REVIEW, IMPLEMENTATION AND DEMONSTRATION OF DYNAMIC ANALYSIS AND GROUND MOTION MODELS

by

Vasantha Ramani

B.Tech., National Institute of Technology, Tiruchirappalli, 2012

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Civil Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

July 2015

Abstract

Dynamic structural analysis in recent years has gained importance due to increasing need to design structures for seismic resistance and also meet different performance demands. Also, structures are becoming more complex and it is difficult to accurately simulate their dynamic behavior using static analyses. With the advent of better computational capacity, engineers have been adopting computer programs for design and analyses of buildings. Making use of the advancement in computation and the need for dynamic analyses, tools required for dynamic analyses are implemented in a software called *Rts*. *Rts* is the next version of *Rt*, a computer program for risk and reliability analysis. For dynamic analyses it is required to have a ground motion input which can either be selected from existing databases of ground motion records or be generated as synthetic ground motions. If using actual ground motions it is required to modify them so that they match the anticipated hazard level. To overcome the limitation of scarcity of ground motion records, synthetic ground motion models can be used. To accomplish this in *Rts*, a synthetic ground motion model is implemented. The dynamic analysis algorithm and the ground motion models are implemented using object oriented programming. These implementations can be seen as a stepping stone to develop a computer program that would be robust and closely simulate the behavior of structures. It also forms the platform for future research for performance based earthquake engineering design and reliability analysis.

Preface

This manuscript is an outcome of my original research work. It was written by me and reviewed by my supervisor Dr. Terje Haukaas. I have developed, verified and implemented the dynamic analysis tool, scaled ground motion models and synthetic ground motion model in *Rts,* which is the next version of *Rt* software developed by Dr. Mojtaba Mahsuli and Dr. Haukaas. The example for verification of dynamic analysis is developed in collaboration with Dr. Haukaas. The building example used for demonstration purpose is a modification of the model developed by Abbas Javaherian and Dr. Haukaas.

Table of Contents

List of Tables

List of Figures

Acknowledgements

I would like to express my gratitude to my supervisor Dr. Terje Haukaas for providing me an opportunity to work on this topic. He has guided me and provided me with interesting discussion and suggestions throughout my research, which has greatly helped me to grow as a researcher. I would also like to thank Dr. Thomas Tannert for being the second reader of this thesis.

I would like to thank the risk and reliability research group members (Abbas, Sai Ganesh, Steve, Alfred and Gurvinder) for sharing their views on my research and providing me with ideas for improvement. I would like to specially thank Abbas for providing me with the example input file. It was a pleasure working with the research group.

I would also like to acknowledge the instructors of UBC who had helped to improve my knowledge on structural and earthquake engineering. I would also like to thank the Department of Civil Engineering for providing with the amenities required for carrying out research. I would like to thank my funding agency for providing me with the financial support in the form of discovery grant.

I would like to thank my friends Malar, Divya, Shreya, Aarthi, Brook, Kai, Manav, Sai, Farooq, Yasasvy, Nayantara, Sharan, Abhinav and others for supporting me during my stay in Vancouver and making it a memorable one. Finally, this would not have been possible without the unflagging support of my family. I would like to thank God for without whose grace this would not have been possible.

Dedication

to

my

grandfather

S. Somasundaram

Chapter 1: Introduction

1.1 Motivation

In performance based earthquake engineering (PBEE) the main objective is to seek desired performance levels of the structure for different intensity measure of earthquake hazard. See for example Gunay and Mosalam (2012) and other references. PBEE has four main components (Moehle and Deierlein 2004). First is the intensity measure (IM) that is the ground motion hazard. Second component is the engineering demand parameter (EDP) in the form of response of the system. The third component is the damage measure (DM) which is the level of damage to the structure. The last component is the decision variable (DV) which is usually described in terms of cost for a particular level of damage. The basic procedure of PBEE is illustrated in Figure 1.1.

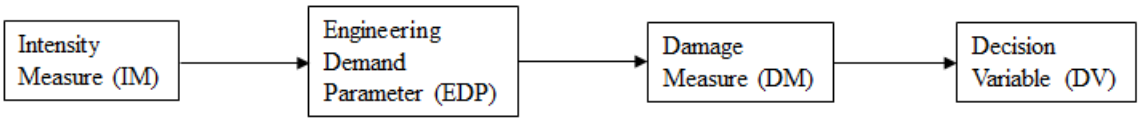

Figure 1.1: Performance based earthquake engineering.

It is seen from the diagram in Figure 1.1 that the hazard level for the site is first determined. This is achieved through probabilistic seismic hazard analysis (PSHA) (Kramer 1996). Ground motions for the determined IM is given as input load for the structure. The response of the structure (EDP) is determined as displacement, story drift or acceleration. The EDP determined, relates to DM. The final step is determination of DV, which is related to DM and that would assist engineers to make risk based decisions. Every level of PBEE is associated with uncertainty and this is better described in probabilistic terms. One way to summarize PBEE is using the following expression which yields the annual frequency of

exceeding some decision variable (Gunay and Mosalam 2012):

$$\lambda(DV) = \iiint G(DV \mid DM) \mid dG(DM \mid EDP) \mid dG(EDP \mid IM) \mid d\lambda(IM) \qquad (1)$$

where $G(x|y)$ is the complementary cumulative distribution function of X given Y, $\lambda(z)$ is the annual frequency of exceeding z, $dG(x|y)$ and $d\lambda(z)$ are the differentials of $G(x|y)$ and $\lambda(z)$ respectively.

From the preceding discussion it is realized that the decision making in earthquake engineering is a complex process. Also, in order to simulate built environment of buildings it is required to have predictive models. Thus, a computer program is required to have probabilistic models for hazard, damage and cost to achieve simulation, which would assist engineers to make risk based decisions. The computer program, *Rts* which is the next version of *Rt*, already houses numerous tools for risk and reliability analyses. To make it more robust the need for implementation of few more models and tools are identified. The motivation for implementation of each of the models and tools in *Rts* are as follows:

1. It is realized from the preceding discussion that it is important to perform nonlinear dynamic analysis. In dynamic analyses of structures subjected to ground motions, a time series is required to be given as input. The response of the structure is required to be determined at each time step. In order to accomplish linear and nonlinear dynamic analyses of structures in *Rts*, it is required to implement algorithms for dynamic analysis.
2. Determination of the hazard level, is one of the first steps in PBEE. For this hazard level it is required to subject the structure to ground motions which is anticipated at the site. The ground motions selected from the databases are required to be scaled to match the anticipated intensity level. For this reason it is required to have ground

motion models for scaling of ground motion in *Rts*.

3. One way of subjecting the structure to dynamic loading is to use the actual ground motion records as suggested in various codes (ASCE/SEI 7-10; ATC-63 2009). The other way is to use synthetic ground motions (Iyengar and Iyengar 1969; Der Kiureghian and Crempien 1989; Li and Der Kiureghian 1995; Rezaeian and Der Kiureghian 2010). Considering the advantages of using synthetic ground motions, which is discussed in the later sections of this thesis, it is required to have a synthetic ground motion model.

1.2 Objectives

Based on the discussion in Section 1.1, the objectives of this research are as follows:

- Implement dynamic analysis algorithm in *Rts*
- Review the scaled ground motions and synthetic ground motion models
- Implement synthetic and scaled ground motion models in *Rts*

The implementation of dynamic analysis and ground motion models in *Rts* are achieved through object-oriented programming (Deitel and Deitel 2013). *QtCreator* (2014) is the platform used to create the user interface.

1.3 Overview of the Thesis

In this section the general overview of organization of the thesis is laid out. The second chapter introduces to the theory behind dynamic analysis algorithm implemented in *Rts*. It also includes the procedure and results obtained with regard to verification of the implemented code. The dynamic analysis of a cantilever column and one-storey building

subjected to ground motion in *Rts* is also demonstrated in this chapter. The third chapter deals with the review of the methods adopted to select and scale ground motions. The scaled ground motion models implemented in *Rts* are presented in this chapter. The fourth chapter deals with the review of synthetic ground motions. It also talks about the synthetic ground motion model implemented in *Rts*. Conclusions and summary of the work is provided in the sixth chapter of the thesis. The code for dynamic analysis and ground motion models, and the input files for the examples are included as appendices.

Chapter 2: Dynamic Analysis with Ground Motions

2.1 Governing Equations for SDOF Problems

For detailed design of structures, dynamic analysis is performed by subjecting the structures modelled in a finite element computer program to ground motions. The most basic model consists of a single degree of freedom system (SDOF) as shown in Figure 2.1. A SDOF system is described as a system with a mass, *M*, connected to a spring with stiffness, *K*, and a dashpot with damping, *C*. The mass is allowed to move only in one direction. The displaced shape of the mass at any time *t* relative to the ground is given by $u(t)$, when subjected to external time varying load $F(t)$.

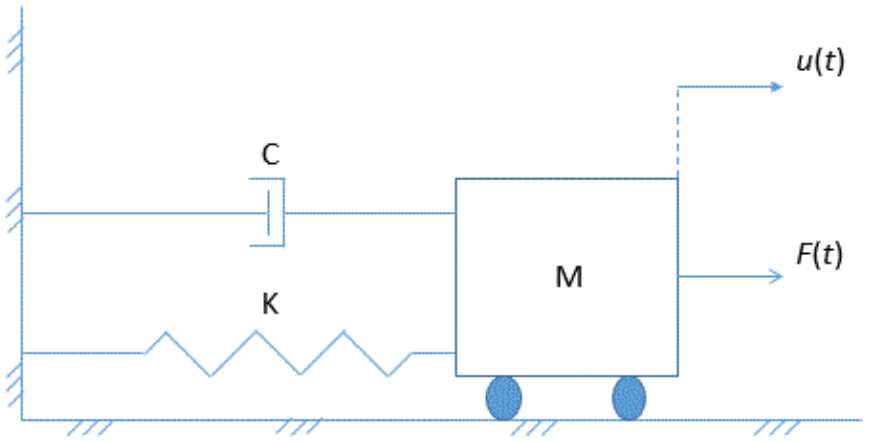

Figure 2.1: SDOF system.

When the mass is displaced, it is subjected to four forces namely the inertia force ($M.\ddot{u}$), the damping force ($C.\dot{u}$), the elastic force ($K.u$) and the external force ($F(t)$). For equilibrium the following equation must be satisfied at all times (See for example Haukaas (2014) and others):

$$M \cdot \ddot{u} + C \cdot \dot{u} + K \cdot u = F(t) \quad (2)$$

When the external load in Equation (2) is an earthquake ground motion, then the external load is represented as:

$$F(t) = -M \cdot \ddot{u}_g(t) \quad (3)$$

where $\ddot{u}_g(t)$ is the ground acceleration.

Substituting Equation (3) in Equation (2) yields:

$$M \cdot \ddot{u} + C \cdot \dot{u} + K \cdot u = -M \cdot \ddot{u}_g \quad (4)$$

Equation (4) is solved at each time step to get the response of a SDOF system. This forms the basis for response of a system when subjected to a ground motion. The dynamic analysis of a multiple degree of freedom system is explained in Section 2.2 of this chapter.

2.2 Governing Equations for MDOF Problems

All structures have infinite degrees of freedom. But to study the dynamic behavior, the structures are idealized as a multiple degree of freedom system (MDOF). The mass is lumped at different locations of the structure for a MDOF system. The deformed shape of the structure is defined by the displacement of multiple degrees of freedom. An example of a MDOF system is shown in Figure 2.2, where the mass of the system is assumed to be lumped at the center of the slab. For a MDOF system, the equation of motion is:

$$\mathbf{M} \cdot \ddot{\mathbf{u}} + \mathbf{C} \cdot \dot{\mathbf{u}} + \mathbf{K} \cdot \mathbf{u} = \mathbf{F}(t) \quad (5)$$

where $\boldsymbol{M}$ is the mass matrix, $\boldsymbol{C}$ is the damping matrix, $\boldsymbol{K}$ is the stiffness matrix and $\boldsymbol{F}(t)$ is the external load vector.

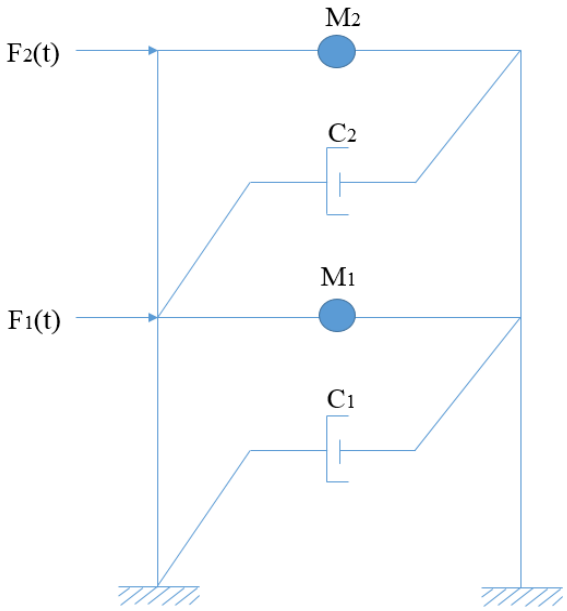

Figure 2.2: MDOF system.

In Equation (5) the mass matrix, the stiffness matrix, the damping matrix and the external load vector are required to be determined. This is achieved using the finite element method (FEM) (Cook 1995). To describe the derivation of the mass matrix, the stiffness matrix and the external load vector using FEM, a two dimensional and a three dimensional beam element are chosen as examples and described in the following section.

2.2.1 Finite Element Formulation in Dynamic Analysis

This section describes the equations governing the vibration of continuum. The equations are formulated using the weak form of boundary value problem. To illustrate the finite element formulation, consider a 2D beam element, shown in Figure 2.3 with a mass $m(x)$ per unit length and flexural rigidity $EI(x)$ vibrating in traverse direction.

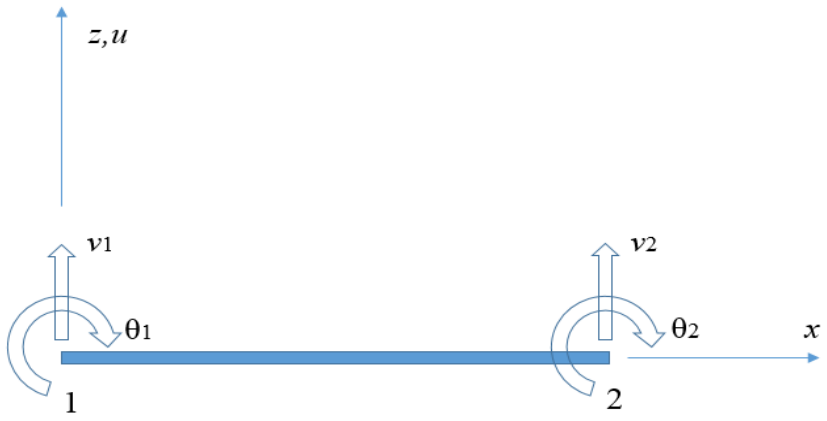

Figure 2.3: 2D Frame element.

According to the principle of virtual work, when a system is subjected to a virtual displacement, $\delta u(x)$, and the system is in equilibrium then the external virtual work, δW_E, is equal to the internal virtual work, δW_I, given as:

$$\delta W_E = \delta W_I \tag{6}$$

The external virtual work due to the virtual displacement, $\delta u(x)$, is:

$$\delta W_E = \int_0^L q_z(x,t)\delta u(x)dx \tag{7}$$

where $q_z(x,t)$ is the inertia force given as:

$$q_z(x,t) = -m(x)[\ddot{u}(x,t) + \ddot{u}_g(t)] \tag{8}$$

Substituting Equation (8) in Equation (7) yields:

$$\delta W_E = -\left(\int_0^L m(x)[\ddot{u}(x,t) + \ddot{u}_g(t)]\delta u(x)dx\right) \tag{9}$$

The internal virtual work due to the virtual displacement, $\delta u(x)$, is given as:

$$\delta W_I = \int_V \sigma \delta \varepsilon dV \tag{10}$$

Substituting for material law yields:

$$\delta W_I = \int_V E\varepsilon \delta \varepsilon dV \tag{11}$$

Substituting for kinematics yields:

$$\delta W_I = \int_V Ez^2 u''(x,t)\delta u''(x) dV \tag{12}$$

$$\delta W_I = \int_0^L EI(x) u''(x,t)\delta u''(x) dx \tag{13}$$

Substituting Equations (9) and (13) in Equation (6) yields:

$$\int_0^L EI(x) u''(x,t)\delta u''(x) dx = -\left(\int_0^L m(x)[\ddot{u}(x,t) + \ddot{u}_g(t)]\delta u(x) dx \right) \tag{14}$$

This is the boundary value problem for a beam vibrating in traverse direction. The problem is discretized using the shape functions. The beam shown in Figure 2.3 has two nodes. Node 1 has the coordinates (0,0) and node 2 has the coordinates (L,0). The beam has four degrees of freedom, two at each node. v_1 and v_2 are the translational degrees of freedom and θ_1 and θ_2 are the rotational degrees of freedom at node 1 and node 2 respectively. The displaced shape is described using the shape functions, ***N***, and degrees of freedom, ***z***, as:

$$u(x) = \mathbf{Nz} = \begin{bmatrix} N_1(x) & N_2(x) & N_3(x) & N_4(x) \end{bmatrix} \begin{bmatrix} v_1 \\ \theta_1 \\ v_2 \\ \theta_2 \end{bmatrix} \tag{15}$$

where

$$N_1(x) = \frac{2x^3}{L^3} - \frac{3x^2}{L^2} + 1 \tag{16}$$

$$N_2(x) = -\frac{x^3}{L^2} + \frac{2x^2}{L} - x \tag{17}$$

$$N_3(x) = -\frac{2x^3}{L^3} + \frac{3x^2}{L^2} \tag{18}$$

$$N_4(x) = -\frac{x^3}{L^2} + \frac{x^2}{L^2} \tag{19}$$

In order to represent the virtual work equation in terms of the shape functions and the displacement vector the following terms are obtained:

$$u''(x,t) = \mathbf{N}''\mathbf{z}(t) \tag{20}$$

$$\ddot{u}(x,t) = \mathbf{N}\ddot{\mathbf{z}}(t) \tag{21}$$

$$\delta u(x) = \mathbf{N}\delta\mathbf{z} \tag{22}$$

$$\delta u''(x) = \mathbf{N}''\delta\mathbf{z} \tag{23}$$

Substituting the derived terms in Equation (14) yields:

$$\delta\mathbf{z}\left[\mathbf{z}\int_0^L EI(x)\mathbf{N}''^T\mathbf{N}''dx\right] = -\delta\mathbf{z}\left(\ddot{\mathbf{z}}\int_0^L m(x)\mathbf{N}^T\mathbf{N}dx + u_g(t)\int_0^L m(x)\mathbf{N}dx\right) \tag{24}$$

Rearranging,

$$\delta\mathbf{z}\left[\ddot{\mathbf{z}}\underbrace{\int_0^L m(x)\mathbf{N}^T\mathbf{N}dx}_{Mass\ Matrix}\right] + \delta\mathbf{z}\left[\mathbf{z}\underbrace{\int_0^L EI(x)\mathbf{N}''^T\mathbf{N}''dx}_{Stiffness\ Matrix}\right] = \delta\mathbf{z}\left[\underbrace{-\ddot{u}_g(t)\int_0^L m(x)\mathbf{N}dx}_{External\ Load\ Vector}\right] \tag{25}$$

As δ**z** is arbitrary, dropping it from the equation yields:

$$\mathbf{M}\ddot{\mathbf{z}} + \mathbf{K}\mathbf{z} = \mathbf{F}(t) \tag{26}$$

The consistent mass matrix when the mass of the beam is uniform (i.e., m(x) = *m*) is:

$$\mathbf{M} = \int_0^L m(x)\mathbf{N}^T\mathbf{N}dx = \frac{mL}{420}\begin{bmatrix} 156 & 22L & 54 & -13L \\ 22L & 4L^2 & 13L & -3L^2 \\ 54 & 13L & 156 & -22L \\ -13L & -3L^2 & -22L & 4L^2 \end{bmatrix} \tag{27}$$

The stiffness matrix when *EI* is constant throughout the beam is:

$$\mathbf{K} = \int_0^L EI(x)\mathbf{N}''^T\mathbf{N}''dx = \frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L \\ 6L & 4L^2 & -6L & 2L^2 \\ -12 & -6L & 12 & -6L \\ 6L & 2L^2 & -6L & 4L^2 \end{bmatrix} \tag{28}$$

When the beam is subjected to a ground acceleration, $\ddot{u}_g(t)$, the external load vector is:

$$\mathbf{F}(t) = -\ddot{u}_g(t)\int_0^L m(x)\mathbf{N}dx = mL\begin{bmatrix} \frac{-\ddot{u}_g(t)}{2} \\ \frac{\ddot{u}_g(t)L}{12} \\ \frac{-\ddot{u}_g(t)}{2} \\ \frac{-\ddot{u}_g(t)L}{12} \end{bmatrix} \tag{29}$$

For a 3D frame element as shown in Figure 2.4 the mass matrix, the stiffness matrix and the external load vector is derived similar to that of a 2D frame element. The beam has a uniformly distributed mass of *m(x)* per unit length, flexural rigidity EI_y, and EI_z, torsional rigidity *GJ*, and axial rigidity *EA*. *Ix* and *Iy* are the moment of inertia, and *J* is the polar moment of inertia respectively. The beam has two nodes, each with six degrees of freedom. v_1, v_2 and v_3 are the translational degrees of freedom and θ_4, θ_5 and θ_6 are the rotational degrees of freedom at node 1. v_7, v_8 and v_9 are the translational degrees of freedom and θ_{10}, θ_{11} and θ_{12} are the rotational degrees of freedom at node 2. The subscript resembles the degree of freedom shown in the Figure 2.4.

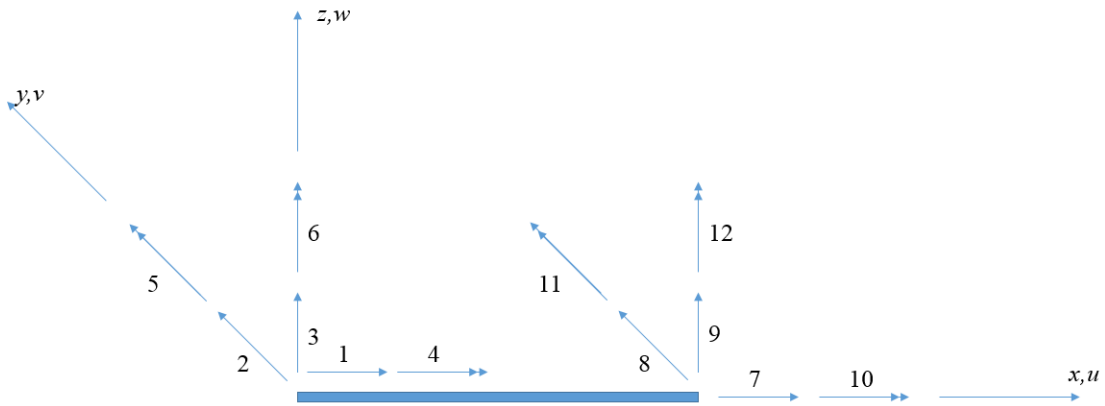

Figure 2.4: 3D Frame element.

The beam deformation is discretized as flexure in the two traverse direction, axial deformation and deformation due to torsion. The shape functions, not shown here are arrived for each of the cases. Integration of the shape functions results in the following stiffness matrix:

$$\mathbf{K} = \begin{bmatrix}
\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} & 0 & -\frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} \\
0 & 0 & \frac{12EI_y}{L^3} & 0 & -\frac{6EI_y}{L^2} & 0 & 0 & 0 & -\frac{12EI_y}{L^3} & 0 & -\frac{6EI_y}{L^2} & 0 \\
0 & 0 & 0 & \frac{GJ}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{GJ}{L} & 0 & 0 \\
0 & 0 & -\frac{6EI_y}{L^2} & 0 & \frac{4EI_y}{L} & 0 & 0 & 0 & \frac{6EI_y}{L^2} & 0 & \frac{2EI_y}{L} & 0 \\
0 & \frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L} & 0 & -\frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{2EI_z}{L} \\
\frac{-EA}{L} & 0 & 0 & 0 & 0 & 0 & \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{-12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} & 0 & \frac{12EI_z}{L^3} & 0 & 0 & 0 & -\frac{6EI_z}{L^2} \\
0 & 0 & \frac{-12EI_y}{L^3} & 0 & \frac{6EI_y}{L^2} & 0 & 0 & 0 & \frac{12EI_y}{L^3} & 0 & \frac{6EI_y}{L^2} & 0 \\
0 & 0 & 0 & \frac{-GJ}{L} & 0 & 0 & 0 & 0 & 0 & \frac{GJ}{L} & 0 & 0 \\
0 & 0 & -\frac{6EI_y}{L^2} & 0 & \frac{2EI_y}{L} & 0 & 0 & 0 & \frac{6EI_y}{L^2} & 0 & \frac{4EI_y}{L} & 0 \\
0 & \frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{2EI_z}{L} & 0 & -\frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L}
\end{bmatrix} \quad (30)$$

Following is the consistent mass matrix upon integration of the shape functions:

$$\mathbf{M} = \frac{mL}{420}\begin{bmatrix} 140 & 0 & 0 & 0 & 0 & 0 & 70 & 0 & 0 & 0 & 0 & 0 \\ 0 & 156 & 0 & 0 & 0 & 22L & 0 & 54 & 0 & 0 & 0 & -13L \\ 0 & 0 & 156 & 0 & -22L & 0 & 0 & 0 & 54 & 0 & 13L & 0 \\ 0 & 0 & 0 & \frac{140J}{A} & 0 & 0 & 0 & 0 & 0 & \frac{70J}{A} & 0 & 0 \\ 0 & 0 & -22L & 0 & 4L^2 & 0 & 0 & 0 & -13L & 0 & -13L^2 & 0 \\ 0 & 22L & 0 & 0 & 0 & 4L^2 & 0 & 13L & 0 & 0 & 0 & -3L^2 \\ 70 & 0 & 0 & 0 & 0 & 0 & 140 & 0 & 0 & 0 & 0 & 0 \\ 0 & 54 & 0 & 0 & 0 & 13L & 0 & 156 & 0 & 0 & 0 & -22L \\ 0 & 0 & 54 & 0 & -13L & 0 & 0 & 0 & 156 & 0 & 22L & 0 \\ 0 & 0 & 0 & \frac{70J}{A} & 0 & 0 & 0 & 0 & 0 & \frac{140J}{A} & 0 & 0 \\ 0 & 0 & 13L & 0 & -3L^2 & 0 & 0 & 0 & 22L & 0 & 4L^2 & 0 \\ 0 & -13L & 0 & 0 & 0 & -3L^2 & 0 & -22L & 0 & 0 & 0 & 4L^2 \end{bmatrix} \quad (31)$$

It is common in dynamic analyses that the mass matrix is considered as a diagonal matrix. That is, the elements other than the diagonal elements of the matrix are set to zero. But in *Rts*, a consistent mass matrix is implemented so that the contribution due to all the elements of the mass matrix is considered. It is identified that for the structures with a short fundamental time period the contribution due to degrees of freedom other than the translational degrees of freedom are significant.

