=-ST(i,1);
    Ke(i,i-1)=-ST(i,1)*ST(i,2);

    Kt(i,i)=ST(i,1)+ST(i+1,1);
    Ke(i,i)=ST(i,1)*ST(i,2)+ST(i+1,1)*ST(i+1,2);
    Kh(i,i)=ST(i,1)*(1-ST(i,2));

    Kt(i,i+1)=-ST(i+1,1);
    Ke(i,i+1)=-ST(i+1,1)*ST(i+1,2);
    Kh(i,i+1)=-ST(i+1,1)*(1-ST(i+1,2));
end
```

hcreatec.m

```
%Function to perform modal analysis and thus formulate appropriate damping
%matrix - Ishan Singh-Levett

function [C,freqs,v]=hcreatec(K,M,m,dtype,a,b,lamda)

%We first perform a modal analysis (assume negligible damping during this)

%Take flexibility matrix as inverse of stiffness matrix (note with large
%DOF it is inefficient to do this, calculate from repeated static analysis
%instead

F=inv(K);
```

```
A=F*M;
B=eye(m);

[v,d]=eig(A,B);
%note these eigenvectors are already normalised such that the euclidean norm
is one

freqs=1./d;
freqs=diag(freqs);
freqs=sqrt(freqs);
freqs=freqs./2/pi;
%freqs now contains modal frequencies in Hz

fprintf('Fundamental Frequency is (s) : %g.\n\n  ',min(freqs))

%Next we formulate the damping matrix

if dtype==0
%This method provides 5% damping at each mode

Mstar=v'*M*v;           % ie phi transpose m phi
Kstar=v'*K*v;           % ie phi transpose k phi

Cstar=[];
for i=1:m
    Cstar(i,i)=Mstar(i,i)*lamda*freqs(i)*2*pi*2;
end

C=v*Cstar*v';

%This damping matrix constant, whether the members are yielding or not, it
%will provide 5% damping at every mode.

elseif dtype==1
    C=a*M+b*K;
    %This is rayleigh damping
elseif dtype==2
    mass=M(1,1);
    c=2*0.31455*mass;
    C=c*diag(diag(ones(m,m)));
    %Create diagonal damping matrix
end
```

heuler.m

```
%Euler Integration for Bouc-Wen Hysteresis - Ishan Singh-Levett 2005

function
[T,Disp,Vel,bodyacc,accel2,divisor,Z,RD,RV,RB,RFORCE,CDisp]=hEuler(IC,tspan,M
,ST,accel,C,inte,Kt,Ke,Kh,ntype,nmag,dntype,dmag,m)

h=0.001;
%integration time step
divisor=int32(inte/h);
%divisor for vector scaling purposes

if nmag==1
    fac=0.03;
```

```
elseif nmag==2
    fac=0.05;
elseif nmag==3
    fac=0.1;
elseif nmag==4
    fac=0.2;
elseif nmag==5
    fac=0.5;
end
%Noise magnitude setup

if dmag==1
    dfac=0.03;
elseif dmag==2
    dfac=0.05;
elseif dmag==3
    dfac=0.1;
elseif dmag==4
    dfac=0.2;
elseif dmag==5
    dfac=0.5;
end
%Noise magnitude setup

T=0:h:tspan(2);

n=length(T)-1;         %number of integration steps

Disp=zeros(n+1,m);      %floor displacements (noisy)
Vel=zeros(n+1,m);       %floor velocities
bodyacc=zeros(n+1,m);   %floor accelerations
Z=zeros(n+1,m);         %floor hysteretic displacements
RFORCE=zeros(n+1,m);    %elemental restoring forces
CDisp=zeros(n+1,m);     %clean displacements for comparison in fitter

[accel2]=hinterp(accel,divisor,n);
%enlarge acceleration vector to match sampling rates

Xn=IC;
%Initial conditions

B=[zeros(m,m) eye(m) zeros(m,m);-inv(M)*Ke -inv(M)*C -inv(M)*Kh;zeros(m,m)
zeros(m,m) zeros(m,m)];
%State space matrix for Bouc-Wen elements

Fn=zeros(3*m,1);
%Forcing vector at current time step
Fm=zeros(3*m,1);
%Forcing vector at next time step

