PACIFIC EARTHQUAKE ENGINEERING
RESEARCH CENTER

Finite Element Reliability and Sensitivity Methods for Performance-Based Earthquake Engineering

Terje Haukaas
Department of Civil and Environmental Engineering
University of British Columbia

and

Armen Der Kiureghian
Department of Civil and Environmental Engineering
University of California, Berkeley

PEER 2003/14
APRIL 2004

Finite Element Reliability and Sensitivity Methods for Performance-Based Earthquake Engineering

Terje Haukaas
Department of Civil Engineering
University of British Columbia

and

Armen Der Kiureghian
Department of Civil and Environmental Engineering
University of California, Berkeley

PEER Report 2003/14
Pacific Earthquake Engineering Research Center
College of Engineering
University of California, Berkeley

April 2004

ABSTRACT

The work in this report is motivated by the performance-based engineering approach advocated by PEER. A comprehensive, object-oriented software framework for finite element sensitivity and reliability analysis is developed. The work builds on the existing software OpenSees.

An essential ingredient in finite element reliability analysis is accurate, consistent and efficient computation of response sensitivities. Using the direct differentiation method, a unified formulation of finite element response sensitivities with respect to material, load and shape parameters is developed and implemented. Shape sensitivity results allow inclusion of uncertainty in nodal coordinates in reliability analysis.

The developed software framework is used to investigate and address challenges particular to nonlinear finite element reliability analysis. As a result, smoothed material models, modifications in existing search algorithms, and a search algorithm hitherto not used in reliability analysis are developed. The first-order reliability method and the importance sampling method are used for computing probabilities and mean out-crossing rates, the latter for dynamic problems.

A User's and Developer's Guide for the sensitivity and reliability modules in OpenSees is developed. Several numerical examples, including a highway bridge used as a test-bed by PEER, are presented to demonstrate the new capabilities of the software.

ACKNOWLEDGEMENTS

This research investigation is primarily funded by the Pacific Earthquake Engineering Research (PEER) Center through the Earthquake Engineering Research Centers Program of the National Science Foundation under award number EEC-9701568. This support is gratefully acknowledged.

The first author would like to acknowledge The Research Council of Norway for a three-year doctoral fellowship and the U.S.-Norway Fulbright Foundation for a Fulbright Fellowship.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 BACKGROUND: PERFORMANCE-BASED EARTHQUAKE ENGINEERING

In recent decades, a revolution in computer efficiency and capacity has occurred. Simultaneous advances in the fields of mechanics and structural engineering have made it possible to simulate the behavior of complex structures on the computer. This has provided a strong impetus for the introduction of the performance-based engineering approach. The client, or society, may now require information about how a structure meets prescribed performance criteria for a given hazard level. This approach is obviously a lot more informative than mere satisfaction of prescriptive code regulations, and is the approach advocated by the Pacific Earthquake Engineering Research (PEER) Center.

Predictions of structural performance can only be done in a probabilistic sense. Unavoidable uncertainties are present in the geometry, material, and load parameters of structures as well as in the model itself and in the analysis procedure. Hence, the field of uncertainty analysis and structural reliability is becoming a mainstream topic as the field of structural engineering enters the performance-based engineering paradigm.

The finite element method is currently the dominating tool for simulating structural behavior. A coupling of this method with reliability analysis algorithms leads to the finite element reliability method described in this work. The first coupling between FORM (first-order reliability method) reliability analysis and the finite element method is found in Der Kiureghian and Taylor (1983). Since then, a number of advances have been reported, including those by Liu and Der Kiureghian (1991a), Gutierrez *et al.* (1994), Zhang and Der Kiureghian (1997), Der Kiureghian and Zhang (1999), Sudret and Der Kiureghian (2000), Imai and Frangopol (2000), Haldar and Mahadevan (2000), and Frier and Sorensen (2003). Such methods address the key issue in performance-based engineering. Based on performance criteria mandated by the client or the society, probability estimates for reaching specified structural performance thresholds are computed. In addition, sensitivity and importance measures for the model parameters are available.

Structural failures normally occur in the nonlinear structural response range. It is therefore imperative that inelastic material behavior and geometric nonlinearity are considered in finite element reliability analysis. The need for a modern and comprehensive computational framework to reveal and address challenges in such analyses provides the motivation for the present study. This study present the first attempt at incorporating reliability methods in an open-source, general-purpose finite element software framework.

1.2 OPENSEES AND OBJECT-ORIENTED PROGRAMMING

The work in this report extends the OpenSees software framework. OpenSees (Open System for Earthquake Engineering Simulation) is intended to serve as a computational platform for research at PEER. It is designed to compute the seismic response of structural and geotechnical systems. The source code of this software including the models developed as part of this report, can be found at the website
http://opensees.berkeley.edu, where additional information about OpenSees can be obtained.

Software developed within an academic setting is often prone to developer discontinuity, as students graduate and leave to pursue their careers. A nontransparent, personalized software design is often employed. The end result often is a patchwork-type software. This is not desirable for several reasons. Such software often become increasingly difficult to debug and maintain and extensions can be added only in an ad-hoc manner. Furthermore, the cost of climbing the learning curve for new students/developers who wish to utilize the software tends to become discouragingly high with time. This adversely affects motivation and frequently leads to a decision to abandon the existing software. These time- and resource-costly development cycles clearly are not desirable. These problems can be remedied by the software architecture enabled by the object-oriented programming approach. Object-oriented programming is employed in all implementations in OpenSees.

The introduction of object-oriented programming has brought with it a revolution in software development (Deitel and Deitel 1998). This revolution is based on the notion of standardized, interchangeable software components. These components are called “objects” or, abstractly, “classes.” Objects are instantiated at run-time based on specifications made by the developer in the corresponding classes. Each class, and hence object, may contain member functions, often called ”methods,” and member data.

Detailed specification of the methods and the data members is found in the class “interfaces.” The

information in the class interfaces contains the information key to understanding an object-oriented software framework. Appendix 6.2 describes the class interfaces used for the implementations in this study. At the OpenSees website, these specifications are provided in separate files with the `.h` extension. The interface of a class contains detailed specification of its methods, including the argument list and return type, and data members. This is all the information needed to make use of an object-oriented software library. Detailed knowledge of the algorithms that implement the promised features of a class, provided in `.cpp` files, is not necessary. (Such information is useful, however, if one insists on not making a "black box" use of the software.) The class interfaces facilitate the transparent nature of object-oriented programming. Their structure is common to all object-oriented software. With general knowledge of the syntax rules of the programming language, a user is able to understand the software architecture. Such software design has extensibility and maintainability as an integral part of the software design.

In object-oriented programming, increased emphasis is placed on the planning and design phase of the software development. The task at hand is abstracted and separated into logical components. Member functions are designed to handle the interaction between objects and to perform operations on member data. Various design patterns can be employed, depending on the structure of the problem (Gamma *et al.* 1995). In the present work, as in the core OpenSees framework, an analysis-domain decomposition is employed. A framework of reliability analysis components acts on the domain that contains random variables, correlation structures, performance functions, etc. Features of object-oriented programming that enable such developments are now described.

Object-oriented programming is based on four key principles. These are general concepts, which, in fact, are independent of the programming language. The concept of *abstraction* addresses transformation of a real-world procedure into abstract data types, e.g., classes. Hence, data abstraction is the process of organizing the problem into logical, self-contained components that interact. The planning phase of object-oriented software development cannot be overemphasized. The second principle, the concept of *encapsulation*, addresses a security concern. It is unsafe not to have clear rules regarding which part of the code has access to modify the model data. In general, data access should be restricted as much as possible. For example, it would be unsafe programming that could potentially lead to extremely difficult debugging if data objects such as nodal coordinates were passed around and could be modified by any subroutine. In object-oriented programming, this is handled by usually allowing only the methods of a class to modify its data members. Thirdly, the principle of *inheritance* permits the organization of classes into a framework of subclasses, which inherit functionality from their associated base classes. Such a hierarchy can have several layers.

This concept is used extensively in the implementations presented in this study. For instance, a base class named `ProbabilityTransformation` exists in the reliability analysis framework. This class promises features such as transformation to/from the standard normal space and computation of the corresponding Jacobian matrix, but it does not provide any specific implementations. For this reason it is called a "virtual class." A specific implementation of the probability transformation task is provided by the `NatafProbabilityTransformation` subclass. This leads directly to the final principle of object-oriented programming, namely *polymorphism*. This powerful concept further enhances the extensibility of the software by allowing virtual methods to be created. This means that the base classes contain virtual methods, namely methods without any implementations, that are implemented by any number of subclasses. That is, a software framework can be built using base classes, which make calls to virtual methods. Subclasses that actually carry out the promised features can be implemented without making modifications to the framework itself.

Figure 1.1 illustrates the features described above. The concepts of inheritance and polymorphism are indicated by a base class, which contains a virtual member function. The promises made by this virtual method in the base class are implemented by two subclasses, each presumably solving the task in a different way. The concept of composition implies that a class can contain data instances of other classes. For instance, a software component responsible for iterating towards the design point may contain a probability transformation, an object to compute the value of the performance function, among others. Such aggregated classes that act as data members are referred to as "referenced" classes.

In addition to the advantages outlined above, the object-oriented programming approach lends itself to development of software libraries. Instead of downloading a packaged code, a user familiar with the principles of object-oriented programming can download and make use of single software components based on the specifications provided in the class interfaces.

The advantages of the above-described features of object-oriented programming in developing software are presented this report. In particular, the analysis procedures are disaggregated into a framework of interacting components. This software framework is easily extendable with new algorithms to solve tasks such as evaluating performance functions, computing search direction vectors, and selecting the step sizes. In this manner, unforeseen future developments are accommodated.

A disadvantage of the object-oriented programming approach may be a slight increase in the computational time on tasks not associated with the finite element or reliability computations, but rather with interaction between objects. It is believed, however, that the time and resources saved in learning, debugging, maintaining, extending and continually using an object-oriented software

framework is well worth the possible sacrifice in computational time. Studies show that the increase in computational effort with the object-oriented approach, as compared with the common procedural approach, is of the order of 10-15% (McKenna 1997).

1.3 PREVIOUSLY DEVELOPED SOFTWARE

Previously developed software for reliability and sensitivity analysis are categorized into three groups: (1) general purpose reliability codes, (2) finite element reliability codes and (3) finite element codes extended to compute response sensitivities. For software in the first category the probabilistic model is specified by algebraic expressions or user-defined algorithms involving the basic random variables. Software in the second category enable reliability analysis of structures represented by finite element models. In the first category are software such as CalREL (Liu *et al.* 1989), COMREL/SYSREL (Reliability Consulting Programs 2003), ISPUD (IfM 2003), PROBAN (Det Norske Veritas 2003) and UNIPASS (PredictionProbe, Inc. 2003). Software in the second category are either couplings between separate codes, exemplified by the coupling of CalREL with the finite element code FEAP (Taylor 2003) and UNIPASS with the finite element code NASTRAN (MSC Software 2003). Alternative finite element reliability codes include COSSAN (IfM 2003), DAKOTA (Sandia National Laboratories 2003), FERUM (Haukaas *et al.* 2003), NESSUS (Southwest Research Institute 2003) and STRUREL (Reliability Consulting Programs 2003). To our knowledge, with the exception of the reliability codes coupling with FEAP and NASTRAN, all existing finite element reliability codes are limited to linear structural models.

Extended finite element codes which produce response sensitivities can be accomplished by employing finite difference methods or the Direct Differentiation Method. Of interest in this study is the latter method which has unique efficiency and accuracy properties. Work by Zhang and Der Kiureghian (1997) extended the finite element code FEAP (Taylor 2003) with such capabilities for the J_2 plasticity material. Similar work was performed by Roth and Grigoriu (2001) for the finite element code DIANA (TNO Building and Construction Research 2003). Some codes, such as ABAQUS (ABAQUS, Inc. 2003), employ a semi-analytical approach that combines direct differentiation with finite difference computations to obtain response sensitivities.

The software developed in this study is unique in several aspects, including the focus on nonlinear structural problems and the open-source, object-oriented software architecture.

1.4 OBJECTIVES AND SCOPE

The primary objective of this study is to develop a modern and comprehensive computational framework for nonlinear finite element reliability analysis. Programming features that facilitate easy maintenance and extensibility of the software are important goals. Particular attention is given to obtaining accurate, consistent, and efficiently computed response sensitivities. Discontinuity in response sensitivities, which poses a challenge in convergence of reliability analysis is to be addressed. Documentation of the software implementations are to be provided for users and developers. The developed state-of-the-art software is used to identify and address challenges particular to reliability analysis of inelastic static and dynamic problems.

1.5 ORGANIZATION

Following the introduction of Chapter 1, finite element response sensitivity equations are derived in Chapter 2. Starting from the fundamental equations of the finite element method, general top-level sensitivity equations are first derived. Thereafter, sensitivity equations for particular parameters, materials, and elements are developed. Important computer implementation issues related to inelastic static and dynamic problems are treated. New developments include shape sensitivity equations, sensitivity results for smoothed material models, and a proof of continuity of the displacement sensitivity at points of elastic unloading. In Chapter 3 a review of the implemented reliability methods is provided, together with developments to address challenges particular to nonlinear finite element reliability methods. A User's Guide for the new options for reliability and sensitivity analysis in OpenSees is presented in Chapter 4. Chapter 5 presents numerical studies using the new software framework. An array of examples are provided to demonstrate the various features of the software. A comprehensive practical example involving a freeway overpass bridge (PEER's I-880 testbed) is included. Chapter 6 summarizes the main findings of the study and presents suggestions for further study and development.

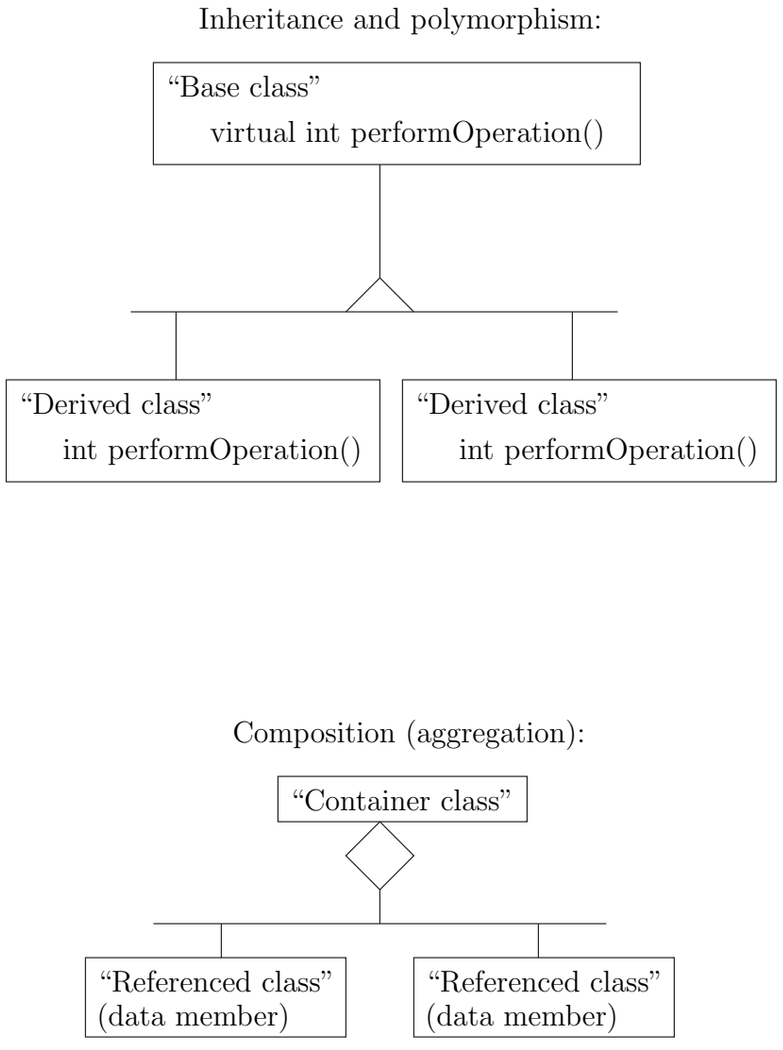

Figure 1.1: Symbolic representation of the principal concepts of an object-oriented software framework.

2 Finite Element Response Sensitivity Analysis

Derivatives of response quantities obtained from a finite element code with respect to model parameters are desirable for a number of reasons. Such sensitivity results may be used in a variety of applications, including (a) as indicators of parameter importance, guiding the allocation of resources for gathering information, (b) in assessing the effect of parameter uncertainties on the response, (c) in determining search directions in optimal design and system identification, and (d) for finding in the first-order reliability method the so-called design point and for reliability sensitivity analysis. The motivation in this study stems from applications to uncertainty propagation and reliability analysis. There, gradients of limit-state functions involving finite element response quantities are needed, as described in Chapter 2.12.2. It is essential that such gradients are computed accurately, efficiently, and consistent with the approximations inherent in the computation of the response quantities themselves. The Direct Differentiation Method (DDM) (Arora and Haug 1979, Zhang and Der Kiureghian 1993, Kleiber *et al.* 1997, Roth and Grigoriu 2001) is ideal for this purpose, at the cost of an initial investment of effort in deriving and implementing sensitivity algorithms into the finite element code.

The response sensitivity (or response gradient) is a measure of the change in a response quantity due to a unit change in a system parameter. In the context of this report, the parameters may describe material properties, cross-sectional geometry, nodal coordinates or applied loads of a finite element model. The structural response refers to any quantity that may be used to characterize the system behavior. Typically, these include deformations, such as displacements, rotations, and strains; forces, such as bending moments, shear forces, axial forces or stresses; or integrated quantities, such as dissipated energy and accumulated damage.

This chapter presents a unified framework for finite element response sensitivity analysis. First, the top-level sensitivity equations are outlined. In subsequent sections, equations are developed for particular parameters. These include nodal coordinates, leading to shape sensitivity equations.

Material models are developed and presented together with their sensitivity equations. These models attempt to remedy the gradient discontinuity problem that arises with certain conventional material models. The analytical models and derivations presented in this chapter are all implemented in OpenSees and are available for download and use from the web site `http://opensees.berkeley.edu`.

2.1 GENERAL RESPONSE SENSITIVITY EQUATIONS BY DIRECT DIFFERENTIATION

The use of response sensitivities in reliability analysis poses three requirements for the sensitivity computations, namely, consistency, efficiency and accuracy. These are all addressed by the DDM. Consistency with the computed response is provided by the fact that it is the discretized finite element response equations themselves that are differentiated. Sensitivity derivations are performed after the space- and time-discretization of the boundary value problem is carried out. The sensitivity equations are then implemented on the computer as an extension of the finite element code so that sensitivity results are produced along with the response itself. Efficiency is achieved because no re-run of the finite element analysis is necessary, as would be the case in a finite difference approach. The response sensitivity is solved from a linear equation upon convergence of the finite element response in each step.

The accuracy of sensitivity results is a twofold issue. If the accuracy is compared to the "exact" analytical solution of the boundary value problem, then the accuracy of the sensitivity results of an order lower than the response itself is observed. This is analogous to the reduced accuracy of derivative responses, such as strain and stress, compared to the accuracy of primary displacement responses. Convergence studies performed by refining the finite element mesh confirm slower convergence of sensitivity results, compared to that of the response itself (Gu and Conte 2003). However, this type of accuracy is not of primary interest in application of sensitivity results in reliability analyses. Of interest here is the accuracy of the sensitivity result relative to the exact derivative of the approximate (discretized) finite element response. From this viewpoint, the accuracy of the DDM is of the same precision as the response itself, as opposed to approximate finite difference methods.

A number of researchers have contributed to the field of response sensitivity analysis. Early works include those of Frank (1978), Ray *et al.* (1978) and Arora and Haug (1979). While the sensitivity of linear systems has long been a wellestablished field, developments for nonlinear structural problems have appeared throughout the past decade. The DDM is currently recognized as the most accurate,

efficient and universal approach. This methodology has been advanced in a number of recent papers, including those of Choi and Santos (1987), Tsay and Arora (1990), Liu and Der Kiureghian (1991a), Zhang and Der Kiureghian (1993), Kleiber *et al.* (1997), Conte *et al.* (1999) and Roth and Grigoriu (2001).

In this chapter, the fundamental finite element response equations are first reviewed. These expressions are subsequently differentiated to obtain top-level sensitivity equations. Particular attention is devoted to the evaluation of element integrals. Understanding of this topic is essential for subsequent derivations of response sensitivities with respect to nodal coordinates. In the initial sections of this chapter no assumption will be made regarding the parameter type with respect to which the differentiation is performed. The generic parameter, denoted h, could represent a material parameter, a cross-sectional geometry parameter, a nodal coordinate, or a nodal load. Furthermore, the general case of inelastic dynamic finite element analysis is considered. Specializations to obtain equations for elastic and/or static cases can be derived from this general case.

2.1.1 Review of Finite Element Response Equations

While the tensor notation may be visually more appealing, index notation is used here in order to avoid confusion regarding dimensions and the order of multiplications. The index notation also yields expressions which are closer to those implemented in the computer code. A comma is used to indicate a partial derivative. Furthermore, unless noted otherwise, Einstein's summation convention over repeated indices is enforced. For clarity, the indices are distinguished as follows: i, j, k, and l are used to denote spatial coordinate directions; the index n is used to denote quantities evaluated at time t_n; m is used to denote the iteration number; and p, q, r, and s are used as auxiliary indices for vector elements. The latter are used, for instance, with the nodal displacement vector, where the vector elements do not correspond to coordinate directions and can be placed in an arbitrary but consistent order.

The Boundary Value Problem. Following Zienkiewicz and Taylor (2000), the strong form of the boundary value problem reads:

$$\text{Balance of linear momentum} \quad : \quad \sigma_{ij,j} + \rho\, b_i = -\rho\, \ddot{\tilde{u}}_i \tag{2.1}$$

$$\text{Balance of angular momentum} \quad : \quad \sigma_{ij} = \sigma_{ji} \tag{2.2}$$

$$\text{Kinematics} \quad : \quad \epsilon_{ij} = \frac{1}{2}\left(\tilde{u}_{i,j} + \tilde{u}_{j,i} + \tilde{u}_{k,i}\tilde{u}_{k,j}\right) \tag{2.3}$$

$$\text{Constitutive law} \quad : \quad \sigma_{ij} = \sigma_{ij}\left(\epsilon_{ij},\, \dot{\epsilon}_{ij}\right) \tag{2.4}$$

$$\text{Surface tractions} \quad : \quad t_i = \sigma_{ij} n_j \quad \text{ on } \Gamma_t \tag{2.5}$$

$$\text{Prescribed displacements} \quad : \quad \tilde{u}_i = \tilde{u}_i^{pre} \quad \text{ on } \Gamma_u \tag{2.6}$$

In the above, σ_{ij} is the stress tensor, ρ is the mass density, b_i is the applied body force, typically arising from gravity, $\tilde{u}_i$ represents the displacement, where a tilde is used to distinguish the "exact" displacement field from the nodal displacements to be defined below, ϵ_{ij} is the strain tensor, $\dot{\epsilon}_{ij}$ is the strain rate tensor, t_i is the traction force acting on a surface with outward unit normal vector n_i, Γ_t is the surface on which traction forces are prescribed, and Γ_u is the surface where displacements $\tilde{u}_i^{pre}$ are prescribed.

The kinematic relation in Eq. (2.3) is stated in terms of the Green-Lagrange strain and has not been linearized. Thus, non-linear geometrical effects are also included. It is noted that, since the Green-Lagrange strain is employed, which is a "material tensor" existing in the original configuration, the conjugate stress is the second Piola-Kirchhoff stress tensor. This is implied above, even though σ and ϵ are kept as symbols. This Lagrangian formulation, referring quantities to a reference configuration, is common in finite element methods applied to solids (Zienkiewicz and Taylor 2000).

Virtual Work Formulation. The displacement form of the principle of virtual work states that a deformable body is in equilibrium if the sum of the external and internal works due to a virtual displacement field satisfying kinematic restrictions on Γ_u is zero. The virtual work form of the boundary value problem can be seen as a weighted, integrated form of the equation of balance of linear momentum:

$$\int_\Omega \sigma_{ij}\, \delta\epsilon_{ij}\, \mathrm{d}V = \int_{\Gamma_t} t_i\, \delta\tilde{u}_i\, \mathrm{d}A + \int_\Omega \rho\, b_i\, \delta\tilde{u}_i\, \mathrm{d}V - \int_\Omega \rho\, \ddot{\tilde{u}}_i\, \delta\tilde{u}_i\, \mathrm{d}V \tag{2.7}$$

In the above, the prescript δ indicates a virtual quantity. This weak form of the boundary value problem is equivalent to the strong form if the virtual displacement field is arbitrary, though satisfying kinematic restrictions on Γ_u.

Discretization by shape functions. In the displacement-based finite element method, the displacement field $\tilde{u}_i$ is discretized by the use of shape functions N multiplied by nodal displacements:

$$\tilde{u}_i = N_{ip}\, u_p \tag{2.8}$$

Here, i runs over the number of dimensions of the problem, while p runs over the number of degrees of freedom. Commonly, the same shape functions are used for the virtual and the real displacement fields.

Kinematic relations. The strain is determined from the displacement field according to the kinematic relation in Eq. (2.3). For the discretized displacement field, this relation is written as $\mathrm{d}\epsilon_p = \bar{B}_{pq}\ \mathrm{d}u_q$, where the strain components are collected in a vector with the off-diagonal terms in the strain tensor translated into engineering shear strains $\gamma_{ij} = 2\ \epsilon_{ij}$. The relation is written in an incremental form since, in general, the $\bar{B}$ matrix depends on the nodal displacement values. When nonlinear geometrical effects are neglected, the relation is simply written as $\epsilon_p = B^o_{pq}\ u_q$. In the geometrically nonlinear case it is possible to separate the "linear" and "non-linear" parts of the $\bar{B}$-matrix (Zienkiewicz and Taylor 2000). First, a relation between the strain ϵ_p and a vector θ_q containing the displacement derivatives $\tilde{u}_{i,j}$ is written:

$$\epsilon_p = \left(H_{pq} + \frac{1}{2} A_{pq} \right)\ \theta_q \tag{2.9}$$

The matrix H_{pq} contains only 0's and 1's, while the matrix A_{pq} contains the elements of $\tilde{u}_{i,j}$ and takes care of the second-order term in Eq. (2.3). By defining a matrix ∇_{pq} containing elements of 0 and $\frac{\partial}{\partial x_i}$, where x_i denotes coordinate direction i, we can write $\theta_q = \nabla_{pi}\ \tilde{u}_i$. Hence, the equation to relate strains to nodal displacements now reads:

$$\mathrm{d}\epsilon_p = \left(H_{pq} + \frac{1}{2} A_{pq} \right) \nabla_{qk}\ N_{kr}\ \mathrm{d}u_r = \left(B^o_{pr} + B^{nl}_{pr} \right) \mathrm{d}u_r = \bar{B}_{pr}\, \mathrm{d}u_r \tag{2.10}$$

where $B^o_{pr} = H_{pq}\ \nabla_{kq}\ N_{kr}$ is the linear term and $B^{nl}_{pr} = \frac{1}{2} A_{pq}\ \nabla_{kq}\ N_{kr}$ introduces the non-linear geometrical effects. The strain rate vector is obtained in the same way from the velocity vector:

$$\mathrm{d}\dot{\epsilon}_p = \hat{B}_{pr}\, \mathrm{d}\dot{u}_r \tag{2.11}$$

where $\hat{}$ is used to indicate that the velocities instead of the displacements enter in the geometrically nonlinear term of the "B-matrix."

Space-discretization of the virtual work formulation. The finite element discretization in Eq. (2.8) is applied to the virtual work equation in Eq. (2.7). The result is:

$$\int_\Omega \sigma_p \underbrace{\bar{B}_{pq} \delta u_q}_{\delta\epsilon_p}\ \mathrm{d}V = \int_{\Gamma_t} t_i\ N_{iq}\ \delta u_q\ \mathrm{d}A + \int_\Omega \rho\ b_i\ N_{iq}\ \delta u_q\ \mathrm{d}V - \int_\Omega \rho\ N_{ip}\ \ddot{u}_p\ N_{iq}\ \delta u_q\ \mathrm{d}V \tag{2.12}$$

where σ_p is the stress tensor in vector notation. Since the virtual displacement is assumed to be arbitrary while satisfying the kinematic constraints, δu_q is an arbitrary vector. Hence, Eq. (2.12) can be written:

$$\underbrace{\int_\Omega \sigma_p\ \bar{B}_{pq}\ \mathrm{d}V}_{P^{int}_q} = \underbrace{\int_{\Gamma_t} t_i\ N_{iq}\ \mathrm{d}A + \int_\Omega \rho\ b_i\ N_{iq}\ \mathrm{d}V}_{P^{ext}_q} - \underbrace{\int_\Omega \rho\ N_{ip} N_{iq}\ \mathrm{d}V}_{M_{qp}}\ \ddot{u}_p \tag{2.13}$$

where the internal force vector P_q^{int}, the external force vector P_q^{ext}, and the mass matrix M_{qp} have been defined. An artificial damping term, written as $C_{qp}\,\dot{u}_p$, where C_{qp} is a user-selected damping matrix, is usually also included so that the following equation of motion emerges:

$$M_{qp}\,\ddot{u}_p \;+\; C_{qp}\,\dot{u}_p \;+\; P_q^{int} \;=\; P_q^{ext} \tag{2.14}$$

As seen in Eq. (2.13), the internal forces P_q^{int} implicitly depend on the displacements through the stress vector σ_p, and also through $\bar{B}_{pq}$ if nonlinear geometry is considered. If the constitutive law is linear ($\sigma_p = D_{pq}\epsilon_q$), then it is easily derived that the internal force term is given by $P_q^{int} = \hat{K}_{qp}\,u_p$, where $\hat{K}_{qp} = \int_\Omega \bar{B}_{rq}\;D_{rs}\;\bar{B}_{sp}\;\mathrm{d}V$ is the stiffness matrix. If non-linear geometrical effects are included, it is common to distinguish between the linear part of the stiffness matrix and the displacement-dependent part, as shown below.

Time-discretization. In the space-discretized equation of motion in Eq. (2.14), it is assumed that the external load P_q^{ext} and the response quantities u_p, $\dot{u}_p$, and $\ddot{u}_p$, and hence P_q^{int}, vary with time. A time-discretization scheme is constructed by first writing the equation of motion at a specific time instant t_{n+1}:

$$M_{qp}\;\ddot{u}_{p\,(n+1)} \;+\; C_{qp}\;\dot{u}_{p\,(n+1)} \;+\; P^{int}_{q\,(n+1)} \;=\; P^{ext}_{q\,(n+1)} \tag{2.15}$$

Next, the nodal acceleration vector $\ddot{u}_{p\,(n+1)}$ and the nodal velocity vector $\dot{u}_{p\,(n+1)}$ are expressed in terms of $u_{p\,(n+1)}$, $u_{p\,(n)}$, $\dot{u}_{p\,(n)}$ and $\ddot{u}_{p\,(n)}$. The following general time-stepping scheme, which includes the well-known Newmark and Wilson schemes, is used:

$$\begin{aligned}
\ddot{u}_{p\,(n+1)} &= a_1\,u_{p\,(n+1)} + a_2\,u_{p\,(n)} + a_3\,\dot{u}_{p\,(n)} + a_4\,\ddot{u}_{p\,(n)} \\
\dot{u}_{p\,(n+1)} &= a_5\,u_{p\,(n+1)} + a_6\,u_{p\,(n)} + a_7\,\dot{u}_{p\,(n)} + a_8\,\ddot{u}_{p\,(n)}
\end{aligned} \tag{2.16}$$

The coefficients a_i are determined by selection of the time-stepping scheme.

Newton-Raphson solution scheme. Unless the constitutive law is linear and non-linear geometrical effects are neglected, the time- and space-discretized equation of motion in Eq. (2.15) is a non-linear equation. The unknown variable is the nodal displacement vector u_p at time t_{n+1}. A Newton-Raphson scheme to solve for this quantity is developed by considering the residual:

$$R_{q\,(n+1)} = M_{qp}\,\ddot{u}_{p\,(n+1)} + C_{qp\,(n+1)}\,\dot{u}_{p\,(n+1)} + P^{int}_{q\,(n+1)} - P^{ext}_{q\,(n+1)} = 0 \tag{2.17}$$

and linearizing it by performing a Taylor series expansion:

$$R^{(m+1)}_{q\,(n+1)} = R^{(m)}_{q\,(n+1)} + \underbrace{\frac{\partial R^{(m)}_{q\,(n+1)}}{\partial u_p}}_{\tilde{K}_{qp}} \left(u^{(m+1)}_{p\,(n+1)} - u^{(m)}_{p\,(n+1)}\right) + \ldots = 0 \tag{2.18}$$

An iterative scheme, with superscript m denoting step number, to solve for the displacement at time instant t_{n+1}, is:

$$u^{(m+1)}_{p\,(n+1)} = u^{(m)}_{p\,(n+1)} - \tilde{K}^{-1}_{qp}\; R^{(m)}_{q\,(n+1)} \tag{2.19}$$

Various schemes may be employed to update the tangent $\tilde{K}_{qp}$ at each iteration. The so-called Modified Newton-Raphson algorithm keeps the same tangent through all iterations at a time step. Convergence is reached when, e.g., the difference between the norms of $u^{(m+1)}_{p\,(n+1)}$ and $u^{(m)}_{p\,(n+1)}$ is less than a specified tolerance. The importance of updating the tangent before sensitivity computations is emphasized in later derivations.

The needed derivative of the residual with respect to the displacement vector can be found by differentiation of Eq. (2.17):

$$\begin{aligned}
\frac{\partial R^{(m)}_{q\,(n+1)}}{\partial u_p} &= M_{qp}\,\frac{\partial \ddot{u}_{p\,(n+1)}}{\partial u_{p\,(n+1)}} + C_{qp\,(n+1)}\,\frac{\partial \dot{u}_{p\,(n+1)}}{\partial u_{p\,(n+1)}} + \frac{\partial P^{int}_{q\,(n+1)}}{\partial u_{p\,(n+1)}} \\
&= M_{qp}\,a_1 \;+\; C_{qp\,(n+1)}\,a_5 \;+\; \frac{\partial P^{int}_{q\,(n+1)}}{\partial \dot{u}_{p\,(n+1)}}\frac{\partial \dot{u}_{p\,(n+1)}}{\partial u_{p\,(n+1)}} \;+\; \left.\frac{\partial P^{int}_{q\,(n+1)}}{\partial u_{p\,(n+1)}}\right|_{\dot{u}_{p\,(n+1)}\;fixed} \\
&= M_{qp}\,a_1 \;+\; C_{qp\,(n+1)}\,a_5 \;+\; C^{visc}_{qp\,(n+1)}\,a_5 \;+\; K_{qp\,(n+1)} \\
&= M_{qp}\,a_1 \;+\; \widehat{C}_{qp\,(n+1)}\,a_5 \;+\; K_{qp\,(n+1)} \\
&\stackrel{def}{=} \tilde{K}_{qp\,(n+1)}
\end{aligned} \tag{2.20}$$

A tilde is used to distinguish the dynamic tangent and $C^{visc}_{qp\,(n+1)}$ defines the damping tangent stemming from viscosity. It is assumed that the external forces do not depend on the displacements. This is a valid assumption for so-called conservative systems, which is the case considered here. This assumption would not be valid for structures with follower loads, e.g., surface pressure loads.

Stiffness matrix. The static tangent stiffness matrix K_{qp} may be computed with the definition of P^{int}_q from Eq. (2.13) in mind: $P^{int}_q = \int_\Omega \sigma_r\; \bar{B}_{rq}\; \mathrm{d}V$. When no geometrical non-linearities are included the tangent of the internal forces reads:

$$K^o_{qp} = \frac{\partial P^{int}_q}{\partial u_p} = \int_\Omega \frac{\partial \sigma_r}{\partial u_p}\; B^o_{rq}\; \mathrm{d}V = \int_\Omega \frac{\partial \sigma_r}{\partial \epsilon_s}\;\frac{\partial \epsilon_s}{\partial u_p}\; B^o_{rq}\; \mathrm{d}V = \int_\Omega \frac{\partial \sigma_r}{\partial \epsilon_s}\; B^o_{sp}\; B^o_{rq}\; \mathrm{d}V \tag{2.21}$$

It is seen that the global tangent is obtained by integration of the tangent of the stress-strain relationship at the material level. In the general nonlinear case the $\bar{B}$-matrix depends on the displacements so that additional contributions to the tangent appear (Zienkiewicz and Taylor 2000):

$$\begin{aligned}
K_{qp} = \frac{\partial P^{int}_q}{\partial u_p} &= \int_\Omega \frac{\partial \sigma_r}{\partial u_p}\; \bar{B}_{rq}\; \mathrm{d}V + \int_\Omega \sigma_r\; \frac{\partial \bar{B}_{rq}}{\partial u_p}\; \mathrm{d}V \\
&= \int_\Omega \bar{B}_{rq}\frac{\partial \sigma_r}{\partial \epsilon_s}\bar{B}_{sp}\; \mathrm{d}V + \int_\Omega \sigma_r\; \frac{\partial \bar{B}_{rq}}{\partial u_p}\; \mathrm{d}V
\end{aligned}$$

$$
\begin{aligned}
&= \int_\Omega \frac{\partial \sigma_r}{\partial \epsilon_s} \left(B^o_{sp} + B^{nl}_{sp}\right) \left(B^o_{rq} + B^{nl}_{rq}\right) \, \mathrm{d}V + \int_\Omega \sigma_r \, \frac{\partial \bar{B}_{rq}}{\partial u_p} \, \mathrm{d}V \\
&= \int_\Omega \frac{\partial \sigma_r}{\partial \epsilon_s} \, B^o_{sp} \, B^o_{rq} \, \mathrm{d}V \\
&+ \int_\Omega \frac{\partial \sigma_r}{\partial \epsilon_s} \left(B^o_{sp} \, B^{nl}_{rq} + B^{nl}_{sp} \, B^o_{rq} + B^{nl}_{sp} \, B^{nl}_{rq}\right) \, \mathrm{d}V \\
&+ \int_\Omega \sigma_r \, \frac{\partial \bar{B}_{rq}}{\partial u_p} \, \mathrm{d}V \\
&= K^o_{qp} + K^{nl}_{qp} + K^\sigma_{qp}
\end{aligned} \tag{2.22}
$$

where use has been made of Eq. (2.10). K^{nl}_{qp} and K^σ_{qp} are due to nonlinear geometrical effects, while K^o_{qp} is the contribution from material stiffness. Analogously, the tangent damping matrix due to material viscosity is obtained by use of Eq. (2.11):

$$
C^{visc}_{qp} = \frac{\partial P^{int}_q}{\partial \dot{u}_p} = \int_\Omega \bar{B}_{rq} \frac{\partial \sigma_r}{\partial \dot{\epsilon}_s} \hat{B}_{sp} \, \mathrm{d}V \tag{2.23}
$$

$\frac{\partial \sigma_r}{\partial \dot{\epsilon}_s} = \eta_{rs}$ is the the material damping tangent and the fact that $\frac{\partial \bar{B}_{rq}}{\partial \dot{u}_p} = 0$ has been used.

Element integration. A key issue in the finite element method is the evaluation of volume integrals to obtain the stiffness matrix, the mass matrix and the internal force vector. The domain of the boundary value problem is divided into finite elements connected at the nodes. Integrals are then calculated over each element and assembled:

$$
\int_\Omega (\cdot) \, \mathrm{d}V = \bigcup_{el} \int_{\Omega_{el}} (\cdot) \, \mathrm{d}V \tag{2.24}
$$

It is noted that the nodal coordinates enter the integration boundaries in Eq. (2.24). This must be kept in mind when response sensitivities with respect to the nodal coordinates are computed. Furthermore, the element integrals may be evaluated in several ways. For instance, a "direct stiffness" approach may be used for linear truss and beam-column elements. However, in this report a general approach is employed, with subsequent simplifications for special elements. The approach is necessitated by two facts, namely that the element integrals may be evaluated exactly only for a limited group of element types, and that the geometrical shape of the elements may vary throughout the domain even though the same type of element is used. The first issue is handled by using numerical integration, e.g., Gaussian quadrature. The second issue has motivated the use of the so-called isoparametric formulation. The element integrals are transformed into a standardized "parent domain." The coordinates of the parent domain are denoted ξ_i and the shape functions N_{ip} are defined over this domain: $N_{ip} = N_{ip}(\xi_i)$. The geometrical shape of the element is described similar to Eq. (2.8):

$$
x_i = N_{ip} \, \hat{x}_p \tag{2.25}
$$

where $\hat{x}_p$ is the vector of nodal coordinates and N_{ip} is commonly selected as the same shape functions as the ones used to interpolate the displacement field. This motivates the term isoparametric. As known from elementary calculus, the integrals are transformed into the parent domain through the following relation:

$$\int_{\Omega_{el}} f(x_i)\,\mathrm{d}x_i = \int_{\diamond} f(\xi_j(x_i))\,|J_{x_k,\xi_l}|\,\mathrm{d}\xi_j \tag{2.26}$$

where $\diamond$ denotes the boundaries of the parent domain. $|J_{x_k,\xi_l}|$ is the determinant of the Jacobian matrix containing all information about the geometry of a particular element. For the purpose of later sensitivity derivations, it is of interest to explain the factors entering the right-hand side of Eq. (2.26). The Jacobian matrix is computed as:

$$J_{x_i,\xi_j} = \frac{\partial x_i}{\partial \xi_j} = \frac{\partial N_{ip}}{\partial \xi_j}\,\hat{x}_p \tag{2.27}$$

The derivatives of the shape functions are easily found since they usually are simple functions of ξ_i. The integrand $f(\xi_i(x_i))$ in Eq. (2.26) contains terms from the $\bar{B}$-matrix in Eq. (2.10). These include the terms $\frac{\partial N_{ip}}{\partial x_j}$ and $\frac{\partial \tilde{u}_i}{\partial x_j}$, which are evaluated as follows:

$$\frac{\partial N_{ip}}{\partial x_j} = \frac{\partial N_{ip}}{\partial \xi_k}\,\frac{\partial \xi_k}{\partial x_j} \tag{2.28}$$

and:

$$\frac{\partial \tilde{u}_i}{\partial x_j} = \frac{\partial N_{ip}}{\partial x_j}\,u_p = \frac{\partial N_{ip}}{\partial \xi_k}\,\frac{\partial \xi_k}{\partial x_j}\,u_p \tag{2.29}$$

Here, $\frac{\partial N_{ip}}{\partial \xi_k}$ is again easily derived and $\frac{\partial \xi_k}{\partial x_j}$ are elements of the inverse Jacobian matrix.

The numerical integration for the parent domain integral in Eq. (2.26) is written:

$$\int_{\diamondsuit} f(\xi_i(x_i))\,|J_{x_k,\xi_l}|\,\mathrm{d}\xi_i \approx \sum_{m=1}^{numPoints} \omega_m\, f(\xi_i^{(m)})\,|J_{x_k,\xi_l}^{(m)}| \tag{2.30}$$

where ω_m are the integration weights and $\xi_i^{(m)}$ is the matrix containing the coordinates of the integration points.

2.1.2 Top-Level Response Sensitivity Equations

Sensitivity equations can be derived by returning to the space- and time-discretized equation of motion in Eq. (2.15) with the time-stepping scheme in Eq. (2.16) included, namely:

$$\begin{aligned} & M_{qp}\ \left(a_1\,u_{p\,(n+1)} + a_2\,u_{p\,(n)} + a_3\,\dot{u}_{p\,(n)} + a_4\,\ddot{u}_{p\,(n)}\right) \\ + \quad & C_{qp\,(n+1)}\ \left(a_5\,u_{p\,(n+1)} + a_6\,u_{p\,(n)} + a_7\,\dot{u}_{p\,(n)} + a_8\,\ddot{u}_{p\,(n)}\right) \\ + \quad & P^{int}_{q\,(n+1)}\ =\ P^{ext}_{q\,(n+1)} \end{aligned} \tag{2.31}$$

Differentiating throughout with respect to an arbitrary parameter h, leads to the following equation:

$$\begin{aligned}
& \frac{\partial M_{qp}}{\partial h} \left(a_1 \, u_{p\,(n+1)} + a_2 \, u_{p\,(n)} + a_3 \, \dot{u}_{p\,(n)} + a_4 \, \ddot{u}_{p\,(n)} \right) \\
+ \quad & M_{qp} \left(a_1 \frac{\partial u_{p\,(n+1)}}{\partial h} + a_2 \frac{\partial u_{p\,(n)}}{\partial h} + a_3 \frac{\partial \dot{u}_{p\,(n)}}{\partial h} + a_4 \frac{\partial \ddot{u}_{p\,(n)}}{\partial h} \right) \\
+ \quad & \frac{\partial C_{qp\,(n+1)}}{\partial h} \left(a_5 \, u_{p\,(n+1)} + a_6 \, u_{p\,(n)} + a_7 \, \dot{u}_{p\,(n)} + a_8 \, \ddot{u}_{p\,(n)} \right) \\
+ \quad & C_{qp\,(n+1)} \left(a_5 \frac{\partial u_{p\,(n+1)}}{\partial h} + a_6 \frac{\partial u_{p\,(n)}}{\partial h} + a_7 \frac{\partial \dot{u}_{p\,(n)}}{\partial h} + a_8 \frac{\partial \ddot{u}_{p\,(n)}}{\partial h} \right) \\
+ \quad & \frac{\partial P^{int}_{q\,(n+1)}}{\partial u_{p\,(n+1)}} \frac{\partial u_{p\,(n+1)}}{\partial h} \\
+ \quad & \frac{\partial P^{int}_{q\,(n+1)}}{\partial \dot{u}_{p\,(n+1)}} \frac{\partial \dot{u}_{p\,(n+1)}}{\partial h} \\
+ \quad & \frac{\partial P^{int}_{q\,(n+1)}}{\partial h} \Big|_{\substack{u_{p\,(n+1)}\, fixed \\ \dot{u}_{p\,(n+1)}\, fixed}} \\
= \quad & \frac{\partial P^{ext}_{q\,(n+1)}}{\partial h}
\end{aligned} \tag{2.32}$$

The implicit dependence of the internal force vector on the parameter h through the displacement and velocity vectors is taken into account, together with the explicit dependence on h. Furthermore, the term involving the derivative of the internal force vector with respect to the velocity vector at time t_{n+1} can be re-written as:

$$\frac{\partial P^{int}_{q\,(n+1)}}{\partial \dot{u}_{p\,(n+1)}} \frac{\dot{u}_{p\,(n+1)}}{\partial h} = C^{visc}_{qp\,(n+1)} \left(a_5 \frac{\partial u_{p\,(n+1)}}{\partial h} + a_6 \frac{\partial u_{p\,(n)}}{\partial h} + a_7 \frac{\partial \dot{u}_{p\,(n)}}{\partial h} + a_8 \frac{\partial \ddot{u}_{p\,(n)}}{\partial h} \right) \tag{2.33}$$

The damping tangent C^{visc}_{qp} stemming from material viscosity can be evaluated in a manner similar to the stiffness matrix in Eq. (2.22), and combined with the user-defined damping matrix C_{qp}. By denoting the combined damping matrix $\tilde{C}_{qp}$, using the notation $v_p = \frac{\partial u_p}{\partial h}$, and re-arranging to have the unknown displacement sensitivity $v_{p\,(n+1)}$ on the left-hand side, the following top-level sensitivity equation is obtained:

$$\begin{aligned}
\widetilde{K}_{qp\,(n+1)} \; v_{p\,(n+1)} \quad = \quad & \frac{\partial P^{ext}_{q\,(n+1)}}{\partial h} - \frac{\partial P^{int}_{q\,(n+1)}}{\partial h} \Big|_{\substack{u_{p\,(n+1)}\, fixed \\ \dot{u}_{p\,(n+1)}\, fixed}} \\
- \quad & \frac{\partial M_{qp}}{\partial h} \, \ddot{u}_{p\,(n+1)} - \frac{\partial C_{qp\,(n+1)}}{\partial h} \, \dot{u}_{p\,(n+1)} \\
- \quad & M_{qp} \left(a_2 \, v_{p\,(n)} + a_3 \, \dot{v}_{p\,(n)} + a_4 \, \ddot{v}_{p\,(n)} \right) \\
- \quad & \tilde{C}_{qp\,(n+1)} \left(a_6 \, v_{p\,(n)} + a_7 \, \dot{v}_{p\,(n)} + a_8 \, \ddot{v}_{p\,(n)} \right)
\end{aligned} \tag{2.34}$$

This result agrees with previous results obtained by Zhang and Der Kiureghian (1993), Kleiber *et al.* (1997), Roth and Grigoriu (2001), and others. The tangent $\widetilde{K}_{qp\,(n+1)}$ of this equation corresponds to

the one in Eq. (2.20):

$$\widetilde{K}_{qp\,(n+1)} = a_1\, M_{qp} + a_5\, \tilde{C}_{qp\,(n+1)} + \overbrace{\frac{\partial P^{int}_{q\,(n+1)}}{\partial u_{p\,(n+1)}}}^{K_{qp\,(n+1)}} \tag{2.35}$$

It is noted in passing that the simplified sensitivity equation obtained by neglecting dynamic forces reads:

$$K_{qp}\, v_p = \frac{\partial P^{ext}_q}{\partial h} - \frac{\partial P^{int}_q}{\partial h}\bigg|_{u_p\, fixed} \tag{2.36}$$

Under certain assumptions to be described in Section 2.1.4, Eq. (2.34) is a linear equation in the displacement sensitivities $v_{p\,(n+1)}$. Once $v_{p\,(n+1)}$ is determined, the time-stepping scheme in Eq. (2.16) may be used to obtain $\dot{v}_{p\,(n+1)}$ and $\ddot{v}_{p\,(n+1)}$.

The linearity of Eq. (2.34) is a cornerstone of the efficiency and accuracy of the DDM. An equally important fact is that the tangent of this equation is the updated tangent of the Newton-Raphson scheme already used to solve for the displacement response itself. Hence, only a new right-hand side needs to be assembled to solve for the displacement sensitivity vector. It is emphasized that if a Modified Newton-Raphson scheme is used to solve for the displacement vector, then it is critical that the tangent be updated before the sensitivity computations are performed.

In the above derivations, no assumptions were made to exclude nonlinearities due to geometrical effects or nonlinear constitutive laws. Neither have assumptions been made regarding the nature of the parameter h with respect to which the differentiation is performed. Thus, h could represent a material parameter, a cross-sectional geometry parameter, a nodal coordinate, or a load variable. The following sections establish specific equations depending on the nature of the parameter h. It is noted that the last two terms in the right-hand side of Eq. (2.34) must always be computed in the general dynamic case, regardless of the nature of h. Subsequent sections will specify equations for the computation of the remaining terms $\frac{\partial P^{ext}_q}{\partial h}$, $\frac{\partial P^{int}_q}{\partial h}$, $\frac{\partial M_{qp}}{\partial h}$ and $\frac{\partial C_{qp}}{\partial h}$ for particular parameter types. In addition, important implementation aspects related to history variables and assembly procedures for inelastic problems are discussed.

2.1.3 Displacement Sensitivity with Respect to Material Parameters

Material parameters enter the equation of motion, Eq. (2.14), as mass density in the mass matrix or as constitutive parameters in the internal force vector. Additionally, the user may prescribe the artificial damping matrix C_{qp} in terms of the stiffness matrix and/or the mass matrix. This is discussed in

further detail in the following section. In effect, material parameters enter the right-hand side of Eq. (2.34) through $\frac{\partial P_q^{int}}{\partial h}\Big|_{\substack{u_p\ fixed\\ \dot{u}_p\ fixed}}$, $\frac{\partial K_{qp}}{\partial h}$ and $\frac{\partial M_{qp}}{\partial h}$. These quantities are derived in this section.

From Eq. (2.13) the internal force vector at time t_{n+1} is given by:

$$P^{int}_{q\ (n+1)} = \int_\Omega \sigma_{p\ (n+1)}\ \bar{B}_{pq\ (n+1)}\ \mathrm{d}V \tag{2.37}$$

This equation is differentiated with respect to h as a first step towards obtaining the conditional derivative entering the right-hand side of Eq. (2.34). Again, the explicit dependence on h is taken into account as well as the implicit dependence through the displacement and velocity vectors. Omitting the subscript $(n+1)$ on all quantities for notational clarity, differentiation of Eq. (2.37) yields:

$$\frac{\partial P_q^{int}}{\partial u_p}\frac{\partial u_p}{\partial h} + \frac{\partial P_q^{int}}{\partial \dot{u}_p}\frac{\partial \dot{u}_p}{\partial h} + \frac{\partial P_q^{int}}{\partial h}\bigg|_{\substack{u_p\ fixed\\ \dot{u}_p\ fixed}} = \int_\Omega \left[\frac{\partial \sigma_p}{\partial h}\bar{B}_{pq} + \sigma_p\frac{\partial \bar{B}_{pq}}{\partial h}\right]\mathrm{d}V \tag{2.38}$$

In Eq. (2.38) we recognize the global stiffness matrix $\frac{\partial P_q^{int}}{\partial u_p} = K_{qp}$ defined in Eq. (2.22) and the viscous damping matrix $\frac{\partial P_q^{int}}{\partial \dot{u}_p} = C^{visc}_{qp}$. Substituting these relations into Eq. (2.38) and applying the chain rule of differentiation to the terms $\frac{\partial \sigma_p}{\partial h}$ and $\frac{\partial \bar{B}_{pq}}{\partial h}$ leads to:

$$\begin{aligned} & K_{qp}\frac{\partial u_p}{\partial h} + C^{visc}_{qp}\frac{\partial \dot{u}_p}{\partial h} + \frac{\partial P_q^{int}}{\partial h}\bigg|_{\substack{u_p\ fixed\\ \dot{u}_p\ fixed}} \\ = & \int_\Omega \Bigg[\left(\frac{\partial \sigma_p}{\partial \epsilon_r}\frac{\partial \epsilon_r}{\partial h} + \frac{\partial \sigma_p}{\partial \dot{\epsilon}_r}\frac{\partial \dot{\epsilon}_r}{\partial h} + \frac{\partial \sigma_p}{\partial h}\bigg|_{\substack{\epsilon_r\ fixed\\ \dot{\epsilon}_r\ fixed}}\right)\bar{B}_{pq} \\ + & \sigma_p\left(\frac{\partial \bar{B}_{pq}}{\partial u_r}\frac{\partial u_r}{\partial h} + \frac{\partial \bar{B}_{pq}}{\partial h}\bigg|_{\substack{u_r\ fixed\\ \dot{u}_r\ fixed}}\right)\Bigg]\mathrm{d}V \end{aligned} \tag{2.39}$$

The chain rule of differentiation is applied to the strain derivatives:

$$\frac{\partial \epsilon_r}{\partial h} = \frac{\partial \epsilon_r}{\partial u_s}\frac{\partial u_s}{\partial h} + \frac{\partial \epsilon_r}{\partial h}\bigg|_{u_s\ fixed} \tag{2.40}$$

$$\frac{\partial \dot{\epsilon}_r}{\partial h} = \frac{\partial \dot{\epsilon}_r}{\partial \dot{u}_s}\frac{\partial \dot{u}_s}{\partial h} + \frac{\partial \dot{\epsilon}_r}{\partial h}\bigg|_{\dot{u}_s\ fixed} \tag{2.41}$$

By introducing the kinematic relations in Eqs. (2.10) and (2.11) and recognizing the material tangent stiffness $k_{pr} = \frac{\partial \sigma_p}{\partial \epsilon_r}$ and the material tangent viscosity $\eta_{pr} = \frac{\partial \sigma_p}{\partial \dot{\epsilon}_r}$, Eq. (2.39) is rearranged to become:

$$\begin{aligned} & K_{qp}\ \frac{\partial u_p}{\partial h} + C^{visc}_{qp}\frac{\partial \dot{u}_p}{\partial h} + \frac{\partial P_q^{int}}{\partial h}\bigg|_{\substack{u_p\ fixed\\ \dot{u}_p\ fixed}} = \int_\Omega \Bigg[\bar{B}_{pq}k_{pr}\bar{B}_{rs}\frac{\partial u_s}{\partial h} \\ & + \bar{B}_{pq}\eta_{pr}\hat{B}_{rs}\frac{\partial \dot{u}_s}{\partial h} + \ \bar{B}_{pq}\eta_{pr}\frac{\partial \dot{\epsilon}_r}{\partial h}\bigg|_{\dot{u}_s\ fixed} + \bar{B}_{pq}k_{pr}\frac{\partial \epsilon_r}{\partial h}\bigg|_{u_s\ fixed} \\ & + \bar{B}_{pq}\frac{\partial \sigma_p}{\partial h}\bigg|_{\substack{\epsilon_r\ fixed\\ \dot{\epsilon}_r\ fixed}} + \ \sigma_p\frac{\partial \bar{B}_{pq}}{\partial u_r}\frac{\partial u_r}{\partial h} + \sigma_p\ \frac{\partial \bar{B}_{pq}}{\partial h}\bigg|_{u_r\ fixed}\Bigg]\mathrm{d}V \end{aligned} \tag{2.42}$$

By use of Eq. (2.22), the term $K_{qp}\,\frac{\partial u_p}{\partial h}$ on the left-hand side cancels with the terms $\int_\Omega \sigma_p \frac{\partial \bar{B}_{pq}}{\partial u_r}\frac{\partial u_r}{\partial h}\,\mathrm{d}V$ and $\int_\Omega \bar{B}_{pq}k_{pr}\bar{B}_{rs}\frac{\partial u_s}{\partial h}\,\mathrm{d}V$ on the right-hand side. Similarly, using Eq. (2.23), the term $C^{visc}_{qp}\,\frac{\partial \dot{u}_p}{\partial h}$ on the left-hand side cancels with the term $\int_\Omega \bar{B}_{pq}\eta_{pr}\hat{B}_{rs}\frac{\partial \dot{u}_s}{\partial h}\,\mathrm{d}V$ on the right-hand side. Hence, Eq. (2.42) simplifies to:

$$\begin{aligned}\left.\frac{\partial P^{int}_q}{\partial h}\right|_{\substack{u_p\ fixed\\ \dot{u}_p\ fixed}} &= \int_\Omega \left[\bar{B}_{pq}\left.\frac{\partial \sigma_p}{\partial h}\right|_{\substack{\epsilon_r\ fixed\\ \dot{\epsilon}_r\ fixed}} + \sigma_p\left.\frac{\partial \bar{B}_{pq}}{\partial h}\right|_{u_r\ fixed}\right.\\ &+ \bar{B}_{pq}\eta_{pr}\left.\frac{\partial \dot{\epsilon}_r}{\partial h}\right|_{\dot{u}_s\ fixed} + \bar{B}_{pq}k_{pr}\left.\frac{\partial \epsilon_r}{\partial h}\right|_{u_s\ fixed}\Bigg]\mathrm{d}V\end{aligned} \tag{2.43}$$

Furthermore, since material parameters do not enter the kinematic relationships, we have:

$$\left.\frac{\partial \bar{B}_{pq}}{\partial h}\right|_{u_r\ fixed} = \left.\frac{\partial \dot{\epsilon}_r}{\partial h}\right|_{\dot{u}_s\ fixed} = \left.\frac{\partial \epsilon_r}{\partial h}\right|_{u_s\ fixed} = 0 \tag{2.44}$$

We are left with the following expression for the desired conditional derivative of the internal force vector:

$$\left.\frac{\partial P^{int}_q}{\partial h}\right|_{\substack{u_p\ fixed\\ \dot{u}_p\ fixed}} = \int_\Omega \bar{B}_{pq}\left.\frac{\partial \sigma_p}{\partial h}\right|_{\substack{\epsilon_r\ fixed\\ \dot{\epsilon}_r\ fixed}}\mathrm{d}V \tag{2.45}$$

This equation agrees with a result obtained earlier by Liu and Der Kiureghian (1991a). However, the simplification in Eq. (2.44) is not valid when h represents a nodal coordinate. This is discussed in Section 2.1.5.

Turning to the sensitivity of the stiffness matrix K_{qp}, the derivative without the assumption of fixed displacements is to be computed. For reasons discussed in the following section, it is assumed that only the initial linear stiffness, namely K^o_{qp} in Eq. (2.22), enters the right-hand side of the top-level sensitivity equation. Then, the derivative of the stiffness matrix is found by differentiation of the initial material stiffness k^o_{rs}:

$$\frac{\partial K^o_{qp}}{\partial h} = \int_\Omega B^o_{rq}\,\frac{\partial k^o_{rs}}{\partial h}\,B^o_{sp}\,\mathrm{d}V \tag{2.46}$$

The mass matrix is composed of lumped nodal masses and/or contributions from the integrals of $\rho N_{iq}N_{ip}$ over element volumes; see Eq. (2.13). In the former case, $\frac{\partial M_{qp}}{\partial h}$ is a matrix containing 1 at the degree of freedom corresponding to the lumped mass and 0 elsewhere. In the latter case, the following expression is obtained for the derivative of the mass matrix:

$$\frac{\partial M_{qp}}{\partial h} = \int_\Omega \frac{\partial \rho}{\partial h}\,N_{ip}N_{iq}\,\mathrm{d}V \tag{2.47}$$

where $\frac{\partial \rho}{\partial h} = 1$ if h represents the material density.

2.1.4 The Possibility of Non-Linear Sensitivity Equations

As mentioned in the previous section, the user may select an artificial damping model which involves the mass matrix and the stiffness matrix. This is a common approach known as Rayleigh damping. The concept is described in this section, followed by a discussion of the implications of various choices for the sensitivity computations.

In OpenSees, Rayleigh damping is specified as follows (McKenna *et al.* 2002):

$$C_{qp} = \alpha_M \; M_{qp} + \beta_{K_1} \; K_{qp}^{current} + \beta_{K_2} \; K_{qp}^{initial} + \beta_{K_3} \; K_{qp}^{last\ committed} \tag{2.48}$$

where $K_{qp}^{current}$ is the tangent stiffness matrix denoted $K_{qp\,(n+1)}$ in this study, $K_{qp}^{initial}$ is the linear stiffness matrix denoted K_{qp}^{o} in this study, and $K_{qp}^{last\ committed}$ is the tangent stiffness matrix at the previously committed step, namely $K_{qp\,(n)}$. Using $K_{qp}^{current}$ implies that the damping matrix is updated during the iterations to equilibrium at each step. The coefficients α_M, β_{K_1}, β_{K_2} and β_{K_3} are selected by the analyst.

The focus in this section is to illuminate the consequences for sensitivity analysis of selecting $\beta_{K_1} \neq 0$ or $\beta_{K_3} \neq 0$. The discussion is motivated by the term $\frac{\partial C_{qp}}{\partial h}$ appearing in the right-hand side of Eq. (2.34). If the user selects $\beta_{K_1} \neq 0$ or $\beta_{K_3} \neq 0$ then the derivatives $\frac{\partial K_{qp\,(n+1)}}{\partial h}$ or $\frac{\partial K_{qp\,(n)}}{\partial h}$ will appear in the right-hand side of Eq. (2.34). According to Eq. (2.22), these derivatives involve differentiation of the integral $\int_\Omega \bar{B}_{rq} \; k_{rs} \; \bar{B}_{sp} \; \mathrm{d}V$ with respect to h without the assumption of fixed displacement. This poses a problem for sensitivity analysis due to the general dependence of both k_{rs} and $\bar{B}_{sp}$ on the displacement where material nonlinearity or nonlinear geometrical effects are present. In such cases, the displacement sensitivity $\frac{\partial u_p}{\partial h}$ will appear in the right-hand side of Eq. (2.34) and yield a nonlinear equation for the unknown sensitivity.

In the situation described here, namely when the user selects $\beta_{K_1} \neq 0$ or $\beta_{K_3} \neq 0$ for problems with nonlinear material behavior or nonlinear geometrical effects, Eq. (2.34) must be solved by an iterative scheme. This is feasible, but it compromises the attractive efficiency and accuracy properties of the DDM. Therefore, in this study, the choices $\beta_{K_1} = 0$ and $\beta_{K_3} = 0$ are recommended and an iterative solution of Eq. (2.34) is not implemented. In OpenSees, an error message is given if β_{K_1} or β_{K_3} are selected different from zero in conjunction with DDM sensitivity analysis.

One may argue that $\beta_{K_1} = 0$ and $\beta_{K_3} = 0$ are reasonable choices for Rayleigh damping from a physical standpoint. There is no evidence that the damping matrix should vary in proportion to the current stiffness in nonlinear structures. Furthermore, the theoretical attractiveness of Rayleigh

damping lies in the decoupled nature of the system of equations that are derived. This advantage is no longer relevant in non-linear analysis.

2.1.5 Displacement Sensitivity with Respect to Nodal Coordinates

Obtaining response sensitivities with respect to nodal coordinates is often referred to as shape sensitivity analysis. The nodal coordinates affect all element integrals, since they enter into the integral boundaries in addition to the kinematic relations. For this reason the terms $\frac{\partial P_q^{ext}}{\partial h}$, $\left.\frac{\partial P_q^{int}}{\partial h}\right|_{\substack{u_p\ fixed \\ \dot{u}_p\ fixed}}$, $\frac{\partial M_{qp}}{\partial h}$ and $\frac{\partial C_{qp}}{\partial h}$ must all be given attention in this section. As discussed in the previous section, it is assumed that the damping matrix is composed of the initial stiffness matrix K_{qp}^o and the mass matrix according to Eq. (2.48). Hence, it is the derivatives of the stiffness matrix, the mass matrix and the external and internal force vector that are of interest in this section.

Unlike the differentiation with respect to a material parameter, the differentiation with respect to a nodal coordinate depends on the method used to evaluate the element integrals. The general isoparametric formulation combined with Gauss quadrature is the method assumed in this section. Other integration schemes for special elements are dealt with in subsequent sections. According to Eqs. (2.26) and (2.30) the general scheme reads:

$$\int_{\Omega_{el}} f(x_i)\,\mathrm{d}x_i \approx \sum_{m=1}^{numPoints} \omega_m\ f(\xi_i^{(m)})\ |J_{x_k,\xi_l}^{(m)}| \tag{2.49}$$

Hence, the desired derivative is obtained by the product rule of differentiation:

$$\frac{\partial}{\partial h}\left(\int_{\Omega_{el}} f(x_i)\,\mathrm{d}x_i\right) = \sum_{m=1}^{numPoints} \omega_m \left(\frac{\partial f(\xi_i^{(m)})}{\partial h}\ |J_{x_k,\xi_l}^{(m)}| + f(\xi_i^{(m)})\ \frac{\partial |J_{x_k,\xi_l}^{(m)}|}{\partial h}\right) \tag{2.50}$$

Since the determinant of the Jacobian contains all the needed information about the geometry of the element, it is $\frac{\partial |J_{x_k,\xi_l}|}{\partial h}$ that takes care of the variation in integration boundaries. As a first step to obtain this derivative we find by the chain rule of differentiation:

$$\frac{\partial |J_{x_i,\xi_j}|}{\partial h} = \frac{\partial |J_{x_i,\xi_j}|}{\partial J_{x_k,\xi_l}}\ \frac{\partial J_{x_k,\xi_l}}{\partial h} \tag{2.51}$$

From elementary tensor calculus (Gurtin 1981), the derivative of the determinant of a matrix with respect to the matrix itself reads:

$$\frac{\partial |J_{x_i,\xi_j}|}{\partial J_{x_k,\xi_l}} = |J_{x_i,\xi_j}|\ \left(J_{x_k,\xi_l}\right)^{-T} \tag{2.52}$$

where the superscript $-T$ indicates the inverse transpose. Next, an expression for $\frac{\partial J_{x_k,\xi_l}}{\partial h}$ is obtained. Using Eq. (2.27), keeping in mind that h now represents a nodal coordinate, the following expression is obtained:

$$\frac{\partial J_{x_k,\xi_l}}{\partial h} = \frac{\partial}{\partial h}\left(\frac{\partial x_k}{\partial \xi_l}\right) = \frac{\partial N_{k\tilde{s}}}{\partial \xi_l} \tag{2.53}$$

where subscript $\tilde{s}$ denotes the position held by h in the vector of nodal coordinates. In conclusion, the expression for the derivative of the determinant of the Jacobian matrix is:

$$\frac{\partial |J_{x_i,\xi_j}|}{\partial h} = |J_{x_i,\xi_j}| \ \left(J_{x_k,\xi_l}\right)^{-T} \ \frac{\partial N_{k\tilde{s}}}{\partial \xi_l} \tag{2.54}$$

Attention is now focused on the integrand $f(\xi_i^{(m)})$ of the constituents of the right-hand side of Eq. (2.34). We start with the internal force vector, where the derivative with respect to the fixed displacement and velocity vectors is needed. However, as in Section 2.1.3, it is recognized that the proper approach is to first differentiate the expression for P_q^{int}, namely:

$$P_q^{int} \approx \sum_{m=1}^{numPoints} \omega_m \ \sigma_p \ \bar{B}_{pq} \ |J_{x_k,\xi_l}^{(m)}| \tag{2.55}$$

where it is implicitly assumed that the quantities in the right-hand side are evaluated at the integration points. By differentiation, taking into account the explicit dependence of P_q^{int} on h as well as the implicit dependence through the displacement and velocity responses, we obtain:

$$\begin{aligned}\frac{\partial P_q^{int}}{\partial u_p}\frac{\partial u_p}{\partial h} + \frac{\partial P_q^{int}}{\partial \dot{u}_p}\frac{\partial \dot{u}_p}{\partial h} + \left.\frac{\partial P_q^{int}}{\partial h}\right|_{\substack{up\ fixed\\ \dot{u}p\ fixed}} \approx \sum_{m}^{numPoints} \omega_m \left(\frac{\partial \sigma_p}{\partial h}\,\bar{B}_{pq}\,|J_{x_k,\xi_l}^{(m)}|\right.\\ \left. + \sigma_p \frac{\partial \bar{B}_{pq}}{\partial h}\,|J_{x_k,\xi_l}^{(m)}| + \sigma_p\,\bar{B}_{pq}\,|J_{x_i,\xi_j}^{(m)}|\ \left(J_{x_k,\xi_l}^{(m)}\right)^{-T}\ \frac{\partial N_{k\tilde{s}}}{\partial \xi_l}\right)\end{aligned} \tag{2.56}$$

where use has been made of Eq. (2.54). The chain rule of differentiation is now used on the terms $\frac{\partial \sigma_p}{\partial h}$ and $\frac{\partial \bar{B}_{pq}}{\partial h}$ to differentiate through the constitutive law and the kinematic relation:

$$\frac{\partial \sigma_p}{\partial h} = \frac{\partial \sigma_p}{\partial \epsilon_r}\frac{\partial \epsilon_r}{\partial h} + \frac{\partial \sigma_p}{\partial \dot{\epsilon}_r}\frac{\partial \dot{\epsilon}_r}{\partial h} + \left.\frac{\partial \sigma_p}{\partial h}\right|_{\substack{\epsilon_r\ fixed\\ \dot{\epsilon}_r\ fixed}} \tag{2.57}$$

$$\frac{\partial \bar{B}_{pq}}{\partial h} = \frac{\partial \bar{B}_{pq}}{\partial u_r}\frac{\partial u_r}{\partial h} + \left.\frac{\partial \bar{B}_{pq}}{\partial h}\right|_{u_r\ fixed} \tag{2.58}$$

The chain rule of differentiation is applied to the strain derivatives by substituting Eqs. (2.40) and (2.41) into Eq. (2.57). Further, substituting Eqs. (2.57) and (2.58) into Eq. (2.56) we obtain:

$$\begin{aligned}K_{qp}\frac{\partial u_p}{\partial h} + C_{qp}^{visc}\frac{\partial \dot{u}_p}{\partial h} + \left.\frac{\partial P_q^{int}}{\partial h}\right|_{\substack{up\ fixed\\ \dot{u}p\ fixed}} \approx \sum_{m}^{numPoints} \omega_m\ |J_{x_k,\xi_l}^{(m)}|\Bigg(\bar{B}_{pq}k_{pr}\bar{B}_{rs}\frac{\partial u_s}{\partial h}\\ + \bar{B}_{pq}k_{pr}\left.\frac{\partial \epsilon_r}{\partial h}\right|_{u_s\ fixed} + \bar{B}_{pq}\eta_{pr}\hat{B}_{rs}\frac{\partial \dot{u}_s}{\partial h} + \left.\bar{B}_{pq}\eta_{pr}\frac{\partial \dot{\epsilon}_r}{\partial h}\right|_{\dot{u}_s\ fixed}\end{aligned}$$

$$+ \bar{B}_{pq} \left.\frac{\partial \sigma_p}{\partial h}\right|_{\substack{\epsilon_r \, fixed \\ \dot{\epsilon}_r \, fixed}} + \sigma_p \frac{\partial \bar{B}_{pq}}{\partial u_r}\frac{\partial u_r}{\partial h} + \left.\sigma_p \frac{\partial \bar{B}_{pq}}{\partial h}\right|_{u_r \, fixed}$$
$$+ \sigma_p \, \bar{B}_{pq} \left(J^{(m)}_{x_i,\xi_j}\right)^{-T} \frac{\partial N_{i\tilde{s}}}{\partial \xi_j}\Bigg) \tag{2.59}$$

where the definitions $k_{pr} = \frac{\partial \sigma_p}{\partial \epsilon_r}$, $\eta_{pr} = \frac{\partial \sigma_p}{\partial \dot{\epsilon}_r}$, $\bar{B}_{rs} = \frac{\partial \epsilon_r}{\partial u_s}$ and $\hat{B}_{rs} = \frac{\partial \dot{\epsilon}_r}{\partial \dot{u}_s}$ have been used. The terms involving $\bar{B}_{pq} k_{pr} \bar{B}_{rs} \frac{\partial u_s}{\partial h}$ and $\sigma_p \frac{\partial \bar{B}_{pq}}{\partial u_r}\frac{\partial u_r}{\partial h}$ on the right-hand side cancel with the term $K_{qp}\frac{\partial u_p}{\partial h}$ on the left-hand side according to Eq. (2.22). Similarly, the term involving $\bar{B}_{pq}\eta_{pr}\hat{B}_{rs}\frac{\partial \dot{u}_s}{\partial h}$ on the right-hand side cancels with the term $C^{visc}_{qp}\frac{\partial \dot{u}_p}{\partial h}$ on the left-hand side according to Eq. (2.23). The conditional derivative of the internal force vector is then obtained as:

$$\left.\frac{\partial P^{int}_q}{\partial h}\right|_{\substack{u_p \, fixed \\ \dot{u}_p \, fixed}} \approx \sum_{m}^{numPoints} \omega_m \, |J^{(m)}_{x_k,\xi_l}| \Bigg(\left.\bar{B}_{pq} k_{pr} \frac{\partial \epsilon_r}{\partial h}\right|_{u_s \, fixed} + \left.\bar{B}_{pq}\eta_{pr}\frac{\partial \dot{\epsilon}_r}{\partial h}\right|_{\dot{u}_s \, fixed}$$
$$+ \bar{B}_{pq} \left.\frac{\partial \sigma_p}{\partial h}\right|_{\substack{\epsilon_r \, fixed \\ \dot{\epsilon}_r \, fixed}} + \left.\sigma_p \frac{\partial \bar{B}_{pq}}{\partial h}\right|_{u_r \, fixed} + \sigma_p \, \bar{B}_{pq} \left(J^{(m)}_{x_i,\xi_j}\right)^{-T} \frac{\partial N_{i\tilde{s}}}{\partial \xi_j}\Bigg) \tag{2.60}$$

This result is consistent with, but more general than the result previously obtained by Liu and Der Kiureghian (1991a). The terms $\left.\bar{B}_{pq} k_{pr} \frac{\partial \epsilon_r}{\partial h}\right|_{u_s \, fixed}$, $\left.\bar{B}_{pq}\eta_{pr}\frac{\partial \dot{\epsilon}_r}{\partial h}\right|_{\dot{u}_s \, fixed}$ and $\left.\sigma_p \frac{\partial \bar{B}_{pq}}{\partial h}\right|_{u_r \, fixed}$ are new and account for differentiation of kinematic relations when the parameter h enters them. This is the case when h represents a nodal coordinate.

The three new terms involve differentiation of the strain-displacement relationship. Hence, the remaining task is to differentiate the strain-displacement matrix $\bar{B}_{pq}$ and the strain rate-velocity matrix $\hat{B}_{pq}$. This involves obtaining derivatives of the quantities in Eqs. (2.28) and (2.29) with respect to a nodal coordinate for fixed displacement and velocity. The results read:

$$\frac{\partial}{\partial h}\frac{\partial N_{ip}}{\partial x_j} = \frac{\partial N_{ip}}{\partial \xi_k}\frac{\partial}{\partial h}\frac{\partial \xi_k}{\partial x_j} \tag{2.61}$$

$$\frac{\partial}{\partial h}\frac{\partial \tilde{u}_i}{\partial x_j} = \frac{\partial N_{ip}}{\partial \xi_k}\frac{\partial}{\partial h}\frac{\partial \xi_k}{\partial x_j} u_p \tag{2.62}$$

The counterpart of Eq. (2.62) for the $\hat{B}_{qp}$ matrix is identical but with the displacement replaced by velocity. It is observed that the derivative of the inverse Jacobian is needed, namely $\frac{\partial}{\partial h}\frac{\partial \xi_k}{\partial x_j}$. In general, the derivative of an inverse matrix $[T_{ij}]^{-1}$ can be derived by starting with the property $T_{ij}\,[T_{jk}]^{-1} = \delta_{ik}$ where δ_{ik} is the Kronecker-delta. The derivative becomes $\frac{\partial}{\partial h}\left(T_{ij}\,[T_{jk}]^{-1}\right) = \frac{\partial T_{ij}}{\partial h}\,[T_{jk}]^{-1} + T_{il}\frac{\partial [T_{lk}]^{-1}}{\partial h} = 0$, from which the desired result may be extracted: $\frac{\partial [T_{lk}]^{-1}}{\partial h} = -[T_{li}]^{-1}\frac{\partial T_{ij}}{\partial h}[T_{jk}]^{-1}$. The needed derivative of the inverse Jacobian matrix may thus be computed as:

$$\frac{\partial}{\partial h}\left[\frac{\partial x_j}{\partial \xi_k}\right]^{-1} = -\left[\frac{\partial x_i}{\partial \xi_j}\right]^{-T}\frac{\partial}{\partial h}\frac{\partial x_i}{\partial \xi_l}\left[\frac{\partial x_l}{\partial \xi_k}\right]^{-1} \tag{2.63}$$

The derivatives of the elements of the Jacobian matrix are given in Eq. (2.53).