The ground motion acceleration is represented as $\ddot{u}_{gx}(t)$, $\ddot{u}_{gy}(t)$ *and* $\ddot{u}_{gz}(t)$ where the subscript '*g*' refers to a ground motion and '*x*, *y* or *z*' represents the direction of ground motion. The integration of the shape functions according to the Equation (25) yields the following load vector:

$$\mathbf{F}(t) = -\frac{mL}{2}\begin{bmatrix} \ddot{u}_{gx}(t) \\ \ddot{u}_{gy}(t) \\ \ddot{u}_{gz}(t) \\ \ddot{u}_{gx}(t) \\ \frac{-L\ddot{u}_{gy}(t)}{6} \\ \frac{L\ddot{u}_{gz}(t)}{6} \\ \ddot{u}_{gx}(t) \\ \ddot{u}_{gy}(t) \\ \ddot{u}_{gz}(t) \\ \ddot{u}_{gx}(t) \\ \frac{L\ddot{u}_{gy}(t)}{6} \\ \frac{-L\ddot{u}_{gz}(t)}{6} \end{bmatrix} \quad (32)$$

Usually the ground motions are applied along one direction. In that case the elements of force vector corresponding to the directions other than the one along which the acceleration is applied will become zero. In *Rts*, this three dimensional finite element formulation for a beam element is implemented in a class called *RLinearFrameElement*.

2.2.2 Damping Matrix

For a SDOF system, the damping is idealized as a dashpot and the damping coefficient is a single value proportional to the contributions from different damping mechanisms. For a structure, few of the possible energy dissipation mechanisms are repeated straining of the material, friction between the connecting members, and opening and closing of the cracks. Hence, due to different mechanisms of damping it is not possible to define a damping constant based on dimensions of the structure or the structural member sizes. In order to define a damping matrix, modal damping ratio is used. The Rayleigh damping is one such procedure to determine the damping matrix using modal damping ratio. Rayleigh damping matrix is defined as the sum of mass-proportional damping and stiffness-proportional damping given as (Chopra 2012):

$$\mathbf{C} = a_0\mathbf{M} + a_1\mathbf{K} \tag{33}$$

where, the constants a_0 and a_1 have the units of sec^{-1} and sec respectively. The damping ratio for n^{th} mode of vibration is (Chopra 2012):

$$\zeta_n = \frac{a_0}{2}\frac{1}{\omega_n} + \frac{a_1}{2}\omega_n \tag{34}$$

If the ith and jth modal damping ratios are known then the coefficients a_0 and a_1 are determined by solving the following equation:

$$\frac{1}{2}\begin{bmatrix} \frac{1}{\omega_i} & \omega_i \\ \frac{1}{\omega_j} & \omega_j \end{bmatrix}\begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \begin{Bmatrix} \zeta_i \\ \zeta_j \end{Bmatrix} \tag{35}$$

Figure 2.5 shows an example of Rayleigh damping for the case when the damping ratio corresponding to 0.05 and 0.5 sec is 0.05. For this case the value of coefficients are a_0=1.14 sec^{-1} and a_1=0.000723 sec.

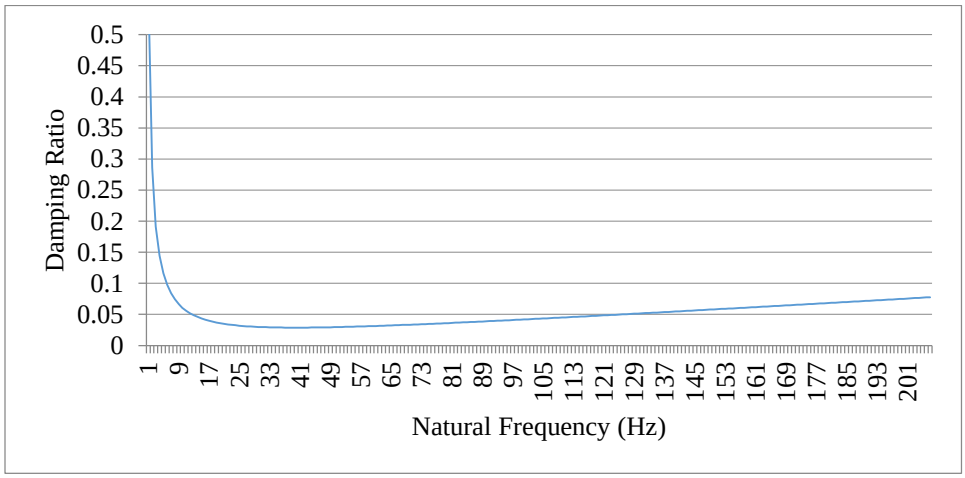

Figure 2.5: Rayleigh damping.

The Rayleigh damping is implemented in *Rts* to compute the damping matrices of the structures.

2.3 Time-stepping Algorithm

Analytical solutions for a MDOF system are not possible for excitations like the earthquake ground motions which vary arbitrarily with time. In such cases numerical methods of integration are adopted for solving the equation of motion. The Newmark time stepping method is one such method, it was developed by Newmark based on the following equations for a SDOF system (Newmark 1959):

$$\dot{u}_{i+1} = \dot{u}_i + [(1-\gamma)\Delta t]\ddot{u}_i + (\gamma\Delta t)\ddot{u}_{i+1} \tag{36}$$

$$u_{i+1} = u_i + (\Delta t)\dot{u}_i + [(0.5-\beta)(\Delta t)^2]\ddot{u}_i + [\beta(\Delta t)^2]\ddot{u}_{i+1} \tag{37}$$

where $\ddot{u}_i$, $\dot{u}_i$ and u_i are the acceleration, velocity and displacement respectively at time step i and $\ddot{u}_{i+1}$, $\dot{u}_{i+1}$ and u_{i+1} are the acceleration, velocity and displacement respectively at time step i+1.

The parameters γ and β define the variation of acceleration over a time step. Typically γ = 1/2 and $1/6 \leq \beta \leq 1/4$. For example, Figure 2.6 depicts the case when the acceleration over the time step τ is constant, which is expressed as:

$$\ddot{u}(\tau) = \frac{1}{2}(\ddot{u}_{i+1} + \ddot{u}_i) \tag{38}$$

For this case, the acceleration over the time step, τ, is integrated to arrive on the velocity and the velocity is integrated over the time step to arrive on the displacement, which is given as:

$$\dot{u}(\tau) = \dot{u}_i + \frac{\tau}{2}(\ddot{u}_{i+1} + \ddot{u}_i) \tag{39}$$

$$u(\tau) = u_i + \dot{u}_i\tau + \frac{\tau^2}{4}(\ddot{u}_{i+1} + \ddot{u}_i) \tag{40}$$

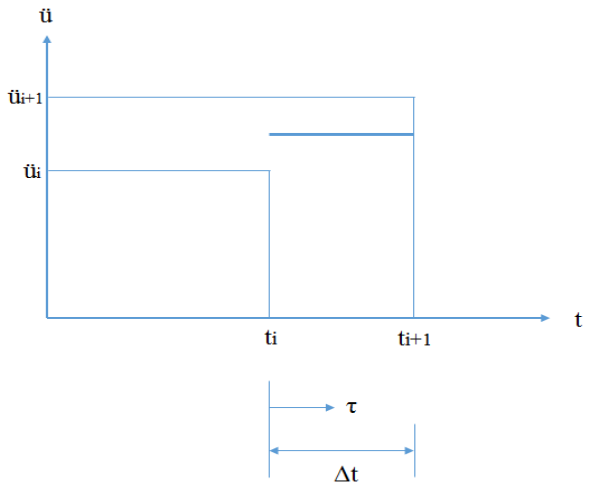

Figure 2.6: Constant average acceleration.

When the time step $\tau = \Delta t$, then:

$$\dot{u}_{i+1} = \dot{u}_i + \frac{\Delta t}{2}(\ddot{u}_{i+1} + \ddot{u}_i) \tag{41}$$

$$u_{i+1} = u_i + \dot{u}_i \Delta t + \frac{(\Delta t)^2}{4}(\ddot{u}_{i+1} + \ddot{u}_i) \tag{42}$$

On comparing Equations (36), (37), (41) and (42) it is concluded that the constant average acceleration is a special case when $\gamma = 1/2$ and $\beta = 1/4$.

The Newmark system of Equations (36) and (37) are modified to express $\ddot{u}_{i+1}$ and $\dot{u}_{i+1}$ in terms of u_{i+1}:

$$\ddot{u}_{i+1} = \frac{1}{\beta(\Delta t)^2}(u_{i+1} - u_i) - \frac{1}{\beta \Delta t}\dot{u}_i - \left(\frac{1}{2\beta} - 1\right)\ddot{u}_i \tag{43}$$

$$\dot{u}_{i+1} = \frac{\gamma}{\beta \Delta t}(u_{i+1} - u_i) + \left(1 - \frac{\gamma}{\beta}\right)\dot{u}_i + \Delta t\left(1 - \frac{\gamma}{2\beta}\right)\ddot{u}_i \tag{44}$$

For a linear system, the equation of motion at time step t+1 is given as:

$$M\ddot{u}_{i+1} + C\dot{u}_{i+1} + Ku_{i+1} = P_{i+1} \tag{45}$$

where M is the mass, C is the damping, K is the stiffness and P_{i+1} is the external load at time step i+1.

Substituting Equations (43) and (44) in Equation (45) and rearranging yields:

$$\hat{K}u_{i+1} = \hat{F}_{i+1} \tag{46}$$

where

$$\hat{K} = K + \frac{\gamma}{\beta\Delta t}C + \frac{1}{\beta(\Delta t)^2}M \tag{47}$$

and

$$\hat{F}_{i+1} = P_{i+1} + \left[\frac{1}{\beta(\Delta t)^2}M + \frac{\gamma}{\beta\Delta t}C\right]u_i + \left[\frac{1}{\beta\Delta t}M + \left(\frac{\gamma}{\beta} - 1\right)C\right]\dot{u}_i + \left[\left(\frac{1}{2\beta} - 1\right)M + \Delta t\left(\frac{\gamma}{2\beta} - 1\right)C\right]\ddot{u}_i \tag{48}$$

The displacement at time step i+1 is computed from the following expression:

$$u_{i+1} = \frac{\hat{F}_{i+1}}{\hat{K}_{i+1}} \tag{49}$$

The acceleration and velocity at time step i+1 are determined by substituting the value of the displacement at i+1 in Equations (43) and (44).

For an inelastic system the term Ku_{i+1} in Equation (45) is replaced with the resisting force $(f_s)_{i+1}$ to give:

$$P_{i+1} = M\ddot{u}_{i+1} + C\dot{u}_{i+1} + (f_s)_{i+1} \tag{50}$$

As the force $(f_s)_{i+1}$ is a nonlinear function of u_{i+1}, it is required to adopt an iterative method like Newton-Raphson or modified Newton-Raphson to compute the force at each time instant. Without getting into detail, a short description of the modified Newton-Raphson procedure for computing the response is described. In the modified Newton-Raphson

method, the initial stiffness is considered instead of the tangent stiffness for determination of the increment in displacement, Δu, at every iteration j, at each time instant from i to $i+1$. The first step is to initialize the displacement, u, and the force, f_S, as follows:

$$u_{i+1}^{(1)} = u_i \tag{51}$$

$$f_S^{(1)} = (f_S)_i \tag{52}$$

Once the values are initialized, the following equation for the residual, R_{j+1}, is arrived by considering the equation of motion at every iteration (Chopra 2012):

$$R_{i+1}^{(j)} = \hat{F}_{i+1} - (f_S)_{i+1}^{(j)} - \left(\frac{1}{\beta(\Delta t)^2} M + \frac{\gamma}{\beta \Delta t} C \right) u_{i+1}^{(j)} \tag{53}$$

The increment in displacement, Δu, is computed as follows:

$$\Delta u^{(j)} = \frac{R_{i+1}^{(j)}}{\hat{K}_T} \tag{54}$$

Then the displacement is computed as follows:

$$u_{i+1}^{(j+1)} = u_{i+1}^{(j)} + \Delta u^{(j)} \tag{55}$$

For the determined value of displacement, $(f_S)_{i+1}$ is computed. The above steps are repeated until Δu or R becomes small enough, which is less than the tolerance limit, for instance 10^{-5}.

For a MDOF system the scalar displacement, velocity, acceleration, and external load are replaced by vectors, with dimension equal to the number of degrees of freedom. Similarly, the mass, damping, and stiffness in the equation of motion are replaced by matrices for a MDOF case, to give the following equation of motion for linear problems:

$$\mathbf{M} \bullet \ddot{\mathbf{u}}_{i+1} + \mathbf{C} \bullet \dot{\mathbf{u}}_{i+1} + \mathbf{K} \bullet \mathbf{u}_{i+1} = \mathbf{P}_{i+1} \tag{56}$$

Substituting the vector form of Equations (43) and (44) in Equation (56) and rearranging:

$$\hat{\mathbf{K}} \bullet \mathbf{u}_{i+1} = \hat{\mathbf{F}}_{i+1} \tag{57}$$

where

$$\hat{\mathbf{K}} = \mathbf{K} + \mathbf{a_1} \tag{58}$$

and

$$\hat{\mathbf{F}}_{i+1} = \mathbf{P}_{i+1} + \mathbf{a_1} \bullet \mathbf{u}_i + \mathbf{a_2} \bullet \dot{\mathbf{u}}_i + \mathbf{a_3} \bullet \ddot{\mathbf{u}}_i \tag{59}$$

$$\mathbf{a_1} = \frac{1}{\beta(\Delta t)^2}\mathbf{M} + \frac{\gamma}{\beta \Delta t}\mathbf{C} \tag{60}$$

$$\mathbf{a_2} = \frac{1}{\beta \Delta t}\mathbf{M} + \left(\frac{\gamma}{\beta} - 1\right)\mathbf{C} \tag{61}$$

$$\mathbf{a}_3 = \left(\frac{1}{2\beta} - 1\right)\mathbf{M} + \Delta t\left(\frac{\gamma}{2\beta} - 1\right)\mathbf{C} \tag{62}$$

The displacement at time step i+1 is computed as follows:

$$\mathbf{u}_{i+1} = \hat{\mathbf{K}}_{i+1}^{-1} \bullet \hat{\mathbf{F}}_{i+1} \tag{63}$$

Once the displacement vector is computed, the acceleration and velocity vectors are determined as follows:

$$\ddot{\mathbf{u}}_{i+1} = \frac{1}{\beta(\Delta t)^2}(\mathbf{u}_{i+1} - \mathbf{u}_i) - \frac{1}{\beta \Delta t}\dot{\mathbf{u}}_i - \left(\frac{1}{2\beta} - 1\right)\ddot{\mathbf{u}}_i \tag{64}$$

$$\dot{\mathbf{u}}_{i+1} = \frac{\gamma}{\beta \Delta t}(\mathbf{u}_{i+1} - \mathbf{u}_i) + \left(1 - \frac{\gamma}{\beta}\right)\dot{\mathbf{u}}_i + \Delta t\left(1 - \frac{\gamma}{2\beta}\right)\ddot{\mathbf{u}}_i \tag{65}$$

For an inelastic MDOF system the procedure is similar to the case of an inelastic SDOF system, except that the scalars in Equations (51) to (55) are replaced with matrices and vectors.

The Newmark algorithm presented in this section is well suited for implementation on the

computer. In this thesis, it is implemented in *Rts*. In the following section the software architecture and the implementation of the dynamic analysis is discussed.

2.4 Software Architecture in *Rts* for Dynamic Analysis

Figure 2.7 shows a map of the relevant part of the object-oriented software architecture in *Rts*. Every class in *Rts* starts with the letter *R*. The dynamic analysis is implemented as a subclass of *RStructuralAnalysis*, which is called by the orchestrating class *RCompomentResponseModel*. The *RStructuralAnalysis* class has four subclasses namely *RLinearStaticStructuralAnalysis*, *RInelasticStaticStructuralAnalysis*, *RLinearDynamicStructuralAnalysis*, and *RInelasticDynamicStructuralAnalysis*.

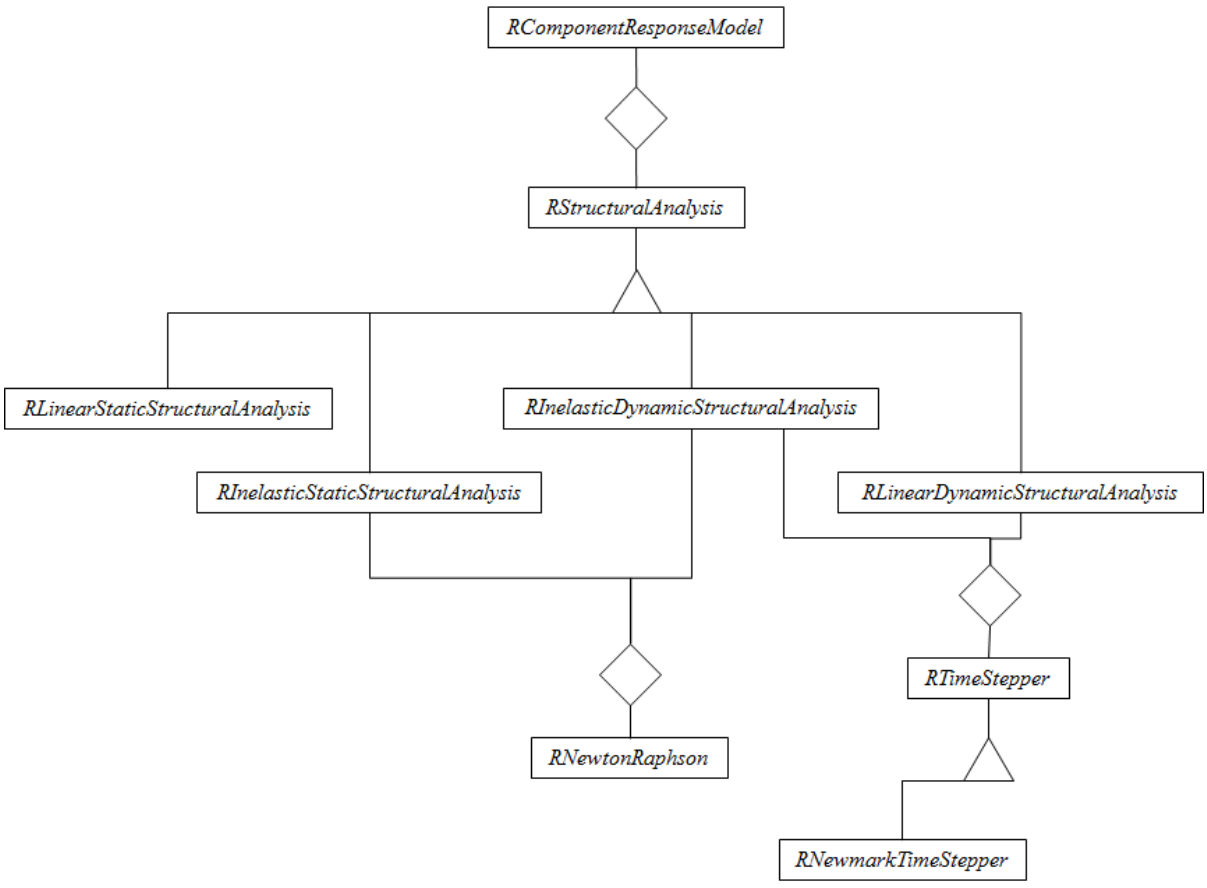

Figure 2.7: Software architecture.

As the names suggest, *RLinearStaticStructuralAnalysis* and *RInealsticStaticStructualAnalysis* perform linear and inelastic static structural analyses. Similarly, *RLinearDynamicStructuralAnalysis* and *RInelasticDynamicStructuralAnalysis* performs linear and inelastic dynamic analyses. All four tools access a tool to assemble the mass matrix, stiffness matrix, damping matrix, internal load vector, and external load vector. Although inelastic analysis is outside the scope of this thesis, it is noted that the *RInelasticStaticStruturalAnalysis* and *RInelasticDynamicStructralAnalysis* use the *RNewtonRaphson* class, which consists of a solver for an inelastic system. The *RLinearDynamicStructuralAnalysis* and *RInelasticDynamicStructuralAnalysis* classes can access *RTimeStepper* tool, which has a subclass called *RNewmarkTimeStepper* to perform time-stepping dynamic analysis. There are other classes which are accessed by all four classes for performing matrix and vector operations, plotting of time series, and for updating the plots, which are not shown in the figure.

In order to understand the dynamic analysis algorithm implemented it is important to be familiar with the user interface of *Rts*. A screenshot of the user interface of *Rts* is shown in Figure 2.8. The objects pane displays the models and tools required for analysis. The objects can be created by right clicking on them or by loading an input file. One way to load the input file is by dragging and dropping the input text file on the user interface of *Rts*. The other method is to click on the *File* menu, and then click on the *Open* option, and then choosing the desired input file. The properties pane displays the properties of the object selected by the user. The output pane displays the messages and output from the model being analysed. The central pane is for visualization purpose, for example to view the deformed shape of the structure being analysed.

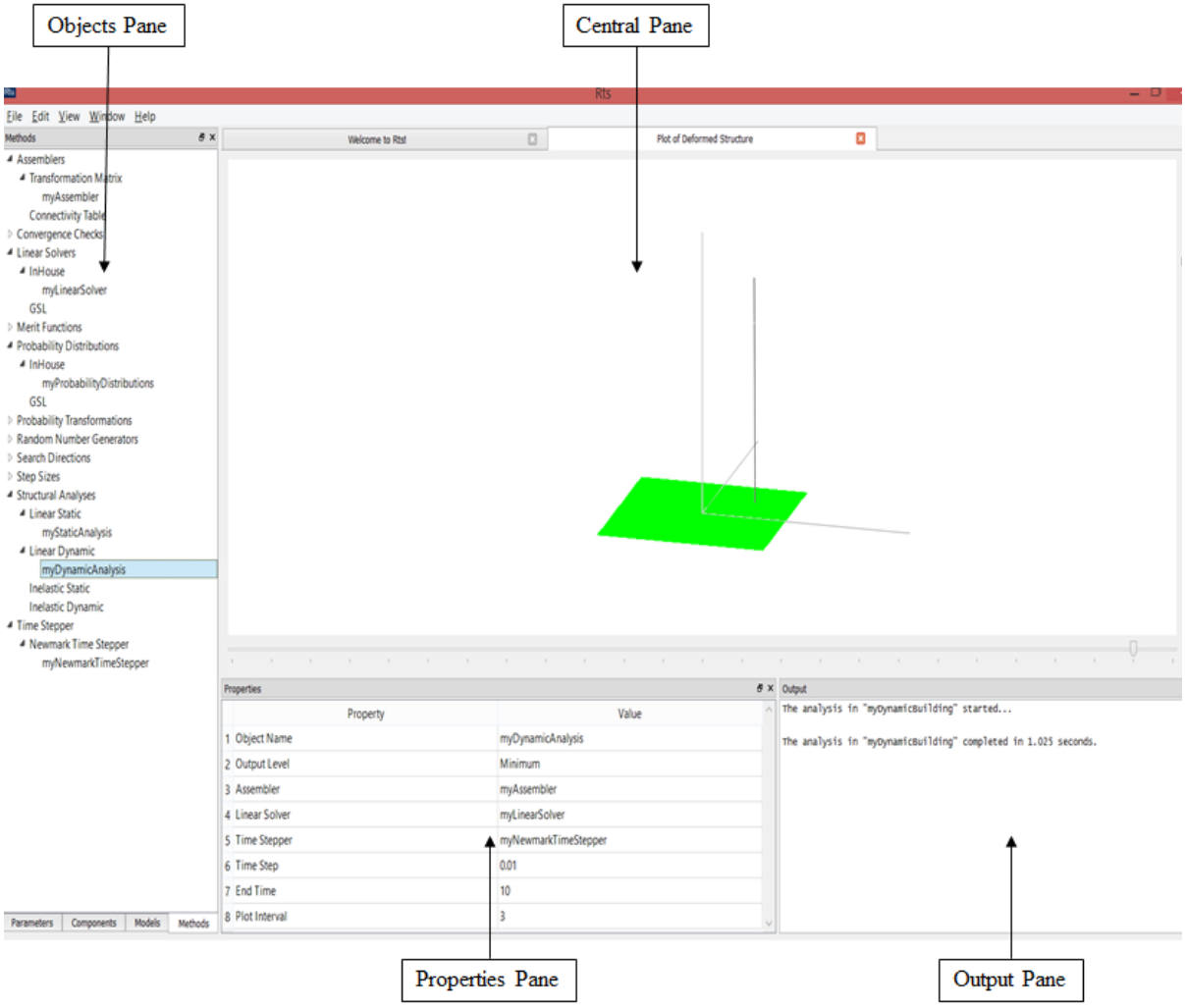

Figure 2.8: A screenshot of the user interface of *Rts*.

A description of implementation of the *RLinearDynamicStructuralAnalysis* class in *Rts* is as follows:

1. A structure is first modelled in *Rts*.
2. A ground motion file is imported which is stored in *Rts* as a time series.
3. Properties such as the *Time Step* and *End Time* are entered by the user in the properties pane of *Rts*, as shown in Figure 2.8. The *Time Step* is the increment in time at which the response of the structure is to be determined. The *End Time* is the time up to which the analysis is to be performed. There are other properties such as the

Plot Interval, *Level of Output*, *Assembler*, *Linear Solver* and *Time Stepper* which are also defined by the user.

4. Once the required properties are obtained from the user, within the class *theAssembler* function is called to get *numDOFs,* which is the number of free degrees of freedom of the structural system under consideration.
5. Memory for *K*, *M*, and *C* matrices are allocated, which are the stiffness, mass and damping matrices respectively.
6. Memory for the *displacement*, *velocity*, *acceleration*, and *PatTimeStepi* vectors are allocated, which are the displacement, velocity, acceleration, and external load vectors at the *i*th time step.
7. Memory for the *dispacementatiplus1*, *velocityatiplus1*, *accelerationatiplus1*, and *PatTimeStepiplus1* vectors are allocated, which are the displacement, velocity, acceleration, and external load vectors respectively at the time step *i*+1.
8. A *for* loop is initiated with the initial time equal to zero and the end time equal to the *End Time*. The time step of the loop is incremented by an amount the *Time Step*.
9. For every time step, the *RNewmarkTimeStepper* class is called to get the response of the structure at the next time step. Within this class, first the constants γ and β are obtained from the user and checked whether the values given by the user are appropriate. In total, eight items are sent to the *RNewmarkTimeStepper* class. The constant matrices similar to those in Equations (58), (60), (61) and (62) are computed. The modified external load vector is also determined according to Equation (58). After performing all the operations for determination of response at the next time step, the new displacement, velocity and acceleration vectors are sent back to the

RLinearDynamicStructuralAnalysis class.

10. At the end of every time step the responses are updated.

Figure 2.9 depicts the summary of organization of the *RLinearDynamicStructuralAnalysis* class.