D=diag(ones(m,1));
for j=2:m
    D(j,j-1)=-1;
end
%Co-ordinate transformation matrix for absolute to relative displacement

dzr=zeros(m,m);
%Matrix to store relative hysteretic velocity

r=zeros(m,1);
```

```
rdot=zeros(m,1);
%Relative displacement and velocity

Disp(1,:)=Xn(1:m)';
Vel(1,:)=Xn(m+1:2*m)';
bodyacc(1,:)=-accel2(1)*ones(1,m);
Z(1,:)=Xn(2*m+1:3*m)';
%Setup initial conditions

%Now perform euler integration of equations of motion
for i=2:n+1

    xdot=Xn(m+1:2*m);
    z=Xn(2*m+1:3*m);
    x=Xn(1:m);

    r=D*x;
    rdot=D*xdot;

    for j=1:m
    dzr(j,j)=1-0.5*(1+sign(rdot(j)*z(j)))*abs(z(j)/ST(j,3))^ST(j,4);
    end

    dzx=dzr*D;

    B(2*m+1:3*m,m+1:2*m)=dzx;

    Fn(m+1:2*m)=-accel2(i-1)*ones(m,1);
    Fm(m+1:2*m)=-accel2(i)*ones(m,1);

    Xdn=B*Xn+Fn;
    Xstar=Xn+h*Xdn;
    Xdstar=B*Xstar+Fm;
    Xnew=Xn+h*(Xdn+Xdstar)/2;
    %Modified euler method calculates displacement at next time step

    Z(i,:)=Xnew(2*m+1:3*m);

    %Next, add noise to displacement
    if dntype==1
        Disp(i,:)=Xnew(1:m)+rand(m,1).*Xn(1:m)*dfac/3;
    elseif dntype==2
        Disp(i,:)=Xnew(1:m)+2*(rand(m,1)-0.5*ones(m,1)).*Xn(1:m)*dfac;
    else
        Disp(i,:)=Xnew(1:m);
    end

    %Store Clean displacement and velocity for later comparison
    CDisp(i,:)=Xnew(1:m);
    Vel(i,:)=Xnew(m+1:2*m);

    %Now add noise to acceleration data
    if ntype==1
    bodyacc(i-1,:)=Xdn(m+1:2*m)+randn(m,1).*Xdn(m+1:2*m)*fac/3;
    accel2(i)=accel2(i)+randn*accel2(i)*fac/3;
    elseif ntype==2
    bodyacc(i-1,:)=Xdn(m+1:2*m)+2*(rand(m,1)-
0.5*ones(m,1)).*Xdn(m+1:2*m)*fac;
    accel2(i)=accel2(i)+2*(rand-0.5)*accel2(i)*fac;
    else
```

```
    bodyacc(i-1,:)=Xdn(m+1:2*m);
    end

    %Prepare for next time step
    Xn=Xnew;
end

%=========================================
%=========================================

    xdot=Xn(m+1:2*m);
    z=Xn(2*m+1:3*m);
    x=Xn(1:m);

    r=D*x;
    rdot=D*xdot;

    for j=1:m
    dzr(j,j)=1-0.5*(1+sign(rdot(j)*z(j)))*abs(z(j)/ST(j,3))^ST(j,4);
    end

    dzx=dzr*D;
    B(2*m+1:3*m,m+1:2*m)=dzx;
    Xdn=B*Xn+Fm;

    if ntype==1
    bodyacc(n+1,:)=Xdn(m+1:2*m)+randn(m,1).*Xdn(m+1:2*m)*fac/3;
    elseif ntype==2
    bodyacc(n+1,:)=Xdn(m+1:2*m)+2*(rand(m,1)-
0.5*ones(m,1)).*Xdn(m+1:2*m)*fac;
    else
    bodyacc(n+1,:)=Xdn(m+1:2*m);
    end

%=======================================
%=======================================


%Determine noise constants for uniformly distributed noise
con=mean(abs(bodyacc(:,1)))';
dcon=mean(abs(Disp(:,1)))';


%Now, add noise to simulated data
if ntype==3
for i=1:n+1
        bodyacc(i,:)=bodyacc(i,:)'+2*(rand(m,1)-0.5*ones(m,1)).*con*fac;
        accel2(i)=accel2(i)+2*(rand-0.5)*con*fac;
end
end

if dntype==3

    for i=1:n+1
        Disp(i,:)=Disp(i,:)'+2*(rand(m,1)-0.5*ones(m,1)).*dcon*dfac;
    end
end