The derivation of the conditional derivative of the internal force vector with respect to a nodal coordinate is thereby complete. Eq. (2.60) has been implemented and verified in OpenSees, including nonlinear geometrical effects but excluding viscous effects ($\eta_{pr} = 0$). It is noted that in the absence of nonlinear geometrical effects, Eq. (2.60) becomes:

$$\frac{\partial P_q^{int}}{\partial h}\bigg|_{\substack{u_p\ fixed\\ \dot{u}_p\ fixed}} \approx \sum_{m=1}^{numPoints} \omega_m\ |J_{x,\xi}^{(m)}| \Bigg(B_{pq}^o k_{pr} \frac{\partial B_{rs}^o}{\partial h} u_s + B_{pq}^o \eta_{pr} \frac{\partial \hat{B}_{rs}^o}{\partial h} \dot{u}_s + B_{pq}^o \left.\frac{\partial \sigma_p}{\partial h}\right|_{\substack{\epsilon_r\ fixed\\ \dot{\epsilon}_r\ fixed}} + \sigma_p \frac{\partial B_{pq}^o}{\partial h} + \sigma_p\, B_{pq}^o \left(J_{x_i,\xi_j}^{(m)}\right)^{-T} \frac{\partial N_{i\tilde{s}}}{\partial \xi_j} \Bigg) \tag{2.64}$$

For the mass matrix the integrand reads $\rho\ N_{iq} N_{ir}$. The fact that the nodal coordinates do not enter implies that only the determinant of the Jacobian needs to be differentiated. Making use of Eq. (2.54), we obtain the following expression for the derivative of the mass matrix:

$$\frac{\partial M_{qr}}{\partial h} = \sum_{m=1}^{numPts} \omega_m\ \rho\ N_{iq} N_{ir}\ |J_{x,\xi}^{(m)}|\ \left(J_{x_k,\xi_l}^{(m)}\right)^{-T}\ \frac{\partial N_{k\tilde{s}}}{\partial \xi_l} \tag{2.65}$$

Now turning to the derivative of the stiffness matrix, only the initial stiffness is considered due to the remarks made in Section 2.1.4. The derivative of the integrand then becomes:

$$\frac{\partial \left(k_{rs}^o\ B_{sp}^o\ B_{rq}^o\right)}{\partial h} = k_{rs}^o \left(\frac{\partial B_{sp}^o}{\partial h}\ B_{rq}^o + B_{sp}^o\ \frac{\partial B_{rq}^o}{\partial h}\right) \tag{2.66}$$

where it is noted that $k_{rs}^o = \left.\frac{\partial \sigma_r}{\partial \epsilon_s}\right|_{\epsilon=0}$ is a constant. The derivative of the B^o-matrix is computed according to Eq. (2.61). Thus, the following expression for the derivative of the stiffness matrix is obtained:

$$\frac{\partial K_{qp}}{\partial h} = \sum_{m=1}^{numPts} \omega_m \left[\left(k_{rs}^o \left(\frac{\partial B_{sp}^o}{\partial h}\ B_{rq}^o + B_{sp}^o\ \frac{\partial B_{rq}^o}{\partial h} \right) \right) |J_{x_k,\xi_l}^{(m)}| + k_{rs}^o\ B_{sp}^o\ B_{rq}^o\ |J_{x,\xi}^{(m)}|\ \left(J_{x_i,\xi_j}^{(m)}\right)^{-T}\ \frac{\partial N_{i\tilde{s}}}{\partial \xi_j} \right] \tag{2.67}$$

The external load vector P_q^{ext} may be composed of prescribed nodal loads or contributions from distributed loads as defined in Eq. (2.13). In the former case, the derivative $\frac{\partial P_q^{ext}}{\partial h}$ when h represents a nodal coordinate is zero. In the latter case the equations $P_q^{ext,\ surface} = \int_{\Gamma_t} t_i\ N_{iq}\ \mathrm{d}A$ and $P_q^{ext,\ volume} = \int_\Omega \rho\ b_i\ N_{iq}\ \mathrm{d}V$ must be differentiated. Nodal coordinates do not enter the integrands but enter the integration boundaries in the same manner as the other element integrals discussed in this section. Hence, we obtain:

$$\frac{\partial P_q^{ext,\ surface}}{\partial h} = \sum_{m=1}^{numPts} \omega_m\ (t_i\ N_{iq})\ |J_{x,\xi}^{(m)}|\ \left(J_{x_k,\xi_l}^{(m)}\right)^{-T}\ \frac{\partial N_{k\tilde{s}}}{\partial \xi_l} \tag{2.68}$$

$$\frac{\partial P_q^{ext,\ volume}}{\partial h} = \sum_{m=1}^{numPts} \omega_m\ (\rho\ b_i\ N_{iq})\ |J_{x,\xi}^{(m)}|\ \left(J_{x_k,\xi_l}^{(m)}\right)^{-T}\ \frac{\partial N_{k\tilde{s}}}{\partial \xi_l} \tag{2.69}$$

We note that the Jacobian of the surface integral is of one dimension less than that of the volume integral.

2.1.6 Displacement Sensitivity with Respect to Load Parameters

When h represents an external load parameter then the term $\frac{\partial P_q^{ext}}{\partial h}$ in the right-hand side of Eq. (2.34) is of main concern. However, it is noted that when inelastic materials are employed then $\left.\frac{\partial P_q^{int}}{\partial h}\right|_{\substack{u_p\, fixed \\ \dot{u}_p\, fixed}}$ is non-zero even when h represents a nodal load. This fact is further discussed in the following Section 2.2.

It is assumed that external load is applied as distributed element loads, base motion or prescribed nodal loads. In the first case the desired derivatives are obtained by employing Eqs. (2.13) and (2.30):

$$\frac{\partial P_q^{ext,\, surface}}{\partial h} = \int_{\Gamma_t} \frac{\partial t_i}{\partial h}\, N_{iq}\, \mathrm{d}A \approx \sum_{m=1}^{numPoints} \omega_m \left(\frac{\partial t_i}{\partial h}\, N_{iq}\right) |J_{x_k,\xi_l}^{(m)}| \tag{2.70}$$

$$\frac{\partial P_q^{ext,\, volume}}{\partial h} = \int_{\Omega} \rho\, \frac{\partial b_i}{\partial h}\, N_{iq}\, \mathrm{d}V \approx \sum_{m=1}^{numPoints} \omega_m \left(\frac{\partial b_i}{\partial h}\, N_{iq}\right) |J_{x_k,\xi_l}^{(m)}| \tag{2.71}$$

The case of sensitivities with respect to base motion parameters is discussed in Section 3.2.5.

Attention is now given to the case of prescribed nodal loads. By defining a vector h_p of parameters for which sensitivities are desired and a matrix of deterministic constants q_{qp} the external force vector is defined as:

$$P_q^{ext} = q_{qp}\, h_p \tag{2.72}$$

The desired derivative in the right-hand side of Eq. (2.34) reads:

$$\frac{\partial P_q^{ext}}{\partial h} = q_{qp}\, \frac{\partial h_p}{\partial h} \tag{2.73}$$

where $\frac{\partial h_p}{\partial h}$ is a vector containing zero elements and one element with unit value. It is noted that Eq. (2.72) includes the special case of one parameter representing one nodal load, in which case q_{qp} is a diagonal matrix. Additionally, Eq. (2.72) allows one parameter to represent all the nodal loads in the structure, each multiplied by a user-defined coefficient.

2.1.7 Sensitivity of Derived Response Quantities

After Eq. (2.34) has been solved for the displacement sensitivity vector $v_{(n+1)}$, it can be used to obtain sensitivity results for derived response quantities. Such quantities may include: (1) material strain, (2) material stress, (3) element end rotations, (4) cross-sectional stress resultants such as bending moment and axial force, (5) total base shear force, (6) inter-story drift and (7) accumulated response quantities, such as accumulated plastic strain or dissipated energy. The following remarks are offered on the computation of such response sensitivity results.

Material strain and stress: In OpenSees, these quantities are computed at the material level. The strain sensitivity is passed to the material object in "phase 2" of the sensitivity computations (see Section 2.2.1), namely after the displacement sensitivity vector is solved for. It is therefore directly available to the user. The unconditional derivative of the stress at the material level can also be easily computed once the strain sensitivity is known. The computations are similar to the ones for the conditional stress sensitivity, simply with a few added terms to include non-zero strain sensitivity. This is implemented for most material models with DDM capabilities in OpenSees.

Inter-story drift: Upon solving for the displacement sensitivity vector, sensitivity of quantities such as inter-story drift is obtained in a trivial manner. Inter-story drift is defined as difference between two nodal displacements. The corresponding sensitivity is the difference between the corresponding nodal displacement sensitivity values.

Cross-sectional stress resultants: These quantities represent integration of stress over the cross-section. In this study only fiber-discretized cross-sections are utilized. Hence, the sensitivities of the bending moment and axial force associated with a cross-section are found by differentiating the corresponding integrals of stress over the fiber cross-section. It is assumed that the unconditional derivatives of the stress are available from the material (see above).

Element end rotations: These computations are simply a matter of differentiating the algorithm which is used to obtain the element end rotations. This may involve obtaining the strain sensitivities of the fibers from the material level, which again is provided through "phase 2" of the sensitivity calculations (see Section 2.2.1).

Accumulated quantities: A typical accumulated response quantity is the accumulated plastic strain parameter in, e.g., J_2 plasticity. The sensitivity of this parameter is computed and stored as a sensitivity history variable, see Section 2.7.4, and is therefore available to the user without further implementation.

2.2 SPECIAL CONSIDERATIONS FOR INELASTIC PROBLEMS

The response of an inelastic material depends upon the previous loading history. In OpenSees, history variables are committed upon convergence at each step and used in the subsequent material state determinations. Response sensitivity analysis involving such materials require particular attention to the issues discussed in the following subsections.

2.2.1 Conditional and Unconditional Sensitivities; Two Phases

A central task in sensitivity computations by the DDM is assembly of the derivative of the internal force vector. According to Eq. (2.34), this derivative is to be computed with the assumption of fixed current displacement and velocity. From Eqs. (2.45) and (2.60) it is clear that this translates into assembly of stress derivatives from the material objects with the assumption of fixed current strain and strain rate. It is essential for the implementation of the DDM that only the *current* deformation be kept fixed. No assumption of fixed strains for the previous time steps should be made. For this reason the derivatives of the history variables must be stored without any assumption of fixed strain and strain rate. As emphasized by Zhang and Der Kiureghian (1993), this leads to a need for calling the material objects twice during the sensitivity computations. In "phase 1," the conditional stress sensitivity is propagated up to the element level, where the conditional derivative of the internal force vector is assembled. In "phase 2," the displacement sensitivity vector for the current step is available so that the corresponding strain sensitivities can be given to the material objects. Unconditional sensitivity history variables can then be saved.

This procedure is implemented in OpenSees. The sensitivity equations to be presented in Sections 2.6 to 2.7 are organized so as to distinguish the computations in "phase 1" and "phase 2."

2.2.2 Element Assembly Procedures

When assembling the conditional derivative of the internal force vector $\left.\frac{\partial P^{int}_{q\,(n+1)}}{\partial h}\right|_{\substack{u_{p\,(n+1)}\ fixed \\ \dot{u}_{p\,(n+1)}\ fixed}}$ the question may arise as to which elements contribute. Intuitively, one might assume that only the element containing the parameter h need to be included. The issue in this section is to discuss why this is

not the case.

In general, all elements of the displacement sensitivity vector $v_p = \frac{\partial u_p}{\partial h}$ are non-zero at every analysis step. In turn, this implies that the strain sensitivity $\frac{\partial \epsilon_p}{\partial h}$ at all material points is generally non-zero. Since the sensitivity history variables must be computed and stored without the assumption of fixed deformation, $\frac{\partial \epsilon_p}{\partial h}$ enters the expressions for the sensitivity history variables. Hence, after the initial step of the finite element analysis, $\left.\frac{\partial P^{int}_{q\,(n+1)}}{\partial h}\right|_{\substack{u_{p\,(n+1)}\,fixed \\ \dot{u}_{p\,(n+1)}\,fixed}}$ is non-zero for all inelastic elements. In OpenSees, inelastic material objects are called to contribute to the derivative of the internal force vector regardless of the nature of the parameter h.

2.2.3 On Using the Updated Consistent Tangent

Use of inelastic materials leads to a non-linear equation of motion, which can be solved by, e.g., a Newton-Raphson scheme as shown in Eq. (2.19). As emphasized by Simo and Hughes (1998), it is important that the algorithmically consistent tangent $K_{qp\,(n+1)}$ is employed. The algorithmically consistent tangent may be different from the analytical tangent and has the property of leading to optimal convergence properties of the numerical solution procedure. It is derived by differentiating the discretized equations instead of the continuous rate equations. Convergence may still occur, though with a slower rate, even if the analytical tangent is used or if small errors are present in the computation of the algorithmically consistent tangent.

In sensitivity analysis by the DDM this issue carries increased importance. Correct sensitivity results depend on the use of the correct algorithmically consistent tangent. This is made clear by the fact that no iterations are performed to solve Eq. (2.34). Hence, the accuracy of the constituent $\widetilde{K}_{qp\,(n+1)}$, which contains $K_{qp\,(n+1)}$, is critical.

Furthermore, one must make sure that the tangent is updated upon convergence of the response before it is employed in the sensitivity computations. In particular, this fact must be kept in mind when employing a Modified Newton-Raphson scheme, where the tangent is not updated at every step. In the implementations in OpenSees, an update of the algorithmically consistent tangent is automatically performed prior to sensitivity computations.

2.3 ELEMENT-LEVEL SENSITIVITY EQUATIONS

In this study, DDM sensitivity equations have been derived and implemented in OpenSees for the following structural components:

- Elements: (1) Four-node quad element, (2) displacement-based beam-column element with fiber cross-sections, and (3) nonlinear truss element.
- Cross-sections: Fiber cross-section composed of uniaxial material models.
- Materials: (1) Uniaxial degrading Bouc-Wen material, (2) 3-D generalized plasticity material, (3) uniaxial smoothed bi-linear steel material, and (4) uniaxial smoothed concrete material. In addition, the DDM equations developed by Zhang and Der Kiureghian (1993) for uniaxial and 3-D J_2 plasticity material models have been implemented.

The following sections describe the development of sensitivity equations for the above listed structural components. The Bouc-Wen material model and the generalized plasticity material model are introduced in OpenSees as part of this study. Also the hysteretic smoothing rules for the piecewise linear steel and concrete materials are developed as part of this study.

2.3.1 Isoparametric Quad4 Element

In "phase 1" of the sensitivity computations, the task of the element is to return its contribution to the derivative $\frac{\partial P_q^{int}}{\partial h}\Big|_{u_p\, fixed}$. The viscosity is assumed to be zero. For the isoparametric four-node quad element this involves implementation of the equations provided in the previous sections for the three types of parameters considered.

In "phase 2," the element receives the displacement sensitivity vector from the top-level solution algorithm and passes the strain sensitivity (or cross-section deformation sensitivity) down to the material (or the cross-section). This is done according to the same procedure that is used to produce the strain or cross-section deformation based on the current trial displacement vector in the ordinary finite element analysis.

2.3.2 Displacement-Based Beam-Column Element

The beam-column element used in this study is of a distributed hinge type. The element is "displacement-based" in the sense that the stiffness formulation is used as opposed to the flexibility (force) approach. A user-selected number of integration points along the element is prescribed. For the purpose of subsequent sensitivity derivations, the general framework of equilibrium and kinematic relations for this element is shown in Figure 2.1. The following notation is used: $\mathbf{p}$ and $\mathbf{u}$ are, respectively, the force vector and the displacement vector associated with the degrees of freedom in the global configuration; $\mathbf{q}$ and $\mathbf{v}$ are, respectively, the force vector and the displacement vector associated with the degrees of freedom in the basic configuration (in the basic configuration the degrees of freedom are capable of describing the element deformation, but not rigid body motions); $\mathbf{s}$ and $\mathbf{e}$ are, respectively, the section force vector and section deformation vector. For the purpose of observing where nodal coordinates enter, the transformation matrices $\mathbf{b}$, $\mathbf{B}$, and $\mathbf{a}$ are given below for the 2-D case:

$$\mathbf{a} = \begin{bmatrix} -\cos(\theta) & -\sin(\theta) & 0 & \cos(\theta) & \sin(\theta) & 0 \\ \frac{\sin(\theta)}{L} & -\frac{\cos(\theta)}{L} & 1 & -\frac{\sin(\theta)}{L} & \frac{\cos(\theta)}{L} & 0 \\ \frac{\sin(\theta)}{L} & -\frac{\cos(\theta)}{L} & 0 & -\frac{\sin(\theta)}{L} & \frac{\cos(\theta)}{L} & 1 \end{bmatrix} \tag{2.74}$$

$$\mathbf{B} = \begin{bmatrix} \frac{1}{L} & 0 & 0 \\ 0 & \frac{1}{L}\left(6\frac{x}{L} - 4\right) & \frac{1}{L}\left(6\frac{x}{L} - 2\right) \end{bmatrix} \tag{2.75}$$

$$\mathbf{b} = \begin{bmatrix} 1 & z \end{bmatrix} \tag{2.76}$$

In the above equations, L is the element length, x is the longitudinal coordinate running from 0 to L, θ is the angle of the undeformed element with the horizontal axis, and z is the coordinate perpendicular to the longitudinal coordinate.

Numerical integration over the specified number of integration points is performed along the element, usually in a normalized parent domain characterized by the coordinate $-1 \leq \xi \leq 1$:

$$I = \int_0^L f(x)\mathrm{d}x \approx L \sum_i \omega_i \, f(\xi_i) \tag{2.77}$$

where ω_i are weights corresponding to the integrations points ξ_i.

By combining the equilibrium equations in Figure 2.1, the internal force vector for the displacement-based beam-column element is expressed as:

$$\mathbf{p}_{int} = \mathbf{a}^T \int_0^L \mathbf{B}^T \int_A \mathbf{b}^T \, \sigma \, \mathrm{d}A \, \mathrm{d}x \tag{2.78}$$

We differentiate Eq. (2.78) with respect to h to subsequently arrive at the conditional derivative needed in Eq. (2.34). This approach is also employed by Scott *et al.* (2003) for sensitivity analysis with respect to material parameters for beam-column elements in a flexibility formulation. Neglecting viscous effects, we obtain:

$$\begin{aligned}\frac{\partial \mathbf{p}_{int}}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial h} + \left.\frac{\partial \mathbf{p}_{int}}{\partial h}\right|_{\mathbf{u}\ fixed} &= \frac{\partial \mathbf{a}^T}{\partial h}\int_0^L \mathbf{B}^T \int_A \mathbf{b}^T\,\sigma\ \mathrm{d}A\ \mathrm{d}x \\ + \mathbf{a}^T\int_0^L \frac{\partial \mathbf{B}^T}{\partial h}\int_A \mathbf{b}^T\,\sigma\ \mathrm{d}A\ \mathrm{d}x &+ \mathbf{a}^T\int_0^L \mathbf{B}^T\int_A \mathbf{b}^T\frac{\partial \sigma}{\partial h}\ \mathrm{d}A\ \mathrm{d}x \end{aligned} \tag{2.79}$$

The chain rule of differentiation is then applied to the term $\frac{\partial \sigma}{\partial h}$ so that all kinematic relations can be introduced and differentiated:

$$\begin{aligned}\frac{\partial \sigma}{\partial h} &= \frac{\partial \sigma}{\partial \epsilon}\frac{\partial \epsilon}{\partial h} + \left.\frac{\partial \sigma}{\partial h}\right|_{\epsilon\ fixed} \\ &= k_m\mathbf{b}\frac{\partial \mathbf{e}}{\partial h} + \left.\frac{\partial \sigma}{\partial h}\right|_{\epsilon\ fixed} \\ &= k_m\mathbf{b}\left(\frac{\partial \mathbf{B}}{\partial h}\mathbf{v} + \mathbf{B}\frac{\partial \mathbf{v}}{\partial h}\right) + \left.\frac{\partial \sigma}{\partial h}\right|_{\epsilon\ fixed} \\ &= k_m\mathbf{b}\left(\frac{\partial \mathbf{B}}{\partial h}\mathbf{v} + \mathbf{B}\left(\frac{\mathbf{a}}{\partial h}\mathbf{u} + \mathbf{a}\frac{\partial u}{\partial h}\right)\right) + \left.\frac{\partial \sigma}{\partial h}\right|_{\epsilon\ fixed} \\ &= k_m\mathbf{b}\frac{\partial \mathbf{B}}{\partial h}\mathbf{v} + k_m\mathbf{b}\mathbf{B}\frac{\mathbf{a}}{\partial h}\mathbf{u} + k_m\mathbf{b}\mathbf{B}\mathbf{a}\frac{\partial u}{\partial h} + \left.\frac{\partial \sigma}{\partial h}\right|_{\epsilon\ fixed}\end{aligned} \tag{2.80}$$

When substituted into Eq. (2.79), the third term of Eq. (2.80) cancels against the term $\frac{\partial \mathbf{p}_{int}}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial h}$. We are then left with the following expression for the conditional derivative of the internal force vector:

$$\begin{aligned}\left.\frac{\partial \mathbf{p}_{int}}{\partial h}\right|_{\mathbf{u}\ fixed} &= \frac{\partial \mathbf{a}^T}{\partial h}\int_0^L \mathbf{B}^T\int_A \mathbf{b}^T\,\sigma\ \mathrm{d}A\ \mathrm{d}x + \mathbf{a}^T\int_0^L \frac{\partial \mathbf{B}^T}{\partial h}\int_A \mathbf{b}^T\,\sigma\ \mathrm{d}A\ \mathrm{d}x \\ &+ \mathbf{a}^T\int_0^L \mathbf{B}^T\int_A \mathbf{b}^T\left(k_m\mathbf{b}\frac{\partial \mathbf{B}}{\partial h}\mathbf{v} + k_m\mathbf{b}\mathbf{B}\frac{\partial \mathbf{a}}{\partial h}\mathbf{u}\right)\ \mathrm{d}A\ \mathrm{d}x \\ &+ \mathbf{a}^T\int_0^L \mathbf{B}^T\int_A \mathbf{b}^T\left.\frac{\partial \sigma}{\partial h}\right|_{\epsilon\ fixed}\ \mathrm{d}A\ \mathrm{d}x\end{aligned} \tag{2.81}$$

This formulation is implemented in OpenSees. If h represents a material parameter then only the last term in the right-hand side of Eq. (2.81) is non-zero:

$$\left.\frac{\partial \mathbf{p}_{int}}{\partial h}\right|_{\mathbf{u}\ fixed} = \mathbf{a}^T\int_0^L \mathbf{B}^T\int_A \mathbf{b}^T\left.\frac{\partial \sigma}{\partial h}\right|_{\epsilon\ fixed}\ \mathrm{d}A\ \mathrm{d}x \tag{2.82}$$

On the other hand, if h represents a nodal coordinate then all terms in Eq. (2.81) must be computed. Derivatives of the transformation matrices $\mathbf{a}$ and $\mathbf{B}$ with respect to a nodal coordinate are easily derived.

In phase 2 of the sensitivity computations, the kinematic relations on the left side of Figure 2.1 are used to pass the displacement sensitivity down to the material object.

2.3.3 Nonlinear Truss Element

Similar to the beam-column element, for the truss element it is common to directly evaluate the element integrals circumventing the general isoparametric formulation presented in Section 2.1. Remarks are therefore made here regarding the derivative of the internal force vector for this particular element. The internal force vector is written:

$$\mathbf{P}_{int} = \sigma \; A \; \mathbf{T} \tag{2.83}$$

where A is the cross-sectional area and $\mathbf{T}$ is a transformation vector between the basic configuration (with one degree of freedom) and the global element configuration (with 4 or 6 degrees of freedom).

Differentiation of Eq. (2.83) with respect to h gives:

$$\frac{\partial \mathbf{p}_{int}}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial h} + \left. \frac{\partial \mathbf{p}_{int}}{\partial h} \right|_{\mathbf{u} \; fixed} = \frac{\partial \sigma}{\partial h} \; A \; \mathbf{T} + \sigma \; \frac{\partial A}{\partial h} \; \mathbf{T} + \sigma \; A \; \frac{\partial \mathbf{T}}{\partial h} \tag{2.84}$$

Following the procedure described in the previous sections, the stress derivative is expanded by introducing kinematic relations. In this case the strain is obtained as $\epsilon = \frac{\Delta L}{L}$, where ΔL is found from the global element displacement vector as: $\Delta L = \mathbf{u} \; \mathbf{T}$. The stress derivative is therefore expanded as follows:

$$\frac{\partial \sigma}{\partial h} = \frac{\partial \sigma}{\partial \epsilon} \frac{\partial \epsilon}{\partial h} + \left. \frac{\partial \sigma}{\partial h} \right|_{\epsilon \; fixed} = k \left(-\frac{1}{L^2} \frac{\partial L}{\partial h} \mathbf{u}\mathbf{T} + \frac{1}{L} \frac{\partial \mathbf{u}}{\partial h} \mathbf{T} + \frac{1}{L} \mathbf{u} \frac{\partial \mathbf{T}}{\partial h} \right) + \left. \frac{\partial \sigma}{\partial h} \right|_{\epsilon \; fixed} \tag{2.85}$$

where $k = \frac{\partial \sigma}{\partial \epsilon}$ is the material tangent stiffness. Substituting Eq. (2.85) into Eq. (2.84) and cancelling equal terms we obtain:

$$\left. \frac{\partial \mathbf{p}_{int}}{\partial h} \right|_{\mathbf{u} \; fixed} = A \; \mathbf{T} \left(-k \frac{\Delta L}{L^2} \frac{\partial L}{\partial h} + k \frac{1}{L} \mathbf{u} \frac{\partial \mathbf{T}}{\partial h} + \left. \frac{\partial \sigma}{\partial h} \right|_{\epsilon \; fixed} \right) + \sigma \; \frac{\partial A}{\partial h} \; \mathbf{T} + \sigma \, A \frac{\partial \mathbf{T}}{\partial h} \tag{2.86}$$

This is the general expression that is implemented in OpenSees. Certain terms cancel depending on the nature of the parameter represented by h. In particular, if h represents a material parameter, then Eq. (2.87) simplifies to:

$$\left. \frac{\partial \mathbf{p}_{int}}{\partial h} \right|_{\mathbf{u} \; fixed} = A \; \mathbf{T} \; \left. \frac{\partial \sigma}{\partial h} \right|_{\epsilon \; fixed} \tag{2.87}$$

2.4 CROSS-SECTIONAL LEVEL SENSITIVITY EQUATIONS

The equations governing equilibrium and kinematics at the cross-sectional level are shown in Figure 2.1. In this work a fiber cross section is employed. In the ordinary finite element procedure the

cross sections are responsible for two tasks. Firstly, to assemble stiffness and internal force contributions from the material fibers and pass them on to the element level. Secondly, to compute fiber strains based on the cross-sectional deformations provided by the element.

In "phase 1" of the sensitivity computations, the fiber cross section is responsible for assembling conditional stress derivatives from the fibers, as outlined in Eq. (2.82). In "phase 2" of the sensitivity computations, the task of the fiber cross section is to convert the cross-section deformation sensitivity $\frac{\partial \mathbf{e}}{\partial h}$ into strain sensitivity $\frac{\partial \epsilon}{\partial h}$ at each fiber. These operations are implemented in OpenSees as part of this study.

2.5 MATERIAL-LEVEL SENSITIVITY EQUATIONS

According to Eq. (2.45), the stress sensitivity for a fixed strain and strain rate is needed from the material object. As discussed in Section 2.2, this quantity must be computed regardless of whether the parameter h is part of the constitutive model of the material point in question. Equations to obtain such results are presented in the following sections. The selection of material models is motivated by applications in finite element reliability analysis. The material models presented here attempt to remedy the gradient discontinuity that occurs at the transition between elastic and plastic phases. This issue is further discussed in Section 2.10.

Four material models are considered in this study: namely, the uniaxial Bouc-Wen model, the 3-D generalized plasticity model, a smoothed uniaxial bilinear model and a smoothed uniaxial concrete model. In the past, sensitivity equations have been developed for the J_2 plasticity model (Zhang and Der Kiureghian 1993) and for the cap plasticity model (Vijalapura 1998, Conte 2000).

2.6 UNIAXIAL BOUC-WEN MATERIAL

The Bouc-Wen model is a smooth hysteretic material model developed by Bouc (1971) and Wen (1976). As will be demonstrated, the smooth transition between elastic and plastic responses is an attractive feature of this model for reliability analysis applications, since it avoids gradient discontinuities. Baber and Noori (1985) have extended the original Bouc-Wen model to include a degrading behavior. This version is considered for this implementation.

2.6.1 Fundamental Model Assumptions

The stress is defined as the sum of a linear part and a hysteretic part:

$$\sigma = \alpha\, k_o\, \epsilon + (1-\alpha)\; k_o\, z \tag{2.88}$$

In the above, ϵ is the strain, z represents the hysteretic deformation, k_o is the elastic stiffness and α is the ratio of the post-yielding to elastic stiffness. To accommodate degradation, Baber and Noori (1985) formulated the rate of hysteretic deformation in the form:

$$\dot{z} = \frac{A\dot{\epsilon} - \left\{\beta\,|\dot{\epsilon}|\,z\,|z|^{n-1} + \gamma\,\dot{\epsilon}|z|^n\right\}\nu}{\eta} \tag{2.89}$$

where β, γ, and n are parameters that control the shape of the hysteretic loop, while A, ν, and η are variables that control the material degradation. The model may be rewritten as:

$$\dot{z} = \frac{A - |z|^n\left\{\beta\,\mathrm{sgn}(\dot{\epsilon}z) + \gamma\right\}\nu}{\eta}\;\dot{\epsilon} = \frac{\partial z}{\partial \epsilon}\frac{\partial \epsilon}{\partial t} \tag{2.90}$$

This leads to the following expression for the continuum tangent (not the algorithmically consistent tangent):

$$k = \frac{\partial \sigma}{\partial \epsilon} = \alpha\, k_o + (1-\alpha)\; k_o\, \frac{A - |z|^n\left\{\gamma + \beta\,\mathrm{sgn}(\dot{\epsilon}z)\right\}\nu}{\eta} \tag{2.91}$$

It is seen that the stiffness is composed of a linear term and a hysteretic contribution.

The evolution of material degradation is governed by the following choice of equations (Baber and Noori 1985):

$$\begin{aligned} A &= A_o - \delta_A\; e \\ \nu &= 1 + \delta_\nu\; e \\ \eta &= 1 + \delta_\eta\; e \end{aligned} \tag{2.92}$$

where e is defined by the rate equation

$$\dot{e} = (1-\alpha)\; k_o\, \dot{\epsilon}\; z \tag{2.93}$$

and A_o, δ_A, δ_ν and δ_η are user-defined parameters.

2.6.2 Incremental Response Equations

To make the governing equations computer implementable one must first derive incremental response equations. From the above equations the stress at time t_{n+1} is obtained as:

$$\sigma_{(n+1)} = \alpha\, k_o\, \epsilon_{(n+1)} + (1-\alpha)\; k_o\, z_{(n+1)} \tag{2.94}$$

The rate equation for z is next discretized by a Backward Euler solution scheme. For a first-order ordinary differential equation of the form $\dot{y} = f(y(t))$, the scheme reads $y_{(n+1)} = y_n + \Delta t f(y_{(n+1)})$. Applied to Eq. (2.89) the following is obtained:

$$z_{(n+1)} = z_{(n)} + \Delta t\; \frac{A_{(n+1)} - |z_{(n+1)}|^n \left\{ \gamma + \beta\, \mathrm{sgn}\left(\frac{\left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)}{\Delta t} z_{(n+1)} \right) \right\} \nu_{(n+1)}}{\eta_{(n+1)}}\; \frac{\left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)}{\Delta t} \tag{2.95}$$

It is seen that Δt cancels from the equation, yielding a nonlinear equation in $z_{(n+1)}$. A Newton scheme of the form $x_{m+1} = x_m - f(x_m)/f'(x_m)$ to solve a general nonlinear equation $f(x) = 0$ may be used to solve for $z_{(n+1)}$ in Eq. (2.95).

The equations describing the degrading behavior are discretized as follows:

$$\begin{aligned} A_{(n+1)} &= A_o - \delta_A\; e_{(n+1)} \\ \nu_{(n+1)} &= 1 + \delta_\nu\; e_{(n+1)} \\ \eta_{(n+1)} &= 1 + \delta_\eta\; e_{(n+1)} \end{aligned} \tag{2.96}$$

where $e_{(n+1)}$ is found by discretization of the rate equation in Eq. (2.93), again utilizing the Backward Euler scheme:

$$e_{(n+1)} = e_{(n)} + \Delta t\; (1-\alpha)\; k_o\, \frac{\left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)}{\Delta t}\; z_{(n+1)} \tag{2.97}$$

where Δt again cancels. From the above equations it is clear that $z_{(n)}$, $\epsilon_{(n)}$, and $e_{(n)}$ are history variables that must be saved at each converged step.

The procedure implemented in OpenSees to compute the stress for a given strain can now be summarized as follows:

1. While $(|z^{old}_{(n+1)} - z^{new}_{(n+1)}| > tol)$

   (a) Evaluate function $f(z_{(n+1)})$:

$$\begin{aligned} e_{(n+1)} &= e_{(n)} + (1-\alpha)\; k_o\; \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)\; z_{(n+1)} \\ A_{(n+1)} &= A_o - \delta_A\; e_{(n+1)} \end{aligned}$$

$$
\begin{aligned}
\nu_{(n+1)} &= 1 + \delta_\nu \, e_{(n+1)} \\
\eta_{(n+1)} &= 1 + \delta_\eta \, e_{(n+1)} \\
\Psi &= \gamma + \beta \, \mathrm{sgn}\left(\left(\epsilon_{(n+1)} - \epsilon_{(n)} \right) z_{(n+1)} \right) \\
\Phi &= A_{(n+1)} - |z_{(n+1)}|^n \, \Psi \, \nu_{(n+1)} \\
f(z_{(n+1)}) &= z_{(n+1)} - z_{(n)} - \frac{\Phi}{\eta_{(n+1)}} \, \left(\epsilon_{(n+1)} - \epsilon_{(n)} \right)
\end{aligned}
$$

(b) Evaluate function derivatives (prime denotes derivative with respect to $z_{(n+1)}$):

$$
\begin{aligned}
e'_{(n+1)} &= (1-\alpha) \, k_o \, \left(\epsilon_{(n+1)} - \epsilon_{(n)} \right) \\
A'_{(n+1)} &= -\delta_A \, e'_{(n+1)} \\
\nu'_{(n+1)} &= \delta_\nu \, e'_{(n+1)} \\
\eta'_{(n+1)} &= \delta_\eta \, e'_{(n+1)} \\
\Phi' &= A'_{(n+1)} - n \, |z_{(n+1)}|^{n-1} \, \mathrm{sgn}\left(z_{(n+1)} \right) \Psi \nu_{(n+1)} - |z_{(n+1)}|^n \Psi \nu'_{(n+1)} \\
f'(z_{(n+1)}) &= 1 - \frac{\Phi' \, \eta_{(n+1)} - \Phi \, \eta'_{(n+1)}}{\eta^2_{(n+1)}} \, \left(\epsilon_{(n+1)} - \epsilon_{(n)} \right)
\end{aligned}
$$

(c) Obtain trial value in the Newton scheme:

$z^{new}_{(n+1)} = z_{(n+1)} - \frac{f(z_{(n+1)})}{f'(z_{(n+1)})}$

(d) Update $z_{(n+1)}$ (and store the old value for the convergence check):

$z^{old}_{(n+1)} = z_{(n+1)}$ and $z_{(n+1)} = z^{new}_{(n+1)}$

2. Compute stress: $\sigma_{(n+1)} = \alpha \, k_o \, \epsilon_{(n+1)} + (1-\alpha) \, k_o \, z_{(n+1)}$

In addition to the stress, the material algorithm must return the current algorithmically consistent tangent. This tangent is used in the global scheme to compute the nonlinear structural response. As pointed out in Section 2.2.3, the accuracy and consistency of the tangent is also crucial for the sensitivity analysis. It may be tempting to simply discretize Eq. (2.91) to obtain the tangent. However, this will lead to erroneous sensitivity results. We need to derive the tangent $\frac{\partial \sigma_{(n+1)}}{\partial \epsilon_{(n+1)}}$ by utilizing the formulation by which the stress $\sigma_{(n+1)}$ is actually computed. Hence, the equations used in the Newton scheme must be differentiated with respect to $\epsilon_{(n+1)}$. It turns out that the equation for $\frac{\partial z_{(n+1)}}{\partial \epsilon_{(n+1)}}$ is linear. The resulting implementable equations for the algorithmically consistent tangent read as follows:

1. Compute material degradation parameters:

$$
e_{(n+1)} = e_{(n)} + (1-\alpha) \, k_o \, \left(\epsilon_{(n+1)} - \epsilon_{(n)} \right) \, z_{(n+1)}
$$

$$
\begin{aligned}
A_{(n+1)} &= A_o - \delta_A \, e_{(n+1)} \\
\nu_{(n+1)} &= 1 + \delta_\nu \, e_{(n+1)} \\
\eta_{(n+1)} &= 1 + \delta_\eta \, e_{(n+1)}
\end{aligned}
$$

2. Compute auxiliary parameters:

$$
\begin{aligned}
\Psi &= \gamma + \beta \, \mathrm{sgn}\left(\left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) \; z_{(n+1)}\right) \\
\Phi &= A_{(n+1)} - |z_{(n+1)}|^n \, \Psi \, \nu_{(n+1)} \\
b_1 &= (1-\alpha) \; k_o \, z_{(n+1)} \\
b_2 &= (1-\alpha) \; k_o \; \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) \\
b_3 &= \frac{\left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)}{\eta_{(n+1)}} \\
b_4 &= -b_3 \, \delta_A \, b_1 - b_3 \, |z_{(n+1)}|^n \, \Psi \, \delta_\nu \, b_1 \\
&- \frac{\Phi}{\eta_{(n+1)}^2} \; \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) \; \delta_\eta \, b_1 + \frac{\Phi}{\eta_{(n+1)}} \\
b_5 &= 1 + b_3 \, \delta_A \, b_2 + b_3 \, n \, |z_{(n+1)}|^{n-1} \, \mathrm{sgn}\left(z_{(n+1)}\right) \; \Psi \, \nu_{(n+1)} \\
&+ b_3 \, |z_{(n+1)}|^n \, \Psi \, \delta_\nu \, b_2 \\
&+ \frac{\Phi}{\eta_{(n+1)}^2} \; \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) \; \delta_\eta \, b_2
\end{aligned}
$$

3. Compute $\frac{\partial z_{(n+1)}}{\partial \epsilon_{(n+1)}}$ and tangent:

$$
\begin{aligned}
\frac{\partial z_{(n+1)}}{\partial \epsilon_{(n+1)}} &= \frac{b_4}{b_5} \\
k = \frac{\partial \sigma_{(n+1)}}{\partial \epsilon_{(n+1)}} &= \alpha \, k_o + (1-\alpha) \; k_o \, \frac{\partial z_{(n+1)}}{\partial \epsilon_{(n+1)}}
\end{aligned}
$$

2.6.3 Conditional Stress Sensitivity Equations

According to the general procedures of sensitivity analysis by the DDM presented earlier in this study, the material must compute $\left.\frac{\partial \sigma_{(n+1)}}{\partial h}\right|_{\epsilon_{(n+1)} \; fixed}$ during "phase 1." To accomplish this, the incremental response equations are differentiated with respect to h, where h could be any of the involved material parameters, a nodal coordinate or a load parameter.

First the derivative of $z_{(n+1)}$ is derived:

$$
\begin{aligned}
\frac{\partial e_{(n+1)}}{\partial h} &= \frac{\partial e_{(n)}}{\partial h} \\
&- \frac{\partial \alpha}{\partial h}\, k_o \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)\, z_{(n+1)} \\
&+ (1-\alpha)\, \frac{\partial k_o}{\partial h} \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)\, z_{(n+1)} \\
&+ (1-\alpha)\, k_o \left(\frac{\partial \epsilon_{(n+1)}}{\partial h} - \frac{\partial \epsilon_{(n)}}{\partial h}\right)\, z_{(n+1)} \\
&+ (1-\alpha)\, k_o \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)\, \frac{\partial z_{(n+1)}}{\partial h} \\
\frac{\partial A_{(n+1)}}{\partial h} &= \frac{\partial A_o}{\partial h} - \frac{\partial \delta_A}{\partial h}\, e_{(n+1)} - \delta_A\, \frac{\partial e_{(n+1)}}{\partial h} \\
\frac{\partial \nu_{(n+1)}}{\partial h} &= \frac{\partial \delta_\nu}{\partial h}\, e_{(n+1)} + \delta_\nu\, \frac{\partial e_{(n+1)}}{\partial h} \\
\frac{\partial \eta_{(n+1)}}{\partial h} &= \frac{\partial \delta_\eta}{\partial h}\, e_{(n+1)} + \delta_\eta\, \frac{\partial e_{(n+1)}}{\partial h} \\
\frac{\partial \Psi}{\partial h} &= \frac{\partial \gamma}{\partial h} + \frac{\partial \beta}{\partial h}\, \mathrm{sgn}\left(\left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) z_{(n+1)}\right) \\
\frac{\partial \Phi}{\partial h} &= \frac{A_{(n+1)}}{\partial h} \\
&- \left\{ |z_{(n+1)}|^n \left(\frac{\partial n}{\partial h}\, log\left(|z_{(n+1)}|\right) + n\, \frac{1}{|z_{(n+1)}|}\, \frac{\partial z_{(n+1)}}{\partial h}\, \mathrm{sgn}\left(z_{(n+1)}\right) \right) \right\}\, \Psi\, \nu_{(n+1)} \\
&- |z_{(n+1)}|^n\, \frac{\partial \Psi}{\partial h}\, \nu_{(n+1)} \\
&- |z_{(n+1)}|^n\, \Psi\, \frac{\partial \nu_{(n+1)}}{\partial h} \\
\frac{\partial z_{(n+1)}}{\partial h} &= \frac{\partial z_{(n)}}{\partial h} \\
&+ \frac{\partial \Phi}{\partial h}\, \frac{1}{\eta_{(n+1)}}\, \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) \\
&- \Phi\, \frac{1}{\eta_{(n+1)}^2}\, \frac{\partial \eta_{(n+1)}}{\partial h}\, \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) \\
&- \frac{\Phi}{\eta_{(n+1)}}\, \frac{\partial \epsilon_{(n)}}{\partial h}
\end{aligned}
$$

When differentiating $|z_{(n+1)}|^n$, allowance is made for h to be the parameter n. For that case, $|z_{(n+1)}|^n = e^{n\, \log\left(|z_{(n+1)}|\right)}$ is used, where log() is the natural logarithm. The quantity $\frac{\partial \epsilon_{(n+1)}}{\partial h}$ appears in the above equations. During "phase 1" of the sensitivity computations, this quantity is set equal to zero to obtain the desired conditional derivatives. It is included here so that the same equations can be used in the following section when the unconditional sensitivity history variables are computed in "phase 2."

By combining the above equations it becomes clear that a linear equation for $\frac{\partial z_{(n+1)}}{\partial h}$ is obtained. By rearranging, the complete procedure to compute the conditional stress sensitivity (where $\frac{\partial \epsilon_{(n+1)}}{\partial h}$

should be set equal to zero to obtain the conditional derivatives) is as follows:

$$
\begin{aligned}
c_1 &= \frac{\partial e_{(n)}}{\partial h} \\
&- \frac{\partial \alpha}{\partial h} k_o \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) z_{(n+1)} \\
&+ (1-\alpha) \frac{\partial k_o}{\partial h} \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) z_{(n+1)} \\
&+ (1-\alpha) k_o \left(\frac{\partial \epsilon_{(n+1)}}{\partial h} - \frac{\partial \epsilon_{(n)}}{\partial h}\right) z_{(n+1)} \\
c_2 &= (1-\alpha) k_o \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) \\
c_3 &= \frac{\partial A_o}{\partial h} - \frac{\partial \delta_A}{\partial h} e_{(n+1)} - \delta_A c_1 \\
c_4 &= -\delta_A c_2 \\
c_5 &= \frac{\partial \delta_\nu}{\partial h} e_{(n+1)} + \delta_\nu c_1 \\
c_6 &= \delta_\nu c_2 \\
c_7 &= \frac{\partial \delta_\eta}{\partial h} e_{(n+1)} + \delta_\eta c_1 \\
c_8 &= \delta_\eta c_2 \\
A_{(n+1)} &= A_o - \delta_A e_{(n+1)} \\
\nu_{(n+1)} &= 1 + \delta_\nu e_{(n+1)} \\
\eta_{(n+1)} &= 1 + \delta_\eta e_{(n+1)} \\
\Psi &= \gamma + \beta \, \mathrm{sgn}(\left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) z_{(n+1)}) \\
\frac{\partial \Psi}{\partial h} &= \frac{\partial \gamma}{\partial h} + \frac{\partial \beta}{\partial h} \, \mathrm{sgn}(\left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) z_{(n+1)}) \\
\Phi &= A_{(n+1)} - |z_{(n+1)}|^n \, \Psi \, \nu_{(n+1)} \\
c_9 &= \frac{\left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)}{\eta_{(n+1)}} \\
c_{10} &= \frac{\partial z_{(n)}}{\partial h} + c_9 c_3 - c_9 |z_{(n+1)}|^n \frac{\partial n}{\partial h} log(|z_{(n+1)}|) \, \Psi \, \nu_{(n+1)} \\
&- c_9 |z_{(n+1)}|^n \frac{\partial \Psi}{\partial h} \nu_{(n+1)} - c_9 |z_{(n+1)}|^n \Psi \, c_5 \\
&- \frac{\Phi}{\eta_{(n+1)}^2} c_7 \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) + \frac{\Phi}{\eta_{(n+1)}} \left(\frac{\partial \epsilon_{(n+1)}}{\partial h} - \frac{\partial \epsilon_{(n)}}{\partial h}\right) \\
c_{11} &= 1 - c_9 c_4 + c_9 |z_{(n+1)}|^n \Psi \, c_6 \\
&+ c_9 |z_{(n+1)}|^n \frac{n}{|z_{(n+1)}|} \, \mathrm{sgn}(z_{(n+1)}) \, \Psi \, \nu_{(n+1)} \\
&+ \frac{\Phi}{\eta_{(n+1)}^2} c_8 \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right) \\
\frac{\partial z_{(n+1)}}{\partial h} &= \frac{c_{10}}{c_{11}}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \sigma_{(n+1)}}{\partial h} &= \frac{\partial \alpha}{\partial h}\, k_o\, \epsilon_{(n+1)} \\
&+ \alpha\, \frac{\partial k_o}{\partial h}\, \epsilon_{(n+1)} \\
&- \frac{\partial \alpha}{\partial h}\, k_o\, z_{(n+1)} \\
&+ (1-\alpha)\, \frac{\partial k_o}{\partial h}\, z_{(n+1)} \\
&+ (1-\alpha)\, k_o\, \frac{\partial z_{(n+1)}}{\partial h}
\end{aligned}
$$

2.6.4 Unconditional Sensitivity History Variables

From the equations in Section 2.6.3 it is observed that $\frac{\partial z_{(n)}}{\partial h}$, $\frac{\partial e_{(n)}}{\partial h}$, and $\frac{\partial \epsilon_{(n)}}{\partial h}$ are sensitivity history variables. These must be stored at each step without the assumption of a fixed strain level (see Section 2.2.1). The corresponding algorithm reads:

1. $\frac{\partial \epsilon_{(n+1)}}{\partial h}$ is passed into the material routine from the element/section level.
2. $\frac{\partial e_{(n+1)}}{\partial h} = \frac{\partial e_{(n)}}{\partial h} - \frac{\partial \alpha}{\partial h}\, k_o \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)\, z_{(n+1)} + (1-\alpha)\, \frac{\partial k_o}{\partial h} \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)\, z_{(n+1)} +$
   $(1-\alpha)\, k_o \left(\frac{\partial \epsilon_{(n+1)}}{\partial h} - \frac{\partial \epsilon_{(n)}}{\partial h}\right)\, z_{(n+1)} + (1-\alpha)\, k_o \left(\epsilon_{(n+1)} - \epsilon_{(n)}\right)\, \frac{\partial z_{(n+1)}}{\partial h}$
3. c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8, $A_{(n+1)}$, $\nu_{(n+1)}$, $\eta_{(n+1)}$, Ψ, $\frac{\partial \Psi}{\partial h}$, Φ, c_9, c_{10}, and c_{11} are computed by the same equations as when computing the conditional sensitivities above, but now with the actual value of $\frac{\partial \epsilon_{(n+1)}}{\partial h}$.
4. $\frac{\partial z_{(n+1)}}{\partial h} = \frac{c_{10}}{c_{11}}$

2.6.5 Example Results

Sensitivity results for two examples are presented in this section. First, consider a single material point subjected to a uniaxial stress history linearly interpolated between the following points:

$$
\begin{array}{rccccccccc}
\text{Pseudo-time:} & 0.0 & 1.0 & 2.0 & 3.0 & 4.0 & 5.0 & 6.0 & 7.0 & 8.0 \\
\text{Stress:} & 0.0 & 1.0 & 0.0 & -1.0 & 0.0 & 1.0 & 0.0 & -1.0 & 0.0
\end{array}
\tag{2.98}
$$

The resulting stress-strain curve is shown in Figure 2.2 for the following selection of material parameters: $\alpha = 0.1$, $k_o = 1.0$, $n = 1.5$, $\gamma = 0.5$, $\beta = 0.5$, $A_o = 1.0$, $\delta_A = 0.02$, $\delta_\nu = 0.08$, and $\delta_\eta = 0.08$.

Figure 2.3 presents the displacement sensitivity results in the form $\frac{\partial u}{\partial h} \cdot h$. That is, for each parameter the displacement sensitivity is multiplied by the value of the parameter itself. This normalizes the sensitivity results to the same unit as that of the response. Alternative normalization procedures are available (Roth and Grigoriu 2001), e.g., $\frac{\partial u}{\partial h} \cdot \sigma_h$ where σ_h is the standard deviation of h. Additional importance measures obtained from reliability analysis will be discussed in Section 3.6. The results in Figure 2.3 have been verified by finite difference calculations.

Several observations can be made in Figure 2.3. Firstly, the sensitivity results are continuous. This is an important quality for the application of sensitivity results in reliability analysis. Secondly, it is seen that the displacement is most sensitive to parameter k_o for this example and the chosen scaling. That is, k_o is the parameter with the most influence on the displacement response if all parameters are perturbed by the same relative magnitude.

The second example is the 3-D truss in Figure 5.4 on page 195, details of which are presented in Section 5.2. By applying the load series

$$\begin{array}{lccccc} \text{Time:} & 0.0 & 1.0 & 2.0 & 3.0 & 4.0 \\ \text{Load:} & 0.0 & 400\,kN & 0.0 & -400\,kN & 0.0 \end{array} \tag{2.99}$$

at each of the top nodes, the load-displacement curve in Figure 2.4 is obtained for the x-direction displacement of the top node versus the load factor. In Figure 2.5, the displacement sensitivities (for x-displacement of the top node 21) are presented for a selection of parameters. The sensitivity results are scaled by the respective standard deviations: $\frac{\partial u}{\partial h} \cdot \sigma_h$. The standard deviations are calculated by assuming 5% coefficient of variation for all variables, except for the nodal coordinates for which a standard deviation of 10mm is assumed. From numerical studies in Chapter 4.6 it is found that element 4 is the most important element for the top displacement in the x-direction. It is observed that the top displacement is most sensitive to the parameter n of this element. It is also seen that the x-coordinate of node 4 is also relatively important.

2.7 GENERALIZED PLASTICITY MATERIAL

The material model presented above is uniaxial. For reliability analysis purposes it is desirable to also have available a multi-axial material model, which exhibits the same smooth features in sensitivity. One of the most commonly used non-linear multi-axial material models is the J_2 plasticity model (Simo and Hughes 1998). However, this model has a nonsmooth transition between the elastic

regime and the plastic flow regime. Therefore, an alternative is sought. Casciati (1989) developed an extension of the uniaxial Bouc-Wen model for the multi-axial case. However, in this study the work of Lubliner *et al.* (1993) on the so-called generalized plasticity model is utilized. This material model can be seen as an extension of the J_2 plasticity model. Its main characteristic compared with J_2 plasticity is a smooth transition between the elastic and plastic response regimes.

2.7.1 Fundamental Model Assumptions

In classical plasticity theory the stress state is bounded by a yield function f, whose value is determined by the deviatoric stress state. $f < 0$ implies an elastic stress state, while plastic flow may occur when the stress state is on the yield surface characterized by $f = 0$. In generalized plasticity the stress state is allowed to exceed the yield surface. $f \geq 0$ implies that inelastic effects are or are not occurring, depending on whether loading or unloading occurs.

The generalized plasticity model is characterized by 7 parameters. The Young's modulus E, Poisson's ratio ν, yield stress σ_y, isotropic hardening modulus H_{iso}, and kinematic hardening modulus H_{kin} are known from classical plasticity theory. In generalized plasticity two additional parameters β and δ are present. β is a scalar measure of the distance between the asymptotic and the original yield surfaces. δ measures the speed by which the asymptotic yield state is reached. Furthermore, from the algorithm described below it is seen that the deviatoric plastic strain $\mathbf{e}^p_{(n)}$, the isotropic hardening parameter $\bar{e}^p_{(n)}$, and the norm of the stress tensor $\boldsymbol{\sigma}_{(n)}$ are history variables.

2.7.2 Incremental Response Equations

In the following the generalized plasticity model is presented in the form it has been implemented in OpenSees. Kinematic hardening effects are neglected in this implementation. However, this effect can be easily added according to the classical plasticity theory.

The stress and strain tensors are written in Voigt notation: $\boldsymbol{\sigma} = [\sigma_{11}\ \sigma_{22}\ \sigma_{33}\ \sigma_{12}\ \sigma_{23}\ \sigma_{13}]$, $\boldsymbol{\epsilon} = [\epsilon_{11}\ \epsilon_{22}\ \epsilon_{33}\ \gamma_{12}\ \gamma_{23}\ \gamma_{13}\ ]$, where γ denotes engineering shear strains. To handle the transformation

between continuum mechanics tensors and Voigt vectors, the following auxiliary matrices are defined:

$$\mathbf{m} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \mathbf{I}_o = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix} \quad \mathbf{I}_{dev} = \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} & 0 & 0 & 0 \\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} & 0 & 0 & 0 \\ -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix}$$

Furthermore, the following alternative elastic constants are used: $\mu = \frac{E}{2(1+\nu)}$, $K = \frac{E}{3(1-2\nu)}$ and $\lambda = \frac{\nu\, E}{(1+\nu)(1-2\nu)}$. The algorithm then proceeds as follows:

1. Compute the volumetric and deviatoric parts of the current strain tensor:

$$\begin{aligned} \epsilon^{vol}_{(n+1)} &= \epsilon_{11,n+1} + \epsilon_{22,n+1} + \epsilon_{33,n+1} \\ \mathbf{e}_{(n+1)} &= \boldsymbol{\epsilon}_{(n+1)} - \mathbf{m}\, \frac{\epsilon^{vol}_{(n+1)}}{3} \end{aligned}$$

2. Compute a trial elastic deviatoric stress vector:

$$\mathbf{s}^{trial}_{(n+1)} = 2\mu \left(\mathbf{e}_{(n+1)} - \mathbf{e}^{p}_{(n)} \right)$$

3. Based on the trial elastic state, evaluate the yield function:

$$f^{trial}_{(n+1)} = \|\mathbf{s}^{trial}_{(n+1)}\| - \sqrt{\frac{2}{3}} \left(\sigma_y + H_{iso} \bar{e}^{p}_{(n)} \right)$$

4. Compute A_2 as the difference between the current and the previous norm of the deviatoric stress tensor:

$$A_2 = \|\mathbf{s}^{trial}_{(n+1)}\| - \|\mathbf{s}_{(n)}\|$$

5. **If** $f^{trial}_{(n+1)} < 0$ **or** $A_2 < 0$ **then** the step is elastic (either because the stress state is inside the elastic region or because the stress state moves towards the elastic region due to unloading). Compute:

   (a) Stress:

   $$\boldsymbol{\sigma}_{(n+1)} = \mathbf{s}^{trial}_{(n+1)} + \mathbf{m}\, K\, \epsilon^{vol}_{(n+1)}$$

   (b) Tangent:

   $$\mathbf{D}_{(n+1)} = \mathbf{I}_o\, 2\mu + \mathbf{m}\mathbf{m}^T \lambda$$

(c) History variables:

$\mathbf{e}^p_{(n+1)} = \mathbf{e}^p_{(n)}$

$\bar{e}_{(n+1)} = \bar{e}_{(n)}$

$\|\mathbf{s}_{(n+1)}\|$

6. **Else**, compute the plastic flow by solving the second-order equation (Lubliner *et al.* 1993):

(a) Auxiliary parameters:

$$\begin{aligned} A_1 &= f^{trial}_{(n+1)} \\ A_3 &= \delta - 2\mu \\ A_4 &= \beta\left(\delta + \frac{2}{3}H_{iso}\right) \\ a &= \left(2\mu + \frac{2}{3}H_{iso}\right) A_3 \\ b &= A_4 - A_1\, A_3 + \left(2\mu + \frac{2}{3}H_{iso}\right) A_2 \\ c &= -A_1\, A_2 \end{aligned}$$

(b) The plastic flow parameter is the smallest positive root of the equation $a\,(\Delta\gamma)^2 + b\,(\Delta\gamma) + c = 0$:

$$\Delta\gamma = \min_{+}\left\{\frac{-b \pm \sqrt{b^2 - 4\,a\,c}}{2\,a}\right\}$$

(c) Yield surface normal: $\mathbf{n} = \frac{\mathbf{s}^{trial}_{(n+1)}}{\|\mathbf{s}^{trial}_{(n+1)}\|}$

(d) Deviatoric stress: $\mathbf{s}_{(n+1)} = \mathbf{s}^{trial}_{(n+1)} - \mathbf{n}\,(2\mu\Delta\gamma)$

(e) Stress:

$\boldsymbol{\sigma}_{(n+1)} = \mathbf{s}_{(n+1)} + \mathbf{m}\,K\,\epsilon^{vol}_{(n+1)}$

(f) Tangent:

$B_1 = \|\mathbf{s}_{(n+1)}\| - \|\mathbf{s}_{(n)}\| + \delta\ \Delta\gamma$

$B_2 = A_1 - \left(2\mu + \frac{2}{3}H_{iso}\right)\ \Delta\gamma$

$B_3 = A_4$

$A^{GP} = 2\mu\frac{(B_1+B_2)}{\left(\left(2\mu+\frac{2}{3}H_{iso}\right)B1 - A3\,B2 + B3\right)}$

$C^{GP} = 2\mu\frac{\Delta\gamma}{\|\boldsymbol{\sigma}_{(n+1)}\|}$

$\mathbf{D}_{vol} = K\ \mathbf{m}\mathbf{m}^T$

$\mathbf{D}_{dev} = 2mu\left(1.0 - C^{GP}\right)\mathbf{I}_{dev} + \left(2\mu\left(C^{GP} - A^{GP}\right)\right)\ \mathbf{n}\mathbf{n}^T$

$\mathbf{D}_{(n+1)} = D_{dev} + D_{vol}$

(g) History variables:

$$\mathbf{e}^p_{(n+1)} = \mathbf{e}^p_{(n)} + \mathbf{n}\Delta\gamma$$
$$\bar{e}^p_{(n+1)} = \bar{e}^p_{(n)} + \sqrt{\tfrac{2}{3}}\,\Delta\gamma$$
$$\|\mathbf{s}_{(n+1)}\| = \|\mathbf{s}_{(n+1)}\|$$

2.7.3 Conditional Stress Sensitivity Equations

As stated previously, the stress derivatives for fixed strain must first be computed. For this material model the algorithm takes the following form:

1. First the derivative of the trial elastic deviatoric stress state is computed:

$$\frac{\partial \mathbf{s}^{trial}_{(n+1)}}{\partial h} = 2\frac{\partial \mu}{\partial h}\left(\mathbf{e}_{(n+1)} - \mathbf{e}^p_{(n)}\right) - 2\mu\frac{\partial \mathbf{e}^p_{(n)}}{\partial h}$$

2. **If** the step is elastic, compute:

   (a) Conditional stress derivative:
   $\frac{\partial \boldsymbol{\sigma}_{(n+1)}}{\partial h} = \frac{\partial \mathbf{S}^{trial}_{(n+1)}}{\partial h} + \mathbf{m}\,\frac{\partial K}{\partial h}\,\epsilon^{vol}_{(n+1)}$

   (b) Derivative of elastic (initial) tangent:
   $\frac{\partial \mathbf{D}_{(n+1)}}{\partial h} = \mathbf{I}_o\,2\frac{\partial \mu}{\partial h} + \mathbf{m}\mathbf{m}^T\frac{\partial \lambda}{\partial h}$

3. **Else** plastic step, compute:

   (a) Derivative of auxiliary parameters (it is assumed that the values of the parameters themselves are already computed and available):

$$\begin{aligned}
\frac{\partial \|\mathbf{s}^{trial}_{(n+1)}\|}{\partial h} &= \frac{1}{2\|\mathbf{s}^{trial}_{(n+1)}\|}\Big(2\,s^{trial}_{11}\frac{\partial s^{trial}_{11}}{\partial h} + 2\,s^{trial}_{22}\frac{\partial s^{trial}_{22}}{\partial h} + 2\,s^{trial}_{33}\frac{\partial s^{trial}_{33}}{\partial h} \\
&\qquad + \;4\,s^{trial}_{12}\frac{\partial s^{trial}_{12}}{\partial h} + 4\,s^{trial}_{23}\frac{\partial s^{trial}_{23}}{\partial h} + 4\,s^{trial}_{13}\frac{\partial s^{trial}_{13}}{\partial h}\Big) \\
\frac{\partial f^{trial}_{(n+1)}}{\partial h} &= \frac{\partial \|\mathbf{s}^{trial}_{(n+1)}\|}{\partial h} - \sqrt{\frac{2}{3}}\left(\frac{\partial \sigma_y}{\partial h} + \frac{\partial H_{iso}}{\partial h}\bar{e}^p_{(n)} + H_{iso}\frac{\partial \bar{e}^p_{(n)}}{\partial h}\right) \\
\frac{\partial A_1}{\partial h} &= \frac{\partial f^{trial}_{(n+1)}}{\partial h} \\
\frac{\partial A_2}{\partial h} &= \frac{\partial \|\mathbf{s}^{trial}_{(n+1)}\|}{\partial h} - \frac{\partial \|\mathbf{s}_{(n)}\|}{\partial h} \\
\frac{\partial A_3}{\partial h} &= \frac{\partial \delta}{\partial h} - 2\frac{\partial \mu}{\partial h}
\end{aligned}$$

$$
\begin{aligned}
\frac{\partial A_4}{\partial h} &= \frac{\partial \beta}{\partial h}\left(\delta + \frac{2}{3}H_{iso}\right) + \beta\left(\frac{\partial \delta}{\partial h} + \frac{2}{3}\frac{\partial H_{iso}}{\partial h}\right) \\
\frac{\partial a}{\partial h} &= \frac{\partial A_3}{\partial h}\left(2\mu + \frac{2}{3}H_{iso}\right) + A_3\left(2\frac{\partial \mu}{\partial h} + \frac{2}{3}\frac{\partial H_{iso}}{\partial h}\right) \\
\frac{\partial b}{\partial h} &= \frac{\partial A_4}{\partial h} - \frac{\partial A_1}{\partial h}\,A_3 - A_1\,\frac{\partial A_3}{\partial h} + \frac{\partial A_2}{\partial h}\left(2\mu + \frac{2}{3}H_{iso}\right) \\
&\qquad + A_2\left(2\frac{\partial \mu}{\partial h} + \frac{2}{3}\frac{\partial H_{iso}}{\partial h}\right) \\
\frac{\partial c}{\partial h} &= -\frac{\partial A_1}{\partial h}\,A_2 - A_1\,\frac{\partial A_2}{\partial h}
\end{aligned}
$$

(b) The derivative of the plastic flow parameter (the plus or minus sign is selected according to the root that governs):

$$
\frac{\partial \Delta\gamma}{\partial h} = \frac{\left[-\frac{\partial b}{\partial h} \pm \frac{1}{2\sqrt{b^2-4\,a\,c}}\left(2\,b\frac{\partial b}{\partial h} - 4\,\frac{\partial a}{\partial h}c - 4\,a\,\frac{\partial c}{\partial h}\right)\right]\,2\,a - \left(-b \pm \sqrt{b^2-4\,a\,c}\right)\,2\,\frac{\partial a}{\partial h}}{(2\,a)^2}
$$

(c) Derivative of yield surface normal: $\frac{\partial \mathbf{n}}{\partial x} = \frac{\frac{\partial \mathbf{s}^{trial}_{(n+1)}}{\partial x}\,\|\mathbf{s}^{trial}_{(n+1)}\| - \mathbf{s}^{trial}_{(n+1)}\,\frac{\partial \|\mathbf{s}^{trial}_{(n+1)}\|}{\partial x}}{\|\mathbf{s}^{trial}_{(n+1)}\|^2}$

(d) Derivative of deviatoric stress tensor: $\frac{\partial \mathbf{s}_{(n+1)}}{\partial h} = \frac{\partial \mathbf{s}^{trial}_{(n+1)}}{\partial h} - \frac{\partial \mathbf{n}}{\partial h}\,(2\mu\Delta\gamma) - \mathbf{n}\left(2\frac{\partial \mu}{\partial h}\Delta\gamma + 2\mu\frac{\partial \Delta\gamma}{\partial h}\right)$

(e) Derivative of final stress:
$\frac{\partial \boldsymbol{\sigma}_{(n+1)}}{\partial h} = \frac{\partial \mathbf{S}_{(n+1)}}{\partial h} + \mathbf{m}\,\frac{\partial K}{\partial h}\,\epsilon^{vol}_{(n+1)}$

2.7.4 Unconditional Sensitivity History Variables

As seen in the previous section, three sensitivity history variables must be stored. These are $\frac{\partial \mathbf{e}^p_{(n+1)}}{\partial h}$, $\frac{\partial \bar{e}^p_{(n+1)}}{\partial h}$ and $\frac{\partial \|\mathbf{s}_{(n+1)}\|}{\partial h}$. In the "phase 2" call to the material object, the sensitivity of the strain tensor is known since the displacement sensitivity vector is available. The needed unconditional sensitivity history variables can then be stored in the following manner:

1. Compute the derivative of the deviatoric part of the current strain tensor:

$$
\frac{\partial \mathbf{e}_{(n+1)}}{\partial h} = \frac{\partial \boldsymbol{\epsilon}_{(n+1)}}{\partial h} - \mathbf{m}\,\frac{1}{3}\frac{\partial \epsilon^{vol}_{(n+1)}}{\partial h}
$$

where $\frac{\partial \epsilon^{vol}_{(n+1)}}{\partial h} = \frac{\partial \epsilon_{11}}{\partial h} + \frac{\partial \epsilon_{22}}{\partial h} + \frac{\partial \epsilon_{33}}{\partial h}$.

2. Determine the derivative of the trial elastic deviatoric stress tensor:

$$
\frac{\partial \mathbf{s}^{trial}_{(n+1)}}{\partial h} = 2\frac{\partial \mu}{\partial h}\left(\mathbf{e}_{(n+1)} - \mathbf{e}^p_{(n)}\right) + 2\mu\left(\frac{\partial \mathbf{e}_{(n+1)}}{\partial h} - \frac{\partial \mathbf{e}^p_{(n)}}{\partial h}\right)
$$

3. **If** the step is elastic:

   (a) There is no change in the first two sensitivity history variables, while the value of the third is set by using the result of the previous item:
   1) $\frac{\partial \mathbf{e}^p_{(n+1)}}{\partial h} = \frac{\partial \mathbf{e}^p_{(n)}}{\partial h}$
   2) $\frac{\partial \bar{e}_{(n+1)}}{\partial h} = \frac{\partial \bar{e}_{(n)}}{\partial h}$
   3) $\frac{\partial \|\mathbf{s}_{(n+1)}\|}{\partial h} = \frac{\partial \|\mathbf{s}^{trial}_{(n+1)}\|}{\partial h}$ (Computed as shown in item 3 (a) in the previous section.)

4. **else** plastic step:

   (a) The derivatives of the plastic flow parameter $\Delta\gamma$ and its auxiliary parameters are computed by the equations in item 3 in the previous section, but now including the strain sensitivity. The same equations are also used for the derivative of the normal $\mathbf{n}$ to the yield surface and the derivative of the deviatoric stress tensor $\mathbf{s}_{(n+1)}$.

   (b) Compute the derivative of the norm of the deviatoric stress tensor:
   $\frac{\partial \|\mathbf{s}_{(n+1)}\|}{\partial h} = \frac{1}{\|\mathbf{s}_{(n+1)}\|}\left(s_{11}\frac{\partial s_{11}}{\partial h} + s_{22}\frac{\partial s_{22}}{\partial h} + s_{33}\frac{\partial s_{33}}{\partial h} + 2\,s_{12}\frac{\partial s_{12}}{\partial h} + 2\,s_{23}\frac{\partial s_{23}}{\partial h} + 2\,s_{13}\frac{\partial s_{13}}{\partial h}\right)$

   (c) Compute and store the sensitivity history variables:
   1) $\frac{\partial \mathbf{e}^p_{(n+1)}}{\partial h} = \frac{\partial \mathbf{e}^p_{(n)}}{\partial h} + \frac{\partial \mathbf{n}}{\partial h}\Delta\gamma + \mathbf{n}\frac{\partial \Delta\gamma}{\partial h}$
   2) $\frac{\partial \bar{e}^p_{(n+1)}}{\partial h} = \frac{\partial \bar{e}^p_{(n)}}{\partial h} + \sqrt{\frac{2}{3}}\frac{\partial \Delta\gamma}{\partial h}$
   3) $\frac{\partial \|\mathbf{s}_{(n+1)}\|}{\partial h} = \frac{\partial \|\mathbf{s}_{(n+1)}\|}{\partial h}$

2.7.5 Example Results

A single four-node quad element with four integration points is used to exemplify the sensitivity results obtained by the plain strain generalized plasticity model. It is of interest to see how this material model compares to the classical J_2 plasticity model. The element geometry is selected to be square with $L_1 = L_2 = 0.2\,m$. See Figure 2.6. The element is fixed against any displacement at the lower left node and fixed against horizontal displacement at the upper left node. Concentrated nodal loads are applied at the upper right node with magnitude $P_h = 100.0\,kN$ in the horizontal direction and magnitude $P_v = 25.0\,kN$ in the vertical direction. These loads are multiplied by a load factor interpolated linearly between the following points:

$$\begin{array}{rccccc} \text{Pseudo-time:} & 0.0 & 1.0 & 2.0 & 3.0 & 4.0 \\ \text{Load factor:} & 0.0 & 1.0 & 0.0 & -1.0 & 0.0 \end{array} \tag{2.100}$$

For the classical J_2 plasticity model, the following material parameters are used: $\sigma_y = 400.0\,\mathrm{N/mm^2}$, $E = 210,000.0\,\mathrm{N/mm^2}$, $\nu = 0.2$ and $H_{iso} = 10\%$ of E. For the generalized plasticity material the distance between the asymptotic and the original yield surfaces is selected as $\beta = 100.0\,\mathrm{N/mm^2}$. To facilitate comparison between the two models the yield stress of the generalized plasticity model is selected as $\sigma_y^{GP} = \sigma_y - \beta = 300.0\,\mathrm{N/mm^2}$. The speed by which the asymptotic yield state is reached is chosen as $\delta = 40.0\,\mathrm{N/mm^2}$. The other parameters are equal to those of the J_2 plasticity material.

The resulting load-displacement curves for the horizontal displacement at the upper right node for the two material models are shown in Figure 2.7. It is observed that the J_2 plasticity material exhibits kinks as the integration points reach yielding. The generalized plasticity material has a smooth load-displacement curve. In this example the difference in response between the two models is not significant. It is emphasized, however, that the difference between the two material models may become substantial, depending on selection of material parameters and applied load series.

Figures 2.9, 2.8, 2.10 and 2.11 present the displacement sensitivity results. The results are obtained by the DDM and have been verified by finite differences.

Figure 2.8 shows that the sensitivity results with respect to the yield stress are discontinuous for the J_2 plasticity model. This finding is consistent with the previous results of, e.g., Zhang and Der Kiureghian (1993) and Kleiber *et al.* (1997). In contrast, Figure 2.8 shows that the generalized plasticity model considered in this study exhibits continuous displacement sensitivity results. This has an important bearing for the convergence behavior in the search for the design point in reliability analysis. Figure 2.8 also reveals that, for the J_2 plasticity model, only three of the four integration points reach plasticity during the applied load series.

Figures 2.9, 2.10 and 2.11 further confirm the smoothing effects of the generalized plasticity material model. The differences in the sensitivity results between the two material models are not significant except for parameter σ_y. The discrepancy for σ_y is explained by the gradual versus instant yielding for the two models, and the fact that the yield stress is different for the two models.

2.8 UNIAXIAL SMOOTHED BI-LINEAR MATERIAL

In practical applications, the uniaxial bi-linear material model is commonly used to represent the behavior of steel. In this section we develop a smoothed version of this material model. The objective is to avoid the gradient discontinuities that occur at the point of yielding. In this study, the transition between the elastic and plastic response states is smoothed by a circular segment. By varying the

radius it is possible to investigate the effect of various levels of smoothing.

2.8.1 Fundamental Model Assumptions

The bi-linear model is characterized by two response regimes, namely elastic and plastic. In the elastic range the tangent stiffness is equal to E. In the plastic range the stiffness is equal to $b\,E$, where $0 < b < 1$. The stress at which the transition between elastic and plastic states occurs is determined by the yield strength σ_y. Unloading is assumed to be elastic.

The circular segment used to smooth the transition between elastic and plastic states is tangent to both the elastic and plastic responses. The tangent stiffnesses coincide at the points of intersection. See Figure 2.12. Due to the different scales of the strain and the stress, the smoothing line is developed in a normalized $x - y$ plane. The $x - y$ plane is characterized by a yield strength equal to 1.0. The corresponding yield strain is η^{-1} where $\eta > 0$ is a user-defined parameter. A second user-defined parameter $0 < \gamma < 1$ denotes the fraction of the yield strength at which the smoothing segment intersects the elastic response. It is noted that, to obtain the correct hysteretic behavior, the coordinates of the center of the circle must be updated during the analysis.

An example stress-strain curve using this material model as it is implemented in OpenSees is shown in Figure 2.13.

2.8.2 Geometry of the Smoothing Circular Segment

In this section the coordinates of the center and the radius of the circular segment are derived. Reference is made to Figure 2.14. It is noted that the elastic stiffness in the normalized plane is η while the hardening stiffness is $b\,\eta$. The points A, B, C and D are distinguished. A_x and A_y denotes the x and y coordinate of the center of the circle, respectively, and C is the point of yielding. Points B and D mark the intersection between the circular segment and the elastic and hardening slopes, respectively.

The length $\overline{BC} = \overline{CD}$ is determined by the triangle equality:

$$\frac{\overline{BC}}{\sqrt{1+\eta^2}} = \frac{1-\gamma}{\eta} \quad \Rightarrow \quad \overline{BC} = \overline{CD} = \frac{1-\gamma}{\eta}\sqrt{1+\eta^2} \tag{2.101}$$

Next, the components of the line segment $\overline{BD}$ on the x and y axes are determined by similar triangle

equality considerations, adding contributions from lines $\overline{BC}$ and $\overline{CD}$:

$$\begin{aligned} \Delta x_{BD} &= \frac{1-\gamma}{\eta} + \frac{\overline{BC}\, b\, \eta}{b\, \eta \sqrt{1+(b\eta)^2}} \\ \Delta y_{BD} &= (1-\gamma) + \frac{\overline{BC}\, b\, \eta}{\sqrt{1+(b\eta)^2}} \end{aligned} \tag{2.102}$$

Equations defining line segments $\overline{AB}$ and $\overline{AD}$ are:

$$y_{AB} = -\frac{x}{\eta} + \gamma + \frac{\gamma}{\eta^2} \tag{2.103}$$

$$y_{AD} = -\frac{x}{b\eta} + \gamma + \Delta y_{B-D} + \frac{1}{b\eta}\left(\frac{\gamma}{\eta} + \Delta x_{B-D}\right) \tag{2.104}$$

The point at which lines $\overline{AB}$ and $\overline{AD}$ intersect defines the center of the circular segment. Its coordinates are:

$$A_x = \frac{\Delta y_{BD} + \frac{1}{b\eta}\left(\frac{\gamma}{\eta} + \Delta x_{BD}\right) - \frac{\gamma}{\eta^2}}{\left(\frac{1}{b\eta} - \frac{1}{\eta}\right)} \tag{2.105}$$

$$A_y = -\frac{1}{\eta}\, A_x + \gamma\left(1 + \frac{1}{\eta^2}\right) \tag{2.106}$$

The equation for the circle reads $(x - A_x)^2 + (y - A_y)^2 = R^2$, where R is the radius. Since the point $\left(\frac{\gamma}{\eta},\ \gamma\right)$ is located on the circle, R is given by:

$$R = \sqrt{(\gamma - A_y)^2 + \left(\frac{\gamma}{\eta} - A_x\right)^2} \tag{2.107}$$

In conclusion, the equation for the circle on the tension side is:

$$y = \sqrt{R^2 - (x - A_x)^2} + A_y \tag{2.108}$$

while on the compression side it is:

$$y = -\sqrt{R^2 - (x + A_x)^2} - A_y \tag{2.109}$$

The transformation into the strain-stress plane is performed by the rules: $x = \frac{\epsilon}{\sigma_y}\frac{E}{\eta}$ and $y = \frac{\sigma}{\sigma_y}$. In implementation, the current strain is first transformed into its corresponding x value before the above parameters of the circular segment are evaluated. The resulting y value is then multiplied by σ_y to obtain the returned stress value.