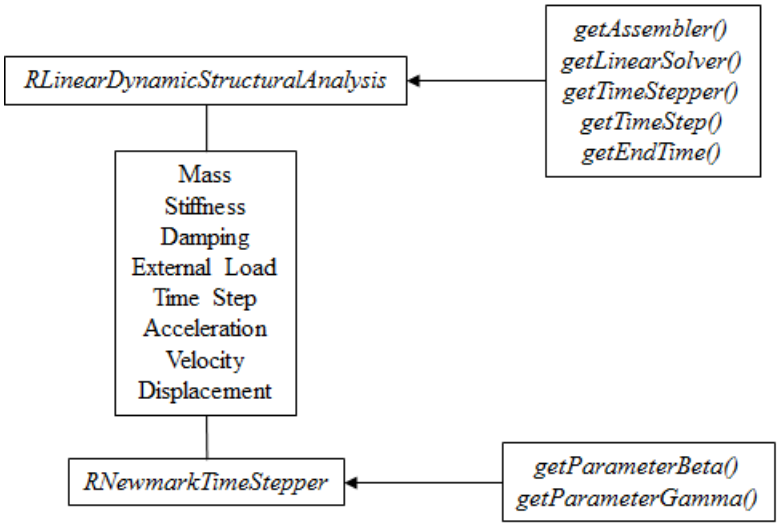

Figure 2.9: *RLinearDynamicStructuralAnalysis*.

The Newmark time stepping part of the *RInelasticDynamicStructuralAnlysis* class is similar to that of the *RLinearDynamicStructuralAnalysis* class. The difference is that for an inelastic analysis, the displacement vector at each time step is determined using the modified Newton-Raphson method described in Section 2.3. The implemented code is included in Appendix A of this thesis.

2.5 Verification of the Implementations in *Rts*

The implemented linear dynamic analysis code is required to be verified. Verification is checking the accuracy and appropriateness of the model implemented. For the same purpose, the response of a cantilever column subjected to a load is compared to the analytical solution. For verifying the dynamic analysis model implemented in *Rts*, a cantilever column (Young's

modulus, E = 2x10^{5}N/mm^2) with uniform I cross-section (I_{xx} = 5.85x10^{6}mm^4) subjected to a load whose analytical solution is available is chosen, see Figure 2.10.

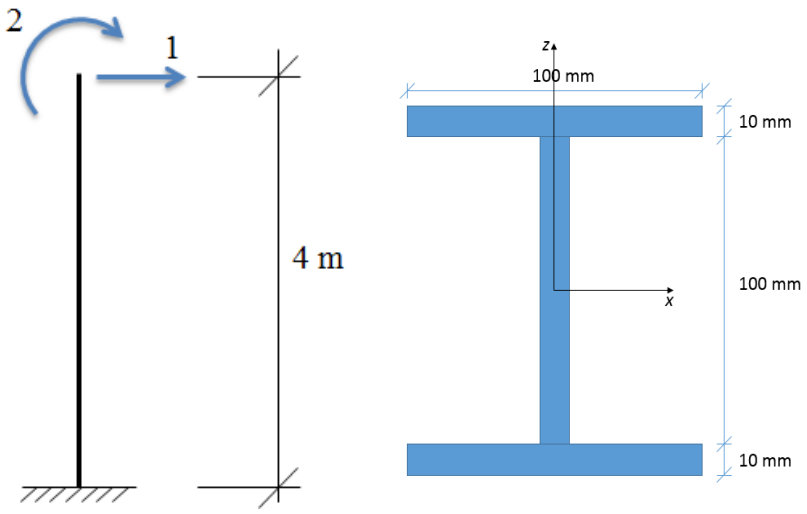

Figure 2.10: Vertical steel column (left) and cross-section (right).

The column is subjected to a constant load, shown in Figure 2.11. This load is applied to the free end of the column. In *Rts*, there are more degrees of freedom (DOF) for this column than those shown in Figure 2.10, but they are out-of-plane DOFs that do not influence the present solution. Thus, for the MDOF system only the degree of freedom 1 and 2 shown in Figure 2.10 are active. By using the expressions for the stiffness and mass matrices derived previously, the column in Figure 2.10 is governed by the following system of equations:

$$\begin{bmatrix} \frac{156mL}{420} & \frac{-22ml^2}{420} \\ \frac{-22ml^2}{420} & \frac{4ml^3}{420} \end{bmatrix} \cdot \ddot{\mathbf{u}} + \left(a_0 \cdot \begin{bmatrix} \frac{156mL}{420} & \frac{-22ml^2}{420} \\ \frac{-22ml^2}{420} & \frac{4ml^3}{420} \end{bmatrix} + a_1 \cdot \begin{bmatrix} \frac{12EI}{L^3} & \frac{-6EI}{L^2} \\ \frac{-6EI}{L^2} & \frac{4EI}{L} \end{bmatrix} \right) \cdot \dot{\mathbf{u}} + \begin{bmatrix} \frac{12EI}{L^3} & \frac{-6EI}{L^2} \\ \frac{-6EI}{L^2} & \frac{4EI}{L} \end{bmatrix} \cdot \mathbf{u} = p(t) \quad (66)$$

where

$$p(t) = 100 \quad (67)$$

The value of damping coefficients are a_0=1.14 sec^{-1} and a_1=0.000723 sec.

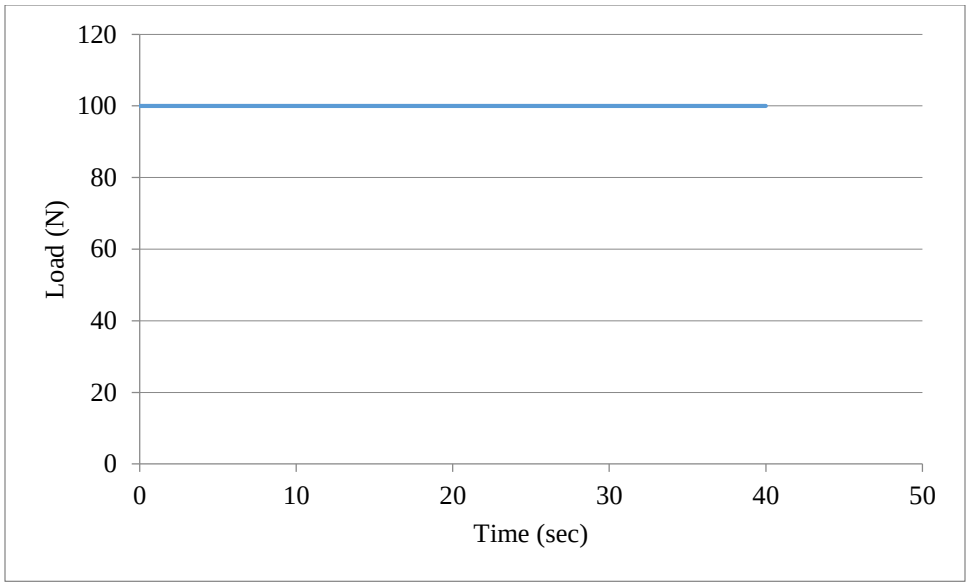

Figure 2.11: Step loading.

For the same column in order to compare the response from *Rts* with the analytical solution it is required to consider it as a SDOF system. This is done by using a technique called static condensation to remove the degree of freedom number 2 without clamping it, yielding the following stiffness:

$$K = \frac{3EI}{L^3} \tag{68}$$

and following mass:

$$M = \frac{33mL}{140} \tag{69}$$

Analytical solutions are available in the literature (Chopra 2012) for SDOF problems subjected to certain types of loading, and these are now compared with the results from the *Rts* algorithm. The solution given for this problem is (Chopra 2012):

$$u(t) = (u_{st})_0 \left[1 - e^{-\zeta\omega_n t} \left(\cos\omega_D t + \frac{\zeta}{\sqrt{1-\zeta^2}} \sin\omega_D t \right) \right] \tag{70}$$

where

$$(u_{st})_0 = \frac{p}{K} \tag{71}$$

where p=100 N for this case.

The analytical solution is shown in Figure 2.12 together with the solution given by the algorithm in Section 2.3 implemented in *Rts*. Table 2.1 shows the displacement values of the *Rts* response and the analytical solution for different peaks of vibration. The last column of the table shows the percentage difference in displacement of the analytical solution and the *Rts* response. Based on the analytical solution and the *Rts* response, the following observations are made:

1. It is observed that the static displacement of the *Rts* response matches with that of the analytical solution. That is, the percentage difference in the *Rts* response and the analytical solution is zero for the static displacement.
2. It is observed that the percentage difference in displacement at different peaks of vibration is minimal. For instance, the difference in displacement for the 15th peak is 0.31%.
3. It is noted that the phase difference increases as the time progresses. For instance, the time difference in occurrence of the 5th, 15th and 25th peaks are 0.01 sec, 0.03 sec and 0.06 sec respectively. It is seen that the phase difference for the 25th cycle is close to 180°, which is a significant difference. The elongation in the natural period of the *Rts* response could be due to error inherent in the numerical solution.
4. The logarithmic decrement of the *Rts* response closely matches with that of the analytical solution. For instance, the logarithmic decrement computed over the 1st and 20th cycle is 0.0315 for the *Rts* response and 0.0322 for the analytical solution. The

difference is 2.02% for this case.

5. The frequency of the SDOF system and the *Rts* MDOF column closely match. For instance, the time period of the structure computed by considering the 1st and 25th cycle, is 0.13 sec for the *Rts* MDOF column and 0.128 sec for the SDOF system. The difference in time period is 1.62%.

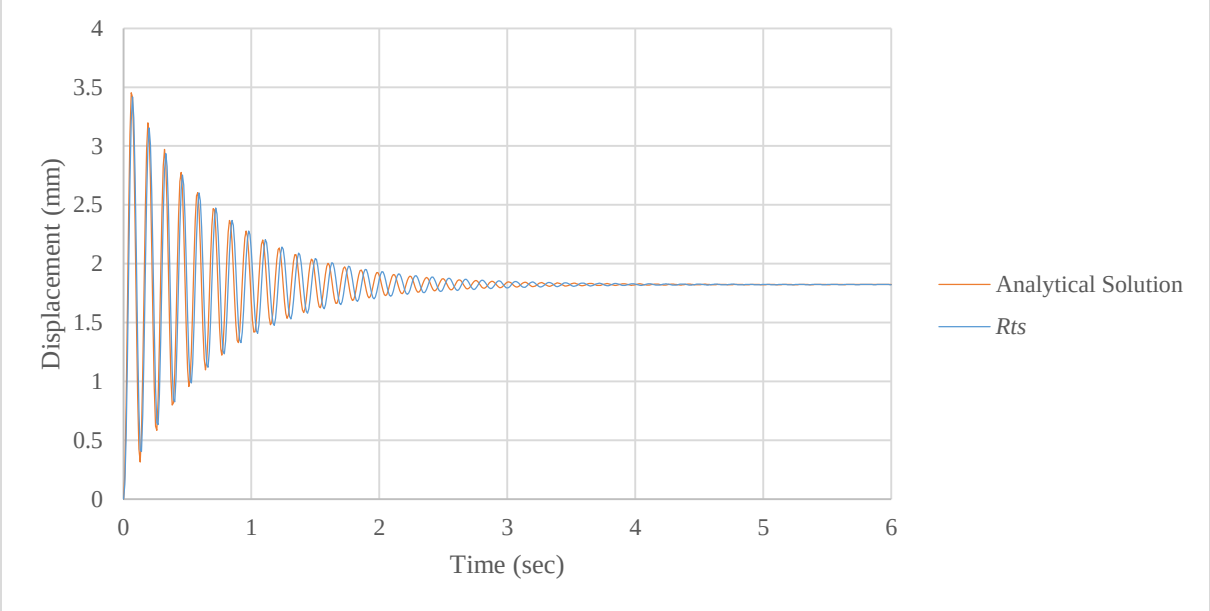

Figure 2.12: Comparison of analytical solution and *Rts* solution.

Table 2.1 Comparison of Analytical solution and *Rts* response.

Peak	*Rts* Response		Analytical Solution		% Difference in Displacement
	Time (sec)	Displacement (mm)	Time (sec)	Displacement (mm)	
1	0.07	3.416	0.06	3.453	1.07
5	0.59	2.602	0.58	2.604	0.08
10	1.24	2.142	1.22	2.134	-0.37
15	1.89	1.953	1.86	1.947	-0.31
20	2.55	1.876	2.5	1.873	-0.16
25	3.2	1.845	3.14	1.843	-0.11

To further verify the algorithm, the *Rts* solution is compared to the response from *OpenSees*

(McKenna et al. 2010), a computer program for modelling and simulating the behavior of structural systems subjected to earthquakes. The response of the cantilever column from *Rts* and *OpenSees* subjected to the step load shown in Figure 2.11 is compared. For this verification purpose, a lumped mass matrix is used in *Rts* instead of a consistent mass matrix, which was previously derived in Section 2.2.1. In a lumped mass matrix, half of the mass of the column is lumped at each node along the translational degrees of freedom. The elements of the mass matrix other than the one along the diagonal are set to zero. This is done to compare the *Rts* analysis with the exact analysis that would have the same value for the mass matrix in *OpenSees*. The Newmark parameters and the damping constants in *OpenSees* are set to the same values as the ones used for the *Rts* analysis. Figure 2.13 shows the result obtained from *OpenSees* and *Rts*. The response obtained from *Rts* overlaps with the *OpenSees* response, i.e. the percentage difference in response is zero. Thus, based on the analysis the implemented code is verified for this example. In the following section the dynamic analysis algorithm for a structure subjected to a ground motion is demonstrated.

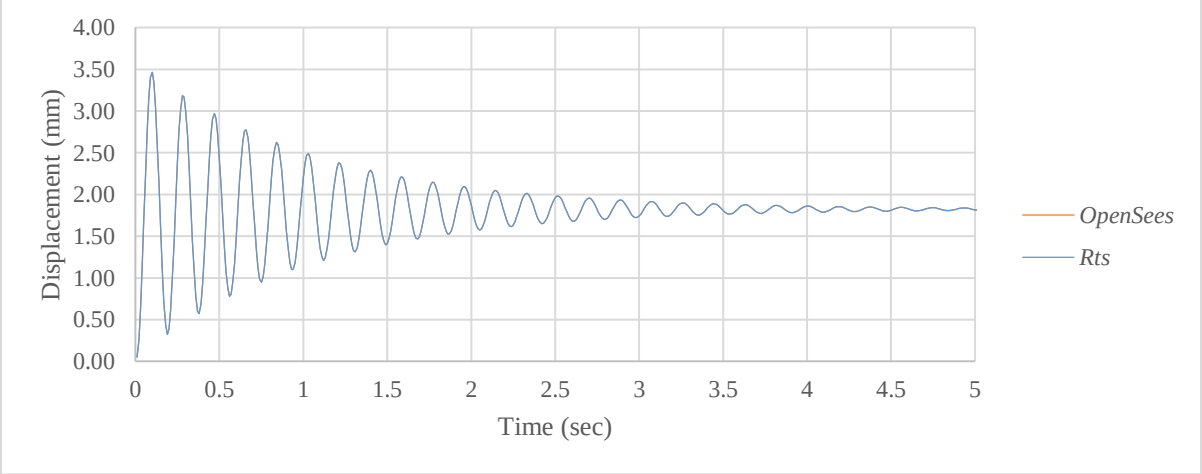

Figure 2.13: Comparison of *OpenSees* solution and *Rts* solution.

2.6 Example 1: Cantilever Column Subjected to Ground Motion

One of the initial steps in *Rts* for performing dynamic analysis using a ground motion is to load a ground motion text file and define it as a time series. There are two ways of loading a ground motion file in *Rts*. First the user needs to go to the *Rts* objects pane and right click on the "*Time Series*" and choose the "*Create*" option. A new time series will be created. Now, it is required to right click on this newly created time series and choose the "*Load File*" option. A dialog box similar to the one shown in Figure 2.14 will pop up. Here the user needs to load the ground motion input file.

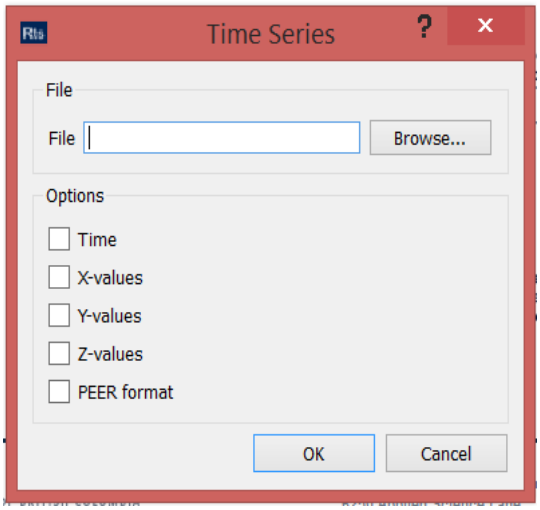

Figure 2.14: Time series dialog box.

If the user wishes to load a file downloaded from the PEER database, which would be a text file, then the user needs to click on the check box corresponding to "*PEER format*" and the check box corresponding to the direction of motion, which could be "*X-values*", "*Y-values*" or "*Z-values*". In case the user wishes to load a ground motion file not downloaded from the PEER database then the user should click on the check boxes corresponding to "*Time*" and the direction of ground motion which is "*X-values*", "*Y-values*" or "*Z-values*". The time series input file has to be similar to the one shown in Figure 2.15. It should have four

columns. The first column is the time, the second column is the acceleration in *x* direction, the third column is the acceleration in *y* direction, and the fourth column is the acceleration in *z* direction.

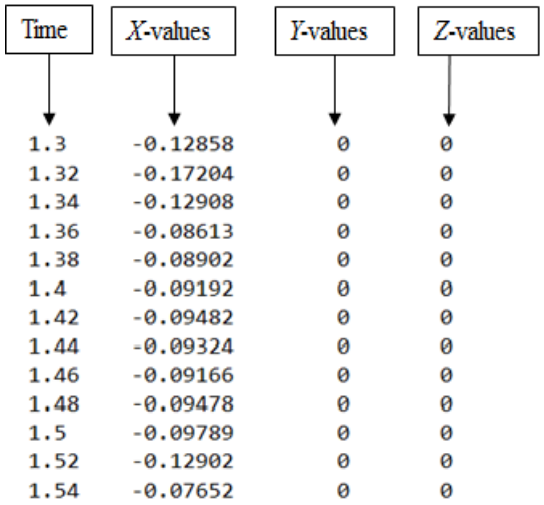

Figure 2.15: Screenshot of a sample ground motion record file.

The "*Time Series*" tool also allows the user to plot an acceleration time series, velocity time series and displacement time series. The velocity time series is computed by integrating the acceleration time series and the displacement time series is computed by integrating the velocity time series. Integration in *Rts* is achieved using the trapezoidal rule of integration.

After loading the ground motion file, the ground motion time series is required to be scaled. The scaling procedure is described in Chapter 3 of this thesis. A structure which is required to be subjected to a ground motion is modelled in *Rts*. For the purpose of demonstration, the cantilever column described in Section 2.5 is subjected to El Centro 1940 (http://nisee.berkeley.edu/data/strong_motion/a.k.chopra/) ground motion, shown in Figure 2.16 . The response of the free end of the column is shown in Figure 2.17

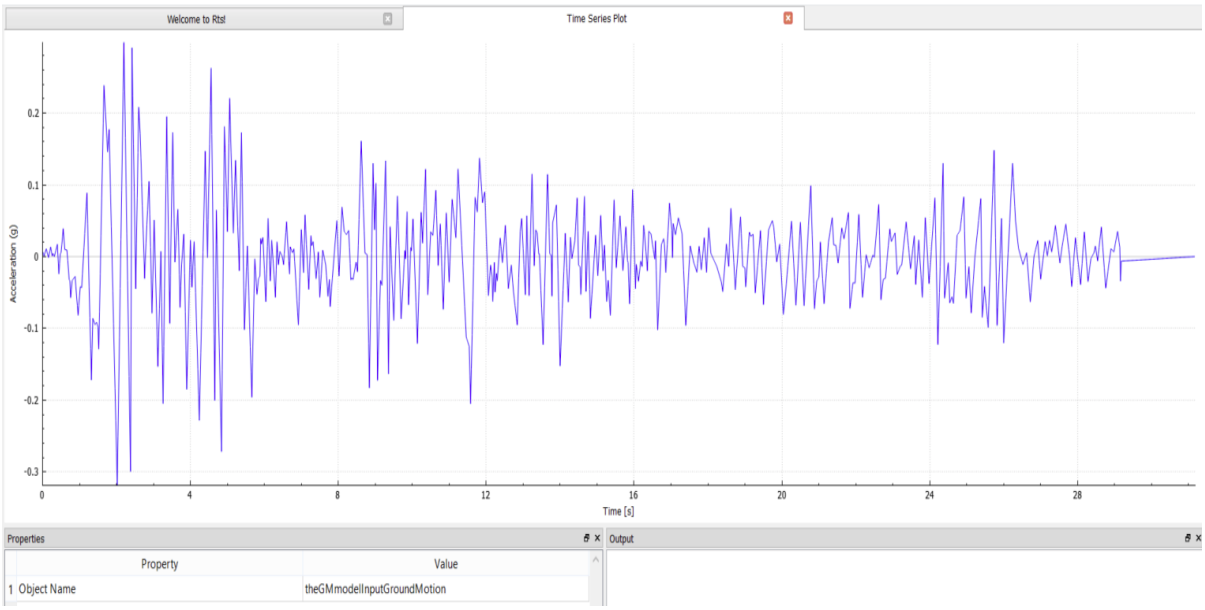

Figure 2.16: El Centro (1940) ground motion record.

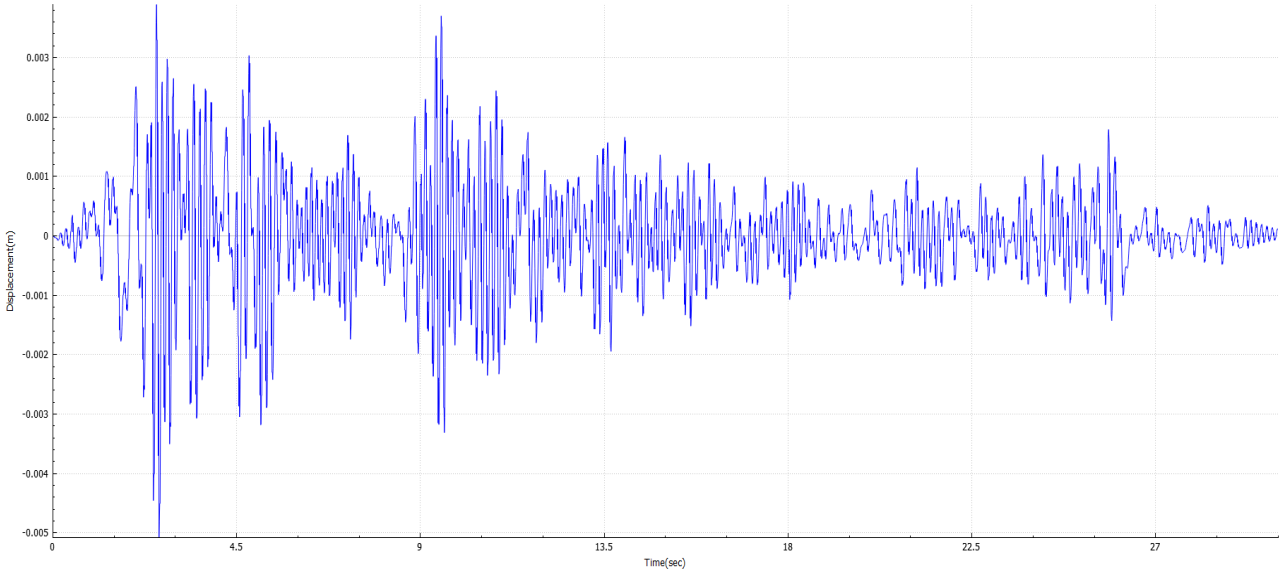

Figure 2.17: Response of the column to El Centro (1940) ground motion.

It is inferred from the Figure 2.17 that the maximum displacement of the column is very small (5 mm). The reason for this is that the column has a short fundamental time period close to 0.13 sec.

2.7 Example 2: One-Storey Building Subjected to Ground Motion

In this section, dynamic analysis of a one storey building subjected to an earthquake ground motion is conducted. The example used is a simple structure, which is a step up to the cantilever column structure used for verification of the time-stepping algorithm. The example used is primarily modelled by Javaherian Yazdi (2015) to capture the visual damage to the structure when subjected to a pushover load. The original model is modified by removing the shear walls and changing the position of columns. The input file for the model is provided in Appendix A.3 of this thesis.

2.7.1 Structural Model

The plan and the elevation of the structure are shown in Figure 2.18, Figure 2.19 and Figure 2.20. The building has 12 columns with spacing along *y* axis between column 1 and column 2 being 8 m and that between column 2 and column 3 4 m. Thus, the building is asymmetric in plan. The columns are modelled as reinforced concrete columns in *Rts* using *RRCColumnComponent*, which is the inelastic component. One end of the column is fixed to a plane defined using a component called *RFixedPlaneComponent* in *Rts*. Each column has a square cross-section of 200 mm. The longitudinal reinforcement ratio for all the columns is defined as 2% of the cross-section area. The other end of the column is connected to a slab component defined in *Rts* as *RRCSlabComponent*. The 1 m thick reinforced concrete slab on the top of the columns is modelled by plane elastic finite elements. For this demonstration a thick slab and slender columns are used to get a reasonable fundamental time period greater than 0.1 sec for the structure. Figure 2.21 shows the screenshot of the isometric view of the building in *Rts*.

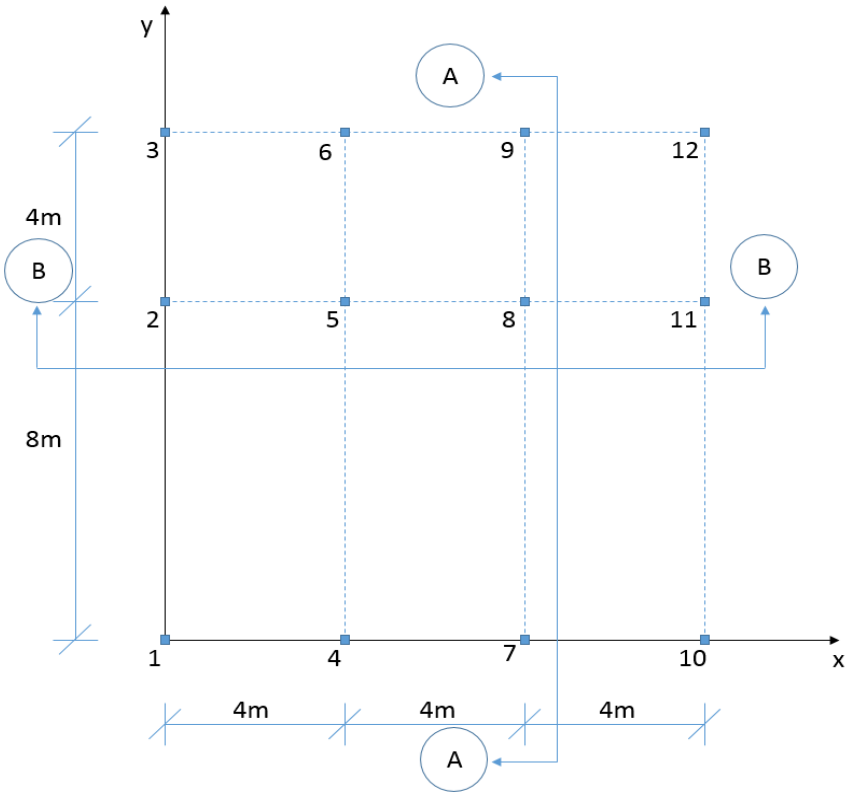

Figure 2.18: Building plan.