RD=Disp*D';
```

```
RV=Vel*D';
RB=bodyacc*D';

%Calculate restoring forces to ensure hysteresis is correctly implemented
for i=1:m
    RFORCE(:,i)=ST(i,2)*ST(i,1)*RD(:,i)+(1-ST(i,2))*ST(i,1)*Z(:,i);
End
```

hinterp.m

```
%Function to interpolate acceleration file for simulation purposes - Ishan
%Singh-Levett Feb 2006

function [accel2]=hinterp(accel,divisor,n)

P=zeros(n+1,1);

P(1)=accel(1);

divisor=single(divisor);

for i=2:n
    ind1=ceil(i/divisor);
    ind2=ind1+1;
    ind3=rem(i-1,divisor);
    P(i)=accel(ind1)+(accel(ind2)-accel(ind1))/divisor*ind3;
end

P(n+1)=accel(length(accel));

accel2=P;
```

Published with MATLAB®7.0

A4: Functions for Stiffness and Permanent Displacement Identification of Non-Linear Systems

fitter.m

```
%N-DOF Bouc-Wen Fitting Algorithm - Ishan Singh-Levett May 2006

clear all
clc
close all

m=4;
%no of DOF
quake=2;
%1 For Pacoima, 2 For El Centro, 3 for Kobe, 4 for table,5 for simple model,
6 for complex model
y=0.04;
%yield point (only for file numbering purposes)
yield=num2str(y*1000);
dof=num2str(m);

%First, load ground and structural data

if quake==1
    name=strcat('pacoima',yield,'-',dof);
elseif quake==2
    name=strcat('elcentro',yield,'-',dof);
elseif quake==3
    name=strcat('kobe',yield,'-',dof);
end

mass=1;
%mass of each DOF
M=diag(mass*diag(ones(m,m)));
%Mass Matrix

PK=1.6131e+003*ones(4,1);
%Previously estimated stiffness for comparison

dname=strcat('Disp',name);
bname=strcat('Body',name);

Disp=load(dname);
%Generate realistic displacement measurement from this
bodyacc=load(bname);
%Structural acceleration

aname=strcat('Ground',name);
accel=load(aname);
%Ground acceleration

t1=cputime;
inte=0.001;
%Sampling rate

quakespan=inte*(length(bodyacc)-1);
%Time length of earthquake
```

```
t=0:inte:quakespan;

rt=10;
%displacement reporting interval

 C=2*[2.1799   -0.6279   -0.1125   -0.0545
   -0.6279    2.0674   -0.6824   -0.1670
   -0.1125   -0.6824    2.0128   -0.7949
   -0.0545   -0.1670   -0.7949    1.3850];
%Estimated damping matrix

[newv,newd,v,d,d2,md,smoothd]=adjust2(bodyacc,rt,quakespan,t,inte,Disp,m);
%Calculate estimated displacement response

[smoothv]=cdiff(smoothd,inte);
[smoothb]=cdiff(smoothv,inte);
%Numerical differentiation to find estimated velocity and acceleration

fp=2;
%Stiffness fitting period
div=5;
%Divisor to find permanent displacement fitting period

alpha=0.1;
%Bi-linear factor estimate

D=diag(ones(m,1));
for j=2:m
    D(j,j-1)=-1;
end
%Co-ordinate transformation matrix from absolute to relative

RD=smoothd*D';
RV=smoothv*D';
RB=smoothb*D';
%Calculate relative displacement, velocity and accelerations

[KK,zp,ZX,Ktime,Ztime,ZK,Z,KK2]=boucwen(fp,div,quakespan,accel,smoothb,smooth
v,smoothd,inte,M,C,alpha,m,RD);
%Use least squares procedure to identify stiffness and permanent
%displacement

%Save parameters to ASCII file for resimulation
O=[fp div];

t2=cputime;
fact='FStiff';
save(fact,'KK','-ASCII');
fact='FDZ';
save(fact,'ZK','-ASCII');
fact='Op';
save(fact,'O','-ASCII');

%Plot fitted parameters
KZPlot(KK,ZX,Ktime,Ztime,fp,zp,m,PK,t,smoothd);
```

adjust2.m

```
%Function to adjust integrated data - Ishan Singh-Levett 2005

function
[newv,newd,v,d,d2,md,smoothd]=adjust2(bodyacc,rt,quakespan,t,inte,Disp,m)