2.8.3 Updating of Circular Segments

The coordinates of the smoothing circle determined in the previous section are applicable to the first monotonic loading. To accommodate hysteretic behavior they must be updated at each material state determination. Referring to the regions defined in Figure 2.14, four situations are distinguished:

1. Loading/unloading in the elastic region
2. Loading along the circle in the inelastic region in a) tension or b) compression
3. Elastic unloading in the inelastic region in a) tension or b) compression
4. Loading along the $b\,E$ slope in a) tension or b) compression

When situation 1 is encountered then the circular segments in tension and compression are determined so that they align with the elastic loading/unloading line and the second slope. The amount of shift compared to the original circular segment parameters is derived by considering Figure 2.15. The strain at point F is simply $\epsilon_F = \frac{\sigma_y}{E}$, while the strain at point G is determined by line intersection as:

$$\epsilon_G = \frac{\sigma_y}{E} + \frac{\epsilon_{i+1}E - \sigma_{i+1}}{E\,(1-b)} \tag{2.110}$$

The shift of the circular segment along the strain axis is $\epsilon_G - \epsilon_F = \frac{\epsilon_{i+1}E - \sigma_{i+1}}{E(1-b)}$. This quantity is transformed into the $x-y$ plane where it is named Δx. The corresponding shift along the y axis is $b\eta\Delta x$.

When situation 2a is encountered, the parameters of the circular segment on the tension side remain unchanged while the parameters of the circular segment on the compression side are updated by the equations developed for situation 1, and visa versa for situation 2b.

Similarly, when situation 4a is encountered, the parameters of the circular segment on the tension side remain unchanged, while the parameters of the circular segment on the compression side are updated by the equations developed for situation 1, and visa versa for situation 4b.

When situation 3a is encountered, i.e., elastic unloading in tension, then the parameters of the circular segment on the compression side are updated by the equations developed for situation 1. The same is the case for the tension side when situation 3b, i.e., elastic unloading in compression, is encountered.

The two remaining cases are concerned with elastic unloading in the inelastic region. For such a case, the parameters of the circular segment need to be updated for a possible subsequent loading. If the unloading stops in the inelastic region and is followed by loading then this must instantaneously

occur along a circular segment. With reference to Figure 2.16, the challenge is to determine the coordinates of point K, which is the tangent point on the hardening slope, when point H is the current material state. The coordinates of point K enable us to determine the coordinates of the new center of the circle. The coordinates of point J are found to be:

$$J_x = \frac{H_y + \frac{H_x}{b\eta} - (1-b)}{b\eta + \frac{1}{b\eta}} \tag{2.111}$$

$$J_y = b\eta J_x + (1-b) \tag{2.112}$$

The distance between points H and J is $\overline{HJ} = \sqrt{(H_x - J_x)^2 + (H_y - J_y)^2}$ and the distance between points J and K is $\overline{JK} = \sqrt{R^2 - \left(\overline{HJ} - R\right)^2}$, where R is the radius of the circle. Components of vector JK on the x and y axes are determined by triangle equality considerations:

$$\Delta x_{JK} = \frac{\overline{JK}}{\sqrt{1 + (b\eta)^2}} \tag{2.113}$$

$$\Delta y_{JK} = \frac{b\eta\,\overline{JK}}{\sqrt{1 + (b\eta)^2}} \tag{2.114}$$

The components of the vector AK on the two axes are obtained by use of triangle equalities:

$$\Delta x_{AK} = \frac{R}{\sqrt{1 + \left(\frac{1}{b\eta}\right)^2}} \tag{2.115}$$

$$\Delta y_{AK} = \frac{R}{b\eta\sqrt{1 + \left(\frac{1}{b\eta}\right)^2}} \tag{2.116}$$

In terms of the above defined quantities, the updated coordinates of the center of the circle in situation 3a read:

$$A_x = J_x + \Delta x_{JK} + \Delta x_{AK} \tag{2.117}$$

$$A_y = J_y + \Delta y_{JK} + \Delta y_{AK} \tag{2.118}$$

The coordinates of the center of the circle for the compression side in situation 3b are found analogously.

2.8.4 Incremental Response Equations

In the implementation of the smoothed bi-linear material model in OpenSees, the initial parameters Δx_{BD}, Δy_{BD}, A_x, A_y and R of the circular segment are pre-computed in the constructor of the material object according to the equations in the previous sections. The detailed incremental response algorithm used in subsequent state determinations is presented in Section B.1 in Appendix A.4.

2.8.5 Conditional Stress Sensitivity Equations

The incremental response equations are differentiated to obtain conditional derivatives of the stress with respect to the material parameters E, σ_y, b, γ or η for fixed strain. The resulting equations are presented in detail in Section B.2 in Appendix A.4.

2.8.6 Unconditional Sensitivity History Variables

Detailed equations used to compute and store sensitivity history variables in "phase 2" of the sensitivity computations for the smoothed bi-linear material model are derived and presented in Section B.3 of Appendix A.4.

2.8.7 Example Results

The same examples as in Section 2.6.5 are used to examine the sensitivity results obtained by the smoothed bi-linear material model. Of interest is to compare the sensitivity results obtained with the smoothed and the non-smoothed models. For the example of a single material point being subjected to the stress time-history in Eq. (2.98), the following material parameters are selected: $\sigma_y = 400.0\, N/mm^2$, $E = 210,000.0\, N/mm^2$, $b = 0.05$, $\eta = 4.0$ and $\gamma = \{0.25,\ 0.5,\ 0.6,\ 0.7,\ 0.8,\ 0.9\}$. Several values of γ are analyzed to investigate the effect of different degrees of smoothing.

The stress-strain curves for all cases are shown in Figure 2.17. It is observed that the discrepancy between the models for this example is located around the point of yielding. As expected, a value of γ closer to 1 leads to a response estimate that is closer to the bi-linear case. Figures 2.18, 2.19 and 2.20 present the corresponding strain sensitivity results. Again it is observed that increasing γ brings the results closer to those of the bi-linear model. Furthermore, it is seen in Figure 2.18 that the smoothed model avoids the gradient discontinuities evident in the predictions by the bi-linear model. In all strain sensitivity results obtained by the smoothed model, discontinuities are present neither in loading nor in unloading phases.

Next, we consider the 3-D truss structure that was investigated in Section 2.6.5. The model data specified in Section 5.2 are used, with the exception of varying the degree of smoothing: $\gamma =$

$\{0.25,\ 0.5,\ 0.6,\ 0.7,\ 0.8,\ 0.9\}$. The applied load series is:

$$\begin{array}{lccccc} \text{Time:} & 0.0 & 1.0 & 2.0 & 3.0 & 4.0 \\ \text{Load:} & 0.0 & 430\,kN & 0.0 & -430\,kN & 0.0 \end{array} \tag{2.119}$$

Load-displacement curves for different degrees of smoothing are shown in Figure 2.21. The response is sampled at the top of the truss. It is observed that the discrepancy between the response of the bi-linear model and the smoothed model with low γ value is significant. This is because the states of many members in the structure fall in the region close to the yield point. This illuminates the need for caution when selecting γ. If the bi-linear model is believed to best model reality, then a γ higher than, say, 0.7 or 0.8 must be selected for cases when the member states are close to the bi-linear yield point.

Four cases are considered to investigate the effect of smoothing on sensitivity results. Figure 2.22 shows the sensitivity of the top displacement of the truss with respect to the yield stress of the element that first reaches yielding, namely element number 4. (Figure 5.7 on page 197 provides element and node numbers.) The bi-linear model is seen to exhibit sensitivity discontinuities at instances when elements enter yielding. These discontinuities are avoided with the smoothed model. As expected, the smoothing effect increases with decreasing γ. It is interesting to observe a strong change in the sensitivity result when γ is taken to be 0.7 or less. Figures 2.23 and 2.24 show the sensitivities of the x and y-direction displacements at mid-height of the truss with respect to the yield stress of element 4, respectively. Figure 2.25 shows the sensitivity of the axial force of element number 4 with respect to its yield stress. In all results it is observed that the smoothed material model avoids discontinuity in the sensitivity results. It is also seen that low γ values lead to significant discrepancy between sensitivity results of the bi-linear model and the smoothed model. According to this example, $\gamma \geq 0.8$ can be recommended to avoid results that differ significantly from those obtained with the bi-linear model.

2.9 UNIAXIAL SMOOTHED CONCRETE MATERIAL

In this section, smoothing techniques are applied to the "Concrete01" material model in OpenSees. This is a model characterized by by a modified Kent-Park backbone curve (Scott *et al.* 1982), zero stress in tension and linear unloading/reloading. The kinks in the backbone curve are smoothed by polynomial functions. This procedure is also applied to the transitions between the linear unloading/reloading and the backbone curve/zero-stress axis.

2.9.1 Fundamental Model Assumptions

The backbone curve of the "Concrete01" model in OpenSees is shown in Figure 2.26. It consists of a parabola and two straight lines. Unloading and subsequent loading occur along the dotted lines. The original model in OpenSees is described by four user-defined material parameters, namely f'_c, f'_{cu}, ϵ_{c0} and ϵ_{cu}, as shown in Figure 2.26. The parameters ϵ_{min} and ϵ_{end} serve as history variables. They mark the strain at the two end points of the linear unloading/reloading line currently in effect. For convenience, a third history variable k_u, namely the slope of the unloading/reloading line, is stored in the original OpenSees implementation. The rules according to which the unloading/reloading line is updated are provided in the following section.

In this study, the backbone curve is smoothed by a third-order polynomial between points 1 and 2 in Figure 2.26. The tangent of the polynomial at point 1 coincides with the tangent of the parabola of the original model. The tangent of the polynomial at point 2 corresponds to the tangent of the constant-stress line of the original model. An example of the resulting backbone curve is shown in Figure 2.27 and detailed equations are provided in the following subsection.

For the purpose of smoothing the unloading-reloading line, two user-selected parameters $0 < \gamma < 0.5$ and $0 < \eta$ are introduced. These are explained with reference to Figure 2.28. In this figure, ϵ_{min2} and ϵ_{end2} are the points of intersection between the original unloading/reloading line and the smoothing curves. ϵ_{min3} and ϵ_{end3} identify the end points of the smoothing curves. γ denotes the distance from ϵ_{min} to ϵ_{min2} and from ϵ_{end} to ϵ_{end2} as a fraction of the distance between ϵ_{end} and ϵ_{min}. Similarly, η denotes the distance from ϵ_{min} to ϵ_{min3} and from ϵ_{end} to ϵ_{end3} as a fraction of the distance between ϵ_{end} and ϵ_{min}. A parabola with matching tangents at the end points is then used to connect the points. Figure 2.29 shows an example stress-strain curve for the smoothed material as it is implemented in OpenSees. An arbitrary sequence of loading/unloading events is applied. A stress-strain curve for the original material model subjected to a similar load series is shown in Figure 2.30.

2.9.2 Incremental Response Equations

The algorithmic implementation in OpenSees is now presented. First, the strain measures ϵ_{end2}, ϵ_{end3}, ϵ_{min2} and ϵ_{min3} are computed by using the current value of the history variables:

$$\epsilon_{end2} \quad = \quad \epsilon_{end} + \gamma \, (\epsilon_{min} - \epsilon_{end})$$

$$
\begin{aligned}
\epsilon_{end3} &= \epsilon_{end} - \eta\ (\epsilon_{min} - \epsilon_{end}) \\
\epsilon_{min2} &= \epsilon_{min} - \gamma\ (\epsilon_{min} - \epsilon_{end}) \\
\epsilon_{min3} &= \epsilon_{min} + \eta\ (\epsilon_{min} - \epsilon_{end})
\end{aligned}
$$

The history variables are zero prior to the first state determination, which is described in the following. For a given strain increment $\Delta\epsilon = \epsilon_{i+1} - \epsilon_i$, the stress and the algorithmically consistent tangent are computed as follows:

1. **if** $\Delta\epsilon < 0$, i.e., applying compression to the material:

   (a) **if** $\epsilon_{i+1} < \epsilon_{min3}$, i.e., loading along the backbone curve.

      i. Compute the stress according to the algorithm presented in Section B.5 in Appendix A.4.

      ii. Compute and save history variables according to the algorithm presented in Section B.4 in Appendix A.4.

   (b) **else if** $\epsilon_{i+1} < \epsilon_{min2}$, i.e., loading along the backbone curve or reloading on a smoothing line between ϵ_{min2} and ϵ_{min3}. The choice between these two states is determined by examining the material state at the previous step. If the previous step is not on the backbone curve, then reloading along the smoothed curve is the case and the stress is computed according to Section B.6. If the previous step was unloading in the region between ϵ_{min2} and ϵ_{min3} then the smoothing line is updated so that its end points are the current stress-strain state and the ϵ_{min3}-point. If loading along the backbone curve is the case then the history variables are updated according to Section B.4 in Appendix A.4.

   (c) **else if** $\epsilon_{i+1} < \epsilon_{end2}$, i.e., reloading on a straight line:

      i. $\sigma_{i+1} = k_u\ (\epsilon_{i+1} - \epsilon_{end})$

      ii. $k_T = k_u$

   (d) **else if** $\epsilon_{i+1} < \epsilon_{end3}$, i.e., reloading on a smoothing line between ϵ_{end2} and ϵ_{end3}. The stress is computed according to Section B.6 in Appendix A.4.

   (e) **else**, reloading in the zero stress tension region:

      i. $\sigma_{i+1} = 0$

      ii. $k_T = 0$

2. **else**, strain increment in direction of tension:

(a) **if** previous material state involved loading along the smoothing line between ϵ_{min2} and ϵ_{min3} then the history variables are updated according to Section B.4 in Appendix A.4 using the previous stress-strain state. This is done by first determining ϵ_{min}, which is at the point of intersection between the backbone curve and the line $\sigma_i + (\epsilon - \epsilon_i)\, k_u$. Since part of the backbone curve is a third-order polynomial, a Newton algorithm is implemented for this purpose in OpenSees.

(b) **if** $\epsilon_{i+1} < \epsilon_{min2}$, i.e., linear elastic unloading:

i. $\sigma_{i+1} = k_u\ (\epsilon_{i+1} - \epsilon_{end})$

ii. $k_T = k_u$

(c) **else if** $\epsilon_{i+1} < \epsilon_{end3}$, i.e., unloading along a smooth line between ϵ_{end2} and ϵ_{end3} according to Section B.6 in Appendix A.4.

(d) **else**, unloading all the way down to zero stress:

i. $\sigma_{i+1} = 0$

ii. $k_T = 0$

2.9.3 Conditional Stress Sensitivity Equations

Derivation of the conditional stress sensitivity equations involves differentiation of the response algorithm presented above. As seen, the stress was determined by three algorithms, depending on the material state: The state is on the backbone curve, on the smoothed unloading/reloading line, or on a linear elastic unloading line. The analysis in this section involves differentiation of these stress states. The last stress state is the simplest, where the stress is computed by $\sigma_{i+1} = k_u\ (\epsilon_{i+1} - \epsilon_{end})$ and the conditional stress derivative simply reads: $\frac{\partial \sigma_{i+1}}{\partial h}|_{\epsilon_{i+1}\ fixed} = \frac{\partial k_u}{\partial h}\ (\epsilon_{i+1} - \epsilon_{end}) - k_u \frac{\partial \epsilon_{end}}{\partial h}$. For the other two cases the stress derivative is found by differentiation of the algorithms in Sections B.5 and B.6 in Appendix A.4. This is done in Sections B.7 and B.8 of Appendix A.4, respectively.

2.9.4 Unconditional Sensitivity History Variables

In the previous subsection $\frac{\partial \epsilon_{min}}{\partial h}$, $\frac{\partial \epsilon_{end}}{\partial h}$ and $\frac{\partial k_u}{\partial h}$ are sensitivity history variables. Section B.9 in Appendix A.4 contains the detailed algorithm to compute these. It is noted that this algorithm is

not called at each step; only when ordinary history variables are committed. Additional sensitivity history variables are present. These include the derivatives of the points of stress and strain, between which smoothing lines are drawn, together with the derivative of the corresponding stiffness. These are computed in Sections B.7 and B.8 in Appendix A.4.

2.10 DISCONTINUITIES IN SENSITIVITY RESULTS

In finite element reliability analysis, response sensitivities are used in the search for the so-called design point. Most search algorithms used for this purpose assume that the response sensitivities are continuous. Violation of this assumption may lead to difficulties in or failure to converge to the design point. However, discontinuities in the response sensitivity occur even for commonly used material models, unless proper measures are taken. For instance, the J_2 plasticity model exhibits discontinuities in the displacement sensitivity with respect to the yield stress along the pseudo-time axis for static problems. This is observed, for instance, in Figure 2.8. The issue of how such discontinuities can be remedied is the topic of this section.

A source of discontinuity in the response sensitivity is the computational "noise" in the finite element response due to numerical approximations. This may occur, for example, when too large a time step $\Delta t = t_{n+1} - t_n$ is used in a dynamic analysis. However, the sensitivity discontinuities caused by numerical noise is usually of a much smaller scale than the discontinuities stemming from the material. The standard approach to handle the discontinuities arising from noise is proper selection of approximation parameters and tolerances.

It is emphasized that it is discontinuities in the parameter space which are of concern; the search for the design point is performed in the space of these parameters when they are considered as random variables. Denoting the response sensitivity as $\nabla G(h)$, the requirement for the continuity of the sensitivity reads:

$$\lim_{dh \to 0} \left(\nabla G(h \pm dh) - \nabla G(h) \right) = 0 \tag{2.120}$$

Continuity is proven if, for any small $\epsilon > 0$, we can select a finite but arbitrarily small dh such that $|\nabla G(h \pm dh) - \nabla G(h)| < \epsilon$. However, it is often more convenient to study sensitivity results along the (pseudo-) time axis. One can argue that these discontinuities are equivalent, since discontinuities that appear along the time axis may also appear in the parameter space. This is exemplified in Figure 2.31. A static single degree-of-freedom (DOF) system with bi-linear material behavior is

subjected to loading followed by unloading. It is seen that discontinuity in the displacement sensitivity may occur along two lines; namely, along the thick line, where yielding occurs, or along the line where unloading starts. It is observed that both these cases may be investigated by studying the displacement sensitivity along the pseudo-time axis for a prescribed yield stress. Hence, the first part of this section will discuss discontinuities along this axis (or the real time axis for dynamic problems).

The static case is considered first. Only the load-control analysis procedure is considered. This is the most commonly used analysis scheme for static pushover analysis. Furthermore, the response sensitivity for alternative analysis schemes may involve additional considerations. For instance, consider the displacement-control procedure where the displacement at one (or more) control DOF's are prescribed at each pseudo-time step. The sensitivity derivations earlier in this chapter aim at computing the sensitivity of the response by differentiating the governing equilibrium equation $P^{int} = P^{ext}$ at a single time step. It is clear that the displacement sensitivity at the control DOF is zero. The force sensitivity, on the other hand, is not. In the analyses considered in this work the structures are considered to deform under applied load, not applied displacement.

In the numerical studies presented earlier in this study, gradient discontinuities at points of sudden elastic unloading were not observed. To discuss and explain this finding we consider Eq. (2.36). Assuming that h represents a structural property (not an external load parameter), Eq. (2.36) reduces to:

$$\frac{\partial u_{p\,(n+1)}}{\partial h} = K^{-1}_{qp\,(n+1)} \left(-\frac{\partial P^{int}_{q\,(n+1)}}{\partial h}\Big|_{u_{p\,(n+1)}\,fixed} \right) \tag{2.121}$$

The issue of continuity in the sensitivity result over the event of a sudden elastic unloading at t_n is first addressed. In such an event the displacement vector that provides equilibrium is found by using the following linear expression for the internal force vector:

$$P^{int}_{q\,(n+1)} = P^{int}_{q\,(n)} + K^{o}_{qp} \left(u_{p\,(n+1)} - u_{p\,(n)} \right) \tag{2.122}$$

where K^o_{qp} is the elastic stiffness matrix. Noting that $K_{qp\,(n+1)} = K^o_{qp}$, substitution of Eq. (2.122) into Eq. (2.121) leads to:

$$\frac{\partial u_{p\,(n+1)}}{\partial h} = -\left(K^o_{qp}\right)^{-1} \left(\frac{\partial P^{int}_{q\,(n)}}{\partial h} + \frac{\partial K^o_{qp}}{\partial h} \left(u_{p\,(n+1)} - u_{p\,(n)} \right) - K^o_{qp} \frac{\partial u_{p\,(n)}}{\partial h} \right) \tag{2.123}$$

By rearranging, we obtain the following expression for the difference between the displacement sensitivities at the two time steps:

$$\frac{\partial u_{p\,(n+1)}}{\partial h} - \frac{\partial u_{p\,(n)}}{\partial h} = -\left(K^o_{qp}\right)^{-1} \left(\frac{\partial P^{int}_{q\,(n)}}{\partial h} + \frac{\partial K^o_{qp}}{\partial h} \left(u_{p\,(n+1)} - u_{p\,(n)} \right) \right) \tag{2.124}$$

To prove continuity, we must prove that the two terms on the right-hand side of Eq. (2.124) approach zero as $(t_{n+1} - t_n) \to 0$. We start by studying the quantity $\frac{\partial P^{int}_{q\,(n)}}{\partial h}$. We prove that this unconditional derivative is zero by using the governing equilibrium equation $P^{int}_{q\,(n)} = P^{ext}_{q\,(n)}$. As long as h does not enter the external force vector, it is clear that $\frac{\partial P^{int}_{q\,(n)}}{\partial h} = 0$. Next, by assuming continuity of the displacement response itself, it is clear that:

$$\lim_{(t_{n+1}-t_n)\to 0} \left(u_{p\,(n+1)} - u_{p\,(n)} \right) = 0 \tag{2.125}$$

Hence, we conclude from Eq. (2.124):

$$\lim_{(t_{n+1}-t_n)\to 0} \left(\frac{\partial u_{p\,(n+1)}}{\partial h} - \frac{\partial u_{p\,(n)}}{\partial h} \right) = 0 \tag{2.126}$$

This proves continuity of the displacement sensitivity result at the event of sudden elastic unloading.

Even though $\frac{\partial P^{int}_{q\,(n)}}{\partial h} = 0$, we cannot conclude that the sensitivities of individual element forces are zero. On the contrary, considering $P^{int}_{q\,(n)}$ itself, we recall that the common situation is that the sum of the forces along a degree-of-freedom is zero while the individual element forces of the adjacent elements are not.

Now turning to the case of discontinuities due to material state changes during loading, Eq. (2.122) is no longer generally valid. For instance, a return mapping algorithm must be employed in place of the linear step for classical plasticity materials. Hence, we are not able to generally prove continuity of response sensitivity results. Indeed, numerical results indicate discontinuities as previously mentioned. For specific constitutive models one may analytically predict when discontinuities will appear. For instance, it is observed that the sensitivity results for a single DOF system with a bilinear material model are continuous for all parameters except the decision parameter σ_y. The term "decision parameter" is used to describe parameters which to not determine the stiffness/flexibility of a constitutive model but are used to determine the material state changes, e.g., yielding. The only remedy to avoid such discontinuities is to make use of smooth material models, such that the transition between material states occur in a gradual rather than sudden manner. It is easily proven, by a finite-difference type formulation, that discontinuities in displacement sensitivity results cannot appear when the displacement response itself does not have "kinks." Hence, according to the above derivations smooth material models will not exhibit gradient discontinuities. For this reason a number of such material models have been presented in this chapter. Numerical results confirm the conclusion made here.

In finite element analysis of dynamic problems all terms of Eq. (2.34) must be considered. The question is whether these additional terms will introduce discontinuities. No proofs are presented

here. However, it is generally observed that the displacement response from a dynamic analysis is smooth due to inertia effects. Again, based on a finite difference type formulation, one can argue that discontinuities in the displacement sensitivity results cannot appear when the displacement response itself does not have "kinks."

2.11 APPLICABILITY OF THE ADJOINT METHOD

In finite element reliability applications the gradient of the performance rather than of the constituent response quantities is needed. Strictly implementing Eq. (2.34) implies that the sensitivity of each response quantity is computed prior to computing the gradient of the performance function itself. The adjoint method (Arora and Haug 1979) is designed to make sensitivity computations more efficient in such cases.

Consider the linear static case. If h represents a material parameter, the displacement sensitivity is obtained according to Eq. (2.36) as:

$$v_p = \frac{\partial u_p}{\partial h} = \left(K^o_{qp}\right)^{-1} \left(\frac{\partial P^{ext}_q}{\partial h} - \frac{\partial K^o_{qp}}{\partial h} u_p \right) \tag{2.127}$$

Assuming that only displacement response quantities enter the performance function g, its gradient reads:

$$\frac{\partial g}{\partial h} = \frac{\partial g}{\partial u_p} \, \frac{\partial u_p}{\partial h} \tag{2.128}$$

Substitution of Eq. (2.127) into Eq. (2.128) leads to:

$$\frac{\partial g}{\partial h} = \underbrace{\frac{\partial g}{\partial u_p} \, K^{-1}_{qp}}_{\lambda_q} \left(\frac{\partial P^{ext}_q}{\partial h} - \frac{\partial K_{qp}}{\partial h} u_p \right) \tag{2.129}$$

where $\lambda_q = \frac{\partial g}{\partial u_p} \, K^{-1}_{qp}$ is defined. The adjoint method for finding the gradient of g now proceeds as follows:

1. Solve for λ_q from $K_{qp} \, \lambda_q = \frac{\partial g}{\partial u_p}$. Note that $\frac{\partial g}{\partial u_p}$ is easily computed since g is usually a simple algebraic expression of u.

2. For each parameter h:

   (a) Assemble the right-hand side $\left(\frac{\partial P^{ext}_q}{\partial h} - \frac{\partial K_{qp}}{\partial h} u_p \right)$

(b) Find the gradient of the performance function for each h by a vector-matrix product: $\frac{\partial g}{\partial h} = \lambda_q \left(\frac{\partial P_q^{ext}}{\partial h} - \frac{\partial K_{qp}}{\partial h} u_p \right)$

The advantage of the adjoint method is that the back-substitution required in solving for the constituent $\frac{\partial u_p}{\partial h}$ of Eq. (2.128) for each h is replaced by a one-time back-substitution and a subsequent vector-matrix products for each h.

The adjoint method avoids computing the displacement sensitivities. In analyses where these are needed, such as inelastic or dynamic problems, no benefit is gained from use of the method. Such problems are the focus of this study. Hence, although the adjoint method can be applied to nonlinear and dynamic problems that are not path-dependent (Kleiber *et al.* 1997), the method has not been implemented in OpenSees because of its limitations for inelastic problems.

2.12 IMPLEMENTATIONS IN OPENSEES

A characteristic of the DDM is that the core finite element procedure needs to be modified. The present work includes such developments in OpenSees. While detailed class interfaces are provided in Appendix 6.2, this section aims at explaining the overall principles of the framework for implementation of the response sensitivities in OpenSees.

At the top level, two new classes are implemented in OpenSees to orchestrate the sensitivity computations. The "sensitivity algorithm" loops over all the parameters with respect to which sensitivities are desired. The "sensitivity integrator" is responsible for assembling the right-hand side of the sensitivity equation. At each step of an inelastic static analysis or a dynamic analysis, the following operations are performed to obtain response sensitivities:

1. Convergence is achieved for the finite element response itself.

2. Before the finite element code commits history variables for the converged state, the sensitivity algorithm is called to compute response sensitivities.

3. The sensitivity algorithm starts by making sure that the tangent matrix is updated (in case a scheme other than the Newton-Raphson is used). If it is not updated, an update is initiated.

4. The sensitivity algorithm then calls the sensitivity integrator to form the part of the right-hand side, which is independent of the sensitivity parameters. This is only relevant for dynamic problems, see Eq. (2.34).

5. The sensitivity algorithm then loops over all parameters, performing the following operations:

   (a) The current parameter is made "active." This is done so that all elements, sections and material objects know what to return. For example, a material object knows if one of its parameters is tagged as active. As noted in Section 2.2.2, inelastic materials generally return non-zero conditional stress derivative, even if the parameter in question does not belong to it.

   (b) The sensitivity integrator is called to form the right-hand side of the sensitivity equation.

   (c) The system of equations is solved for the displacement sensitivity vector.

   (d) Sensitivity results are saved for later recovery by the user or the reliability algorithm.

   (e) The sensitivity integrator is called to store sensitivity history variables. The displacement sensitivity vector is passed down and given as strain sensitivity to the inelastic material objects.

For path-independent static problems, it is sufficient to compute response sensitivities only at the last step of the analysis. No history variables need to be committed. In that case, the call to the sensitivity algorithm can be made after convergence of the very last step. A special command for this case is implemented in OpenSees.

2.12.1 Forming the Right-Hand Side for Static Problems

In the above procedure it is seen that the sensitivity integrator is responsible for forming the right-hand side of Eq. (2.34). Two types of sensitivity integrators are implemented in OpenSees. One for static problems and one for dynamic problems. This section describes how the static version interacts with the finite element code to assemble the needed contributions.

Two terms in Eq. (2.34) contribute in the static case; namely, $\frac{\partial P^{int}_{q\ (n+1)}}{\partial h}\Big|_{\substack{u_{p\ (n+1)}\ fixed \\ \dot{u}_{p\ (n+1)}\ fixed}}$ and $\frac{\partial P^{ext}_{q\ (n+1)}}{\partial h}$. The latter term is simpler. It is assembled by calling all load patterns. Each load pattern returns a vector, which contains a list of node numbers and degrees-of-freedom represented by h. See Appendix 6.2 for detailed additions to the class interfaces of load patterns and nodal load classes.

Assembly of the derivative of the internal force vector involves a more elaborate procedure. To understand it, it is necessary to be familiar with the interaction between the `FE_Element` class in OpenSees and the integrator class (McKenna 1997). In short, `FE_Element` is an "umbrella" class,

not base class, for all finite elements in OpenSees. During the program execution each `FE_Element` contains one finite element. Its usefulness may be seen in its interaction with the integrator class, particularly in dynamic analysis (see below). In the context of static sensitivity analysis, the following chain of calls is performed:

1. The sensitivity integrator loops over all `FE_Element`'s, calling the `getResidual` member function.

2. The `FE_Element` then calls the `formEleResidual` member function of the integrator. A pointer to the `FE_Element` itself is passed in the call. The motivation for this call back to the integrator is readily seen in dynamic analysis. The time-stepping parameters are then needed to form the right-hand side. These parameters are encapsulated in the integrator.

3. The integrator now calls the `FE_Element` in question, first for initialization, then for assembly. The assembly member function of the `FE_Element` being called by the sensitivity integrator is `addResistingForceSensitivity`.

4. The `getResistingForceSensitivity` member function of the real element, e.g., a fiber beam-column element, is now called.

5. In turn the section or material level is called to return the sensitivity contributions.

In "phase 2" of the sensitivity computations, the sensitivity integrator simply loops over all `FE_Elements` which in turn call `commitSensitivity` member functions of the elements, sections, and materials. It is noted that the procedure implemented in OpenSees is particularly convenient for inelastic problems.

2.12.2 Forming the Right-Hand Side for Dynamic Problems

The implementation of DDM sensitivity options in OpenSees for dynamic problems raises a strategic question; namely, how the dynamic sensitivity integrator should be related to the ordinary time-stepping integrator. The static sensitivity integrator is simply a sub-class of the general `SensitivityIntegrator`. This is not sufficient for the dynamic case because the dynamic sensitivity integrator needs access to the time-stepping parameters of the ordinary integrator.

In the current version of OpenSees, this is solved in the following way. For each ordinary integrator type (Newmark, HHT, etc.), one dynamic sensitivity integrator is implemented. Each such integrator

is a sub-class of both `SensitivityIntegrator` and of the ordinary integrator. For instance, a sensitivity integrator called `NewmarkSensitivityIntegrator` has been implemented. It is a sub-class of both the class `SensitivityIntegrator` and the class `Newmark`, the latter being the ordinary integrator. Hence, the use of multiple inheritance is used to solve the problem. By making the parameters of the `Newmark` class "protected," the `NewmarkSensitivityIntegrator` has access to them as intended.

All terms in the right-hand side of Eq. (2.34) must be assembled in the dynamic case. Three key member functions are present in the dynamic sensitivity integrator: (1) `formSensitivityRHS` is the top-level function being called by the sensitivity algorithm, (2) `formEleResidual` is called by the `FE_Element` exactly as described for the static case, and (3) `formNodUnbalance` is called by the `DOF_Group` class. The `DOF_Group` class is analogous to the `FE_Element` class, but for the nodes. The operations in each of the three member functions are outlined in an algorithmic form in the following.

The "formSensitivityRHS" member function

1. Loop over all load patterns to assemble contributions if the parameter in question is a nodal load. This is exactly the same as for the static case described above, and accounts for the term $\frac{\partial P^{ext}_{q\ (n+1)}}{\partial h}$ in Eq. (2.34).

2. Loop over all `FE_Element`s as described for the static case. The `FE_Element` immediately calls the `formEleResidual` member function described below, passing a pointer to itself in the call.

3. Loop over all `DOF_Group`s to assemble contributions for the cases when nodal masses are represented by h. The `DOF_Group` immediately calls the `formNodUnbalance` member function described below, passing a pointer to itself in the call.

The "formEleResidual" member function

1. Compute/retrieve parameters a_1, a_2, etc., of Eq. (2.16).

2. Obtain displacement sensitivity vectors ($\frac{\partial \mathbf{u}}{\partial h}$, $\frac{\partial \dot{\mathbf{u}}}{\partial h}$ and $\frac{\partial \ddot{\mathbf{u}}}{\partial h}$) from the previous step.

3. Call the element to obtain the resisting force sensitivity, using the same member functions as for the static case. This accounts for the term $\left.\frac{\partial P^{int}_{q\ (n+1)}}{\partial h}\right|_{\substack{u_{p\ (n+1)}\ fixed \\ \dot{u}_{p\ (n+1)}\ fixed}}$ in Eq. (2.34).

4. Call the element to obtain the sensitivity of the element mass. This accounts for the element-part of the term $\frac{\partial M_{qp}}{\partial h}\ \ddot{u}_{p\ (n+1)}$ in Eq. (2.34).

5. Call the element to obtain the element mass, accounting for the element-part of the term $M_{qp}\left(a_2\, v_{p\,(n+1)} + a_3\, \dot{v}_{p\,(n)} + a_4\, \ddot{v}_{p\,(n)}\right)$.

6. Call the element to obtain the element viscosity, accounting for part of the term $\tilde{C}_{qp\,(n+1)}\left(a_6\, v_{p\,(n)} + a_7\, \dot{v}_{p\,(n)} + a_8\, \ddot{v}_{p\,(n)}\right)$.

7. **if** Rayleigh damping is present:

   (a) Call the element to obtain the mass and stiffness matrices, multiply by the user-defined coefficients from Eq. (2.48), thus accounting for the remainder of the term $\tilde{C}_{qp\,(n+1)}\left(a_6\, v_{p\,(n)} + a_7\, \dot{v}_{p\,(n)} + a_8\, \ddot{v}_{p\,(n)}\right)$.

The "formNodUnbalance" member function

1. Call the node to obtain the sensitivity of the element mass. This accounts for the node part of the term $\frac{\partial M_{qp}}{\partial h}\, \ddot{u}_{p\,(n+1)}$ in Eq. (2.34).

2. Call the node to obtain nodal mass, accounting for the node-part of the term $M_{qp}\left(a_2\, v_{p\,(n+1)} + a_3\, \dot{v}_{p\,(n)} + a_4\, \ddot{v}_{p\,(n)}\right)$.

3. **if** Rayleigh damping is present:

   (a) Call the node to obtain the mass force, multiply by user-defined coefficients from Eq. (2.48), thus accounting for parts of the term $\tilde{C}_{qp\,(n+1)}\left(a_6\, v_{p\,(n)} + a_7\, \dot{v}_{p\,(n)} + a_8\, \ddot{v}_{p\,(n)}\right)$.

4. In the dynamic case, the node is called to add the sensitivity of possible external forces representing the parameter h. This contribution has already been formed by an `applyLoadSensitivity` member function. See item 1 of the `formSensitivityRHS` member function above.

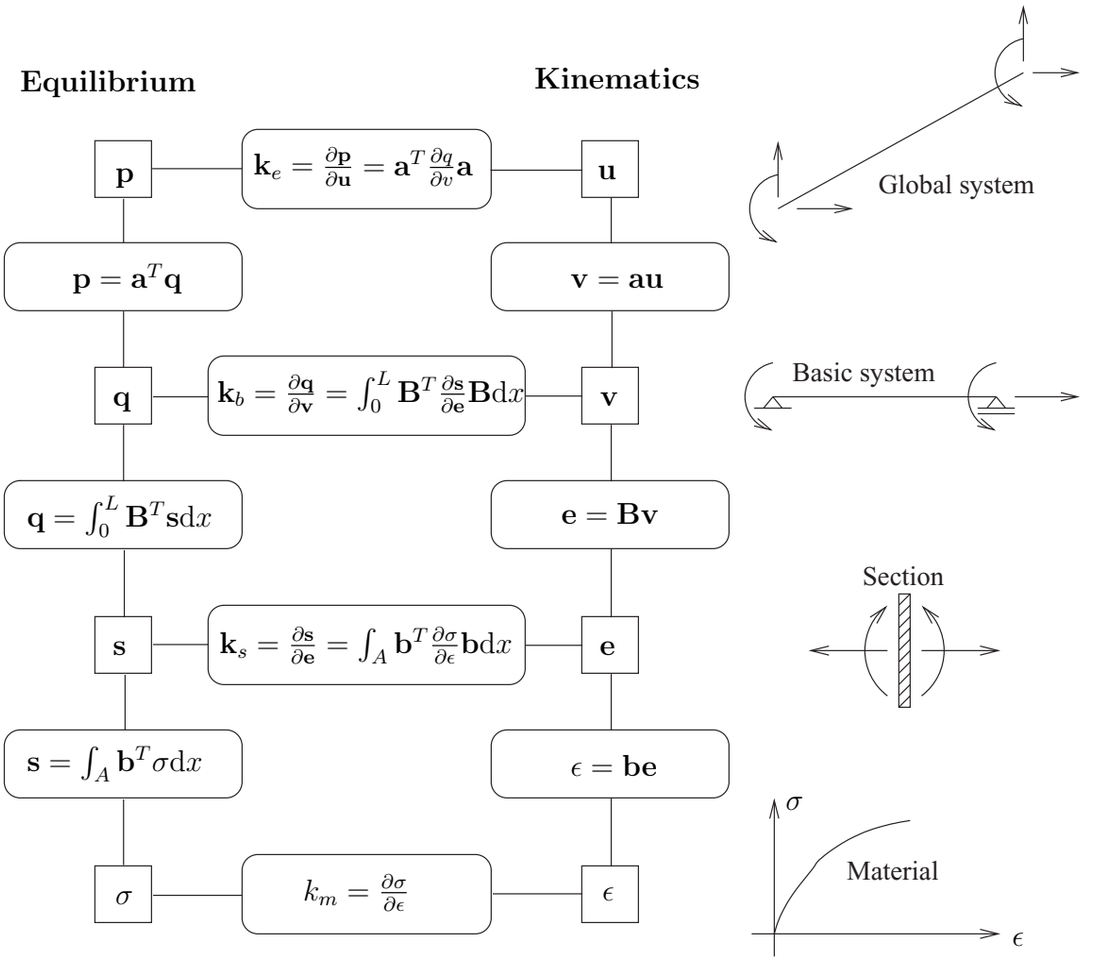

Figure 2.1: Framework of equilibrium and kinematics equations for displacement-based beam-column element.

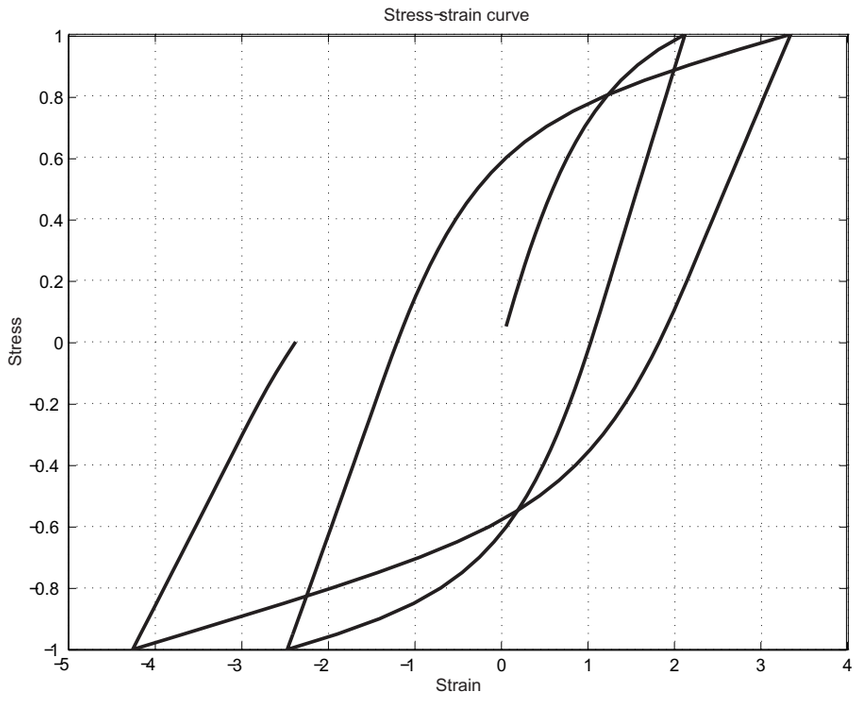

Figure 2.2: Load-displacement curve for one-member truss model with degrading Bouc-Wen material.

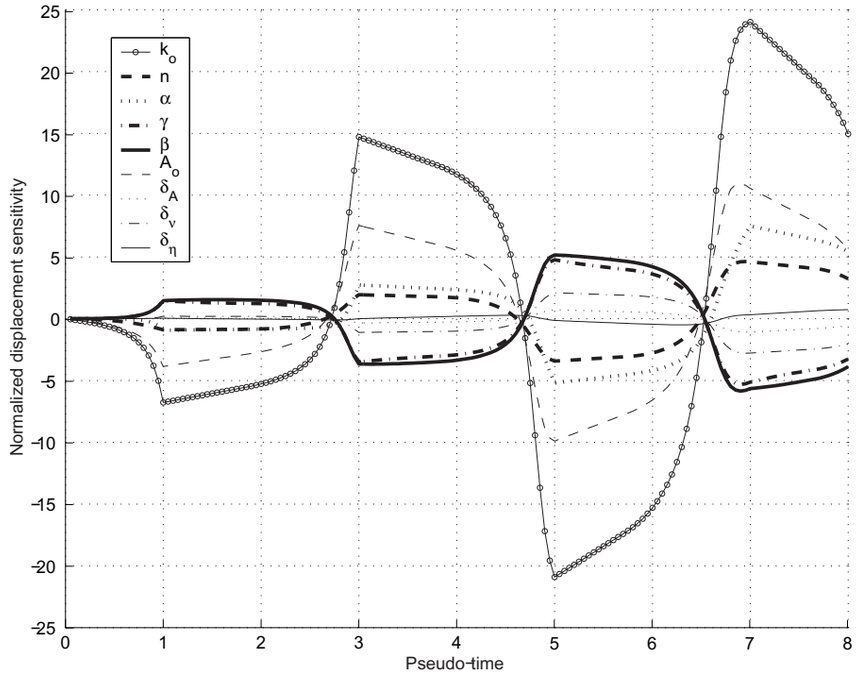

Figure 2.3: Displacement sensitivity results $\frac{\partial u}{\partial h} \cdot h$ for a material point with degrading Bouc-Wen model.

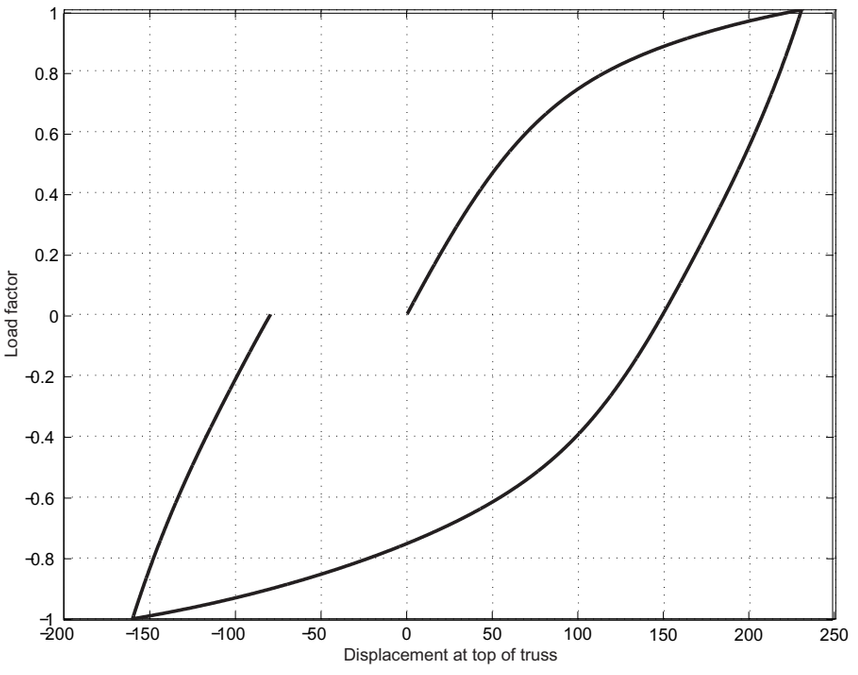

Figure 2.4: Load-displacement curve for 3-D truss structure with Bouc-Wen material.

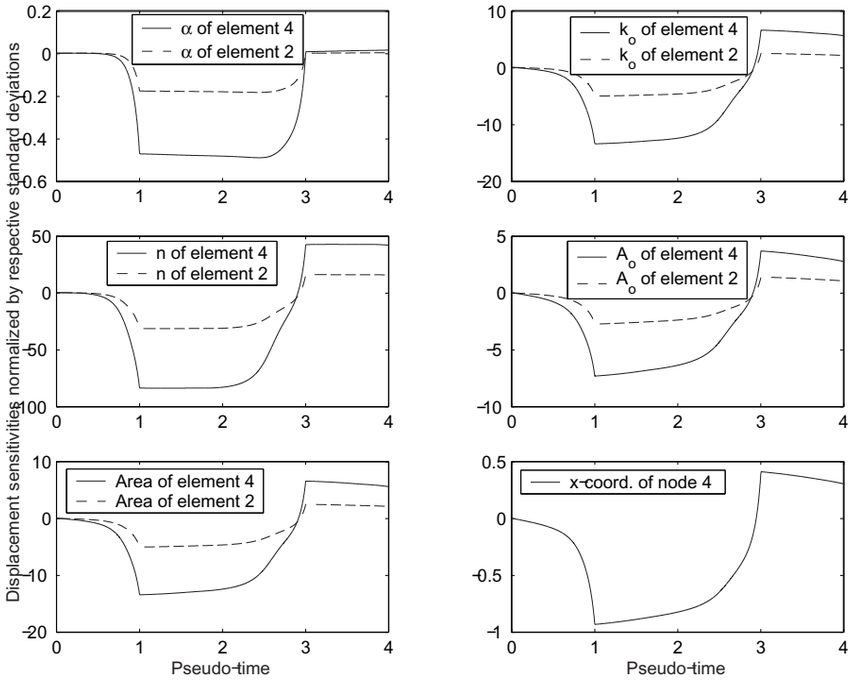

Figure 2.5: Displacement sensitivity results for 3-D truss structure with Bouc-Wen material.

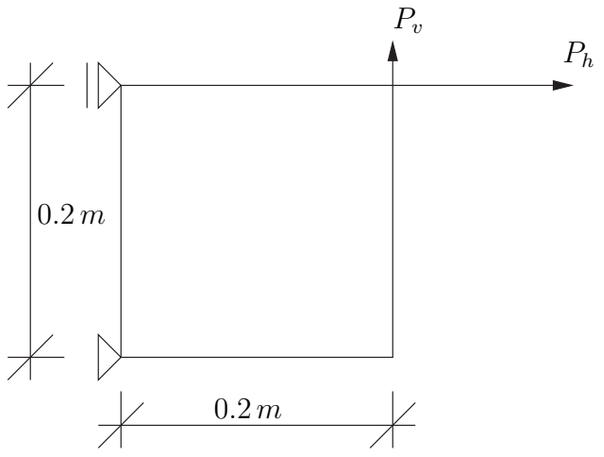

Figure 2.6: Single four-node quad element for demonstration of sensitivity results for J_2 and Generalized Plasticity material models.

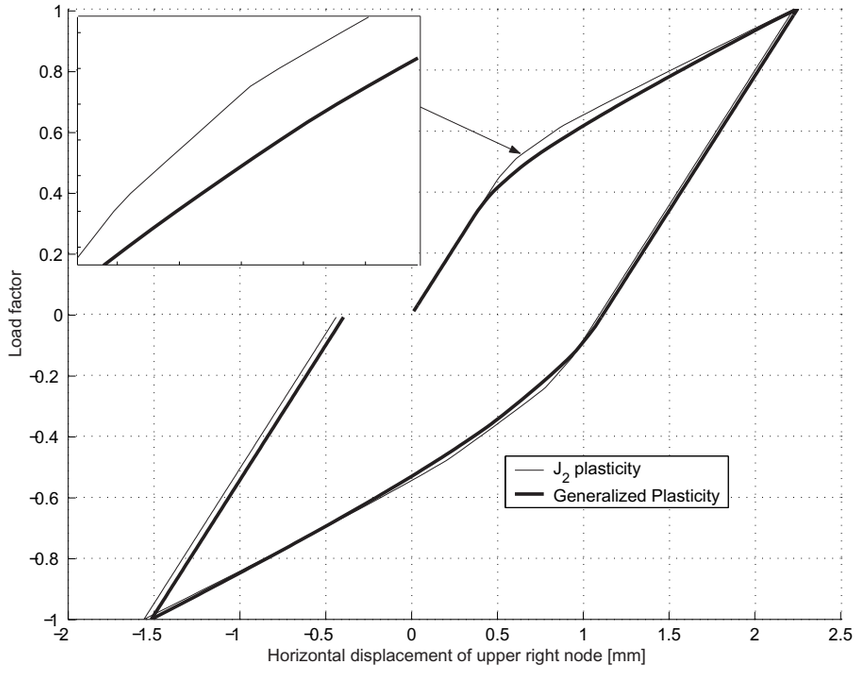

Figure 2.7: Load-displacement curve for one four-node quad element with Generalized Plasticity and J_2 plasticity material.

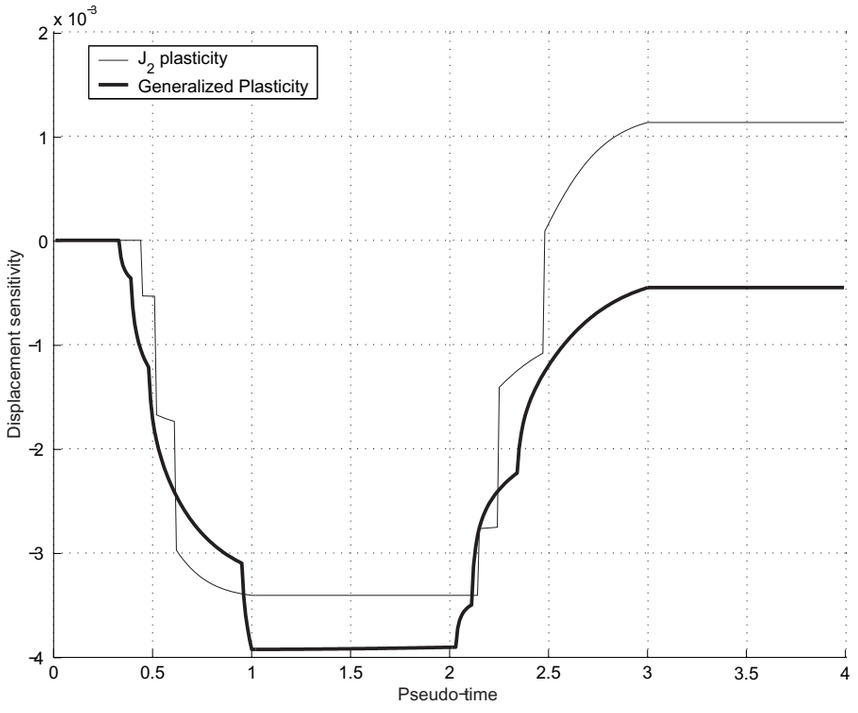

Figure 2.8: Displacement sensitivity results $\frac{\partial u}{\partial \sigma_y}$ for one Quad4 element with two plasticity models.

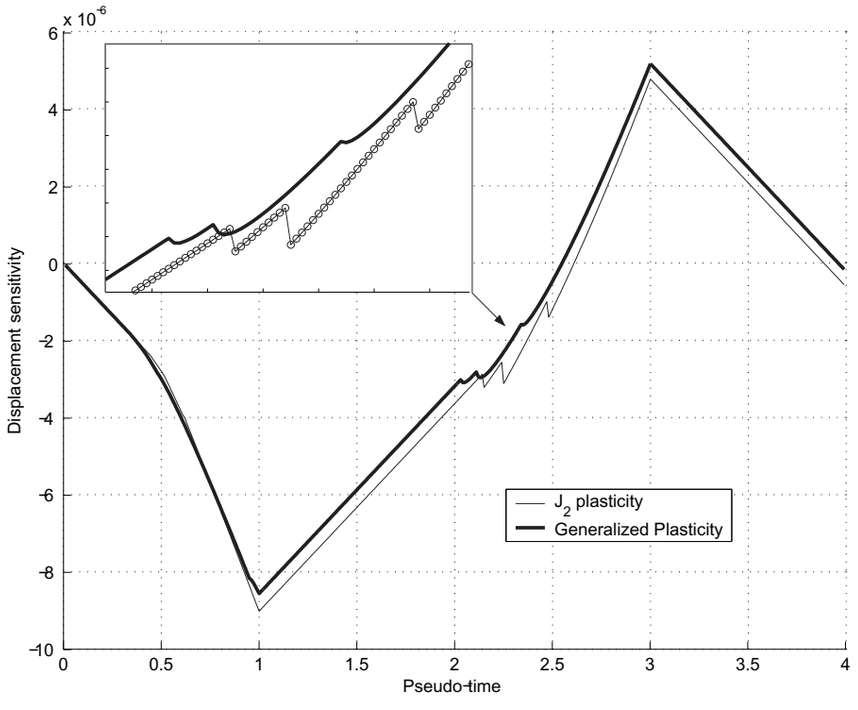

Figure 2.9: Displacement sensitivity results $\frac{\partial u}{\partial E}$ for one Quad4 element with two plasticity models.

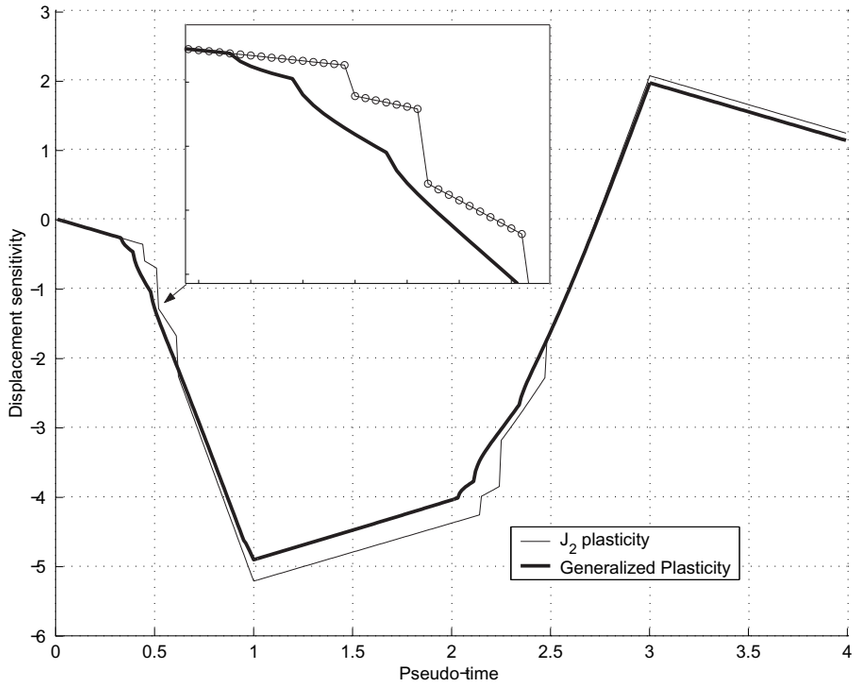

Figure 2.10: Displacement sensitivity results $\frac{\partial u}{\partial \nu}$ for one Quad4 element with two plasticity models.

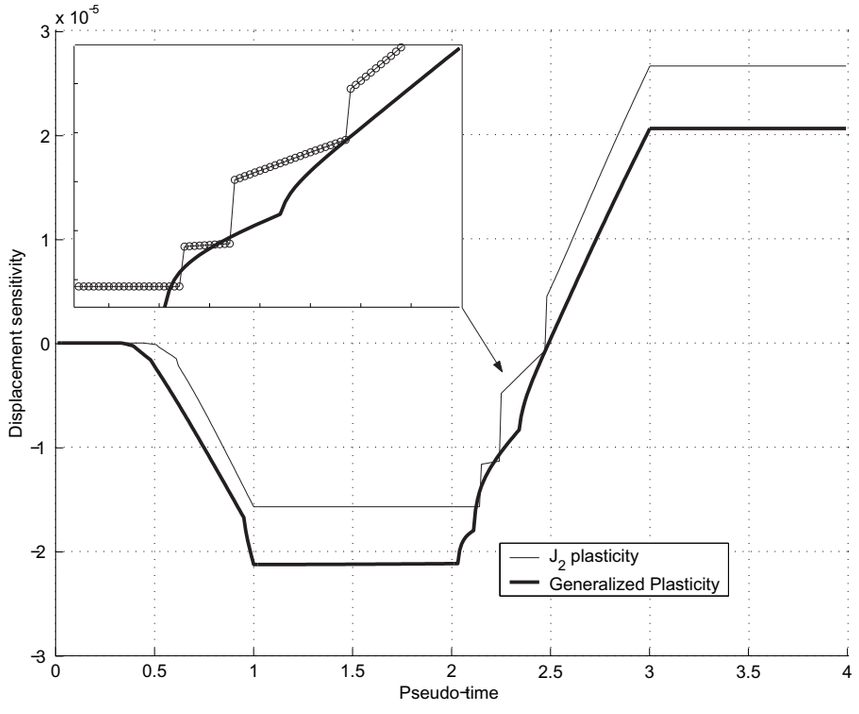

Figure 2.11: Displacement sensitivity results $\frac{\partial u}{\partial H_{iso}}$ for one Quad4 element with two plasticity models.

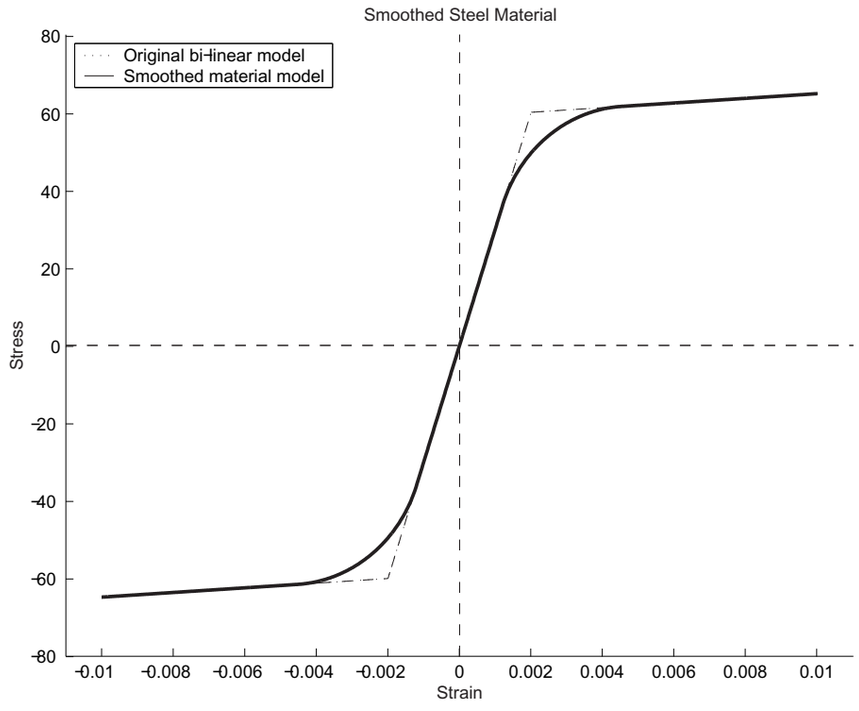

Figure 2.12: Bi-linear steel material smoothed with circular segment.

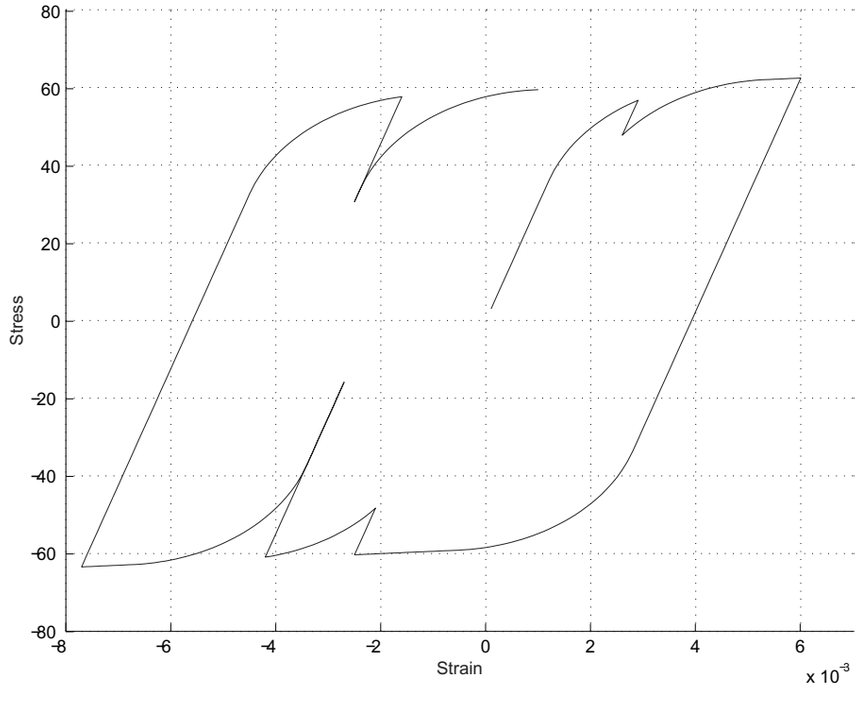

Figure 2.13: Example of stress-strain curve produced by smoothed material for $E = 30000$, $\sigma_y = 60$, $b = 2\%$, $\gamma = 60\%$ and $\eta = 3.0$.

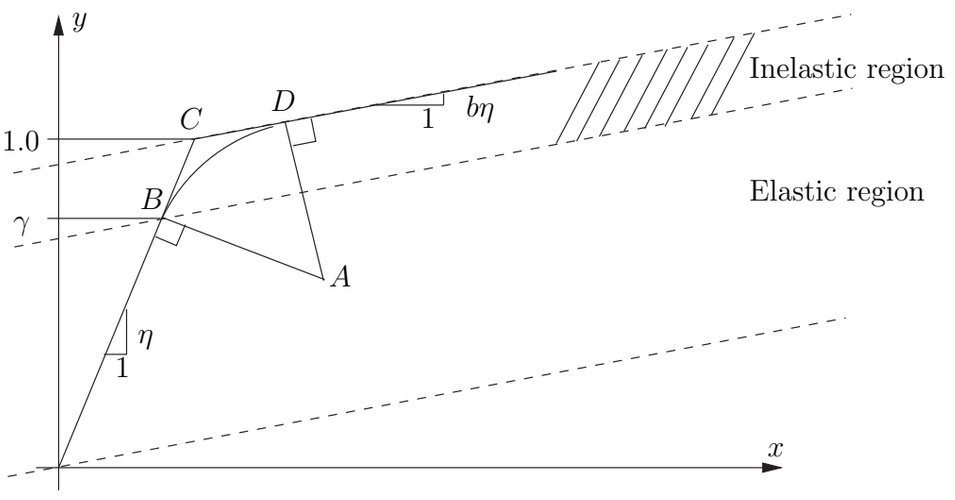

Figure 2.14: Determination of the center of the smoothing circular segment.

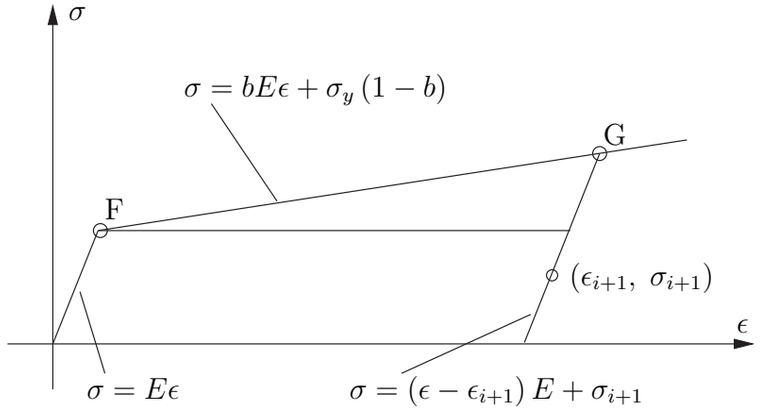

Figure 2.15: Shift of circle center after elastic material state determination. The point $(\epsilon_{i+1},\ \sigma_{i+1})$ denotes the current material state.

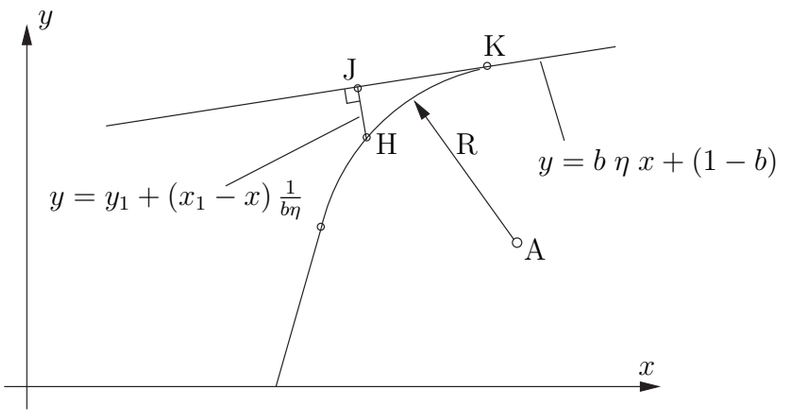

Figure 2.16: Determination of the new coordinates of the center of the circle after elastic unloading in inelastic region.

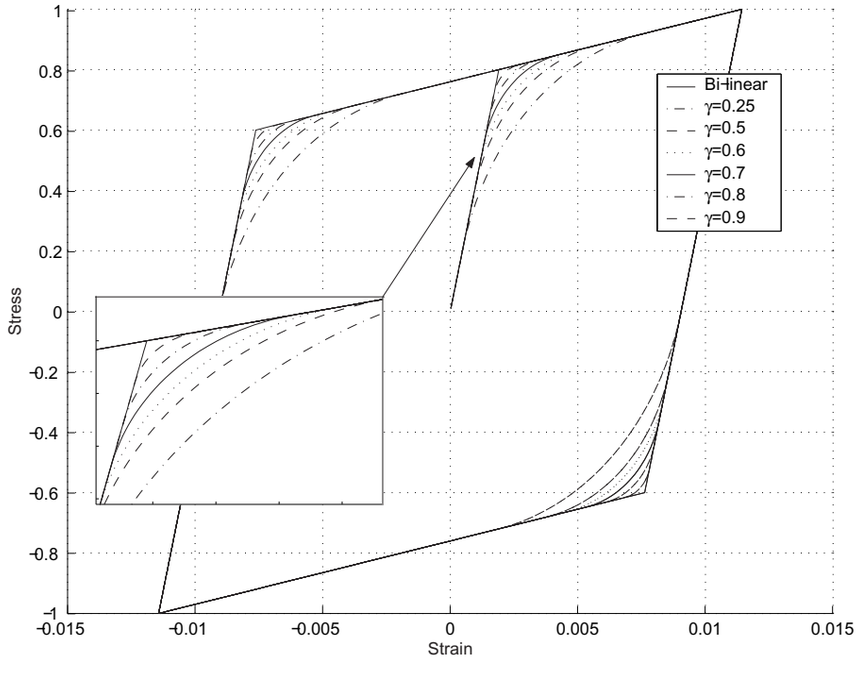

Figure 2.17: Stress-strain curves for bi-linear and smoothed bi-linear material.

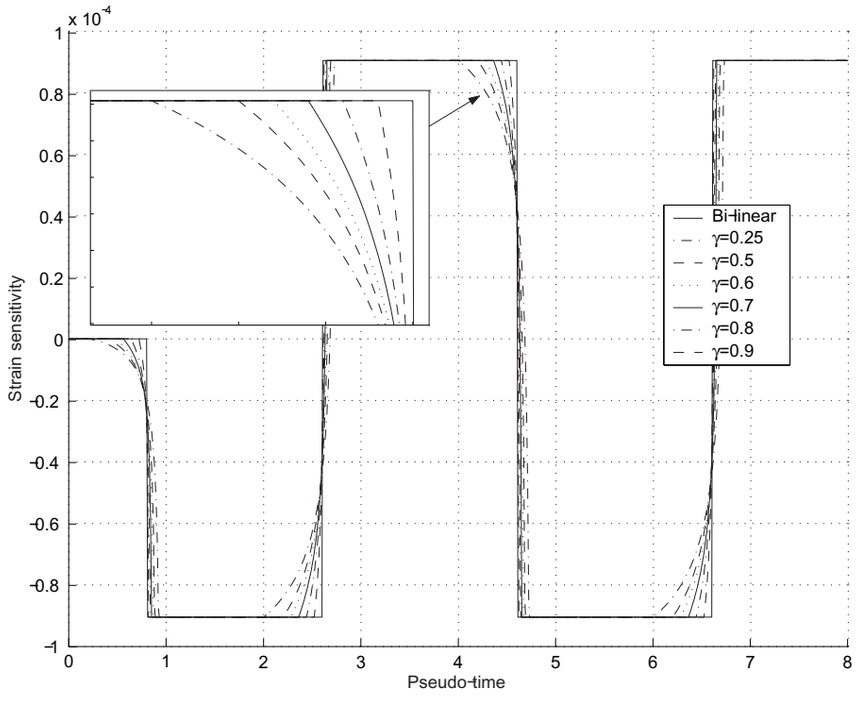

Figure 2.18: Strain sensitivity results $\frac{\partial \epsilon}{\partial \sigma_y}$ for bi-linear and smoothed bi-linear material.

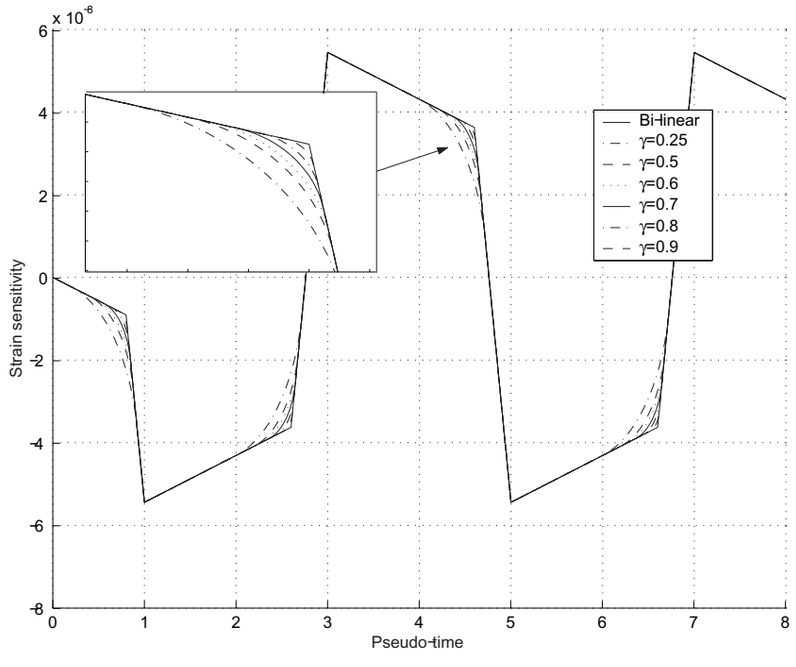

Figure 2.19: Strain sensitivity results $\frac{\partial \epsilon}{\partial E}$ for bi-linear and smoothed bi-linear material.

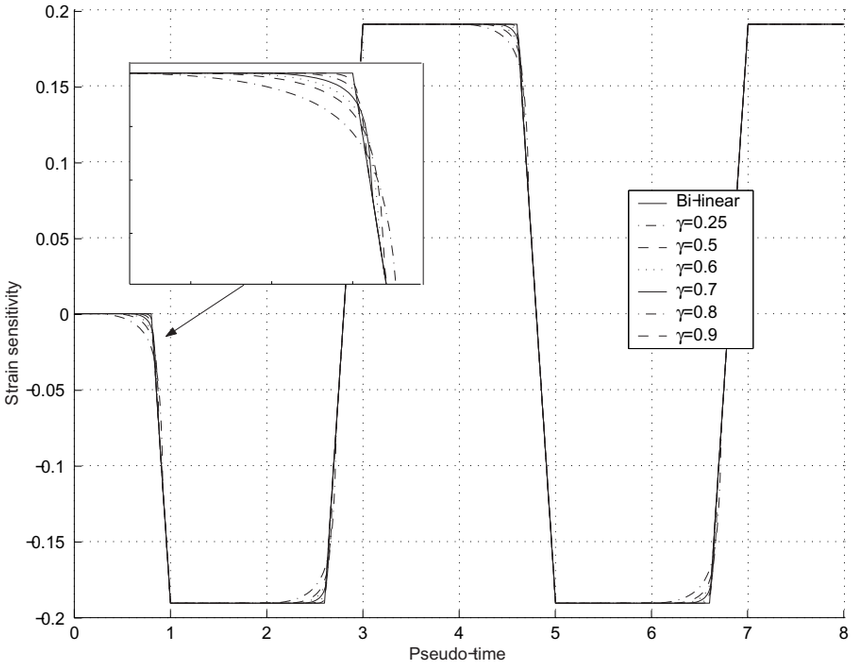

Figure 2.20: Strain sensitivity results $\frac{\partial \epsilon}{\partial b}$ for bi-linear and smoothed bi-linear material.

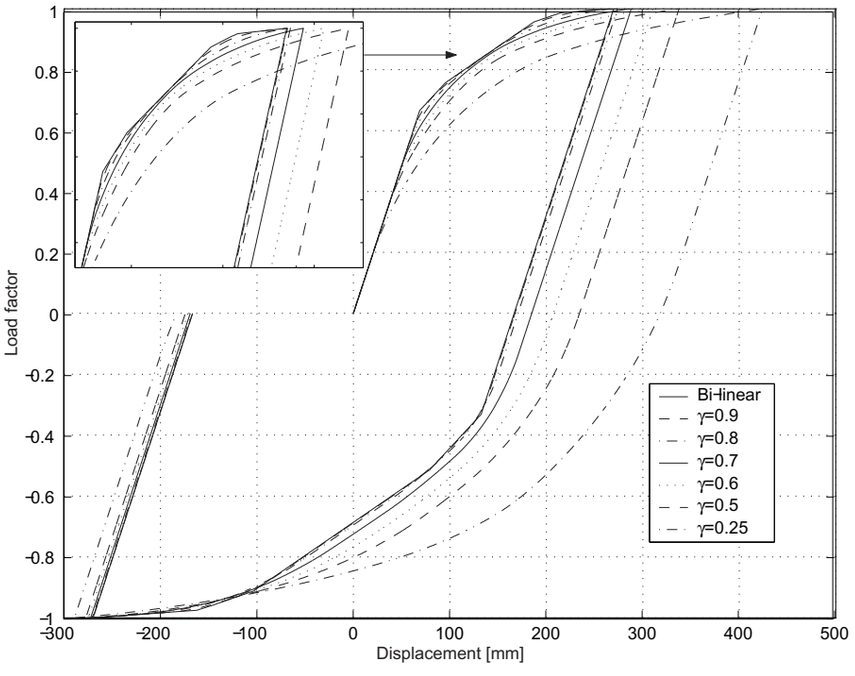

Figure 2.21: Load-displacement curve for 3-D truss model with bi-linear and smoothed bi-linear material.

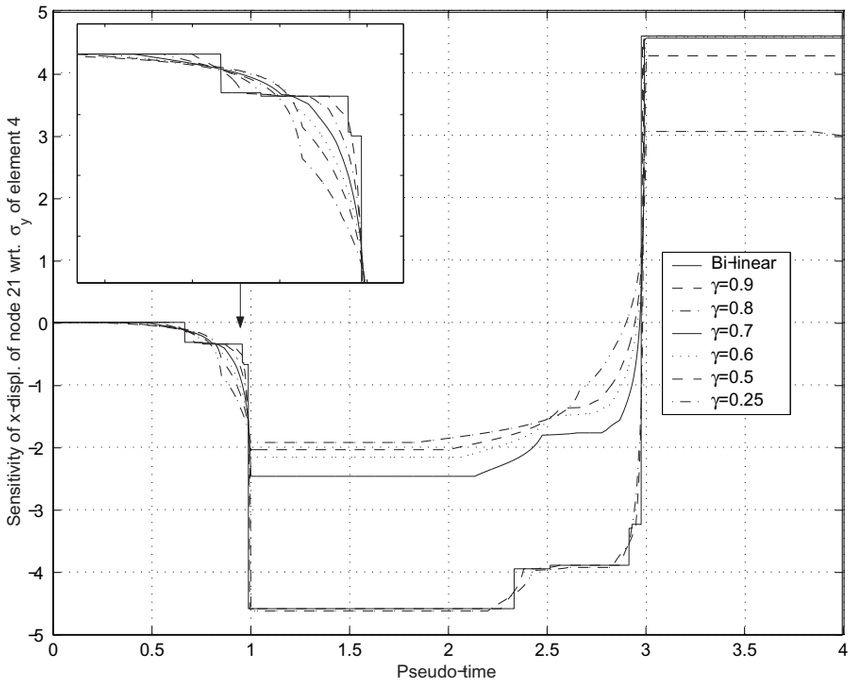

Figure 2.22: Sensitivity results of x-displacement at node 21 for 3-D truss with bi-linear and smoothed bi-linear material models.