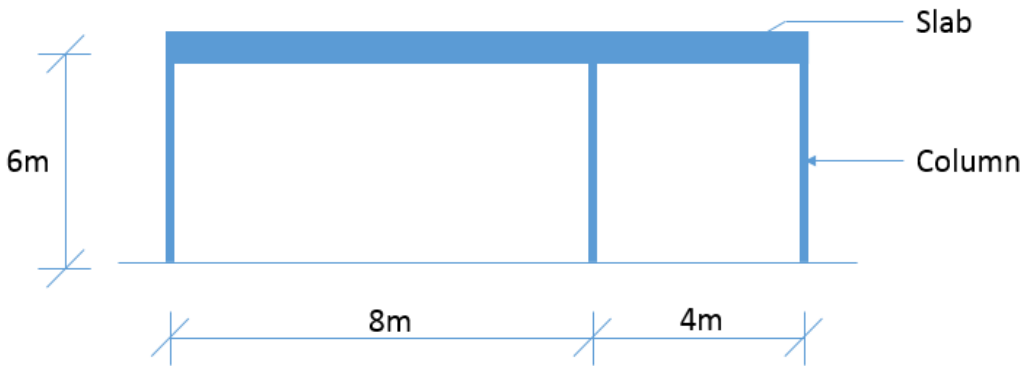

Figure 2.19: Cross-section A-A.

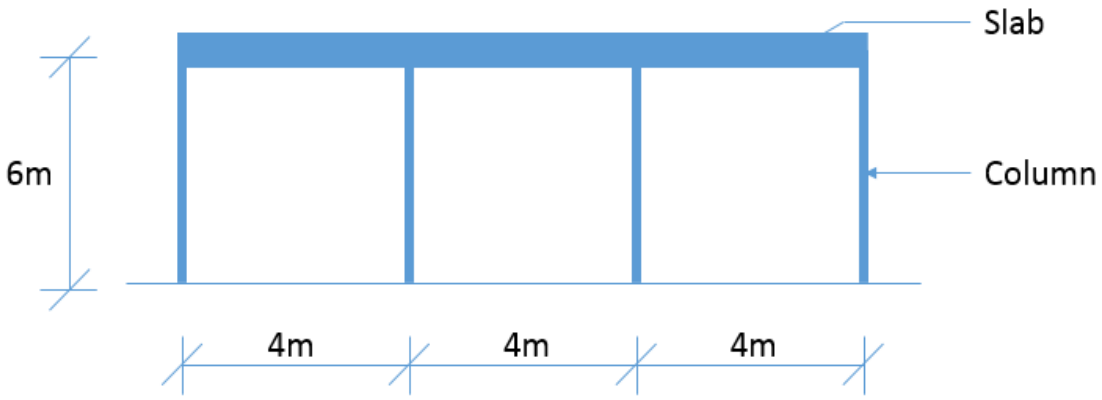

Figure 2.20: Cross-section B-B.

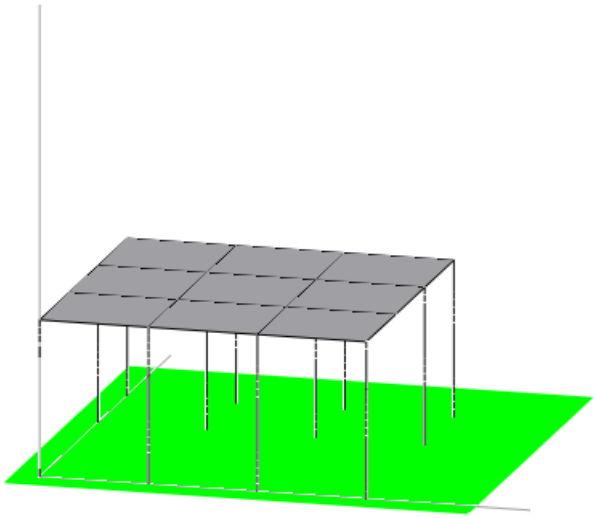

Figure 2.21: Isometric view of the building in *Rts*.

In the concrete columns, a bilinear material model is used for both concrete and steel. This model, which has the same yield strength in tension and compression, is the only inelastic material model that is currently available in *Rts* and is used here for demonstration purpose. The yield strength of the concrete and steel is 40 N/mm^2 and 400 N/mm^2 respectively. The strain hardening ratio, which is the ratio between the modulus of elasticity before yielding

and modulus of elasticity post yielding for both steel and concrete is 0.01. Figure 2.22 and Figure 2.23 shows the material properties of steel and concrete respectively.

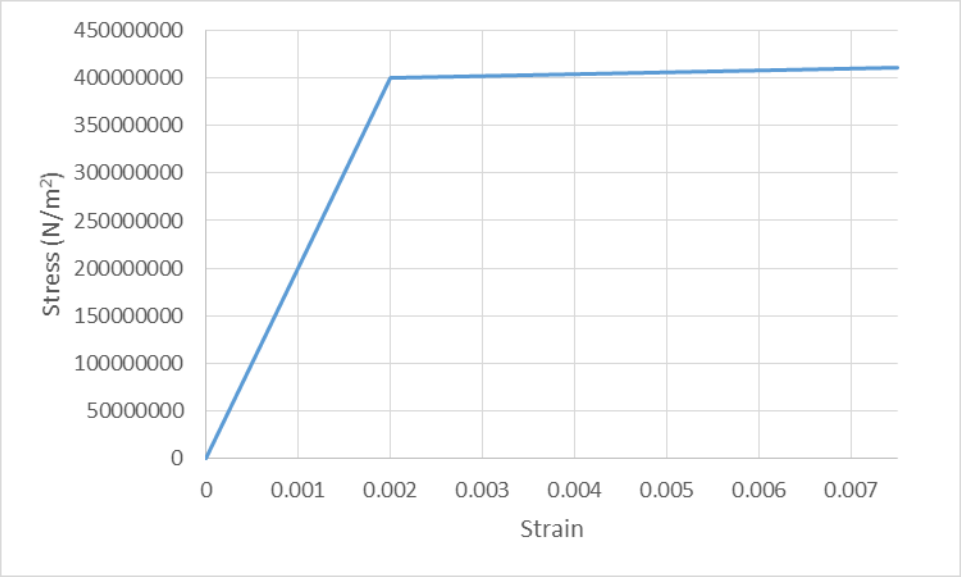

Figure 2.22: Steel material.

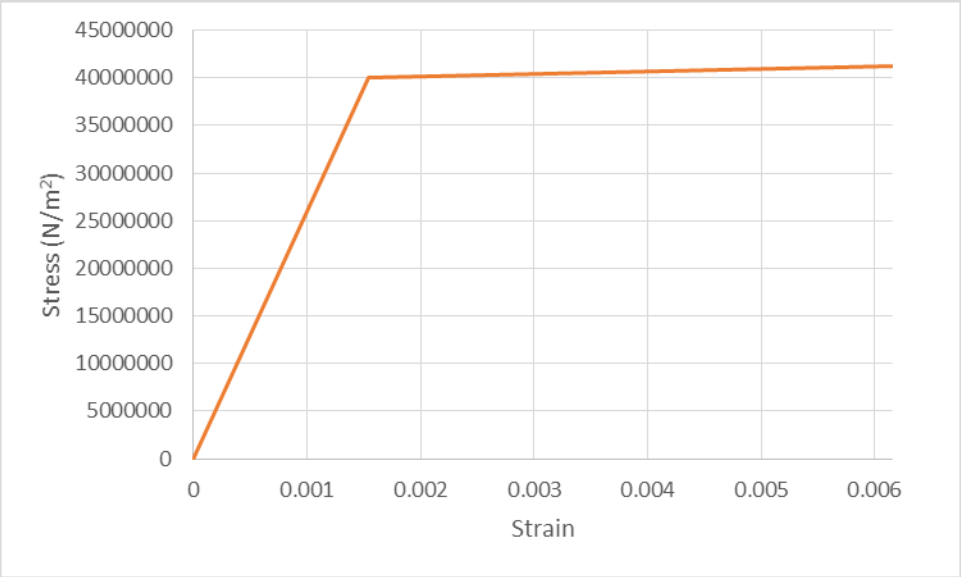

Figure 2.23: Concrete material.

For the columns, the mesh option is given a value of '1'. The mesh option in *Rts* is required to be entered by the user to define the level of meshing. Meshing is performed to divide the

cross-section into discrete fibers. Generally, the level of meshing affects the solution accuracy and computation time. For the slab, mesh option 303 is specified, which means that the slab is divided into 3 by 3 elements, i.e., 9 elements. Table 2.2 to Table 2.6 summarizes the model input parameters for the example building.

Table 2.2 Example building: RC column component

Parameter	Value
Depth	200mm
Width	200mm
Mesh Option	1
Concrete Strength	C30
Longitudinal Reinforcement	2%

Table 2.3 Example building: RC slab component

Parameter	Value
Thickness	1m
Mesh Option	303
Concrete Strength	30000000N/m^2

Table 2.4 Example building: Inelastic dynamic structural analysis

Parameter	Value
Newmark Parameter β	0.25
Newmark Parameter γ	0.5
Time Step	0.01 sec
End Time	35 sec

Table 2.5 Example building: Scaled ground motion model

Parameter	Value
Ground Motion Record	El Centro
Scaling Factor	1.0

Table 2.6 Example building: Component response model

Parameter	Value
Analysis	Earthquake Ground motion
Response	Column 1; Column 2; Column 3

2.7.2 Structural Analysis

The dynamic analysis of the structure is performed by subjecting the structure to the El Centro (1940) ground motion record. The *RScaledGroundMotionModel* class, which will be discussed in Chapter 3 is used to scale the input ground motion. The time series plot of the ground motion is shown in Figure 2.16. For dynamic analysis, the Newmark parameters β and γ are assigned a value of 0.25 and 0.5 respectively. The result of dynamic analysis is the displacement of the top node of the columns 1, 2 and 3, which are shown in Figure 2.24, Figure 2.25, and Figure 2.26 respectively.

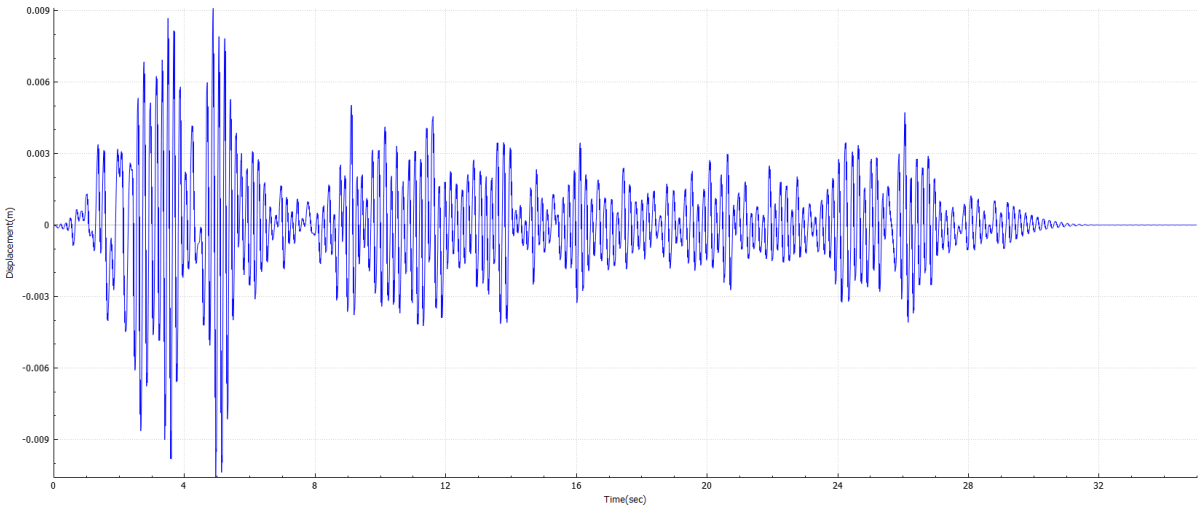

Figure 2.24: Displacement of column 1.

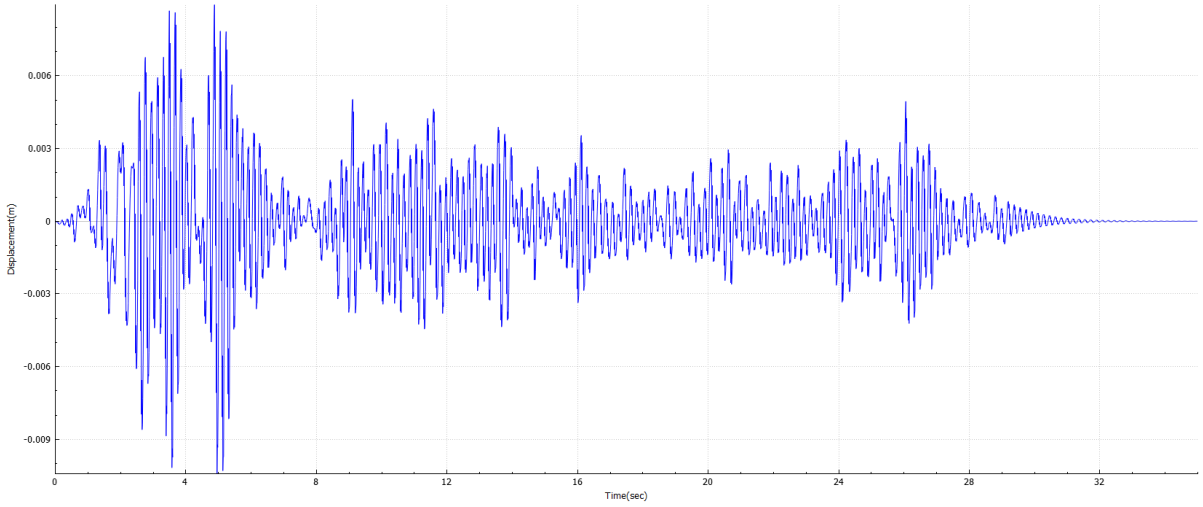

Figure 2.25: Displacement of column 2.

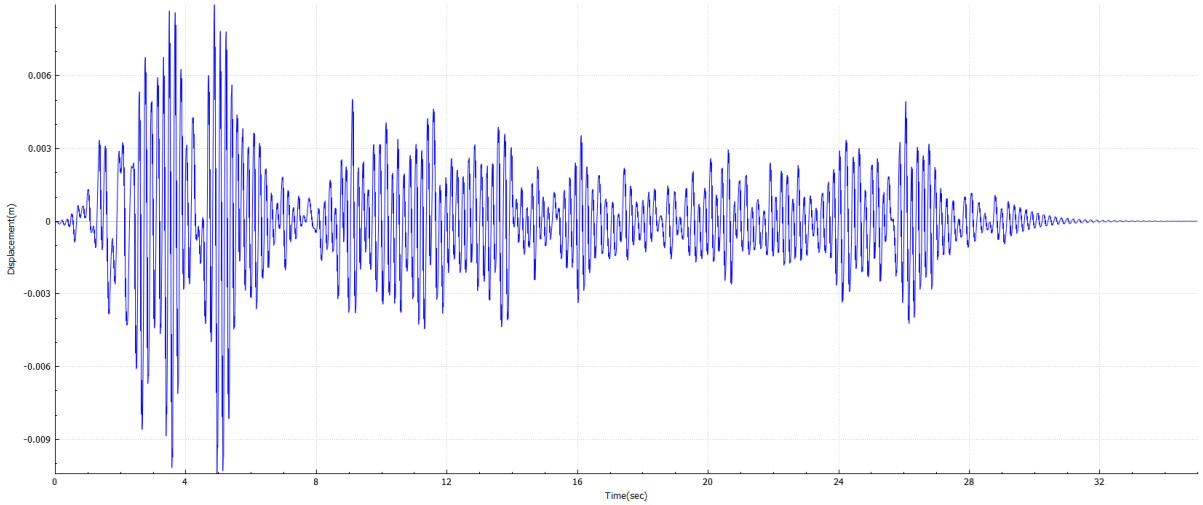

Figure 2.26: Displacement of column 3.

2.7.3 Discussion of Results

Based on the analysis, the following observations are made:

1. It is observed that the displacement of the structure is small and also somewhat similar to the plot of the ground motion. The reason is that the example building has a

relatively short fundamental time period. When a structure has a short time period, implying it is stiff, the structure moves almost rigidly with the ground and the displacements will be relatively small.

2. It is observed that the values of maximum displacement of the three columns are not the same. The maximum displacement for all the columns is observed at 4.97sec. The maximum displacement for the column 1 is 10.59 mm, column 2 is 10.41 mm and column 3 is 10.33 mm. The difference in the maximum displacement is due to the asymmetrical placement of the columns.
3. One of the important observations is with regard to computation time. This is one of the main challenge that is required to be addressed in future. The computation time is more for the case when the *Level of Output* property for dynamic analysis is set by the user as "*Maximum"* in comparison to "*Medium"* or "*Minimum" Level of Output*. For "*Maximum*" level of output case the graphic user interface of the *Rts* is continuously updated at an interval defined by the user using the property *Plot Interval*. For example if the user sets the *Plot Interval* as three then the graphic interface will be updated at every third time step. Thus, if the structure has several components, then in order to reduce computation time it would be better to set the output level as "*Medium*" or "*Minimum*" or set a high value for *Plot Interval*. Therefore it is very important to select the parameters of dynamic analysis in *Rts* judiciously.

Chapter 3: Scaled Records

Detailed analysis of buildings subjected to earthquakes requires ground motions as input to the structural analysis. Presently, the most popular and practical approach is to use scaled versions of ground motions recorded in past earthquakes (ASCE/SEI 7-10; ATC-63 2009). This chapter reviews and implements several techniques to conduct such scaling. Several characteristics of ground motions are used as basis for the scaling, including peak ground acceleration, velocity and displacement, frequency content, duration of ground motion and the energy content, largely affects the design of structure (Bozorgnia and Bertero 2004). These characteristics are affected by various factors, such as type of faulting, soil through which the waves propagate, and amount of energy produced during the earthquake. Thus, the ground motions recorded at different locations would be different. Due to this variation, appropriate ground motion selection criteria for structural analysis must be adopted. In this chapter a summary of how ground motion records are selected and scaled is provided, together with the implementation of a few approaches in *Rts*.

3.1 Ground Motion Databases

Ground motion records can be selected from various databases. There are federal and state agencies that maintains and regulates the ground motion network. For example, a Canada-wide ground motion network is maintained and regulated by the Geological Survey of Canada. The seismographs, which are the instruments for recording ground motions are distributed strategically in different regions, which forms a network. The ground motion data obtained from these networks are analysed to get useful information regarding earthquakes. This useful information is available through various databases. The Pacific Earthquake

Engineering Research (PEER) Center (http://peer.berkeley.edu/products/strong_ground_motion_db.html), U.S. Geological Survey (USGS) (http://earthquake.usgs.gov/data/) and National Earthquake Database (http://www.earthquakescanada.nrcan.gc.ca/stndon/NEDB-BNDS/index-eng.php) are a few of the ground motion sources.

3.2 Selection of Ground Motions

One important guideline for the number of ground motion records that should be selected for detailed dynamic analysis is laid out in ASCE/SEI 7 (ASCE 2010) . That document prescribes the use of a minimum of three ground motions for detailed dynamic analyses of structures. When seven or more than seven ground motion records are used for analyses then the design parameter depends on the average response values obtained from the analyses. In case less than seven ground motion records are used for analyses, then the maximum of response value is used. See for example Reyes and Kalkan (2011).

Selection of ground motion is an important step in dynamic analyses and must be performed with care. Usually records are selected based on earthquake magnitude, *M*, and distance, *R*, anticipated for the site (Katsanos et al. 2010). The representative earthquake for the site is determined using the deaggregation of the hazard curve to obtain physical values (*M*, *R* pair) that can be used for engineering decision making. Once *M*, *R* pair is obtained there are ground motion databases from which the ground motion records can be selected. Another additional selection criterion can be the soil profile. Usually the shear wave velocity at the uppermost 30m is used as the criterion of classification of soil condition. There are other criteria like the type of faulting, fracture mechanism and acceleration to velocity ratio,

which could also be considered for selection of ground motions. Considering several criteria would improve the selection, but due to scarcity of ground motion records that would fit all criteria, it is often necessary to relax the selection criteria.

Records are also selected based on spectral matching. The response spectrum of the ground motion is matched with the target spectrum defined by the structural design code, which is discussed in Section 3.4. The target spectrum depicts the probability of occurrence of an intensity measure. An intensity measure links the hazard analysis performed by the seismologists and the structural analysis conducted by the engineers. In the next section various intensity measures are discussed.

3.3 Ground Motion Intensity Measures

There are different types of intensity measures that can be used for seismic risk analysis, such as peak ground acceleration (PGA) and spectral acceleration, (*Sa*) at a particular time period. Most commonly used intensity measure is the spectral acceleration of a SDOF system with 5% damping. There are three types of spectral acceleration namely i) spectral acceleration of an arbitrary component (Sa_{arb}), ii) spectral acceleration of geometric component ($Sa_{g.m.}$) and iii) spectral acceleration of maximum direction (Sa_{maxDir}) (Whittaker et al. 2012). Usually seismologists use the spectral acceleration of geometric component as the intensity measure, which is the geometric mean of the spectral acceleration of two horizontal components of the ground motion. Structural engineers usually use the spectral acceleration of an arbitrary component as the intensity measure. Any of the spectral acceleration can be used, but it is required to be consistent with the definition of the spectral acceleration for both hazard and structural analysis (Baker and Cornell 2006).

One more intensity measure that is important to note is the vector valued intensity measure (Baker and Cornell 2005). In contrast to the spectral acceleration at a particular time period another additional measure called ε (epsilon) is used for selection of ground motion. The intensity measure ε is defined as the number of standard deviations by which the natural logarithm of spectral acceleration at a particular time period differs from median predicted natural logarithm of spectral acceleration at the same time period (computed using attenuation models) for a given magnitude and distance for a site. According to Baker and Cornell (2005), the vector valued intensity measure is superior to the intensity measure consisting of spectral acceleration at a particular time period alone.

3.4 Target Spectra

As mentioned in Section 3.2, dynamic analyses of structures are performed by subjecting the structure to ground motions which closely match the target response spectrum. The target spectrum can be hazard targeted or risk targeted. In the following section the hazard spectra and risk-targeted spectra are discussed.

3.4.1 Hazard Spectra

Uniform hazard target spectrum, conditional mean target spectrum and conditional target spectra are the three types of hazard spectra discussed in this section.

3.4.1.1 Uniform Hazard Target Spectrum

The uniform hazard spectrum (UHS) is the most commonly used target response spectrum. It depicts the spectral acceleration versus time period of a SDOF system with a certain level of

damping for a low hazard level, for example 2% exceedance in 50 years. The UHS for a particular site is constructed by enveloping the results obtained from PSHA for each time period. See for example Humar and Rahgozar (2000). For a single ground motion, achieving such high spectral acceleration for all time periods is rare. Thus, the hazard spectrum is conservative, especially when the structural response is dominated by the fundamental time period.

3.4.1.2 Conditional Mean Target Spectrum

Conditional Mean Spectrum (CMS) is an alternative to UHS for ground motion selection and matching. For CMS, first the spectral acceleration at a particular time period is determined using seismic hazard analysis. Then the spectral acceleration at other time periods are conditioned on this spectral acceleration value. Since CMS is conditioned on one particular time period, only the *Sa* at that particular time period is to be determined using PSHA. The deaggregation of the hazard level gives the mean of magnitude, distance and the parameter εs (epsilons). The parameter ε defined previously is expressed as:

$$\varepsilon(T) = \frac{\ln Sa(T) - \mu_{\ln Sa}(M, R, T)}{\sigma_{\ln Sa}(T)} \tag{72}$$

where $\mu_{\ln Sa(M,R,T)}$ and $\sigma_{\ln Sa(T)}$ are the mean and the standard deviation, determined using ground motion models.

Equation (72) is similar to the equation for standard normal distribution, which is given as:

$$z = \frac{x - \mu}{\sigma} \tag{73}$$

where z is the standard normal distribution, x is the normal distribution, μ is the mean and σ is the standard deviation.

As the probability distribution for *Sa* is lognormal therefore ln*Sa* has a normal distribution. Equation (73) results in ε having a standard normal distribution with zero mean and unit standard deviation. To study the variation of ε and how they relate to each other at various time periods, a large number of ground motion records were analyzed by Baker (2011). It was observed that there exists a strong correlation in ε for the time periods close to the time period of interest and weaker correlation for time period far away from the time period of interest. The associated ε at other time periods for ε at a given period is:

$$\mu_{\varepsilon(T_i)/\varepsilon(T^*)} = \rho(T_i, T^*)\varepsilon(T^*) \tag{74}$$

$$\rho(T_{\min}, T_{\max}) = 1 - \cos\left\{\frac{\pi}{2} - \left[0.359 + 0.163 I_{(T_{\min}<0.189)} \ln\frac{T_{\min}}{0.189}\right] \ln\frac{T_{\max}}{T_{\min}}\right\} \tag{75}$$

where $\rho(T_i,T^*)$ is the correlation coefficient and $\varepsilon(T^*)$ is the value of ε at the time period of interest, $\mu_{\varepsilon(Ti)/\varepsilon(T^*)}$ is the mean value of $\varepsilon(T_i)$, given $\varepsilon(T^*)$, T_{min} and T_{max} is the minimum and maximum of the two time period under consideration, and $I_{(Tmin<0.189)}$ is an indicator function equal to 1 if T_{min}<0.189s and equal to 0 otherwise.

The next step in computation of CMS is determination of the mean and standard deviation of median log spectral acceleration at all time periods using the ground motion attenuation models. The last step involves the determination of conditional mean value of ln$Sa(T_i)$ using the following expression:

$$\mu_{\ln Sa(T_i)/\ln Sa(T^*)} = \mu_{\ln Sa(M,R,T_i)} + \rho(T_i, T^*)\varepsilon(T^*)\sigma_{\ln Sa}(T_i) \tag{76}$$

As the determination of spectral acceleration at given a period forms the first step in construction of CMS, hence the choice of time period should be judiciously made. If the time period of excitation is not known with certainty, then it is preferable to construct CMS conditioned at different time periods. For each of the CMS constructed separate set of ground

motions are selected. Based on the analysis for each of the set, the time period that is of maximum important is identified. When the time period of the structure is not known, for example if it is required to select ground motions before the structure is designed, CMS cannot be used as the target spectrum. In such situations even though UHS is a conservative spectrum, as it does not depend on the time period of the structure it is preferred over CMS.

3.4.1.3 Conditional Target Spectra

Variability in the target spectrum is not accounted by CMS. The variability in the response of the structure can be affected by the variability in the spectrum. This is addressed by the conditional target spectra. The standard deviation of $\ln Sa(T_i)$ is given as (Baker 2011):

$$\sigma_{\ln Sa(T_i)/\ln Sa(T^*)} = \sigma_{\ln Sa(T_i)}\sqrt{1-\rho(T_i,T^*)} \tag{77}$$

where $\rho(T_i,T^*)$ is defined in Equation (75).