ID=Disp(1,:);
%initial conditions of displacement

v=zeros(length(t),m);
d=zeros(length(t),m);
%vectors to store integrated displacement and velocity

d(1,:)=ID;

d2=zeros(length(t),m);
md=zeros(length(t),m);

T=0:rt:quakespan;
%rt is the displacement reporting interval

if max(T)<quakespan
    T=[T quakespan];
end

newd=zeros(length(d),m);
newv=zeros(length(v),m);

smoothd=zeros(length(d),m);

AD=zeros(length(T)-1,m);
%Store errors in this matrix
AD2=zeros(length(T)-1,m);
%Store error at next adjustment interval in this matrix

CSS=zeros(length(T)-1,m);

Corrector=[];

oldc=zeros(m,4);
%Store coefficients of polynomial error approximation
oldanalytic=zeros(m,4);
oldt2=0;

oldind1=1;

SCrec=[];

for i=1:length(T)-1

    %Generate indices for integration
    time1=T(i);
    time2=T(i+1);
    ind1=int32(time1/inte+1);
    ind2=int32(time2/inte+1);

    %Numerical integration to estimate structural response
```

```
    v(ind1:ind2,:)=ones(ind2-
ind1+1,1)*v(ind1,:)+cumtrapz(t(ind1:ind2),bodyacc(ind1:ind2,:));
    d(ind1:ind2,:)=ones(ind2-
ind1+1,1)*d(ind1,:)+cumtrapz(t(ind1:ind2),v(ind1:ind2,:));

    %Backwards moving average to smooth noise
    d2(oldind1:ind2,:)=d(oldind1:ind2,:);

md(oldind1:ind2,:)=backav2(Disp(oldind1:ind2,:),md(oldind1:ind2,:),1,m,i);

    %Calculate integration errors
    if i>1
    adjust=md(ind1,:)-d2(ind1,:);
    AD1(i,:)=adjust;
    else
        adjust=zeros(1,m);
        AD1(i,:)=adjust;
    end

    %Calculate slope in integration errors
    if i>1
        cslope=(AD1(i,:)-AD1(i-1,:))/rt;
    else
        cslope=zeros(1,m);
    end

    %Calculate errors for next adjustment interval
    CSS=[CSS;cslope];

    adjustnext=md(ind2,:)-d2(ind2,:);
    AD2(i,:)=adjustnext;

    if i>1
        cslope2=(AD2(i,:)-AD2(i-1,:))/rt;
    else
        cslope2=zeros(1,m);
    end

    %Generate vector to correct integrated properties
    advect=(t(ind1:ind2)'-ones(ind2-ind1+1,1)*t(ind1))*cslope;

    %Generate adjusted estimate of integrated properties
    newd(ind1:ind2,:)=d(ind1:ind2,:)+ones(ind2-ind1+1,1)*adjust+advect;
    newv(ind1:ind2,:)=v(ind1:ind2,:)+ones(ind2-ind1+1,1)*cslope;

    Corrector(ind1:ind2,:)=ones(ind2-ind1+1,1)*adjust+advect;
    oldind1=ind1;

    %Now, find smooth cubic approximation to error in integrated properties

[SC,t2,analytic]=findcubic(Corrector(ind1:ind2,:),m,t(ind1:ind2),inte,rt,i,ad
just,cslope,adjustnext,cslope2,oldt2,oldanalytic);
    %Add cubics to displacement
    smoothd(ind1:ind2,:)=d(ind1:ind2,:)+SC;
    SCrec(ind1:ind2,:)=SC;
    oldanalytic=analytic;
    oldt2=t2;
end
```

findcubic.m

```
%Fits cubic curve through corrector points - Ishan Singh-Levett July 2006

function
[SC,t2,analytic]=findcubic(Corrector,m,t,inte,rt,i,adjust,cslope,adjustnext,c
slope2,oldt2,oldanalytic);

options=optimset('LargeScale','off');

t2=t-ones(1,length(t))*min(t);

for j=1:m

    %Analytically differentiate previous cubic curve
    cder2=polyder(oldanalytic(j,:));
    cderder2=polyder(cder2);

    timenow=t2(1);
    timesoon=max(t2);

    timesoon2=max(oldt2);

    %Assemble matrix to solve for polynomial coefficients
    AA=[rt^3 rt^2;3*rt^2 2*rt];