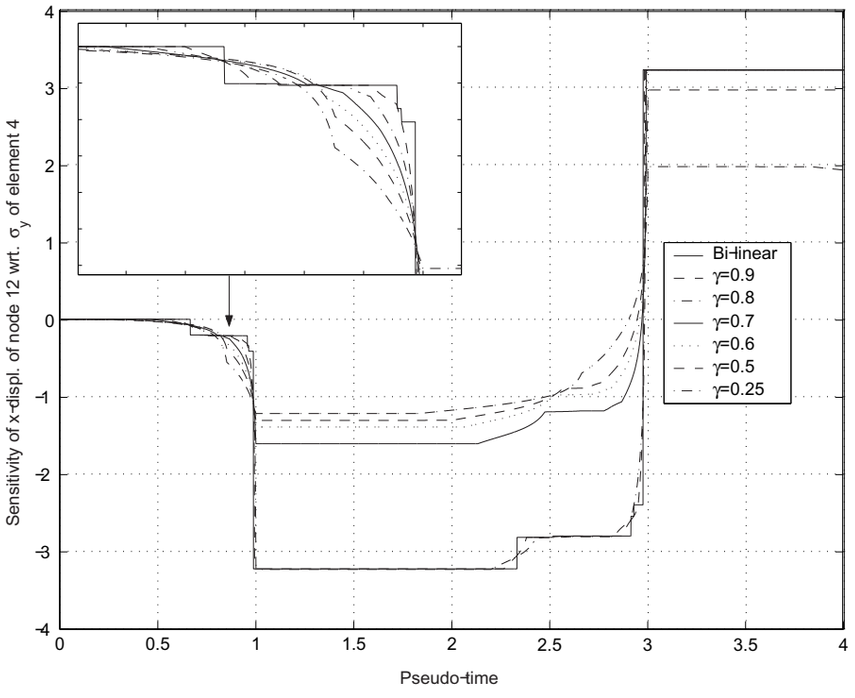

Figure 2.23: Sensitivity results of y-displacement at node 21 for 3-D truss with bi-linear and smoothed bi-linear material models.

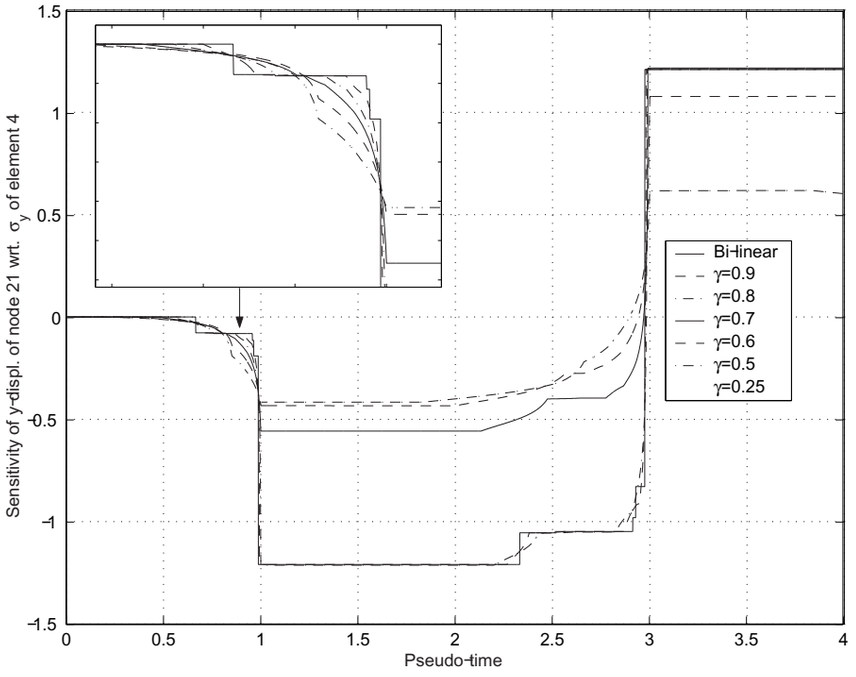

Figure 2.24: Sensitivity results for 3-D truss with bi-linear and smoothed bi-linear material models.

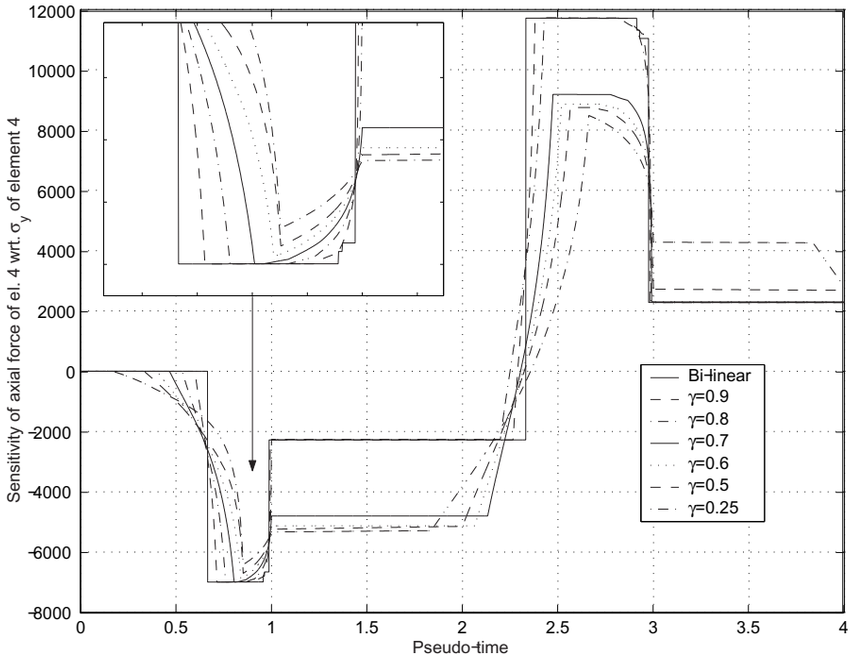

Figure 2.25: Sensitivity results of axial force in element 4 for 3-D truss with bi-linear and smoothed bi-linear material models.

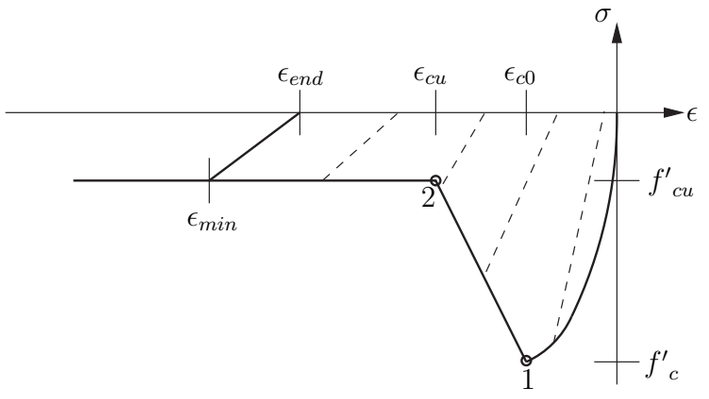

Figure 2.26: The original "Concrete01" material model in OpenSees.

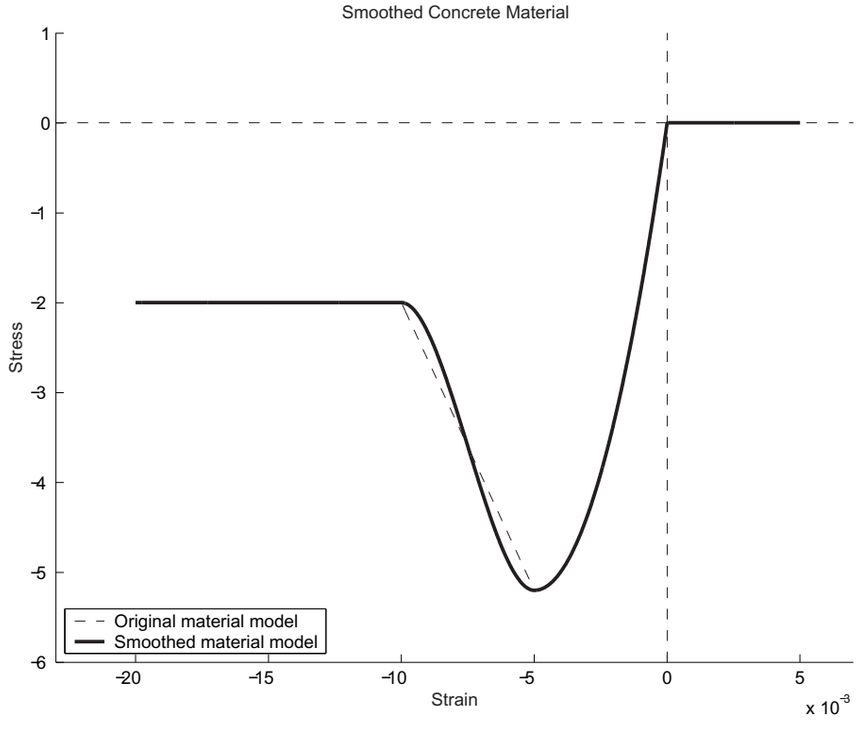

Figure 2.27: Smoothed backbone curve of Concrete01.

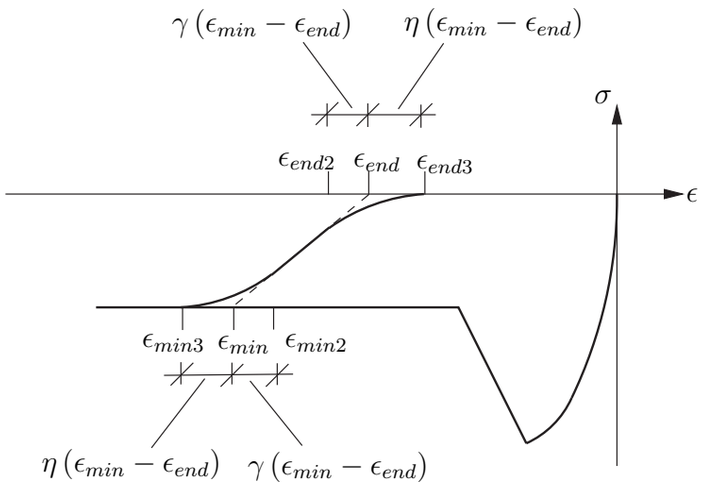

Figure 2.28: Smoothing of the unloading-reloading curve of the Concrete01 material model in OpenSees.

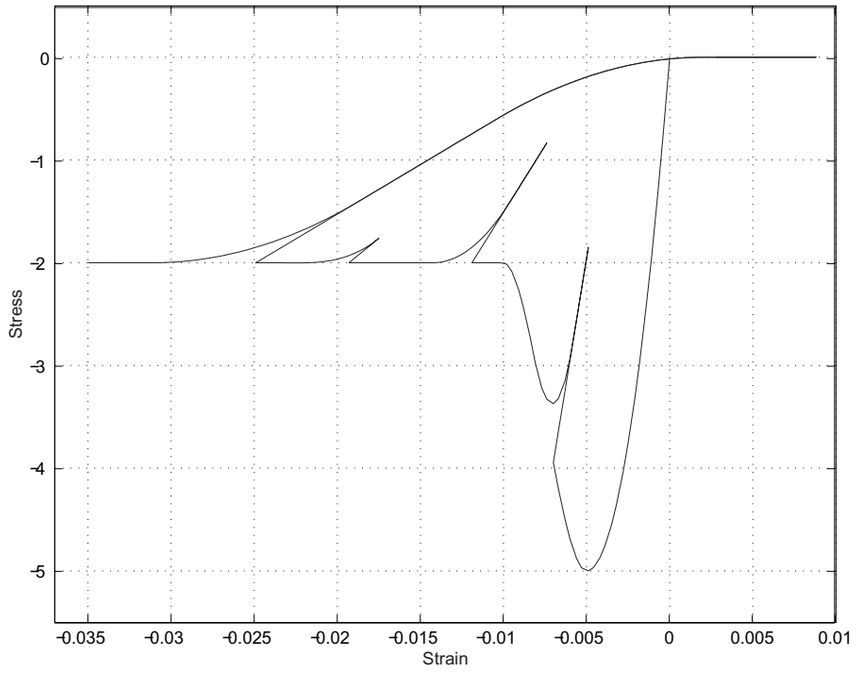

Figure 2.29: Example of stress-strain curve produced by smoothed concrete material model. In this example $f'_c = -5.0$, $f'_{cu} = -2.0$, $\epsilon_{c0} = -0.005$, $\epsilon_{cu} = -0.01$, $\gamma = 0.3$ and $\eta = 0.3$.

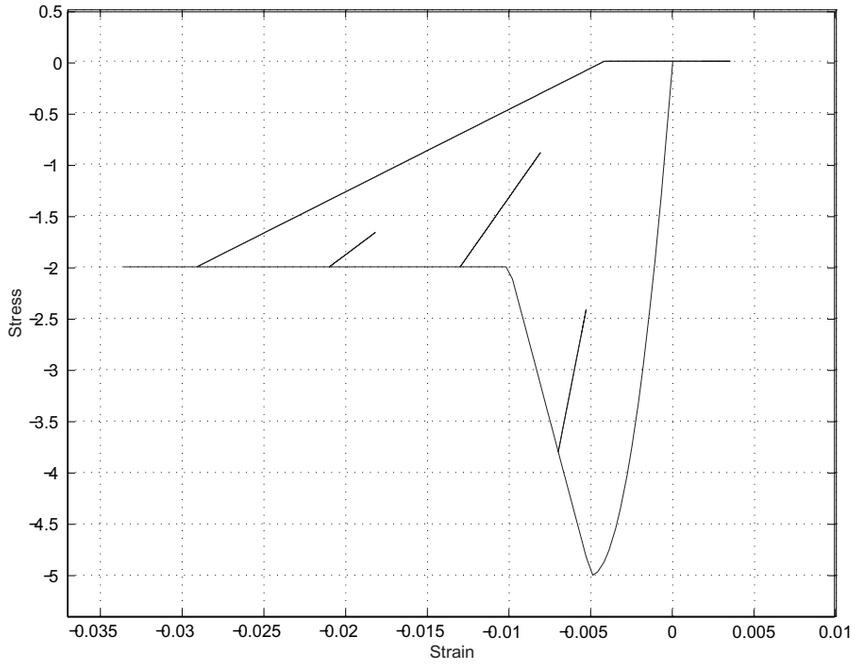

Figure 2.30: Example of stress-strain curve produced by original "Concrete01" material in OpenSees. Parameter values equal to those in Figure 2.29.

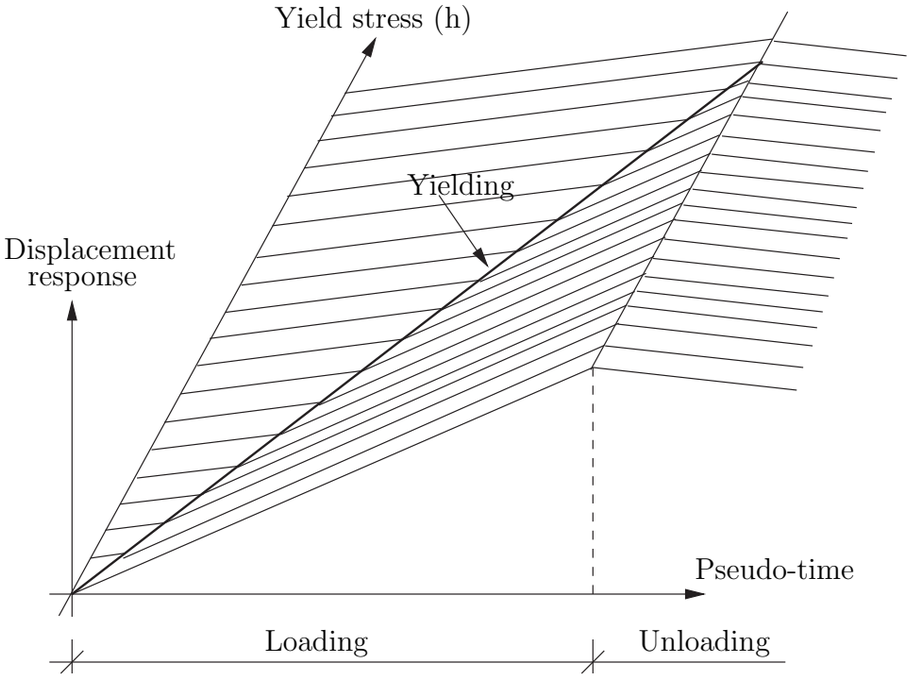

Figure 2.31: Conceptual displacement response for SDOF system with bi-linear material.

3 Finite Element Reliability Analysis

Structural reliability methods are employed in performance-based engineering to obtain probability estimates for various performance events. Structural performance is usually specified in terms of structural response quantities, such as strains and displacements, stresses and forces, and cumulative response measures, such as cumulative plastic strain or cumulative dissipated energy. Deterministic numerical prediction of structural response is commonly accomplished by use of the finite element method (Zienkiewicz and Taylor 2000, Hughes 1987, Bathe 1996, Cook *et al.* 1989). The term "finite element reliability methods" is used to describe the merging of advanced reliability methods, such as first- and second-order reliability methods (FORM and SORM) and importance sampling, with finite element methods to obtain probability estimates for predefined performance criteria. These methods are generally applicable to both linear and nonlinear structural problems.

The purpose of this chapter is twofold. First, to present the methodology that has been implemented in OpenSees and, second, to use the software to identify and address challenges in nonlinear finite element reliability analysis.

3.1 RELIABILITY ANALYSIS IN PERFORMANCE-BASED EARTHQUAKE ENGINEERING

The level of safety in structural design is prescribed by the society or the owner of the structure. In the past the concept of Allowable Stress was in use before the introduction of the load-and-resistance factor-design (LRFD) method. In both approaches, safety or load and resistance factors prescribed by the design code are used to account for uncertainties in structural properties, geometrical imperfections, and environmental demands. More recently, the concept of the performance-based design approach has emerged. This development is motivated by the availability of computational tools to simulate, in close approximation, real structural performance. Instead of satisfying nontransparent code regulations, the society or client can now require satisfaction of realistic performance criteria

for given hazards at prescribed probabilities. Such performance levels may range from demanding immediate occupancy to retainment of structural integrity. It is noted that most failures to meet performance criteria for reasonably designed structures are rare events. That is, the failure event is often in the tail of the probability distribution of the response. This determines the kind of reliability method that can be used. Furthermore, nonlinear structural effects must be accounted for in most failure modes. These facts motivate the developments in this chapter.

In the scheme used by the Pacific Earthquake Engineering Research (PEER) Center, performance is defined in terms of so-called Decision Variables (DV). In general, these are functions of Damage Measures (DM), which are functions of Engineering Demand Parameters (EDP), which again are functions of Intensity Measures (IM). As an illustration, the peak ground acceleration is an IM, the strain in the unconfined concrete of a column is an EDP, the amount of concrete spalling is a DM and the cost of repair is a DV. Eq. (3.1) illustrates the inter-dependence between these quantities:

$$IM \longrightarrow EDP \longrightarrow DM \longrightarrow DV \tag{3.1}$$

Each arrow represents a model. For example, a simulation model provides the EDP for a given IM, and a damage model provides the DM for a given EDP. Independence is assumed for quantities that are not directly linked by a model. For instance, given the EDP, the DM are assumed to be independent of the IM.

IM, EDP, DM and DV must be considered as random variables. Uncertainties are introduced in each model. It is of interest to estimate the probability of events defined in terms of the DV. Several methods are available for this purpose. One approach addresses the scalar case and employs the theorem of total probability for this purpose. The total probability of an event A over a continuous random variable X is given by:

$$\mathrm{P}[A] = \int_{-\infty}^{\infty} \mathrm{P}[A \,|\, X = x]\, f(x)\, \mathrm{d}x \tag{3.2}$$

where $\mathrm{P}[A \,|\, X = x]$ is the conditional probability of A given the outcome $X = x$, and $f(x)$ is the probability density function (PDF) of X. Eq. (3.2) is applied to the random variables in Eq. (3.1) by first assuming that the PDF $f(im)$ of the intensity measure is available. The cumulative distribution function (CDF) $F(edp)$ of the engineering demand parameter is obtained by the rule of total probability:

$$F(edp) = \mathrm{P}[EDP \leq edp] = \int_{-\infty}^{\infty} \mathrm{P}[EDP \leq edp \,|\, IM = im]\, f(im)\, \mathrm{d}im \tag{3.3}$$

The corresponding PDF is obtained by differentiation:

$$f(edp) = \frac{\mathrm{d}F(edp)}{\mathrm{d}edp} \tag{3.4}$$

Similarly, this procedure is applied to the DM and the DV. The end result is the triple integral:

$$F(dv) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(dv|dm) f(dm|edp) f(edp|im) f(im) \, \mathrm{d}im \, \mathrm{d}edp \, \mathrm{d}dm \tag{3.5}$$

where $F(dv|dm)$ denotes the conditional CDF of DV for given $DM = dm$ and $f(dm|edp)$ and $f(edp|im)$ are the conditional PDFs of DM and EDP. The triple-integral can be evaluated as a matrix-vector quadruple product by arranging $f(im)$ into a vector and the conditional CDF and PDF values $F(dv|dm)$, $f(dm|edp)$ and $f(edp|im)$ into matrices with the number of rows and columns equal to the number of discrete values of the random variables.

Provided appropriate models, the reliability methodology described in this report is capable of addressing the entire problem in one analysis, instead of evaluating the conditional probabilities $F(dv|dm)$, $f(dm|edp)$ and $f(edp|im)$ separately. This is done by incorporating IM and model parameters as random variables in the reliability analysis. The finite element model internally provides the $IM \longrightarrow EDP$ and $EDP \longrightarrow DM$ models. The probability of events defined in terms of DV can be estimated when the relevant $DM \longrightarrow DV$ models are provided as input. The following sections describe how OpenSees can be used for this purpose.

3.2 UNCERTAINTY MODELING

The parameters of a classical finite element model are selected deterministically. In finite element reliability analysis, the uncertainty characterization of such parameters becomes an important task. This section describes the formulation of marginal and joint probability distributions to characterize random variables in the OpenSees environment. For convenience, the Nataf family of joint distributions developed by Liu and Der Kiureghian (1986) is used. This family of joint distributions is completely defined by specifying the marginal distributions and correlation structure of the random variables.

3.2.1 Library of Marginal Distribution Functions

In OpenSees an option for a user-defined probability distribution is available together with a library of predefined distributions. The library distributions include Normal, Lognormal, Negative lognormal, Exponential, Shifted exponential, Rayleigh, Shifted Rayleigh, Uniform, Gamma, Beta, Type I Largest

Value (Gumbel), Type I Smallest Value, Type II Largest Value, Type III Smallest Value and Weibull. A detailed description of these distributions and their parameters is given in Appendix B.9. The User's Guide in Chapter 3.7 provides the syntax for creating random variable objects in OpenSees with these marginal distributions.

3.2.2 User-Defined Distributions

The user-defined distribution type allows the user to specify a random variable with a PDF of arbitrary shape. The PDF must obey the normalization rule $\int_{-\infty}^{\infty} f(x)\,\mathrm{d}x = 1$. In fact, OpenSees checks that the distribution is normalized. If not normalized, the user is informed that all distribution function values are uniformly increased/decreased to obtain a normalized distribution. If any distribution function value is negative, an error message is given. Any number of points can be specified, either on the command line or in a file. See Section 4.2 for further description of the command syntax.

Based on the user's specification of the PDF values $f(x_i)$ at discrete realization points $x_1 < x_2 < \cdots < x_n$, the PDF $f(x)$ for an arbitrary value x is found by linear interpolation. The CDF $F(x) = \int_{-\infty}^{x} f(x)\,\mathrm{d}x$ is obtained by applying the trapezoidal integration rule. The inverse CDF $x = F^{-1}(p)$ is obtained by first determining the discrete point $\tilde{x}_i = \min\{x_i \mid F(x_i) > p\}$. The slope of the PDF in the interval where the solution is located is $a = \frac{f(x_i)-f(x_{i-1})}{x_i-x_{i-1}}$. The solution is $x = x_{i-1} + \frac{-B+\sqrt{B^2-4\,A\,C}}{2\,A}$, where $A = \frac{a}{2}$, $B = f(x_{i-1})$ and $C = F(x_{i-1}) - p$, unless $a = 0$, in which case $x = \frac{p-F(x_{i-1})+f(x_{i-1})\,x_{i-1}}{f(x_{i-1})}$.

3.2.3 Correlation Structures

Each pair of random variables is assigned a correlation coefficient $-1 \leq \rho \leq 1$. The special case $\rho = 0$ indicates no correlation, while the special cases $\rho = \pm 1$ indicate perfect positive or negative correlation. In the latter two cases the random variables are linearly dependent.

All the correlation coefficients are assembled in a matrix $\mathbf{R} = [\rho_{ij}]$, where the indices i and j represent random variable numbers. The diagonal of this matrix contains entries with value 1.0. The matrix $\mathbf{R}$ is required to be positive definite, provided no linear relation exists between the random variables, in which case $\mathbf{R}$ is singular. This requirement may not be satisfied if the user specifies the

elements of $\mathbf{R}$ in an arbitrary manner. It is therefore of interest to (1) test for positive definiteness of the matrix and (2) provide library correlation structures for groups of random variables. This is done in OpenSees. One fruitful approach to the latter item is to establish so-called homogeneous correlation matrices by using concepts from stationary random processes. Consider a stationary random process $x(t)$ with an auto-correlation function $\rho(\tau)$, where τ is the absolute difference between two time points. If we select a set of discrete points $t_i,\ i = 1, \ldots, n$, then the auto-correlation matrix having the elements $\rho_{ij} = \rho(|t_i - t_j|)$ is positive definite for any selections of t_i provided $\rho(\tau)$ is a valid auto-correlation function. In OpenSees, such correlation structures are specified by assigning a "time" value to each random variable. This provides a method to establish valid correlation matrices from known valid correlation functions.

The conditions for a valid correlation function $\rho(\tau)$ are as follows:

1. $\rho(\tau)$ must be symmetric.
2. $|\rho(\tau)|$ must be bounded by $\rho(0)$, that is, $|\rho(\tau)| \leq \rho(0)$.
3. $\rho(\tau)$ must be a non-negative definite function, that is, its Fourier transform must be non-negative everywhere.
4. $\rho(\tau)$ must be continuous at all τ if it is continuous at 0.
5. If the random process has no periodic component, then $\lim_{|\tau|\to\infty} \rho(\tau) = 0$

In OpenSees the following correlation structures satisfying the above conditions have been implemented (see Chapter 3.7 regarding command syntax for specification of these correlation structures):

$$\begin{aligned}
\rho_{ij} &= \exp\left(-\frac{|i-j|}{\theta}\right) \\
\rho_{ij} &= \exp\left(-\frac{(i-j)^2}{\theta^2}\right) \\
\rho_{ij} &= \frac{1}{1+\theta\ (i-j)^2} \\
\rho_{ij} &= \begin{cases} 1 - \frac{|i-j|}{\theta} & \text{for} \quad |i-j| \leq \theta \\ 0 & \text{for} \quad |i-j| > \theta \end{cases}
\end{aligned} \tag{3.6}$$

In addition to the above, a correlation structure according to the Dunnett-Sobel model (Dunnett and Sobel 1955) is implemented. In this model the off-diagonal terms in the correlation matrix read:

$$\rho_{ij} = r_i\ r_j \qquad (i \neq j) \tag{3.7}$$

$\mathbf{r}$ is a user-defined vector with its elements satisfying the condition $-1 < r_i < 1$. The diagonal elements of $\mathbf{R}$ are set to unity. This correlation matrix is positive definite in the following manner. A matrix $\mathbf{R}$ is said to be positive definite if, for all $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x}^T\mathbf{R}\mathbf{x} > 0$. For the correlation matrix given by Eq. (3.7), using index notation (summation over repeated indices), we have two contributions to the product $\mathbf{x}^T\mathbf{R}\mathbf{x}$. Firstly, from the diagonal terms of $\mathbf{R}$:

$$x_i \, \delta_{ij} \, x_j = x_i^2 \tag{3.8}$$

where δ_{ij} is the Kronecker delta. It is noted that

$$x_i^2 > x_i^2 \, r_i^2 \tag{3.9}$$

Secondly, the contribution from the off-diagonal terms of $\mathbf{R}$ is:

$$x_i \rho_{ij} x_j = x_i r_i r_j x_j \qquad i \neq j \tag{3.10}$$

It is noted that $x_i r_i r_j x_j = (x_i r_i)^2$. Adding the contributions from Eqs. (3.8) and (3.10) and employing the inequality of Eq. (3.9), we obtain:

$$\mathbf{x}^T\mathbf{R}\mathbf{x} > (x_i r_i)^2 > 0 \tag{3.11}$$

This shows that $\mathbf{R}$ is positive definite for this class of correlation matrices.

3.2.4 Joint Probability Distributions

An essential ingredient in the FORM, SORM and importance sampling techniques in reliability analysis is to find the so-called design point. This is described in Section 3.5. The search for this point is performed in an uncorrelated standard normal space. It is therefore necessary to transform the original vector of random variables into this space. The selection of joint probability distributions and the transformation to the standard normal space are described in this section.

In OpenSees the Nataf model for joint distributions (Liu and Der Kiureghian 1986) is implemented. This class of joint distributions is completely defined by specifying the marginal distributions and the correlation structures of the random variables. Compared to other such joint distribution models, e.g., the Morgenstern model (see Liu and Der Kiureghian 1986), the Nataf model allows a relatively wide range of correlation values, depending on the distribution type. To explain the implementation, first consider the case where the random variables $\mathbf{x}$ are jointly normal. The mean

vector is denoted $\mathbf{M}_x = \{\mu_i\}$ and the covariance matrix $\mathbf{\Sigma}_{xx} = [\rho_{ij}\sigma_i\sigma_j]$. To transform $\mathbf{x}$ to the uncorrelated standard normal space we seek a linear transformation $\mathbf{y} = \mathbf{a}_0 + \mathbf{A}\mathbf{x}$ such that $\mathbf{M}_y = \mathbf{0}$ and $\mathbf{\Sigma}_{yy} = \mathbf{I}$. Expressing the mean and the covariance of the linear function leads to two equations in the unknowns $\mathbf{a}_0$ and $\mathbf{A}$:

$$\mathbf{M}_y = \mathbf{a}_0 + \mathbf{A}\mathbf{M}_x = \mathbf{0} \tag{3.12}$$

$$\mathbf{\Sigma}_{yy} = \mathbf{A}\mathbf{\Sigma}_{xx}\mathbf{A}^T = \mathbf{I} \tag{3.13}$$

Since $\mathbf{\Sigma}_{xx}$ is a positive definite matrix, we can write $\mathbf{\Sigma}_{xx} = \hat{\mathbf{L}}\hat{\mathbf{L}}^T$ where $\hat{\mathbf{L}}$ is the lower triangular Cholesky decomposition of $\mathbf{\Sigma}_{xx}$. Substitution into Eq. (3.13) leads to:

$$\mathbf{A}\mathbf{\Sigma}_{xx}\mathbf{A}^T = \left(\mathbf{A}\,\hat{\mathbf{L}}\right)\,\left(\hat{\mathbf{L}}^T\mathbf{A}^T\right) = \left(\mathbf{A}\,\hat{\mathbf{L}}\right)\,\left(\mathbf{A}\hat{\mathbf{L}}\right)^T = \mathbf{I} \tag{3.14}$$

It is clear that $\mathbf{A}\,\hat{\mathbf{L}} = \mathbf{I}$ and we have:

$$\mathbf{A} = \hat{\mathbf{L}}^{-1} \tag{3.15}$$

Next, Eq. (3.12) yields:

$$\mathbf{a}_0 + \hat{\mathbf{L}}^{-1}\mathbf{M}_x = \mathbf{0} \quad \Rightarrow \quad \mathbf{a}_0 = -\hat{\mathbf{L}}^{-1}\,\mathbf{M}_x \tag{3.16}$$

Hence, the transformation into the uncorrelated standard normal space for a vector $\mathbf{x}$ of dependent normal random variables reads

$$\mathbf{y} = \hat{\mathbf{L}}^{-1}\,(\mathbf{x} - \mathbf{M}_x) \tag{3.17}$$

Before proceeding, the same transformation is derived for the case when the covariance matrix is written $\mathbf{\Sigma} = \mathbf{DRD}$, where $\mathbf{D} = diag\,[\sigma_i]$ is the diagonal matrix of standard deviations and $\mathbf{R} = [\rho_{ij}]$ is the correlation matrix. Using the same procedure as above, we obtain the transformation

$$\mathbf{y} = \mathbf{L}^{-1}\mathbf{D}^{-1}\,(\mathbf{x} - \mathbf{M}_x) \tag{3.18}$$

where $\mathbf{L}$ now is the lower triangular decomposition of the correlation matrix $\mathbf{R}$, i.e., $\mathbf{R} = \mathbf{L}\mathbf{L}^{\mathrm{T}}$. This form of the transformation is preferred, since the decomposition is done on the dimensionless matrix $\mathbf{R}$ rather than the covariance matrix, which in general has elements with a mix of dimensions. The Jacobian of the above transformation is:

$$\mathbf{J}_{\mathbf{y},\mathbf{x}} = \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \mathbf{L}^{-1}\mathbf{D}^{-1} = \hat{\mathbf{L}}^{-1} \tag{3.19}$$

Now, consider the case, where $\mathbf{x}$ consists of non-normal but statistically independent random variables. In this case each random variable is transformed into the standard normal space by the probability preserving transformation

$$\Phi(y_i) = F_i(x_i) \quad \Rightarrow \quad y_i = \Phi^{-1}\left[F_i\left(x_i\right)\right] \tag{3.20}$$

where $\Phi(\,\cdot\,)$ is the standard normal CDF and $F_i(\,\cdot\,)$ is the CDF of x_i. The Jacobian of this transformation is found by differentiating the first part of Eq. (3.20) on both sides with respect to x_i. Employing the chain rule $\frac{\partial}{\partial x} = \frac{\partial}{\partial y}\frac{\partial y}{\partial x}$ we find

$$\mathbf{J}_{\mathbf{y},\mathbf{x}} = \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \operatorname{diag}\left[\frac{f\,(x_i)}{\varphi\,(y_i)}\right] \tag{3.21}$$

We now consider the case of dependent non-normal random variables having the joint Nataf distribution. First a set of standard normal random variables $\mathbf{z}$ is obtained by marginal transformation of the original random variables $\mathbf{x}$:

$$\Phi\,(z_i) = F_i\,(x_i) \qquad \Longrightarrow \qquad z_i = \Phi^{-1}\,[F_i\,(x_i)] \tag{3.22}$$

Next, the assumption is made that the random variables $\mathbf{z}$ are jointly normal. This leads to the so-called Nataf distribution for $\mathbf{x}$ (Liu and Der Kiureghian 1986). The expression for the joint PDF of $\mathbf{x}$ is obtained from the elementary probability transformation rule: $f(\mathbf{x})\; dx_1\, dx_2 \cdots dx_n = \varphi(\mathbf{z}, \mathbf{R}_o)\; dz_1\, dz_2 \cdots dz_n$, i.e.,

$$f(\mathbf{x}) = \varphi(\mathbf{z},\, \mathbf{R}_o)\frac{dz_1\, dz_2 \cdots dz_3}{dx_1\, dx_2 \cdots dx_3} = \varphi(\mathbf{z},\, \mathbf{R}_o)\frac{f(x_1)\, f(x_2) \cdots f(x_3)}{\varphi(z_1)\, \varphi(z_2) \cdots \varphi(z_3)} \tag{3.23}$$

where $\varphi(\mathbf{z},\, \mathbf{R}_o)$ is the joint normal PDF of $\mathbf{z}$ having the correlation matrix $\mathbf{R}_o$. The last equality in Eq. (3.23) is obtained by differentiating Eq. (3.22) with respect to x_i. The final transformation to uncorrelated standard normal variables is obtained by making use of the special cases in Eqs. (3.18) and (3.20):

$$\mathbf{y} = \mathbf{L}_0^{-1}\,\mathbf{z} = \mathbf{L}_0^{-1}\begin{bmatrix} \Phi^{-1}\,[F\,(x_1)] \\ \Phi^{-1}\,[F\,(x_2)] \\ \vdots \\ \Phi^{-1}\,[F\,(x_n)] \end{bmatrix} \tag{3.24}$$

where $\mathbf{L}_0$ now is the lower-triangular decomposition of the correlation matrix $\mathbf{R}_o$ of the standard but correlated normal random variables $\mathbf{z}$. Furthermore, from the preceding special cases, it is readily seen that the Jacobian of the transformation in Eq. (3.24) is:

$$\mathbf{J}_{\mathbf{y},\mathbf{x}} = \mathbf{L}_0^{-1}\,\operatorname{diag}\left[\frac{f\,(x_i)}{\varphi\,(z_i)}\right] \tag{3.25}$$

The remaining task is to obtain the correlation matrix $\mathbf{R}_0 = \rho_{0,\,ij}$ of $\mathbf{z}$ based on the correlation matrix $\mathbf{R} = \rho_{ij}$ of $\mathbf{x}$ as provided by the user. The correlation coefficient between two continuous random variables X and Y can be written

$$\rho_{XY} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left(\frac{X-\mu_X}{\sigma_X}\right)\left(\frac{Y-\mu_Y}{\sigma_Y}\right)\, f_{XY}\,(x, y)\; \mathrm{d}x\, \mathrm{d}y \tag{3.26}$$

where μ denotes the mean values, σ denotes the standard deviation values and $f_{XY}(x, y)$ is the joint PDF. In the present case, the joint PDF is as in Eq. (3.23), leading to the following integral equation to relate ρ_{ij} and $\rho_{0,ij}$:

$$\begin{aligned} \rho_{ij} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left(\frac{x_i-\mu_i}{\sigma_i}\right)\left(\frac{x_j-\mu_j}{\sigma_j}\right)\varphi_2(z_i,\, z_j,\, \rho_{0,ij})\,\frac{f(x_i)\,f(x_j)}{\varphi(x_i)\,\varphi(z_j)}\mathrm{d}x_i\,\mathrm{d}x_j \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left(\frac{F^{-1}(\Phi(z_i))-\mu_i}{\sigma_i}\right)\left(\frac{F^{-1}(\Phi(z_j))-\mu_j}{\sigma_j}\right)\varphi_2(z_i,\, z_j,\, \rho_{0,ij})\,\mathrm{d}z_i\,\mathrm{d}z_j \end{aligned} \tag{3.27}$$

where $F^{-1}(\,)$ indicates the inverse CDF and the bi-variate standard normal PDF reads:

$$\varphi_2(z_i,\, z_j,\, \rho) = \frac{1}{2\pi\sqrt{1-\rho^2}}\, e^{-\frac{1}{2(1-\rho^2)}\left[z_i^2 - 2\,z_i\,z_j\,\rho + z_j^2\right]} \tag{3.28}$$

The desired correlation coefficients $\rho_{0,ij}$ must be solved for by using Eq. (3.27). Closed-form approximate expressions for $\rho_{0,ij}$ for an array of probability distributions are provided in Liu and Der Kiureghian (1986). These are implemented in OpenSees. The difference between the two correlation coefficients $\rho_{0,ij}$ and ρ_{ij} is usually small for all but strongly nonnormal distributions. Eq. (3.27) is solved by a Newton scheme in OpenSees for cases where the user selects the user-defined probability distribution type.

The random variables characterized by their marginal distributions and correlation structures discussed above can be mapped into the finite element model of OpenSees. For instance, a random variable may represent a material parameter, a geometry parameter, a load variable, etc. In the following section an additional option for characterizing a stochastic load process is presented.

3.2.5 Discretized Random Process Loading

Time series objects are used in OpenSees to describe time variation of earthquake ground motions or nodal loads. The work in this section enables specification of a discretized random process loading to be applied to the structure in OpenSees.

Two principally different methods of creating realizations of random processes are currently implemented in OpenSees. The first method is used in conjunction with sampling techniques to generate realizations of a random process based on a pre-defined power spectral density (PSD) function. The second option involves the specification of a filtered and modulated white noise process, which can be used in FORM analysis to estimate the mean out-crossing rates of selected response quantities. The latter option is motivated by the works of Li and Der Kiureghian (1995), Der Kiureghian and Li (1996), Der Kiureghian (2000), and Koo and Der Kiureghian (2003).

First the option of generating a time series input from a user-defined PSD is described. The spectrum is discretized into a number of frequency intervals of width $\Delta\omega$, with $w_i,\ i = 1, \ldots, N$, denoting the center frequency points. For a two-sided PSD function $S(\omega)$ the variance of a frequency interval is $\sigma^2 = 2\,S(\omega_i)\,\Delta\omega$. One method of creating a realization is to create two vectors of randomly selected numbers $A_i = \text{Normal}(0, 1)$ and $\theta_i = \text{Uniform}(0, 2\pi)$ and evaluate the following sum at each time instant t:

$$x(t) = \mu(t) + \sum_{i=1}^{N} \sqrt{2\ S(\omega_i)\ \Delta\omega}\ \ A_i\ \cos\left(\omega_i t + \theta_i\right) \tag{3.29}$$

where $\mu(t)$ is a time-varying mean function. This is implemented in OpenSees.

The starting point for the second random process implementation in OpenSees is a train of pulses, equally spaced along the time axis. By letting each pulse be represented by a standard normal random variable y_i, this train of pulses approaches the Gaussian white noise process as the number of pulses over a finite time interval approaches infinity. This basic process is modified by multiplying each pulse y_i at time point t_i by the unit impulse response function $h(t - t_i)$ of a filter. One option for the filter implemented in OpenSees is the single-degree-of-freedom oscillator characterized by its eigenfrequency and damping ratio. Furthermore, a mean function $\mu(t)$ may be added and a modulating function $q(t)$ can be applied to control the variance along the time axis. The simplest form of this stochastic input model is, thus, written as:

$$x(t) = \mu(t) + c_1\ q(t)\ \sum_{i=1}^{N} y_i\ h(t - t_i) \tag{3.30}$$

where N is the number of equally spaced pulses along the time axis and c_1 is a factor to control the overall intensity of the process. It is noted that this time series may have a physical interpretation in seismic engineering. Each pulse can be regarded as part of a fault rupture. These pulses are then filtered through the ground medium before reaching the base of the structure.

The process definition in Eq. (3.30) contains only one filter. As a result, the output process has a stationary frequency content. To introduce nonstationary frequency content, multiple filters with different modulating functions may be used (Der Kiureghian 2000), i.e.,

$$x(t) = \mu(t) + c_2\ \sum_{k=1}^{K} q_k(t) \sum_{i=1}^{N} y_{ik}\ h_k(t - t_i) \tag{3.31}$$

In this most general case the number of random pulses is equal to the number of time steps multiplied by the number of filters. A simpler alternative is implemented in OpenSees, where the original vector y_i of random variables is used with all filters:

$$x(t) = \mu(t) + c_3\ \sum_{k=1}^{K} q_k(t) \sum_{i=1}^{N} y_i\ h_k(t - t_i) \tag{3.32}$$

In this case the stochastic model can be physically interpreted as a single train of white noise pulses filtered through a model with K peaks in the transfer function. The magnitudes of these peaks are controlled by the user's selection of the modulating function for each filter, for instance to accommodate a target PSD function.

An important aspect of the implementations in OpenSees has been the DDM sensitivity calculations. It is noted that the derivatives of the above random load processes with respect to the random variables y_i or y_{ik} are quite simple. Using the simpler expression in Eq. (3.32), the sensitivity is:

$$\begin{aligned} \frac{\partial x(t)}{\partial y_i} &= c_3 \sum_{k=1}^{K} q_k(t) \sum_{i=1}^{N} h_k(t-t_i) \qquad \text{if } t_i < t \\ &= 0 \qquad\qquad\qquad\qquad\qquad\qquad\quad \text{otherwise} \end{aligned} \tag{3.33}$$

To obtain the sensitivity with respect to a random variable, each random variable is replaced by 1 or 0, depending on whether it is the parameter in question or not.

The factors c_i in Eqs. (3.30) to (3.32) scale the overall variance of the stochastic processes. In OpenSees this factor is computed based on a target standard deviation prescribed by the user. Since y_i (or y_{ik}) are statistically independent random variables with unit variances, it is easy to see that the variance of the process in Eq. (3.31) is:

$$\text{Var}[x(t)] = c_2^2 \sum_{k=1}^{K} \sum_{i=1}^{N} q_k(t)^2 \; h_k(t-t_i)^2 \tag{3.34}$$

whereas that defined by Eq. (3.32) is:

$$\text{Var}[x(t)] = c_3^2 \sum_{k=1}^{K} \sum_{l=1}^{K} \sum_{i=1}^{N} q_k(t) q_l(t) \; h_k(t-t_i) h_l(t-t_i) \tag{3.35}$$

For the simplified process implemented in OpenSees represented by Eq. (3.32) the factor c_3 comes out as:

$$c_3 = \sqrt{\frac{\text{Var}[x(t)]_{target}}{\sum_{k=1}^{K} \sum_{l=1}^{K} \sum_{i=1}^{N} q_k(t) q_l(t) \; h_k(t-t_i) h_l(t-t_i)}} \tag{3.36}$$

In OpenSees the user prescribes the target maximum standard deviation $\sqrt{Var[X(t)]}_{target}$ of the process. The program automatically searches for the time instant at which the maximum variance is achieved and adjusts the parameter c_3 accordingly. A number of modulating function forms are available to control the relative temporal evolution of the outcome of each filter. Options in OpenSees include a constant modulating function, a trapezoidal modulating function and a Gamma modulating function of the form $q(t) = a\, t^b\, e^{-c\, t}$, where a and b and c are user-defined parameters. Chapter 3.7 provides the syntax for creating modulating functions in OpenSees.

3.3 PERFORMANCE FUNCTIONS

When applying reliability analysis methods in performance-based engineering, it is essential to have means of defining performance criteria. This is done by so-called performance functions.[1] In structural reliability analysis the term "failure" is used to denote the event of not meeting performance criteria. While the methodology to compute estimates of the probability of failure is at a mature stage for a range of performance functions, it may not be an obvious task as to how to define what constitutes failure. This topic is discussed in this section.

3.3.1 Component and System Reliability Problems

By using the terminology introduced in Section 3.1, a performance function can be defined in terms of engineering demand parameters, damage measures or decision variables. Estimation of the probability of the defined failure event poses a "component" reliability problem. However, failure according to one performance criterion may not constitute failure of an entire structural system. It is therefore of interest to define "system" reliability problems as sets of components and rules as to which combinations of component failures constitute system failure.

This chapter discusses methods to estimate failure probabilities of both component and system problems. Component probabilities and information about dependence between components are used when addressing the system reliability problem.

3.3.2 General Characteristics of Performance Functions

Performance functions for structural components are commonly denoted $g(\mathbf{x})$, where $\mathbf{x}$ is the vector of basic random variables. The dependence on $\mathbf{x}$ can be implicit and through structural response quantities. However, $g(\mathbf{x})$ must be a continuous and differentiable function of $\mathbf{x}$, at least in the realizable domain of $\mathbf{x}$. The numerical value of the performance function distinguishes the failure state from the safe state:

$$g > 0 \quad : \text{safe}$$

[1] In this report the term "performance function" is used interchangeably with the term "limit-state function."

$$g = 0 \quad : \text{limit-state}$$
$$g \leq 0 \quad : \text{failure}$$

In the space of uncorrelated standard normal variates $\mathbf{y}$ the performance function is denoted $G(\mathbf{y})$. We have that:

$$G\left(\mathbf{y}(\mathbf{x})\right) = g(\mathbf{x}) \quad \Leftrightarrow \quad g\left(\mathbf{x}(\mathbf{y})\right) = G(\mathbf{y}) \tag{3.37}$$

Finite element reliability methods are characterized by response quantities from a finite element solution entering the performance function. For instance, a simple threshold performance function is:

$$g = \text{threshold} - \text{response quantity} \tag{3.38}$$

When the uncertain response quantity exceeds the specified threshold, the performance function takes on a negative value and failure is implied. In OpenSees, any analytical expression involving stresses, stress resultants, strains, displacements and accumulated damage can be specified as the response quantity in Eq. (3.38). Note that both the specified threshold and the computed response can be functions of the random variables $\mathbf{x}$. Limitations on which response quantities can be included in the performance functions are treated in Section 3.3.4.

3.3.3 Performance Functions for Performance-Based Earthquake Engineering

The performance-based engineering approach, as opposed to prescriptive rules of code-based design, is based on simulation of real structural behavior. The client or government regulations prescribe desired performance objectives, which are translated into decision variables, DV, or functions thereof. In formalized performance-based engineering, four performance levels are distinguished relative to earthquake events (Federal Emergency Management Agency 2000):

1. Operational performance: the event does not affect the occupants or functioning of the building.
2. Immediate occupancy performance: the occupants can immediately return to the building.
3. Life safety performance.
4. Collapse prevention performance.

The client or code regulations determine an acceptable hazard level for each of these performance requirements. For an earthquake event with probability, say, 50% in 50 years, immediate occupancy

performance may be demanded. On the other hand, for an earthquake event with probability 2% in 50 years only life safety performance may be desirable. The remaining question in this section is how to translate such performance requirements into performance functions in finite element reliability analysis.

3.3.4 Performance Functions in Nonlinear Finite Element Reliability Analysis

Typically, the steps in a static nonlinear finite element analysis are governed by a load-control scheme. That is, a load increment is applied at each step. Displacement control, on the other hand, implies that a displacement increment is applied at each step. Other alternatives are available, e.g., the arc-length method (Crisfield 1991). In static analysis, the employed finite element analysis scheme may affect the selection of response quantities that can be included in the performance function. This is readily seen by the following example. A user selects a displacement-control analysis scheme. Furthermore, it is decided that the analysis should run to a displacement equal to 10 cm at a control node. The user then defines a limit-state function, where a nodal displacement (not necessarily the control node) at the end of the analysis is included. Such a reliability analysis will most likely not converge. This is due to the fact that displacement responses in such a structure are largely controlled by the imposed displacement at the control node and less by variations in the realizations of the random variables. Similarly, a performance function involving a force quantity sampled at the end of a force-controlled stepping scheme may not lead to convergence in reliability analysis.

Another important aspect of such analysis schemes is that if the response quantities in the performance function are sampled at the end point of the analysis, then the analysis must be run to a consistent pseudo-time, e.g., load factor, in every finite element analysis. This is particularly important when the size of the pseudo-time steps are selected automatically by the finite element analysis procedure. If measures are not taken to assure that the same final pseudo-time is reached in each analysis, the arbitrary pseudo-time step size selection will contribute to determining the final response. This may lead to convergence problems in the reliability analysis. In OpenSees it is possible to let the so-called integrator object select the sizes of the pseudo-time steps, within prescribed boundaries. In Section 4.4 remedies are discussed to employ such schemes in reliability analysis.

In dynamic problems, the steps in the finite element analysis are usually taken with a prescribed time increment Δt. The problem mentioned above then is not encountered. However, it may not be meaningful to define performance functions in the dynamic case in terms of response quantities at

the end of the finite element analysis. Rather, the mean out-crossing rate discussed in Section 3.4.6 may be of interest, or accumulated response/damage measures may be employed. More about this kind of analysis is presented in Chapter 4.6.

Another restriction in defining performance functions for reliability analysis is related to the maximum or minimum of a structural response over a time or space interval. Such problems are known as space- or time-variant reliability problems. The gradient of the maximum or minimum of a function is discontinuous, since it is not a-priori clear which point in time or space will govern. One realization of the random model parameters may lead to one space/time point of maximum, while another realization may shift the location. This violates the assumption of continuous gradients in the search for the design point. Again, one approach to dealing with this challenge is the mean out-crossing rate computations discussed in Section 3.4.6. It should be mentioned, however, that in a static pushover analysis, where only one peak of the load-displacement curve is encountered, a performance function can be defined in terms of the maximum force level.

Table 3.1 summarizes the quantities that can be included in the formulation of the performance function for different types of reliability analysis.

3.4 ESTIMATION OF PERFORMANCE PROBABILITIES AND RESPONSE STATISTICS

The primary concern in structural reliability analysis is to estimate probabilities of failure to achieve predefined performance. In the simplest case of one performance function, the component reliability problem is formulated as

$$p_f = \int_{g(\mathbf{x})\leq 0} f(\mathbf{x})\, d\mathbf{x} \tag{3.39}$$

where p_f is the probability of failure, $\mathbf{x}$ is the vector of random finite element model parameters, $g(\mathbf{x})$ is the performance function and $f(\mathbf{x})$ is the joint PDF of $\mathbf{x}$. Note that the integration is over the set of random variables $\mathbf{x}$, which in finite element reliability analysis can be large. Closed-form solutions of Eq. (3.39) are unavailable except for a few special cases. For this reason, a number of methods have been developed for the purpose of solving the integral approximately. These include the first- and second-order reliability methods, FORM and SORM, sampling analysis, response surface methods and numerical integration schemes. The latter method is usually not a feasible alternative when the number of random variables is greater than 3 or 4. Furthermore, in finite element reliability analysis, it is desirable to limit the number of evaluations of g and its gradient. This makes methods

such as FORM, SORM and importance sampling (IS) tractable, while it excludes the crude Monte Carlo sampling scheme for problems with small failure probabilities. A common aspect of FORM, SORM and IS is that they all employ the so-called design point. This is the most likely point in the failure domain, when the variables are transformed to the standard normal space. As such, this is the ideal point for approximating the limit-state surface separating the safe and failure domains. FORM analysis estimates the failure probability by approximating the limit-state surface by the tangent hyper-plane at the design point. SORM analysis estimates the failure probability by approximating the limit-state surface by a quadratic surface tangent at the design point. An IS analysis may subsequently use the design point as the center of sampling to obtain an improved estimate of the failure probability. The options for FORM and IS analysis are implemented in OpenSees. The framework for SORM analysis is also implemented in OpenSees. However, this option is currently limited to a simple algorithm to estimate the first principal curvature for the fitting paraboloid using a method developed by Der Kiureghian and DeStefano (1991). For this reason, a detailed description of the SORM methodology is not presented in this chapter.

Before proceeding to discuss methods that address the probability integral in Eq. (3.39), the method available in OpenSees for computing second moments of response quantities are described. These statistics of the response describe the propagation of uncertainties through the finite element model.

3.4.1 Second-Moment Response Statistics

Two methods are implemented in OpenSees for the purpose of computing second-moment statistics of finite element response quantities. These can be used to estimate the mean and variance of a response quantity and the correlation coefficient between pairs of response quantities. The first method employs the well-known mean-centered, first-order Taylor series approximation of a function of random variables:

$$g(\mathbf{x}) \approx g\left(\boldsymbol{\mu}\right) + \nabla g\ \left(\mathbf{x} - \boldsymbol{\mu}\right) \tag{3.40}$$

where $\boldsymbol{\mu}$ is the vector of means of the random variables $\mathbf{x}$, and the gradient $\nabla g = \left[\frac{\partial g}{\partial x_1} \cdots \frac{\partial g}{\partial x_n}\right]$ is the gradient row vector. This approach requires a single finite element analysis together with response sensitivity computations at the mean point. If a finite difference scheme is used instead of the DDM to compute the gradients, then additional finite element analyses are necessary.

The first-order estimate of the mean is the function value evaluated at the mean of the random variables:

$$\mathrm{E}[g] = \mu_g \approx g\left(\boldsymbol{\mu}\right) \tag{3.41}$$

The variance is:

$$\mathrm{Var}[g] = \sigma_g^2 \approx \nabla g \; \boldsymbol{\Sigma} \; \nabla g^T \tag{3.42}$$

where $\boldsymbol{\Sigma}$ is the covariance matrix of the random variables $\mathbf{x}$. The covariance between two functions g_1 and g_2 is given by:

$$\mathrm{Cov}[g_1,\, g_2] \approx \nabla g_1 \; \boldsymbol{\Sigma} \; \nabla g_2^T \tag{3.43}$$

and the corresponding correlation coefficient is obtained by normalizing the covariance:

$$\rho_{g_1,\, g_2} = \frac{\mathrm{Cov}[g_1,\, g_2]}{\sigma_{g_1} \; \sigma_{g_2}} \tag{3.44}$$

Eqs. (3.41) to (3.44) are implemented in OpenSees to estimate second-moment statistics of user-defined performance functions. If a user desires to compute the statistics of a particular response quantity, then the performance function should be defined as only that response quantity, e.g., $g = u_{7,1}$ if the statistics of the displacement of node 7 along degree-of-freedom number 1 are required. Correlation coefficients are automatically computed whenever two or more performance functions are defined. In this study, the above method will be referred to as FOSM analysis.

As an alternative to FOSM analysis, a sampling scheme is implemented in OpenSees. Sample realizations $\mathbf{x}_i,\ i = 1, \ldots, N$, of $\mathbf{x}$ are generated and the corresponding performance function values $g(\mathbf{x}_i)$ are computed. The estimate of the mean reads:

$$\mathrm{E}[g] = \mu_g \approx \frac{1}{N} \sum_{i=1}^{N} g(\mathbf{x}_i) \tag{3.45}$$

The estimate of the variance reads:

$$\mathrm{Var}[g] = \sigma_g^2 \approx \frac{1}{N-1} \left[\sum_{i=1}^{N} g(\mathbf{x}_i)^2 - \frac{1}{N} \left(\sum_{i=1}^{N} g(\mathbf{x}_i) \right)^2 \right] \tag{3.46}$$

Similarly, the covariance between two performance functions g_1 and g_2 is obtained from:

$$\mathrm{Cov}[g_1,\, g_2] \approx \frac{1}{N-1} \left[\sum_{i=1}^{N} g_1(\mathbf{x}_i)\, g_2(\mathbf{x}_i) - \frac{1}{N} \left(\sum_{i=1}^{N} g_1(\mathbf{x}_i) \right) \left(\sum_{i=1}^{N} g_2(\mathbf{x}_i) \right) \right] \tag{3.47}$$

The corresponding correlation coefficient is computed from Eq. (3.44).

The coefficient of variation of the estimated mean provides a measure of accuracy of the sampling estimates. It is computed from:

$$\mathrm{c.o.v}_{\mu_g} = \frac{\sigma_g}{\mu_g \sqrt{N}} \tag{3.48}$$

3.4.2 FORM

FORM analysis addresses the reliability problem in Eq. (3.39) by means of two key operations. Firstly, we find the design point in the transformed uncorrelated standard normal space. Secondly, we approximate the limit-state surface at this point and make use of the properties of the standard normal space to obtain the probability estimate. The first item can be a challenging task in nonlinear finite element reliability analysis. It is discussed in detail in Section 3.5.

The distance from the origin of the standard normal space to the design point, denoted $\mathbf{y}^*$, is termed the "reliability index" and denoted β. The first-order probability estimate is then found as

$$p_f \approx p_{f1} = \Phi(-\beta) \tag{3.49}$$

where $\Phi(\cdot)$ denotes the standard normal CDF. It is common to define the "alpha vector" as the negative normalized gradient row vector at the design point, i.e.,

$$\boldsymbol{\alpha} = -\frac{\nabla G}{\|\nabla G\|} \tag{3.50}$$

where $\nabla G = \left[\frac{\partial G}{\partial y_1} \cdots \frac{\partial G}{\partial y_n}\right]$. In terms of $\boldsymbol{\alpha}$, the reliability index is given by $\beta = \boldsymbol{\alpha}\mathbf{y}^*$. Additionally, $\boldsymbol{\alpha}$ is one of several parameter importance measures available as a by-product of FORM analysis. Such measures are valuable in practical engineering design. They can also be used to reduce the number of random variables in a model. This topic is treated separately in Section 3.6.

3.4.3 Importance Sampling Analysis

In the crude Monte Carlo method the sampling distribution is centered at the mean point. Since failure events tend to occur in the tail regions of probability distributions, this implies that a large number of samples are required to obtain good failure probability estimates. A concern in finite element reliability analysis is that the large number of evaluations of the performance function may be computationally costly. This inhibits the use of crude Monte Carlo sampling in most finite element reliability applications. By centering the sampling distribution near the failure domain, a far more efficient sampling scheme is obtained. This is the basic idea behind the importance sampling method.

By introducing an indicator function $I(\mathbf{y})$ such that $I(\mathbf{y}) = 1$ if $g(\mathbf{x}) \leq 0$, and $I(\mathbf{y}) = 0$ otherwise, we can rewrite Eq. (3.39) as follows (Ditlevsen and Madsen 1996):

$$p_f = \int_{\Omega_\mathbf{y}} I(\mathbf{y})\, \varphi(\mathbf{y})\, d\mathbf{y} = \int_{\Omega_\mathbf{y}} \left(I(\mathbf{y}) \frac{\varphi(\mathbf{y})}{f(\mathbf{y})} \right) f(\mathbf{y})\, d\mathbf{y} \tag{3.51}$$

where $\Omega_{\mathbf{y}}$ denotes the entire standard normal space, $\varphi(\cdot)$ is the joint standard normal PDF and $f(\mathbf{y})$ is a joint PDF which must be non-zero within the region where $I(\mathbf{y}) = 1$. It is observed that the last integral in Eq. (3.51) is an expectation of the random variable $I(\mathbf{y})\frac{\varphi(\mathbf{y})}{f(\mathbf{y})}$ relative to the distribution $f(\mathbf{y})$. This expectation can be estimated by generating statistically independent realizations of the random variable $I(\mathbf{y})\frac{\varphi(\mathbf{y})}{f(\mathbf{y})}$ derived from the distribution $f(\mathbf{y})$. The average of this sample is an unbiased estimator of the expectation and, thus, of p_f. Crude Monte Carlo sampling analysis results when $f(\mathbf{y})$ is selected identical to $\varphi(\mathbf{y})$ so that the distribution is centered at the mean point, i.e., the origin in the standard normal space. In IS analysis the center of the sampling distribution is shifted to a point selected by the user, e.g., the design point (Melchers 1999).

The sampling analysis implemented in OpenSees evaluates Eq. (3.51) by repeatedly generating a vector $\tilde{\mathbf{y}}$ of independent and normally distributed random numbers with zero means and unit variances. $\tilde{\mathbf{y}}$ is then transformed according to $\mathbf{y} = \mathbf{y}_{center} + \mathbf{L}\ \tilde{\mathbf{y}}$, where $\mathbf{y}_{center}$ is a user-provided mean vector and $\mathbf{L}$ is the Cholesky decomposition of a user-provided covariance matrix $\mathbf{\Sigma}$. The mean vector defines the center of the sampling density. The most effective choice is the design point. The covariance matrix is typically chosen as a unit matrix. Increasing (decreasing) the value of a diagonal element of $\mathbf{\Sigma}$ leads to a broader (narrower) sampling density along the axis of the corresponding random variable. Off-diagonal elements may be added to make the sampling distribution elongated in preferred directions.

The vector $\mathbf{y}$ is transformed back into the original space $\mathbf{x} = \mathbf{x}(\mathbf{y})$ where the performance function $g(\mathbf{x})$ is evaluated for this realization of the random variables. $I(\mathbf{x})$ is assigned a value based on the outcome of $g(\mathbf{x})$. The variable $q(\mathbf{y}) = I\left(\mathbf{x}(\mathbf{y})\right)\frac{\varphi(\mathbf{y})}{f(\mathbf{y})}$ is then evaluated with the following expressions for the joint PDF in the standard normal space, $\varphi(\mathbf{y})$, and the sampling density $f(\mathbf{y})$:

$$f(\mathbf{y}) = \frac{1}{(2\pi)^{n/2}\ \sqrt{\det\mathbf{\Sigma}}}\ \exp\left[-\frac{1}{2}\left(\mathbf{y}-\mathbf{y}_{center}\right)^T\mathbf{\Sigma}^{-1}\left(\mathbf{y}-\mathbf{y}_{center}\right)\right] \tag{3.52}$$

$$\varphi(\mathbf{y}) = \frac{1}{(2\pi)^{n/2}}\ \exp\left[-\frac{1}{2}\,\mathbf{y}^T\mathbf{y}\right] \tag{3.53}$$

where n is the number of random variables.

The probability of failure is estimated as:

$$p_f \approx p_{f,sim} = \bar{q} = \frac{1}{N}\sum_{i=1}^{N} q_i \tag{3.54}$$

where $q_i = q(\mathbf{y}_i)$ and N is the number of samples. A measure of accuracy of the probability estimate is the variance of $p_{f,sim}$, which noting that q_i are statistically independent and identically distributed,

is given by:

$$\text{Var}[p_{f,sim}] = \sum_{i=1}^{N} \frac{1}{N^2} \text{Var}[q_i] = \frac{1}{N} \text{Var}[q] \tag{3.55}$$

where Var[q] is the common variance, which is estimated from the generated sample using the well-known formula:

$$\text{Var}[q] \approx \frac{1}{N-1} \left[\sum_{i=1}^{N} q_i^2 - \frac{1}{N} \left(\sum_{i=1}^{N} q_i \right)^2 \right] \tag{3.56}$$

Substituting Eq. (3.56) into Eq. (3.55), the variance of the probability estimate is:

$$\text{Var}\left[p_{f,sim}\right] \approx \frac{1}{N(N-1)} \left[\sum_{i=1}^{N} q_i^2 - \frac{1}{N} \left(\sum_{i=1}^{N} q_i \right)^2 \right] \tag{3.57}$$

Thus, we only need to store the sums of the q_i and the q_i^2 values. In OpenSees, the coefficient of variation of the probability estimate

$$\text{c.o.v}[p_{f,sim}] = \frac{\sqrt{\text{Var}\left[p_{f,sim}\right]}}{p_{f,sim}} \tag{3.58}$$

is computed and monitored. The sampling is repeated a user-defined number of times, or until the above c.o.v. estimate falls below a specified target.

3.4.4 Parametric Reliability Analysis

In OpenSees a particular tool is created to generate probability results by a sequence of reliability analyses. CDF's and PDF's (or their complements) of finite element response quantities are obtained by specifying performance functions in terms of parameterized thresholds. Similarly, fragility curves are generated by designating demand parameters in the finite element model to be varied. More generally, reliability results can be generated as a function of a designated parameter.

Consider a performance function $g\left(\mathbf{x}, \theta\right)$, where θ is the parameter to be varied. While fragility and CDF curves contain failure probabilities at discrete θ-values, reliability sensitivity analysis[2] of FORM results is needed to obtain corresponding PDF's. For this purpose, the FORM estimate of the failure probability, $p_{f1} = \Phi(-\beta)$, is differentiated with respect to θ:

$$\frac{\partial}{\partial \theta} \Phi(-\beta) = \frac{\partial}{\partial \theta} \left(1 - \Phi(\beta)\right) = -\frac{\partial \beta}{\partial \theta} \frac{\partial}{\partial \beta} \Phi(\beta) = -\frac{\partial \beta}{\partial \theta} \varphi(\beta) \tag{3.59}$$

[2]The term reliability sensitivity analysis should not be confused with response sensitivity analysis. The former involves differentiation of failure probability estimate or reliability index with respect to a distribution parameter or a performance function parameter. The latter is treated in Chapter 1.5 and deals with derivatives of finite element response quantities with respect to random model parameters.

The derivative of the reliability index is (Hohenbichler and Rackwitz 1986), (Bjerager and Krenk 1989):

$$\frac{\partial \beta}{\partial \theta} = \frac{1}{\|\nabla G\|} \frac{\partial g}{\partial \theta} \tag{3.60}$$

where $\frac{\partial g}{\partial \theta}$ is easily computed since θ usually enters the performance function in a simple algebraic form. The above is implemented in OpenSees such that probability density results for response quantities are automatically generated together with the parametric reliability results.

3.4.5 System Reliability Analysis

A system reliability problem is one where several performance functions, each representing a component, are defined and where the user specifies the sets of components whose joint failure constitutes failure of the system. Three such formulations are considered.

A series system fails if any of its components fail, i.e., any performance function takes on a negative value. The failure domain can thus be written as the union of the failure domains of the individual components:

$$\mathcal{F}_s = \left\{ \bigcup_{k=1}^{K} g_k(\mathbf{x}) \leq 0 \right\} \tag{3.61}$$

A parallel system fails if all of its components fail. The failure domain can thus be written as the intersection of the failure domains of the individual components:

$$\mathcal{F}_p = \left\{ \bigcap_{k=1}^{K} g_k(\mathbf{x}) \leq 0 \right\} \tag{3.62}$$

A cut set formulation is a more general definition of a system problem than the above two special cases. A cut set is any set of components whose joint failure constitutes failure of the system. A minimum cut set is a cut set that ceases to be a cut set if any of its components are removed. The failure domain of such a system is defined as:

$$\mathcal{F}_c = \left\{ \bigcup_{k=1}^{K} \bigcap_{k \in C_k} g_k(\mathbf{x}) \leq 0 \right\} \tag{3.63}$$

where C_k is the k-th index set defining the component indices in the k-th cut set, and the union is over all the cut sets of the system.

Estimation of system failure probabilities is not a trivial task. In OpenSees, the problem is currently addressed by crude Monte Carlo sampling analysis and by computing probability bounds

for series systems. Sampling analysis is performed as described in the previous section with the indicator function I defined in terms of the system failure rather than a component failure. Bounds on the probability of failure of series systems are estimated according to the so-called KHD bounds (Kounias 1968, Hunter 1976, Ditlevsen 1979):

$$\text{Lower bound:} \qquad P_{f,\mathrm{S}} \geq P_1 + \sum_{k=2}^{K} \max \left\{ P_k - \sum_{l=1}^{k-1} P_{kl},\ 0 \right\} \tag{3.64}$$

$$\text{Upper bound:} \qquad P_{f,\mathrm{S}} \leq P_1 + \sum_{k=2}^{K} \left\{ P_k - \max_{l<k} P_{kl} \right\} \tag{3.65}$$

P_k is the probability of failure of the k-th performance function and P_{kl} is the probability of joint failures of the k-th and l-th components. The latter requires a parallel system reliability analysis with two components. In FORM, this is approximated by $P_{kl} \approx \Phi\left(-\beta_i, -\beta_j, \rho_{ij}\right)$ where $\Phi\left(\cdot, \cdot, \rho\right)$ is the bi-normal CDF and is computed from (Ditlevsen and Madsen 1996):

$$\begin{aligned} \Phi\left(-\beta_i, -\beta_j,\ \rho_{ij}\right) &= \Phi\left(-\beta_i\right)\ \Phi\left(-\beta_j\right) \\ &\quad + \int_0^{\rho_{ij}} \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left[-\frac{\beta_i^2 + \beta_j^2 - 2\rho\beta_i\beta_j}{2\left(1-\rho^2\right)} \right] \mathrm{d}\rho \end{aligned} \tag{3.66}$$

where the correlation coefficient is given in terms of the respective alpha vectors: $\rho_{ij} = \boldsymbol{\alpha}_i \boldsymbol{\alpha}_j^T$.

3.4.6 Mean Out-Crossing Rate by FORM

The time- and space-variant reliability problem is a difficult challenge in structural reliability analysis. The time or location of the maximum response is itself a random variable. It can, therefore, not be specified a priori by the user. In time-variant reliability analysis a fundamental problem is the first excursion problem. It stems from our desire to estimate the probability of a time-varying response process $\mathbf{x}(t)$ entering the failure domain $g(\mathbf{x}(t),\, t) \leq 0$ over a finite time interval T:

$$p_f(T) = P\left(\left\{ \min_{0 \leq t \leq T} g(\mathbf{x}(t),\, t) \right\} \leq 0 \right) \tag{3.67}$$

The input for this problem is defined in terms of the discretized stochastic process defined in Section 3.2.5 and Eq. (3.32). One solution to this problem can be found by defining a limit-state function at every time step of the analysis and solve it as a series system reliability problem. At each time instant the problem is reduced to a time-invariant component problem, which can be treated with methods such as FORM and IS analysis. However, it is readily seen that this represents a costly

analysis approach in a nonlinear finite element context. This approach has been used by Au and Beck (2001) and Koo and Der Kiureghian (2003). An alternative is to employ an Active Set Gradient Projection scheme, as suggested by Zhang and Der Kiureghian (1994). The approach employed here is based on the earlier works of Li and Der Kiureghian (1995), Der Kiureghian and Li (1996), Der Kiureghian (2000), and Koo and Der Kiureghian (2003), and estimates the mean out-crossing rate, which is a critical response statistic for time-variant reliability analysis. Various reliability measures, such as the upper bound to the exceedance probability during a time interval T, are available as outlined below.

The out-crossing rate, denoted $\eta(t)$, marks the number of times per unit time interval that the response vector process $\mathbf{x}(t)$ makes a transition from the safe state into the failure state. For a stochastic input, this number is random and varies with time. Our interest is in computing its mean value, $\nu(t) = \mathrm{E}\left[\eta(t)\right]$, as a function of time.

Consider the two events $g(\mathbf{x}(t),\, t) > 0$ and $g(\mathbf{x}(t+\delta t),\, t+\delta t) \leq 0$, indicating the occurrence of one or more out-crossings into the failure domain during $(t,\ t+\delta t)$. Two auxiliary limit-state functions are now defined: $\tilde{g}_1 = -g(\mathbf{x}(t), t)$ and $\tilde{g}_2 = g(\mathbf{x}(t), t) + \dot{g}\ \delta t$ where $\dot{g} = \frac{\partial g}{\partial t}$. $\tilde{g}_2$ is the linear Taylor expansion of $g(\mathbf{x}(t+\delta t), t+\delta t)$. The probability of an out-crossing can be written as the probability of the intersection of the failure events of $\tilde{g}_1$ and $\tilde{g}_2$:

$$p_f(t,\, t+\delta t) = \mathrm{P}[\tilde{g}_1 \leq 0 \bigcap \tilde{g}_2 \leq 0] \tag{3.68}$$

The mean rate of out-crossings is written as (Hagen and Tvedt 1991):

$$\begin{aligned} \nu(t) &= \lim_{\delta t \to 0} \frac{\mathrm{P}[\tilde{g}_1 \leq 0 \ \bigcap \ \tilde{g}_2 \leq 0]}{\delta t} \\ &\approx \frac{\mathrm{P}[\tilde{g}_1 \leq 0 \ \bigcap \ \tilde{g}_2 \leq 0]}{\delta t} \ ,\ \delta t \text{ small} \end{aligned} \tag{3.69}$$

The numerator in Eq. (3.69) represents a parallel system reliability problem with two components. This problem is discussed in Section 3.4.5; the numerator in Eq. (3.69) is addressed by Eq. (3.66). However, the numerator in Eq. (3.69) represents a special case with high negative correlation between $\tilde{g}_1$ and $\tilde{g}_2$ and $\beta_i \approx -\beta_j$. Therefore care must be exercised when evaluating the integral in Eq. (3.66). Koo and Der Kiureghian (2003) developed the following approximate expression to solve Eq. (3.66) for this special case:

$$\Phi\left(\beta_i, -\beta_i,\, -1+\delta\rho\right) \approx \frac{1}{2\pi} \exp\left(-\frac{\beta_i^2}{2}\right) \left[\sin^{-1}\left(-1+\delta\rho\right) - \sin(-1)\right] \tag{3.70}$$

In OpenSees, two approaches are implemented to evaluate the parallel probability at hand. One alternative is to separately obtain the design point of $\tilde{g}_1$ and $\tilde{g}_2$. The expression for $\tilde{g}_2$ is found by

$\tilde{g}_2 = g + \dot{g}\ \delta t$. For example, for a performance function expressed in terms of displacement quantities, the chain rule of differentiating is applied: $\dot{g}\ \delta t = \frac{\partial g}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial t}\ \delta t = \frac{\partial g}{\partial \mathbf{u}} \dot{\mathbf{u}}\ \delta t$, effectively introducing velocity response quantities in the expression for the second performance function.

The alternative method proposed by Koo and Der Kiureghian (2003) circumvents the cost of searching for the design point of $\tilde{g}_2$ by the following observation. Assume the performance function g is of the threshold type, i.e., $g = u_o - u$, where u_o is the threshold and u is a response quantity. When the design point is found for $\tilde{g}_1$, the design point realization of the response shows a displacement u equal to u_o at time t. Similarly, the design point realization for $\tilde{g}_2$ must show a displacement u equal to u_o at time $t + \delta t$. A simple approximation to the design point of $\tilde{g}_2$ is obtained by simply shifting the input time series (the realization of random pulses defined in Eq. (3.32)) of the design point of $\tilde{g}_1$ along the time axis by a value of δt as illustrated in Figure 3.1. This facilitates the use of Eq. (3.70) to evaluate the parallel probability in Eq. (3.69). The error of this approximation is small for small δt and is mainly present at the beginning of the time series, which is not a critical segment of the excitation for systems with damping. Numerical examples have shown that for δt small in relation to the predominant period of the response, this method provides sufficiently accurate results for all practical purposes.

Upon determination of the mean out-crossing rate $\nu(t)$ at discrete points along the time axis, the upper bound to the probability of excursion into the failure domain during time interval T is:

$$\tilde{p}_f(T) = \int_0^T \nu(t)\ \mathrm{d}t \tag{3.71}$$

In cases where the out-crossing events may be assumed independent, an approximation to the true failure probability is derived by the assumption of Poisson distributed out-crossing events (Li and Der Kiureghian 1995):

$$p_f(T) = 1.0 - e^{-\int_0^T \nu(t)\ \mathrm{d}t} \tag{3.72}$$

Caution must be exercised when using Eq. (3.72). For narrow-band processes, the out-crossings occur in clusters. This violates the assumptions of independence between out-crossing events. Furthermore, the assumption may be invalid if structural properties are assumed random in addition to the discretized stochastic input. In these cases, the upper bound in Eq. (3.71) should be used.

3.5 FINDING THE DESIGN POINT

As we have seen, the design point plays an essential role in reliability analysis. By definition, it is the point on the limit-state surface in the standard normal space with highest probability density. This is the ideal point for approximating the limit-state surface in FORM and SORM, and the ideal point to center the sampling distribution in importance sampling.

The design point is the solution to the constrained optimization problem:

$$\mathbf{y}^* = \text{argmin}\,\{\, \|\mathbf{y}\| \;\mid\; G(\mathbf{y}) = 0\} \tag{3.73}$$

where $\mathbf{y}$ is the vector of random variables in the standard normal space, $\mathbf{y}^*$ is the design point, G is the performance function in this space and "argmin" denotes the argument of the minimum of a function.