3.4.2 Risk-Targeted Spectra

Risk-targeted ground motion maps have been studied by Luco et al. (2007) by making adjustments to the seismic design maps to achieve uniform probability of collapse. The ground motion values obtained from the uniform hazard maps for seismic design have a uniform percentage of exceedance of spectral acceleration for a fixed duration in years. For example 2% exceedance in 50 years. But the collapse capacity of structures designed using the uniform hazard ground motions maps may not be uniform due to record to record variability and uncertainty in construction details of the structure like the construction quality and material properties. Hence, the collapse probability may not be uniform even when structures are designed for uniform hazard ground motion intensity. This uncertainty in

collapse capacity is expressed using a random variable with lognormal probability distribution. The distribution is parameterized by 10th percentile of probability distribution and standard deviation, β. Based on ATC-63 project, a low probability of collapse is approximated as 10% for Maximum Considered Earthquake (MCE) ground motions. Thus, it is said that the 10th percentile of collapse capacity, $c_{10\%}$, is equal to the MCE spectral acceleration at fundamental period of vibration of the structure, S_{MT}. Thus, the probability density function for collapse capacity is expressed as:

$$f_{capacity}(c) = \Phi\left[\frac{\ln c - (\ln c_{10\%} + 1.28\beta)}{\beta}\right]\frac{1}{c\beta} \tag{78}$$

$$\Phi[\cdot] = \exp\left\{-[\cdot]^2 / 2\right\} / \sqrt{2\pi} \tag{79}$$

Next, the annual collapse probability, $P[Collapse]$ is determined by combining the probability of collapse distribution with the ground motion hazard curve using the expression:

$$P[Collapse] = \int_0^\infty P[Sa > c]\, f_{capacity}(c)dc \tag{80}$$

where $P[Sa>c]$ is the annual probability of spectral acceleration (demand) exceeding the collapse capacity value.

Following which, the probability of collapse in Y years is computed from the annual collapse probability as follows:

$$P[Collapse\_in\_Y\_years] = 1-(1-P[Collapse])^Y \tag{81}$$

Once the value of probability of collapse in Y years is computed, adjustments are made to the MCE ground motions to achieve target collapse by back calculating iteratively. This would result in a 'risk-targeted' ground motion maps.

3.5 Ground Motion Scaling

The ground motion records are to be scaled to match the demand for a particular site. There are several methods to scale ground motions which is elaborately discussed by Michaud and Léger (2014). Few of the methods are mentioned in this section.

3.5.1 Peak Ground Acceleration (PGA) Scaling

PGA is the maximum amplitude of acceleration for a ground motion acceleration record. In this scaling method the ground motion record is multiplied by a factor, such that the resulting scaled ground motion record has the same PGA value as that of target PGA. The target PGA depends on the site conditions. Scaling of the entire amplitude of the record results in scaling of the elastic response spectrum over the entire period range. Nau and Hall (1984) performed analyses to study the scaling of the response spectra using peak ground motion. They concluded that the dispersion in the response spectra was significant.

3.5.2 $Sa(T_1)$ Scaling

This method involves scaling the ground motion record by a factor such that the resulting spectral acceleration at the fundamental time period is equal to the target spectral acceleration for the same period. The scaling involves:

1. Selection of ground motion based on *M*, *R*, type of faulting and site classification.
2. Determination of the target intensity corresponding to first period of vibration of the structure.
3. Scale the ground motion such that the spectral acceleration of the record at the fundamental period is equal to the target spectral acceleration.

The scatter in the response due to $Sa(T_1)$ scaling method was studied by Weng et al. (2008) for structures having multi-mode effect. They concluded that this type of scaling would result in large scatter of displacement demand for tall buildings.

3.5.3 ASCE/SEI 7 Scaling

For design and analysis of structures according to ASCE (2010) handbook for Minimum Design Loads for Building and other Structures, a minimum of three ground motion records are to be selected. The selected records are scaled such that the average value of the 5% damped response spectra of the ground motions is not less than the design response spectrum between $0.2T$ and $1.5T$, where T is the natural period of vibration of the structure.

3.5.4 ATC-63 Scaling

A two-step scaling method is suggested according to ATC-63 (2009). For collapse analysis two sets of ground motion records are provided. The far field record set consists of 22 component pairs of horizontal ground motions from sites located greater than or equal to 10 km from the fault rupture. The near field record set includes 28 component pairs of horizontal ground motions recorded at sites less than 10 km from the fault rupture. The first step of scaling involves normalization of all the records by their respective peak ground velocity. The second step is to scale the records such that the median spectral acceleration of the record set is equal to the target spectral acceleration at fundament period of vibration of the structure.

3.5.5 Mean Square Error Scaling

In this method, the first step is to compute the mean square error (MSE). MSE is the difference between the spectral acceleration of the record and spectral acceleration of the target spectrum given as (PEER 2013):

$$MSE = \frac{\sum_i w(T_i)\left\{\ln[Sa_{target}(T_i) - \ln[fSa_{response}(T_i)]\right\}^2}{\sum_i w(T_i)} \quad (82)$$

where $w(T_i)$ is a weight function to apply different weights to some period of greater interest, $Sa_{target}(T_i)$ is the spectral acceleration of the target spectrum, $Sa_{response}(T_i)$ is the spectral acceleration of the ground motion being scaled and f is the scaling factor.

The scaling factor is determined such that the value of MSE is minimized.

3.5.6 Frequency Domain and Time Domain Spectral Matching

In frequency domain spectral matching, the spectral ordinates of the record are modified in the frequency domain to match the target spectrum. Whereas, in time domain method of spectral matching, the wavelets are added in the time domain to modify the spectral ordinates of the record to match the target spectrum. The drawback with using frequency domain spectral matching is that there is a possibility of change in the characteristics of the ground motion (Michaud and Léger 2014). There are computer programs like *RspMatch* (Abrahamson 1992) that enables spectral matching.

3.6 Implementation of Scaled Ground Motion Models in *Rts*

Based on the review, it is realized that for design and analysis of structures subjected to the earthquake loading, it is required to select and scale ground motions. To enable scaling of

ground motions in *Rts*, three ground motion scaling models are implemented. The scaling models implemented in *Rts* are called *RScaledGroundMotionModel*, *RPGAScaledGroundMotionModel* and *RSa_T1_ScaledGroundMotionModel*. These three models are a subclass of the *RModel* class, shown in Figure 3.1. The three models uses the *RTimeSeries* class to obtain a ground motion time series and to store the scaled ground motion time series. Each of the ground motion scaling model performs the following functions:

1. The *RScaledGroundMotionModel* class gives an output ground motion, which is the input ground motion multiplied by a factor specified by the user. The properties pane of this model in the user interface of *Rts* is shown in Figure 3.2.
2. The *RPGAScaledGroundMotionModel* class scales the input ground motion such that the PGA of the scaled ground motion is equal to the target PGA defined by the user. The properties pane of PGA scaled ground motion model is shown in Figure 3.3.
3. The *RSa_T1_ScaledGroundMotionModel* class scales the input ground motion according to the procedure explained in Section 3.5.2 for $Sa(T_1)$ scaling. The time period and the target spectral acceleration values are defined by the user. The properties pane of $Sa(T_1)$ scaled ground motion model is shown in Figure 3.4.

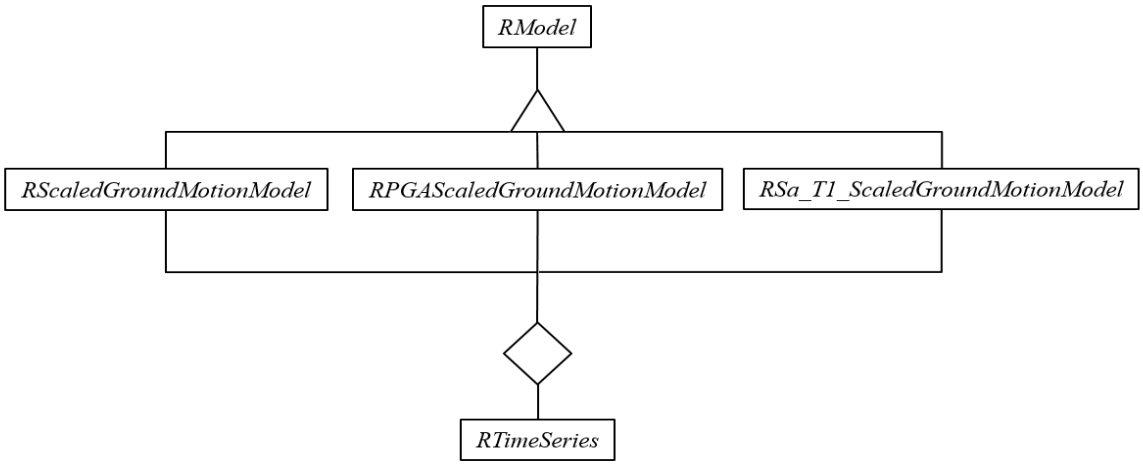

Figure 3.1: *Rts* software architecture for ground motion model.

Properties

	Property	Value
1	Object Name	Scaling
2	Output Level	Maximum
3	Input File	Wave3.txt
4	Scaling Factor	1

Output

Figure 3.2: *Rts* properties pane for scaled ground motion model.

Properties

	Property	Value
1	Object Name	PGAScaling
2	Output Level	Medium
3	Input File	ElCentro.txt
4	Target PGA	0.7

Output

Figure 3.3: *Rts* properties pane for PGA scaled ground motion model.

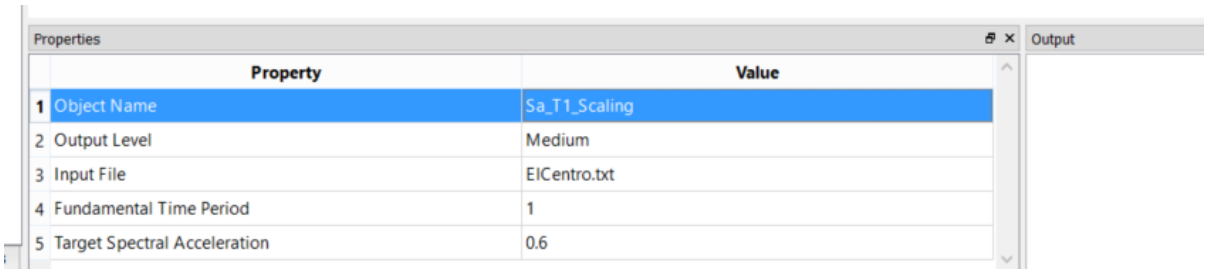

Figure 3.4: *Rts* properties pane for *Sa*(T_1) scaled ground motion model.

3.7 Example: Ground Motion Scaling

In order to demonstrate the ground motion models implemented in *Rts*, El Centro (1940) record, shown in Figure 2.16 is used as the input time series. Following scaling is performed using the implemented ground motion models:

1. The ground motion is scaled using the *RScaledGroundMotionModel* to units of '*g*'. That is, the scaling factor is set to a value of 1. The resultant time series is shown in Figure 3.5. The corresponding response spectrum generated using *SiesmoSignal* (2015) is shown in Figure 3.6.
2. Figure 3.7 shows the resultant ground motion time series when the record is scaled to a target PGA of 0.7*g* using the *RPGAScaledGroundMotionModel*. The unscaled record has a PGA value of 0.32*g*. The corresponding response spectrum of the scaled record is shown in Figure 3.8.
3. *Sa*(T_1) scaling is performed using the *RSa_T1_ScaledGroundMotionModel*, such that the target spectral acceleration at a time period of 1 sec is set to 0.6*g*. The resultant time series in shown in Figure 3.9. The corresponding response spectrum of the scaled record is shown in Figure 3.10.

In Figure 3.6 and Figure 3.8 consider the time period of 1 sec for observation. It is observed

that the spectral intensity value is 4.5 m/sec^2 and 10 m/sec^2 for records scaled using the *RScaledGroundMotionModel* and *RPGAScaledGroundMotionModel* respectively. Similarly in Figure 3.10, at a time period equal to 1 sec, the spectral intensity is 6 m/sec^2 for $Sa(T_1)$ scaled ground motion record. It is concluded that the scaling methods implemented would assist the user to scale ground motions to achieve desired level of spectral intensity.

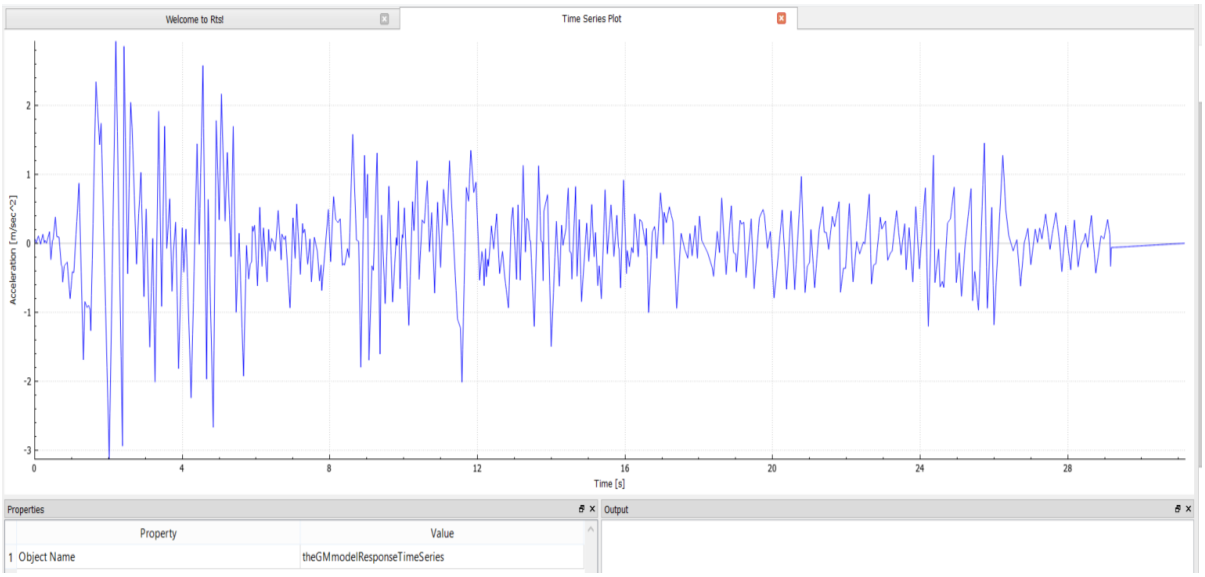

Figure 3.5: Scaled El Centro ground motion record.

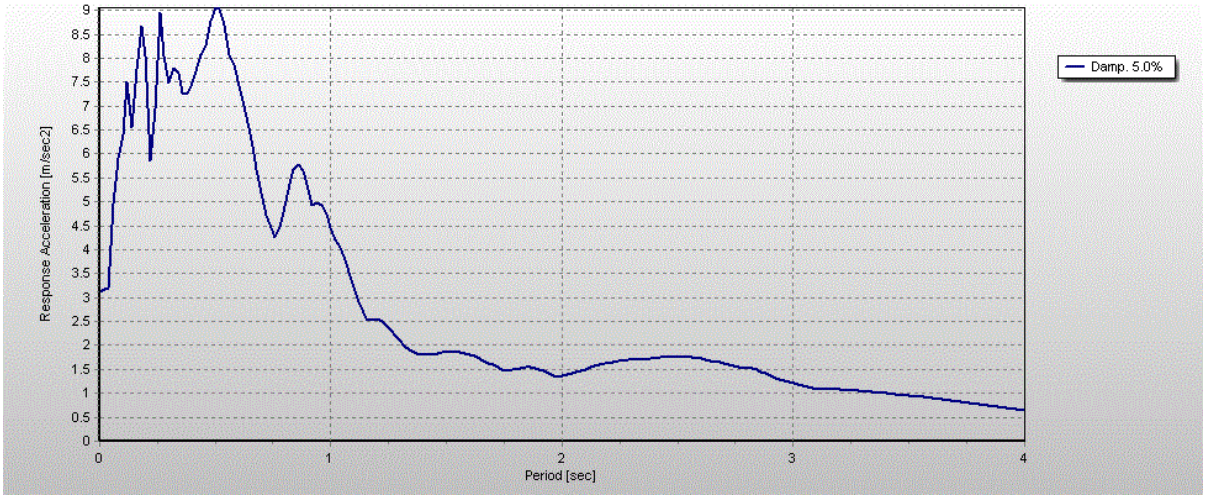

Figure 3.6: Response spectrum of unscaled El Centro ground motion record.

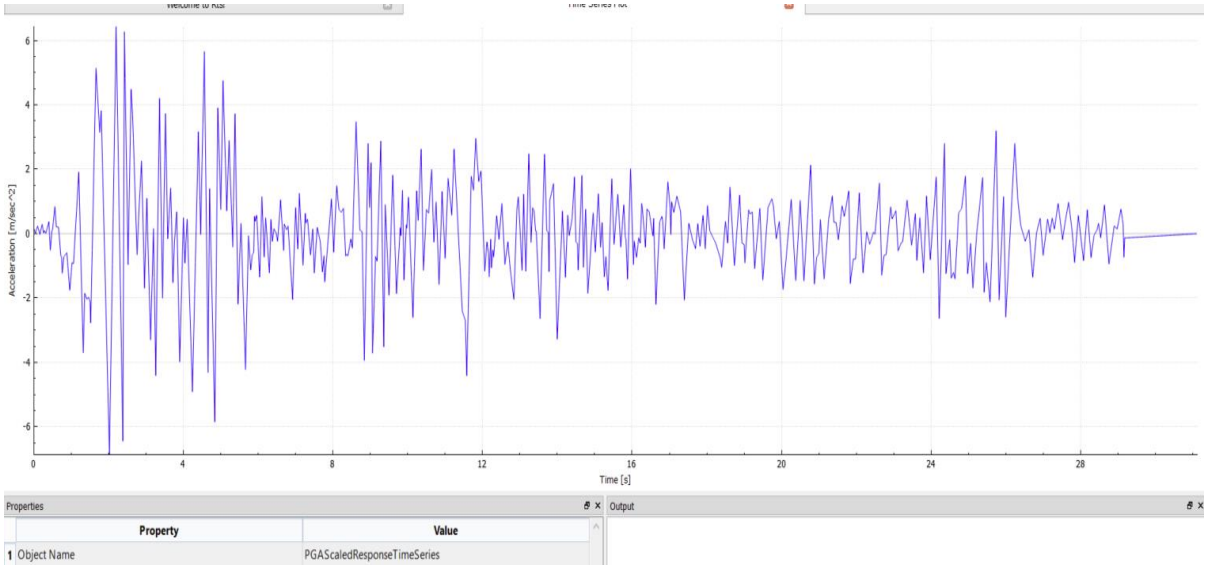

Figure 3.7: PGA scaled El Centro ground motion record.

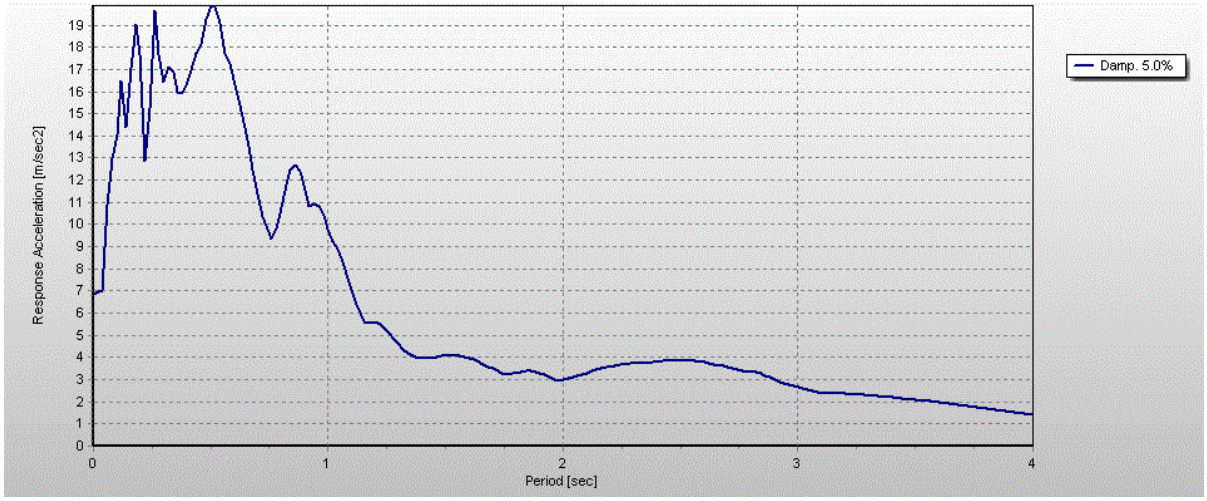

Figure 3.8: Response spectrum of PGA scaled El Centro ground motion record.

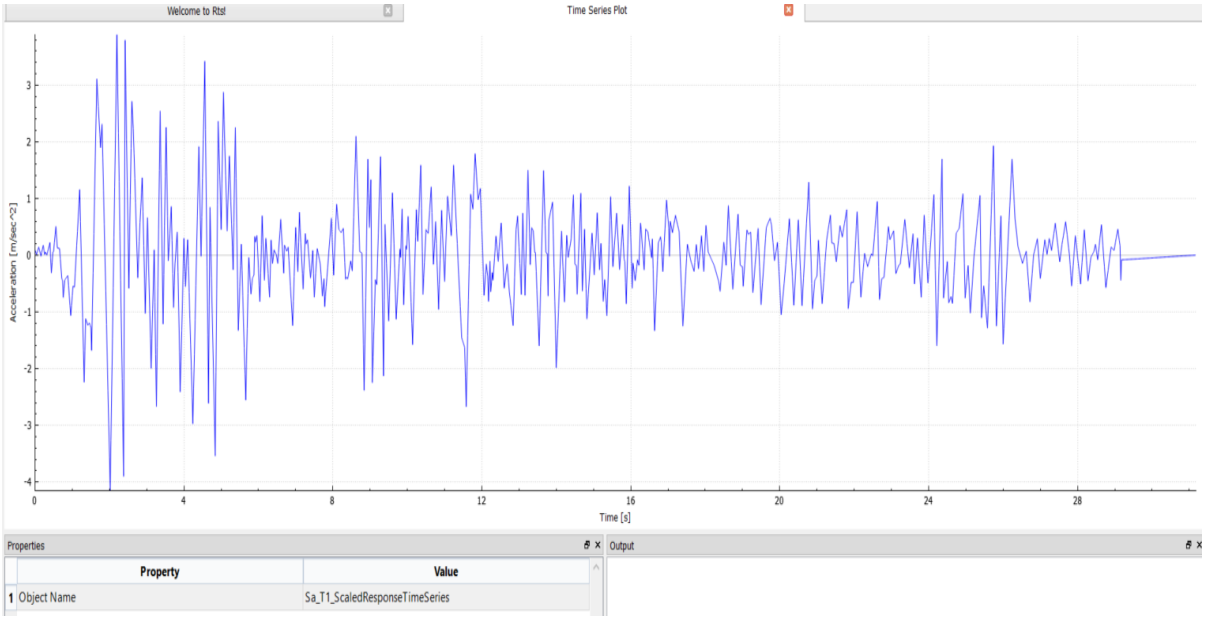

Figure 3.9: $Sa(T_1)$ scaled El Centro ground motion record.

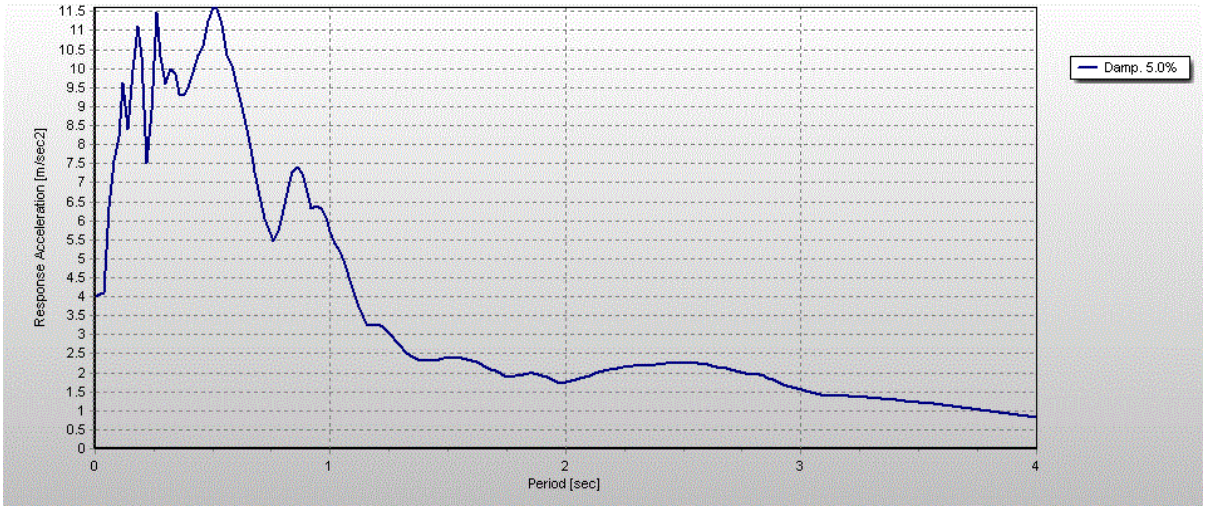

Figure 3.10: Response spectrum of $Sa(T_1)$ scaled El Centro ground motion record.

Chapter 4: Synthetic Ground Motions

In Chapter 3 the use of scaled ground motions for design and analysis was discussed. It was inferred that it is difficult to obtain records that would fit all the selection criteria. Furthermore, Grigoriu (2011) pointed out that the scaled ground motion records are not suitable for making fragility estimates, which is the probability of exceeding certain levels of damage or loss as a function of ground motion intensity. The reason behind this unsuitability is that there is a fundamental discrepancy between the properties of the recorded ground motion and the scaled versions.

One solution to overcome the problem of scarcity of the ground motion records that would match all the criteria mentioned previously is to use synthetic ground motion models. It is also advantageous to use synthetic ground motions in reliability analysis as they can take uncertainty explicitly into account by the use of random variables. The next section of this chapter contains a review of the use of "filtered white noise" to create ground motions. That approach was adopted in the paper by Koduru and Haukaas (2010) and others. Due to the advantages of using a synthetic ground motion model, it is implemented in *Rts*. In this section few of the synthetic ground motion model are reviewed. Also synthetic ground motion model that is implemented in *Rts* is discussed and a few sample ground motions generated using the model is illustrated.

4.1 Review of Synthetic Ground Motion Models

Several synthetic ground motion models have been developed to date. Shinozuka and Deodatis (1988), reviewed different synthetic ground motion models and stated that these can be based on filtered white noise processes, filtered Poisson processes, spectral representation

of stochastic processes or stochastic wave theory. In this section few of the synthetic ground motion models are reviewed briefly and one model in detail. Also a few computer programs that assist in generation of ground motions are mentioned.