    %Generate constraint vector
    BB=[adjustnext(j)-polyval(cder2,timesoon2)*rt-
polyval(oldanalytic(j,:),timesoon2);cslope2(j)-polyval(cder2,timesoon2)];

    %Solve for polynomial coefficients
    coeffs=AA\BB;

    %Generate cubic curve for this time interval
    if i==1
        analytic(j,:)=[coeffs(1) coeffs(2) cslope(j) adjust(j)];
        analytic(j,:)=[coeffs(1) coeffs(2) polyval(cder2,timesoon2)
polyval(oldanalytic(j,:),timesoon2)];
    end

    SC(:,j)=polyval(analytic(j,:),t2);

end
```

cdiff.m

```
%Central Difference Differentiator - Ishan Singh-Levett July 2006

function [A]=cdiff(B,step)

A=zeros(size(B));

for i=2:length(B)-1
    A(i,:)=(B(i+1,:)-B(i-1,:))/2/step;
end

A(1,:)=A(2,:)+A(2,:)-A(3,:);
A(end,:)=A(end-1,:)+A(end-1,:)-A(end-2,:);
```

boucwen.m

```
%Bouc-Wen Least Squares fitting for N-DOF - Ishan Singh-Levett

function
[KK,zp,ZX,Ktime,Ztime,ZK,Z,KK2]=boucwen(fp,div,quakespan,accel,bodyacc,newv,n
ewd,inte,M,C,alpha,m,RD);

zp=fp/div;
%hence fit z at these intervals

T=0:fp:quakespan;
if max(T)<quakespan
    T=[T quakespan];
end
%Times for stiffness fitting

T2=0:zp:quakespan;
if max(T2)<quakespan
    T2=[T2 quakespan];
end
%Times for permanent displacement fitting

Time=0:inte:quakespan;
if max(Time)<quakespan
    Time=[Time quakespan];
end

DM=diag(M);
DC=diag(C);
%Separate coefficients of mass and damping matrices for fitting process

KK=zeros(length(T)-1,m);
KK2=zeros(length(T)-1,m);
ZX=zeros(length(T2)-1,m);
ZK=zeros(length(T2)-1,m);
Z=zeros(size(newd));
%Generate storage matrices for fitted parameters

Ktime=[];
Ztime=[];
%Vectors for storing fitting times

for i=1:length(T)-1

    time1=T(i);
    time2=T(i+1);
    ind1=int32(time1/inte+1);
    ind2=int32(time2/inte);
    %Generate indices for construction of least square matrices

    Ktime=[Ktime time2];

    intdterm=(C*newv(ind1:ind2,:)')';
    %Calculate damping forces for this period

    for k=1:m
```

```
        ls=zeros(ind2-ind1+1,div+1);
        rs=zeros(ind2-ind1+1,1);

        ls(1:ind2-ind1+1,1)=RD(ind1:ind2,k);
        %Setup 1st column of LHS matrix

        %Perform summations for RHS vector
        mterm=-accel(ind1:ind2)*sum(DM(k:m));
        aterm=-bodyacc(ind1:ind2,k:m)*DM(k:m);

        if k<m
            dterm=-sum(intdterm(:,k:m)')';
        else
            dterm=-intdterm(:,m);
        end


        %Add damping and inertial forces for RHS vector
        rs=mterm+aterm+dterm;

        T2=time1:zp:time2;

        %Complete LHS matrix by adding unit values
        if time2==quakespan & (time2-time1)<fp
            if max(T2)<quakespan
                T2=[T2 quakespan];
            end
            cut=ceil((time2-time1)/zp);
            ls=ls(:,1:cut+1);
        end

        for j=1:length(T2)-1
            time3=T2(j);
            time4=T2(j+1);
            ind3=int32(time3/inte+1)-ind1+1;
            ind4=int32(time4/inte)-ind1+1;
            ls(ind3:ind4,j+1)=ones(ind4-ind3+1,1);
            if k==1
                Ztime=[Ztime time4];
            end
        end

        %Perform least squares solution
        b=lsqlin(ls,rs);


        KK(i,k)=b(1);
        ZK((i-1)*div+1:(i-1)*div+length(b)-1,k)=b(2:length(b));