In the optimization literature, see, e.g., Polak (1997), it is common to develop algorithms for constrained optimization problems with inequality constraints. This is a different problem than Eq. (3.73), where the constraint is an equality constraint. For this reason several of the algorithms presented in this work address the following alternative problem:

$$\mathbf{y}^* = \min\,\{\, \|\mathbf{y}\| \;\mid\; G(\mathbf{y}) \leq 0\} \tag{3.74}$$

When the origin in the standard normal space is in the safe domain, then Eqs. (3.73) and (3.74) lead to equivalent solutions. However, when the origin is in the failure domain, that is, when $g(\|\mathbf{y}\| = 0) < 0$, then the solution of Eq. (3.74) is the origin and not the actual design point that is the solution of Eq. (3.73). Hence, caution must be exercised when using the formulation in Eq. (3.74).

The degree of difficulty involved in determining the design point $\mathbf{y}^*$ depends on the problem at hand. In finite element reliability applications the performance function g is defined in terms of response quantities from a finite element analysis. This may introduce a number of challenges:

1. The limit-state function g may exhibit nonlinearities due to nonlinearities in the finite element responses entering the performance function.

2. Local nonlinearities, or "noise," in g may be experienced due to numerical approximations in the finite element solution. These are regarded as "inner" approximations, while the design point convergence tolerances in the reliability analysis are regarded as "outer" approximations.

3. The gradient of the performance function, which is needed in the search for the the design point, may exhibit discontinuities. This was demonstrated in Chapter 1.5.

4. It is well known that a non-linear finite element analysis may not converge to equilibrium. This is the case, for instance, when the choice of model parameters constitutes a nonphysical configuration. Hence, for some realizations of the random variables it may not be possible to evaluate the performance function. In the worst case the computer program terminates due to lack of convergence in the finite element solution. It is reasonable to assume that this situation may occur mostly in the failure domain, where gross nonlinearities in structural response are expected.

The above challenges are addressed in the presented work.

3.5.1 The General Search Scheme

The so-called HLRF algorithm originally developed by Hasofer and Lind (1974) and later extended to non-normal random variables by Rackwitz and Fiessler (1978) is perhaps the most popular algorithm used to solve the constrained optimization problem of Eq. (3.73) in structural reliability analysis. It is well known that this original form of the algorithm is unstable and may not converge under certain conditions. Liu and Der Kiureghian (1991b) and Zhang and Der Kiureghian (1997) improved this algorithm by adding a line search scheme. This improved algorithm, denoted iHLRF, is presented in the following together with several alternatives. Of main interest is application of the algorithm to the challenging problems of nonlinear finite element reliability analysis. Modifications to accommodate challenges particular to such problems are proposed.

The key steps are the same for all algorithms considered here. They are implemented in OpenSees as follows:

1. Transform the user-given starting point $\mathbf{x}_1$ in the original space into the corresponding point $\mathbf{y}_1$ in the standard normal space. This is done according to Eq. (3.24).

2. Compute and store as G_o the value of the performance function $g(\mathbf{x}_1)$. This is done for scaling purposes. If the start point is close to the limit-state surface then $G_o = 1.0$ is used. This issue is further discussed in the next section.

3. While convergence is not achieved and the user-specified maximum number of iterations is not reached:

(a) Back-transform vector $\mathbf{y}_i$ from the standard normal space to $\mathbf{x}_i$ in the original space. The details of this transformation are discussed in Section 3.2.4. Skip this step in the first iteration, where $\mathbf{x}_1$ is already available.

(b) Compute the value of the performance function $g(\mathbf{x}_i)$ by executing the OpenSees finite element code. Note that the updated realization of the random variables must be given to the finite element code. The interface between the finite element code and the reliability algorithm is described in Section 3.7.

(c) Along with the finite element response, gradients are computed by the DDM within OpenSees or by finite differences. For performance functions only involving displacement quantities, the needed gradient vector is $\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$. Elements of this vector enter into the following expression for the needed gradient of the performance function: $\nabla_{\mathbf{y}} G = \frac{\partial g}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \mathbf{y}} = \frac{\partial g}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial \mathbf{x}} \mathbf{J}_{\mathbf{x},\,\mathbf{y}} = \frac{\partial g}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial \mathbf{x}} \left(\mathbf{J}_{\mathbf{y},\,\mathbf{x}}\right)^{-1}$. The gradient $\frac{\partial g}{\partial \mathbf{u}}$ is easily found, since g is normally an algebraic function of $\mathbf{u}$. The Jacobian $\mathbf{J}_{\mathbf{y},\,\mathbf{x}}$ is found as outlined in Section 3.2.4.

(d) Check convergence according to convergence criteria discussed in Section 3.5.2.

(e) Take a step if convergence is not achieved: $\mathbf{y}^{(m+1)} = \mathbf{y}^{(m)} + \lambda\, \mathbf{d}$, $\mathbf{d}$ is the search direction vector and λ is the step size.

The essential difference between the algorithms presented in the subsequent sections is the selection of the search direction $\mathbf{d}$ and step size λ.

3.5.2 Convergence Criteria

None of the algorithms presented in this report are guaranteed to find the global design point. This is a fundamental challenge in optimization. However, proofs of convergence may be established for local solutions satisfying certain convergence criteria or optimality conditions. The local solution most often corresponds to the actual global solution. Usually this can be checked from the context of the problem. When in doubt, one should repeatedly solve the problem with different starting points to gain confidence that the solution is the global one.

Two convergence criteria must be met in our case. First, the point should be located on the limit-state surface characterized by $G = 0$. Second, the point should be as close as possible to the

origin. In equation form, the first criterion reads:

$$\left| \frac{G}{G_o} \right| < e_1 \tag{3.75}$$

where G_o is a scaling value usually selected as the value of the performance function at the start point. In OpenSees, the user may alter this by providing G_o in the input. This option is useful for restarts of the search, where the limit-state function value at the start point is already small. The constant e_1 is a user-defined acceptance tolerance. A common choice is $e_1 = 10^{-3}$.

The second convergence criterion may be expressed by considering the two vectors $\mathbf{y}$ and $\boldsymbol{\alpha} = -\frac{\nabla G}{\|\nabla G\|}$. At the design point the gradient vector must point towards the origin in the standard normal space. Thus, $\boldsymbol{\alpha}$ and $\mathbf{y}$ must be collinear at the design point. This convergence criterion may be expressed as the vector difference between $\mathbf{y}$ and the component of $\mathbf{y}$ in the direction of $\boldsymbol{\alpha}$. The latter quantity is expressed as $\boldsymbol{\alpha}\mathbf{y}\boldsymbol{\alpha}^T$; Figure 3.2. The criterion is then written as:

$$\|\mathbf{y} - \boldsymbol{\alpha}\mathbf{y}\boldsymbol{\alpha}^T\| < e_2 \tag{3.76}$$

where e_2 is a user-defined acceptance tolerance, commonly selected as 10^{-3} (Liu *et al.* 1989).

In this report the convergence criterion in Eq. (3.76) is modified. The criterion is meant to be a measure of how coincident the vectors $\boldsymbol{\alpha}$ and $\mathbf{y}$ are. It is clear from Figure 3.2 that the criterion in Eq. (3.76) becomes stricter when the magnitude of the vector $\mathbf{y}$ increases. This is because it is the difference vector that is measured and not the angle of deviation between the two vectors. This may cause lack of convergence when the value of β is large. Two remedies are suggested.

In the first approach, the $\mathbf{y}$ vector is normalized to unit length for the purpose of the convergence check. That is, it is assumed that the convergence criterion in Eq. (3.76) is appropriate at a unit distance from the origin in the $\mathbf{y}$ space. The following revised criterion is introduced so that the requirement is consistent, independent of the distance of the trial point from the origin:

$$\left\| \frac{\mathbf{y}}{\|\mathbf{y}\|} - \left(\boldsymbol{\alpha} \frac{\mathbf{y}}{\|\mathbf{y}\|} \right) \boldsymbol{\alpha}^T \right\| < e_2 \tag{3.77}$$

This is implemented as the standard second convergence criterion in OpenSees, but with the following condition: if $\|\mathbf{y}\| < 1.0$, then the original Eq. (3.76) is employed.

The second alternative is to consider the angle between the vectors $\boldsymbol{\alpha}$ and $\mathbf{y}$. From elementary vector geometry we know that the cosine of this angle, denoted θ, is given by:

$$\cos(\theta) = \frac{\boldsymbol{\alpha}\mathbf{y}}{\|\mathbf{y}\|} \tag{3.78}$$

Since θ is equal to zero at the design point, $\cos(\theta)$ must be equal to unity. Hence, an alternative to the criterion in Eq. (3.77) is:

$$1 - \frac{\boldsymbol{\alpha}\mathbf{y}}{\|\mathbf{y}\|} < e_2 \tag{3.79}$$

It is noted that Eq. (3.79) is related to the optimality conditions of the problem in Eq. (3.73) in the following way. A Lagrange formulation of the optimization problem in Eq. (3.73) can be written as $l = \frac{1}{2}\|\mathbf{y}\|^2 + \gamma G(\mathbf{y}) = 0$. The first optimality condition is $g = 0$, while the second condition is obtained by requiring that the gradient of the Lagrangian be zero at the design point, namely

$$\nabla l = \mathbf{y} + \gamma \nabla G^T = 0 \tag{3.80}$$

It is clear that a Lagrange multiplier $\gamma = \frac{\|\mathbf{y}\|}{\|\nabla G\|}$ scales the two vector terms such that the sum in Eq. (3.80) is zero if the vectors are parallel:

$$\frac{\mathbf{y}}{\|\mathbf{y}\|} + \frac{\nabla G^T}{\|\nabla G\|} = 0 \tag{3.81}$$

By multiplying Eq. (3.81) by $\frac{\mathbf{y}^T}{\|\mathbf{y}\|}$ and using $\boldsymbol{\alpha} = -\frac{\nabla G}{\|\nabla G\|}$, Eq. (3.79) is obtained. Hence, Eq. (3.79) can be regarded as a check on one of the optimality conditions of the problem in Eq. (3.73).

3.5.3 Step Size Selection and Restricting the Search to the Safe Domain

While the search direction is the most distinguishing characteristic of a search scheme, the step size selection is equally important in nonlinear finite element reliability applications. The algorithms outlined in this report are all termed "line search" schemes. This is due to the posterior step size selection along a pre-selected search direction. Ideally, the step size is determined so that a so-called merit function is minimized. Usually, this leads to an optimal rate of convergence. However, this strategy poses a new optimization problem, namely the minimization of the merit function along the search direction vector. A common strategy is therefore to employ the Armijo rule (Polak 1997). With this rule the step size is selected as:

$$\lambda = b^k \tag{3.82}$$

The user selects a value $0.0 < b < 1.0$, while k is an integer with initial value 0. k increases by unit steps until an acceptable step size is found, that is, until the selected merit function value decreases by a certain amount. See the subsequent sections for specific merit functions.

A typical value for b is 0.5, in which case the step size is divided in half each time a trial step size is rejected, that is, for each increment of k. As seen, the initial step size with the Armijo

rule is $\lambda_o = b^0 = 1.0$. This is altered in OpenSees, since an initial step size of 1.0 may lead to trial points too far out in the failure domain, with resulting non-convergence in the finite element analysis. Figure 3.3 illustrates the problem though the force-displacement curves of a hypothetical structure at successive trial points. Usually, the structure is more nonlinear at the design point than at the start point, which is often the mean point. For this reason the search algorithm often overestimates the necessary reduction of random strength/stiffness and the increase in random load variables. Hence, it is often experienced that the first trial step of the search algorithm falls too far in the nonlinear domain, which sometimes results in non-convergence of the finite element code.

Several remedies for this problem are implemented in OpenSees. First, the step size of the Armijo rule is implemented as

$$\lambda = b_o \cdot b^k \tag{3.83}$$

The user-given factor b_o is by default equal to 1.0, but it gives the user the opportunity to force the initial step size to be different from 1.0. The user also determines the number of steps for which this modification is in effect. This simple modification has proven to be effective for successful convergence of many nonlinear finite element reliability analyses in OpenSees. Furthermore, the Armijo rule implemented in OpenSees in many cases is able to detect whether the finite element analysis converges or not. This depends on how severe the error event in the finite element code is. In case of a recoverable non-convergence event, the trial point is rejected and a step size reduction is invoked according to the ordinary rule $k = k + 1$.

The second remedy to avoid random variable realizations in the domain where the finite element code cannot converge is a "bounding sphere." The user selects the radius of an initial hyper-sphere, within which the trial point is restricted to stay. An initial guess of the radius of the sphere could be, for instance, $\beta_{sphere} = 2.0$. This choice of course depends on the expected reliability index. During the search the sphere is gradually allowed to grow if the design point is not found inside it. Rules for the evolution of the size of the sphere are implemented in OpenSees. These are described in Chapter 3.7.

Another alternative is to modify the algorithm that selects the search direction vector so that the search is performed within the safe domain. Several such suggestions are made in the subsequent sections. These include a modification to the iHLRF algorithm, and the introduction of a special case of the Polak-He algorithm, which makes use of certain steering parameters that are useful for this purpose.

3.5.4 The Gradient Projection Algorithm

The Gradient Projection algorithm has the simplest features of the algorithms presented in this work and it addresses Eq. (3.73). Its main characteristic is that it performs the search along the limit-state surface, namely, in the sub-domain where $G = 0$. Of course, for non-linear limit-state functions it is not possible to immediately find a trial point exactly on the limit-state surface. Instead, a guess is made based on the assumption that the limit-state surface is linear and a subsequent root-finding algorithm is employed to bring the trial point onto the limit-state surface. The fundamental concepts of the algorithm are first presented, followed by a discussion on possible remedies to ensure that the search is restricted to the safe domain.

An expression for the initial search direction $\mathbf{d}$ at each step is derived by requiring that it be perpendicular to ∇G and that it lie in the plane spanned by ∇G and $\mathbf{y}$; see Figure 3.4. The first requirement translates into $\nabla G\,\mathbf{d} = 0$. The second reads $\mathbf{d} = a\ \mathbf{y} + b\ \nabla G^T$ where a and b are unknown constants. By selecting $\mathbf{d}$ as in Figure 3.4, namely such that $\mathbf{d}$ is the solution for the linearized function, we have that $b\nabla G^T = \mathbf{y} + \mathbf{d}$. Hence, $a = -1$. b can be solved from the requirement $\nabla G\,\mathbf{d} = 0$:

$$\nabla G\,\mathbf{d} = \nabla G\left(-\mathbf{y} + b\nabla G^T\right) = -\nabla G\,\mathbf{y} + b\|\nabla G\|^2 = 0 \quad \Rightarrow \quad b = \frac{\nabla G\,\mathbf{y}}{\|\nabla G\|^2} \tag{3.84}$$

The search direction vector comes out as:

$$\mathbf{d} = -\mathbf{y} + \frac{\nabla G\,\mathbf{y}}{\|\nabla G\|^2}\nabla G^T = -\left[\mathbf{I} - \boldsymbol{\alpha}^T\boldsymbol{\alpha}\right]\mathbf{y} \tag{3.85}$$

Note that $\boldsymbol{\alpha}^T\boldsymbol{\alpha}$ yields a matrix and that $\mathbf{I}$ is the identity matrix.

As mentioned earlier, unless the limit-state function is linear the new trial point will generally not be located on the limit-state surface. This is remedied by employing a root-finding algorithm. The problem is formulated as finding the root $\mathbf{y}$ of the function $G(\mathbf{y}) = 0$ along the direction ∇G starting from the trial point. Several algorithms are available for this purpose. Two well-known methods are employed in this work and implemented in OpenSees, namely, the secant method and the Modified Newton method. Both these methods use the recursive rule:

$$\mathbf{y}^{(m+1),(p+1)} = \mathbf{y}^{(m+1),(p)} - \frac{G(\mathbf{y}^{(m+1),(p)})}{k}\frac{\nabla G(\mathbf{y}^{(m)})^T}{\|\nabla G(\mathbf{y}^{(m)})\|} \tag{3.86}$$

Superscript m is the step number in the search for the design point, p is the counter in the root-finding scheme and k is the tangent variable. The last quotient in the above expression is the normalized direction vector, namely ∇G from the previous trial point of the global search algorithm. The distinguishing factor between the two algorithms is the tangent variable k.

In the Modified Newton method, k is equal to the norm of the gradient vector from the previous trial point, namely $\|\nabla G(\mathbf{y}^{(m)})\|$, as shown in Figure 3.5. In the Modified Newton algorithm the tangent is not updated. This may make this scheme less efficient than the strict Newton method, but overall less costly, since it involves updating the gradient.

The secant method is characterized by updating k based on the previous values of the limit-state function according to:

$$k = \frac{G(\mathbf{y}^{(m+1),(p)}) - G(\mathbf{y}^{(m+1),(p-1)})}{\|\mathbf{y}^{(m+1),(p)} - \mathbf{y}^{(m+1),(p-1)}\|} \tag{3.87}$$

The question arises as to how to find the initial point on the limit-state surface for the first step of the search for the design point. The user usually is unable to specify a point on the surface as a start point. If the user selects the origin in the standard normal space as the starting point, then $\mathbf{y} = \mathbf{0}$ and $\mathbf{d} = \mathbf{0}$ and the algorithm fails.

In OpenSees, a direction towards the limit-state surface is found by solving the linearized problem. As will be discussed in Section 3.5.5, the design point for the linearized performance function is found by employing the search direction vector $\mathbf{d} = \left(\frac{G}{\|\nabla G\|} + \boldsymbol{\alpha}\,\mathbf{y}\right)\boldsymbol{\alpha}^T - \mathbf{y}$. The first trial point is found by applying the root-finding scheme along this direction vector. That is, the first trial point in the gradient projection algorithm is found by the iterative scheme:

$$\mathbf{y}^{(p+1)} = \mathbf{y}^{(p)} + \frac{G(y^{(p)})}{k}\frac{\left(\frac{G}{\|\nabla G\|} + \boldsymbol{\alpha}^T\,\mathbf{y}\right)\boldsymbol{\alpha} - \mathbf{y}}{\left\|\left(\frac{G}{\|\nabla G\|} + \boldsymbol{\alpha}^T\,\mathbf{y}\right)\boldsymbol{\alpha} - \mathbf{y}\right\|} \tag{3.88}$$

The selection $k = \|\nabla G\mathbf{y}^{(m)}\|$ for the Modified Newton method is used since the gradient vector in the direction of $\mathbf{d}$ is not available.

As described in Section 3.5.3, a general feature of the search algorithms is that a step size is selected along the selected search direction by monitoring a merit function. One may develop a merit function appropriate for the gradient projection algorithm. However, this is not as effective as for other algorithms. The search direction in the gradient projection algorithm is always selected to be tangent to the limit-state surface. Thus, for non-linear performance functions the first trial point of the next step will most likely be "worse," at least for one of the optimality conditions, than the previous step. Hence, the usual concept of the search direction being in a descent direction of a merit function is not useful here. One might argue that a merit function could be developed involving the $\boldsymbol{\alpha}$-$\mathbf{y}$-collinear convergence criterion. However, guarantee of convergence could still not be proven and it is doubtful that the convergence rate would be much improved.

In the implementation of the Gradient Projection algorithm in OpenSees, the user may select either a fixed step size or an Armijo rule with one of the available merit function implementations. If

a fixed step size is selected, the search directions are computed such that the trial points are located on the limit-state surface, regardless of the value of the step size λ. If the Armijo rule is employed, all trial points in the line search are projected onto the limit-state surface. That is, the line search is actually performed along the limit-state surface. Since the merit-function check is separated from the step size rule this allows for a flexible line search approach for the gradient projection algorithm.

3.5.5 The Improved HLRF Algorithm

This is perhaps the most popular algorithm used to solve the problem in Eq. (3.73). The original form was developed by Hasofer and Lind (1974) and Rackwitz and Fiessler (1978). This was later modified and improved by Liu and Der Kiureghian (1991b) and Zhang and Der Kiureghian (1997) by adding a line search scheme.

The search direction for this algorithm is considered as an extension of the gradient projection search direction in Eq. (3.85). While the gradient projection algorithm assumes that the trial point is located on the limit-state surface, the iHLRF algorithm does not. It is shown in Figure 3.6 how this leads to an additional term in the expression for the search direction vector, namely $\frac{G}{\|\nabla G\|^2}\nabla G_k^T$. This term is naturally the same as the step in Eq. (3.86) with a Newton-type root-finding scheme. The complete expression for the search direction vector reads:

$$\mathbf{d} = -\mathbf{y} - \left(\frac{\nabla G\,\mathbf{y}}{\|\nabla G\|^2}\right)\nabla G^T - \left(\frac{G}{\|\nabla G\|^2}\right)\nabla G^T \tag{3.89}$$

With this choice of search direction vector and an initial step size $\lambda = 1$, the design point is found in one step for linear performance functions. Eq. (3.89) is implemented in OpenSees in the form:

$$\mathbf{d} = \left(\frac{G}{\|\nabla G\|} + \boldsymbol{\alpha}\,\mathbf{y}\right)\boldsymbol{\alpha}^T - \mathbf{y} \tag{3.90}$$

For an Armijo rule of the form

$$m\left(\mathbf{y}^{(m+1)}\right) - m\left(\mathbf{y}^{(m)}\right) \leq +a\,\lambda\,\left(\nabla m\left(\mathbf{y}^{(m)}\right)^T\,\mathbf{d}\right) \tag{3.91}$$

the following merit function was suggested by Zhang and Der Kiureghian (1997):

$$m\left(\mathbf{y}\right) = \frac{1}{2}\left\|\mathbf{y}\right\|^2 + c\,|G| \tag{3.92}$$

Several remarks are made regarding this merit function check:

1. The constant $a > 0$ is defined by the user. It is a common parameter for the merit function checks usually associated with Armijo-type line search schemes. It influences how much the user wants the merit function to decrease at each step. A typical value is $a = 0.5$.

2. Note the plus sign in Eq. (3.91). This is to emphasize that the minus sign found in this position in Zhang and Der Kiureghian (1997) is a misprint. As shown by Zhang and Der Kiureghian (1997), the inner product $\nabla_{\mathbf{y}} m \left(\mathbf{y}^{(m)}\right)^T \mathbf{d}$ is negative. Thus, the right-hand side of Eq. (3.91) indicates the decrease in the merit function value at the new trial point compared to the value of the merit function at the previous trial point.

3. Zhang and Der Kiureghian (1997) have shown that the search direction vector in Eq. (3.90) is a descent direction of the merit function in Eq. (3.92). However, the proof assumes the penalty parameter c in Eq. (3.92) to be a constant along the search direction vector. For this reason the gradient of the merit function reads: $\nabla m = \mathbf{y} + c\,\nabla G\,\mathrm{sgn}(G)$.

4. The penalty parameter c is not uniquely defined. Its selected value may influence the convergence rate. It is shown in Zhang and Der Kiureghian (1997) that the condition $c \geq \frac{\|\mathbf{y}\|}{\|\nabla G\|}$ must be satisfied for the search direction to be a descent direction of the merit function. In the present implementations in OpenSees the penalty parameter c is selected as

$$c = \gamma \frac{\|\mathbf{y}\|}{\|\nabla G\|} + \eta \tag{3.93}$$

   where $\mathbf{y}$ and ∇G are values at the previous trial point. The constants $\gamma > 1$ and $\eta \geq 0$ are specified by the user to ensure that the above-stated condition is satisfied. Currently, γ and η may be varied through the analysis only by stopping the search and restarting it from the last trial point with new values for γ and η. In OpenSees, the default values are $\gamma = 2$ and $\eta = 10$.

3.5.6 The Polak-He Algorithm

This is a general nonlinear optimization algorithm developed by Polak and He (1991) and Polak (1997). It is also discussed by Royset (2002). It can be specialized to solve the problem in Eq. (3.74). This algorithm has not been used in the reliability community. However, it has features which make it attractive for finite element reliability analysis. Its main advantage is that it has steering parameters to force the search to be performed in the safe domain. These features are explored in this study. It is noted, however, that the Polak-He algorithm only possesses linear convergence properties. This

implies that it is scale-variant: the numerical value of the performance function matters. For this reason the user should scale the performance function so that the starting value is of the order of 10.

The algorithm presented in Polak (1997) aims at solving a constrained optimization problem with p number of cost functions and q number of constraints. The procedure involves solving an unconstrained optimization problem prior to obtaining the search direction vector at each step. The unknowns in the unconstrained optimization problem are $\mu_o,\ \mu_1,\ \mu_2,\ \ldots, \mu_q$ and $\nu_1,\ \nu_2,\ \ldots, \nu_p$. They are subject to the conditions:

$$\sum_{j=0}^{q} \mu_j = 1, \quad \sum_{k=1}^{p} \nu_k = 1, \quad 0 \leq \mu_j \leq 1 \quad \text{and} \quad 0 \leq \nu_k \leq 1 \tag{3.94}$$

Our problem in Eq. (3.74) is characterized by $p = q = 1$. Hence, $\nu_1 = 1$ and the unconstrained minimization problem takes the form:

$$\theta = -\min_{\mu_o, \mu_1} \left\{ \mu_o \gamma\, G_+ + \mu_1 \left(G_+ - G\right) + \frac{1}{2\,\delta} \left(\mu_o \mathbf{y} + \mu_1\, \nabla G\right)^2 \right\} \tag{3.95}$$

where $G_+ = \max\{0,\ G\}$ and γ and δ are user-defined parameters discussed in the following. This unconstrained optimization problem can be formulated in matrix form as follows:

$$\theta = -\min_{\boldsymbol{\mu}} \left\{ \boldsymbol{\mu}^T \mathbf{A} \boldsymbol{\mu} + \mathbf{b}^T \boldsymbol{\mu} \right\} \tag{3.96}$$

where: $\mathbf{A} = \frac{1}{2\,\delta} \begin{bmatrix} \mathbf{y}^T\mathbf{y} & \nabla G \mathbf{y} \\ \nabla G \mathbf{y} & \nabla G \nabla G^T \end{bmatrix}$ and $\mathbf{b} = [\gamma G_+ \quad G_+ - G]$.

Due to the first condition in Eq. (3.94), the unconstrained minimization problem has its solution along the line $\mu_o = 1 - \mu_1,\ 0 \leq \mu_j \leq 1$. Thus, a single auxiliary unknown variable can be expressed as $x = \mu_o = 1 - \mu_1$. The problem in Eq. (3.95) is then reformulated into $\theta = -\min_x \{a\, x^2 + b\, x + c\}$, where the constants a, b and c are found as extracts from $\mathbf{A}$ and $\mathbf{b}$ in Eq. (3.96): $a = \frac{1}{2\delta}\mathbf{y}^T\mathbf{y} + \frac{1}{2\delta}\nabla G \nabla G^T - \frac{1}{\delta}\nabla G \mathbf{y}$, $b = \gamma G_+ - (G_+ - G) + \frac{1}{\delta}\nabla G \mathbf{y} - \frac{1}{\delta}\nabla G \nabla G^T$ and $c = \frac{1}{2\delta}\nabla G \nabla G^T + (G_+ - G)$. The extremum point is found as $x = -\frac{b}{2\,a}$ unless $a = 0$, in which case the function is linear and the extremum is found at either of the end points $\mu_o = 0$ or $\mu_1 = 0$. One of the end points may in any case be the correct solution; this is checked in OpenSees before accepting the solution.

Upon solving for μ_o and μ_1, the search direction vector is obtained as:

$$\mathbf{d} = -\mu_0\, \mathbf{y} - \mu_1\, \nabla G^T \tag{3.97}$$

In Polak (1997) the merit function criterion is written as follows:

$$m\left(\mathbf{y}^{(m)},\ \mathbf{y}^{(m+1)}\right) \leq +a\, \lambda\, \theta \tag{3.98}$$

where θ is the solution of Eq. (3.95). The composite merit function m is defined as follows:

$$m\left(\mathbf{a}\,,\,\mathbf{b}\right) = \max\left\{\frac{1}{2}\|\mathbf{b}\|^2 - \frac{1}{2}\|\mathbf{a}\|^2 - \gamma\, G(\mathbf{a})_+ \;\;,\;\; G(\mathbf{b}) - G(\mathbf{a})_+\right\} \tag{3.99}$$

As mentioned earlier, an attractive feature of the Polak-He algorithm is that the user may influence the search path. In fact, the γ parameter in the equations above is used for this purpose. Both γ and δ have 1.0 as their default values. γ can be increased to make the trial steps approach the failure domain faster. However, that is the opposite of what is desired in finite element analysis. Hence, a small value, e.g., $\gamma = 0.1$, is recommended for highly non-linear finite element problems. In the OpenSees implementation, the parameters γ and δ are user-selected parameters. The parameter δ could alternatively be set to 1.0 in the first step and subsequently re-computed according to the following expression (Polak 1997):

$$\delta = \frac{\frac{1}{2}\|\mathbf{y}^{(m+1)}\|^2 - \frac{1}{2}\|\mathbf{y}^{(m)}\|^2 - \mathbf{y}^{(m)^T}\left(\mathbf{y}^{(m+1)} - \mathbf{y}^{(m)}\right)}{\|\mathbf{y}^{(m+1)} - \mathbf{y}^{(m)}\|^2} \tag{3.100}$$

3.5.7 The Sequential Quadratic Programming (SQP) Algorithm

This is the most complex of the algorithms presented in this report for solving the problem in Eq. (3.73). It is often recommended to use a well-tested library implementation of this algorithm. In this work the SQP algorithm presented in Liu and Der Kiureghian (1991a) for our particular optimization problem is implemented in OpenSees.

The sequential quadratic programming method is based on a Lagrangian formulation of the optimization problem. The Lagrangian formulation of the problem in Eq. (3.73) reads:

$$l = \frac{1}{2}\|\mathbf{y}\|^2 + \gamma\, g = 0 \tag{3.101}$$

where γ is the Lagrange multiplier which enforces the requirement that the point be on the limit-state surface. Hence, $\mathbf{y}$ and γ are the unknowns in Eq. (3.101). The problem is now of an unconstrained form so that a Newton scheme may be applied. The iterative Newton scheme for Eq. (3.101) can be written in the form (Luenberger 1984):

$$\begin{bmatrix} \mathbf{y}^{(m+1)} \\ \gamma^{(m+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{y}^{(m)} \\ \gamma^{(m)} \end{bmatrix} - \alpha^{(m)} \underbrace{\overbrace{\begin{bmatrix} \nabla^2_{\mathbf{y}} l^{(m)} & \nabla G^{T\,(m)} \\ \nabla G^{(m)} & 0 \end{bmatrix}^{-1}}^{\mathbf{A}^{-1}} \overbrace{\begin{bmatrix} \mathbf{y}^{(m)} \\ g^{(m)} \end{bmatrix}}^{\mathbf{b}}}_{[\,\mathbf{d} \quad \kappa\,]^T} \tag{3.102}$$

where $\nabla^2_{\mathbf{y}} l^{(m)}$ denotes the Hessian (second-derivative) matrix of the Lagrangian. In finite element analysis it is not feasible to compute the Hessian. For this reason, a sequential computation scheme is employed. An approximation for $\nabla^2_{\mathbf{y}} l$ is first assumed, usually the identity matrix, and then updated after each solution of the Newton scheme in equation (3.102). In this study the BFGS scheme (Schittkowski 1985) is used to approximate the Hessian matrix. In summary, the algorithm described below is implemented in OpenSees. In this formulation, $\bar{c}$ and $\bar{e}$ are user-selected parameters, while $\nabla^2_{\mathbf{y}} l^{(m)}$, δ, c and γ are stored as history variables at each step. Initial values are $\nabla^2_{\mathbf{y}} l^{(m)} = \mathbf{I}$, $\delta = 1$, $c = \bar{c}$ and $\gamma = 1$, where $\mathbf{I}$ is the identity matrix.

1. Compute the search direction and update history variables δ and c:

   (a) Construct the coefficient matrix $\mathbf{A}$ and vector $\mathbf{b}$ of Eq. (3.102).

   (b) Solve for the direction vector $\mathbf{d}$ and a coefficient κ by solving $\left[\begin{array}{cc} \mathbf{d} & \kappa \end{array} \right]^T = \mathbf{A}^{-1}\, \mathbf{b}$.

   (c) Update history variable δ: $\delta^{(m+1)} = \min \left\{ \delta^{(m)} \, , \, \frac{\mathbf{d}^T \left[\nabla^2_{\mathbf{y}} l \right] \mathbf{d}}{\mathbf{d}^T \mathbf{d}} \right\}$.

   (d) Compute temporary variable $e = \frac{\mathbf{d}^T \mathbf{d}}{(\kappa - \gamma)^2}$ if $\gamma \neq \kappa$, else $e = \bar{e}$.

   (e) Compute temporary parameter $i = \text{ceiling} \left(\frac{\ln(0.25\, e\, \delta\, (1 - 0.25\, \delta))}{\ln(\bar{c})} \right)$, where the ceiling function rounds the argument to the smallest integer greater than the argument.

   (f) Update history variable $c = \max \{ c, \, \bar{c}^i \}$, where $\bar{c}^i$ indicates $\bar{c}$ to the power of i.

2. Check the merit function criterion:

   (a) Compute the new value of the Lagrange multiplier (but do not store it as a history variable, yet since the old value is needed by the update scheme for the approximation of the Hessian): $\gamma^{(m+1)} = \gamma^{(m)} + \lambda \, (\kappa - \gamma^{(m)})$, where λ is the step size determined by, e.g., the Armijo rule.

   (b) Evaluate the Lagrange function in Eq. (3.101) at the old and the new trial point, yielding $l^{(m+1)} = l \left(\mathbf{y}^{(m)} + \lambda\, \mathbf{d} \right)$ and $l^{(m)} = l \left(\mathbf{y}^{(m)} \right)$, respectively. If $l^{(m+1)} - l^{(m)} \leq a \; \lambda \; \left(\nabla_{\mathbf{y}} l \right)^T \mathbf{d}$ then the step size λ is accepted. Otherwise a step size reduction as described in Section 3.5.3 is necessary. The gradient of the Lagrange function should be evaluated at the old trial point according to:$\nabla_{\mathbf{y}} l = \mathbf{y} + \gamma \; \nabla G$

3. Update the approximation of the Hessian matrix, $\mathbf{B} \approx \left[\nabla^2_{\mathbf{y}} l \right]$ if a new step is necessary:

   (a) Compute the gradient of the Lagrange function at the old trial point: $\nabla_{\mathbf{y}} l^{(m)}$.

(b) Update the Lagrange multiplier history variable: $\gamma^{(m+1)} = \gamma^{(m)} + \lambda \left(\kappa - \gamma^{(m)}\right)$.

(c) Compute the gradient of the Lagrange function at the new trial point: $\nabla_{\mathbf{y}} l^{(m+1)}$.

(d) Compute the intermediate quantity $\tilde{\mathbf{q}}$ as the difference between the gradient of the Lagrange function at the new and the old trial point: $\tilde{\mathbf{q}} = \nabla_{\mathbf{y}} l^{(m+1)} - \nabla_{\mathbf{y}} l^{(m)}$.

(e) Compute the intermediate parameter $\theta = 1.0$ if $\mathbf{d}^T \tilde{\mathbf{q}} \geq 0.2\ \lambda\ \mathbf{d}^T \left[\nabla^2_{\mathbf{y}} l\right] \mathbf{d}$ and $\theta = \frac{0.8\ \lambda\ \mathbf{d}^T \left[\nabla^2_{\mathbf{y}} l\right] \mathbf{d}}{\lambda\ \mathbf{d}^T \left[\nabla^2_{\mathbf{y}} l\right] \mathbf{d} - \mathbf{d}^T \tilde{\mathbf{q}}}$ otherwise.

(f) Compute the intermediate quantity $\mathbf{q} = \theta\ \tilde{\mathbf{q}} + (1 - \theta)\ \lambda\ (\mathbf{B}\ \mathbf{d})$.

(g) Update the Hessian approximation: $\mathbf{B}^{(m+1)} = \mathbf{B}^{(m)} + \frac{\mathbf{q}^T \mathbf{q}}{\lambda\ \mathbf{q}^T \mathbf{d}} - \frac{1}{\mathbf{d}^T \mathbf{B}^{(m)} \mathbf{d}} \left(\mathbf{B}^{(m)} \mathbf{d} \mathbf{d}^T \mathbf{B}^{(m)}\right)$, where $\mathbf{B}^{(m)}$ denotes the approximation of the Hessian at the previous point.

Comparison of the stability and efficiency of the algorithms discussed above are presented in Chapter 4.6.

3.6 PARAMETER IMPORTANCE MEASURES

Parameter importance measures represent a valuable by-product of finite element reliability analyses. Such measures allow the user to rank model parameters according to the relative significance of their uncertainty for specific performance functions. This information is useful for several purposes. It may be used to gain physical insight into a problem or to reduce the problem size by neglecting uncertainty in unimportant random variables. It may also be used as indicators to allocate resources in the design phase.

In this study, importance measures are obtained from FOSM response statistics analysis and FORM reliability analysis. An importance measure from FOSM analysis is obtained by examining the contributions to the variance of the performance function. Expanding Eq. (3.42), the variance is written as

$$\mathrm{Var}[g] = (\nabla g_1\, \sigma_1)^2 + (\nabla g_2\, \sigma_2)^2 + \cdots + (\nabla g_n\, \sigma_n)^2 + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \nabla g_i \nabla g_j\, \sigma_i \sigma_j \rho_{ij} \tag{3.103}$$

where $\nabla g_i = \frac{\partial g}{\partial x_i}$ and σ_i is the standard deviation of random variable x_i. It is observed that the product $(\nabla g_i\, \sigma_i)^2$ indicates the direct contribution of random variable x_i to the total variance of the performance function. Thus, the products $\nabla g_i\, \sigma_i$ represent inexpensive, yet useful, importance measures from a single finite element analysis. Care must be exercised, however, in circumstances

when the structural behavior at the mean point is principally different from the behavior at the design point for the performance function in question. In such cases, importance measures from FORM analysis must be considered.

In this study, four importance measures from FORM analysis are considered. First, consider the performance function in the standard normal space linearized around the design point:

$$G \approx \bar{G} = \nabla G \left(\mathbf{y} - \mathbf{y}^*\right) = \|\nabla G\| \left(\beta - \boldsymbol{\alpha}\mathbf{y}\right) \tag{3.104}$$

where use has been made of $\boldsymbol{\alpha} = -\frac{\nabla G}{\|\nabla G\|}$ and $\beta = \boldsymbol{\alpha}\mathbf{y}^*$ in the last equality. Similar to the approach employed above, we investigate the variance of $\bar{G}$:

$$\mathrm{Var}[\bar{G}] = \|\nabla G\|^2 \left(\alpha_1^2 + \alpha_2^2 + \cdots + \alpha_n^2\right) = \|\nabla G\|^2 \tag{3.105}$$

The result confirms the well-known regard of the absolute values of the elements of $\boldsymbol{\alpha}$ as indicative of the relative importance of the corresponding random variables in the standard normal space. Furthermore, a positive α-value indicates a load variable and a negative α-value indicates a resistance variable. Care must be exercised when working with random variables with negative mean values. The importance indicator $(\alpha_i \,\mathrm{sign}(\mu_i))$ where μ_i is the mean of random variable x_i is then more intuitive to judge the nature (load or resistance) of the random variable.

$\boldsymbol{\alpha}$ is a valid importance measure in the dimensionless standard normal random space. However, when the random variables are correlated then there is no one-to-one mapping between $\mathbf{y}$ and the original random variables $\mathbf{x}$. In that case, an importance ordering of the elements of $\mathbf{y}$ does not imply the same importance ordering of the elements of $\mathbf{x}$. To this end, following Der Kiureghian (2003), consider the linearized probability transformation $\mathbf{y} = \mathbf{T}(\mathbf{x})$ at the design point:

$$\mathbf{y} \approx \mathbf{y}^* + J_{\mathbf{y}^*,\mathbf{x}^*} \left(\mathbf{x} - \mathbf{x}^*\right) \tag{3.106}$$

Replacing the approximation by an equality, we can write:

$$\mathbf{y} = \mathbf{y}^* + J_{\mathbf{y}^*,\mathbf{x}^*} \left(\hat{\mathbf{x}} - \mathbf{x}^*\right) \tag{3.107}$$

where $\hat{\mathbf{x}}$ is slightly different from $\mathbf{x}$. Since $\hat{\mathbf{x}}$ is a linear function of $\mathbf{y}$, it must have the joint normal distribution. Its covariance matrix is:

$$\hat{\boldsymbol{\Sigma}} = J^{-1}_{\mathbf{y}^*,\mathbf{x}^*} \; J^{-T}_{\mathbf{y}^*,\mathbf{x}^*} \tag{3.108}$$

The random variables $\hat{\mathbf{x}}$ are considered as "equivalent normals" of $\mathbf{x}$ at the design point. The covariance matrix $\hat{\boldsymbol{\Sigma}}$ in general depends on the design point and is slightly different from the covariance

matrix $\mathbf{\Sigma}$ of $\mathbf{x}$. The magnitude of the difference depends on the degree of non-normality of $\mathbf{x}$. Substitution of the linearized transformation in Eq. (3.107) into Eq. (3.104) leads to the following expression for the linearized performance function:

$$\bar{G} = -\|\nabla G\| \boldsymbol{\alpha} J_{\mathbf{y}^*, \mathbf{x}^*} \left(\hat{\mathbf{x}} - \mathbf{x}^*\right) \tag{3.109}$$

The corresponding variance is:

$$\mathrm{Var}[\bar{G}] = \|\nabla G\|^2 \left(\boldsymbol{\alpha} J_{\mathbf{y}^*, \mathbf{x}^*} \hat{\mathbf{\Sigma}} J^T_{\mathbf{y}^*, \mathbf{x}^*} \boldsymbol{\alpha}^T\right) \tag{3.110}$$

The contribution from the individual variances of the basic random variables can be separated from the contribution from the covariances. The former is the importance measure of interest and reads $\|\nabla G\|^2 \left(\|\boldsymbol{\alpha} J_{\mathbf{y}, \mathbf{x}} \hat{\mathbf{D}}\|^2\right)$ where $\hat{\mathbf{D}}$ is the diagonal matrix of standard deviations σ_i. A normalized importance vector for the vector of original random variables is thus defined as:

$$\boldsymbol{\gamma} = \frac{\boldsymbol{\alpha} J_{\mathbf{y}, \mathbf{x}} \hat{\mathbf{D}}}{\|\boldsymbol{\alpha} J_{\mathbf{y}, \mathbf{x}} \hat{\mathbf{D}}\|} \tag{3.111}$$

It is noted that for statistically independent random variables $\boldsymbol{\gamma} = \boldsymbol{\alpha}$.

Next, it is of interest to obtain importance rankings of the means and the standard deviations of the random variables. For this purpose we make use of the reliability sensitivity measures discussed by Hohenbichler and Rackwitz (1986) and Bjerager and Krenk (1989):

$$\frac{\partial \beta}{\partial \mu_i} = \boldsymbol{\alpha}^T \frac{\partial \mathbf{y}^*}{\partial \mu_i} \tag{3.112}$$

$$\frac{\partial \beta}{\partial \sigma_i} = \boldsymbol{\alpha}^T \frac{\partial \mathbf{y}^*}{\partial \sigma_i} \tag{3.113}$$

where $\frac{\partial \mathbf{y}^*}{\partial \theta_f}$ is obtained by differentiation of the probability transformation $\mathbf{y} = \mathbf{T}(\mathbf{x})$ at the design point. The elements of the vectors $\frac{\partial \beta}{\partial \boldsymbol{\mu}}$ and $\frac{\partial \beta}{\partial \boldsymbol{\sigma}}$ are not immediately comparable. This is due to the possible different units of the random variables. Importance measures are obtained from Eqs. (3.112) and (3.113) by scaling each element by the corresponding standard deviation (Liu *et al.* 1989):

$$\boldsymbol{\delta} = \nabla_{\boldsymbol{\mu}} \beta \, \hat{\mathbf{D}} \tag{3.114}$$

$$\boldsymbol{\eta} = \nabla_{\boldsymbol{\sigma}} \beta \, \hat{\mathbf{D}} \tag{3.115}$$

where $\nabla_{\boldsymbol{\mu}} \beta$ and $\nabla_{\boldsymbol{\sigma}} \beta$ are row vectors.

The four importance measures $\boldsymbol{\alpha}$, $\boldsymbol{\gamma}$, $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$ are defined and implemented in OpenSees. They are provided as part of the standard output from FORM analysis.

3.7 IMPLEMENTATIONS IN OPENSEES

Following recommended practice (Gamma *et al.* 1995) and as in the finite element analysis module, the reliability module is divided into Domain and Analysis parts (Gamma *et al.* 1995). The Analysis part contains a framework of analysis components, each component designed to solve a particular task. Specific implementations (sub-classes) of these framework components are implemented to actually solve the task. Several such implementations may be available for the same task. For instance, there are several ways to select the search direction vector for finding the design point. Hence, several subclasses are available to deliver the task promised by the general search direction class. The Domain contains objects such as random variables, correlation coefficients and performance functions.

An overview of the framework of analysis components is provided in Table 3.2. It is seen that some of the base classes, such as the one representing the search direction computation tool, have several sub-classes implemented already. Other framework components, such as the probability transformation, have only one option available at this time. Table 3.3 provides an overview of the available implementations in the Domain part. The random variable base class has a number of sub-classes corresponding to the distributions listed in Section 3.2.1. The Domain classes representing correlation coefficients, performance functions, random variable positioners and parameter positioners are self-contained and do not need sub-classes. Table 3.4 provides an overview of the available analysis types. Each of these analysis classes make use of aggregations of analysis components. The user is prompted if adequate analysis tools are not provided. For further details, the implementations can be browsed at `http://opensees.berkeley.edu`.

The interface between the reliability algorithm and the finite element code is an essential issue in finite element reliability analysis. It involves mapping random variables onto the finite element domain. Furthermore, uncertain parameters must be updated for each new trial point. Each time the performance function is to be evaluated, the finite element model needs to receive the updated realizations of the basic random variables. Upon completion of the finite element analysis, response quantities entering the performance function must be returned to the reliability module. This is also the case for DDM response sensitivities. This section describes how this interaction is handled in the present version of the code.

Three classes of the framework are involved in this communication, namely the `RandomVariablePositioner`, the `OpenSeesGFunEvaluator` and the `OpenSeesSensitivityEvaluator`. The last comes into play only when the gradient of response quantities is to be computed by the DDM.

We first consider the issue of updating parameters in the finite element model. This is handled by two features: (1) Each random variable in the finite element model has one or more `RandomVariablePositioner` associated with it. (2) Two member functions called `setParameter` and `updateParameter` are available in every object of the finite element domain. Recall that a key issue in object-oriented programming is data encapsulation. The scheme presented here does not break this principle, while providing the needed flexibility to alter data member values. Figure 3.7 shows the interaction that allows updating of parameters in the finite element code. Below is a step-by-step description of the components in Figure 3.7 and their actions in the interface:

1. The analyst creates nodes, elements, materials, etc., of the finite element domain. Each of these objects is referred to as "An object in the FE domain" as in Figure 3.7.

2. Using commands outlined in Chapter 3.7, the analyst then creates `RandomVariable` and `RandomVariablePositioner` objects. An instance of the latter is shown in Figure 3.7.

3. When a `RandomVariablePositioner` is instantiated it obtains a pointer to the object in the finite element model which contains the uncertain parameter. This is shown in Figure 3.7 where the `RandomVariablePositioner` contains an object in the finite element domain as one of its data members.

4. When the link in the previous item is created, the constructor (the member function which is automatically called when an object is instantiated) of the `RandomVariablePositioner` object calls the `setParameter` member function of the structural object (see Figure 3.7). It tries to find the parameter that the user has specified and, if successful, a `parameterID` value is stored for later use (see below). If not, an error message is given.

5. During the analysis phase, there is need to update the model parameters. This task is orchestrated from within the `OpenSeesGFunEvaluator` object. It loops over all `RandomVariablePositioner` objects and calls the `update` function shown in Figure 3.7 with the current realization of the basic random variables. In turn, this member function calls the `updateParameter` member function of its structural object (see Figure 3.7). A number to identify the parameter (the `parameterID` data member) and its new value are passed during the call.

6. The command `analyze` is executed from within the `OpenSeesGFunEvaluator` object. After convergence of the finite element calculations, the response quantities entering the performance

function are obtained from the finite element domain, e.g., by commands such as `nodeDisp` of the Tcl interface of OpenSees. That is, the `OpenSeesGFunEvaluator` has a pointer to the interpreter and can call ordinary Tcl commands from inside the C++ code. Values are then assigned to the respective variables in the performance function and the expression is evaluated, using Tcl also for this purpose.

7. If gradients are computed by DDM, the `OpenSeesSensitivityEvaluator` is used. The `getGrad_g` member function of this class uses similar concepts as described above. The member function `activateParameter` shown in Figure 3.7 is used to determine the contribution from each object in the finite element model to the right-hand side of the sensitivity equation. Commands such as `sensNodeDisp` are used to obtain desired gradients. Alternatively, response recorder objects can be used to store such results to a file. It is emphasized that the `gFunEvaluator` is the only component in the reliability framework that executes the finite element analysis. Once this is done, all needed information is available from the converged finite element domain or from the recorder files.

Detailed information about the class interfaces for the reliability implementations in OpenSees is provided in Appendix 6.2.

Table 3.1: Permissible response quantities in the performance function, depending on the type of finite element analysis.

Response quantity	Static load-control	Static displacement-control	Dynamics
Deformation-type	Yes	No	Yes
Force-type	No	Yes	Yes
Max. deformation	Yes	No	No
Max. force	No	Yes	No
Accumulated resp.	Yes	Yes	Yes

Table 3.2: Framework of analysis components and currently available specific implementations.

Base class	**Currently available sub-classes**
GFunEvaluator	BasicGFunEvaluator OpenSeesGFunEvaluator TclGFunEvaluator MatlabGFunEvaluator
GradGEvaluator	OpenSeesGradGEvaluator FiniteDifferenceGradGEvaluator
ProbabilityTransformation	NatafProbabilityTransformation
FindDesignPointAlgorithm	SearchWithStepSizeAndStepDirection
SearchDirection	GradientProjectionSearchDirection HLRFSearchDirection PolakHeSearchDirectionAndMeritFunction SQPSearchDirectionAndMeritFunction
StepSizeRule	FixedStepSizeRule ArmijoStepSizeRule
MeritFunctionCheck	AdkZhangMeritFunctionCheck CriteriaReductionMeritFunctionCheck PolakHeSearchDirectionAndMeritFunction SQPSearchDirectionAndMeritFunction
RandomNumberGenerator	CStdLibRandGenerator
ReliabilityConvergenceCheck	StandardReliabilityConvergenceCheck OptimalityConditionReliabilityConvergenceCheck
RootFindingAlgorithm	ModifiedNewtonRootFindingAlgorithm SecantRootFindingAlgorithm
FindCurvaturesAlgorithm	FirstPrincipalCurvature

Table 3.3: Domain components for reliability analysis in OpenSees.

Base class	**Currently available sub-classes**
RandomVariable	Normal Lognormal etc. (see overview in Section 3.2.1)
CorrelationCoefficient	N/A
PerformanceFunction	N/A
RandomVariablePositioner	N/A
ParameterPositioner	N/A
Filter	StandardOscillatorFilter
ModulatingFunction	ConstantModulatingFunction GammaModulatingFunction TrapezoidalModulatingFunction
Spectrum	NarrowBandSpectrum PointsSpectrum JonswapSpectrum
TimeSeries	DiscretizedRandomProcessSeries SimulatedRandomProcessSeries

Table 3.4: Analysis types related to reliability analysis available in OpenSees.

FORMAnalysis
SamplingAnalysis
FOSMAnalysis
SORMAnalysis
SystemAnalysis
OutCrossingAnalysis
FragilityAnalysis
GFunVisualizationAnalysis

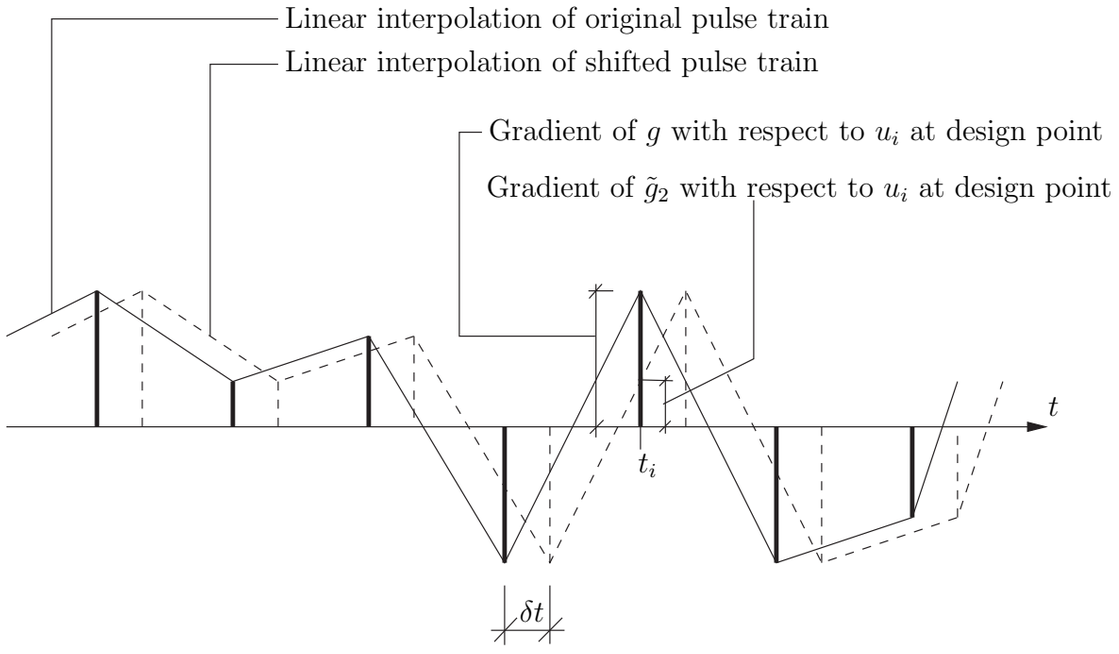

Figure 3.1: Determination of design point values of random variables for shifted time series.

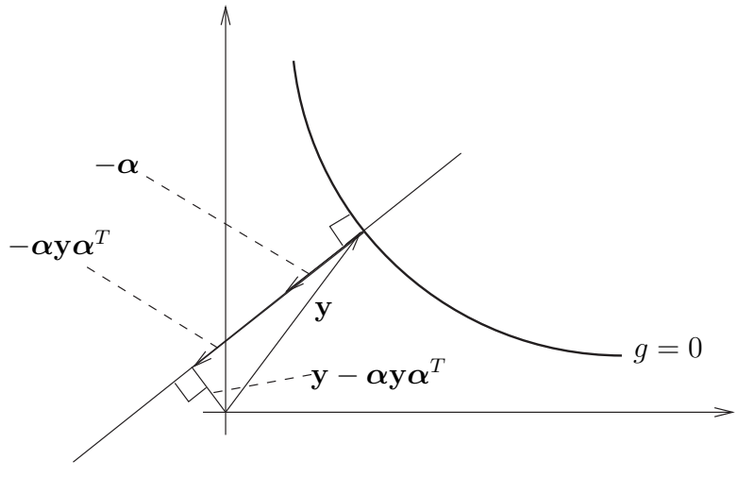

Figure 3.2: Illustration of design point convergence criterion.

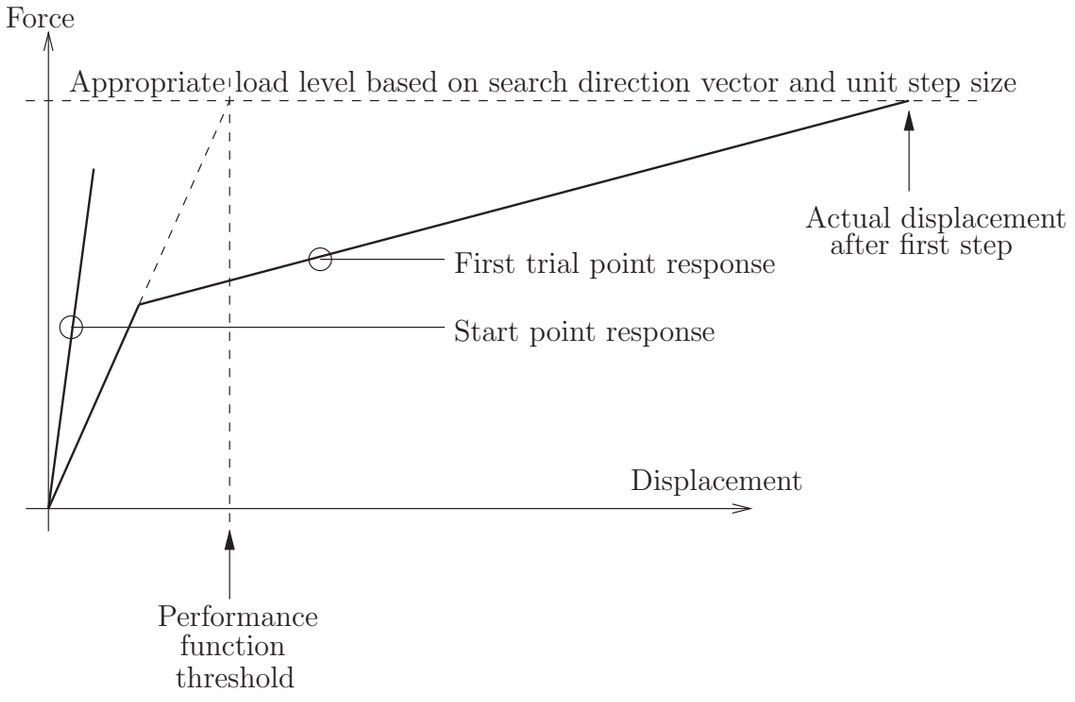

Figure 3.3: Example of a "too large" step size in nonlinear finite element reliability analysis.

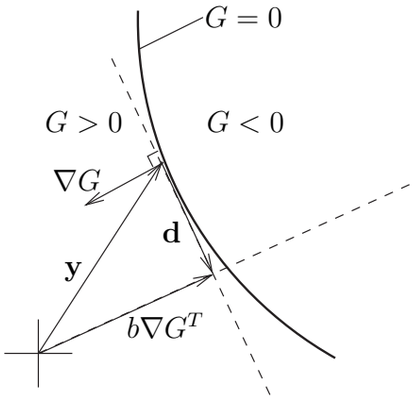

Figure 3.4: Initial search direction for the gradient projection algorithm.

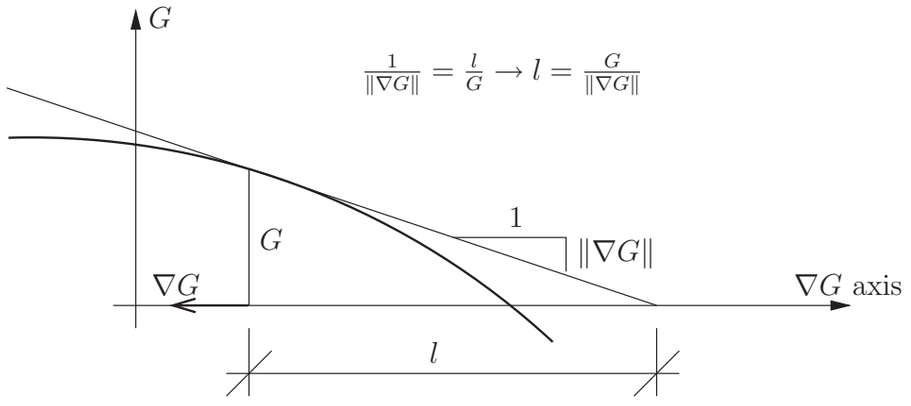

Figure 3.5: Step length of a directional Newton scheme in multi dimensions.

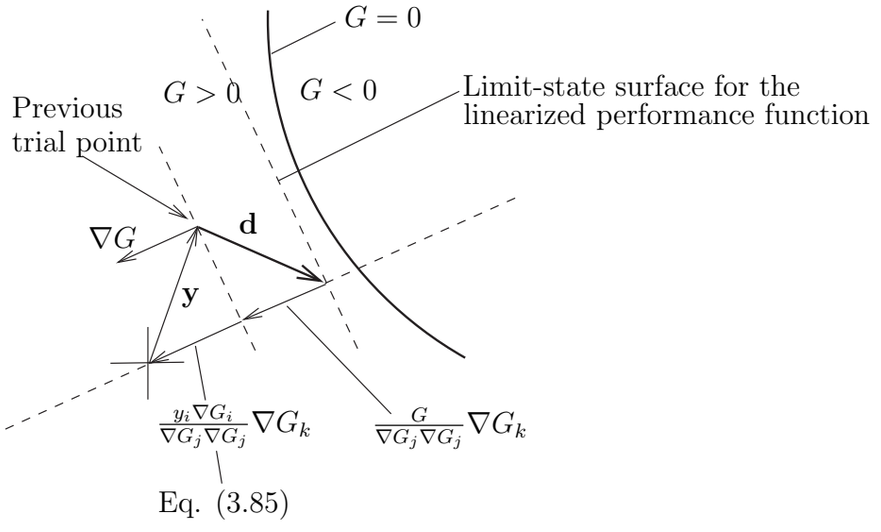

Figure 3.6: Search direction for the HLRF algorithm as a sum of vectors.

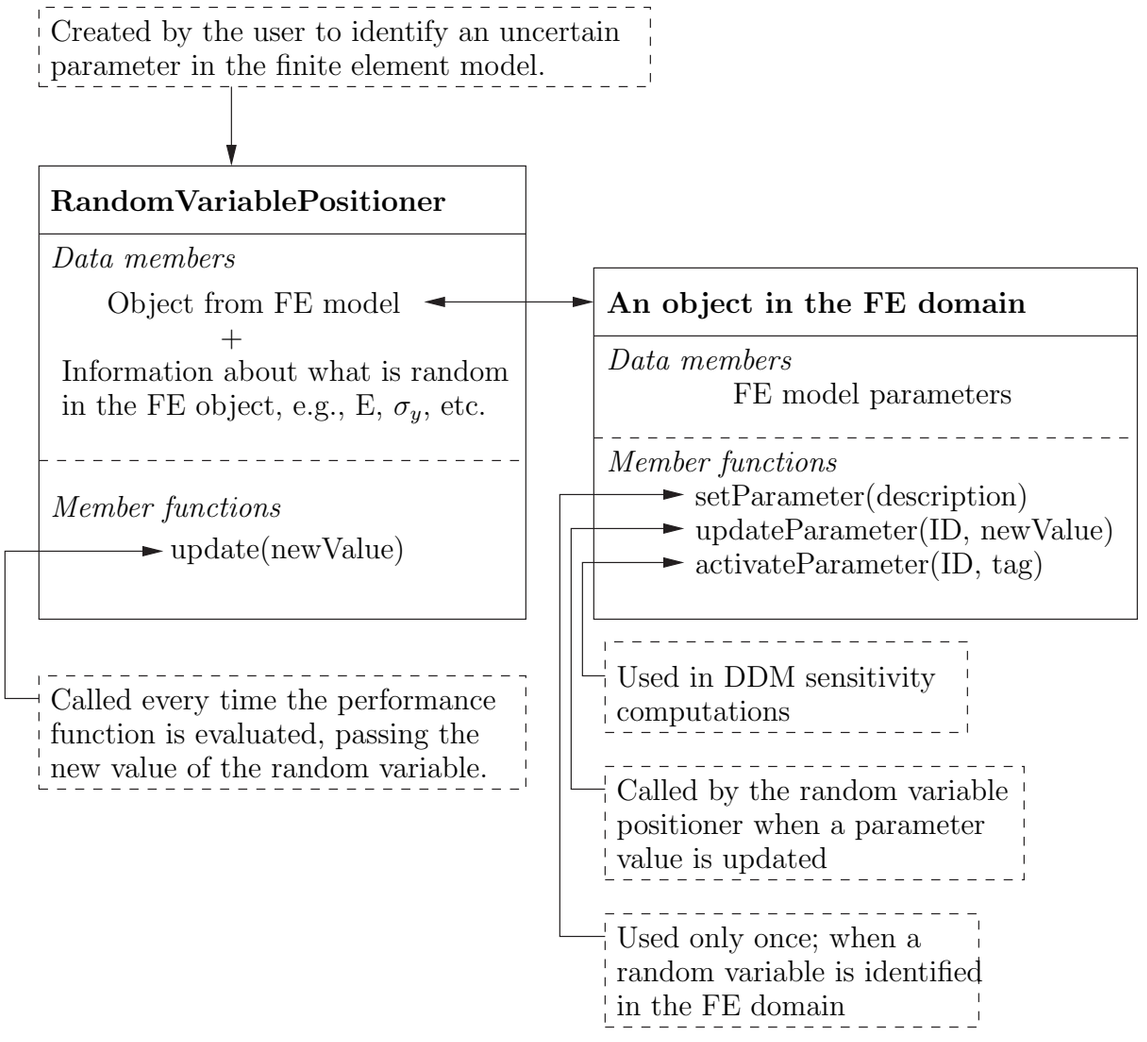

Figure 3.7: Scheme for updating parameters in the finite element model with new realizations of random variables.

4 User's Guide to Reliability and Sensitivity Analysis in OpenSees

In essence, OpenSees is a software framework for developing computer applications. It should not be regarded as a packaged "code" but as a collection of software components. The object-oriented programming approach facilitates this development strategy. The reliability and sensitivity implementations of this work conform to this development strategy. Thus, maintainability and extensibility of the software are central issues.

The scripting language Tcl (Welch 2000) is employed as an interface to make use of the OpenSees software framework. Commands are added to Tcl to create model components and to aggregate analysis tools. This chapter provides a detailed overview of the commands added to Tcl to make use of the reliability and sensitivity implementations presented in the previous chapters. Together with the finite element analysis module, OpenSees is now capable of conducting comprehensive finite element reliability and sensitivity analyses.

It is assumed that the user is familiar with executing an ordinary OpenSees analysis by creating and running Tcl scripts. The inexperienced user is referred to the documentation provided by McKenna and Fenves (2002) for further details. The reliability and sensitivity commands are presented below in the same format as in other OpenSees documentations. Namely, a command is given in the form

```
commmandName arg1? arg2 arg3? <arg4? ...>
```

A question mark after an argument indicates that an integer or a floating point number should be provided; otherwise, a character string should be given. Optional arguments are enclosed in angular brackets. Figure 4.1 provides an overview of the commands that are presented in the subsequent sections.

4.1 ELEMENTARY REQUIREMENTS

None of the commands presented in this chapter will work before the command

`reliability`

is issued. This command "activates" the reliability and sensitivity commands by loading them into the Tcl script interpreter.

A typical reliability analysis with OpenSees involves the following tasks:

1. Create an input file. This can be done in any text editor provided it saves the file in an `ascii` format (pure text). The alternative is to give the commands one-by-one at the command prompt of the interpreter. The advantage of working with an input file is that the user may experiment and edit parameters without having to re-type all input parameters and commands for each re-run.

2. In reliability analysis, the reliability analysis object orchestrates the computations. For this reason the user should not issue the "`analyze`" command that is issued in other OpenSees analyses. The finite element analysis is executed from inside the reliability algorithm. However, the finite element model and analysis setup must be specified as for any OpenSees finite element analysis.

3. Execute the analysis by loading the input file. This is done either from the OpenSees command prompt by issuing the command "source filename," or in a command window of the operating system by issuing the command "opensees inputfilename."

4. Inspect the results in the specified output file. An error message is generated by the program if the analysis fails.

4.2 UNCERTAINTY MODELING

An essential ingredient in finite element reliability analysis is to characterize parameters in the finite element model as uncertain. This is done by mapping random variables into the finite element domain. This section describes how this is done and which components need to be created.

A random variable object can be created in several ways. The simplest command for this purpose is:

`randomVariable tag? type mean? stdv? <startPt?>`

The `tag` argument indicates the identification number of the random variable. These objects must be ordered in a consecutive and uninterrupted manner. `mean` is the mean and `stdv` is the standard deviation of the random variable. `type` identifies the distribution type. The following library of types is available: `normal`, `lognormal`, `uniform`, `chiSquare`, `exponential`, `gamma`, `gumbel`, `laplace`, `pareto`, `rayleigh`, `shiftedExponential`, `shiftedRayleigh`, `type1LargestValue`, `type1SmallestValue`, `type2LargestValue`, `type3SmallestValue`, `weibull` and `beta`. It is noted that `type1LargestValue` and `gumbel` are identical. Some of the distributions cannot be completely defined by the mean and the standard deviation and must be specified by using the command described next. The `startPt` argument allows the user to specify a value for the random variable to be used as the start point in the search for the design point or as the center of the sampling distribution in an importance sampling analysis. By default the `startPt` is equal to the mean. See also the `startPoint` object in Section 4.4.

Alternatively, a random variable can be created by the command:

`randomVariable tag? type par1? par2? par3? par4? <startPt?>`

Here the parameters of the probability distribution are given instead of the mean and standard deviation. See Appendix B.9 for a detailed list of the available probability distributions and their parameters. This command option is needed for distributions with more than 2 parameters. In such cases the mean and standard deviation do not uniquely define the distribution. An example is the beta distribution type, which requires four parameters. For 2 or 3 parameter distributions `par3` and/or `par4` are only dummy input but still must be provided.

Refer to the `randomVariablePositioner` command later in this section for yet another alternative for creating random variable objects in OpenSees.

A user-defined random variable type is also available. See Section 3.2.2. Two alternatives are available to specify its probability distribution. Either it can be specified on the command line (all on one line):

```
randomVariable tag? userdefined -list x1? pdf1? x2? pdf2?
          < x3? pdf3? x4? pdf4? x5? pdf5? ...> <startPt?>
```

or in a separate file:

`randomVariable tag? userdefined -file filename.txt`

The file format is two columns. The first column consecutively defines the points x_i while the second column specifies the corresponding values $f(x_i)$ of the PDF. In both commands, the PDF is considered to be a piecewise linear function between the specified points. Any number of points can be specified.

A random variable reduction command is available. It may be used to reduce the number of random variables in the model based on knowledge of the relative importance of the random variables. This command provides a means of retaining consecutive random variable numbering without rearranging the uncertainty model. For instance, importance vectors obtained by running a computationally inexpensive second-moment analysis prior to a more costly FORM analysis may be used to decide to disregard the uncertainty in some of the random variables. Two command syntaxes are available:

```
rvReduction arg1? arg2? ...
rvReduction -file filename arg?
```

The arguments `arg1? arg2? ...` represent the tag numbers of the random variables, which should be kept for subsequent analysis. In the latter command option these numbers are provided in a file. The file contains one column, listing the importance ordering of the random variables by their tag numbers. The argument `arg` states the number of random variables, which should be kept for subsequent analysis. For instance, if `arg` is set to 30 then only the 30 most important random variables will be kept. After execution of either of these commands the tags of the remaining random variables are altered to correspond to their importance ordering. This is done to ensure that the random variable tags are consecutive and uninterrupted numbers.

Correlation between random variables may be specified in three different ways:

```
correlate rv1? rv2? correlation?
correlateGroup firstRV? lastRV? correlation?
correlationStructure type firstRV? lastRV? theta?
```

The first version specifies a correlation coefficient equal to `correlation` between random variable number `rv1` and random variable number `rv2`. The second command specifies that all pairs of random variables between random variable number `firstRV` and random variable number `lastRV` are equi-correlated with a correlation coefficient equal to `correlation`. The last command specifies a correlation structure for a group of random variables. The following correlation structures are available. See Section 3.2.3 for more details:

$$\texttt{homogeneous1}: \quad \rho_{ij} = \exp\left(-\frac{|i-j|}{\theta}\right)$$

$$\texttt{homogeneous2}: \quad \rho_{ij} = \exp\left(-\frac{(i-j)^2}{\theta^2}\right)$$

$$\texttt{homogeneous3}: \quad \rho_{ij} = \frac{1}{1+\theta\,(i-j)^2}$$

$$\texttt{homogeneous4}: \quad \rho_{ij} = \begin{cases} 1-\frac{|i-j|}{\theta} & \text{for} \quad |i-j| \le \theta \\ 0 & \text{for} \quad |i-j| > \theta \end{cases}$$

$$\texttt{vectorProduct}: \quad \rho_{ij} \;=\; r\Big(i - \text{rvFirst} + 1\Big)\, r\Big(j - \text{rvFirst} + 1\Big)$$

where ρ_{ij} represents the correlation coefficient between random variable number i and random variable number j. The positive-valued parameter θ is specified by the `theta` input of the command, while the vector $\mathbf{r}$ must be provided in a file named *correlationVector.txt* if the `vecorProduct` type is used. As discussed in Section 3.2.3, the values of the elements of $\mathbf{r}$ must be between -1 and 1.

To position/map the random variables into the finite element model the following command structure is available (all on one line):

```
randomVariablePositioner tag? (...random variable identification...)
                              (...parameter identification...)
```

The `tag` argument indicates the identification number of the random variable positioner. If the random variable in question has already been created by the `randomVariable` command, then the random variable identification simply reads: `-rvNum rvNum?`. However, the random variable identification can also be given as a command to create a new random variable. In this case the random variable identification can have any of the following alternative forms (the random variable is automatically assigned the same tag as the random variable positioner tag):

```
-createRV3 type mean? stdv?
-createRV4 type mean? stdv? startPt?
-createRV5 type par1? par2? par3? par4?
-createRV6 type par1? par2? par3? par4? startPt?
```

The same distribution types as listed above for the `randomVariable` command are available. Note how the first argument informs the interpreter about the number of arguments that follows to characterize the random variable.

The available parameter identification alternatives in the positioner command depend on which parameters in the finite element code have been enabled for uncertainty characterization. New parameters are frequently made available on request from users. The current list of available parameters is given below. The parameters are classified as either (1) belonging to material, section or element objects, (2) being a nodal coordinate, (3) being a nodal load, or (4) representing a parameter of a time series object. The first category of parameters is identified in the following manner:

```
-element arg1? <-section arg2?> <-material arg3?> type
```

Two examples are:

```
-element 4 E
-element 5 -section 6 -material 7 sigmaY
```

The first example identifies the Young's modulus of element number 4. This is a typical positioner command for a linear element, where the Young's modulus is a parameter of the element object itself. The second example identifies the yield strength of material 7 of section 6 of element 5. Table 4.1 contains an overview of the element, section and material parameters that have been made available for uncertainty characterization.

A nodal coordinate is identified as follows:

```
-node nodenumber?  -coord direction?
```

A nodal mass is identified as follows:

```
-node nodenumber?  -mass direction?
```

A nodal load is identified as follows:

```
-loadPattern patternTag? -loadAtNode nodeNumber? dof?
```

A random variable of the discretized random process object described in Section 3.2.5 is identified as follows:

```
-loadPattern patternTag? -randomProcessDiscretizer kickInTime?
```

The `kickInTime` argument indicates the point along the time axis, where the random impulse occurs.

Positioning of quantities other than random variables may be done with the command:

```
parameterPositioner tag? (...parameter identification...)
```

This command is especially useful in conjunction with parametric reliability analysis, see Section 4.5. By employing this command and including quantities such as `par_1` and `par_2` in the performance function (see Section 4.3) "fragility curves" for various demand parameters in the finite element model may be obtained by automatic execution of a sequence of reliability analyses. The parameter identification syntax is identical to the `randomVariablePositioner` command described above.

Discretized random processes can be used as a time series object. This object may be used to create a stochastic ground motion input in OpenSees. In the core OpenSees framework, there exists no stand-alone command to create time series objects McKenna and Fenves (2002). However, time series may be used as arguments for a number of commands. Two new time series objects have been added, corresponding to the following commands:

```
DiscretizedRandomProcess mean? maxStdv? modFuncTag1? <...>
SimulatedRandomProcess spectrumTag? mean? numFreqIntervals?
```

The `mean` argument is used to specify a constant mean which is added to the generated process. The `maxStdv` argument is used to specify the target maximum standard deviation over all time points with random pulses. The `modFuncTag1?` argument denotes the tag number of a modulating function. The angular brackets emphasize that several modulating functions can be specified. Each modulating function has a filter associated with it. A modulating function is created by the following command:

```
modulatingFunction tag? type filterTag? <arg1? arg2? ...>
```

The `type` argument can be either `gamma` (in which case `arg1` represents a, `arg2` represents b and `arg3` represents c in the equation $a\ t^b\ e^{-c\,t}$); `constant` (in which case the amplitude is provided by `arg1`) and `trapezoidal` (in which case `arg1`, `arg2`, `arg3` and `arg4` represent the four time instants t_1, t_2, t_3 and t_4 defining the trapezoidal and `arg5` denotes the constant amplitude between points t_2 and t_3).

Filters to specify the frequency content of a discretized random process are created by the command:

```
filter tag? type <arg1? arg2? ...>
```

Currently only one type of filter is available, namely `standard`, which denotes the impulse-response function for a standard linear oscillator.

Power spectral density (PSD) functions to be used in conjunction with the `SimulatedRandomProcess` time series are created by the following command:

```
spectrum tag? type <arg1? arg2? ...>
```

The realization of the time series is based on the following equation: $x(t) = \mu + \sum_{i=1}^{N} \sqrt{2\ S(\omega_i)\ \Delta\omega}\quad A_i\ \cos\left(\omega_i t + \theta_i\right)$, where $S(\omega)$ is the PSD amplitude at the frequency ω_i and θ_i and A_i are random generated numbers having uniform and normal distributions, respectively. Available spectrum types are: (1) `bandedWN`, in which case `arg1`, `arg2` and `arg3` denotes ω_{min}, ω_{max} and PSD amplitude, respectively, of a banded white noise spectrum; (2) `jonswap`, in which case `arg1`, `arg2`, `arg3`, `arg4` and `arg5` denotes ω_{min}, ω_{max}, α, ω_p and γ in the equation

$$S(\omega) = \gamma^{exp\left(-\frac{\omega-\omega_p}{2\,\sigma^2\,\omega_p^2}\right)}\ \alpha\,\omega^{-0.5}\ exp\left(-\frac{5}{4}\left(\frac{\omega}{\omega_p}\right)^{-4}\right)$$

where $\sigma = 0.07$ for $\omega \leq \omega_p$ and $\sigma = 0.09$ for $\omega > \omega_p$; and (3) `points`, in which case `arg1`, `arg2`, `arg3`, `arg4`, etc., represent ω_1, PSD_1, ω_2, PSD_2, etc., of the arbitrary number of PSD coordinates along the frequency axis.