In the model developed by Jennings et al. (1968) ground motions were generated as a random process with a power spectral density multiplied with an envelope function. Iyengar and Iyengar (1969) developed a model for generating a stationary random process modulated by a deterministic function. The rate of zero crossing and the time variation of the ground motion are the two properties that are required to be defined for this model. Der Kiureghian and Crempien (1989) developed a model where the ground motion is defined as a superposition of modulated stationary component, each representing the ground motion in a distinct frequency band. They also described the evaluation of model parameters from a recorded accelerogram so that the generated ground motion would have similar frequency and temporal nonstationarity. Fan and Ahmadi (1990) developed a model by considering nonstationarity in the frequency and amplitude content based on the Kanai-Tajimi model. Ground motions were generated using this model and compared with the actual records of El Centro 1940 and Mexico City 1985.

The synthetic ground motion model developed by Rezaeian and Der Kiureghian (2010) reproduces the characteristics of real earthquake ground motions such as evolving intensity, predominant frequency, duration and bandwidth. The model is based on the modulation of white noise. The time modulation function and frequency function used in this model are related to the effective duration of the motion, arias intensity, time at the middle of strong shaking phase, filter frequency, rate of change of the filter frequency and filter damping ratio. To generate ground motions for a target earthquake, regression models were

developed by considering a subset of NGA ground motion database to relate the parameters of the model to the earthquake and site characteristics. The type of faulting, moment magnitude of the earthquake, closest distance from the recording site to the ruptured area and shear wave velocity of the top 30m of site soil are the four characteristics that were considered. The model parameters are related to these characteristics through empirical equation constructed through regression analysis. The correlations between the parameters are also determined. Hence, for a given earthquake and site condition, ground motions are simulated without the requirement of previously recorded ground motions at the site. The synthetic ground motion model was validated by comparing the statistics of 5% damped elastic response spectra of 500 simulated ground motions with the response spectrum developed using prediction equations. Based on the comparison it was concluded that the model is viable for source to site distance greater than 10km and moment magnitudes greater than 6.5.

The generation of ground motions requires extensive computation. There are computer programs that are available for generation of ground motions. For example SIMQKE (Vanmarcke and Gasparini 1976) is a software developed at the Massachusetts Institute of Technology in 1976 for generating synthetic ground motions. A target ground motion spectral density function is given as input. A fast Fourier transformation is performed to produce ground motions that are non-stationary. The program operates on both conditional and unconditional simulation. In conditional simulation the generated ground motions are conditioned by recorded ground motion. In unconditional simulation the generated ground motions are not conditioned on recorded ground motion. It is a computationally intensive software. There are other computer programs for generating ground motions such as

SIMULSIS developed by Estêvão and Oliveira (2012) and FINSIM developed by Beresnev and Atkinson (1998). For implementation in *Rts*, the synthetic ground motion model developed by Li and Der Kiureghian (1995) is used. The model is discussed in detail in the next section.

4.2 Synthetic Ground Motion Model Implemented in *Rts*

The synthetic ground motion model that is implemented in *Rts* is defined as a filtered white noise process. A representation of near-white noise is shown in Figure 4.1. It is a series of pulses that may be taken to represent the rupture at the source of an earthquake. Each pulse is represented by a random variable with a zero mean and unit standard deviation. The filter, which is explained shortly, represents the medium through which the waves propagate.

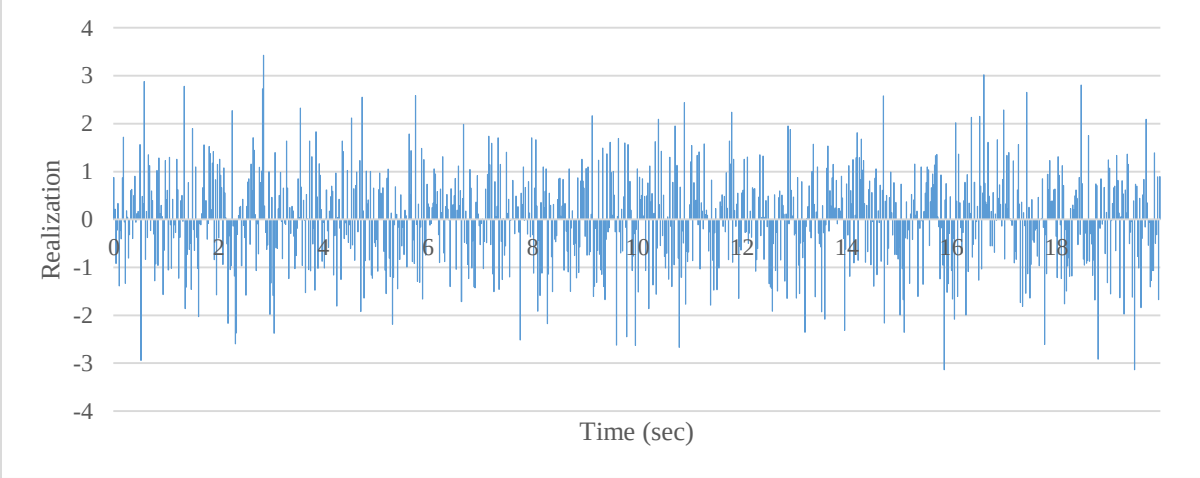

Figure 4.1: Near-white noise.

A ground motion has both temporal and spectral nonstationarity. That means the ground motion varies in both time domain and frequency domain. Usually the amplitude of ground motion increases from a value of zero and then comes back to zero after a duration of time. This temporal nonstationarity can be represented in a synthetic ground motion model by a

time modulating function such as the gamma modulating function, triangular function or piecewise linear function. The ground motion consists of different frequency content. The variation in the frequency content in a synthetic ground motion model is achieved by using filters with different level of damping and frequency. A typical expression of filtered white noise can be represented as (Li and Der Kiureghian 1995):

$$x(t) = \sum_{k=1}^{K} q_k(t) \sum_{i=1}^{N} y_{ik} h_k(t - t_i) \tag{83}$$

where $q_k(t)$ is the time modulating function, k is the number of modulating function which varies from 1 to K, i is the number of random variables which varies from 1 to N, y_{ik} is the realization of random variable and $h_k(t-t_i)$ is the impulse response function.

In the subsequent section each component of the synthetic ground motion model implemented in *Rts* is described.

4.2.1 Gamma Time Modulating Function

In *Rts*, a gamma modulating function is implemented for the synthetic ground motion model to achieve temporal nonstationarity. Due to the similarity in shape with the gamma probability distribution it is called gamma time modulating function. The mathematical form of gamma modulating function is as follows (Haukaas and Der Kiureghian 2004):

$$q(t) = at^b e^{-ct} \tag{84}$$

where the parameter a and c should be greater than zero and the parameter b should be greater than one.

The variation in modulating function on changing each of the parameter is studied. Figure 4.2 shows the case when the parameters b and c are kept constant and the parameter a is varied.

It is inferred from the figure that the parameter *a* controls the intensity of the modulating function.

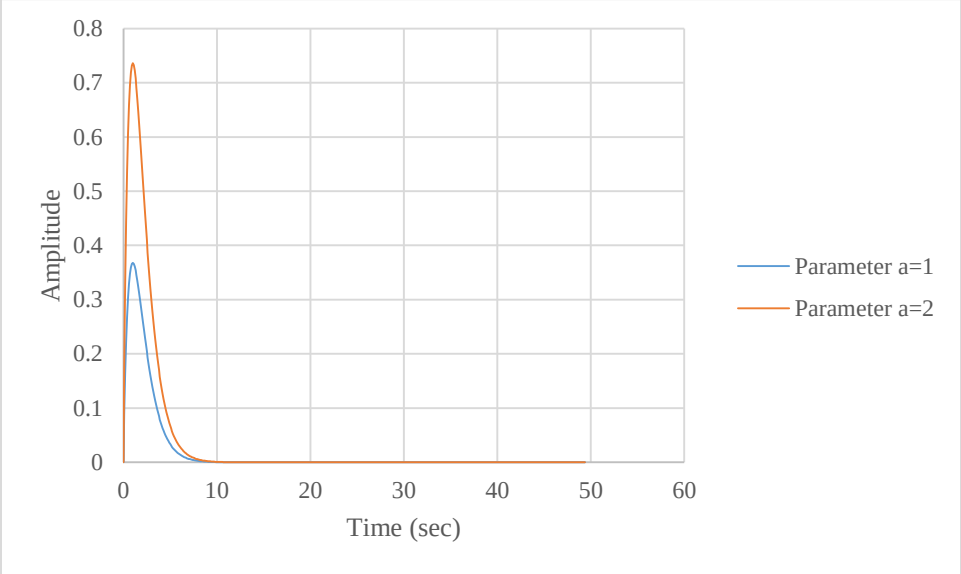

Figure 4.2: Modulating function for two different '*a*' values.

Figure 4.3 depicts the case when the parameters *a* and *c* are kept constant and the parameter *b* is varied. As the value of parameter *b* is increases the curve shifts to right. It is inferred from the figure that the parameter *b* controls the shape of the modulating function.

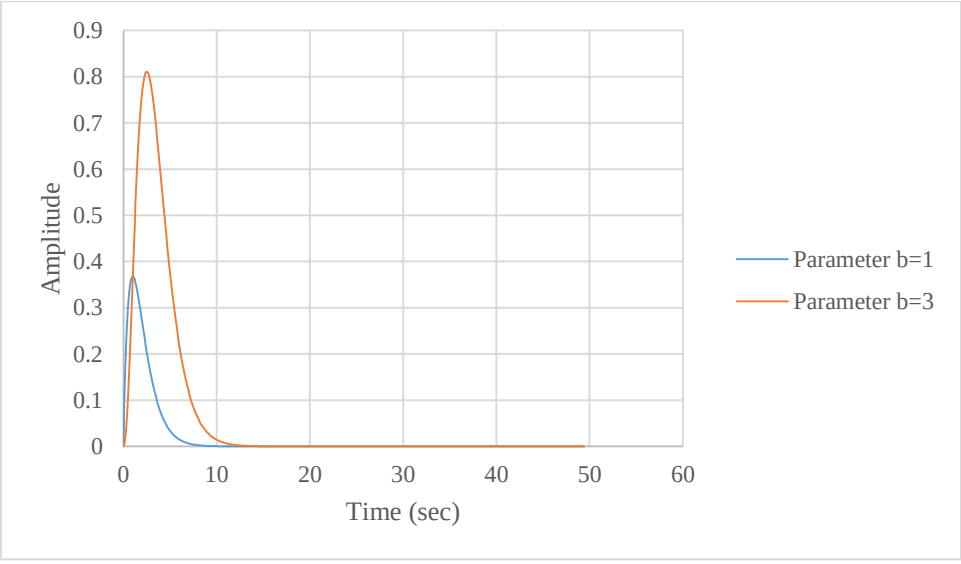

Figure 4.3: Modulating function for two different '*b*' values.

Figure 4.4 shows the case when the parameters *a* and *b* are kept constant and the parameter *c* is varied. As the value of parameter *c* decreases the duration of the modulating function increases. It is inferred that the parameter *c* controls the duration of motion.

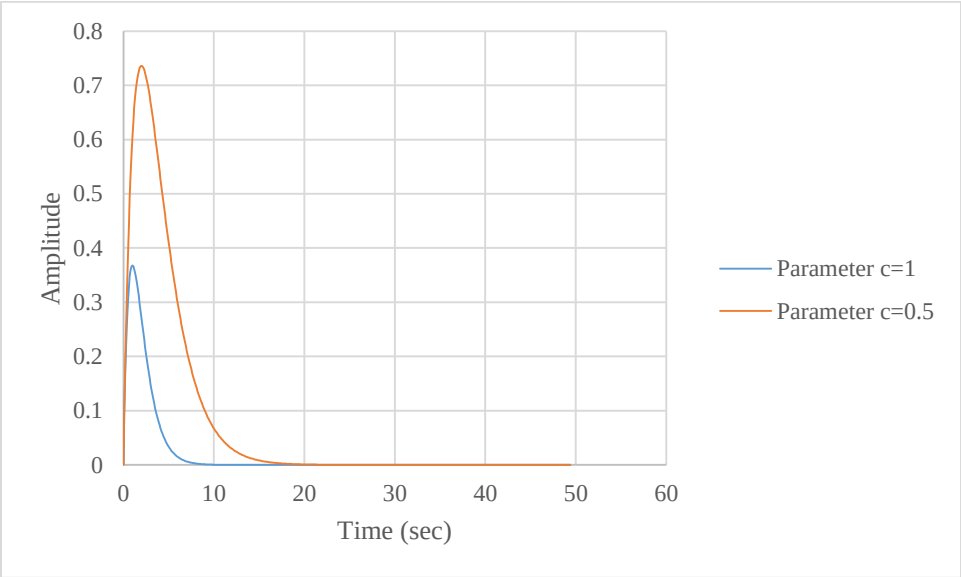

Figure 4.4: Modulating function for two different '*c*' values.

As it is seen from the expression for modulating function there are three unknown parameters. It is possible to determine the value of parameters of the modulating function if the time at maximum amplitude, maximum amplitude and duration of ground motion is known. Let A_c, t_p, and T be the maximum amplitude, time at maximum amplitude and duration of ground motion respectively. The time at which the maximum amplitude is observed is determined by equating the differentiated gamma modulating function with respect to time to zero:

$$\frac{d\left(at^b e^{-ct}\right)}{dt} = 0 \tag{85}$$

$$abt^{b-1}e^{-ct} - act^b e^{-ct} = 0 \tag{86}$$

Solving,

$$\frac{b}{c} = t_p \tag{87}$$

Substituting Equation (87) in Equation (84):

$$a t_p^b e^{-c t_p} = A_c \tag{88}$$

Also at the time equal to T the value of gamma modulating function is a small, say ε:

$$a T^b e^{-cT} = \varepsilon \tag{89}$$

The three unknown parameters are determined by solving the Equations (87), (88) and (89).

4.2.2 Standard Oscillator Filter

In *Rts*, a standard oscillator filter is implemented for the synthetic ground motion model to achieve spectral nonstationarity. The properties of the filter are damping and frequency. The impulse response function is expressed as:

$$h(t) = \left\{ \begin{array}{c} \sin\left(w_d(t)\right) e^{-z(t)}, \quad \text{when } (t) \geq 0 \\ 0, \quad \text{when } (t) < 0 \end{array} \right\} \tag{90}$$

where

$$\omega_d = \omega\sqrt{1-\zeta^2} \tag{91}$$

where ω is the filter frequency and ζ is the filter damping.

In Equation (90), the magnitude of the impulse response function at each time instant depends on the frequency and damping of the filter. Figure 4.5 shows the plot of impulse response function of two filters with different frequencies but same damping. It is inferred from the plot that, as the frequency of the filter increases the number of cycles of the harmonic motion over a duration of time increases. For example, within 25 sec the wave with

a frequency of 150 Hz completes three cycles, while the wave with a frequency of 50 Hz completes one cycle. Thus, it can be inferred that the filter frequency controls the predominant frequency of the generated ground motion. Figure 4.6 depicts the case when the frequency of the filter are kept constant while the damping is varied. It is observed that the intensity of the wave with 5% damping reduces drastically compared to the wave with 2% damping as the time progresses. Thus, it is concluded that the filter property damping controls the bandwidth of the generated ground motion.

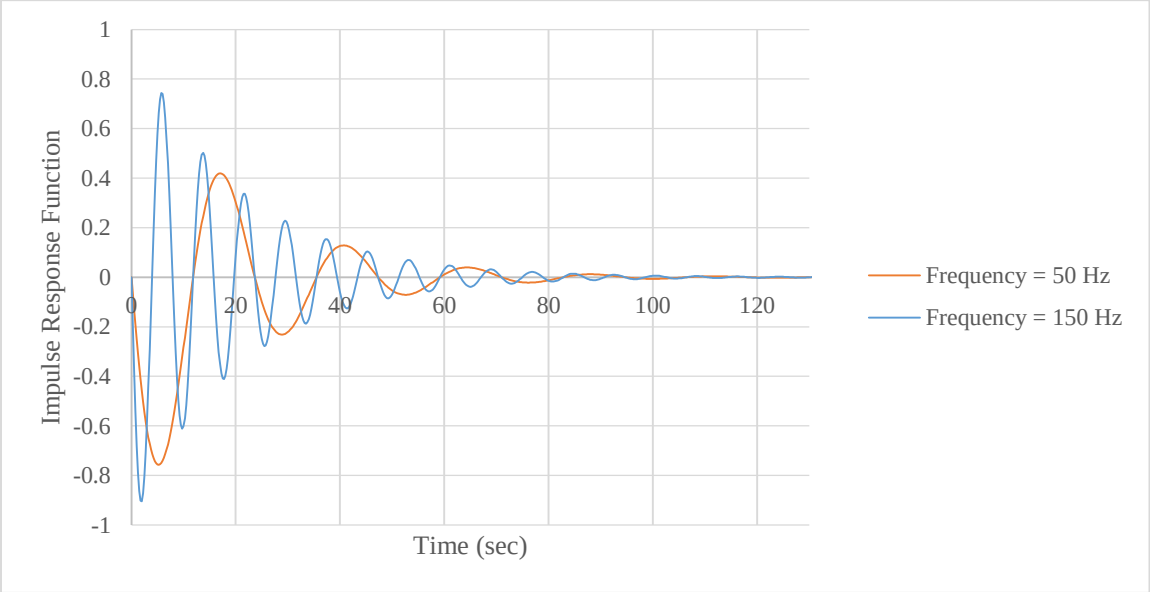

Figure 4.5: Impulse response function for two different frequencies.

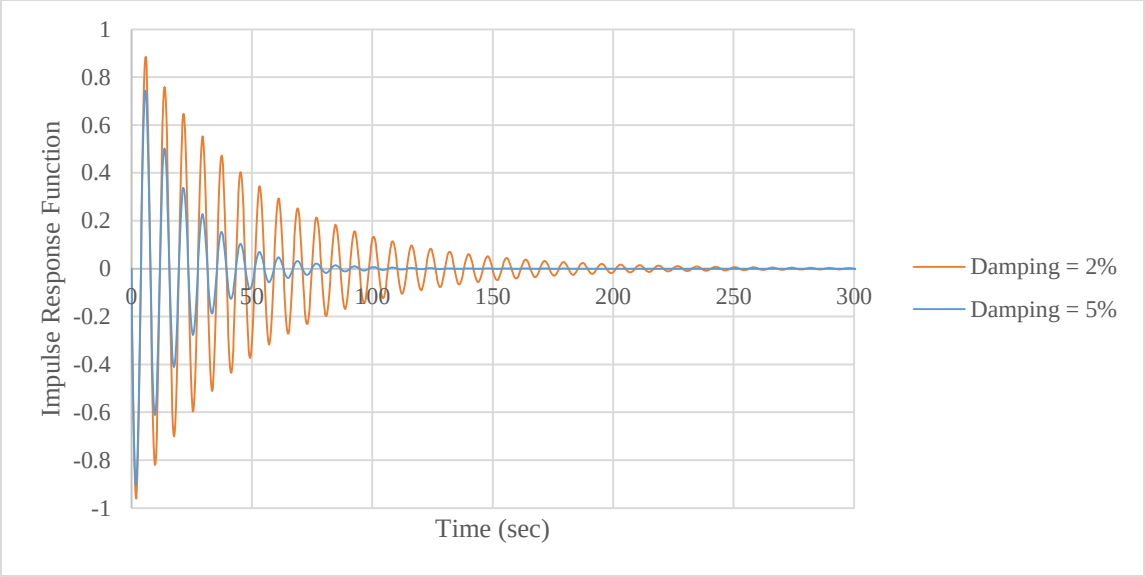

Figure 4.6: Impulse response function for two different damping.

4.3 Software Architecture of Synthetic Ground Motion Model in *Rts*

In this section implementation of the synthetic ground motion model in *Rts* is discussed. Figure 4.7 shows the relevant class map for the synthetic ground motion model in *Rts*. The synthetic ground motion model proposed by Li and Der Kiureghian (1995) is implemented in a class called *RSyntheticGroundMotionModel*. It is a subclass of the *RModel* class. The *RSyntheticGroundMotionModel* uses other classes such as the *RTimeSeriesResponse*, *RContinuousRandomVariable*, *RParameter*, and *RRandomNumberGenerator*. The *RTimeSeriesResponse* class is used to store the generated ground motion. The *RContinuousRandomVariable* class is used to create a continuous random variable. The *RParameter* class performs the function of creating a parameter. The *RRandomNumberGenerator* class is used to generate random numbers. The implemented code is listed in Appendix C of this thesis.

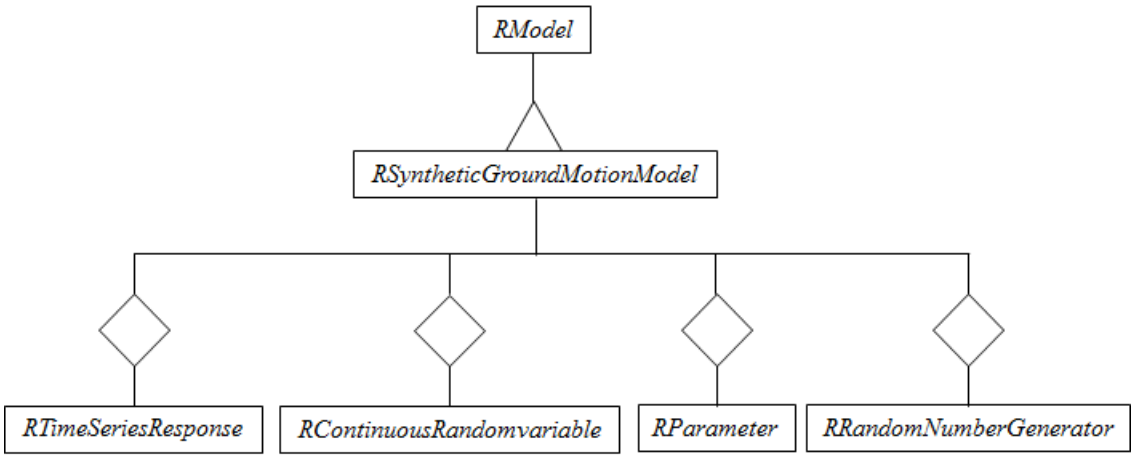

Figure 4.7: *Rts* class map for synthetic ground motion model.

The step by step algorithm of the synthetic ground motion model implemented in *Rts* is as follows:

1. The parameters of the gamma modulating function and filter are obtained as user input.
2. The start time, end time, and number of random variables are also user defined. The properties pane of the user interface in *Rts* for the synthetic ground motion model is shown in Figure 4.8.
3. A *for* loop is initiated starting from time equal to start time to time equal to end time. The *for* loop is incremented by a time step, Δt, which is one fifth the time period of the filter with the highest frequency value.

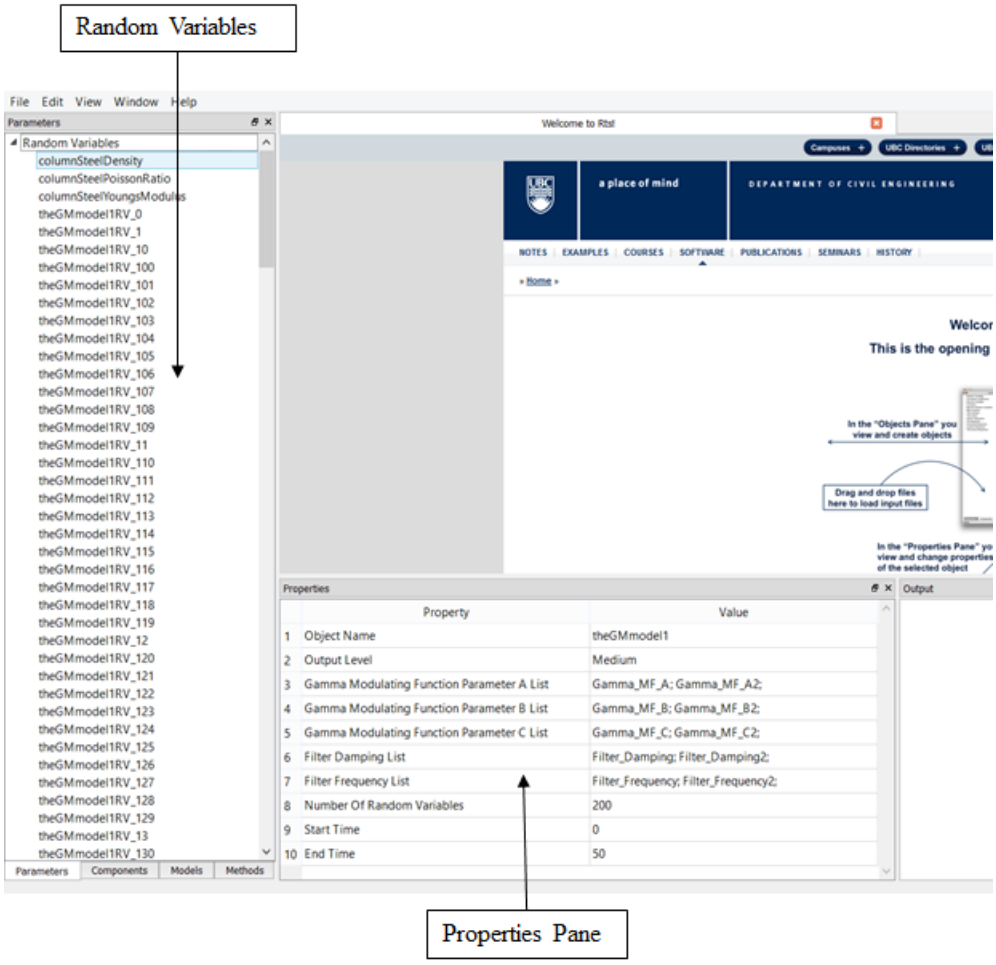

Figure 4.8: Screenshot of the user interface of *Rts*.

4. Another *for* loop is initiated to loop over all the modulating function. Within this loop the amplitude of the modulating function is determined according to Equation (84).
5. Within the loop for modulating function another *for* loop is initiated starting from zero to the number of random variables. At the beginning of the loop a check is performed to make sure that $(t\text{-}t_i)>0$, where t is the current time of the loop initiated in the Step 3 and t_i is product of the time step and the current random variable count. If this condition is not satisfied then the loop is broken. Within this loop the amplitude of impulse response function is computed using the filter properties for the corresponding modulating function. A cumulative sum of the magnitude of impulse

response function multiplied by the realization of the random variable is computed at every increment of the loop.

6. Once the *for* loop initiated in step 5 is complete, the magnitude of the ground motion is computed as the cumulative sum of the product of the amplitude of modulating function and the sum determined in step 5.
7. Once the *for* loop initiated in step 4 is complete, the amplitude of the ground motion is given to the time series at time *t*.
8. The steps 4-7 are repeated for each time step.