        %Convert from fitted parameter to actual permanent displacement
        if k==1
            ZX((i-1)*div+1:(i-1)*div+length(b)-1,k)=-ZK((i-1)*div+1:(i-
1)*div+length(b)-1,k)/(1-alpha)/KK(i,k)/(1+(alpha/(1-alpha)));
        else
            ZX((i-1)*div+1:(i-1)*div+length(b)-1,k)=-ZK((i-1)*div+1:(i-
1)*div+length(b)-1,k)/(1-alpha)/KK(i,k)/(1+(alpha/(1-alpha)))+ZX((i-
1)*div+1:(i-1)*div+length(b)-1,k-1);
        end

        %Calculate Z for inspection purposes
```

```
        for w=1:length(Ztime)
            time5=Ztime(w)-zp;
            time6=Ztime(w);
            ind5=int32(time5/inte+1);
            ind6=int32(time6/inte);
            Z(ind5:ind6,k)=ZX(w,k)*ones(ind6-ind5+1,1)+newd(ind5:ind6,k);
        end

    end
end
```

kzplot.m

```
%Function to plot ks and zs - Ishan Singh-Levett May 2006

function KZPlot(KK,ZX,Ktime,Ztime,fp,zp,m,PK,t,smoothd);

for k=1:m

figure(2*(k-1)+1)

Ktimeforplot=[0 Ktime];

Ktp=zeros(length(Ktime)*2,1);
Krecord=zeros(length(Ktp),1);

Ktp(1)=0;
Krecord(1)=KK(1,k);

Ktp(end)=Ktime(end);
Krecord(end)=KK(end,k);

for z=1:length(Ktime)-1
    Ktp(2*z)=Ktime(z);
    Ktp(2*z+1)=Ktime(z);
    Krecord(2*z)=KK(z,k);
    Krecord(2*z+1)=KK(z+1,k);
end

hold on
plot(Ktimeforplot,PK(k)*ones(length(Ktime)+1,1),'LineStyle','--
','Color','r','LineWidth',2)
plot(Ktp,Krecord,'Color','b','LineWidth',2);
legend('Actual Stiffness','Fitted Stiffness')

index=num2str(k);
title(strcat('\bfFitted Stiffness Variation - Story  ',index),'FontSize',20);
% ylim([min(Krecord)-0.5*max(Krecord) max(Krecord)*1.5]);
ylim([0 PK(k)*1.5]);
% legend('Actual Stiffness','Fitted Stiffness')
xlabel('\bfTime (s)','FontSize',20);
ylabel('\bfStiffness (N/m)','FontSize',20);
grid on

figure(2*k)

Ztp=zeros(length(Ztime)*2,1);
Zrecord=zeros(size(Ztp));
```

```
Ztp(1)=0;
Zrecord(1)=ZX(1,k);

Ztp(end)=Ztime(end);
Zrecord(end)=ZX(end,k);

for z=1:length(Ztime)-1
    Ztp(2*z)=Ztime(z);
    Ztp(2*z+1)=Ztime(z);
    Zrecord(2*z)=ZX(z,k);
    Zrecord(2*z+1)=ZX(z+1,k);
end

plot(t,smoothd(:,k),'r',Ztp,Zrecord,'b');

title(strcat('\bfFitted Permanent Displacement - Story
',index),'FontSize',20);
ylim([max(max(Zrecord))-(max(max(Zrecord))-min(min(Zrecord)))*1.5
(max(max(Zrecord))-min(min(Zrecord)))*1.5+min(min(Zrecord))]);
ylim([1.2*min(min(smoothd)) max(max(smoothd))*1.2]);legend('Storey
Displacement','Fitted Permanent Displacement')
xlabel('\bfTime (s)','FontSize',20);
ylabel('\bfDisplacement (m)','FontSize',20);
grid on

end
```

backav2.m

```
%Backwards moving average for more realistic data for fitting - Ishan
%Singh-Levett 2005

function [A]=backav(C,A,b,m,i)

if b>length(C)
    b=length(C);
end

B=zeros(length(C),m);

if i==1
    for j=1:b
        %cant calculate average until after this point
        B(j,:)=C(j,:);
    end
end

divisor=0;
Q=zeros(b,1);

factor=1.005;

for j=0:b-1
    divisor=divisor+factor^j;
    Q(j+1)=factor^j;
end

for j=b+1:length(A)
    index=j-b+1;
```

```
    P=C(index:j,:)';
    B(j,:)=P*Q/divisor';
end

A(1:b,:)=A(1:b,:);
A(b+1:length(A),:)=B(b+1:length(A),:);
```

Published with MATLAB®7.0