4.3 PERFORMANCE FUNCTIONS

In structural reliability analysis performance functions are the means by which the "failure" and "safe" states are distinguished. In OpenSees the command `performanceFunction` is available for defining the performance function. The general syntax of the command is:

```
performanceFunction tag? "expression"
```

The `expression` must to be enclosed by double-quotes, as shown, and can be any analytical expression that can be evaluated by the Tcl interpreter (Welch 2000). Various quantities may be used in the expression, including random variables, response quantities from an OpenSees finite element analysis, quantities stored to file, and parameters defined in the Tcl interpreter. The syntax shown below should be used. Note that the curly braces are mandatory and that the recorders are created automatically for quantities starting with the phrase `rec` (the syntax corresponds to the syntax used to create ordinary recorders in OpenSees)

- `{x_5}`: random variable number 5.
- `{u_5_2}`: displacement of node 5 along dof number 2.
- `{ud_5_2}`: velocity of node 5 along dof number 2.
- `{rec_node_disp_5_2}`: recorded displacement of node 5 along dof number 2.
- `{rec_node_vel_5_2}`: recorded velocity of node 5 along dof number 2.
- `{rec_node_accel_5_2}`: recorded acceleration of node 5 along dof number 2.
- `{rec_element_5_globalForce_2}`: recorded global element force in direction 2 of element number 5.
- `{rec_element_5_localForce_2}`: recorded local element force in direction 2 of element number 5.
- `{rec_element_5_section_2_force_1}`: recorded section force component 1 (for instance axial force) of section 2 in element 5.
- `{rec_element_5_section_2_deformation_1}`: recorded section deformation component 1 (for instance axial deformation) of section 2 in element 5.

- `{rec_element_5_section_2_stiffness_1}`: recorded section stiffness component 1 of section 2 in element 5.
- `{rec_element_5_section_2_fiber_2_36_6_54_stress}`: stress of fiber at location $y = 2.36$, $z = 6.54$ in section 2 of element number 5.
- `{rec_element_5_section_2_fiber_2_36_6_54_strain}`: strain of fiber at location $y = 2.36$, $z = 6.54$ in section 2 of element number 5.
- `{par_5}`:parameter number 5 (e.g., in parametric reliability analysis).
- `{file_filename_5_2}`: quantity in row 5 and column 2 of a file. Tcl does not allow dots in parameter names. Hence, it is not possible to include dots in the file name.
- `{variableName}`: parameter previously defined by a command `set variableName value?` in the Tcl interpreter session.

All finite element response quantities are sampled at the final converged state of the analysis. An example of a performance function for response statistics analysis of a single response quantity is:

```
performanceFunction 1 "{u_7_1}"
```

An alternative example involving more quantities and mathematical operations is:

```
performanceFunction 1 "{par_1}-2*({u_7_1}+{file_myFile_4_1})*log({x_4})"
```

A performance function involving the base shear force of a building may, for instance, be defined in terms of the sum of global element forces.

4.4 ANALYSIS TOOLS

An important part of the development of the reliability analysis framework has been to modularize the analysis algorithms. This is done in a fashion similar to the analysis component framework of the core finite element implementations. A high level of maintainability and extensibility of the software is obtained with this approach.

Before a reliability analysis is executed the user must create an aggregation of necessary analysis components or "tools." Which analysis components are needed depends on the analysis type, as described in Section 4.5. Below is a review of the available tools. The order in which the tools are

provided are of importance, since some tools make use of others. The user will be notified by an error message if such dependencies are violated.

A probability transformation object is needed for transformation of the random variables between the original and standard normal spaces. This analysis tool must also provide the Jacobian matrix of the transformation. Currently, the Nataf model (Liu and Der Kiureghian 1986) is the available implementation of this component in OpenSees. The corresponding command reads:

```
probabilityTransformation Nataf <-print flag?>
```

The optional print flag is `0` by default but can be set to `1` to print realizations of the random variables to screen. For instance, information about how many standard deviations the realizations are from the mean can be provided. This information may be useful to detect causes of non-convergence due to physically unrealistic realizations.

An object to compute the value of the performance function(s) for a given realization of the random variables is created by the `gFunEvaluator` command. The following five alternative implementations are available at this time:

```
gFunEvaluator Basic
gFunEvaluator OpenSees -analyze numIncr? <dt?>
gFunEvaluator OpenSees -file filename
gFunEvaluator Tcl -file codeFileName
gFunEvaluator Matlab realizationFile codeFile gFunFile
```

The `Basic` g-function evaluator is used when only basic random variables are included in the performance function. In this case, no external analysis is needed to evaluate the performance function.

When the `OpenSees` g-function evaluator is specified, an OpenSees finite element analysis is executed each time the performance function is evaluated. Hence, response quantities from such analysis may be included in the performance function. The user may specify the number of analysis increments (and Δt for dynamic analysis) or a file with analysis commands. The latter option is included since all finite element analysis commands must be orchestrated from the reliability analysis algorithm. For a finite element analysis scheme that requires specification of more than the number of increments and Δt, the file option must be used. Each time the g-function evaluator is invoked the commands in the file are executed. As mentioned in Section 3.3.4, it is essential that the finite element domain reach the same final (pseudo-) time at each performance function evaluation. Variable time-stepping schemes may be handled by using the `-file` option of the OpenSees g-function evaluator, typically with the following file content:

```
set targetTime ...
```

```
set currentTime 0.0

while {$currentTime < $targetTime } {
     analyze 1
     set currentTime [getTime]

     if { ($targetTime-$currentTime)<$maxLambda } {

          set deltaLambda [expr $targetTime-$currentTime]
          # (Or choose some other scheme to approach the target time.)

          # Make integrator with step size equal to $deltaLambda

          analyze 1
     }
}
```

The `Tcl` g-function evaluator is provided so users may write their own algorithms in the Tcl language to evaluate performance functions. A finite element analysis is not automatically executed. If the user wants to execute an OpenSees finite element analysis as part of the performance function evaluation, it is necessary that the `reset` command be issued in each evaluation. All necessary computations and commands must be specified by the user in the file named by the `codeFileName` argument. Values of the current realization of the random variables are available as parameters `$x_1`, `$x_2`, etc., in the Tcl interpreter. At the end of the computations the user must make sure that all quantities in the performance function(s) have been computed and either `set` as parameters in the Tcl interpreter or stored in appropriate files. See Section 4.3 for syntax of the performance functions. When this option is used together with a finite difference `gradGEvaluator` scheme, there is a danger of round-off error. If sufficient accuracy cannot be provided, then the `perturbationFactor` option of the finite difference gradGEvaluator may be used to increase the perturbation size.

The `Matlab` g-function evaluator is available for users who have their own Matlab (The Mathworks, Inc. 2003) algorithms available to evaluate performance functions. The essential ingredients are: (1) the user must have the dynamic link libraries *libeng.dll*, *libmi.dll*, *limmx.dll* and *libut.dll* available in the directory from where OpenSees is run, (2) realizations of the random variables are printed to the file named as the `realizationFile` argument, (3) a Matlab file named as the `codeFile` argument is executed, and (4) the values of the performance functions are stored in a column in a file named as the `gFunFile` argument. Each row provides one performance function value and the row number corresponds to the performance function number.

Gradients of the performance functions with respect to the random variables are computed by the `gradGEvaluator` object. Currently two alternatives are available:

```
gradGEvaluator OpenSees <-check>
gradGEvaluator FiniteDifference <-pert arg1?> <-check>
```

The first command makes use of the DDM implementations available in OpenSees, while the second command perturbs the values of the random variables and re-evaluates the performance functions to obtain a forward finite difference estimate of the gradient vectors. The perturbation is equal to

$$\Delta x_i = \frac{\sigma_i}{\text{arg1}} \tag{4.1}$$

where σ_i is the standard deviation of random variable x_i. The default value of the perturbation factor `arg1` is 1000, but can be modified by the user. The `-check` flag is available to have the gradient vector printed to the screen. This option may be useful to developers for debugging purposes; comparison of gradients computed by finite difference and the DDM can be obtained by using this option.

A search direction object to obtain the search direction vector in the algorithm for finding the design point is created by the `searchDirection` command. Specific implementations at this time are:

```
searchDirection iHLRF
searchDirection PolakHe
searchDirection SQP c_bar? e_bar?
searchDirection GradientProjection
```

See Section 3.5 for details on the four algorithms. $\bar{c}$ and $\bar{e}$ are parameters of the sequential quadratic programming (SQP) algorithm. A root-finding algorithm to find the limit-state surface must be provided before a Gradient Projection algorithm is created.

A step size selection object to obtain a step size in the line search along a search direction is created by the `stepSizeRule` command. Two specific implementations are available at this time. The simplest gives the user an opportunity to force the step size to be fixed throughout the search:

```
stepSizeRule Fixed stepSize?
```

`stepSize` is the user-selected constant step size. The alternative is to employ the Armijo line search algorithm. The current implementation of the Armijo rule is tailor-made for finite element reliability analysis. The command reads (all on one line):

```
stepSizeRule Armijo -maxNum arg1? -base arg2? <-print arg?>
         <-initial b0? numSteps?>  <-sphere radius? dist? evol?>
```

where `arg1` represents the maximum number of step size reductions before the step size is accepted; `arg2` represents the base number b in the step size value b^k, where $k \geq 0$ is the smallest integer satisfying the merit function check (see next item). The optional arguments in the command address

an important issue in finite element reliability analysis. In parts of the outcome space of the random variables the performance function may not be computable due to non-convergence of the finite element algorithm. The `-initial` and `-sphere` flags are available to avoid trial steps too far out in the failure domain. With the `-initial` option the user specifies the value b_o in Eq. 3.83. After `numSteps` trial steps, the value of this parameter is restored to $b_o = 1.0$. The alternative `-sphere` option allows the user to define a hyper-sphere, within which the trial steps are restricted to stay. The sphere is defined in the standard normal space and its radius is equal to $\|y\|$ =`radius`. The user should select a conservative initial value of the radius to avoid trial steps in the failure domain. Since the design point may not be located inside the hyper-sphere, the radius may need to be increased. The evolution of the radius is governed by the input argument `evol` in the following manner. If two consecutive trial points have been located within a distance of `dist` of the hyper-sphere surface, then the radius is increased by the amount specified by the `evol` argument. The default values of the parameters are as follows: `arg1` = 10, `arg2` = 0.5, `b0` = 1.0, `numSteps` = 2, `radius` = 50.0, `dist` = 0.1 and `evol` = 0.5. The user may consider using `b0`=0.3 or `radius`=3.0 to invoke the above-mentioned remedies in typical finite element reliability problems.

A root finding algorithm is used by the gradient projection search algorithm and by a tool to visualize the limit-state surface. Other applications are also possible. The `rootFinding` object is created by the command:

```
rootFinding type  -maxIter arg1?  -tol arg2?  -maxStepLength arg3?
```

Currently, the `type` can be `Secant` or `ModNewton`, corresponding to the secant method and the modified Newton method, respectively. `arg1` denotes the maximum number of iterations, while `arg2` denotes the tolerance measure on the performance function value, scaled by the initial value. The argument `arg3` can be used to limit the length of the steps of the search for the root. This may be useful in nonlinear finite element reliability problems, where too large a step may lead to non-convergence of the finite element analysis.

A merit function object to determine the suitability of a step size is created by the `meritFunctionCheck` command. The following specific implementations are available at this time:

```
meritFunctionCheck AdkZhang -add arg? -multi arg? -factor arg?
meritFunctionCheck PolakHe -factor arg?
meritFunctionCheck SQP -factor arg?
meritFunctionCheck criteriaReduction
```

The arguments `add` and `multi` are used to compute the factor c so that $c > \frac{\|u\|}{\|\nabla G\|}$ by the following equation $c = (multi)\,\frac{\|u\|}{\|\nabla G\|} + (add)$, see Section 3.5.5. Default values are `multi`=2 and `add`=10. The

argument `-factor` is used in the equation f_1 (old and new trial point) $\leq$ *factor* λ $f_2(\ldots)$, which is a typical format of a merit function check. The `criteriaReduction` type requires that all convergence criteria be improved to accept the step size. This option is not scientifically based and may jam the algorithm, but is included for research purposes.

A convergence check object to determine if a design point has been found is created by the `reliabilityConvergenceCheck` command. At this time there are two convergence check objects available, each involving two convergence criteria (see Section 3.5.2):

```
reliabilityConvergenceCheck Standard -e1 arg?  -e2 arg?
reliabilityConvergenceCheck OptimalityCondition -e1 arg?  -e2 arg?
```

For the `standard` convergence check `e1` governs the convergence criterion $\frac{G}{G_o} \leq e_1$, which determines the closeness of the design point to the limit-state surface, while `e2` governs the convergence criterion $\|\mathbf{u} - \boldsymbol{\alpha}^T \mathbf{u} \boldsymbol{\alpha}^T\| \leq e_2$, which determines how closely the gradient vector points towards the origin in the standard normal space. The alternative convergence check examines the optimality conditions for the constrained optimization problem.

A start point object is used either as a starting point for the search for the design point, or as the center of the sampling density in an importance sampling analysis. It is created by the `startPoint` command:

```
startPoint Mean
startPoint Origin
startPoint Given
startPoint -file filename
```

These options respectively select the start point as the mean of the random variables, the origin in the standard normal space, the values given by the `startPt` argument of the random variables, or a vector provided in a file with name specified by the `filename` argument. The latter vector must be specified in the original space of random variables.

An object responsible for finding the design point is created by the `findDesignPoint` command:

```
findDesignPoint StepSearch -maxNumIter arg? <print option>
```

The search is performed by a step-by-step search scheme. Each trial point is determined by computing a step size in a line search along a selected search direction. Objects to perform these tasks must have been created before the `findDesignPoint` object can be instantiated. The print option enables the user to have the trial points or the design point printed to a file named `filename`:

```
-printAllPointsX filename
-printAllPointsY filename
-printDesignPointX filename
-printDesignPointY filename
-printCurrentPointX filename
-printCurrentPointY filename
```

In the above, the suffix `X` denotes the vector in the original space, whereas `Y` denotes the vector in the standard normal space. This option is useful for the restart of a design point search.

A random number generator object is created by the `randomNumberGenerator` command. One specific implementation is available at this time, namely employing the standard library function available in the programming language C++:

```
randomNumberGenerator CStdLib
```

When a generated realization of independent standard normal random variables is used by the sampling analysis object then the pre-selected probability transformation object transforms the realization into the original space for subsequent performance function evaluation.

An object responsible for finding the curvatures of the limit-state surface at the design point is created by the `findCurvatures` command. At this time only the first principal curvature can be computed according to the method developed by Der Kiureghian and DeStefano (1991):

```
findCurvatures firstPrincipal
```

4.5 ANALYSIS EXECUTION AND RESULTS

Eight analysis types are available in the reliability module of OpenSees. This section describes the corresponding commands to execute them. Needed analysis tools must have been specified prior to any of these commands. During the course of a reliability analysis, status information may be printed to a file or to the computer monitor. The complete results from a successful analysis are printed to an output file with a name specified by the user, as shown below.

A FORM analysis object is created and the corresponding analysis is executed by the following command:

```
runFORMAnalysis outputfilename <-relSens arg1?>
```

The optional flag `arg1` is set to 1 if reliability sensitivities (the sensitivity of the reliability index and estimated probability of failure with respect to means and standard deviations) are desired. A

probability transformation and a find-design-point algorithm (with their respective analysis tools) must have been created before the FORM analysis object is created. The results in the output file are self-explanatory except for the vectors `x*`, `u*`, `alpha`, `gamma`, `delta` and `eta`, as described below (see also Chapter 2.12.2):

- `x*` is the design point realization of the random variables in the original space.
- `u*` is the design point realization of the random variables in the standard normal space.
- `alpha` is the negative normalized gradient of the limit-state function at the design point in the standard normal space. This vector is also an importance measure for the random variables when no correlation is present. High absolute value indicates high importance. A negative value identifies a resistance-type variable and a positive value indicates a load-type variable.
- `gamma` is the importance vector for the original random variables when correlation is present. In the case of independent random variables `gamma`=`alpha`.
- `delta` is the sensitivity of the reliability index β with respect to the mean of each random variable, as scaled by the corresponding standard deviation. This vector is scaled to unit magnitude in OpenSees.
- `eta` is the sensitivity of the reliability index β with respect to the standard deviation of each random variable, as scaled by the same standard deviation. This vector is scaled to unit magnitude in OpenSees.

A sampling analysis object is created and the corresponding analysis is executed by the following command (all on one line):

```
runSamplingAnalysis outputfilename  -type arg1 -maxNum arg2?
            <-targetCOV arg3?> <-stdv arg4?> <-print arg5?>
```

The order of the optional arguments is arbitrary. `arg1` must be given as either `failureProbability` or `responseStatistics`, `arg2` denotes the maximum number of simulations, `arg3` denotes the target coefficient of variation of the estimate (either the estimate of failure probability or the estimate of the mean) (default = 0.05), `arg4` denotes the standard deviation of the sampling distribution (default = 1.0) and `arg5` is a print flag with the following meaning: If `arg5` is set to 0, then the status of the sampling analysis is not printed to the screen nor to a file. If `arg5` is set to 1, then the status after each sample is printed only to the screen. If `arg5` is set to 2, then it is assumed that the

sampling analysis is to be restarted. A file with the name "restart" preceding the output file name is created the first time the analysis is executed. Subsequent restart analyses read information from this file about the sampling number, the seed for the random number generation, and the status of the failure probability estimates and the corresponding coefficients of variation. The restart option is only available for failure probability sampling analysis.

The `startPoint` (see Section 4.4) is used as the center point for the sampling distribution. If a file containing a vector is specified as the center of the sampling distribution, then it is assumed that this is specified in the original space.

Note that it is inefficient to compute DDM sensitivities while performing a sampling analysis. Gradients may be costly to compute and they are not needed in the implemented sampling analysis schemes.

A probability transformation, a g-function evaluator and a random number generator must have been created before the sampling analysis object is created. The results in the output file are self-explanatory.

A mean out-crossing rate analysis object is created and the corresponding analysis is executed by the following command (all on one line):

```
runOutCrossingAnalysis outputfilename
       -results arg1? arg2? arg3?  -littleDt arg4?  arg5
```

`arg1` indicates the number of the time steps (of the dynamic finite element analysis) until the start of the interval in which out-crossing results are computed. `arg2` indicates the number of time steps until the end of the interval in which results are computed. `arg3` indicates the number of time steps between the points at which results are computed. `arg4` is the quantity δt explained in Section 3.4.6. `arg5` can be specified as either `-Koo` or `-twoSearches`. The latter option implies that two design point searches will be performed for each evaluation point, while the former employs the method developed by Koo (2003). See Section 3.4.6 for further details. It is assumed that the user has specified a discretized random process as time series for at least one load pattern and that the corresponding random variable positioners have been created to map random variables into this time series object. See Section 4.2 for the syntax.

A performance function evaluator, a gradient evaluator and an algorithm to find the design point must have been created before the out-crossing analysis object is created. The results in the output file are self-explanatory.

A parametric reliability analysis object is created and the corresponding analysis is executed by the following command (on one line):

```
runParametricReliabilityAnalysis outputfilename
  -par arg1?  -range arg2? arg3?  -numInt arg4?
```

`arg1` is the identification number of the parameter in the performance function, which is to be varied to create the probability curve. See Section 4.3 for the syntax of the performance function and Section 4.2 for the options of mapping the parameter values into the OpenSees finite element model. `arg2` and `arg3` denote the start value and the end value of the parameter and `arg4` represents the number of intervals of equal length between the start value and the end value. A FORM analysis and a subsequent reliability sensitivity analysis is performed at each discrete value of the parameter to obtain parametric reliability results. A gradient evaluator and an algorithm to find the design point must have been created before the parametric reliability analysis object is created. The results in the output file are self explanatory.

A first-order second-moment (FOSM) analysis object is created and the corresponding analysis is executed by the following command:

```
runFOSMAnalysis outputfilename
```

A performance function evaluator and a gradient evaluator must have been created before the FOSM object is instantiated. The results in the output file are self-explanatory.

A SORM analysis object is created and the corresponding analysis is executed by the following command:

```
runSORMAnalysis outputfilename
```

An algorithm to compute curvatures must have been created before the SORM object is created. The results in the output file are self-explanatory.

A system reliability analysis object is created and the corresponding analysis is executed by the following command:

```
runSystemAnalysis outputfilename  arg1
```

where the only option for `arg1` currently is `allInSeries`. This option indicates that all the performance functions are treated as belonging to a series system. No analysis tools are needed for the system analysis. However, similar to the SORM analysis, a prior FORM analysis is required. Currently, the system reliability does not read any information from file. Hence, the prior FORM analysis must have been run in the same Tcl interpreter session.

A visualization analysis to graphically visualize the performance function or the limit-state surface is available. This may be useful to investigate reasons for convergence problems or to obtain

a visual picture of a reliability problem. While the dimension of the space in which the limit-state surface is defined is equal to the number of random variables, only 2-D and 3-D visualization options are available. The user selects the axes (random variables) to be spanned by the visualization plot. The command reads as follows (all on one line):

```
runGFunVizAnalysis outputfile -space arg1 -funSurf arg2 ...
   <-dir arg3 arg4> <-convResults filename>
```

`arg1` should be set to either `X` or `Y`, depending on the space in which the input data is given. `arg2` can be either `function` or `surface`, depending on whether the user wishes to visualize the performance function itself or the limit-state surface. The `-dir` flag is used only in the visualization of the limit-state surface to indicate the direction along which the search for the limit-state surface is to be performed. A root-finding algorithm is used in this direction. `arg3` can be either `rv` or `file`, depending on whether the search is to be conducted along the direction of random variable number `arg4` or along a vector specified in a file named `arg4`. The optional flag `-convResults` can be included to have results from the convergence check, including the merit function value, printed to a separate file. This is optional since the analysis then becomes more costly; the gradients need to be computed at each visualization point. The three dots following `arg2` can be replaced by either one of the following specifications of the axes to be spanned by the visualization plot:

```
-coords1 rv1? from1? to1? numPts1?
-coords2 rv1? from1? to1? numPts1? rv2? from2? to2? numPts2?
-file filename numPts?
```

If the `-coords` option is used then the coordinate axes of the random variable space is used. A 2-D or 3-D plot is obtained depending on whether one or two random variables are specified, distinguished by `-coords1` and `-coords2`. The value of all other random variables are as specified by the "start point" object. If the `-file` option is used then two or more points in the space of random variables must be stored in one column in a file named `filename`. `numPts` denotes the number of evaluations of either the performance function or the limit-state surface along straight lines stretched between those points. For both the `-coord` and the `-file` options it is essential that the specification of the `space` flag is consistent with the stored points or the to/from values. For instance, the user must specify the `X`-space if the search points are stored in that space.

A performance function evaluator object, and possibly a root-finding object, must have been created before the visualization analysis object is instantiated. When the `-convResults` flag is used then the results in the corresponding output file are arranged into columns in the following order:

1. Performance function value

2. Distance from the origin to the current point in the standard normal space

3. The value of the merit function at the current point

4. The value of convergence criterion number 1 at the current point

5. The value of convergence criterion number 2 at the current point

If the visualization is performed along vectors between trial search points, then a division line is printed in the file to distinguish the lines from each other. Such lines are characterized by containing only zeros. In all other circumstances the visualization results (the values of the performance function or the distance to the limit-state surface) are printed to a matrix in the corresponding output file. This matrix has dimensions equal to the number of points along the axes specified by the `-coord` or `-file` options.

4.6 RESPONSE SENSITIVITY ANALYSIS BY DIRECT DIFFERENTIATION

This section presents the commands currently available in OpenSees to compute response sensitivities by direct differentiation. By setting up a "sensitivity algorithm" and a "sensitivity integrator," DDM sensitivity results are generated, which can then be used in a reliability analysis scheme or inspected as stand-alone results from a deterministic ordinary finite element analysis.

Two alternatives are available to inspect sensitivity results in OpenSees. The first is by Tcl commands that have been added to the interpreter:

```
sensNodeDisp nodeNum? dofNum? gradNum?
```

Alternatively, results can be obtained by creating recorder objects. This is analogous to the way ordinary finite element results are obtained. To use the latter option, add `-sensitivity arg?` to an ordinary recorder command. Then, instead of recording the response quantity itself, its sensitivity is recorded. In this command, `arg?` denotes the gradient number, namely the tag number of the variable in question. A typical command to set up a recorder for a DDM sensitivity analysis of a response quantity reads:

```
recorder Node file.out disp -time -node 6 -dof 1 -sensitivity 1
```

DDM sensitivities are computed only if a so-called sensitivity algorithm has been created by the command:

```
sensitivityAlgorithm arg1 <arg2>
```

The `arg1` flag can be either `-computeAtEachStep` or `-computeByCommand`. The former alternative must be used in all dynamic analysis and in all inelastic analysis. This is due to the need for sensitivity results from the previous step in such analysis. When the `-computeAtEachStep` flag is used, the response gradients are computed automatically at each step of the finite element analysis. In elastic static analysis the `-computeByCommand` option is sufficient. The sensitivities can then be computed at any time by issuing the Tcl command `computeGradients`. The optional `arg2` flag can be set to `-parameters` but is by default set to `-randomVariables`. This relates to whether the user wants sensitivity results with respect to random variables or with respect to parameters defined by the `parameterPositioner` command. The latter option is useful for stand-alone sensitivity analysis.

The sensitivity algorithm requires a sensitivity integrator to have been created:

```
sensitivityIntegrator arg1
```

This is the object responsible for assembling the right-hand side of the sensitivity equation. `arg1` can be either `-static` or `-dynamic`. If a dynamic sensitivity integrator is specified, then an additional requirement is present. Namely, the ordinary integrator that has been specified for the ordinary finite element analysis must have sensitivity capabilities. Currently, only one such "sensitivity enabled" dynamic integrator is available (all on one line):

```
integrator NewmarkWithSensitivity gamma? beta?
  <alphaM? betaK? betaKinit? betaKcomm?>
```

This command is identical to the one used to create an ordinary Newmark integrator object, except for the `NewmarkWithSensitivity` type.

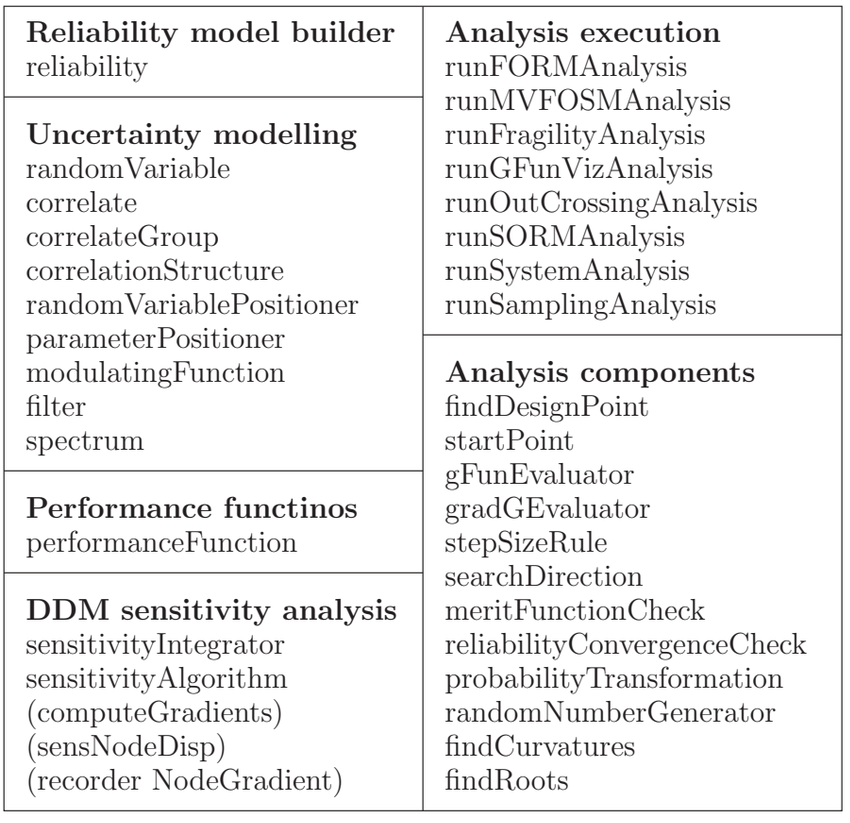

Figure 4.1: Overview of Tcl commands for reliability and sensitivity analysis in OpenSees.

Table 4.1: Syntax to identify material, section and element parameters available for uncertainty characterization.

Steel01	E, σ_y, b
SmoothSteel01	E, σ_y, b
Concrete01	f_c, ϵ_{co}, f_{cu}, ϵ_{cu}
SmoothConcrete01	f_c, ϵ_{co}, f_{cu}, ϵ_{cu}
BoucWen	α, k_o, n, γ, β, A_o, δ_A, δ_ν, δ_η
Hardening	E, σ_y, H_{kin}, H_{iso}
J2Plasticity	E, ν, σ_y, H_{iso}, H_{kin}
GeneralizedPlasticity	E, ν, σ_y, H_{iso}, H_{kin}, β, δ
Truss	A
ElasticBeamColumn	E, A, I (for the 2-D case)
DispBeamColumn	(this object does not contain physical parameters)
NonlinearBeamColumn	(this object does not contain physical parameters)
BeamWithHinges	E, A, I_z (for the 2-D case)
Quad	(this object does not contain physical parameters)

5 Numerical Examples and Case Studies

A number of examples are presented in this chapter. The main purpose is to demonstrate the new features and analysis capabilities of OpenSees. The presentation starts with simple problems and proceeds to a comprehensive example involving reliability analysis of a highway bridge used as a "testbed" structure by the Pacific Earthquake Engineering Research Center. The chapter is concluded with case studies discussing the problem of computing mean out-crossing rates of nonlinear response to stochastic input.

5.1 A BASIC EXAMPLE

The first example model is termed "basic" since the performance function is defined explicitly in terms of the basic random variables. A finite element analysis is not needed to evaluate it. This example is also analyzed by the general-purpose structural reliability code CalREL (Liu *et al.* 1989) and the results are compared. Three random variables are present. The first is assigned a lognormal probability distribution with mean 500 and standard deviation 100. The second is also lognormal but with mean 2000 and standard deviation 400. The last random variable is assigned the uniform probability distribution with mean 5 and standard deviation 0.5. The correlation structure between the random variables is specified in terms of the following correlation matrix:

$$[\rho_{ij}] = \begin{bmatrix} 1.0 & 0.3 & 0.2 \\ 0.3 & 1.0 & 0.2 \\ 0.2 & 0.2 & 1.0 \end{bmatrix} \tag{5.1}$$

The performance function reads (Liu *et al.* 1989, Example 2):

$$g(\mathbf{x}) = par_1 - \frac{x_2}{1000\ x_3} - \left(\frac{x_1}{200\ x_3}\right)^2 \tag{5.2}$$

where $par_1 = 1.0$ is a deterministic parameter.

First a FORM reliability analysis is performed. Initially, the following input is used for the analysis in OpenSees:

```
reliability
randomVariable 1 lognormal  500.0   100.0  500.0
randomVariable 2 lognormal  2000.0  400.0 2000.0
randomVariable 3 uniform    5.0       0.5    5.0
correlate   1 2  0.3
correlate   1 3  0.2
correlate   2 3  0.2
set a "{x_2}/(1000.0*{x_3})"
set b "{x_1}/(200.0*{x_3})"
performanceFunction 1 "1.0 - $a - $b*$b"
probabilityTransformation Nataf -print 0
reliabilityConvergenceCheck Standard -e1 1.0e-3 -e2 1.0e-3 -print 1
gFunEvaluator Basic
gradGEvaluator FiniteDifference -pert 1000
searchDirection iHLRF
meritFunctionCheck AdkZhang -multi 2.0 -add 10.0 -factor 0.5
stepSizeRule Armijo -maxNum 50 -base 0.5 -print 0
startPoint Mean
findDesignPoint StepSearch -maxNumIter 100
runFORMAnalysis FORMoutput.out
```

As seen, the search for the design point is performed by the improved HL-RF algorithm and it starts at the mean point.

The search for the design point converges in 8 iterations. The following records are printed to the screen during the analysis:

```
FORM Analysis is running ...
Limit-state function number: 1
 Limit-state function value at start point, g=0.35
 STEP #0: check1=( 1.000e+000), check2=(6.798e-002), dist=0.126840
 STEP #1: Armijo trial point rejected; reducing step size...
 .......: check1=( 4.168e-001), check2=(8.174e-002), dist=1.169291
 STEP #2: check1=( 3.782e-002), check2=(1.197e-001), dist=1.832142
 STEP #3: check1=( 6.233e-003), check2=(7.790e-002), dist=1.767500
 STEP #4: Armijo trial point rejected; reducing step size...
 .......: check1=( 3.693e-003), check2=(1.696e-002), dist=1.767607
 STEP #5: check1=( 1.165e-004), check2=(1.087e-002), dist=1.772304
 STEP #6: Armijo trial point rejected; reducing step size...
 .......: check1=( 6.924e-005), check2=(2.166e-003), dist=1.772303
 STEP #7: check1=( 6.385e-006), check2=(1.268e-003), dist=1.772392
 STEP #8: check1=( 3.523e-007), check2=(7.677e-004), dist=1.772399
Design point found!
Done analyzing limit-state function 1, beta=1.7724
FORMAnalysis completed.
```

Upon convergence the following output is stored in the output file:

```
########################################################################
#  FORM ANALYSIS RESULTS, LIMIT-STATE FUNCTION NUMBER 1                #
#                                                                      #
#  Limit-state function value at start point: ......... 0.35          #
#  Limit-state function value at end point: ........... -1.233e-007 #
#  Number of steps: ................................... 8             #
#  Number of g-function evaluations: .................. 39            #
#  Reliability index beta: ............................ 1.7724        #
#  FO approx. probability of failure, pf1: ............ 0.038164      #
#                                                                      #
# rv#     x*          u*         alpha    gamma    delta    eta      #
#  1    6.319e+002  1.281e+000  0.72330  0.69625 -0.59008 -0.78831  #
#  2    2.320e+003  4.815e-001  0.27190  0.36613 -0.34354 -0.24828  #
#  3    4.526e+000 -1.126e+000 -0.63475 -0.61740  0.63013 -0.59776  #
########################################################################
```

The 39 evaluations of the performance function include 9 evaluations of the performance function and its gradient by finite difference at the start point and the 8 trial points ($9 \cdot 4 = 36$) plus 3 evaluations during line searches that required reduction of the step size. The above results correspond to those presented by Liu *et al.* (1989), except that 15 steps are reported for CalREL as opposed to 8 for OpenSees. This is due to the use of the *modified* HLRF algorithm in CalREL while the *improved* HLRF algorithm (Zhang and Der Kiureghian 1997) is used in OpenSees.

It is of interest to investigate the performance of the alternative search algorithms for this problem. First the Gradient Projection algorithm (Section 3.5.4) is employed. To this end, the `meritFunctionCheck` object is removed from the input file and a root-finding algorithm is introduced. In addition, the step size and search direction algorithms are modified:

```
rootFinding  Secant  -maxIter 50  -tol 1.0e-2  -maxStepLength 1.0
stepSizeRule  Fixed  -stepSize 1.0
searchDirection  GradientProjection
```

The following output is displayed during the analysis:

```
FORM Analysis is running ...
Limit-state function number: 1
 Limit-state function value at start point, g=0.35
 STEP #0: check1=( 1.000e+000), check2=(6.798e-002), dist=0.126840
 STEP #1: check1=( 4.451e-003), check2=(2.223e-001), dist=1.798900
 STEP #2: check1=( 1.298e-005), check2=(1.258e-001), dist=1.781871
 STEP #3: check1=( 5.475e-003), check2=(7.073e-002), dist=1.767726
 STEP #4: check1=( 2.294e-007), check2=(4.237e-002), dist=1.773418
 STEP #5: check1=( 6.778e-004), check2=(2.515e-002), dist=1.771825
 STEP #6: check1=( 1.032e-006), check2=(1.520e-002), dist=1.772529
```

```
 STEP #7: check1=( 9.100e-005), check2=(9.144e-003), dist=1.772325
 STEP #8: check1=( 1.233e-004), check2=(5.563e-003), dist=1.772251
 STEP #9: check1=( 1.294e-004), check2=(3.361e-003), dist=1.772223
 STEP #10: check1=( 1.415e-004), check2=(2.039e-003), dist=1.772213
 STEP #11: check1=( 1.386e-004), check2=(1.233e-003), dist=1.772209
 STEP #12: check1=( 1.377e-004), check2=(7.478e-004), dist=1.772208
Design point found!
Done analyzing limit-state function 1, beta=1.77221
FORMAnalysis completed.
```

The number of steps with the Gradient Projection algorithm is comparable to the iHLRF algorithm. However, the Gradient Projection algorithm requires 67 evaluations of the performance function compared to 39 for the iHLRF algorithm. This is due to the root-finding procedure of the Gradient Projection algorithm to project the trial points onto the limit-state surface. It is verified in the output file that the same design point is found.

Next, the Polak-He algorithm is employed. For this purpose the root-finding algorithm is removed and the Armijo step size rule from the iHLRF analysis is re-introduced. Furthermore, the search direction and the merit function check are specified as:

```
searchDirection  PolakHe
meritFunctionCheck  PolakHe  -factor 0.5
```

In addition, the performance function must be scaled due to the properties of the Polak-He algorithm discussed in Section 3.5.6. As seen above, the value of the performance function at the mean point is 0.35. Based on this information, it is selected to multiply the performance function by a factor of 15.0, that is:

```
performanceFunction 1 "15.0*(1.0 - $a - $b*$b)"
```

Keeping all other input the same, the following analysis progress is now observed:

```
FORM Analysis is running ...
Limit-state function number: 1
 Limit-state function value at start point, g=5.25
 STEP #0: check1=( 1.000e+000), check2=(6.798e-002), dist=0.126840
 STEP #1: check1=( 2.451e-001), check2=(2.803e-001), dist=2.143408
 STEP #2: check1=( 8.753e-002), check2=(1.006e-001), dist=1.895190
 STEP #3: check1=( 3.075e-002), check2=(1.551e-002), dist=1.813694
 STEP #4: check1=( 9.815e-003), check2=(6.251e-003), dist=1.785576
 STEP #5: check1=( 3.084e-003), check2=(1.602e-003), dist=1.776539
 STEP #6: check1=( 9.655e-004), check2=(4.771e-004), dist=1.773694
Design point was found!
Done analyzing limit-state function 1, beta=1.77369
FORMAnalysis completed.
```

In this case the total number of evaluations of the performance function is only 28 (4 per trial point due to the use of finite difference to compute the gradient). However, the scaling of the performance function has an important influence on the convergence of the Polak-He algorithm. Figure 5.1 shows the number of trial points for this algorithm with different start values of the performance function. It is clear that proper scaling of the performance function is essential when using the Polak-He algorithm.

Finally, the SQP algorithm discussed in Section 3.5.7 is applied to the problem. The same input as the previous case is used, except for the following modifications:

```
searchDirection  SQP -c_bar 200.0  -e_bar 0.5
meritFunctionCheck  SQP -factor 0.5
```

This leads to the following display during the analysis:

```
FORM Analysis is running ...
Limit-state function number: 1
 Limit-state function value at start point, g=0.35
 STEP #0: check1=( 1.000e+000), check2=(6.798e-002), dist=0.126840
 STEP #1: Armijo trial point rejected; reducing step size...
 .......: check1=( 4.168e-001), check2=(8.174e-002), dist=1.169291
 STEP #2: Armijo trial point rejected; reducing step size...
 .......: check1=( 1.991e-001), check2=(8.555e-002), dist=1.499568
 STEP #3: Armijo trial point rejected; reducing step size...
 .......: check1=( 9.661e-002), check2=(3.124e-002), dist=1.641192
 STEP #4: check1=( 2.840e-003), check2=(4.043e-003), dist=1.776221
 STEP #5: check1=( 1.187e-006), check2=(8.522e-004), dist=1.772405
Design point found!
Done analyzing limit-state function 1, beta=1.7724
FORMAnalysis completed.
```

As seen, 5 steps and 3 step size reductions are used in the SQP algorithm to find the design point; $6 \cdot 4 + 3 = 27$ calls were made to evaluate the performance function.

In summary, for this example the SQP algorithm is most efficient followed closely by the Polak-He algorithm when the performance-function is properly scaled. Next in efficiency comes the iHLRF algorithm while the Gradient Projection algorithm ranks last in efficiency. Obviously, one example is not sufficient to draw conclusions regarding the behavior of the different search algorithms. Results of further testing are presented in the following sections.

An importance sampling analysis around the design point is now performed by executing the following command (on one line):

```
runSamplingAnalysis file.out -type failureProbability
    -variance 1.0 -maxNum 10000 -targetCOV 0.01 -print 0
```

The output file reads:

```
######################################################################
#  SAMPLING ANALYSIS RESULTS, LIMIT-STATE FUNCTION NUMBER   1       #
#                                                                    #
#  Reliability index beta: ........................... 1.8145      #
#  Estimated probability of failure pf_sim: ........... 0.034799    #
#  Number of simulations: ............................. 10000       #
#  Coefficient of variation (of pf): .................. 0.015994    #
#                                                                    #
######################################################################
```

It is seen that after 10,000 samples the probability estimate is obtained with a coefficient of variation of 1.6%. The slightly higher reliability index of 1.81 from the sampling analysis as compared to the FORM value of 1.77 suggests that the limit-state surface is curved away from the origin around the design point. This may be investigated by employing the visualization analysis option in OpenSees. Figure 5.2 shows different views of the limit-state surface in the standard normal space with the following modification: To make it easier to interpret the surface the values exceeding the plot limits, namely $(\mathbf{y}^* \pm 1)$, are set equal to ± 1. The trial steps of the iHLRF algorithm are shown by squares and connected by lines. The start point, which is close to the origin in the standard normal space, is outside the plot limits. Figure 5.2 confirms that the limit-state surface is curved away from the origin around the design point.

The parametric reliability analysis option in OpenSees is used to create a complementary CDF for the random variable $X = \frac{x_2}{1000\, x_3} - \left(\frac{x_1}{200\, x_3}\right)^2$ as defined in the performance function in Eq. (5.2). This is done by automatically varying the parameter $0.3 < par_1 < 1.3$. By reliability sensitivity analysis, as described in Section 3.4.4, the corresponding PDF is computed. Figure 5.3 shows the result. Each point on the complementary CDF curve indicates one FORM reliability analysis.

5.2 PUSH-OVER ANALYSIS OF 3-D TRUSS

In this example the performance functions are specified in terms of response quantities from an OpenSees finite element analysis involving a 3-dimensional truss model. DDM response sensitivity equations for both 2-D and 3-D nonlinear truss elements are available in OpenSees. Typical reliability results from OpenSees are presented to demonstrate features of the software framework.

The truss structure shown in Figure 5.4 is analyzed. Concentrated nodal loads are applied in the x-direction at the four top nodes. The deterministic and uncertain model parameters are listed in

Table 5.1. Four different material models are considered: (1) elastic material, (2) bi-linear material, (3) smoothed bi-linear material (as presented in Section 2.8) and (4) Bouc-Wen material (as presented in Section 2.6). Example stress-strain curves for the mean-value materials are shown in Figure 5.5. The bi-linear and smoothed bi-linear materials do not exhibit degrading behavior, as opposed to the Bouc-Wen material model as shown in the figure. Degradation is, however, not considered in this static example. While the load is applied along the x-axis direction, the truss is rotated 30 degrees around its vertical axis compared to being aligned with axis directions. Hence, the example is not symmetric and must be analyzed by a 3-D model.

First a linear static finite element reliability analysis is performed. 232 random variables are present according to Table 5.1. Of interest is the probability that the top displacement in the x-direction exceeds 110.0mm. The performance function for this event is formulated as

$$g = 110.0 - \text{x-displacement}_{node\ 21} \tag{5.3}$$

Using the iHLRF algorithm, the first trial point is close to the design point for this linear structural problem. The search converges in 2 steps to a reliability index $\beta = 3.83$.

Next, an inelastic static finite element reliability analysis is performed with the bi-linear material model. 392 random variables are present according to Table 5.1. We now decide to seek the probability that the top displacement in the x-direction exceeds 500.0mm. Hence, we define:

$$g = 500.0 - \text{x-displacement}_{node\ 21} \tag{5.4}$$

An additional goal of this analysis is to identify which parameters/elements are important for this performance function.

Using the iHLRF algorithm, the total analysis time was 4 minutes and 16 seconds on a 2.0 GHz processor computer. The screen output is shown below:

```
DONE CREATING 392 RANDOM VARIABLES
FORM Analysis is running ...
Limit-state function number: 1
 Limit-state function value at start point, g=325.027
 STEP #0: check1=( 1.000e+000), check2=(2.410e-001), dist=0.260802
 STEP #1: Armijo starting gFun evaluation at distance 15.838...
 .......: Armijo trial point rejected; reducing step size...
 .......: Armijo starting gFun evaluation at distance 7.87007...
 .......: Armijo trial point rejected; reducing step size...
 .......: Armijo starting gFun evaluation at distance 3.88893...
 .......: Armijo trial point rejected; reducing step size...
 .......: Armijo starting gFun evaluation at distance 1.90422...
 .......: check1=( 1.862e-001), check2=(4.545e-001), dist=1.904219
```

```
 STEP #2: Armijo starting gFun evaluation at distance 1.89189...
 .......: check1=( 3.623e-003), check2=(2.075e-002), dist=1.891885
 STEP #3: Armijo starting gFun evaluation at distance 1.89531...
 .......: check1=( 2.237e-006), check2=(4.825e-004), dist=1.895307
Design point found!
Done analyzing limit-state function 1, beta=1.89531
FORMAnalysis completed.
```

Response gradients are computed by the DDM. Hence, as seen by the screen output, only 7 finite element analyses were performed. The recorded displacement response for each iteration of the search algorithm is shown in Figure 5.6. The results clearly show a problem commonly encountered in nonlinear finite element reliability analysis when searching for the design point. Namely, the start point is linear and the search algorithm tends to vastly overshoot the design point far into the nonlinear domain in the first step. This is discussed below.

The resulting reliability index of $\beta = 1.89$ has a corresponding FORM estimate of the failure probability of $p_{f1} = 0.029026$. Importance sampling around the design point gives the result $p_{fsi} = 0.0447$ ($\beta = 1.70$) with 6.5% coefficient of variation on the failure probability estimate after 1000 simulations. This indicates that the limit-state surface is curved towards the origin around the design point in the standard normal space.

As discussed in Section 3.6, parameter importance measures are available as a by-product of FORM analysis. These are useful in identifying structural elements and model parameters, which have significant influences on the performance of the structure. A ranking according to the importance vector $\boldsymbol{\gamma}$ defined in Section 3.6 is provided in Table 5.2. Figure 5.7 shows the corresponding element and node numbers. It is observed that the yield strengths of the vertical elements 4, 20 and 2 are most important, followed by the yield strengths of the surrounding cross braces. Based on physical intuition it is reasonable that elements 2, 4 and 20 are important for the given performance function. This exemplifies the usefulness of parameter importance measures from reliability analysis to gain physical insight into a problem. It is also interesting to observe that the uncertainty in the nodal coordinates is quite important, though the standard deviations are only 10.0mm.

The above reliability analysis converged to a reliability index of 1.89 for a displacement threshold of 500.0mm. An attempt to compute a probability estimate for the displacement threshold equal to 600.0mm using the same iHLRF algorithm fails. This is because the trial point after the first step in this algorithm is too far in the failure domain and the finite element analysis (for the selected analysis setup) does not converge. The remedies discussed in Section 3.5 are now applied to solve this problem. The effect of using alternative material models is also studied.

With a displacement threshold of 600.0mm the first trail point of the iHLRF algorithm is at a

distance $\|\mathbf{y}\| = 20.74$ from the origin in the standard normal space. The expected reliability index, on the other hand, is around 3.0. First the approach of Eq. (3.83) is employed, namely to reduce the step size of the first few steps of the search algorithm. The following step size command is used:

```
stepSizeRule  Armijo  -maxNum 10  -base 0.5  -initial 0.5 2
```

The reliability analysis then successfully converges to a reliability index of $\beta = 2.22$ in 4 steps with a total of 7 performance function evaluations. An importance sampling analysis around the design point yields the result $p_{f\,sim} = 0.0209$ ($\beta = 2.04$) with a coefficient of variation of 9% after 1000 simulations.

The option of using a "bounding sphere" is also tested. The following command is now used to create a step size selection object:

```
stepSizeRule  Armijo  -maxNum 5  -base 0.5  -sphere 5.0 0.5 1.0
```

With this command the trial points are restricted to the inside of a sphere with radius 5.0 in the standard normal space. If the design point is not found within the sphere, then the radius of the sphere is increased. This is done by increasing the radius by 1.0 when two consecutive trial steps are located within a distance of 0.5 of the surface of the sphere. This also turns out to be a successful remedy to the problem. The reliability analysis converges to $\beta = 2.22$ in 4 trial steps. This time the total number of performance function evaluations is 5; no trial points are rejected by performance function evaluations in the Armijo rule.

Next, the performance of the Polak-He search algorithm (see Section 3.5.6) is tested. This algorithm possesses steering parameters, which may guide the search to be performed in the safe domain. First the parameters γ and δ are set to their default values of 1.0. The performance function is scaled by a factor of 100 so that its value at the mean point is 4.25. The search algorithm produces the following output to the computer display:

```
DONE CREATING 392 RANDOM VARIABLES
FORM Analysis is running ...
Limit-state function number: 1
 Limit-state function value at start point, g=4.25027
 STEP #0: check1=( 1.000e+000), check2=(2.410e-001), dist=0.26080
 STEP #1: Armijo starting gFun evaluation at distance 0.262565...
 .......: check1=( 9.903e-001), check2=(2.413e-001), dist=0.26256
 STEP #2: Armijo starting gFun evaluation at distance 0.389141...
 .......: check1=( 9.807e-001), check2=(2.421e-001), dist=0.38914
 STEP #3: Armijo starting gFun evaluation at distance 0.559386...
 .......: check1=( 9.641e-001), check2=(3.013e-001), dist=0.55938
 STEP #4: Armijo starting gFun evaluation at distance 0.979161...
 .......: check1=( 8.747e-001), check2=(5.422e-001), dist=0.97916
```

```
 STEP #5: Armijo starting gFun evaluation at distance 2.14232...
 .......: check1=( 1.230e-001), check2=(2.941e-001), dist=2.14232
 STEP #6: Armijo starting gFun evaluation at distance 2.21184...
 .......: check1=( 1.798e-002), check2=(1.222e-001), dist=2.21184
 STEP #7: Armijo starting gFun evaluation at distance 2.22084...
 .......: check1=( 1.821e-003), check2=(5.021e-002), dist=2.22084
 STEP #8: Armijo starting gFun evaluation at distance 2.22137...
 .......: check1=( 2.025e-004), check2=(2.051e-002), dist=2.22137
 STEP #9: Armijo starting gFun evaluation at distance 2.22105...
 .......: check1=( 2.472e-004), check2=(8.395e-003), dist=2.22105
 STEP #10: Armijo starting gFun evaluation at distance 2.22083...
 .......: check1=( 1.312e-004), check2=(3.456e-003), dist=2.22082
 STEP #11: Armijo starting gFun evaluation at distance 2.22072...
 .......: check1=( 5.981e-005), check2=(1.434e-003), dist=2.22071
 STEP #12: Armijo starting gFun evaluation at distance 2.22067...
 .......: check1=( 2.595e-005), check2=(5.996e-004), dist=2.22066
Design point found!
Done analyzing limit-state function 1, beta=2.22067
FORMAnalysis completed.
```

It is interesting to observe that the problem of trial steps too far out in the failure domain does not occur with the Polak-He algorithm with the default parameters. However, 12 trial steps are necessary to obtain convergence as opposed to 4 steps for the iHLRF algorithm with the restricting sphere. Alternative choices for γ and δ were tried. As expected, a smaller value of γ leads to a slower approach towards the failure domain. For $\gamma = 0.1$, 108 trial steps are necessary. A search with $\gamma = 5.0$ requires 13 trial steps. The trial points quickly go out into the failure domain; however, maximum trial point distance from the origin is only $\|\mathbf{y}\| = 2.99$. Thereafter, the trial points approach the limit-state surface from inside the failure domain. This trend is also observed with higher values of γ. It is concluded that the Polak-He algorithm is an effective search algorithm for finite element reliability applications.

We now consider a smoothed steel material model together with a 600.0mm threshold and the original iHLRF algorithm. The value of the material parameters are provided in Table 5.1. The search algorithm converges in 3 steps with the reliability index $\beta = 2.19$. The reliability index does not change significantly due to the smoothing. However, an advantage in the search for the design point has been achieved. The maximum trial point distance from the origin is $\|\mathbf{y}\| = 9.3$, as opposed to $\|\mathbf{y}\| = 20.74$ with the bi-linear material. The improvement can be explained by the fact that a larger portion of the structure experiences nonlinearity at the mean point when a smoothed material model is used. This leads to a more reasonable estimate of the initial trial points.

Next, the Bouc-Wen material model is employed. 312 random variables are now present according to Table 5.1. Again we employ the original iHLRF algorithm and seek the probability that the top

displacement in the x-direction exceeds 600.0mm. The analysis converges in 13 steps with a total of 29 evaluations of the performance function. The reliability index $\beta = 2.61$ is obtained. The maximum trial point distance from the origin is $\|\mathbf{y}\| = 9.04$. These results indicate that the use of the Bouc-Wen material model leads to the same advantageous effect regarding avoiding trial points too far in the failure domain as the smooth material model. However, a greater number of trial steps were necessary to converge to the design point. An explanation for this is that the Bouc-Wen material model exhibits nonlinearity also in the initial part of the stress-strain relation. This implies that all members of the truss exhibit some degree of nonlinear behavior at the design point. As a consequence the performance function is more nonlinear, leading to an increased number of steps in the search for the design point as compared to the smoothed bi-linear material model.

5.3 REINFORCED-CONCRETE FIBER-FRAME EXAMPLE

Consider the two-bay, two-story reinforced concrete structure shown in Figure 5.8. A fiber cross section model is employed for the members of the frame. This enables the use of nonlinear uniaxial steel and concrete material models to realistically describe the behavior of the reinforcing steel and the confined and unconfined concrete. Particular attention is given to the importance of including uncertainty in the nodal coordinates. Most finite element reliability analysis examples in the literature neglect this source of uncertainty. Their inclusion in this report is made possible by the development of and implementation of shape sensitivity results in Chapter 1.5.

All material parameters, nodal coordinates and lateral loads are characterized as uncertain. Each member of the frame is assigned one random variable for the yield strength of its steel, for the compressive strength of its concrete, etc. In total, 94 random variables define the reliability problem, as listed in Table 5.3. The right-most column in the table indicates the correlation coefficient between the random variables for pairs of members.

Three performance functions are considered:

$$\begin{aligned} g_1 &= 0.02 \cdot (4.57\ m + 3.66\ m)\ -\ u_9 \\ g_2 &= 0.02 \cdot (3.66\ m)\ -\ (u_9 - u_8) \\ g_3 &= 0.02 \cdot (4.57\ m)\ -\ u_8 \end{aligned}$$

The first defines the failure event as the exceedance of the horizontal displacement at node 9 above 2% of the height of the building. The second defines a similar threshold for the interstory drift. The

third defines the failure event as the exceedance of the horizontal displacement at node 8 above 2% of the height of the first story.

Table 5.4 summarizes the reliability estimates. Importance sampling analysis is conducted for the first performance function, to demonstrate the good accuracy of the FORM result. Based on this observation, FORM analyses are carried out for the second and third performance functions with enhanced confidence. It is noted that a new importance sampling analysis is required for each performance function, since the design points differ. Furthermore, non-convergence in the finite element analysis is frequently experienced during the sampling analysis when the simulated point is too far in the failure domain. Provided the program does not terminate, which is usually not the case unless, e.g., zero stiffness is encountered, this situation is recorded as a failure event in OpenSees. The results in Table 5.4 indicate that the structure is least reliable with respect to the inter-story drift limit.

Figure 5.9 shows the complementary CDF and the PDF of the displacement at node 9 obtained by varying the drift threshold from 1% to 2%. Each point on the complementary CDF involves one reliability analysis. The corresponding PDF is obtained by the available reliability sensitivity analysis option in OpenSees. The figure shows results based on the bilinear material and the smoothed bi-linear material model with smoothing starting at 70% of yield strength. It is clear that the smoothing has little influence on the computed probabilities.

Table 5.5 shows the ranking of the 20 most important random variables for performance function g_1 for the importance vectors defined in Section 3.6. The value $\nabla g(\boldsymbol{\mu})_i \ \sigma_i$, namely the product of the gradient vector and the corresponding standard deviations, is the importance measure from a first-order second-moment analysis which requires one finite element reliability analysis. The most important variables are the upper and lower lateral loads, the yield stress of the reinforcing steel in the lower middle column, followed by the horizontal coordinates of columns at the foundation level. Several observations are made based on the results in Table 5.5. First, in this example the random variables are dependent, which implies that the importance vector $\boldsymbol{\gamma}$ must be considered instead of $\boldsymbol{\alpha}$. Second, it is seen that the importance measure obtained by a first-order second-moment analysis centered at the mean has similarities with $\boldsymbol{\gamma}$. This indicates that the structural behavior at the mean point is similar to the behavior at the design point. This is not always the case. It is common that the response at the design point is more nonlinear than at the mean point. Thirdly, four of the nodal coordinates enter the list of the 20 most important random variables according to the importance vector $\boldsymbol{\gamma}$, even with a standard deviation as small as 10.0mm. This is an interesting finding that corroborates observations made in other examples. It is also observed that the importance vector $\boldsymbol{\delta}$

exhibits the same characteristics as $\boldsymbol{\gamma}$. However, the numerical values of their elements, not shown in Table 5.5, are not identical. The vector $\boldsymbol{\eta}$, on the other hand, does not rank nodal coordinates equally high. The x-direction coordinate of node 4 ranks as number 21 in $\boldsymbol{\eta}$. This indicates that the reliability is not sensitive to changes in the standard deviation of the nodal coordinates. This emphasizes the importance of including uncertainty in nodal coordinates, even when the standard deviations are small.

The importance of the nodal coordinates can be explained in terms of the "P-Δ" effect. Perturbation of nodal coordinates leads to added bending moments due to axial force in the members. This is particularly the case when significant gravity loads are present. Since uncertainty in the loads is most significant, any uncertainty in the nodal coordinates leads to a significant uncertainty in the added bending moments due to such P-Δ effects. Hence, nodal coordinates may be considered as variables capable of inducing additional load. One may of course question the selected standard deviation of the nodal coordinates. Nevertheless, these and other examples studied have indicated that uncertainty in the nodal coordinates can be important and should be considered, if present.

5.4 I-880 HIGHWAY BRIDGE EXAMPLE

The Pacific Earthquake Engineering Research (PEER) Center has selected four "testbed" structures for numerical studies and testing of developed analysis tools (*http://www.peertestbeds.net*). In the present study, the I-880 highway bridge testbed is considered for reliability analysis. The finite element model of this structure is developed by Kunnath *et al.* (2003). A frame consisting of four bents is modeled and both static pushover and dynamic time-history analysis are conducted. Figure 5.10 depicts the general features of the model. Node and element numbers are identified for the benefit of the subsequent discussion of the reliability results. It is noted that the foundation is modeled in terms of soil spring elements. Furthermore, the columns are modeled with distributed hinges at the element ends. The cross sections of these hinges are fiber-discretized with realistic modeling of the confined and unconfined concrete and the reinforcement steel by uniaxial material models. The structure is analyzed as a free-standing frame with no interaction from adjacent bridge structures. The nodal coordinates of the model are presented in Table 5.6.

In this study, all parameters describing material properties, cross-section geometry and nodal coordinates are characterized as random. In fact, an important purpose of the reliability analysis is

to obtain parameter importance measures to identify which parameters are most importat. Tables 5.7, 5.8, 5.9 and 5.10 list the uncertainty characterizations used in the subsequent analyses. A total of 320 random variables are considered.

The static analysis model is addressed, for which the probabilistic analysis has three purposes. First, to obtain second-moment response statistics to investigate the propagation of uncertainty through the finite element model, second, to obtain importance measures and, third, to perform reliability analysis to estimate probabilities of achieving specified structural performance criteria. The displacement of node 15005 in the transverse direction is selected as the response quantity of interest. Figure 5.11 shows the pushover response at the mean-point of the random variables.

5.4.1 Performance Criteria

Four performance functions are considered for the static analysis, all involving the displacement response at node 15005, which is denoted u. The corresponding load factor is denoted λ. The main purpose is to demonstrate the capabilities of the reliability module in OpenSees. The first two criteria are used to investigate the probability of reaching a certain displacement u_o for a given load factor λ_o, and conversely, to reach a certain load factor λ_o for a given displacement level u_o. The expressions for these performance functions read:

$$g_1 = u_o - u_{(at\ \lambda_o)} \tag{5.5}$$

$$g_2 = \lambda_o - \lambda_{(at\ u_o)} \tag{5.6}$$

It is seen that probability distributions as sketched in Figure 5.12 can be obtained by varying the threshold u_o in g_1 and λ_o in g_2.

In the finite element analysis the load-displacement curve is usually obtained by a load-control analysis scheme. This is ideal for the evaluation of performance function g_1. g_2 can be evaluated either by switching to a displacement-control analysis scheme or by solving for $\lambda(u_o)$ by, e.g., linear interpolation between points on the load-displacement curve. In this work the latter approach is used.

The next two performance criteria are meant to address the issue of structural failure. For demonstration purposes it is assumed that loss of structural integrity can be expected when the stiffness of the structure down-crosses a certain small level. This limiting tangent is selected to be 20% of the elastic tangent (Vamvatsikos and Cornell 2002). The stiffness is computed directly from

the load-displacement curve by finite difference calculations. Figure 5.13 shows the load-displacement ($\lambda - u$) curve of Figure 5.11 together with the corresponding tangent $k = \frac{\mathrm{d}\lambda}{\mathrm{d}u}$ at the mean point. It is observed that the stiffness goes down in a nonuniform manner with the largest loss of stiffness occurring at displacements around 0.02m and 0.1m.

The following performance criterion is used to estimate the probability of the displacement response being less than a threshold u_o when structural failure according to the above criterion occurs:

$$g_3 = u_{\,(at\ 20\%\ tangent)} - u_o \tag{5.7}$$

Since $u_{\,(at\ 20\%\ tangent)}$ can be seen as a displacement capacity of the structure, g_3 represents a ductility limit-state function. Similarly, a strength-type performance criterion is prescribed in the form:

$$g_4 = \lambda_{\,(at\ 20\%\ tangent)} - \lambda_o \tag{5.8}$$

That is, with g_4 we seek the probability of reaching a certain load level before failure occurs.

5.4.2 Response Statistics

Before starting computationally intensive reliability analyses, it is useful to conduct second-moment (FOSM) analysis. As outlined in Sections 3.4.1 and 3.6, this analysis approach renders response statistics and useful variable importance measures.

FOSM results are obtained for all response quantities present in the performance criteria g_1 to g_4. First, second-moment response statistics for $u_{(at\ \lambda_o)}$ are obtained. For this purpose, λ_o is varied from 0.0 to 0.22 and the mean and standard deviation of the corresponding u along the load-displacement curve are obtained. The first-order approximation of the mean is equal to the load-displacement curve at the mean point of the random variables. The first-order approximation of the standard deviation involves an additional response sensitivity analysis. The results are shown in Figure 5.14. It is observed that the uncertainty in the response due to the uncertainty in the model parameters propagated through the finite element analysis increases along the load-displacement curve. In fact, the standard deviation of the displacement for load factors higher than 0.22 increases by orders of magnitude. This is due to the relatively high response sensitivity in this region of the load-displacement curve. Of course the true standard deviation may not increase so rapidly; nevertheless, this result shows that nonlinear response can be highly uncertain due to its higher sensitivity to model parameter uncertainties.

Next, the response quantity $\lambda_{(at\ u_o)}$ is considered. Results for the displacement threshold ranging from 0.0 to 0.45 meters are shown in Figure 5.15. It is again observed that the standard deviation of the load level steadily increases along the load-displacement curve.

Finally, FOSM analyses were performed for the response quantities $u_{\ (at\ 20\%\ tangent)}$ and $\lambda_{\ (at\ 20\%\ tangent)}$ in performance criteria g_3 and g_4. The resulting second-moments are listed in Table 5.11.

5.4.3 Importance Ranking by FOSM Analysis

A useful by-product of FOSM analysis is the importance measures in Eq. (3.103). Ranking of the random variables by these measures may be used to reduce the number of random variables before proceeding to more costly FORM or importance sampling reliability analysis. However, care must be exercised since the importance ordering at the mean point may be different from that at the design point. Furthermore, the importance ordering may change along the load-displacement curve. For the I-880 bridge, it has been observed that the importance measures from FOSM and FORM analyses are consistent. This is partly due to the fact that the response regimes at both the mean and design points are nonlinear. In situations where the response at the mean is linear, while the response at the design point is nonlinear, the two sets of measures may not be consistent. Furthermore, for the I-880 bridge it has been observed that the importance measures change along the load-displacement curve. This is natural since the initial response is linear, thus insensitive to such parameters as the yield strength parameters, whereas with increasing load the response is more sensitive to parameters that define the nonlinear behavior of the material. The trends observed when obtaining the FOSM results in Figure 5.14 are discussed below. A selection of more detailed results is provided in Tables 5.12 to 5.15.

In the initial region of the load-displacement curve, the stiffness of the soil springs is identified as the most important parameter. The stiffness and cross-sectional geometry of the horizontal elastic elements close to the bent with node 15005 also rank high. Nodal coordinates in the y-direction, in particular for the nodes of this bent, are among the top 25 most important parameters. This is remarkable, given the fact that the standard deviation is only 1.27cm. Parameters f'_c and ϵ_{c0} of the cover concrete of the plastic hinges and the Young's modulus of the elastic regions of elements 151 and 152 rank among the 40 most important parameters. Among the least important parameters in this region of the load-displacement curve are all the parameters related to material yielding, such as σ_y, the hardening parameters, f_{cu} and ϵ_{cu}.

In the yielding region of the load-displacement curve (say λ values in the range $\lambda > 0.17$), the yield strength parameters of both reinforcing steel and core/cover concrete of the columns are most important, followed by vertical and horizontal soil spring stiffnesses. y-direction and z-direction nodal coordinates still hold 10 of the 50 first positions in the importance ranking of all the 320 random variables. Properties of the elastic horizontal elements seem rather unimportant, though properties of element 153 rank as the 36th and 38th most important random variables in the model.

A transition between the two importance rankings discussed above is observed in the middle region of the load-displacement curve, around $\lambda = 0.15$.

The observed high importance of the nodal coordinates may seem counterintuitive. However, the explanation may again be found in the fact that perturbation of nodal coordinates leads to $P - \Delta$ effects due to columns becoming off-vertical. In this case, significant gravity loads are present. (In these analyses, gravity loads are applied before lateral loads are applied.) Since lateral loads generally carry significant importance in reliability analysis, the $P - \Delta$ effect leads to the observed results.

5.4.4 Reliability Analysis

FORM analysis is conducted for performance functions g_1, g_2, g_3 and g_4. When performing such analyses it is desirable to reduce the computational cost by neglecting the uncertainty in "unimportant" random variables. For this purpose, the importance rankings discussed in the previous section are used. For instance, for a performance function that is expected to have its design point in the yielding region, only the most important random variables for this region according to the FOSM analysis are characterized as uncertain. The remaining variables are set deterministically to their mean values. For this purpose the OpenSees command `rvReduction` is implemented. See Section 4.2. The question as to how many random variables must be included to adequately capture the uncertainty effects is addressed below.

We first consider performance function g_1. The load factor is set to $\lambda_o = 0.2$ and a series of FORM analyses are performed for varying threshold u_o. Figure 5.16 shows the resulting probabilities of failure (complementary CDF curve) and the corresponding derivative (PDF curve) obtained by reliability sensitivity analysis in OpenSees. Results are obtained for three cases. First the 10 most important random variables according to the FOSM analysis of $u_{(at\ \lambda_o=0.2)}$ are included. Then the uncertainty in the 100 most important random variables is included. Finally, all 320 random variables are included in the analysis.

It is observed that the results obtained by including the 100 most important random variables are in close agreement with those obtained by including all the 320 random variables. For this reason, in the subsequent reliability analyses only the 100 most important random variables for each performance function are considered.

Next, a number of FORM reliability analyses are performed for limit-state function g_2. The displacement threshold is selected as $u_o = 0.3$m. Figure 5.17 shows the resulting probability curves. It is noted that the distributions shown in Figures 5.16 and 5.17 are particularly accurate in the tail regions due to the asymptotic accuracy of FORM (Breitung 1989).

We now turn to the reliability analyses of performance functions g_3 and g_4. These are more challenging, in part due to the need for the second-derivative of the response. This requirement is understood by noting that the response tangent appears in the expressions for g_3 and g_4 and that the gradient of these performance functions are needed in the reliability analysis procedure. The finite difference method is used to compute these derivatives.

Initial efforts to find design points for g_3 and g_4 did not succeed. The reason for this may be found by observing that the tangent curve shown in Figure 5.13 exhibits a "jagged" behavior. This is due to many sudden changes in the stiffness due to yielding of individual material fibers. As a remedy for this problem, smooth material models are used. The original bi-linear steel material is replaced with the smoothed version developed in Section 2.8. The two smoothing parameters are chosen to be $\gamma = 0.5$ and $\eta = 4.0$. This implies that the smoothing circle starts taking effect at 50% of the yield stress. The material model is shown in Figure 5.18. The effects on the load-displacement curve and the corresponding tangent are shown in Figure 5.19. It is clear that the overall response does not change significantly as a result of the smoothing. In fact, with the smooth model it is possible to obtain convergence of the finite element analysis for higher load factors. The effect of smoothing on the tangent curve is more significant and is emphasized in Figure 5.20. As seen, the jagged behavior is replaced by a smooth curve. This turns out to lead to successful reliability analyses. Again the 100 most important random variables from FOSM analysis are included in the FORM analyses. The results are shown in Figure 5.21 for performance function g_3 and in Figure 5.22 for performance function g_4.

In the above, the displacement has been the primary response quantity of interest. Analogous results can be obtained for any other available response quantity, such as element forces, strains and stresses.

5.4.5 Importance Ranking by FORM Analysis

A number of results and importance vectors are available from the FORM analyses performed above. It is not regarded useful to include all this material in this report. However, it is noted that for the analysis of the I-880 bridge the importance measures from FOSM and FORM were consistent. Only small differences were observed. As mentioned earlier, this is most importantly because the nonlinear response characteristics for this structure are similar at the mean and design points.

5.4.6 Verification of Results by Sampling Analysis

It is of interest to verify the reliability results obtained by the FORM approximation. In this study it has proved useful to employ an importance sampling scheme around the design point for this purpose. This approach has the advantage of revealing possible gross errors in the FORM approximation. This approach is also computationally feasible as opposed to the crude Monte Carlo sampling schemes around the mean point. For this verification, we consider the performance function

$$g = 0.35 - u(\lambda_o = 0.2) \tag{5.9}$$

The FORM analysis converges in 4 iterations with the results $\beta = 2.26$ and $p_{f\,1} = 0.01189$.

An importance sampling analysis around the design point with 1000 samples leads to the estimate $p_{f,\,IS} = 0.01074$ with a coefficient of variation of 5.49%. This confirms the accuracy of the FORM approximation, though it indicates a weak curvature away from the origin around the design point of the limit-state surface in the standard normal space.

For demonstration purposes a crude Monte Carlo analysis is also executed. 2000 samples were made. However, this brought the coefficient of variation of the probability estimate down only to 26.6%. The obtained probability estimate, namely $p_{f,\,MC} = 0.0070$, is therefore not accurate. It is emphasized that 2000 evaluations of the performance function, that is, 2000 finite element analyses, are computationally costly for this model. The advantages of the FORM and importance sampling solution methods are thus highlighted.

5.4.7 Dynamic Time-Variant Reliability Analysis, Mean Out-Crossing Rates

Two reliability analysis types are considered for the dynamic analysis of the I-880 highway bridge. First, the mean out-crossing rate analysis discussed in Section 3.4.6 is employed. The results from this analysis are presented in this section. In the following section a performance function involving an accumulated damage measure is treated.

The filter properties and modulating functions used in defining the input process for this example are shown in Figure 5.23. A filtered train of standard normal pulses with $\Delta t = 0.1$ sec. is used. The filter with period 0.1 sec. and 5% damping is modulated by the function $q_1(t) = 0.14\, t^5\, e^{-1.9t}$ and the filter with period 0.3 sec. and 5% damping is modulated by the function $q_2(t) = 0.0035\, t^{10}\, e^{-2.5t}$. The target maximum standard deviation is selected to be $0.2\, g$. A corresponding sample ground motion is shown in Figure 5.24.

The structural parameters are set equal to their mean values. When linear material models are used for both steel and concrete, then the mean out-crossing rate shown in Figure 5.25 is obtained for the threshold u_o =0.1m at node 15005. By integration of the mean out-crossing rate over time, the upper bound to the probability of the displacement response exceeding the threshold u_o =0.1m during the time interval $0 < t < 10$ sec. is obtained. The result in this case is $p_f < 1.67 \cdot 10^{-3}$.

It is of interest to study the design point realization of the ground motion excitation and the response for the failure event at a given time. These are the most likely realizations that will give rise to the event of interest, i.e., the displacement response exceeding the threshold u_o =0.1m at the given time. Figure 5.26 shows the response at the design point for the mean out-crossing rate computated at $t = 4.5$ sec. It is observed that the response reaches the threshold u_o =0.1m at $t = 4.5$ sec. It is also seen that the amplitude of the response increases gradually, in a manner opposite to a free vibration response. In fact, Koo and Der Kiureghian (2003) have shown that the design point excitation can be obtained by investigating the mirror image of the free vibration response of a system released from a displacement equal to the performance function threshold. The corresponding design point excitation is shown in Figure 5.27. It is observed that high-frequency content is visibly present in the first part of the ground motion due to the selected filter properties and modulating functions.

While the case of linear material models and the selected stochastic input leads to a linear reliability problem, the nonlinear case is more challenging. Computation of mean out-crossing rates of nonlinear response to stochastic input for complex multi-degree-of-freedom structures are both costly and often require special strategies to determine the design point. Such strategies may involve

gradual increase of the threshold, each time using the previous design point as the start point, or using the design point from a point close in time as the start point for the search for the design point. Other interesting approaches, such as the above mentioned use of the mirror image of the free vibration response, are also topics for future research. While further results will not be presented for the I-880 highway bridge model, the subsequent Section 5.5 discusses the problem of finding the design point for these problems in more detail.

5.4.8 Dynamic Reliability Analysis with Damage Index

In this section an alternative to the mean out-crossing rate analysis approach for dynamic problems demonstrated above is presented. Here, the performance function is defined in terms of an accumulated response quantity representing damage. Several such damage indices are found in the literature, for instance that developed by Park and Ang (1985) for reinforce concrete. In this section, however, a simple performance function is prescribed to demonstrate the availability of such analysis options in OpenSees.

As an example, the case of deterministic excitation and random structural properties is considered. Due to the costliness of the dynamic finite element analysis, only the 20 most important random variables, based on a prior FOSM analysis for the performance function presented below, were included. The ground motion recorded at the Gilroy Historic station during the Loma Prieta earthquake in 1989, scaled by a factor of 1.94 (Kunnath *et al.* 2003), is applied to the I-880 bridge structure. Figure 5.28 shows the displacement response for the mean structure together with hysteresis curves for three different material fibers located in the plastic hinge section of column 152, as indicated in Figure 5.29.

Consider the performance function

$$g = 22 \cdot 10^6 \,\mathrm{N/m^2} - E_h \tag{5.10}$$

prescribed in terms of the hysteretic energy E_h of the reinforcing steel in fiber 2 in Figure 5.29. This performance function is suitable for reliability analysis because accumulated response quantities have continuous derivatives. This leads to good convergence behavior in the search for the design point. In this case, a reliability index of $\beta = 2.93$ was obtained with a corresponding probability of failure of $p_f = 1.67 \cdot 10^{-3}$. While in this example a deterministic input was used, it is also possible to use a stochastic input for such analysis.

5.5 MEAN OUT-CROSSING RATE ANALYSIS OF SIMPLE STRUCTURES

Mean out-crossing rate analysis for nonlinear multi-degree-of-freedom structures is computationally costly, as mentioned in Section 5.4.7 above. To demonstrate the analysis capabilities in OpenSees and to further emphasize the need for using smooth material models, two simple structures are considered in this section. First, consider the single-degree-of-freedom (SDOF) structure in Figure 5.30 with $m_1 = 10,000$kg and $k_1 = 400$kN/m. In OpenSees, this structure is modeled by a single truss element with unit length, unit cross-sectional area and a uniaxial material model. The truss is clamped at one end and has a mass of $m_1 = 10,000$kg attached to the other end, which is free to displace in the axial direction. Three material models are considered: namely, a linear model, a bi-linear model, and a smoothed bi-linear model as developed in Section 2.8. The following material parameters are selected: $E = 400\,\text{kN/m}^2$, $\sigma_y = 20\,\text{kN/m}^2$, $b = 0.1$, $\gamma = 0.7$ and $\eta = 3.0$. Structural damping is specified as 1/5 of the initial stiffness, leading to a damping ratio of ζ =6.3%.

First, it is of interest is to employ the visualization options in OpenSees to obtain a visual impression of the nonlinearities that appear in the performance function. For this purpose, consider the case of a base motion applied at the clamped end and represented by two Gaussian pulses, the first occurring at 0.0 sec. and the second occurring at 1.0 sec. Each pulse is filtered through a unit-impulse-response function of a standard linear oscillator with period $T = 1.0$ sec. and damping ratio $\zeta = 0.6$. The maximum amplitude of each filtered pulse is scaled to $1.3g$. Figure 5.31 shows the excitation and the response at the point $x_1 = x_2 = 1.0$ for this example. For reference, the linear response is also shown in Figure 5.31.

The performance function $g = 0.3\text{m} - u_1$ is first considered. Figure 5.32 shows the values of the performance function for a selected range of values of the two random variables for the case of the bi-linear material. The zero-plane is also shown. It is observed that the performance function exhibits kinks. The limit-state line characterized by $g = 0$ is the intersection between the limit-state function and the zero plane. Figure 5.33 shows this line. Kinks in the limit-state line are clearly present. This indicates that the gradient of the performance function has discontinuities. This violates the assumption of continuous gradients in reliability analysis and may lead to nonconvergence in the search for the design point. Figure 5.34 shows the result of replacing the bi-linear material with its smoothed counterpart. It is observed that the limit-state line with the smooth material model is smooth. The fact that the kinks in the limit-state are removed by using smooth material models indicates that the convergence problems experienced with the use of bi-linear material models can be overcome by use of smoothed material models.