4.4 Sample Ground Motions

In this section few of the sample ground motions generated using the model implemented in *Rts* are demonstrated. The ground motions shown in this section may not be physical, but are generated to demonstrate the model. Consider the two modulating functions as shown in Figure 4.9, the properties of which are listed in Table 4.1. The first modulating function has a frequency of 60 Hz and the second one has a frequency of 20 Hz. A sample ground motion generated is shown in Figure 4.10. It is observed that the first segment of the ground motion is jagged to a greater extent than the second segment due to change in the frequency of the filter from high to low. Now, consider the modulating functions shown in Figure 4.11 with properties listed in Table 4.2. The first modulating function has a frequency of 20 Hz and the second one has a frequency of 60 Hz. The sample ground motion generated for this case is shown in Figure 4.12. It is observed that the second segment is jagged to a greater extent than the first segment of the ground motion. From the sample ground motion plots it is inferred that the two ground motions (Figure 4.10 and Figure 4.12) will have different effects on the

structure. The frequency content is an important aspect of the ground motion. Especially for an inelastic system, the lower frequency wave coming in the end can have a drastic impact when the structure has already deformed and the stiffness has reduced, thereby having a greater time period.

Table 4.1 Example 1: Model parameters

	Modulating Function 1	Modulating Function 2
Parameter *a*	0.05	$7x10^{-9}$
Parameter *b*	5	10
Parameter *c*	1.5	0.7
Filter Frequency (Hz)	60.0	20.0
Filter Damping	0.05	0.05

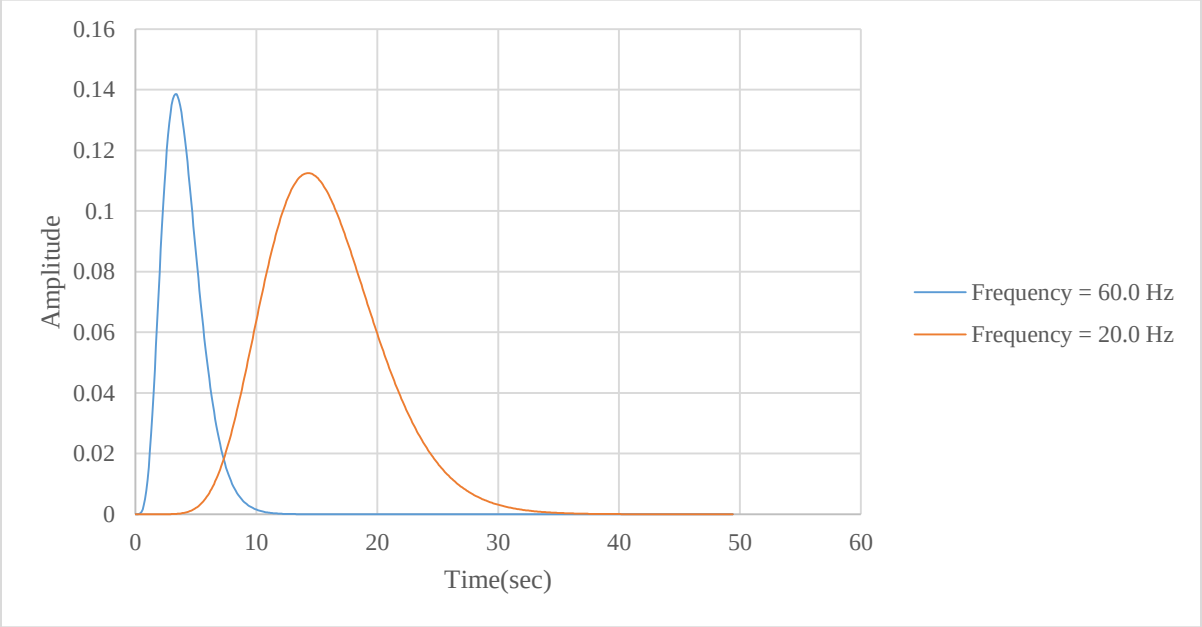

Figure 4.9: Example 1: Modulating functions

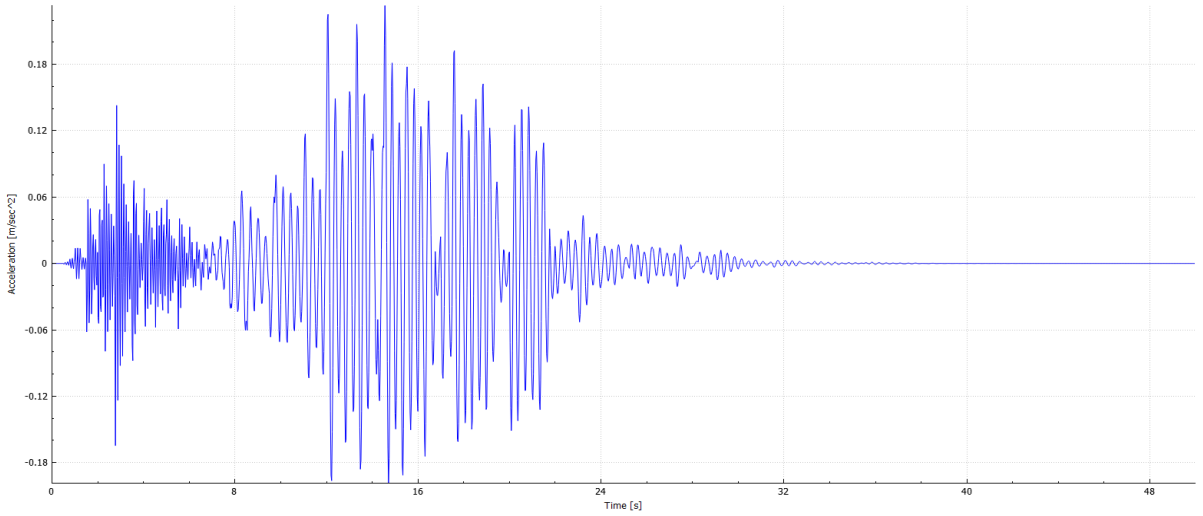

Figure 4.10: Example 1: Sample ground motion

Table 4.2 Example 2: Model parameters

	Modulating Function1	Modulating Function 2
Parameter *a*	0.05	$7x10^{-9}$
Parameter *b*	5	10
Parameter *c*	1.5	0.7
Filter Frequency (Hz)	20.0	60.0
Filter Damping	0.05	0.05

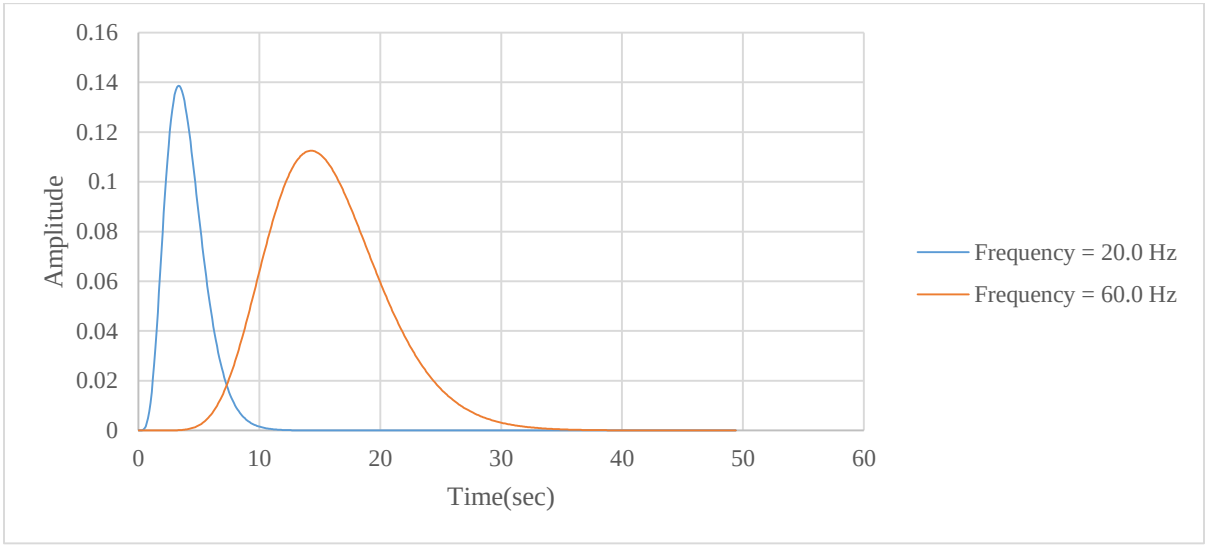

Figure 4.11: Example 2: Modulating functions

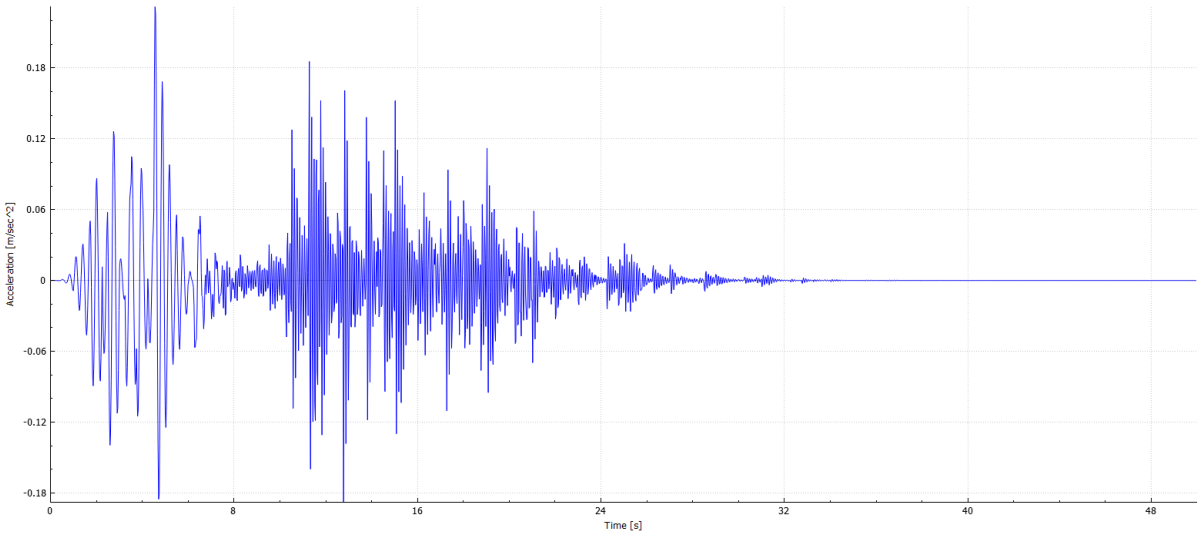

Figure 4.12: Example 2: Sample ground motion

Chapter 5: Conclusions

5.1 Summary of Research

This section summarizes the contribution of this research and also reflects on the purpose of this research from a broader point of view.

Firstly, a tool for dynamic structural analysis of linear and inelastic structures is added to the existing tools in *Rts*. The dynamic analysis uses the time stepping Newmark method for determination of response at every time step. For solving the equation of motion at every time step modified Newton-Raphson method is adopted for inelastic dynamic analysis. The user performing dynamic analysis can decide the parameters required for the Newmark time stepping method and also the parameters for the Rayleigh damping matrix. The response of the structure is recorded as a time series. These implementations provide a platform for forthcoming developments in *Rts* related to detailed structural analysis for performance based earthquake engineering.

Based on this research it is reflected that the hazard models are one of the important aspects in design and analysis of structures. There are different methods by which the ground motions can be selected and scaled. The choice of method depends on various factors of which the building model is an important one. Besides scaling of ground motion records, a synthetic ground motion model is identified and implemented in *Rts*. This allows the generation of artificial ground motion records with a variety of characteristics. Combining all these models and tools, *Rts* provides a holistic platform for reliability analysis.

5.2 Future Work

Rts is now moving towards a goal of providing the engineers with a computer program to simulate built environment of structures. In order to be successful in achieving this goal it is required to overcome a few of the limitations. A few tasks that can be addressed in future research are mentioned below:

1. Presently, dynamic analysis is implemented with Newmark method as the only time stepping method. Other time stepping methods like the central difference method can be implemented. It would give the user an option to choose other time stepping methods for performing dynamic analysis.
2. The synthetic ground motion model requires parameters for the modulating function and filter as input from the user. The choice of parameters depends upon the site condition and the intensity anticipated at the site. It would be advantageous to implement models in *Rts* so that the parameter values are computed in *Rts* and the user input would be site conditions and the energy released during the earthquake.
3. There are different scaling and ground motion modification methods. The choice of ground motion scaling method has not been clearly defined in literature. Future research can be to compare different scaling methods or study the extent to which the ground motions can be scaled so that the resulting ground motion would still be physical. Also other scaling methods can be implemented in *Rts*.
4. For a few of the scaling methods and also for Rayleigh damping it is required to compute the fundamental time period of the structure. For this purpose it would be beneficial to have a tool to perform Eigen value analysis and determine the first few modes of vibration and time period of the structure.

5. Presently, for the synthetic ground motion model, the only modulating function that can be used is the gamma modulating function. Other modulating functions, such as the triangular or piece-wise linear functions can be implemented, which would give the user the advantage of choosing from different modulating functions.
6. One of the important aspects of design and analysis of a structure is cost. In terms of this research, computational cost and time involved in analysis and design is one aspect that needs to be looked into. It is required to reduce the computation time without foregoing the accuracy of results.

References

Abrahamson, N. A. (1992). "Non-stationary spectral matching." *Seismological research letters*, 63(1), 30.

American Society of Civil Engineers (ASCE). (2010). *Minimum design loads of building and other structures (ASCE/SEI 7)*. Reston, VA.

Baker, J. W. (2011). "Conditional mean spectrum: tool for ground-motion selection." *Journal of Structural Engineering*, 137(3), 322–331.

Baker, J. W., and Cornell, C. A. (2005). "A vector-valued ground motion intensity measure consisting of spectral acceleration and epsilon." *Earthquake Engineering and Structural Dynamics*, 34(10), 1193–1217.

Baker, J. W., and Cornell, C. A. (2006). "Which spectral acceleration are you using?" *Earthquake Spectra*, 22(2), 293–312.

Beresnev, I. A., and Atkinson, G. M. (1998). "FINSIM-a FORTRAN program for simulating stochastic acceleration time histories from finite faults." *Seismological Research Letters*, 69(1), 27–32.

Bozorgnia, Y., and Bertero, V. V. (2004). "Earthquake engineering: from engineering seismology to performance-based engineering, Chapter 5: Engineering characterization of ground motion." *CRC Press*.

Chopra, A. K. (2012). *Dynamics of structures: theory and applications to earthquake engineering*. Prentice Hall, Englewood Cliffs, New Jersey.

Cook, R. D. (1995). *Finite element modeling for stress analysis*. Wiley.

Deitel, P. J., and Deitel, H. M. (2013). *C++ how to program*. Prentice Hall.

Estêvão, J. M. C., and Oliveira, C. S. (2012). "Ground motion simulation for dynamic structural analysis : pros and cons." *15th World conference on earthquake engineering*, Lisboa.

Fan, F.-G., and Ahmadi, G. (1990). "Nonstationary Kanai-Tajimi models for El Centro 1940 and Mexico City 1985 earthquakes." *Probabilistic Engineering Mechanics*, 5(4), 171–181.

Federal Emergency Management Agency (FEMA). (2009). *Quantification of building seismic performance factors (ATC-63). FEMA-P695, Washington, D.C.*

Grigoriu, M. (2011). "To scale or not to scale seismic ground-acceleration records." *Journal*

of Engineering Mechanics, 137(4), 284–293.

Gunay, M., and Mosalam, K. (2012). "PEER performance based earthquake engineering methodology , revisited." *Journal of Earthquake Engineering*, 17(6), 829–858.

Haukaas, T. (2014). "SDOF oscillator." <www.inrisk.ubc.ca> (March 30, 2014).

Haukaas, T., and Der Kiureghian, A. (2004). "Finite element reliability and sensitivity methods for performance-based earthquake engineering." *Pacific Earthquake Engineering Research Center*, University of California, Berkeley,.

Humar, J. L., and Rahgozar, M. A. (2000). "Application of uniform hazard spectra in seismic design of multistorey buildings." *Canadian Journal of Civil Engineering*, 27(3), 563–580.

Iyengar, R. N., and Iyengar, K. T. S. R. (1969). "A nonstationary random process model for earthquake accelerograms." *Bulletin of the Seismological Society of America*, 59(3), 1163–1188.

Javaherian Yazdi, A. (2015). "Damage modelling for performance-based earthquake engineering." University of British Columbia.

Jennings, P. C., Housner, G. W., and Tsai, N. C. (1968). "Simulated earthquake motions."

Katsanos, E. I., Sextos, A. G., and Manolis, G. D. (2010). "Selection of earthquake ground motion records: A state-of-the-art review from a structural engineering perspective." *Soil Dynamics and Earthquake Engineering*, 30(4), 157–169.

Der Kiureghian, A., and Crempien, J. (1989). "An evolutionary model for earthquake ground motion." *Structural Safety*, 6(2), 235–246.

Koduru, S. D., and Haukaas, T. (2010). "Probabilistic seismic loss assessment of a vancouver high-rise building." *Journal of Structural Engineering*, 136(3), 235–245.

Kramer, S. L. (1996). *Geotechnical earthquake engineering*. Pearson Education, Upper Saddle River, NJ.

Li, C.-C., and Der Kiureghian, A. (1995). "Mean out crossing rate of nonlinear response to stochastic input." *Proceedings of ICASP-7, Balkema, Rotterdam*, 295–302.

Luco, N., Ellingwood, B. R., Hamburger, R. O., Hooper, J. D., Kimball, J. K., and Kircher, C. A. (2007). "Risk-targeted versus current seismic design maps for the conterminous United States."

McKenna, F., Scott, M. H., and Fenves, G. L. (2010). "Nonlinear finite-element analysis software architecture using object composition." *Journal of Computing in Civil*

Engineering, 24(February), 95–107.

Michaud, D., and Léger, P. (2014). "Ground motions selection and scaling for nonlinear dynamic analysis of structures located in eastern North America." *Canadian journal of civil engineering*, 41(3), 232–244.

Moehle, J., and Deierlein, G. G. (2004). "A framework methodology for performance-based earthquake engineering." *Proceedings of the 13th World conference on earthquake engineering*, 3812–3814.

Nau, J. M., and Hall, W. J. (1984). "Scaling methods for earthquake response spectra." *Journal of Structural Engineering*, 110(7), 1533–1548.

Newmark, N. M. (1959). "A method of computation for structural dynamics." *Journal of the Engineering mechanics Division, ASCE*, 85, 67–94.

PEER Report 2009. (2013). *Evaluation of ground motion selection and modification methods: predicting median interstory drift response of buildings*.

"Qt." (2014). Computer program downloaded from <http://download.qt.io/archive/qt/5.3/5.3.1/> in March, 2014.

Reyes, J. C., and Kalkan, E. (2011). "Evaluation of the required number of records for the ASCE / SEI 7 ground motion scaling procedure." *US geological survey open-file report*, 1083(2011), 34.

Rezaeian, S., and Der Kiureghian, A. (2010). "Simulation of synthetic ground motions for specified earthquake and site characteristics." *Earthquake Engineering & Structural Dynamics*, 39(10), 1155–1180.

"SeismoSignal." (2015). Computer program downloaded from <http://seismosoft.com/SeismoSignal-v5.1.2> in March, 2015.

Shinozuka, M., and Deodatis, G. (1988). "Stochastic process models for earthquake ground motion." *Probabilistic Engineering Mechanics*, 3(3), 114–123.

Vanmarcke, E. H., and Gasparini, D. A. (1976). "Simulated earthquake motions compatible with prescribed response spectra." Massachusetts Institute of Technology, Department of Civil Engineering, Constructed Facilities Division.

Weng, Y., Tsai, K., and Chan, Y. (2008). "Development of a ground motion scaling method considering multi-mode effects." *In Proceedings*.

Whittaker, A. S., Atkinson, G. M., Baker, J. W., Bray, J. D., Grant, D. N., Hamburger, R., Haselton, C., and Somerville, P. G. (2012). "Selecting and scaling earthquake ground motions for performing response-history analyses." *Proceedings of Fifteenth World*

Conference on Earthquake Engineering, Lisboa, Portugal.

Appendices

Appendix A : Dynamic Analysis in *Rts*

A.1 *Rts* Code for Dynamic Analysis

In this section the dynamic analysis code implemented in *Rts* is listed. This would enable the future developers to understand the code and make modifications if necessary. The code is listed following a short description of what the code does.

The first step in the dynamic analysis is to define the Assembler, Linear Solver, Time Stepper, Time Step, End Time and Plot Interval which would be obtained as a user input:

```
Q_PROPERTY(QObject *Assembler READ getAssembler WRITE setAssembler)
Q_PROPERTY(QObject *LinearSolver READ getLinearSolver WRITE setLinearSolver)
Q_PROPERTY(QObject *TimeStepper READ getTimeStepper WRITE setTimeStepper)
Q_PROPERTY(double TimeStep READ getTimeStep WRITE setTimeStep)
Q_PROPERTY(double EndTime READ getEndTime WRITE setEndTime)
Q_PROPERTY(int PlotInterval READ getPlotInterval WRITE setPlotInterval)
```

Following the definitions, the user values are obtained and the values are set. For example the code for getting and setting the Assembler from the user is:

```
QObject * RLinearDynamicStructuralAnalysis::getAssembler() const
{
    return theAssembler;
}

void RLinearDynamicStructuralAnalysis::setAssembler(QObject *value)
{
    theAssembler = qobject_cast<RAssembler *>(value);
}
```

Once all the required data is obtained, the assembler function is called to get the number of free degrees of freedom:

```
// Call the assembler to get the number of free DOFs
int numDOFs = theAssembler->getNumFreeDOFs();
```

Next the mass, stiffness and damping matrices are defined as a square matrix of size equal to the number of degrees of freedom:

```
// Allocate memory for stiffness (K), mass (M) and damping (C) matrices
QVector< QVector<double> > K(numDOFs, QVector<double>(numDOFs));
QVector< QVector<double> > M(numDOFs, QVector<double>(numDOFs));
QVector< QVector<double> > C(numDOFs, QVector<double>(numDOFs));
```

The mass, damping and stiffness matrices values are obtained by calling the assembler:

```
// Establish M, K and C
theAssembler->assembleMatricesAndVectors(&M, &C, &K, 0, 0, 0.0);
```

Next memory is allocated for displacement, acceleration, velocity and external vectors of size equal to the number of degrees of freedom:

```
// Allocate memory for displacement, acceleration, velocity and external force vectors at time step
i and i+1
QVector<double> displacement(numDOFs);
QVector<double> displacementatiplus1(numDOFs);
QVector<double> velocity(numDOFs);
QVector<double> velocityatiplus1(numDOFs);
QVector<double> acceleration(numDOFs);
QVector<double> accelerationatiplus1(numDOFs);
QVector<double> PatTimeStepi(numDOFs);
QVector<double> PatTimeStepiplus1(numDOFs);
```

Displacement of the system is determined at the initial time step for gravity loading by calling the assembler:

```
// Establish external load vector at time step 0
theAssembler->assembleMatricesAndVectors(0, 0, 0, 0, &PatTimeStepi, 0.0);


// Solve for displacement at time step 0.0
theLinearSolver->solveLinearSOE(&K, &PatTimeStepi, &displacement);
```

Once all the required matrices and vectors are defined, a *for* loop is initiated from time equal to zero to last time step with each increment of time by the time step:

```
// Perform dynamic analysis
for (double time = 0; time <(theEndTime+theTimeStep); time += theTimeStep){
```

At every time step, external load is obtained by calling the assembler following which the values are sent to the *RTimeStepper* class for determining response:

```
// Assemble the vector of external forces at current time step
theAssembler->assembleMatricesAndVectors(0, 0, 0, 0, &PatTimeStepiplus1, time);

// Send values to Newmark time stepper to get response at current time step
theTimeStepper->conductNewmarkIteration(true, &M, &K, &C, &PatTimeStepiplus1, 0, &displacement,
&velocity, &acceleration, &displacementatiplus1, &velocityatiplus1, &accelerationatiplus1,
theTimeStep);
```

In the *RNewmarkTimeStepper* class first constant matrices and modified stiffness matrices are defined and computed as described in Equations (58), (60), (61) and (62):

```
// Declare a1, a2 and a3 matrices and modified stiffness matrix (k)
QVector< QVector<double> > k(numDOFs, QVector<double>(numDOFs));
QVector< QVector<double> > a1(numDOFs, QVector<double>(numDOFs));
```

```
QVector< QVector<double> > a2(numDOFs, QVector<double>(numDOFs));
QVector< QVector<double> > a3(numDOFs, QVector<double>(numDOFs));


// Determine a1, a2, a3 and stiffness matrix
for(int i=0; i<numDOFs; i++){

        for(int j=0; j<numDOFs; j++){

            a1[i][j] = ((1/(theParameterBeta*theTimeStep*theTimeStep))*(*M)[i][j]) +
((theParameterGamma/(theParameterBeta*theTimeStep))*((*C)[i][j]));
            a2[i][j] = ((1/(theParameterBeta*theTimeStep))*((*M)[i][j])) +
(((theParameterGamma/theParameterBeta)-1)*((*C)[i][j]));
            a3[i][j] = ((1/(2*theParameterBeta)-1)*((*M)[i][j])) +
(((theParameterGamma/(2*theParameterBeta))-1)*((*C)[i][j]));
            k[i][j] = a1[i][j] + (*K)[i][j];

        }
    }
```

Next external load vector is computed similar to Equation (59):

```
// Determine external load using the expression : F(i+1)=P(i+1)+a1*u[i]+a2*v[i]+a3*a[i]
    for(int i=0; i<numDOFs; i++){

        for(int j=0; j<numDOFs; j++){

            FatTimeStepiplus1[i] = FatTimeStepiplus1[i] + a1[i][j]*(*displacement)[j] +
a2[i][j]*(*velocity)[j] + a3[i][j]*(*acceleration)[j];
        }

        FatTimeStepiplus1[i] = FatTimeStepiplus1[i] + (*Pext)[i];
    }
```

The system of equations similar to Equation (63) are solved using the linear solver to determine the displacement at next time step:

```
// Solving system of equation for linear system
    if (Linear = true){

    // Determine displacement at time step i+1
    theLinearSolver->solveLinearSOE(&k, &FatTimeStepiplus1, displacementatiplus1);

    }
```

Following which the acceleration and velocity at the next time step are determined according to Equations (64) and (65):

```
// Acceleration and velocity at time step i+1
    for(int i=0; i<numDOFs; i++){

        // Determine increment in displacement
        delu[i] = (*displacementatiplus1)[i] - (*displacement)[i];

        // Determine velocity at time step i+1
        (*velocityatiplus1)[i] = ((theParameterGamma/(theParameterBeta*theTimeStep))*(delu[i])) -
(((theParameterGamma/theParameterBeta)-1)*(*velocity)[i]) -
(((theParameterGamma/(2*theParameterBeta))-1)*(theTimeStep*(*acceleration)[i]));

        // Determine acceleration at time step i+1
        (*accelerationatiplus1)[i] = ((1/(theParameterBeta*theTimeStep*theTimeStep))*delu[i]) -
((*velocity)[i]/(theParameterBeta*theTimeStep)) - ((1/(2*theParameterBeta)-1)*(*acceleration)[i]);

    }
```