Next, consider a train of Gaussian pulses with $\Delta t = 0.025$ sec. filtered through the same filter, applied as ground motion for the same one-degree-of-freedom structures with the smoothed material model. A trapezoidal modulating function is applied where the four time instants defining the trapezoid are 0.0 sec., 3.0 sec., 6.0 sec. and 9.0 sec. The target maximum standard deviation is $0.2g$. The finite element analysis is performed with $\Delta t = 0.01$ and the quantity δt in Eq. (3.69) is selected as $\delta t = \frac{0.025}{20}$. Figure 5.35 shows the mean up-crossing results obtained for the performance function $g = 0.2\text{m} - u_1$. For comparison, results obtained with the linear material model are also shown. It is observed that the mean up-crossing rate is highest in the region between 3.0 sec. and 6.0 sec. This is the region where the input excitation is strong. It is also seen that, as expected, the up-crossing rate of the nonlinear structure is higher than that of the linear structure. An upper bound to the probability of exceeding the threshold 0.2m can be obtained by integrating the mean up-crossing rate along the time axis.

Next, consider the two-degree-of-freedom (2-DOF) structure in Figure 5.36. This structure is modeled by two truss elements with unit area and unit length connected to form a series system. The masses are $m_1 = 10,000\text{kg}$ and $m_2 = 5,000\text{kg}$. The material parameters for the truss member attached to the clamped end, representing the lower story, are: $E = 600\,\text{kN/m}^2$, $\sigma_y = 30\,\text{kN/m}^2$, $b = 0.1$, $\gamma = 0.7$ and $\eta = 3.0$. The material parameters for the truss member attached to the free end, representing the upper story, are: $E = 300\,\text{kN/m}^2$, $\sigma_y = 12\,\text{kN/m}^2$, $b = 0.1$, $\gamma = 0.7$ and $\eta = 3.0$. Structural damping is specified as Rayleigh damping with the damping matrix being 1/5 of the initial stiffness matrix. The same pulse train as for the SDOF model is applied and the following two performance functions involving interstory drifts are considered:

$$g_1 = 0.15\text{m} - u_1 \tag{5.11}$$

$$g_2 = 0.15\text{m} - (u_2 - u_1) \tag{5.12}$$

Figures 5.37 and 5.38 show the mean up-crossing rate results for g_1 and g_2, respectively. As for the SDOF structure it is observed that the mean out-crossing rates are highest in the region 3.0 sec. to 6.0 sec. Also for the 2-DOF structure one can obtain upper bounds to exceeding the thresholds of 0.15m by integrating the mean up-crossing rate along the time axis.

Above it is demonstrated how the time-variant reliability problem is addressed in OpenSees by computing mean out-crossing rates. These are, however, computationally costly analyses. This is because, in some cases, up to 50 inelastic dynamic finite element analyses are needed to find the design point at each selected time instant. Future research may provide more efficient implementations,

e.g., making use of prior design points, and address the issue by developing alternative techniques for determining the design point or better start points.

Table 5.1: Uncertain model parameters in 3-D truss model.

Parameter	Distribution	Mean	C.o.V.	Corr.
Cross-sectional area A of vertical truss members (20 r.v.)	Lognormal	7450.0mm^2	5.0%	0.0
Cross-sectional area A of all other truss members (60 r.v.)	Lognormal	4350.0mm^2	5.0%	0.0
Young's modulus E for linear, bi-linear and smoothed bi-linear material (80 r.v.)	Lognormal	$210,000\frac{\text{N}}{\text{mm}^2}$	5.0%	0.3
Yield strength σ_y for bi-linear and smoothed bi-linear material (80 r.v.)	Lognormal	300.0N/mm^2	10.0%	0.3
Second stiffness ratio b for bi-linear and smoothed bi-linear material (80 r.v.)	Lognormal	0.01	10.0%	0.3
γ for smoothed bi-linear material (0 r.v.)	N/A	0.7	0.0	N/A
η for smoothed bi-linear material (0 r.v.)	N/A	4.0	0.0	N/A
α of Bouc-Wen material (80 r.v.)	Uniform	0.01	10.0%	0.3
k_o of Bouc-Wen material (80 r.v.)	Lognormal	$210,000$	5.0%	0.3
n of Bouc-Wen material (0 r.v.)	N/A	1.76	0.0	N/A
γ of Bouc-Wen material (0 r.v.)	N/A	100.0	0.0	N/A
β of Bouc-Wen material (0 r.v.)	N/A	$100,000$	0.0	N/A
A_o of Bouc-Wen material (0 r.v.)	N/A	1.0	0.0	N/A
η of Bouc-Wen material (0 r.v.)	N/A	1.0	0.0	N/A
ν of Bouc-Wen material (0 r.v.)	N/A	1.0	0.0	N/A
δ_A of Bouc-Wen material (0 r.v.)	N/A	0.0	0.0	N/A
δ_ν of Bouc-Wen material (0 r.v.)	N/A	0.02	0.0	N/A
δ_η of Bouc-Wen material (0 r.v.)	N/A	0.02	0.0	N/A
Nodal coordinates in x, y and z direction (72 r.v.)	Lognormal	As is	σ =10mm	0.0
Total nodal load in x-direction, applied at top nodes (0 r.v.)	N/A	120.0 kN	0.0	N/A

Table 5.2: 40 most important random variables at the design point for static 3-D truss structure.

Rank	Importance	Identification of random model parameter
1	-0.66567	σ_y of element 4
2	-0.38914	Cross sectional area of element 4
3	-0.27652	σ_y of element 20
4	-0.27018	σ_y of element 2
5	-0.18242	σ_y of element 15
6	-0.17525	σ_y of element 16
7	-0.1746	σ_y of element 8
8	-0.1689	σ_y of element 10
9	-0.15124	Cross sectional area of element 2
10	-0.14613	Cross sectional area of element 20
11	-0.11553	σ_y of element 12
12	-0.10456	Second stiffness ratio b of element 4
13	-0.10418	σ_y of element 6
14	-0.0953	Cross sectional area of element 15
15	-0.09335	Cross sectional area of element 8
16	-0.08993	Cross sectional area of element 10
17	-0.08985	Cross sectional area of element 16
18	-0.06072	Cross sectional area of element 12
19	-0.05568	Young's modulus E of element 4
20	-0.05461	Cross sectional area of element 6
21	-0.03379	x-direction coordinate of node 4
22	-0.02905	Second stiffness ratio b of element 2
23	-0.02891	σ_y of element 18
24	-0.02377	σ_y of element 36
25	-0.02241	σ_y of element 13
26	0.01812	x-direction coordinate of node 2
27	-0.0159	Young's modulus E of element 2
28	-0.01506	Cross sectional area of element 18
29	-0.01443	x-direction coordinate of node 12
30	-0.01443	x-direction coordinate of node 21
31	-0.01436	x-direction coordinate of node 22
32	-0.01434	x-direction coordinate of node 24
33	-0.01365	y-direction coordinate of node 21
34	-0.01356	y-direction coordinate of node 7
35	-0.0131	x-direction coordinate of node 3
36	-0.01265	x-direction coordinate of node 23
37	-0.01244	Second stiffness ratio b of element 20
38	-0.01216	y-direction coordinate of node 22
39	-0.01211	y-direction coordinate of node 23
40	-0.01206	Cross sectional area of element 36

Table 5.3: Uncertainty characterization of FE model parameters of reinforced concrete frame; mean (μ), standard deviation (σ) and correlation (ρ).

	Distribution	μ	σ	ρ
Steel reinforcement material parameters:				
σ_y	Lognormal	413.3 N/mm^2	16.5 N/mm^2	0.3
E	Lognormal	206,640 N/mm^2	8,266 N/mm^2	0.3
Confined concrete material parameters:				
f_c	Lognormal	-35.8 N/mm^2	1.43 N/mm^2	0.3
f_{cu}	Lognormal	-32.37 N/mm^2	1.29 N/mm^2	0.3
ϵ_c	Lognormal	-5.0 10^{-3}	2.0 10^{-4}	0.3
ϵ_{cu}	Lognormal	-0.02	8.0 10^{-4}	0.3
Unconfined concrete material parameters:				
f_c	Lognormal	-27.6 N/mm^2	1.10 N/mm^2	0.3
f_{cu}	N/A	0.0 N/mm^2	N/A	N/A
ϵ_c	Lognormal	-2.0 10^{-3}	8.0 10^{-5}	0.3
ϵ_{cu}	Lognormal	-6.0 10^{-3}	2.4 10^{-4}	0.3
Nodal coordinates (all):				
$\mathbf{x}$	Normal	As is	10.mm	0.0
Lateral loads:				
P_i	Lognormal	1,000 kN	40 kN	0.6

Table 5.4: Reliability index β and coefficient of variation for importance sampling for reinforced concrete frame.

	FORM	Impt.sampl.	Num.sim/C.o.v.
g_1	3.4404	3.432	1000 / 6.22%
g_2	3.2189	-	-
g_3	3.7414	-	-

Table 5.5: Parameter importance rankings for reinforced concrete frame. Superscripts *inn* and *out* denote inner (confined) and outer (unconfined) concrete, respectively. "El" denotes element number. Element and node numbers are provided in Figure 5.8.

	FOSM	FORM		
	$\nabla g(\boldsymbol{\mu})_i\ \sigma_i$	$\boldsymbol{\gamma}$	$\boldsymbol{\delta}$	$\boldsymbol{\eta}$
1	P_1	P_1	P_1	P_1
2	P_2	P_2	P_2	P_2
3	El.3, E	El.3, E	El.3, E	El.3, E
4	El.3, σ_y	El.8, E	El.8, E	El.8, E
5	El.8, E	El.3, σ_y	El.3, σ_y	El.7, E
6	El.7, E	El.7, E	El.7, E	El.3, σ_y
7	Node 4, x_1	Node 4, x_1	Node 4, x_1	El.5, E
8	El.7, f_c^{out}	El.7, f_c^{out}	El.7, f_c^{out}	El.7, f_c^{out}
9	El.8, σ_y	El.3, f_c^{inn}	El.3, f_c^{inn}	El.1, σ_y
10	El.7, σ_y	El.1, σ_y	El.1, σ_y	El.8, f_c^{out}
11	El.8, f_c^{out}	El.8, f_c^{out}	El.8, f_c^{out}	El.10, E
12	Node 7, x_1	El.8, σ_y	El.8, σ_y	El.8, σ_y
13	El.3, f_c^{inn}	Node 7, x_1	Node 7, x_1	El.7, σ_y
14	El.1, E	El.5, E	El.5, E	El.9, E
15	El.5, E	El.7, σ_y	El.7, σ_y	El.3, f_c^{inn}
16	Node 5, x_1	Node 5, x_1	Node 5, x_1	El.3, f_c^{out}
17	El.3, f_c^{out}	Node 4, x_2	Node 4, x_2	El.4, E
18	El.10, E	El.10, E	El.10, E	El.5, f_c^{out}
19	Node 8, x_1	El.5, f_c^{inn}	El.5, f_c^{inn}	El.5, σ_y
20	Node 4, x_2	El.3, ϵ_c^{inn}	El.3, f_c^{out}	El.9, f_c^{out}

Table 5.6: Nodal coordinates for I-880 Testbed bridge model. Unit: meters.

Node number	x-coord.	y-coord.	z-coord.
14998	-20.489	2.400	18.923
14999	-20.381	12.661	18.923
14001	0.0	2.184	1.753
14002	0.0	12.446	1.753
14003	0.0	2.184	18.703
14005	0.0	12.446	18.311
15001	48.463	2.184	1.676
15002	48.463	12.446	1.600
15003	48.463	2.184	17.944
15005	48.463	12.446	17.726
16001	97.231	3.150	1.219
16002	97.231	12.446	1.219
16003	97.231	3.150	16.940
16005	97.231	12.446	16.764
17001	147.520	2.184	0.991
17002	147.520	11.989	0.991
17003	147.520	2.184	15.784
17005	147.520	11.989	15.590
17998	154.230	2.184	15.494
17999	154.230	11.989	15.494

Table 5.7: Uncertain parameters in I-880 Testbed bridge model, part 1.

Parameter	Distribution	Mean	C.o.V.	Correlation
E of reinforcement steel of column hinges (8 r.v.)	Lognormal	$199,948.04\frac{\text{N}}{\text{mm}^2}$	5.0%	0.3
σ_y of reinforcement steel of column hinges (8 r.v.)	Lognormal	$455.05\frac{\text{N}}{\text{mm}^2}$	10.0%	0.3
Second stiffness ratio of reinforcement steel of column hinges (8 r.v.)	Lognormal	0.01	15.0%	0.3
f'_c of core concrete of column hinges (8 r.v.)	-Lognormal	$-46.97\frac{\text{N}}{\text{mm}^2}$	15.0%	0.3
f_{cu} of core concrete of column hinges (8 r.v.)	-Lognormal	$-35.85\frac{\text{N}}{\text{mm}^2}$	15.0%	0.3
ϵ_{c0} of core concrete of column hinges (8 r.v.)	-Lognormal	-0.003	15.0%	0.3
ϵ_{cu} of core concrete of column hinges (8 r.v.)	-Lognormal	-0.02	15.0%	0.3
f'_c of cover concrete of column hinges (8 r.v.)	-Lognormal	$-35.85\frac{\text{N}}{\text{mm}^2}$	15.0%	0.3
ϵ_{c0} of cover concrete of column hinges (8 r.v.)	-Lognormal	-0.002	15.0%	0.3
ϵ_{cu} of cover concrete of column hinges (8 r.v.)	-Lognormal	-0.006	15.0%	0.3
E of linear elastic region of columns (8 r.v.)	Lognormal	$28337.46\frac{\text{N}}{\text{mm}^2}$	5.0%	0.3
A of linear elastic region of columns (8 r.v.)	Lognormal	6.32m^2	5.0%	0.3
I_z of linear elastic region of columns (8 r.v.)	Lognormal	$\frac{2.59\text{m } (2.44\text{m})^3}{12}$	5.0%	0.3
I_y of linear elastic region of columns (8 r.v.)	Lognormal	$\frac{2.44\text{m } (2.59\text{m})^3}{12}$	5.0%	0.3
G of linear elastic region of columns (8 r.v.)	Lognormal	$0.4 \cdot 28337.46\frac{\text{N}}{\text{mm}^2}$	5.0%	0.3
J of linear elastic region of columns (8 r.v.)	Lognormal	2.93m^4	5.0%	0.3

Table 5.8: Uncertain parameters in I-880 Testbed bridge model, part 2.

Parameter	Distribution	Mean	C.o.V.	Correlation
A of linear elastic transverse beam elements (4 r.v.)	Lognormal	6.65 m^2	5.0%	0.3
E of linear elastic transverse beam elements (4 r.v.)	Lognormal	$28337.46\frac{\mathrm{N}}{\mathrm{mm}^2}$	5.0%	0.3
G of linear elastic transverse beam elements (4 r.v.)	Lognormal	$11376.35\frac{\mathrm{N}}{\mathrm{mm}^2}$	5.0%	0.3
J_x of linear elastic transverse beam elements (4 r.v.)	Lognormal	$0.21\mathrm{m}^4$	5.0%	0.3
I_y of linear elastic transverse beam elements (4 r.v.)	Lognormal	$2.17\mathrm{m}^4$	5.0%	0.3
I_z of linear elastic transverse beam elements (4 r.v.)	Lognormal	$2.17\mathrm{m}^4$	5.0%	0.3
A of linear elastic longitudinal beam elements (10 r.v.)	Lognormal	$12.22\mathrm{m}^2$	5.0%	0.3
E of linear elastic longitudinal beam elements (10 r.v.)	Lognormal	$28337.46\frac{\mathrm{N}}{\mathrm{mm}^2}$	5.0%	0.3
G of linear elastic longitudinal beam elements (10 r.v.)	Lognormal	$11376.35\frac{\mathrm{N}}{\mathrm{mm}^2}$	5.0%	0.3
J_x of linear elastic longitudinal beam elements (10 r.v.)	Lognormal	$0.146\mathrm{m}^4$	5.0%	0.3
I_y of linear elastic longitudinal beam elements (10 r.v.)	Lognormal	$7.49\mathrm{m}^4$	5.0%	0.3
I_z of linear elastic longitudinal beam elements (10 r.v.)	Lognormal	$7.49\mathrm{m}^4$	5.0%	0.3

Table 5.9: Uncertain parameters in I-880 Testbed bridge model, part 3.

Parameter	Distribution	Mean	C.o.V.	Correlation
Stiffness of x-direction soil spring under elements 141 and 142 (1 r.v.)	Lognormal	$1.030 \cdot 10^6$ kN/m	10.0%	N/A
Stiffness of y-direction soil spring under elements 141 and 142 (1 r.v.)	Lognormal	$8.965 \cdot 10^5$ kN/m	10.0%	N/A
Stiffness of z-direction soil spring under elements 141 and 142 (1 r.v.)	Lognormal	$2.1724 \cdot 10^6$ kN/m	10.0%	N/A
Stiffness of x-rotation soil spring under elements 141 and 142 (1 r.v.)	Lognormal	$5.4692 \cdot 10^7 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A
Stiffness of y-rotation soil spring under elements 141 and 142 (1 r.v.)	Lognormal	$5.6403 \cdot 10^7 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A
Stiffness of z-rotation soil spring under elements 141 and 142 (1 r.v.)	Lognormal	$4.8223 \cdot 10^7 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A
Stiffness of x-direction soil spring under elements 151 and 152 (1 r.v.)	Lognormal	$1.064 \cdot 10^5$ kN/m	10.0%	N/A
Stiffness of y-direction soil spring under elements 151 and 152 (1 r.v.)	Lognormal	$1.065 \cdot 10^5$ kN/m	10.0%	N/A
Stiffness of z-direction soil spring under elements 151 and 152 (1 r.v.)	Lognormal	$1.1623 \cdot 10^5$ kN/m	10.0%	N/A
Stiffness of x-rotation soil spring under elements 151 and 152 (1 r.v.)	Lognormal	$1.420 \cdot 10^7 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A
Stiffness of y-rotation soil spring under elements 151 and 152 (1 r.v.)	Lognormal	$1.4232 \cdot 10^7 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A
Stiffness of z-rotation soil spring under elements 151 and 152 (1 r.v.)	Lognormal	$1.004 \cdot 10^7 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A

Table 5.10: Uncertain parameters in I-880 Testbed bridge model, part 4.

Parameter	Distribution	Mean	C.o.V.	Corr.
Stiffness of x-direction soil spring under elements 161 and 162 (1 r.v.)	Lognormal	$1.065 \cdot 10^5$ kN/m	10.0%	N/A
Stiffness of y-direction soil spring under elements 161 and 162 (1 r.v.)	Lognormal	$1.066 \cdot 10^5$ kN/m	10.0%	N/A
Stiffness of z-direction soil spring under elements 161 and 162 (1 r.v.)	Lognormal	$1.1728 \cdot 10^5$ kN/m	10.0%	N/A
Stiffness of x-rotation soil spring under elements 161 and 162 (1 r.v.)	Lognormal	$1.5937 \cdot 10^7 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A
Stiffness of y-rotation soil spring under elements 161 and 162 (1 r.v.)	Lognormal	$1.595 \cdot 10^7 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A
Stiffness of z-rotation soil spring under elements 161 and 162 (1 r.v.)	Lognormal	$5.0049 \cdot 10^6 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A
Stiffness of x-direction soil spring under elements 171 and 172 (1 r.v.)	Lognormal	$1.2692 \cdot 10^5$ kN/m	10.0%	N/A
Stiffness of y-direction soil spring under elements 171 and 172 (1 r.v.)	Lognormal	$1.2698 \cdot 10^5$ kN/m	10.0%	N/A
Stiffness of z-direction soil spring under elements 171 and 172 (1 r.v.)	Lognormal	$1.3988 \cdot 10^5$ kN/m	10.0%	N/A
Stiffness of x-rotation soil spring under elements 171 and 172 (1 r.v.)	Lognormal	$2.3739 \cdot 10^7 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A
Stiffness of y-rotation soil spring under elements 171 and 172 (1 r.v.)	Lognormal	$2.3790 \cdot 10^7 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A
Stiffness of z-rotation soil spring under elements 171 and 172 (1 r.v.)	Lognormal	$4.1651 \cdot 10^6 \frac{\text{kN m}}{\text{rad}}$	10.0%	N/A
Nodal coordinates in x, y and z direction (60 r.v.)	Normal	As is	Stdv= 1.27cm	0.0

Table 5.11: Response statistics results for performance functions g_3 and g_4.

Response quantity	Mean	Standard deviation
$u_{\,(at\ 20\%\ tangent)}$	0.2525m	0.01677m
$\lambda_{\,(at\ 20\%\ tangent)}$	0.2008	0.009621

Table 5.12: 40 most important random variables in initial region of load-displacement curve of I-880 highway bridge.

Rank	Importance	Identification of random model parameter						
1	-0.6842	element	1502	material	153	E		
2	0.5816	element	1501	material	153	E		
3	-0.3098	element	1602	material	163	E		
4	0.2463	element	1601	material	163	E		
5	-0.0760	element	1502	material	152	E		
6	-0.0722	element	1501	material	152	E		
7	-0.0688	element	1504	material	154	E		
8	-0.0631	element	1503	material	154	E		
9	-0.0448	element	1402	material	143	E		
10	-0.0386	element	1702	material	173	E		
11	0.0335	element	1701	material	173	E		
12	-0.0319	element	1602	material	162	E		
13	0.0303	element	1401	material	143	E		
14	-0.0298	element	1604	material	164	E		
15	-0.0293	element	1601	material	162	E		
16	-0.0282	element	43	E				
17	-0.0267	element	1603	material	164	E		
18	0.0218	element	152	section	150	material	2	fc
19	-0.0208	element	152	section	150	material	2	epsco
20	0.0205	node	15003	coord	2			
21	-0.0199	node	15002	coord	2			
22	0.0197	element	151	section	150	material	2	fc
23	-0.0188	element	151	section	150	material	2	epsco
24	-0.0183	element	143	E				
25	0.0179	node	15005	coord	2			
26	0.0170	element	142	section	140	material	2	fc
27	-0.0166	element	42	Iy				
28	-0.0163	element	43	Iy				
29	-0.0160	element	142	section	140	material	2	epsco
30	-0.0144	node	15001	coord	2			
31	-0.0134	element	152	E				
32	-0.0122	element	43	Iz				
33	-0.0109	element	152	Iz				
34	0.0106	element	162	section	160	material	2	fc
35	-0.0104	element	142	E				
36	-0.0100	element	162	section	160	material	2	epsco
37	-0.0097	element	151	Iz				
38	0.0097	element	42	Iz				
39	-0.0095	element	143	Iz				
40	0.0091	node	15005	coord	3			

Table 5.13: 40 least important random variables in initial region of load-displacement curve of I-880 highway bridge.

Rank	Importance	Identification of random model parameter						
281	≈ 0.0	element	161	section	160	material	2	epscu
282	≈ 0.0	element	152	section	150	material	2	epscu
283	≈ 0.0	element	151	section	150	material	2	epscu
284	≈ 0.0	element	142	section	140	material	2	epscu
285	≈ 0.0	element	172	section	170	material	1	fcu
286	≈ 0.0	element	171	section	170	material	1	fcu
287	≈ 0.0	element	162	section	160	material	1	fcu
288	≈ 0.0	element	161	section	160	material	1	fcu
289	≈ 0.0	element	152	section	150	material	1	fcu
290	≈ 0.0	element	151	section	150	material	1	fcu
291	≈ 0.0	element	142	section	140	material	1	fcu
292	≈ 0.0	element	141	section	140	material	1	fcu
293	≈ 0.0	element	172	section	170	material	1	epscu
294	≈ 0.0	element	171	section	170	material	1	epscu
295	≈ 0.0	element	162	section	160	material	1	epscu
296	≈ 0.0	element	161	section	160	material	1	epscu
297	≈ 0.0	element	152	section	150	material	1	epscu
298	≈ 0.0	element	151	section	150	material	1	epscu
299	≈ 0.0	element	142	section	140	material	1	epscu
300	≈ 0.0	element	141	section	140	material	1	epscu
301	≈ 0.0	node	17999	coord	2			
302	≈ 0.0	node	17998	coord	2			
303	≈ 0.0	node	17998	coord	1			
304	≈ 0.0	node	14999	coord	3			
305	≈ 0.0	node	14998	coord	3			
306	≈ 0.0	node	14998	coord	2			
307	≈ 0.0	element	48	Iy				
308	≈ 0.0	element	40	Iy				
309	≈ 0.0	element	40	G				
310	≈ 0.0	element	48	A				
311	≈ 0.0	node	14999	coord	2			
312	≈ 0.0	node	14999	coord	1			
313	≈ 0.0	element	49	Iz				
314	≈ 0.0	element	41	Iy				
315	≈ 0.0	element	49	Jx				
316	≈ 0.0	element	41	Jx				
317	≈ 0.0	element	40	Jx				
318	≈ 0.0	element	41	E				
319	≈ 0.0	element	40	E				
320	≈ 0.0	element	49	A				

Table 5.14: 40 most important random variables in yielding region of load-displacement curve of I-880 highway bridge.

Rank	Importance	Identification of random model parameter						
1	-0.6280	element	141	section	140	material	3	sigmaY
2	-0.5633	element	142	section	140	material	3	sigmaY
3	-0.2808	element	151	section	150	material	3	sigmaY
4	-0.2268	element	1502	material	153	E		
5	0.1711	element	142	section	140	material	2	fc
6	-0.1611	element	152	section	150	material	3	sigmaY
7	0.1428	element	142	section	140	material	2	epscu
8	-0.1360	element	1602	material	163	E		
9	0.1234	element	142	section	140	material	1	fc
10	-0.1188	element	161	section	160	material	3	sigmaY
11	0.0729	element	152	section	150	material	2	fc
12	-0.0656	element	162	section	160	material	3	sigmaY
13	0.0584	element	141	section	140	material	2	fc
14	-0.0541	element	1502	material	152	E		
15	-0.0489	element	141	section	140	material	3	b
16	-0.0451	element	142	section	140	material	1	epsco
17	0.0414	element	142	section	140	material	2	epsco
18	0.0361	element	162	section	160	material	2	fc
19	-0.0324	element	142	section	140	material	3	b
20	0.0312	element	152	section	150	material	1	fc
21	0.0282	element	162	section	160	material	2	epscu
22	-0.0266	element	141	section	140	material	3	E
23	-0.0254	node	14002	coord	2			
24	-0.0252	element	1602	material	162	E		
25	0.0247	node	14005	coord	2			
26	0.0246	element	152	section	150	material	2	epscu
27	0.0244	element	151	section	150	material	2	fc
28	0.0243	element	162	section	160	material	1	fc
29	-0.0204	element	142	section	140	material	3	E
30	-0.0187	element	151	section	150	material	3	b
31	-0.0182	element	152	section	150	material	1	epsco
32	-0.0160	node	15002	coord	2			
33	-0.0159	element	152	section	150	material	3	E
34	-0.0142	element	151	section	150	material	3	E
35	-0.0128	element	1702	material	173	E		
36	-0.0118	element	153	Iz				
37	0.0106	node	15005	coord	2			
38	-0.0106	element	153	E				
39	-0.0106	element	171	section	170	material	3	sigmaY
40	0.0104	element	1501	material	153	E		

Table 5.15: 40 least important random variables in yielding region of load-displacement curve of I-880 highway bridge.

Rank	Importance	Identification of random model parameter						
281	≈ 0.0	element	48	G				
282	≈ 0.0	node	14999	coord	3			
283	≈ 0.0	node	14998	coord	3			
284	≈ 0.0	element	40	Jx				
285	≈ 0.0	element	40	Iz				
286	≈ 0.0	element	49	Jx				
287	≈ 0.0	element	49	G				
288	≈ 0.0	node	14998	coord	1			
289	≈ 0.0	node	17998	coord	3			
290	≈ 0.0	element	48	Iz				
291	≈ 0.0	element	48	Iy				
292	≈ 0.0	element	41	Jx				
293	≈ 0.0	element	41	G				
294	≈ 0.0	node	17998	coord	2			
295	≈ 0.0	element	48	A				
296	≈ 0.0	node	14999	coord	2			
297	≈ 0.0	element	172	section	170	material	1	fcu
298	≈ 0.0	element	171	section	170	material	1	fcu
299	≈ 0.0	element	162	section	160	material	1	fcu
300	≈ 0.0	element	161	section	160	material	1	fcu
301	≈ 0.0	element	152	section	150	material	1	fcu
302	≈ 0.0	element	151	section	150	material	1	fcu
303	≈ 0.0	element	142	section	140	material	1	fcu
304	≈ 0.0	element	141	section	140	material	1	fcu
305	≈ 0.0	element	172	section	170	material	1	epscu
306	≈ 0.0	element	171	section	170	material	1	epscu
307	≈ 0.0	element	162	section	160	material	1	epscu
308	≈ 0.0	element	161	section	160	material	1	epscu
309	≈ 0.0	element	152	section	150	material	1	epscu
310	≈ 0.0	element	151	section	150	material	1	epscu
311	≈ 0.0	element	142	section	140	material	1	epscu
312	≈ 0.0	element	49	A				
313	≈ 0.0	element	41	Iz				
314	≈ 0.0	element	48	Jx				
315	≈ 0.0	element	48	E				
316	≈ 0.0	element	40	A				
317	≈ 0.0	element	141	section	140	material	1	epscu
318	≈ 0.0	element	40	G				
319	≈ 0.0	element	41	Iy				
320	≈ 0.0	node	17998	coord	1			

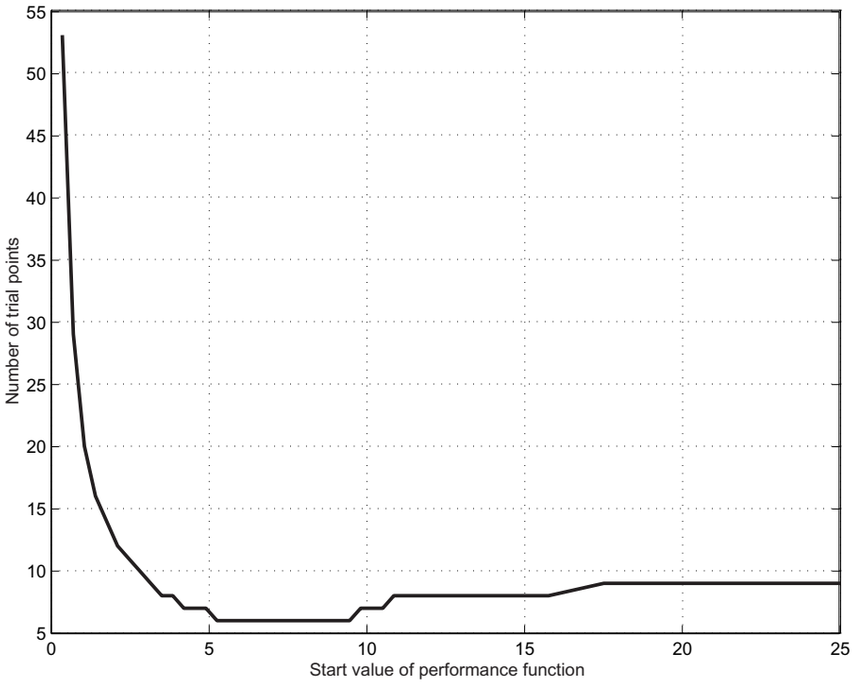

Figure 5.1: Performance of the Polak-He algorithm for different start values of the performance function.

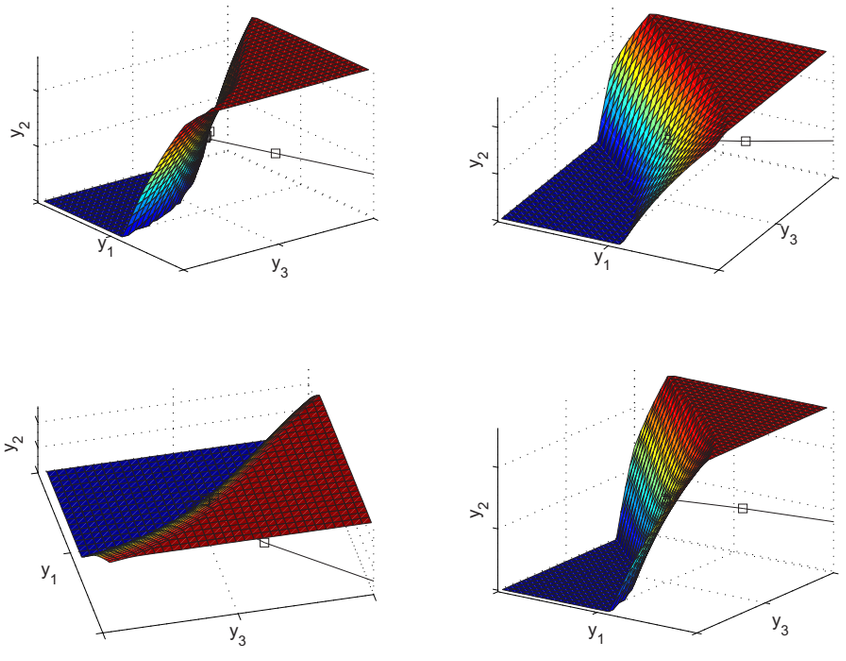

Figure 5.2: Limit-state surface and iHLRF trial steps for the "basic" reliability analysis example.

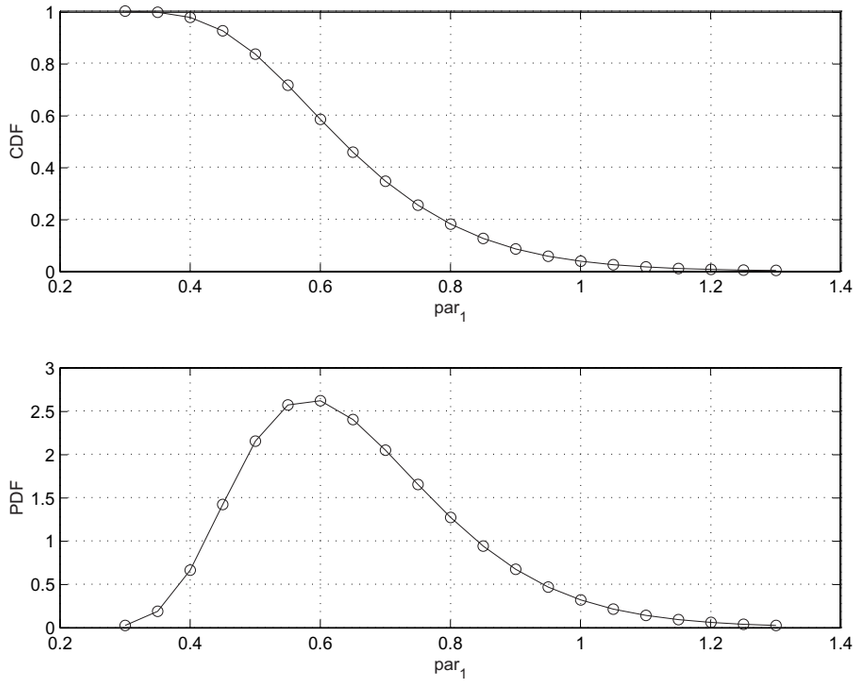

Figure 5.3: Complementary CDF and PDF from parametric reliability analysis of performance function of the basic reliability analysis example.

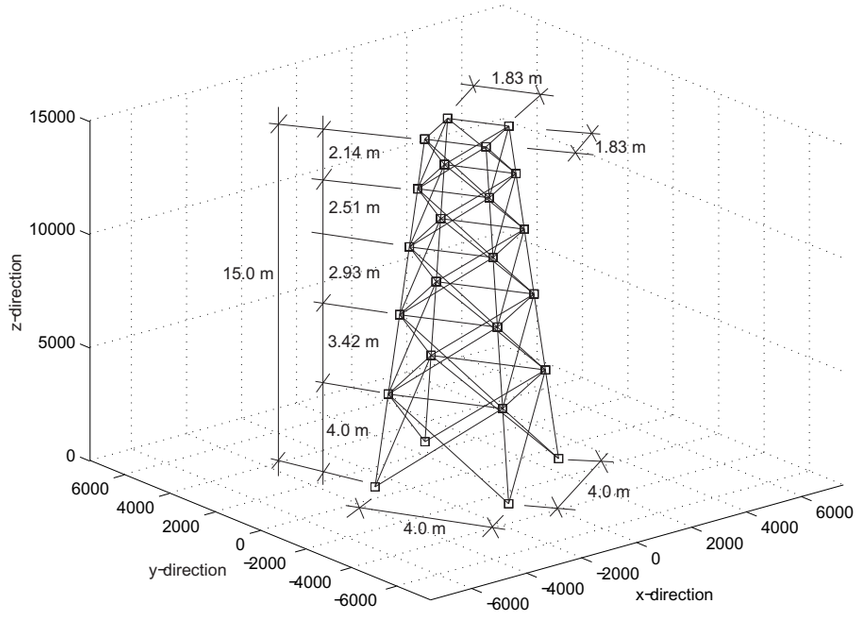

Figure 5.4: 3-D truss example.

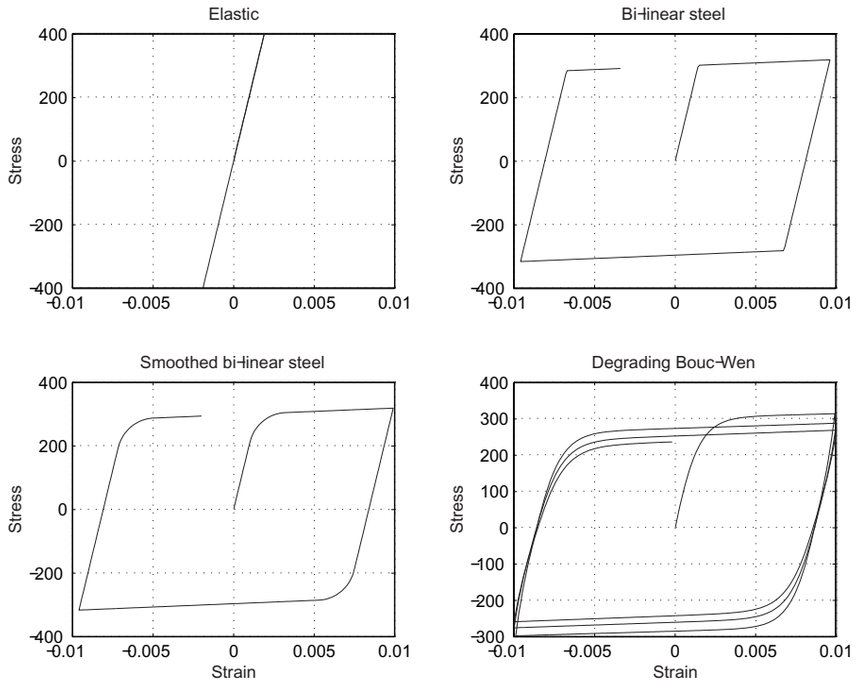

Figure 5.5: Sample stress-strain curves for uniaxial materials employed in 3-D truss example.

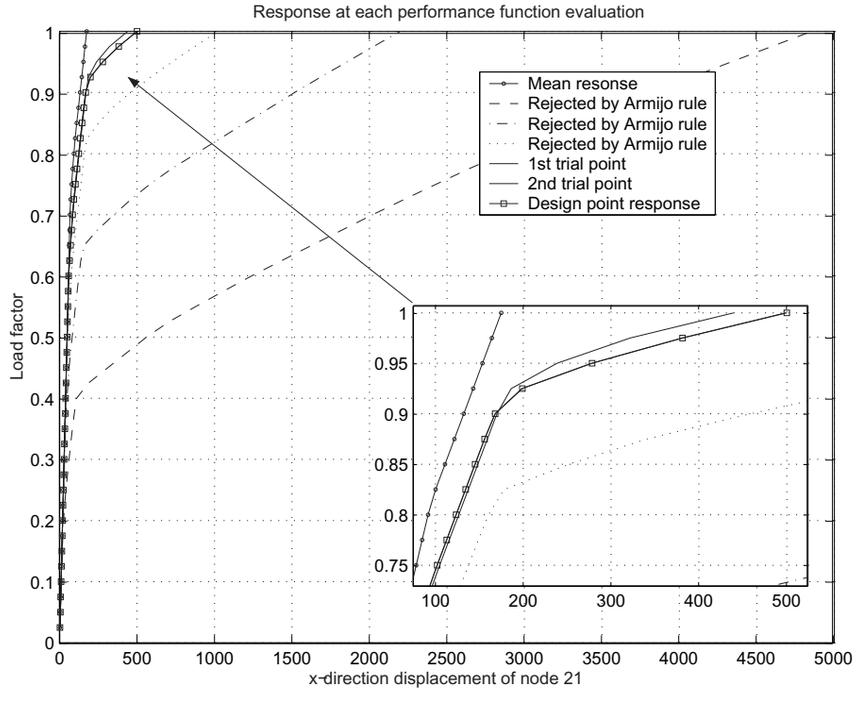

Figure 5.6: Displacement response at top of truss structure during search for the design point.

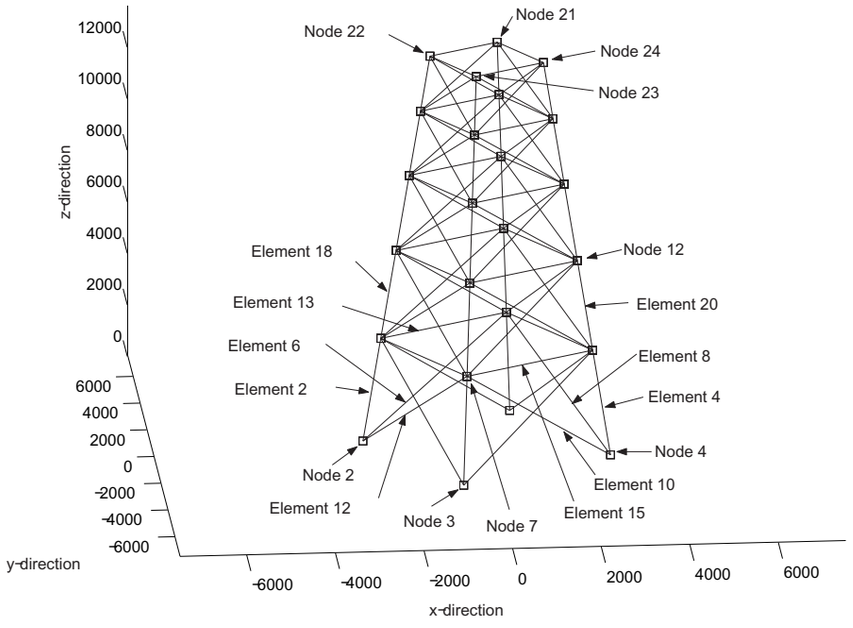

Figure 5.7: Explanation of element and node numbers of 3-D truss structure.

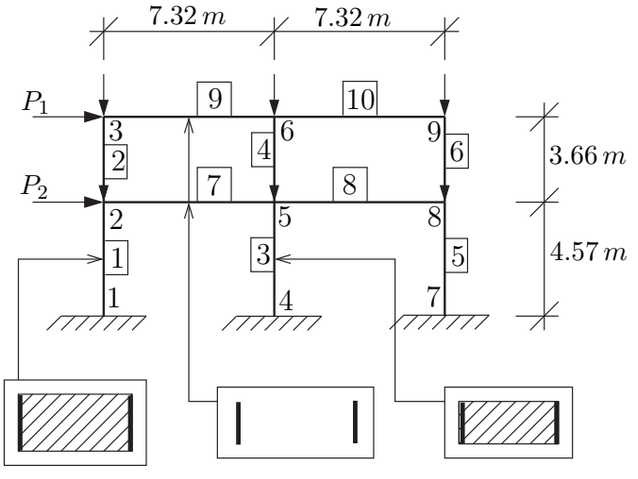

Figure 5.8: Reinforced concrete frame structure with node numbers and element numbers.

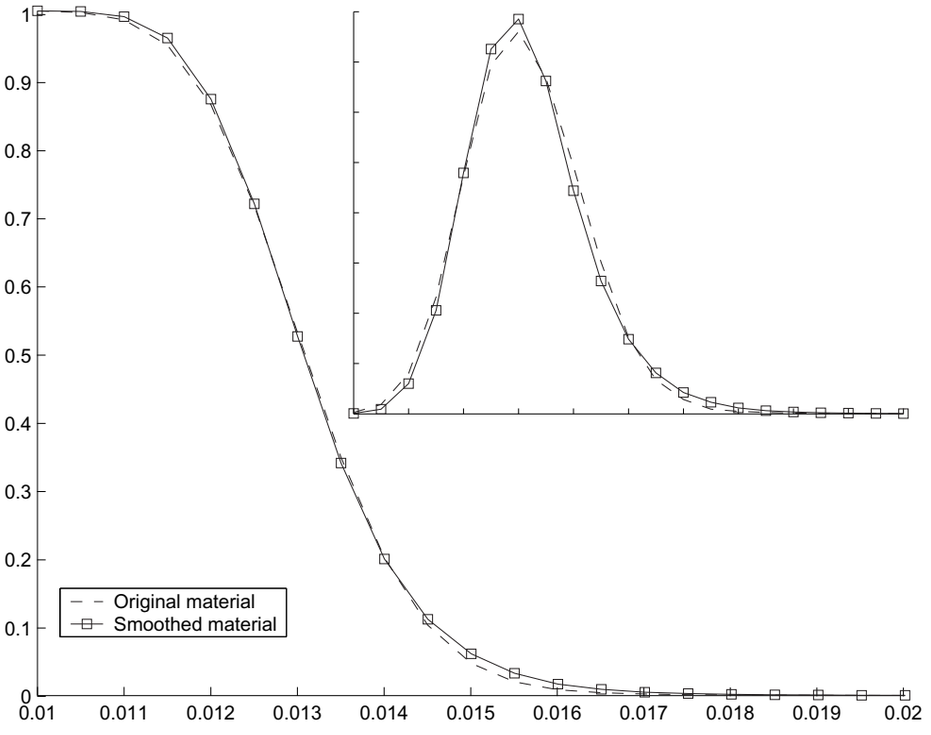

Figure 5.9: CDF and PDF for roof displacement. Threshold is varied from 1% to 2%.

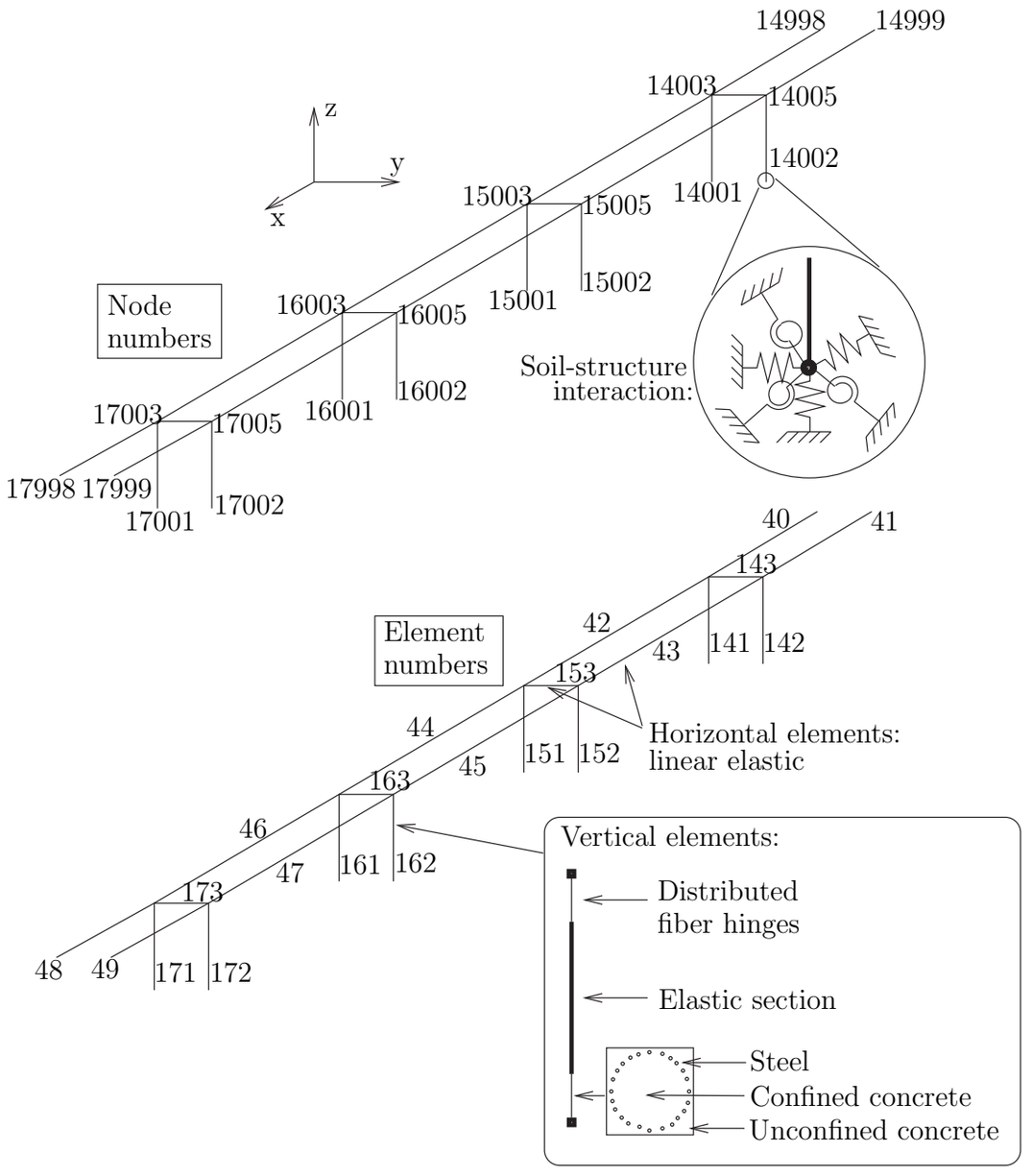

Figure 5.10: Identification of element and node numbers for I-880 highway bridge model.

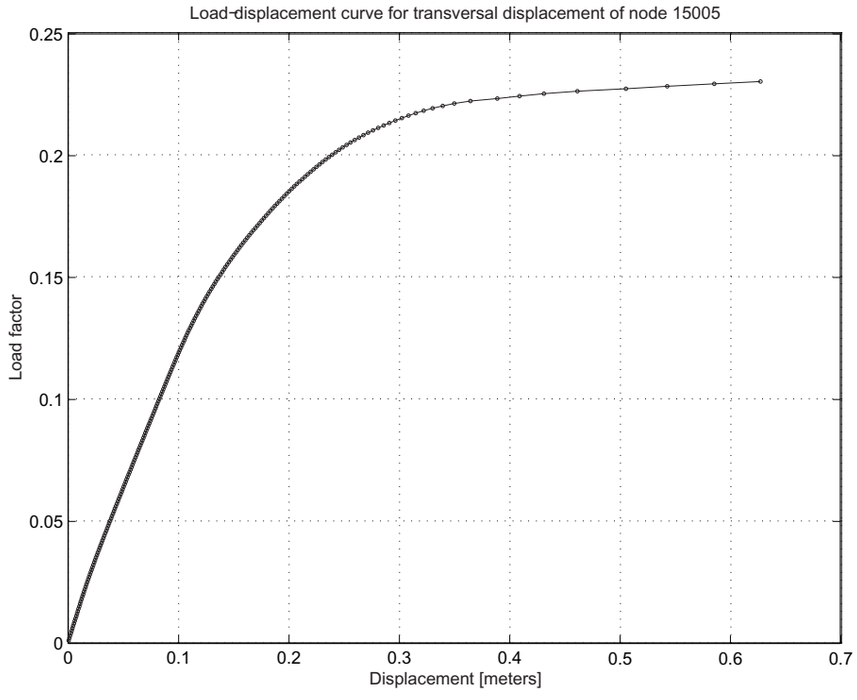

Figure 5.11: Mean point load-displacement curve for transversal (y-direction) displacement of node 15005 due to inelastic static pushover analysis with reference load applied at each bent.

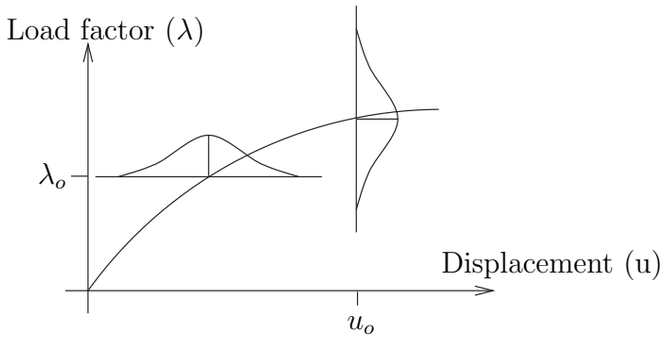

Figure 5.12: Probability distributions of response based on reliability analysis.

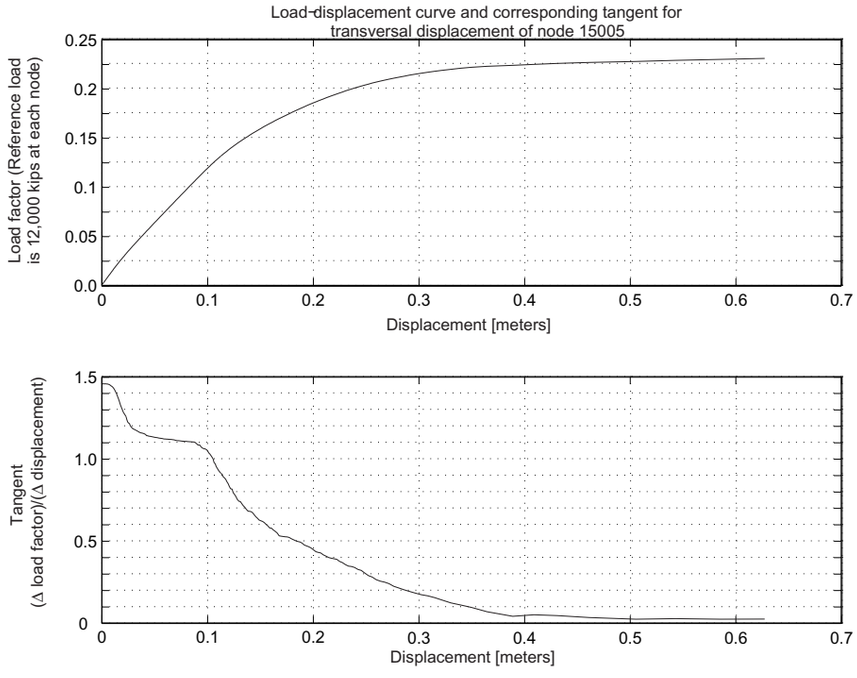

Figure 5.13: Load-displacement curve (top) and corresponding tangent for transversal (y-direction) displacement (bottom) of node 15005 due to inelastic static pushover analysis at the mean point.

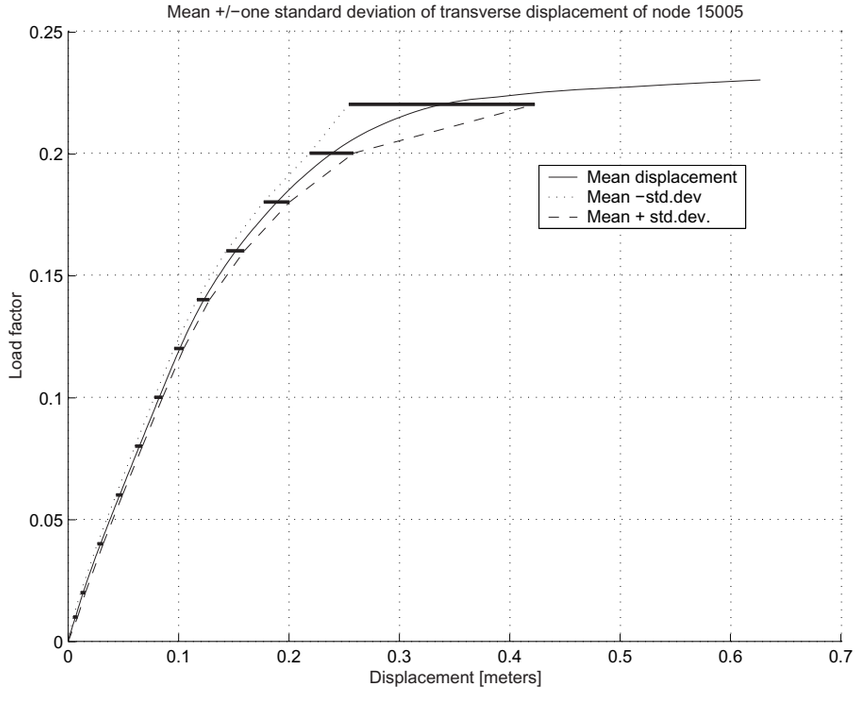

Figure 5.14: Second-moment load-displacement curves from first-order second-moment analysis for response quantity in performance function 1.

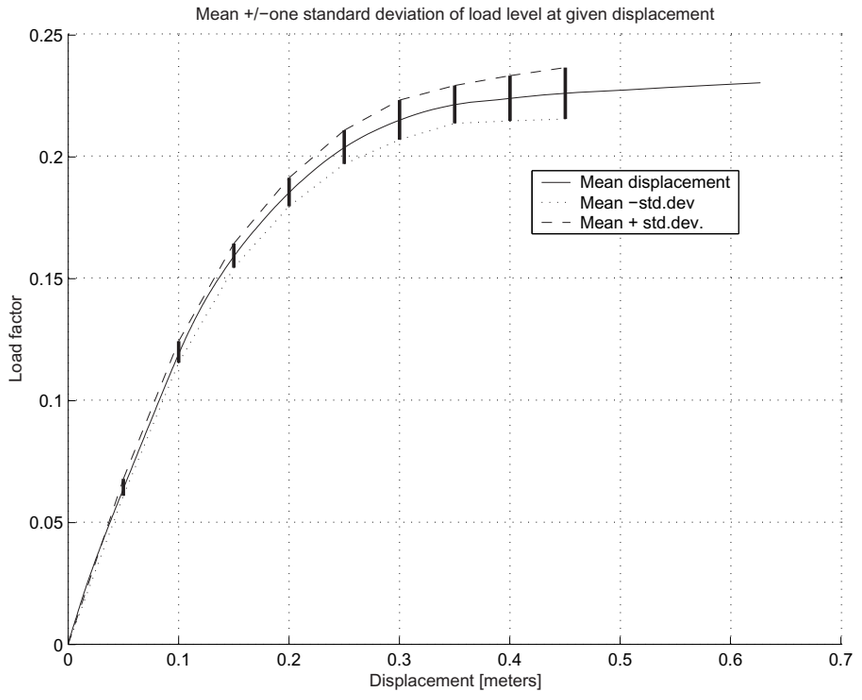

Figure 5.15: Second-moment load-displacement curves from first-order second-moment analysis for response quantity in performance function 2.

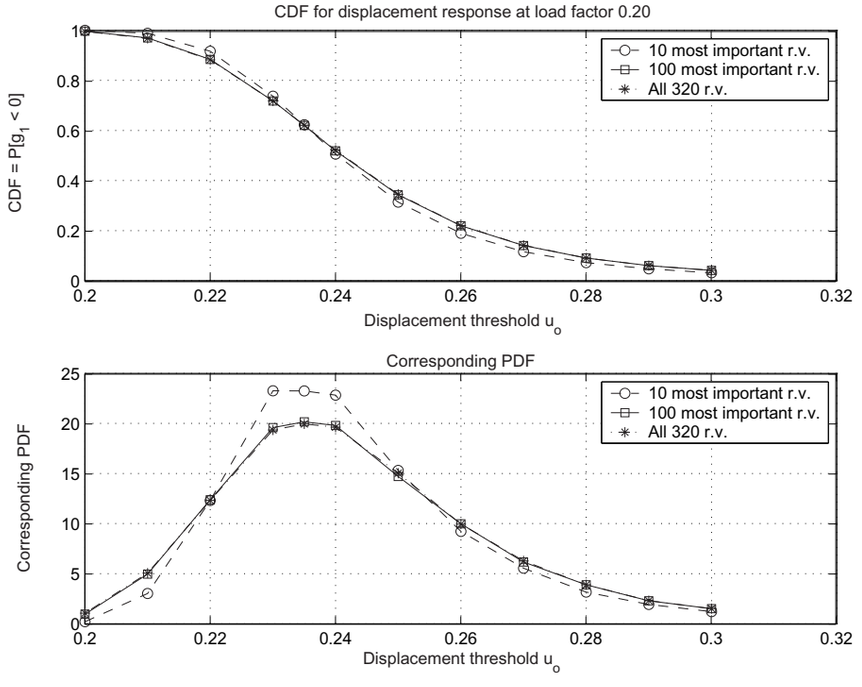

Figure 5.16: Probability distribution for displacement response at load factor 0.20; obtained by a series of FORM reliability analyses of performance function number 1.

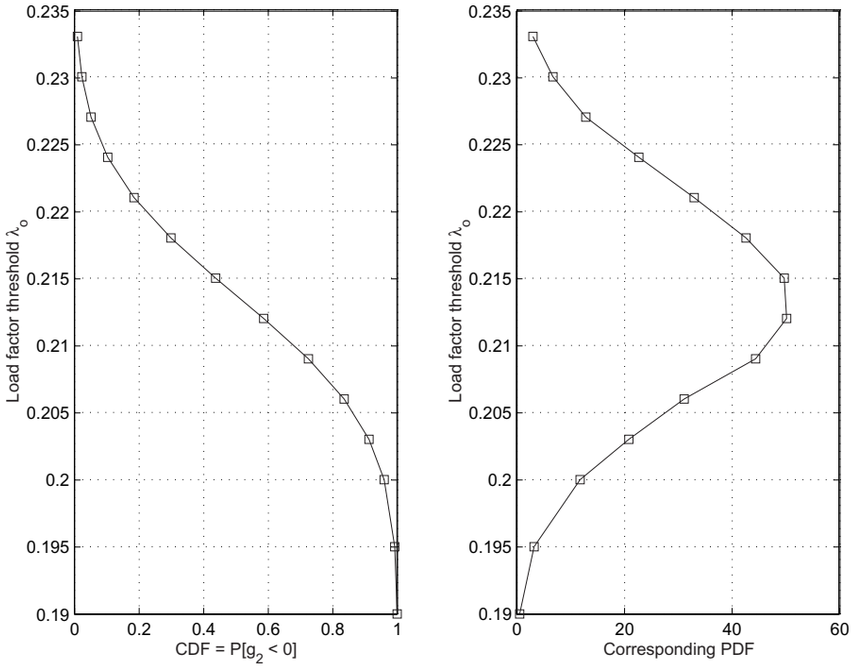

Figure 5.17: Probability distribution for load factor level at displacement 0.3 meters; obtained by a series of FORM reliability analyses of performance function number 2.

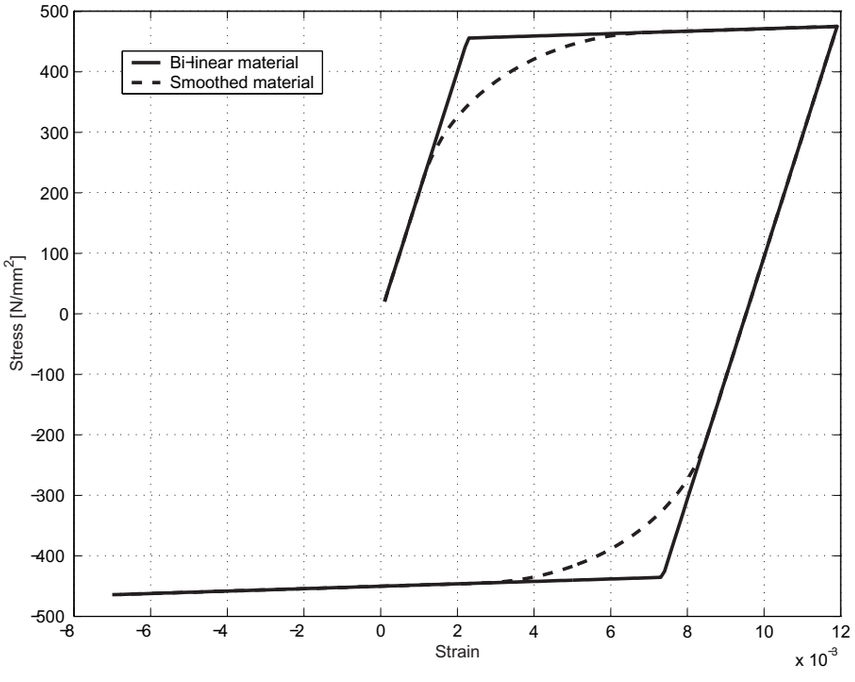

Figure 5.18: Example stress-strain curve for bi-linear and smoothed material model used for reinforcing steel.

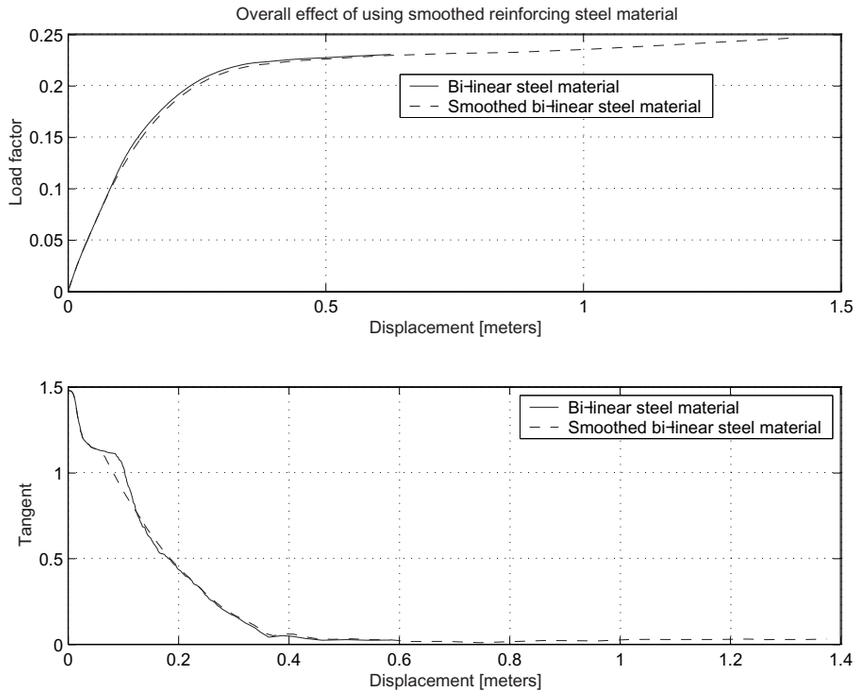

Figure 5.19: Effect on response and tangent of using smoothed material for steel reinforcement.

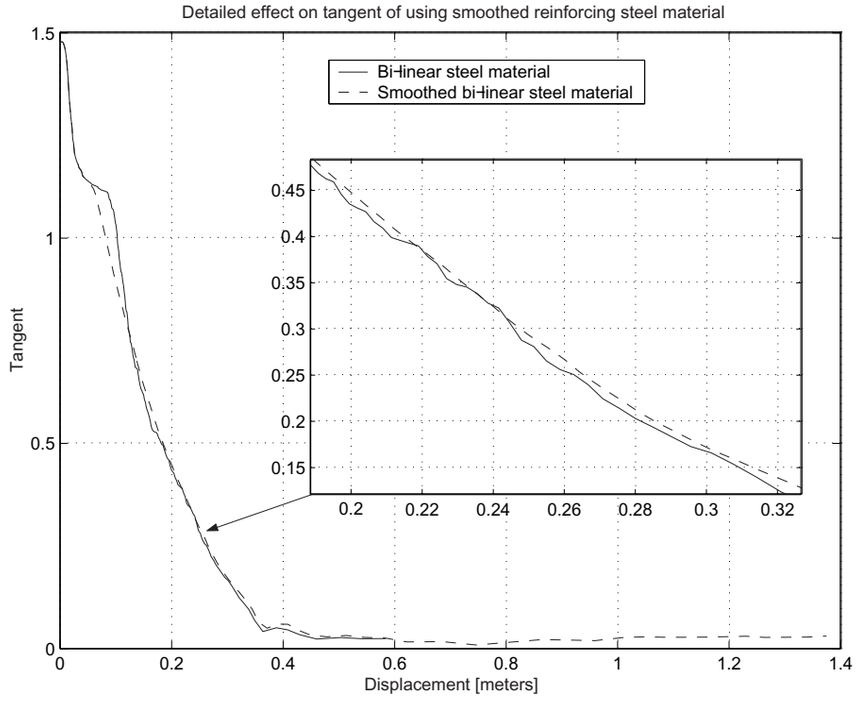

Figure 5.20: Detailed effect on tangent of using smoothed material for steel reinforcement.

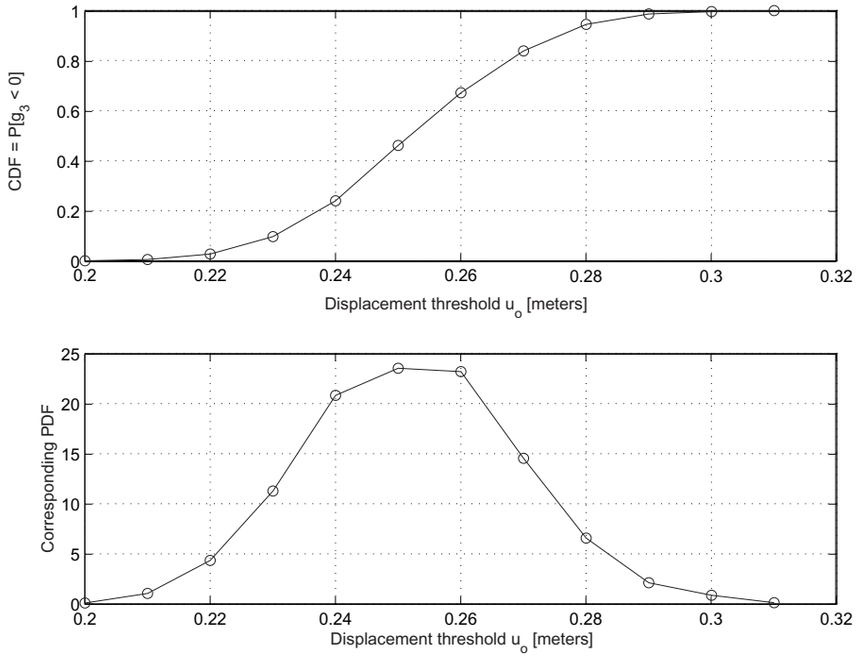

Figure 5.21: Probability distribution for displacement at 20% of elastic tangent; obtained by a series of FORM reliability analyses of performance function number 3.

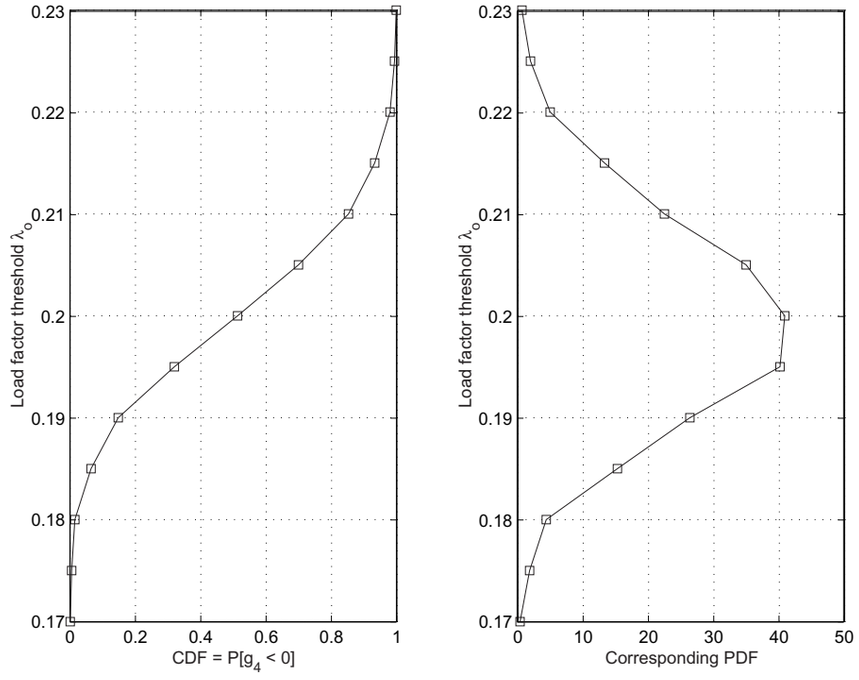

Figure 5.22: Probability distribution for load factor at 20% of elastic tangent; obtained by a series of FORM reliability analyses of performance function number 4.

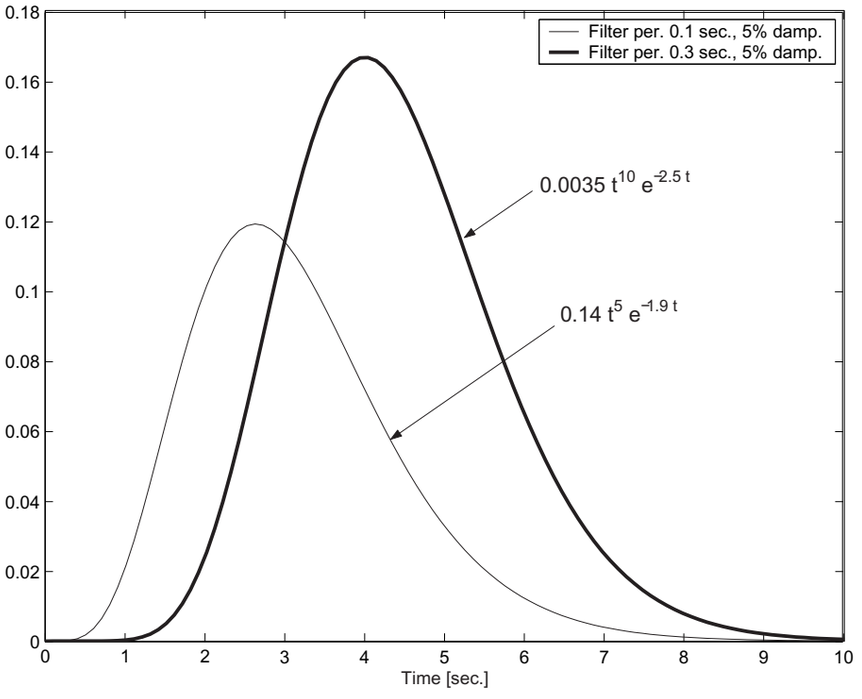

Figure 5.23: Modulating functions and corresponding filter data.

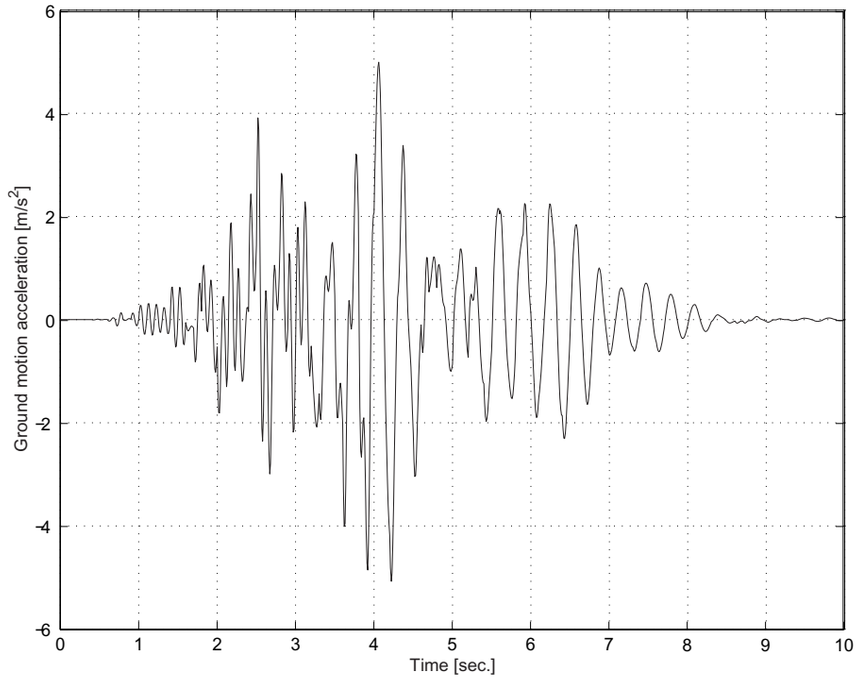

Figure 5.24: Sample stochastic ground motion acceleration. Target standard deviation at 4.0 seconds: $0.2g = 1.96\,m/s^2$.