The response values at the next time step are given to the previous time step. These values are sent back to the *RLinearDynamicStructuralAnalysis* class:

```
// Set response values at time step i+1 equal to zero and response at time step i equal to response
at time step i+1
    for( int i=0; i<numDOFs; i++){
        (*displacement)[i] = (*displacementatiplus1)[i];
        (*velocity)[i] = (*velocityatiplus1)[i];
        (*acceleration)[i] = (*accelerationatiplus1)[i];
        delu[i] = 0.0;
        FatTimeStepiplus1[i] = 0.0;
        (*velocityatiplus1)[i] = 0.0;
        (*accelerationatiplus1)[i] = 0.0;
    }
```

In the *RLinearDynamicStructuralAnalysis* class the deformations are then set to the nodes and the stress and strain are set in the components using the assembler tool:

```
// Give the new displacements to the nodes
theAssembler->setDeformations(&displacement);

// Call commit although this is a linear analysis, to make sure responses are set in the components
(max strain, etc)
theAssembler->commitState();
```

The plot is updated if the user has specified an output level as maximum at every plot interval:

```
// Update the plot
if ( (remainder(i, thePlotInterval) == 0.0) && (theOutputLevel >= RObject::Medium) ) {
            qDebug() << this->objectName() << "has reached time" << time;
            qDebug() << "";
            theEngine->update3D();
        }
        else if (time==0.0) {
            theEngine->update3D();
        }
```

The *RInelasticDynamicStructuralAnalysis* class is different in only one aspect from the *RLinearDynamicStructuralAnalysis* class, which is computation of displacement by solving system of equations. The following code is implemented in the *RInelasticDynamicStructuralAnalysis* class which is similar to Equations (51) to (55):

```
// Newton-Raphson ITERATIONS
numIter = 0;
resNorm = 10.0;

for(int i=0; i<numDOFs; i++)
           displacementatiplus1[i]=displacement[i];

while (resNorm > tol && numIter < 100) {

            numIter++;
```

```
            theAssembler->assembleMatricesAndVectors(0, 0, 0, &Pint, 0, time);
            theMatrixOperations.multiplyMatrixVector(false, 1.0, &a1, &displacementatiplus1, 1.0,
&Pint);


            // Compute the residual and its norm
            theMatrixOperations.subtractVectors(&FatTimeStepiplus1, &Pint, &Residual);


            resNorm = theMatrixOperations.norm(&Residual);


            // Compute displacement increment only if needed
            if (resNorm > tol) {

                // Compute displacement increment
                theLinearSolver->solveLinearSOE(&k, &Residual, &delu);


                // Compute total displacements
                theMatrixOperations.addVectors(&displacementatiplus1,&delu,&displacementatiplus1);


                // Set the new displacements in theStructuralModel
                theAssembler->setDeformations(&displacementatiplus1);
            }


            else {

                // The analysis converged at this time, so history variables can be committed
                theAssembler->commitState();

            }

        }
```

A.2 Input File for Dynamic Analysis of a Cantilever Column

The input file required for running linear dynamic analysis of cantilever column is as follows:

```
// FIXED PLANE
RPoint |ObjectName: pointA  |XCoordinate: -2      |YCoordinate: -2       |ZCoordinate: 0
RPoint |ObjectName: pointB  |XCoordinate:  2      |YCoordinate: -2       |ZCoordinate: 0
RPoint |ObjectName: pointC  |XCoordinate:  2      |YCoordinate:  2       |ZCoordinate: 0
RPoint |ObjectName: pointD  |XCoordinate: -2      |YCoordinate:  2       |ZCoordinate: 0

RFixedPLaneComponent |ObjectName: ground |Point1: pointA |Point2: pointB |Point3: pointC |Point4:
pointD



// POINTS
RPoint |ObjectName: point1  |XCoordinate: 1      |YCoordinate: 1         |ZCoordinate: 0
RPoint |ObjectName: point2  |XCoordinate: 1      |YCoordinate: 1         |ZCoordinate: 4



// COMPONENTS
RSteelIBeamComponent |ObjectName: column |Point1: point1 |Point2: point2 |MeshOption: 1 |WebHeight:
0.1 |FlangeWidth: 0.1 |WebThickness: 0.01 |FlangeThickness: 0.01 |Orientation: 0



// TOOLS
RInHouseLinearSolver |ObjectName: myLinearSolver |OutputLevel: Minimum
```

```
RInHouseProbabilityDistributions |ObjectName: myProbabilityDistributions |OutputLevel: Minimum
RTransformationMatrixAssembler |ObjectName: myAssembler |OutputLevel: Minimum
RLinearStaticStructuralAnalysis |ObjectName: myStaticAnalysis |OutputLevel: Maximum |Assembler:
myAssembler |LinearSolver: myLinearSolver
RNewmarkTimeStepper |ObjectName: myNewmarkTimeStepper |OutputLevel: Medium
|LinearSolver:myLinearSolver |ParameterBeta: 0.25 |ParameterGamma: 0.5
RLinearDynamicStructuralAnalysis |ObjectName: myDynamicAnalysis |OutputLevel: Minimum |Assembler:
myAssembler |LinearSolver: myLinearSolver |TimeStepper: myNewmarkTimeStepper |TimeStep: 0.01
|EndTime: 10.0


// MODELS
RScaledGroundMotionModel |ObjectName: theGMmodel |OutputLevel: Maximum |InputFile:Load.txt
RComponentResponseModel |ObjectName: myDynamicBuilding |OutputLevel: Maximum |StructuralAnalysis:
myDynamicAnalysis  |Responses: point2.XDisplacement;
```

A.3 Input File for Dynamic Analysis of the Building Example

Following is the input file for the building example:

```
//POINTS AT FIXED PLANE CORNERS
RPoint |ObjectName: PointA |XCoordinate: -1 |YCoordinate: -1 |ZCoordinate: 0
RPoint |ObjectName: PointB |XCoordinate: 16 |YCoordinate: -1 |ZCoordinate: 0
RPoint |ObjectName: PointC |XCoordinate: 16 |YCoordinate: 16 |ZCoordinate: 0
RPoint |ObjectName: PointD |XCoordinate: -1 |YCoordinate: 16 |ZCoordinate: 0


//OTHERS POINTS
RPoint |ObjectName: Point1L1 |XCoordinate: 0 |YCoordinate: 0 |ZCoordinate: 0
RPoint |ObjectName: Point2L1 |XCoordinate: 0 |YCoordinate: 8 |ZCoordinate: 0
RPoint |ObjectName: Point3L1 |XCoordinate: 0 |YCoordinate: 12 |ZCoordinate: 0

RPoint |ObjectName: Point4L1 |XCoordinate: 4 |YCoordinate: 0 |ZCoordinate: 0
RPoint |ObjectName: Point5L1 |XCoordinate: 4 |YCoordinate: 8 |ZCoordinate: 0
RPoint |ObjectName: Point6L1 |XCoordinate: 4 |YCoordinate: 12 |ZCoordinate: 0

RPoint |ObjectName: Point7L1 |XCoordinate: 8 |YCoordinate: 0 |ZCoordinate: 0
RPoint |ObjectName: Point8L1 |XCoordinate: 8 |YCoordinate: 8 |ZCoordinate: 0
RPoint |ObjectName: Point9L1 |XCoordinate: 8 |YCoordinate: 12 |ZCoordinate: 0

RPoint |ObjectName: Point10L1 |XCoordinate: 12 |YCoordinate: 0 |ZCoordinate: 0
RPoint |ObjectName: Point11L1 |XCoordinate: 12 |YCoordinate: 8 |ZCoordinate: 0
RPoint |ObjectName: Point12L1 |XCoordinate: 12 |YCoordinate: 12 |ZCoordinate: 0

RPoint |ObjectName: Point1L2 |XCoordinate: 0 |YCoordinate: 0 |ZCoordinate: 6
RPoint |ObjectName: Point2L2 |XCoordinate: 0 |YCoordinate: 8 |ZCoordinate: 6
RPoint |ObjectName: Point3L2 |XCoordinate: 0 |YCoordinate: 12 |ZCoordinate: 6

RPoint |ObjectName: Point4L2 |XCoordinate: 4 |YCoordinate: 0 |ZCoordinate: 6
RPoint |ObjectName: Point5L2 |XCoordinate: 4 |YCoordinate: 8 |ZCoordinate: 6
RPoint |ObjectName: Point6L2 |XCoordinate: 4 |YCoordinate: 12 |ZCoordinate: 6

RPoint |ObjectName: Point7L2 |XCoordinate: 8 |YCoordinate: 0 |ZCoordinate: 6
RPoint |ObjectName: Point8L2 |XCoordinate: 8 |YCoordinate: 8 |ZCoordinate: 6
RPoint |ObjectName: Point9L2 |XCoordinate: 8 |YCoordinate: 12 |ZCoordinate: 6

RPoint |ObjectName: Point10L2 |XCoordinate: 12 |YCoordinate: 0 |ZCoordinate: 6
RPoint |ObjectName: Point11L2 |XCoordinate: 12 |YCoordinate: 8 |ZCoordinate: 6
RPoint |ObjectName: Point12L2 |XCoordinate: 12 |YCoordinate: 12 |ZCoordinate: 6


//FIXED PLANE
RFixedPLaneComponent |ObjectName: ground |Point1: PointA |Point2: PointB |Point3: PointC |Point4:
PointD
```

```
//RC COLUMN COMPONENTS
RRCColumnComponent |ObjectName: Column1 |Point1: Point1L1 |Point2: Point1L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0

RRCColumnComponent |ObjectName: Column2 |Point1: Point2L1 |Point2: Point2L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0

RRCColumnComponent |ObjectName: Column3 |Point1: Point3L1 |Point2: Point3L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0

RRCColumnComponent |ObjectName: Column4 |Point1: Point4L1 |Point2: Point4L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0

RRCColumnComponent |ObjectName: Column5 |Point1: Point5L1 |Point2: Point5L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0

RRCColumnComponent |ObjectName: Column6 |Point1: Point6L1 |Point2: Point6L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0

RRCColumnComponent |ObjectName: Column7 |Point1: Point7L1 |Point2: Point7L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0

RRCColumnComponent |ObjectName: Column8 |Point1: Point8L1 |Point2: Point8L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0

RRCColumnComponent |ObjectName: Column9 |Point1: Point9L1 |Point2: Point9L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0

RRCColumnComponent |ObjectName: Column10 |Point1: Point10L1 |Point2: Point10L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0

RRCColumnComponent |ObjectName: Column11 |Point1: Point11L1 |Point2: Point11L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0

RRCColumnComponent |ObjectName: Column12 |Point1: Point12L1 |Point2: Point12L2 |MeshOption: 1
|CrossSectionDepth: 0.2 |CrossSectionWidth: 0.2 |ConcreteStrength: C30  |
LongitudinalReinforcementRatio: 0.02 |Orientation: 0



//SLAB COMPONENTS
RRCSlabComponent|ObjectName: Slab1 |Point1: Point1L2 |Point2: Point10L2 |Point3: Point12L2 |Point4:
Point3L2  |MeshOption: 303 |Thickness: 1.0 |ConcreteStrength: 30000000



// TOOLS
RInHouseLinearSolver |ObjectName: myLinearSolver |OutputLevel: Minimum
RInHouseProbabilityDistributions |ObjectName: myProbabilityDistributions |OutputLevel: Minimum
RTransformationMatrixAssembler |ObjectName: myAssembler |OutputLevel: Minimum
RLinearStaticStructuralAnalysis |ObjectName: myStaticAnalysis |OutputLevel: Maximum |Assembler:
myAssembler |LinearSolver: myLinearSolver
RNewmarkTimeStepper |ObjectName: myNewmarkTimeStepper |OutputLevel: Medium
|LinearSolver:myLinearSolver |ParameterBeta: 0.25 |ParameterGamma: 0.5
RInelasticDynamicStructuralAnalysis |ObjectName: myInelasticDynamicAnalysis |OutputLevel: Minimum
|Assembler: myAssembler |LinearSolver: myLinearSolver |TimeStep: 0.01 |EndTime: 35.0



// MODELS
RScaledGroundMotionModel |ObjectName: theGMmodel |OutputLevel: Maximum  |InputFile: Wave3.txt
RComponentResponseModel |ObjectName: myDynamicBuilding |OutputLevel: Maximum |Gravity: Yes
|Earthquake: theGMmodelResponse |Wind: None |Snow: None |LiveLoad: None |Deterioration: None |
Responses: Point1L2.XDisplacement; Point2L2.XDisplacement; Point3L2.XDisplacement
|StructuralAnalysis: myInelasticDynamicAnalysis
```

Appendix B : Scaled Ground Motion Models in *Rts*

B.1 *Rts* Code for Scaled Ground Motion Models

In this section the code implemented for scaling ground motion models that is the *RScaledGroundMotionModel*, *RPGAScaledGroundMotionModel* and *RSa_T1_ScaledGroundMotionModel* are listed. For all the three models the code for getting the ground motion input file and reading it is the same. The code for getting the input file:

```
QString RScaledGroundMotionModel::getInputFile() const
{
    return theInputFile;
}

void RScaledGroundMotionModel::setInputFile(QString value)
{
    theInputFile = RDomain::stringToFilePath(value);
}
```

Once the input file is obtained it is read and stored as a time series:

```
// Open the file
    theInputFile.remove('"');
    if (theInputFile.isEmpty()) {
        qCritical() << "The scaled ground motion model" << this->objectName() << "needs an input
file name.";
        return -1;
    }
    QFile file(theInputFile);
    if (!file.open(QFile::ReadOnly /*| QFile::Text*/)) {
        qCritical() << "Error: Could not open the file" << theInputFile << ".";
        return -1;
    }

// Read the file content into a QString
    QTextStream stream(&file);
    stream.seek(0);


// Tokenize the entire input file, i.e., the string, by creating a line break at "\n" and create a
string list LINES
    QString verticalRegExpString =
QString("[\n\v\r%1%2]").arg(QChar(QChar::ParagraphSeparator)).arg(QChar(QChar::LineSeparator));
    QRegExp verticalRegExp = QRegExp(verticalRegExpString);
    QStringList fileContentStringList;
    QString line;
    QStringList lineList;
    stream.setAutoDetectUnicode(true);
    do {
        line = stream.readLine();
        if (!line.isEmpty()) {
            lineList = line.split(verticalRegExp, QString::SkipEmptyParts);
            fileContentStringList.append(lineList);
        }
    } while (!line.isNull());

// Get the number of lines
    int numLines = fileContentStringList.count();

// Tokenize each line
    QRegExp regExp = QRegExp("[ ,;\t]");
    int columnCount = 0;
    for (int i = 0; i < numLines; i++) {
```

```
        QStringList valueList = fileContentStringList[i].split(regExp, QString::SkipEmptyParts);

        columnCount = valueList.count();

        if (columnCount != 4) {
            qCritical() << "The ground motion file given to" << this->objectName() << "does not
have 4 columns.";
        }

// Declare the vector that temporarily stores the numbers in one line
        QVector<double> numbersInReadLine(columnCount);
// Read number by number in this line (line number i)
        for (int j = 0; j < columnCount; j++) {
            QString tempStr = valueList[j].trimmed();
            bool ok;
            double tempValue = tempStr.toDouble(&ok);
            if (!ok) {
                qCritical() << "Error: Could not read the value number" << j + 1 << "at the line"
<< i + 1 << "in the input file.";
                return -1;
            }

            numbersInReadLine[j] = tempValue;
        }

        theInputTimeSeries->setXYZvalues(numbersInReadLine[0], numbersInReadLine[1],
numbersInReadLine[2], numbersInReadLine[3]);
    }

    file.close();
```

Following is the scaling procedure code for the *RScaledGroundMotionModel* class:

```
// Do something with the input time series, such as scaling it, and then save it to the response
    int numEntries = theInputTimeSeries->getNumEntries();
    double t, x, y, z;
    double g;
    g=9.81;

    for (int i=0; i<numEntries; i++) {

        t = theInputTimeSeries->getTimeItem(i);
        x = theInputTimeSeries->getXitem(i);
        y = theInputTimeSeries->getYitem(i);
        z = theInputTimeSeries->getZitem(i);


        x=9.81*x*theScalingFactor;
        y=9.81*y*theScalingFactor;
        z=9.81*z*theScalingFactor;

        theOutputTimeSeries->setXYZvalues(t, x, y, z);
    }
```

Following is the scaling procedure code for the *RPGAScaledGroundMotionModel* class:

```
// Determine the maximum ground acceleration value for the ground motion record
        // Xdirection
        if((x>0)&&(x>maxX))
            maxX=x;
        if((x<0)&&(x<-1*maxX))
            maxX=-1*x;

        // Ydirection
        if((y>0)&&(y>maxY))
            maxY=y;
        if((y<0)&&(y<=-1*maxY))
            maxY=-1*y;
```

```
        // Zdirection
        if((z>0)&&(z>maxZ))
            maxZ=z;
        if((z<0)&&(z<-1*maxZ))
            maxZ=-1*z;

    // Determine the scaling factor for PGA scaling
    // X direction
    double PGAScalingFactorX;
    if(maxX==0)
        PGAScalingFactorX=0;
    else
        PGAScalingFactorX = theTargetPGA/maxX;

        x=PGAScalingFactorX*x*9.81;
```

Following is the scaling procedure code for the *RSa_T1_ScaledGroundMotionModel* class:

```
    // Declare other constants that are to be used in the newmark time stepping method
    double m = 1.0;
    double k = 4.0*PI*PI/(theFundamentalTimePeriod*theFundamentalTimePeriod);
    double c = 2*dampingRatio*2*PI/theFundamentalTimePeriod;

    double k1, a1, b1;
    a1 = (4*m/(TimeInterval*TimeInterval))+(2*c/TimeInterval);
    b1 = (4*m/TimeInterval)+c;
    k1 = k+a1;


    // Declare response quantities like velocity, acceleration, displacement required for newmark
timestepping method
    double v,a, u, deltau, p, pnext, unext, vnext, anext, response;


    // External load
    p = theInputTimeSeries->getXitem(0);
    p = p*9.81;
    a = p/m;
    double f;


    // Start the loop for Newmark method for calculation of acceleration, velocity and displacement
    for (int time = 0; time <(numEntries-1); time++)
    {

        pnext = theInputTimeSeries->getXitem(time+1);
        pnext = 9.81*pnext;
        f = pnext+ a1*u + b1*v + m*a;

        // Displacement, acceleration and velocity at next time period
        unext = f/k1;
        deltau = unext-u;
        vnext = (deltau*2/TimeInterval)-v;
        anext = (4*deltau/(TimeInterval*TimeInterval))-(4*v/TimeInterval)-a;

        u = unext;
        v = vnext;
        a = anext;

        // Determination of maximum displacement
        if((unext>0)&&(unext>=response))
            response=unext;
        if((unext<0)&&(unext<=-1*response))
            response=-1*unext;
    }


    // Determine scaling factor at fundamental time period
    double SaRecord;
    double omega;
    omega = 2*PI/theFundamentalTimePeriod;
```

```
// Sa=omega^2*umax
SaRecord = response*omega*omega/9.81;


// Scaling factor = Sa(target)/Sa(record)
double SaScalingFactor;
SaScalingFactor = theTargetSpectralAcceleration/SaRecord;
```

Appendix C : Synthetic Ground Motion Model in *Rts*

C.1 *Rts* Code for Synthetic Ground Motion Model

The code for the synthetic ground motion model implemented in the class *RSyntheticGroundMotrionModel* of *Rts* is listed in this section.

First the user input are defined which is similar to the one previously explained. Next random variables with mean equal to zero and unit standard deviation are generated:

```
    // Create new standard normal random variables
    theAllParametersList.clear();
    RContinuousRandomVariable *theRV;
    for (int i = 0; i < theNumberOfRandomVariables; i++) {
        theDomain->createObject(this->objectName() + QString("RV_%1").arg(i),
"RContinuousRandomVariable");
        theRV = qobject_cast<RContinuousRandomVariable *>(theDomain->getLastAddedObject());
        theRV->setDistributionType(QString("Normal (mean, stdv)"));
        theRV->setMean(QString("0.0"));
        theRV->setStandardDeviation("1.0");
        theRV->setCurrentValue(0.1);
        theRVlist->append(theRV);
        theAllParametersList << theRV;
    }
```

Next the number of modulating functions are determined and checked if the number of parameters for each of the modulating functions are appropriate:

```
// Extract the number of modulating functions from the list given by the user
numberOfModulatingFunction=theGammaModulatingFunctionParameterAList.size();


if((numberOfModulatingFunction!=theGammaModulatingFunctionParameterBList.size())||(numberOfModulati
ngFunction!=theGammaModulatingFunctionParameterCList.size())){
        qDebug()<<"The number of parameters of each modulating function should be same";
        return 1 ;
    }
```

The ground motion record is generated according to the algorithm described previously. The equations used in the code listed below are similar to Equations (84), (90) and (91):

```
    // Getting pointer to the response time series, and clean it
    RTimeSeries *theOutputTimeSeries = parameterToTimeSeries(theGroundMotionResponse);
    if (!theOutputTimeSeries) {
        qCritical() << "Error: The response object" << theGroundMotionResponse->objectName() <<
"does not contain a times series object.";
        return -1;
    }
    theOutputTimeSeries->clean();

    // Start the loop over the number of time steps
    for(double time=theStartTime; time<=theEndTime; time=time+timeStepofGroundMotion)
    {
        sumOfModFunctionAndIRF=0.0;


        // Start the loop over modulating functions
```

```
        for(int j=0; j<numberOfModulatingFunction; j++)
        {

            // Amplitude of modulating function at this particular time
            a = theGammaModulatingFunctionParameterAList[j]->getCurrentValue();
            b = theGammaModulatingFunctionParameterBList[j]->getCurrentValue();
            c = theGammaModulatingFunctionParameterCList[j]->getCurrentValue();
            modulatingFunctionAmplitude=a*pow(time,b)*exp(-c*time);

            sumOfImpulseResponseFunction=0.0;


            // Start the loop over the random variables
            for(int k=0; k<theNumberOfRandomVariables; k++){


                // Break the loop if current time is less than the product of time step and number
of random variables
                if((time-timeStep*k)<0.0){
                    break;
                }


                w = theFilterFrequencyList[j]->getCurrentValue();
                damping = theFilterDampingList[j]->getCurrentValue();


                // Compute the damped natural frequency of the filter
                wd = w*sqrt(1-pow(damping,2.0));


                // Computer the amplitude of standard linear oscillator displacement filter at this
particular time
                filterAmplitude = sin(wd*(time-timeStep*k))*exp(-damping*w*(time-timeStep*k));


                // Compute the response function at this particular time
                sumOfImpulseResponseFunction += theRVlist->at(k)->getCurrentValue() *
filterAmplitude;

            }

            sumOfModFunctionAndIRF += sumOfImpulseResponseFunction * modulatingFunctionAmplitude;

        }

        // Give the value computed as time series
        x = sumOfModFunctionAndIRF;
        theOutputTimeSeries->setXYZvalues(time, x, y, z);
```

C.2 Sample Input File for Synthetic Ground Motion Model

Following is the input file for the synthetic ground motion model with two modulating function:

```
// GROUND MOTION MODEL
RConstant |ObjectName: Gamma_MF_A |CurrentValue: 0.05
RConstant |ObjectName: Gamma_MF_A2 |CurrentValue: 0.000000007

RConstant |ObjectName: Gamma_MF_B |CurrentValue: 5.0
RConstant |ObjectName: Gamma_MF_B2 |CurrentValue: 10

RConstant |ObjectName: Gamma_MF_C |CurrentValue: 1.5
RConstant |ObjectName: Gamma_MF_C2 |CurrentValue: 0.7

RConstant |ObjectName: Filter_Damping |CurrentValue: 0.05
RConstant |ObjectName: Filter_Damping2 |CurrentValue: 0.05
```

```
RConstant |ObjectName: Filter_Frequency |CurrentValue: 20.0
RConstant |ObjectName: Filter_Frequency2 |CurrentValue: 60.0

RSyntheticGroundMotionModel |ObjectName: theGMmodel1 |OutputLevel: Medium
|GammaModulatingFunctionParameterAList: Gamma_MF_A2;  |GammaModulatingFunctionParameterBList:
Gamma_MF_B2;  |GammaModulatingFunctionParameterCList: Gamma_MF_C2;  |FilterDampingList:
Filter_Damping2;  |FilterFrequencyList: Filter_Frequency2;  |NumberOfRandomVariables: 200
|StartTime: 0 |EndTime: 50



// FINITE ELEMENT MODEL
RPoint |ObjectName: pointA  |XCoordinate: -2      |YCoordinate: -2       |ZCoordinate: 0
RPoint |ObjectName: pointB  |XCoordinate:  2      |YCoordinate: -2       |ZCoordinate: 0
RPoint |ObjectName: pointC  |XCoordinate:  2      |YCoordinate:  2       |ZCoordinate: 0
RPoint |ObjectName: pointD  |XCoordinate: -2      |YCoordinate:  2       |ZCoordinate: 0

RFixedPLaneComponent |ObjectName: ground |Point1: pointA |Point2: pointB  |Point3: pointC |Point4:
pointD

RPoint |ObjectName: point1  |XCoordinate: 1      |YCoordinate: 1         |ZCoordinate: 0
RPoint |ObjectName: point2  |XCoordinate: 1      |YCoordinate: 1         |ZCoordinate: 4

RSteelIBeamComponent |ObjectName: column |Point1: point1 |Point2: point2  |MeshOption: 1
|WebHeight: 0.1 |FlangeWidth: 0.1 |WebThickness: 0.01 |FlangeThickness: 0.01 |Orientation: 0

RComponentResponseModel |ObjectName: myDynamicBuilding |OutputLevel: Maximum |StructuralAnalysis:
myDynamicAnalysis  |Earthquake: theGMmodel1Response |Responses:  point2.XDisplacement;



// TOOLS
RInHouseLinearSolver |ObjectName: myLinearSolver  |OutputLevel: Minimum
RInHouseProbabilityDistributions |ObjectName: myProbabilityDistributions  |OutputLevel: Minimum
RTransformationMatrixAssembler |ObjectName: myAssembler |OutputLevel: Minimum
RLinearStaticStructuralAnalysis |ObjectName: myStaticAnalysis  |OutputLevel: Maximum |Assembler:
myAssembler |LinearSolver: myLinearSolver
RInHouseProbabilityDistributions |ObjectName: theProbDist
RNatafProbabilityTransformation |ObjectName: myProbTransformation |ProbabilityDistributions:
theProbDist |LinearSolver: myLinearSolver |OutputLevel: Minimum
RInHouseRandomNumberGenerator |ObjectName: myGenerator
RNewmarkTimeStepper |ObjectName: myNewmarkTimeStepper |OutputLevel: Medium |LinearSolver:
myLinearSolver |ParameterBeta: 0.25 |ParameterGamma: 0.5
RLinearDynamicStructuralAnalysis |ObjectName: myDynamicAnalysis  |OutputLevel: Minimum |Assembler:
myAssembler |LinearSolver: myLinearSolver |TimeStepper: myNewmarkTimeStepper |TimeStep: 0.01
|EndTime: 50.0 |PlotInterval: 20
RSamplingModel |ObjectName: theSamplingModel |OutputLevel: Minimum  |InputParameter:
myDynamicBuildingpoint2XDisplacement |Threshold: 1.0 |ProbabilityTransformation:
myProbTransformation   |MaxSamples: 1 |PlotInterval: 100 |RandomNumberGenerator: myGenerator
```