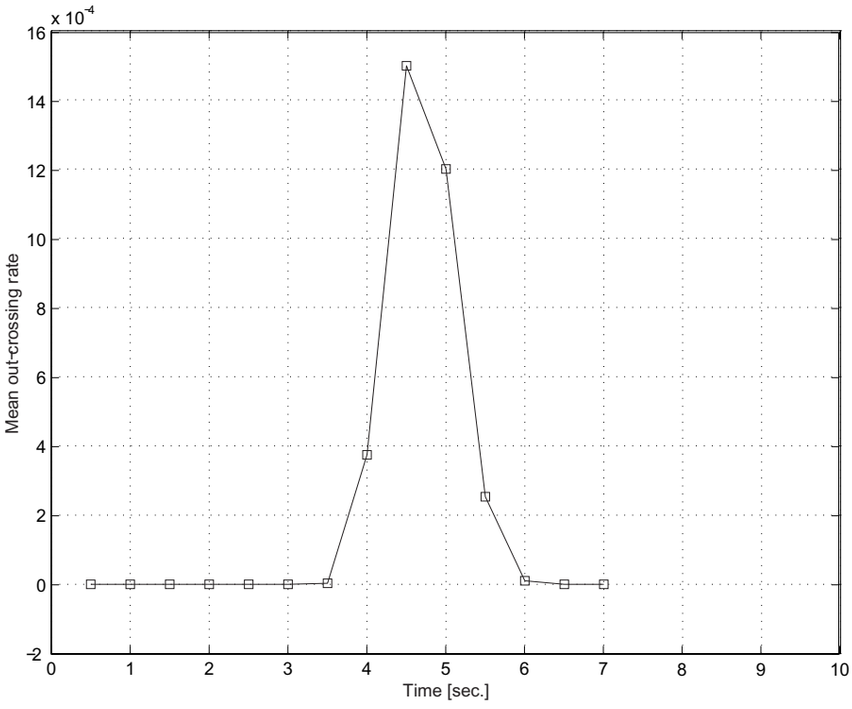

Figure 5.25: Mean out-crossing rate over threshold 0.10m for linear I-880 bridge structure

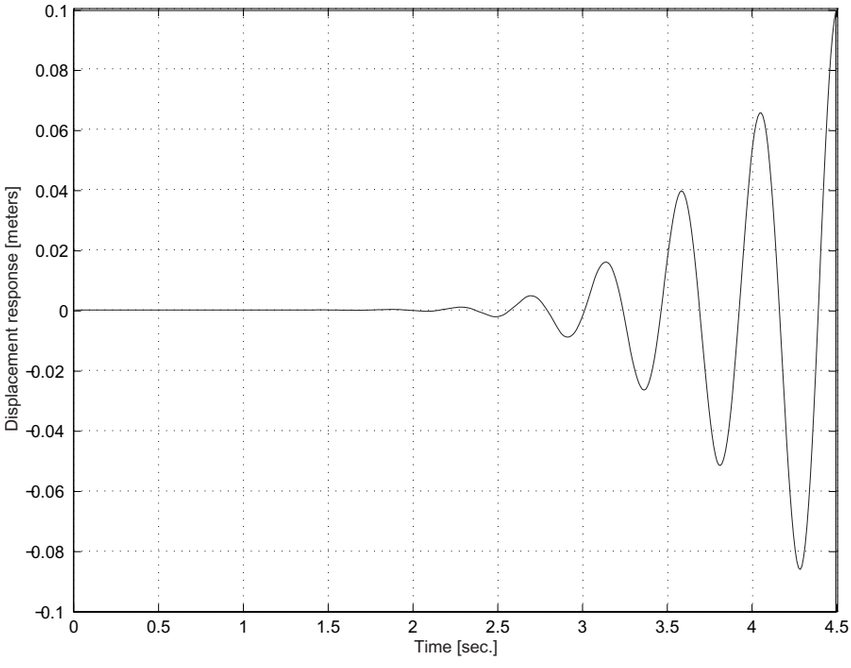

Figure 5.26: Design point response at 4.5 seconds for mean out-crossing rate estimation for linear I-880 bridge.

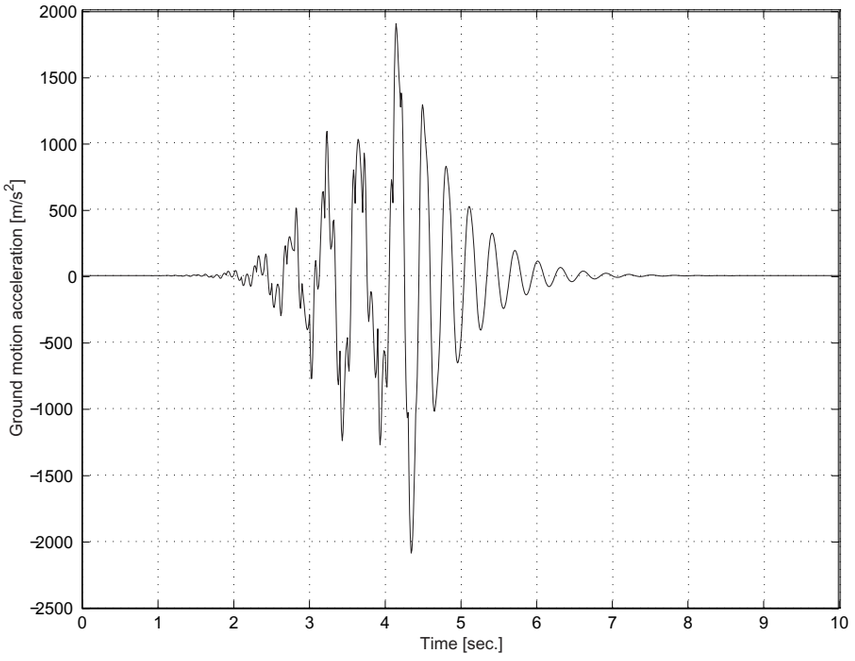

Figure 5.27: Design point excitation at 4.5 seconds for mean out-crossing rate estimation for linear I-880 bridge.

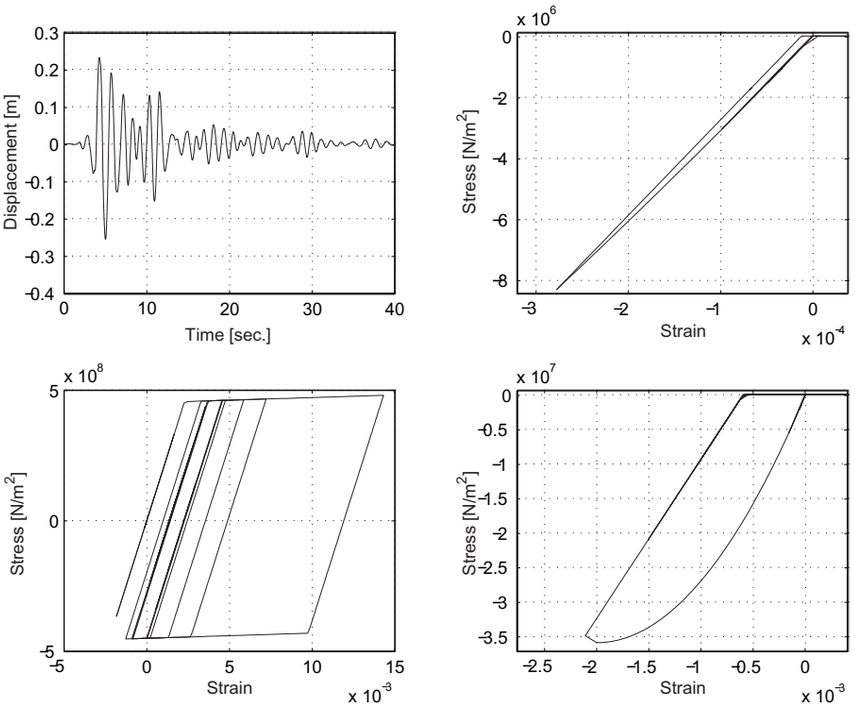

Figure 5.28: Mean point response for applied recorded ground motion. The upper left figure shows the displacement response; the upper right figure shows the stress-strain curve of the concrete fiber 1 in Figure 5.29; lower left figure shows the stress-strain curve of the steel fiber 2 in Figure 5.29; lower right figure shows the stress-strain curve of the concrete fiber 3 in Figure 5.29. Part of the tension strain-history of the concrete fibers is outside the plot area.

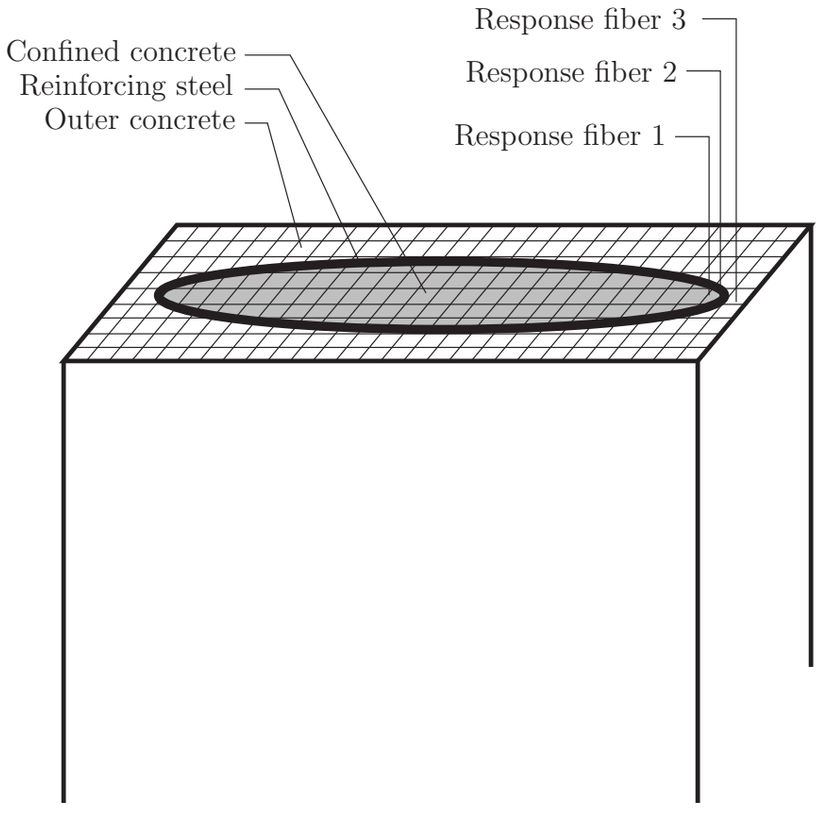

Figure 5.29: Response sampling from fiber-discretized cross section.

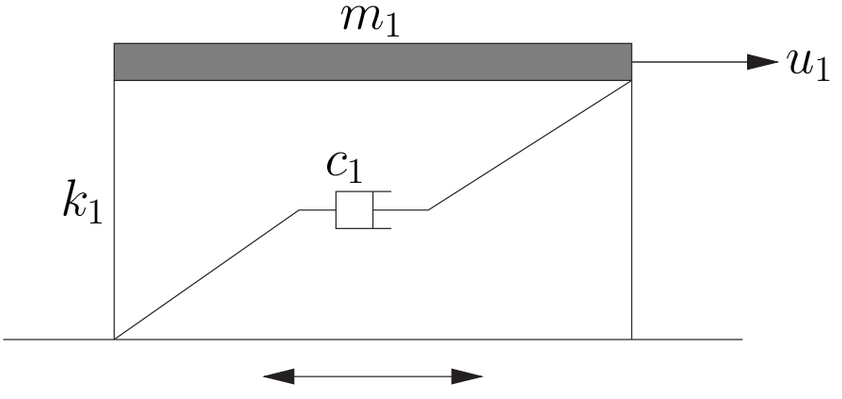

Figure 5.30: One-degree-of-freedom structure.

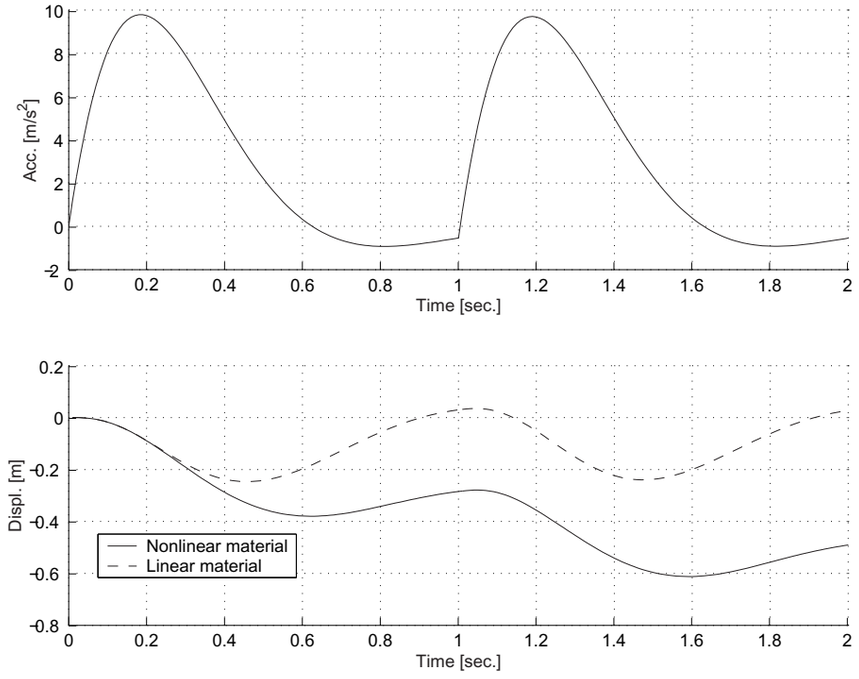

Figure 5.31: Start point excitation (top) and response (bottom) for mean up-crossing analysis of single-degree-of-freedom example.

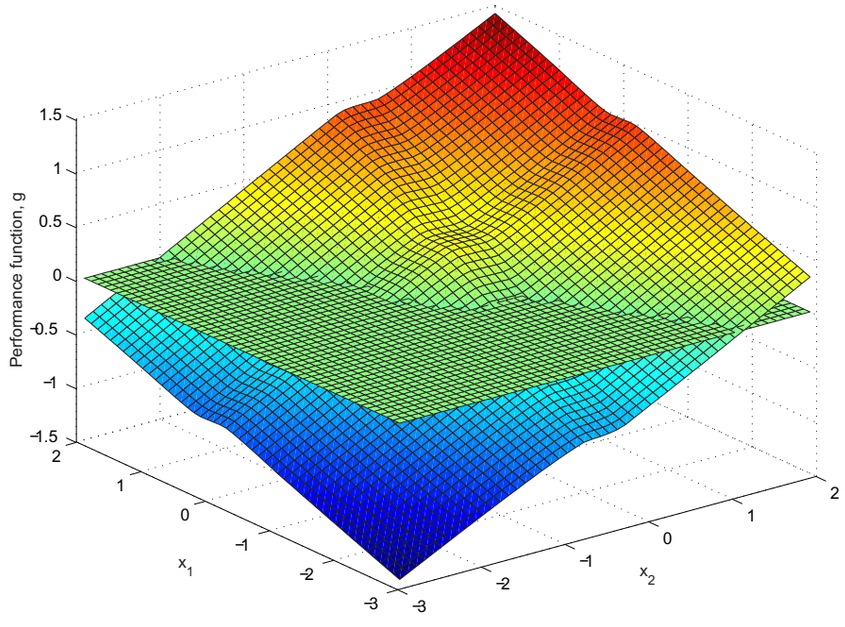

Figure 5.32: Performance function and zero plane for SDOF example with bi-linear material and two random pulses.

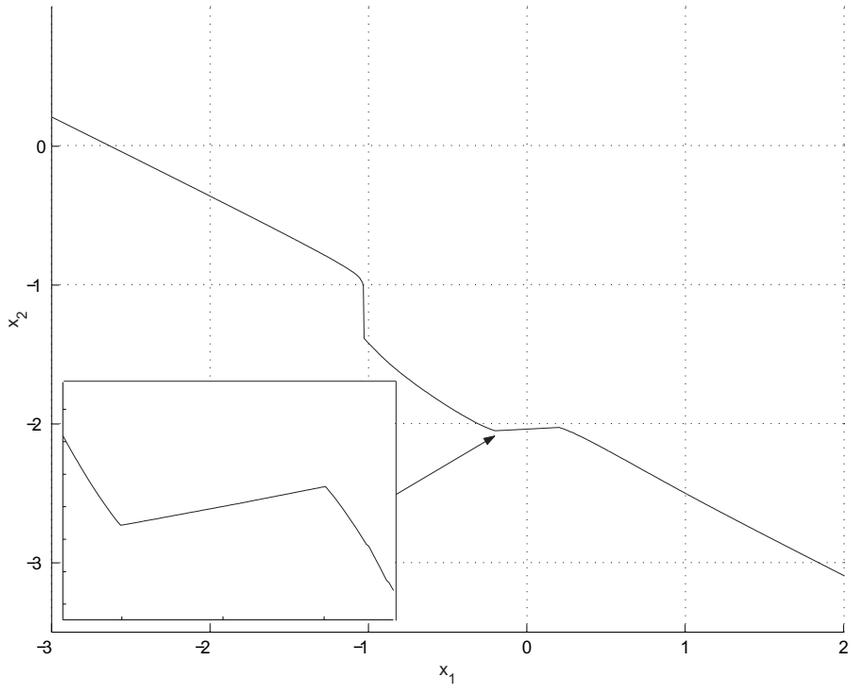

Figure 5.33: Limit-state line for the SDOF example with bi-linear material and two random pulses.

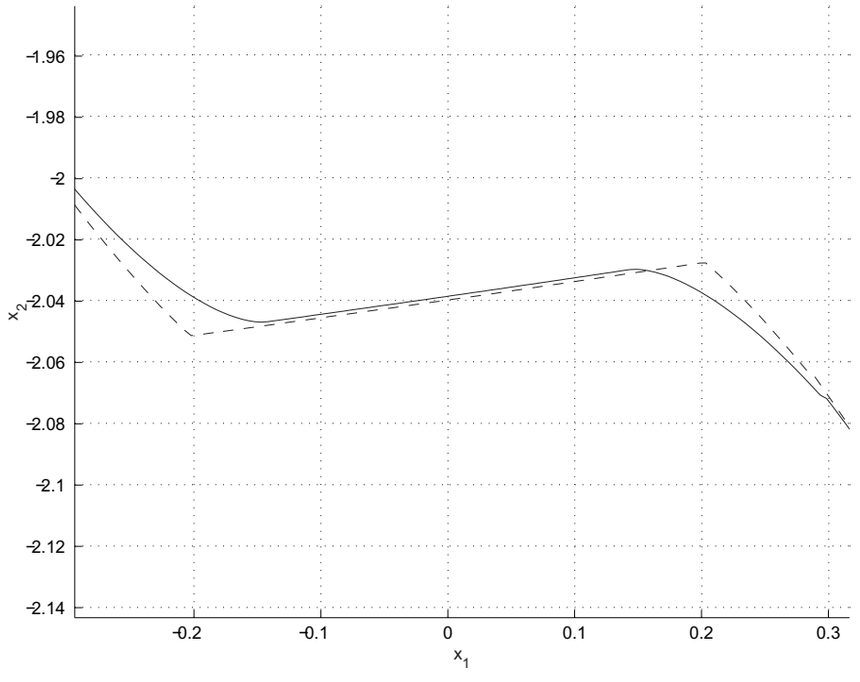

Figure 5.34: Effect on limit-state line of using smoothed material model instead of bi-linear material model.

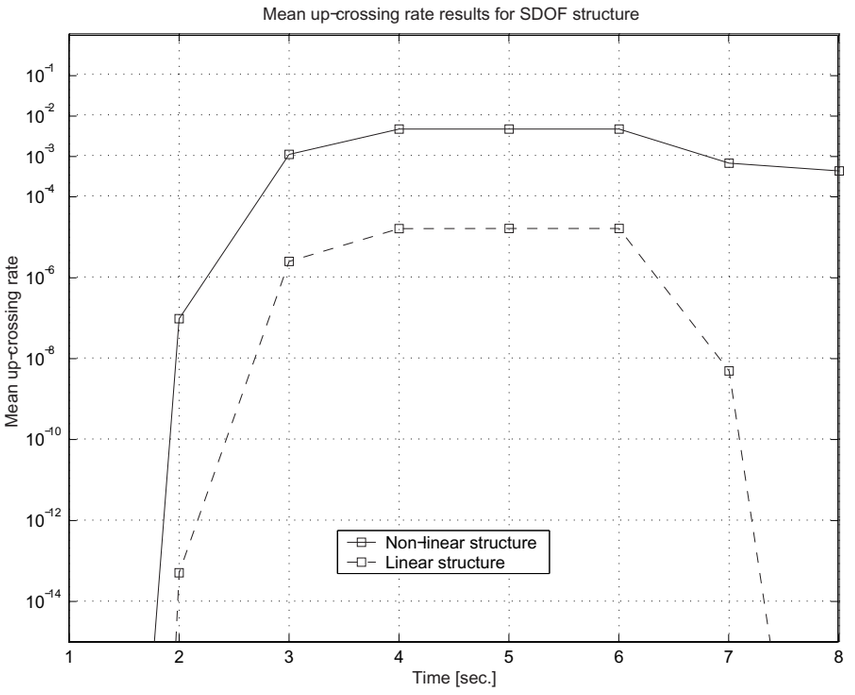

Figure 5.35: Mean up-crossing rate results for SDOF structure.

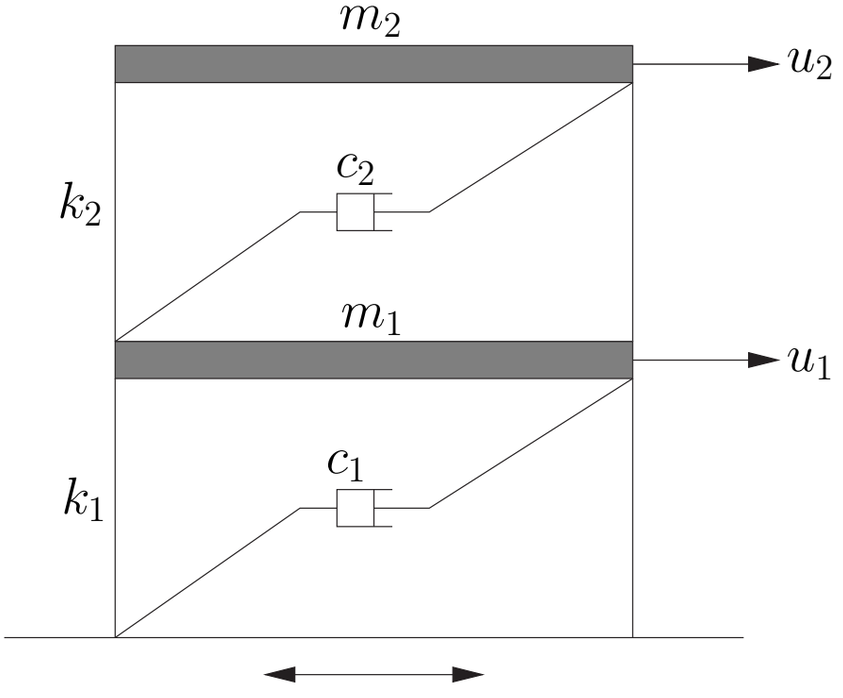

Figure 5.36: Two-degrees-of-freedom structure.

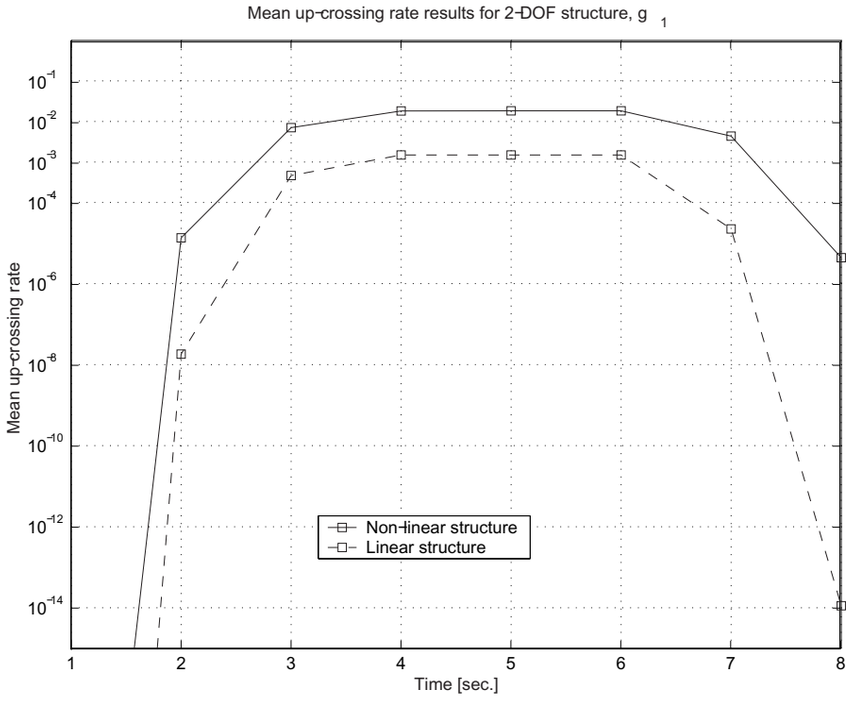

Figure 5.37: Mean up-crossing rate results for interstory drift of first floor of 2-DOF structure.

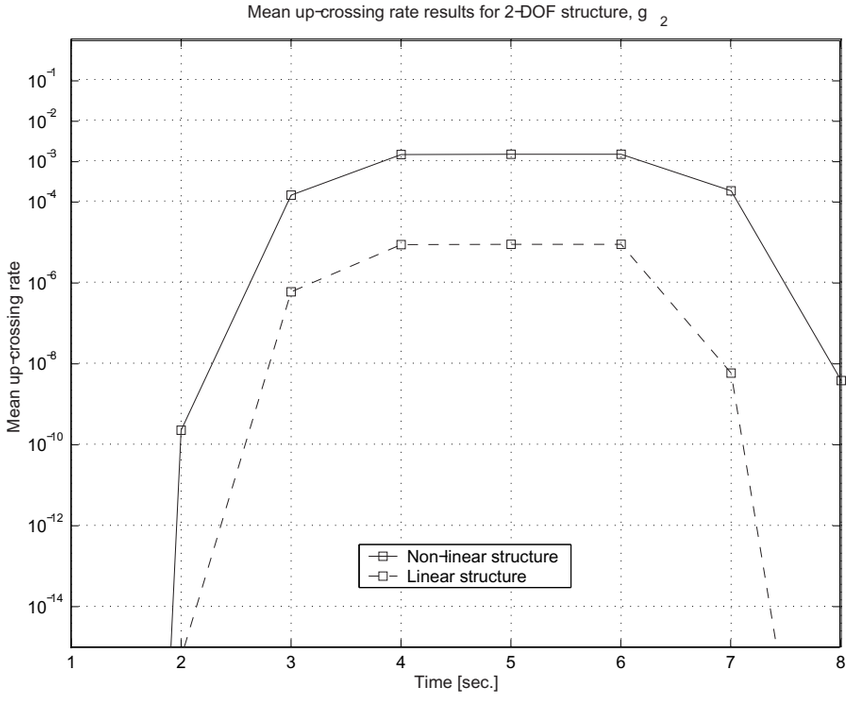

Figure 5.38: Mean up-crossing rate results for interstory drift of second floor of 2-DOF structure.

6 Conclusions

6.1 SUMMARY OF MAJOR FINDINGS

The presented work is motivated by the programmatic objective of the Pacific Earthquake Engineering Research (PEER) Center to develop tools and methodologies for performance-based earthquake engineering. The objective in this study is to develop a modern software framework for finite element reliability analysis. This is done by extending the already existing OpenSees finite element software. Implementations resulting from this study enable the user of OpenSees to characterize material parameters, cross-sectional geometry, nodal coordinates and load parameters of the finite element model as uncertain variables.

The object-oriented programming approach is utilized. This enables a transparent software architecture that facilitates maintainability and extensibility. As in the original OpenSees software, the reliability analysis module is divided into a logical set of interacting components. The framework can be extended by adding new methods of solving the various tasks without modifying the existing code. This is a particularly important feature for research software, where lack of developer continuity often leads to patchwork-type software designs.

Response sensitivity analysis is an important ingredient in finite element reliability methods. In this study, OpenSees has been extended with sensitivity capabilities by use of the Direct Differentiation Method. Unified equations are presented for response sensitivity with respect to material parameters, nodal coordinates, cross-sectional geometry and load parameters. Proper differentiation of kinematic equations leads to accurate response sensitivities with respect to nodal coordinates. The presented equations represent an extension of previously published results. Practical implementation issues, such as element assembly procedures, sensitivity history variables and use of the algorithmically consistent tangents are emphasized. Additionally, conditions are pointed out under which the top-level sensitivity equation becomes nonlinear. Remedies are suggested. Sensitivity equations for an array of elements and material models motivated by reliability analysis applications are presented.

The available shape sensitivity results allow the inclusion of uncertainty in nodal coordinates.

This is often neglected in finite element reliability applications in the literature. It is found that this is a significant source of uncertainty, which must be considered in practical applications when present.

The software framework is used to identify and address challenges in nonlinear finite element reliability analysis. Nonconvergence in the search for the design point may occur due to gradient discontinuities; "noise" in the finite element response; strong nonlinearities in the finite element response; and non-convergence of the finite element solution for certain outcomes of the random model parameters selected by the search algorithm. Modified material models with smooth transition between the elastic and plastic response regimes are developed, which overcome sensitivity discontinuities. The value of these findings are further enhanced by a proof that elastic unloading in an inelastic structure does not lead to sensitivity discontinuity. Motivated by these developments, sensitivity equations are derived for the smooth uniaxial degrading Bouc-Wen model and for the multi-axial generalized plasticity model. Smooth versions of the bi-linear steel material model and a concrete model are developed and the corresponding sensitivity equations are derived.

Modification of search algorithms for finding the design point has proven to be a fruitful approach to circumvent the problem of nonconvergence of the finite element solution when the trial point falls too far in the failure domain. In addition to modifying well-known algorithms, the Polak-He algorithm is introduced in reliability analysis. This algorithm possesses steering parameters that allows the user to control the rate at which the design point is approached.

The practical usefulness of parameter importance measures obtained as by-products from reliability analysis is emphasized. Such measures are used to gain physical insight and, in some cases, to reduce the number of random variables in the problem. When the structural behavior is similar at the mean point and the design point, then an inexpensive importance measure from a single finite element analysis at the mean point is available.

Numerical examples are presented to demonstrate some of the available analysis features and to emphasize the versatility of the software. A comprehensive study of the I-880 Highway Bridge, which is part of the PEER Testbed suite, is presented. Initially, all model parameters are characterized as uncertain. The number of random variables is then reduced by use of parameter importance measures. Propagation of uncertainty through the finite element analysis is demonstrated. Several performance functions are considered for reliability analysis. It is found that the use of smooth material models is necessary for convergence in the reliability analysis. Dynamic time-variant reliability analysis is addressed by implementation of mean out-crossing rate analysis options and discretized random process input. Reliability results for performance functions defined in terms of accumulated response

quantities are also presented.

A comprehensive User's and Developer's Guide for reliability and sensitivity analysis in OpenSees is developed. All source code and several examples are available at the OpenSees web site *http://opensees.berkeley.edu*.

6.2 FUTURE WORK

Further work is recommended along two directions. First, the software framework may be extended with SORM analysis options, alternative probability transformations, more elements and materials enabled for reliability and sensitivity analysis. Furthermore, the trend in large-scale structural analysis is towards network-based parallel computing. Certain types of reliability analysis, such as sampling analysis, lends itself well to separation into parallel analysis tasks. This approach has not been addressed in the present study and its consideration in the future is strongly recommended.

Second, challenges remain in the analysis methodology. In problems such as mean out-crossing rate computation for nonlinear structures subjected to stochastic load it is still challenging to find the design point. This emphasizes the need for further research on methods for time-variant reliability analysis or for more robust methods for finding the design point. In particular, methods suggested in Koo and Der Kiureghian (2003) for approximate solution of the design point through free vibration analysis should be explored.

The challenge of noise in the performance function due to numerical approximations in the finite element procedure is not addressed in this study, other than careful selection of the applicable tolerances and time steps. Development of optimization search algorithms which are less prone to noise in the constraint functions is therefore of interest.

Implementation of random field options in OpenSees will improve the user's ability to characterize uncertain structural properties. This is particularly the case of continuum-type problems in geotechnical engineering. Such work will extend the already implemented options for correlation structures in OpenSees. Such a study should also address the issue of parameter identification in the finite element model. The general concept of having a functional relationship characterizing, e.g., the material properties, of a range of elements (and integration points) is an important software architecture issue that is currently not explored in depth in OpenSees.

In this study, synthetic generation of ground motion is addressed with the implementation of a discretized random process time series. Given the significant uncertainty inherent in the ground

motion, this remains an important area for future research and development. Future research may incorporate developments in geo-hazard analysis and soil-structure interaction methodologies to refine the ground motion modeling.

Performance-based engineering allows the engineer to optimize a design based on a target reliability level. This motivates the implementation of reliability-based optimization algorithms in OpenSees. Such research will involve both software design issues, since optimization procedures may be rather costly, and further development of existing methodologies. In any case, the availability of response sensitivities from the present study will greatly facilitate future research in optimal design.

Model error is a source of uncertainty that is not addressed in the present study. Errors are introduced when idealizing reality into the boundary value problem presented in Chapter 2. Another error is introduced by the time and space discretization of the finite element method. Future research must address these sources of uncertainty and error. It is believed that this will be beneficial for acceptance of the methodology by the practicing engineering community. Furthermore, proper account of such uncertainties and errors is essential for validation of the finite element code.

REFERENCES

ABAQUS, Inc. (2003). "http://www.hks.com. ABAQUS". Pawtucket, RI.

Arora, J. S. and Haug, E. J. (1979). "Methods of design sensitivity analysis in structural optimization." *AIAA Journal*, 17(9), 970–974.

Au, S. K. and Beck, J. L. (2001). "First excursion probabilities for linear systems by very efficient importance sampling." *Probabilistic Engineering Mechanics*, 16, 193–207.

Baber, T. T. and Noori, M. N. (1985). "Random vibration of degrading, pinching systems." *Journal of Engineering Mechanics*, 111(8), 1010–1026.

Bathe, K.-J. (1996). *Finite Element Procedures*. Prentice Hall, Englewood Cliffs, N.J.

Bjerager, P. and Krenk, S. (1989). "Parameter sensitivity in first order reliability theory." *Journal of Engineering Mechanics*, 115(7), 1577–1582.

Bouc, R. (1971). *Mathematical model for hysteresis*. Report to the Centre de Recherches Physiques, pp16-25, Marseille, France.

Breitung, K. (1989). "Asymptotic approximations for probability integrals." *Probabilistic Engineering Mechanics*, 4, 187–190.

Casciati, F. (1989). "Stochastic dynamics of hysteretic media." *Structural Safety*, 6, 259–269.

Choi, K. K. and Santos, J. L. T. (1987). "Design sensitivity analysis of non-linear structural systems, part i: Theory." *International Journal of Numerical Methods in Engineering*, 24, 2039–2055.

Conte, J. P. (2000). "Finite element response sensitivity analysis in earthquake engineering." *Proceedings of the Cina-U.S. Millennium symposium of Earthquake Engineering: Earthquake Engineering Frontiers in the New Millennium*, Beijing, China. Eds: B. F. Spencer and Y. X. Hu.

Conte, J. P., Vijalapura, P. K., and Meghella, M. (1999). "Consistent finite element sensitivity in seismic reliability analysis." *Proceedings of 13th ASCE Engineering Mechanics Division Conference*, Baltimore, MD. The Johns Hopkins University.

Cook, R. D., Malkus, D. S., and Plesha, M. E. (1989). *Concepts and Applications of Finite Element Analysis*. John Wiley and Sons, New York, 3rd edition.

Crisfield, M. A. (1991). *Non-linear finite element analysis of solids and structures*, Vol. 1. John Wiley and Sons, West Sussex, U.K.

Deitel, H. M. and Deitel, P. J. (1998). *C++ How to program*. Prentice Hall, Inc., Upper Saddle River, NJ.

Der Kiureghian, A. (2000). "The geometry of random vibrations and solutions by form and sorm." *Journal of Engineering Mechanics*, 15(1), 81–90.

Der Kiureghian, A. (2003). "Lecture notes in CE229 - structural reliability. Department of Civil and Environmental Engineering, University of California, Berkeley, CA.

Der Kiureghian, A. and DeStefano, M. (1991). "Efficient algorithm for second-order reliability." *JOURNAL of Engineering Mechanics, ASCE*, 117(12), 2904–2923.

Der Kiureghian, A. and Li, C.-C. (1996). "Nonlinear random vibration analysis through optimization." *Proceedings of the 7th IFIP WG 7.5 working conference on reliability and optimization of structural systems*, Boulder, Colorado. Eds: D. Frangopol and R. Rackwitz.

Der Kiureghian, A. and Taylor, R. L. (1983). "Numerical methods in structural reliability." *Proceedings of the ICASP4*, Florence, Italy.

Der Kiureghian, A. and Zhang, Y. (1999). "Space-variant finite element reliability analysis." *Computer methods in applied mechanics and engineering*, 168, 173–183.

Det Norske Veritas (2003). "http://www.dnv.com/software/products/sesam. PROBAN". Hovik, Norway.

Ditlevsen, O. (1979). "Narrow reliability bounds for structural systems." *Journal of structural mechanics*, 7(4), 453–472.

Ditlevsen, O. and Madsen, H. O. (1996). *Structural Reliability Methods*. Wiley, Chichester, New York, NY.

Dunnett, C. W. and Sobel, M. (1955). "Approximations to the probability integral and certain percentage points of a multivariate analogue of student's t-distribution." *Biometrica*, 42, 258–260.

Federal Emergency Management Agency (2000). *Prestandard and Commentary for the Seismic Rehabilitation of Buildings, FEMA 356*. FEMA.

Frank, P. M. (1978). *Introduction to system sensitivity theory*. Academic Press.

Frier, C. and Sorensen, J. (2003). "Stochastic finite element analysis of non-linear structures modelled by plasticity theory." *Proceedings of the Ninth International Conference on Applications of Statistics and Probability in Civil Engineering, ICASP9*, San Francisco, California. Eds: A. Der Kiureghian and S. Madanat and J. Pestana.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns, Elements of Reusable Object-Oriented Software*. Professional Computing Series. Addison Wesley Longman, Inc., Reading, MA.

Gu, Q. and Conte, J. (2003). "Convergence studies in nonlinear finite element response sensitivity analysis." *Proceedings of the Ninth International Conference on Applications of Statistics and Probability in Civil Engineering, ICASP9*, San Francisco, California. Eds: A. Der Kiureghian and S. Madanat and J. Pestana.

Gurtin, M. E. (1981). *An introduction to continuum mechanics*, Vol. 158 of *Mathematics in Science and Engineering*. Academic Press, New York, NY.

Gutierrez, M., Carmeliet, J., and de Borst, R. (1994). "Finite element reliability methods using diana." *Diana Computational Mechanics 1994*. Eds: G.M.A. Kusters and M.A.N. Hendriks.

Hagen, O. and Tvedt, L. (1991). "Vector process out-crossing as parallel system sensitivity measure." *Journal of Engineering Mechanics*, 117(10), 2201–2220.

Haldar, A. and Mahadevan, S. (2000). *Reliability assessment using stochastic finite element analysis*. John Wiley and Sons, New York.

Hasofer, A. M. and Lind, N. C. (1974). "Exact and invariant second-moment code format." *Journal of Engineering Mechanics*, 100(1), 111–121.

Haukaas, T., Hahnel, A., Sudret, B., Song, J., and Franchin, P. (2003). "http://www.ce.berkeley.edu/ haukaas/ferum/ferum.html. FERUM". Department of Civil and Environmental Engineering, University of California, Berkeley, CA.

Hohenbichler, M. and Rackwitz, R. (1986). "Sensitivity and importance measures in structural reliability." *Civil Engineering Systems*, 3, 203–209.

Hughes, T. J. R. (1987). *The finite element method : linear static and dynamic finite element analysis*. Prentice Hall, Englewood Cliffs, N.J.

Hunter, D. (1976). "An upper bound for the probability of a union." *Journal of Applied Probability*, 13, 597–603.

IfM (2003). "http://mechanik.uibk.ac.at/softwaredevelopment. COSSAN and ISPUD". Innsbruck, Austria.

Imai, K. and Frangopol, D. M. (2000). "Geometrically nonlinear finite element reliability analysis of structural systems. i: theory ii: applications." *Computers and Structures*, 77(6), 677–709.

Kleiber, M., Antunez, H., Hien, T., and Kowalczyk, P. (1997). *Parameter Sensitivity in Nonlinear Mechanics*. John Wiley and Sons Ltd., West Sussex, U.K.

Koo, H. (2003). "Form, sorm and simulation techniques for nonlinear random vibrations," PhD thesis, University of California, Berkeley, CA.

Koo, H. and Der Kiureghian, A. (2003). *FORM, SORM and simulation techniques for nonlinear random vibrations*. Report No. UCB/SEMM-2003/01, Department of Civil and Environmental Engineering, University of California, Berkeley, CA.

Kounias, E. G. (1968). "Bounds for the probability of a union, with applications." *Annals of Mathematical Statistics*, 39(6), 2154–2158.

Kunnath, S. K., Jeremic, B., von Felten, A., and Bauer, K. (2003). "Simulation models for performance-based evaluation of the i-880 highway bridge." *Proceedings of the ASCE Structures Congress*, Seattle, WA.

Li, C.-C. and Der Kiureghian, A. (1995). "Mean out-crossing rate of nonlinear response to stochastic input." *Proceedings of ICASP7, 7th International Conference On Applications of Statistics and Probability in Civil Engineering*, Paris, France. Eds: M. Lemaire, J-L. Favre and A. Mbarki.

Liu, P.-L. and Der Kiureghian, A. (1986). "Multivariate distribution models with prescribed marginals and covariances." *Probabilistic Engineering Mechanics*, 1(2), 105–112.

Liu, P.-L. and Der Kiureghian, A. (1991a). "Finite element reliability of geometrically nonlinear uncertain structures." *Journal of Engineering Mechanics*, 17(8), 1806–1825.

Liu, P.-L. and Der Kiureghian, A. (1991b). "Optimization algorithms for structural reliability." *Structural Safety*, 9(3), 161–178.

Liu, P.-L., Lin, H.-Z., and Der Kiureghian, A. (1989). *CalREL User Manual*. Report No. UCB/SEMM-89/18, Department of Civil and Environmental Engineering, University of California, Berkeley, CA.

Lubliner, J., Taylor, R. L., and Auricchio, F. (1993). "A new model of generalized plasticity and its numerical implementation." *Int. J. Solid Structures*, 30(22), 3171–3184.

Luenberger, D. G. (1984). *Linear and Nonlinear Programming*. Addison-Wesley, Reading, Massachusetts, 2nd edition.

McKenna, F. and Fenves, G. L. (2002). "http://opensees.berkeley.edu. TheOpenSees command language primer". Department of Civil and Environmental Engineering, University of California, Berkeley, CA.

McKenna, F., Fenves, G. L., and Scott, M. H. (2002). "Open system for earthquake engineering simulation, http://opensees.berkeley.edu/". Pacific Earthquake Engineering Research Center, University of California, Berkeley, CA.

McKenna, F. T. (1997). "Object-oriented finite element programming: Frameworks for analysis, algorithms and parallel computing," PhD thesis, University of California, Berkeley, CA.

Melchers, R. E. (1999). *Structural Reliability Analysis and Prediction*. John Wiley and Sons, Chichester, 2nd edition.

MSC Software (2003). "http://www.mscsoftware.com. MSC.NASTRAN". Santa Ana, CA.

Park, Y. J. and Ang, A. H. S. (1985). "Mechanistic seismic damage model for reinforced concrete." *Journal of Structural Engineeering*, 111(4), 722–739.

Polak, E. (1997). *Optimization. Algorithms and Consistent Approximations.*, Vol. 124 of *Applied Mathematical Sciences*. Springer Verlag, New York, NY.

Polak, E. and He, L. (1991). "A unified steerable phase i - phase ii method of feasible directions for semi-infinite optimization." *Journal of Optimization Theory and Applications*, 69(1), 83–107.

PredictionProbe, Inc. (2003). "http://www.predictionprobe.com. UNIPASS". Newport Beach, CA.

Rackwitz, R. and Fiessler, B. (1978). "Structural reliability under combined load sequences." *Computers and Structures*, 9, 489–494.

Ray, D., Pister, K. S., and Polak, E. (1978). "Sensitivity analysis for hysteretic dynamic systems: theory and applications." *Computer Methods in Applied Mechanics and Engineering*, 14, 179–208.

Reliability Consulting Programs (2003). "http://www.strurel.de. STRUREL : COMREL, SYSREL, NASCOM, NASREL". Munich, Germany.

Roth, C. and Grigoriu, M. (2001). *Sensitivity Analysis of Dynamic Systems Subjected to Seismic Loads*. Report No. MCEER-01-0003, Multidisciplinary Center for Earthquake Engineering Research, State University of New York, Buffalo, NY.

Royset, J. O. (2002). "Reliability-based design optimization of series structural systems," PhD thesis, University of California, Berkeley, CA.

Sandia National Laboratories (2003). "http://www.sandia.gov. DAKOTA". Albuquerque, NM and Livermore, CA.

Schittkowski, K. (1985). "Nlpql: A fortran subroutine solving contrained nonlinear programming problems." *Annals of Operations Research*, 5, 485–500.

Scott, B. D., Park, R., and Priestley, M. J. N. (1982). "Stress-strain behavior of concrete confined by overlapping hoops at low and high strain rates." *ACI Journal*, 79(1).

Scott, M. H., Franchin, P., Fenves, G. L., and Filippou, F. C. (2003). "Response Sensitivity for Nonlinear Beam-Column Elements." *ASCE Journal of Structural Engineering*. Submitted for publication.

Simo, J. C. and Hughes, T. J. R. (1998). *Computational Inelasticity*. Interdisciplinary Applied Mathematics. Springer-Verlag, New York, NY.

Southwest Research Institute (2003). "http://www.nessus.swri.org. NESSUS". San Antonio, TX.

Sudret, B. and Der Kiureghian, A. (2000). *Stochastic finite element methods and reliability. A State-of-the-Art report*. Report No. UCB/SEMM-2000/08, DEPARTMENT of Civil and Environmental Engineering, University of California, Berkeley.

Taylor, R. L. (2003). "http://www.ce.berkeley.edu/ rlt/feap. FEAP". Department of Civil and Environmental Engineering, University of California, Berkeley, CA.

The Mathworks, Inc. (2003). "http://www.mathworks.com. Matlab". Natick, MA.

TNO Building and Construction Research (2003). "http://www.diana.tno.nl. DIANA". Department of Computational Mechanics, Delft, The Netherlands.

Tsay, J. J. and Arora, J. S. (1990). "Nonlinear structural design sensitivity analysis for path dependent problems. part1: General theory." *Computer Methods in Applied Mechanics and Engineering*, 81, 183–208.

Vamvatsikos, D. and Cornell, A. C. (2002). "Incremental dynamic analysis." *Earthquake Engineering and Structural Dynamics*, 31, 491–514.

Vijalapura, P. K. (1998). *Analysis of nonlinear structural systems accounting for both system uncertainty and excitation stochasticity*. M.S. Thesis, Civil Engineering Department, Rice University, Houston, TX.

Welch, B. B. (2000). *Practical programming in Tcl and Tk*. Prentice Hall, Upper Saddle River, New Jersey, 3rd edition.

Wen, Y.-K. (1976). "Method for random vibration of hysteretic systems." *Journal of Engineering Mechanics Division*, 102(EM2), 249–263.

Zhang, Y. and Der Kiureghian, A. (1993). "Dynamic response sensitivity of inelastic structures." *Computer Methods in Applied Science and Engineering*, 108, 23–36.

Zhang, Y. and Der Kiureghian, A. (1994). "First-excursion probability of uncertain structures." *Probabilistic Engineering Mechanics*, 9, 135–143.

Zhang, Y. and Der Kiureghian, A. (1997). *Finite Element Reliability Methods for Inelastic Structures*. Report No. UCB/SEMM-97/05, Department of Civil and Environmental Engineering, University of California, Berkeley, CA.

Zienkiewicz, O. and Taylor, R. (2000). *The finite element method*. Butterworth-Heinemann, Oxford, Boston, 5th edition.

Appendix A: Class Interfaces for Implementations in OpenSees

This appendix contains the class interfaces for the implementations made in OpenSees for sensitivity and reliability analyses. The information can also be found in the open-source directory at `http://opensees.berkeley.edu`. Class interface specifications show how an object-oriented code is organized. This information is not meant for the front-end user of the software. The class interface specifications are useful for developers, who aim at maintaining and extending the software.

The reliability analysis implementations follow a domain-analysis decomposition (Gamma *et al.* 1995). The data, such as random variables, are contained in a domain, which is acted upon by an analysis object. In the reliability analysis implementation several such analysis classes are available. Furthermore, the analysis makes use of a number of analysis components or "tools." These three sets of classes, namely domain, analysis and analysis tools, are discussed in the following sections.

It is assumed that the reader is familiar with object-oriented programming in the C++ language. Nevertheless, two notational issues are emphasized. First, a member function is characterized by its name, its return type and its argument list. The following example function with the name `doSomething` returns an integer and takes an integer "`a`," a real number "`b`" with double precision and a vector object "`c`" as arguments:

```
int doSomething(int a, double b, Vector c)
```

In the following sections, the member functions available in the reliability module are described. On the other hand, the data members of each class are not provided here. Secondly, the analysis tools and domain components that are listed below are generally base classes. This implies that they promise only certain features. The developer must implement specific subclasses to actually provide the computational features that are promised in the interface of the base class. A number of alternative subclasses may be available for one base class. This feature enables an extensible analysis framework. While the framework and the interaction between its components is completed, it is

still possible to implement improved algorithms to solve the various tasks without modifying the framework itself.

In the following the terms "member function" and "method" are used interchangeably.

A.1 ANALYSIS TYPES

Eight analysis types are available in the reliability module of OpenSees. They have the member function `analyze()` in common. The following is the class interface specification for these classes:

```
FORMAnalysis
    int analyze()
SORMAnalysis
    int analyze()
SamplingAnalysis
    int analyze()
MVFOSMAnalysis
    int analyze()
OutCrossingAnalysis
    int analyze()
FragilityAnalysis
    int analyze()
SystemAnalysis
    int analyze()
GFunVisualizationAnalysis
    int analyze()
```

The `GFunVisualizationAnalysis` class interface contains several additional member functions to set the search direction for the limit-state surface, etc. These details are not listed here. The common `analyze()` method contains the orchestrating algorithm for each analysis type. The integer being returned from these methods is used to distinguish a successful analysis and to characterize possible errors that may have occurred. It is noted that the analysis types do not contain methods to obtain results upon completed analysis. This is because these objects write results to files.

A.2 FRAMEWORK OF ANALYSIS COMPONENTS

The orchestrating analysis algorithms in the previous section make calls to selected analysis "tools" to perform certain tasks. This section describes the available tools. It is noted that this decomposition

of the analysis procedure is a key feature of the OpenSees software (McKenna 1997). Analysis components are aggregated at run-time based on the user's request. This is also the case for the reliability analysis module. The base classes of the reliability analysis framework and the features they promise are as follows:

Interface for tool to find the design point:

```
FindDesignPointAlgorithm
    int findDesignPoint(Vector *startPt, ReliabilityDomain *relDom)
    Vector get_x()
    Vector get_u()
    Vector get_alpha()
    Vector get_gamma()
    int getNumberOfIterations()
    double getFirstGFunValue()
```

In addition to the above, the `FindDesignPointAlgorithm` class contains some methods that are used in gradient-based SORM analysis. These details are not described in this work. See *http://opensees.berkeley.edu* for more information. Currently, one specific subclass implementation of this analysis tool is available, namely `SearchWithStepSizeAndStepDirection`. It is noted that this algorithm makes use of several other analysis tools described in this section

Interface for probability transformation tool:

```
ProbabilityTransformation
    int set_x(Vector x)
    int set_u(Vector u)
    int transform_x_to_u()
    int transform_u_to_x()
    int transform_u_to_x_andComputeJacobian()
    Vector get_x()
    Vector get_u()
    Matrix getJacobian_x_u()
    Matrix getJacobian_u_x()
```

It is seen that the probability transformation class is a typical object-oriented component. It contain methods to first set its data, then to perform operations on them and finally to return the results. The resulting data is stored in the base class. It is also noted that the transformation can be done both with and without computing the associated Jacobian matrix. Currently, `NatafProbabilityTransformation` is the only specific implementation of this analysis tool.

Interface for search direction tool:

```
SearchDirection
    int computeSearchDirection(Vector u, double g, Vector gradG)
    Vector getSearchDirection()
```

Currently, the following specific implementations of the search direction algorithm are available: `iHLRF`, `GradientProjection`, `PolakHe` and `SQP`.

Interface for step size tool:

```
StepSizeRule
    int computeStepSize(Vector u, Vector gradG, double G, Vector d)
    double getStepSize()
```

Two specific implementations of the step size base class are currently available. Namely, `Armijo` and `Fixed`.

Interface for tool to evaluate the value of a performance function:

```
GFunEvaluator
    int evaluateG(Vector x)
    double getG()
    int runGFunAnalysis(Vector x)
    int tokenizeSpecials(char *expression)
```

The tool to evaluate the value of the g-function (performance function) has some unique features in the analysis framework; namely, some of the tasks are implemented in the base class itself. This is the case with the methods `evaluateG` and `getG`. The member functions `runGFunAnalysis` and `tokenizeSpecials`, on the other hand, must be provided by the specific subclasses. The reason for this is as follows. When an algorithm needs to evaluate the performance function it first calls the `runGFunAnalysis` method. It is, of course, the specific type of g-function that must perform this task. For instance, an OpenSees finite element analysis may be run. Next, the `evaluateG` method is called. This algorithm goes through the expression of the performance function and sets the value of its quantities. This can be done by the base class for basic random variables and a few other quantities, but a specific subclass may have its own quantities available for use in the performance function (with pre-defined syntax.) For this reason the `tokenizeSpecials` method is called by the `evaluateG` method in the base class to make sure that all parameters are set. At the end of `evaluateG` the performance function is evaluated. The running Tcl interpreter is used as parser for this purpose. Lastly, the method `getG` can be called to obtain the value of the performance function. A few other member functions are available for tasks specifically needed by the mean out-crossing rate analysis type. Currently, the user can evaluate the value of the performance function by: `OpenSees`, `Tcl`, `Matlab` and `Basic`.

Interface for tool to compute the gradient of a performance function:

```
GradGEvaluator
    int computeGradG(double g, Vector x)
```

```
    int computeAllGradG(Vector g, Vector x)
    Vector getGradG()
    Matrix getAllGradG()
    Matrix getDgDdispl()
    Matrix getDgDpar()
```

The main promise made by the `GradGEvaluator`, namely to compute and return the gradient vector of a performance function (or of several performance functions and returning a matrix), is delivered by subclasses. The value of the performance function itself is passed so that unnecessary evaluations are avoided for the cases where finite difference schemes are employed. The two latter methods listed for the `GradGEvaluator` are used to obtain derivatives with respect to parameters that explicitly enter the performance function. A finite difference scheme is used for this purpose. Currently, two alternatives are available as specific implementations for computing the gradients; namely, `FiniteDifference` and `OpenSees`. The latter makes use of the DDM implementations in OpenSees.

Interface for tool to generate random numbers:

```
RandomNumberGenerator
    int generate_nIndependentStdNormalNumbers(int n)
    Vector getGeneratedNumbers()
```

Currently, the only subclass that is implemented for random number generation is the `CStdLibRandGenerator` based on the random number generator available in the standard library of the C++ programming language.

Interface for tool to compute curvatures of the limit-state surface at the design point:

```
FindCurvatures
    int computeCurvatures(ReliabilityDomain *relDomain)
    Vector getCurvatures()
```

The specific implementation `FirstPrinicipleCurvature` is currently available. This makes use of results from the last step of a prior design point search to compute the major principle curvature.

Class interface for tool to perform a merit function check:

```
MeritFunctionCheck
    double getMeritFunctionValue(Vector u, double g, Vector gradG)
    int updateMeritParameters(Vector u, double g, Vector gradG)
```

Currently the following merit functions are available: `AdkZhang`, `PolakHe`, `CriteriaReduction`, and `SQP`. The former contains the merit function formulated by Zhang and Der Kiureghian (1997), while the next two usually are used in conjunction with their corresponding search direction algorithms. The last one is a self-explanatory simple implementation.

Class interface for tool to check design point convergence:

```
ReliabilityConvergenceCheck
    int check(Vector u, double g, Vector gradG)
    int getNumberOfCriteria()
    double getCriteriaValue(int whichCriteria)
    int setScaleValue(double scaleValue)
```

In the current subclass implementations, two alternatives are available for checking whether a trial point is a design point or not: `Standard` and `OptimalityCondition`.

Class interface for tool to compute an approximation of the Hessian of a performance function:

```
HessianApproximation
    Matrix getHessianApproximation()
    int setHessianToIdentity(int size)
    int updateHessianApproximation(Vector u_old, double g_old,
              Vector gradG_old, double stepSize, Vector searchDir,
              double g_new, Vector gradG_new)
```

Currently, only the BFGS scheme is implemented (as part of the SQP algorithm to determine the design point).

Class interface for tool to find the root of a multi-dimensional function:

```
RootFinding
    Vector findLimitStateSurface(int space, double g,
                    Vector direction, Vector thePoint)
```

Currently, only the `SecantRootFinding` object is implemented to perform this task.

In addition to the analysis components listed above, a "matrix operations" tool has been added to the reliability analysis module. The purpose of the tool is to perform Cholesky decomposition, matrix transposition, etc.

A.3 THE DOMAIN

The domain is the part of the code that contains the model data and possibly results from a previous analysis. The following classes are currently available in the reliability domain.

Random variable:

```
RandomVariable
    double getPDFvalue(double rvValue)
    double getCDFvalue(double rvValue)
    double getInverseCDFvalue(double rvValue)
```

```
    const char* getType()
    double getMean()
    double getStdv()
    double getParameter1()
    double getParameter2()
    double getParameter3()
    double getParameter4()
    double getStartValue()
    int setNewTag(int tag)
```

The last method listed is used, for instance, when the number of random variables is reduced based on pre-computed importance measures. In that case the random variables are renumbered to remain consecutive. The available subclasses for the `randomVariable` base class corresponds to the list of distributions provided in Appendix B.9.

Correlation coefficient:

```
CorrelationCoefficient
    int getRv1()
    int getRv2()
    double getCorrelationCoefficient()
```

In the current OpenSees implementation the correlation matrix does not exist as an entity in the reliability domain. The matrix is established by the selected probability transformation object based on all existing correlation coefficient objects. Commands have been made available, however, to facilitate easy input of correlation structures and correlation groups. The correlation coefficient class does not have subclasses but any number of its objects can exist simultaneously in the reliability domain.

Performance function:

```
PerformanceFunction
    char *getExpression()
    char *getTokenizedExpression()
    int addExpression(char *expression)
    int removeAddedExpression()
```

The last two methods of the performance function object are particularly useful in mean out-crossing rate analysis, where a term may be added to the original expression to obtain a second design point. The performance function class does not have subclasses but any number of its objects can exist simultaneously in the reliability domain.

Random variable positioner:

```
RandomVariablePositioner
    int getRvNumber()
```

```
    int update (double newValue)
    int activate(bool active)
    int setNewTag(int newTag)
    int setRvNumber(int newRvNumber)
```

The random variable positioner class is a vital part of the mapping of random variables into a finite element domain. The `update` method is used to update the finite element model with new realizations of a random model quantity. The `activate` method is used in DDM sensitivity analysis. The latter two methods are employed, e.g., when the number of random variables is reduced based on their importance rankings.

Parameter positioner:

```
ParameterPositioner
    int update (double newValue)
    int activate(bool active)
```

The parameter positioner serves the same purpose as the random variable positioner. However, in this case a random variable need not be created. This is useful, e.g., in stand-alone sensitivity analysis and in "fragility" analysis when a deterministic parameter in the finite element model is to be varied over a range of values.

Filter:

```
Filter
    double getAmplitude(double time)
    double getMaxAmplitude()
    double getTimeOfMaxAmplitude()
```

Currently, only one filter is used in OpenSees; namely, the standard linear oscillator.

Modulating function:

```
ModulatingFunction
    double getAmplitude(double time)
    double getMaxAmplitude()
    Filter *getFilter()
```

Currently, the following types of modulating functions have been implemented: `Constant`, `Trapezoidal` and `Gamma`.

Spectrum:

```
Spectrum
    double getMinFrequency()
    double getMaxFrequency()
    double getAmplitude(double frequency)
```

In this study the power spectral density object has not be extensively used. However, three alternatives are available: `Jonswap`, `NarrowBand` and `Points`. The latter lets the user specify PSD functions of arbitrary shape.

A.4 DDM SENSITIVITY INTERFACE ADDITIONS

The computation of response sensitivities by the Direct Differentiation Method is orchestrated in OpenSees by a "sensitivity algorithm" with the following class interface:

```
SensitivityAlgorithm
    int computeSensitivities()
```

In turn, the sensitivity algorithm calls a "sensitivity integrator" which has the following class interface:

```
SensitivityIntegrator
    int formSensitivityRHS(int gradNum)
    int formIndependentSensitivityRHS()
    int saveSensitivity(const Vector &v, int gradNum, int numGrads)
    int commitSensitivity(int gradNum, int numGrads)
```

A `StaticSensitivityIntegrator` is available for the static case. In the dynamic case, the sensitivity integrator subclasses are also subclasses of specific ordinary integrator classes, such as the Newmark integrator.

Methods are also added to classes belonging to the core OpenSees finite element module. Elements, sections, materials, nodes, load patterns, etc., are extended to enable the assembly of the right-hand side of the sensitivity equation and to store results and sensitivity history variables.

Appendix B: Algorithms for Sensitivity Computations

B.1 INCREMENTAL RESPONSE EQUATIONS FOR UNIAXIAL SMOOTHED BI-LINEAR STEEL MATERIAL

The algorithm referred to in Section 2.8.4 is presented here. The initial values of the radius an center coordinates of the circle are computed as:

1. $\overline{BC} = \frac{1-\gamma}{\eta}\sqrt{1+\eta^2}$
2. $\Delta y_{BD} = (1-\gamma) + \frac{\overline{BC}\, b\, \eta}{\sqrt{1+(b\eta)^2}}$
3. $\Delta x_{BD} = \frac{1-\gamma}{\eta} + \frac{\overline{BC}\, b\, \eta}{b\, \eta\sqrt{1+(b\eta)^2}}$
4. $\tilde{A}_x = \frac{\Delta y_{BD} + \frac{1}{b\eta}\left(\frac{\gamma}{\eta} + \Delta x_{BD}\right) - \frac{\gamma}{\eta^2}}{\left(\frac{1}{b\eta} - \frac{1}{\eta}\right)}$
5. $\tilde{A}_y = -\frac{1}{\eta}\,\tilde{A}_x + \gamma\left(1 + \frac{1}{\eta^2}\right)$
6. $R = \sqrt{\left(\gamma - \tilde{A}_y\right)^2 + \left(\frac{\gamma}{\eta} - \tilde{A}_x\right)}$

The corresponding x-coordinates of the start and end points of the circular segment in the tension and compression sides are computed as:

$$\begin{aligned} \tilde{B}_x^{tension} &= \frac{\gamma}{\eta} \\ \tilde{D}_x^{tension} &= \tilde{B}_x^{tension} + \Delta x_{BD} \\ \tilde{B}_x^{compression} &= -\tilde{B}_x^{tension} \\ \tilde{D}_x^{compression} &= -\tilde{D}_x^{tension} \end{aligned} \tag{B.1}$$

At each step these points are shifted along with the center coordinates according to the updating rules employed for the center coordinates (see later in this section).

The algorithm to compute the stress response for a given strain is as follows:

1. Compute the normalized strain: $x = \frac{\epsilon_{i+1}}{\sigma_y}\frac{E}{\eta}$.

2. Establish the maximum and minimum stress limits, including the circular segment:

   (a) Default minimum stress: $\sigma_{min} = b\ E\ \epsilon_{i+1} - \sigma_y\ (1-b)$

   (b) Default maximum stress: $\sigma_{max} = b\ E\ \epsilon_{i+1} + \sigma_y\ (1-b)$

   (c) Corresponding default tangent: $k_T = b\ E$

   (d) Correct if the strain is within the end points of the circular segment:

   if $(B_x^{tension} < x < D_x^{tension})$

   i. $\tau_1 = \sqrt{R^2 - \left(x - A_x^{tension}\right)^2}$

   ii. $\sigma_{max} = \sigma_y \left(\tau_1 + A_y^{tension}\right)$

   iii. $k_T = -\sigma_y \left(\frac{\left(x - A_x^{tension}\right)\frac{E}{\eta\sigma_y}}{\tau_1}\right)$

   (e) **if** $(D_x^{compression} < x < B_x^{compression})$

   i. $\tau_2 = \sqrt{R^2 - \left(x - A_x^{compression}\right)^2}$

   ii. $\sigma_{min} = \sigma_y \left(-\tau_2 + A_y^{compression}\right)$

   iii. $k_T = \sigma_y \frac{\left(x - A_x^{compression}\right)\frac{E}{\eta\sigma_y}}{\tau_2}$

3. Compute the trial elastic stress: $\sigma_{i+1}^{el} = \sigma_i + E\ (\epsilon_{i+1} - \epsilon_i)$.

4. Possibly correct stress state:

   (a) **if** $(\sigma_{max} < \sigma_{i+1}^{el})$

   i. $\sigma_{i+1} = \sigma_{max}$

   ii. $k_T = k_T$.

   iii. Determine the material state for the subsequent update of circle coordinates: **if** $(B_x^{tension} < x < D_x^{tension})$ $state = 2a$ **else** $state = 4a$.

   (b) **else if** $(\sigma_{min} > \sigma_{i+1}^{el})$

   i. $\sigma_{i+1} = \sigma_{min}$

   ii. $k_T = k_T$.

   iii. Determine the material state for the subsequent update of circle coordinates: **if** $(D_x^{compression} < x < B_x^{compression})$ $state = 2b$ **else** $state = 4b$.

   (c) **else**

i. $\sigma_{i+1} = \sigma_{i+1}^{el}$

ii. $k_T = E$

iii. Determine the boundaries of the elastic region (see Figure 2.14) for the purpose of state determination:

$\sigma_{min}^{inelastic} = b\ E\ \epsilon_{i+1} - \gamma\sigma_y\ (1-b)$

$\sigma_{max}^{inelastic} = b\ E\ \epsilon_{i+1} + \gamma\sigma_y\ (1-b)$

iv. Determine the material state for the subsequent update of circle coordinates: **if** $(\sigma_{i+1} > \sigma_{max}^{inelastic})$ **then** $state = 3a$, **else if** $(\sigma_{i+1} < \sigma_{min}^{inelastic})$ **then** $state = 3b$, **else** $state = 1$.

5. Finally, the parameters of the circle are updated depending on the material state. As opposed to the above items, this is done only after convergence when the material state is committed. The initial parameters are distinguished by a tilde.

(a) $\epsilon_{shift} = \frac{E\,\epsilon_{i+1} - \sigma_{i+1}}{E\,(1-b)}$

(b) $x_{shift} = \frac{E\,\epsilon_{shift}}{\sigma_y\,\eta}$

(c) **if** $(state = 1)$

i. $A_x^{tension} = \tilde{A}_x^{tension} + x_{shift}$

ii. $A_y^{tension} = \tilde{A}_y^{tension} + b\,\eta\,x_{shift}$

iii. $A_x^{compression} = \tilde{A}_x^{compression} + x_{shift}$

iv. $A_y^{compression} = \tilde{A}_y^{compression} + b\,\eta\,x_{shift}$

v. $B_x^{tension} = \tilde{B}_x^{tension} + x_{shift}$

vi. $D_x^{tension} = \tilde{D}_x^{tension} + x_{shift}$

vii. $B_x^{compression} = \tilde{B}_x^{compression} + x_{shift}$

viii. $D_x^{compression} = \tilde{D}_x^{compression} + x_{shift}$

(d) **else if** $(state = 2a)$

i. $A_x^{compression} = \tilde{A}_x^{compression} + x_{shift}$

ii. $A_y^{compression} = \tilde{A}_y^{compression} + b\,\eta\,x_{shift}$

iii. $B_x^{compression} = \tilde{B}_x^{compression} + x_{shift}$

iv. $D_x^{compression} = \tilde{D}_x^{compression} + x_{shift}$

(e) **else if** $(state = 2b)$

i. $A_x^{tension} = \tilde{A}_x^{tension} + x_{shift}$

ii. $A_y^{tension} = \tilde{A}_y^{tension} + b\,\eta\,x_{shift}$

iii. $B_x^{tension} = \tilde{B}_x^{tension} + x_{shift}$

iv. $D_x^{tension} = \tilde{D}_x^{tension} + x_{shift}$

(f) **else if** $(state = 3a)$

i. $A_x^{compression} = \tilde{A}_x^{compression} + x_{shift}$

ii. $A_y^{compression} = \tilde{A}_y^{compression} + b\,\eta\,x_{shift}$

iii. $B_x^{compression} = \tilde{B}_x^{compression} + x_{shift}$

iv. $D_x^{compression} = \tilde{D}_x^{compression} + x_{shift}$

v. $x_1 = \frac{E\,\epsilon_{i+1}}{\sigma_y\,\eta}$

vi. $y_1 = \frac{\sigma_{i+1}}{\sigma_y}$

vii. $x_2 = \frac{y_1 + \frac{x_1}{b\,\eta} - (1-b)}{b\,\eta + \frac{1}{b\,\eta}}$

viii. $y_2 = b\,\eta\,x_2 + (1-b)$

ix. $l_{1,2} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$

x. $l_{2,3} = \sqrt{R^2 - (l_{1,2} - R)^2}$

xi. $l_{2,3,y} = \frac{b\,\eta\,l_{2,3}}{\sqrt{1+b^2\eta^2}}$

xii. $l_{2,3,x} = \frac{l_{2,3}}{\sqrt{1+b^2\eta^2}}$

xiii. $\Delta R_x = \frac{R}{\sqrt{\frac{1}{b^2\eta^2}+1}}$

xiv. $\Delta R_y = \frac{\frac{R}{b\,\eta}}{\sqrt{\frac{1}{b^2\eta^2}+1}}$

xv. $l = D_x^{tension} - B_x^{tension}$

xvi. $A_x^{tension} = x_2 + l_{2,3,x} + \Delta R_x$

xvii. $A_y^{tension} = y_2 + l_{2,3,y} - \Delta R_y$

xviii. $D_x^{tension} = x_2 + l_{2,3,x}$

xix. $B_x^{tension} = D_x^{tension} - l$

(g) **else if** $(state = 3b)$

i. $A_x^{tension} = \tilde{A}_x^{tension} + x_{shift}$

ii. $A_y^{tension} = \tilde{A}_y^{tension} + b\,\eta\,x_{shift}$

iii. $B_x^{tension} = \tilde{B}_x^{tension} + x_{shift}$

iv. $D_x^{tension} = \tilde{D}_x^{tension} + x_{shift}$

v. $x_1 = \frac{E\,\epsilon_{i+1}}{\sigma_y\,\eta}$

vi. $y_1 = \frac{\sigma_{i+1}}{\sigma_y}$

vii. $x_2 = \frac{y_1 + \frac{x_1}{b\,\eta} + (1-b)}{b\,\eta + \frac{1}{b\,\eta}}$

viii. $y_2 = b\,\eta\,x_2 - (1-b)$

ix. $l_{1,2} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$

x. $l_{2,3} = \sqrt{R^2 - (l_{1,2} - R)^2}$

xi. $l_{2,3,y} = \frac{b\,\eta\,l_{2,3}}{\sqrt{1+b^2\eta^2}}$

xii. $l_{2,3,x} = \frac{l_{2,3}}{\sqrt{1+b^2\eta^2}}$

xiii. $\Delta R_x = \frac{R}{\sqrt{\frac{1}{b^2\eta^2}+1}}$

xiv. $\Delta R_y = \frac{\frac{R}{b\,\eta}}{\sqrt{\frac{1}{b^2\eta^2}+1}}$

xv. $l = B_x^{compression} - D_x^{compression}$

xvi. $A_x^{compression} = x_2 - l_{2,3,x} - \Delta R_x$

xvii. $A_y^{compression} = y_2 - l_{2,3,y} + \Delta R_y$

xviii. $D_x^{compression} = x_2 - l_{2,3,x}$

xix. $B_x^{compression} = D_x^{compression} + l$

(h) **else if** $(state = 4a)$

i. $A_x^{compression} = \tilde{A}_x^{compression} + x_{shift}$

ii. $A_y^{compression} = \tilde{A}_y^{compression} + b\,\eta\,x_{shift}$

iii. $B_x^{compression} = \tilde{B}_x^{compression} + x_{shift}$

iv. $D_x^{compression} = \tilde{D}_x^{compression} + x_{shift}$

(i) **else if** $(state = 4b)$

i. $A_x^{tension} = \tilde{A}_x^{tension} + x_{shift}$

ii. $A_y^{tension} = \tilde{A}_y^{tension} + b\,\eta\,x_{shift}$

iii. $B_x^{tension} = \tilde{B}_x^{tension} + x_{shift}$

iv. $D_x^{tension} = \tilde{D}_x^{tension} + x_{shift}$

B.2 CONDITIONAL STRESS DERIVATIVE FOR UNIAXIAL SMOOTHED BI-LINEAR STEEL MATERIAL

The algorithm referred to in Section 2.8.5 is presented below. It is assumed that the quantities from the material state determination are available (see previous section). The sensitivity history variables $\frac{\partial \sigma_i}{\partial h}$, $\frac{\partial \epsilon_i}{\partial h}$, $\frac{\partial A_x^{tension}}{\partial h}$, $\frac{\partial A_x^{compression}}{\partial h}$, $\frac{\partial A_y^{compression}}{\partial h}$ and $\frac{\partial A_y^{compression}}{\partial h}$ are updated by the algorithm presented in the next section.

1. Initialize sensitivity history variables (done only in the initial step):

   (a) $\frac{\partial \overline{BC}}{\partial h} = \sqrt{1+\eta^2}\frac{-\frac{\partial \gamma}{\partial h}\eta-(1-\gamma)\frac{\partial \eta}{\partial h}}{\eta^2} + \frac{1-\gamma}{\eta}\frac{\eta\frac{\partial \eta}{\partial h}}{\sqrt{1+\eta^2}}$

   (b) $\frac{\partial \Delta y_{BD}}{\partial h} = -\frac{\partial \gamma}{\partial h} + \frac{\left(\frac{\partial \overline{BC}}{\partial h}b\eta+\overline{BC}\frac{\partial b}{\partial h}\eta+\overline{BC}b\frac{\partial \eta}{\partial h}\right)\sqrt{1+(b\eta)^2}-\overline{BC}b\eta\frac{1}{\sqrt{1+(b\eta)^2}}b\eta\left(\frac{\partial b}{\partial h}\eta+b\frac{\partial \eta}{\partial h}\right)}{1+(b\eta)^2}$

   (c) $\frac{\partial \Delta x_{BD}}{\partial h} = \frac{-\frac{\partial \gamma}{\partial h}\eta-(1-\gamma)\frac{\partial \eta}{\partial h}}{\eta^2} + \frac{\frac{\partial \overline{BC}}{\partial h}\sqrt{1+(b\eta)^2}-\frac{\overline{BC}b\eta\left(\frac{\partial b}{\partial h}\eta+b\frac{\partial \eta}{\partial h}\right)}{\sqrt{1+(b\eta)^2}}}{1+(b\eta)^2}$

   (d) $\tau_3 = \frac{\left(\frac{\frac{\partial \gamma}{\partial h}\eta-\gamma\frac{\partial \eta}{\partial h}}{\eta^2}+\frac{\partial \Delta x_{BD}}{\partial h}\right)b\eta-\left(\frac{\gamma}{\eta}+\Delta x_{BD}\right)\left(\frac{\partial b}{\partial h}\eta+b\frac{\partial \eta}{\partial h}\right)}{(b\eta)^2}$

   (e) $\tau_4 = \left[\Delta y_{BD} + \frac{1}{b\eta}\left(\frac{\gamma}{\eta}+\Delta x_{BD}\right) - \frac{\gamma}{\eta^2}\right]\left[-(b\eta)^{-2}\left(\frac{\partial b}{\partial h}\eta + b\frac{\partial \eta}{\partial h}\right) + \eta^{-2}\frac{\partial \eta}{\partial h}\right)]$

   (f) $\frac{\partial \tilde{A}_x}{\partial h} = \frac{\left(\frac{\partial \Delta y_{BD}}{\partial h}+\tau_3-\frac{\frac{\partial \gamma}{\partial h}\eta^2-2\gamma\eta\frac{\partial \eta}{\partial h}}{\eta^4}\right)\left(\frac{1}{b\eta}-\frac{1}{\eta}\right)-\tau_4}{\left(\frac{1}{b\eta}-\frac{1}{\eta}\right)^2}$

   (g) $\frac{\partial \tilde{A}_y}{\partial h} = -\frac{\frac{\partial \tilde{A}_x}{\partial h}\eta-\tilde{A}_x\frac{\partial \eta}{\partial h}}{\eta^2} + \frac{\partial \gamma}{\partial h}\left(1+\frac{1}{\eta^2}\right) - 2\gamma\eta^{-3}\frac{\partial \eta}{\partial h}$

   (h) $\frac{\partial A_x^{tension}}{\partial h} = \frac{\partial \tilde{A}_x}{\partial h}$

   (i) $\frac{\partial A_x^{compression}}{\partial h} = -\frac{\partial \tilde{A}_x}{\partial h}$

   (j) $\frac{\partial A_y^{tension}}{\partial h} = \frac{\partial \tilde{A}_y}{\partial h}$

   (k) $\frac{\partial A_y^{compression}}{\partial h} = -\frac{\partial \tilde{A}_y}{\partial h}$

   (l) $\frac{\partial \sigma_i}{\partial h} = 0$

   (m) $\frac{\partial \epsilon_i}{\partial h} = 0$

2. Compute the derivative of the radius of the circle (remains constant throughout the analysis):
   $\frac{\partial R}{\partial h} = \frac{1}{R}\left(\left(\gamma-\tilde{A}_y\right)\left(\frac{\partial \gamma}{\partial h}-\frac{\partial \tilde{A}_y}{\partial h}\right)+\left(\frac{\gamma}{\eta}-\tilde{A}_x\right)\left(\frac{\frac{\partial \gamma}{\partial h}\eta-\gamma\frac{\partial \eta}{\partial h}}{\eta^2}-\frac{\partial \tilde{A}_x}{\partial h}\right)\right)$

3. Compute the derivative of the elastic stress:
   $\left.\frac{\partial \sigma_{i+1}^{el}}{\partial h}\right|_{\epsilon_{i+1}\ fixed} = \frac{\partial \sigma_i}{\partial h} + \frac{\partial E}{\partial h}\ (\epsilon_{i+1}-\epsilon_i) - E\ \frac{\partial \epsilon_i}{\partial h}$

4. Compute the derivative of the default minimum stress:
$\left.\frac{\partial \sigma_{min}}{\partial h}\right|_{\epsilon_{i+1}\ fixed} = \frac{\partial b}{\partial h}\ E\ \epsilon_{i+1} + b\ \frac{\partial E}{\partial h}\ \epsilon_{i+1} - \frac{\partial \sigma_y}{\partial h}\ (1-b) + \sigma_y\ \frac{\partial b}{\partial h}$

5. Compute the derivative of the default maximum stress:
$\left.\frac{\partial \sigma_{max}}{\partial h}\right|_{\epsilon_{i+1}\ fixed} = \frac{\partial b}{\partial h}\ E\ \epsilon_{i+1} + b\ \frac{\partial E}{\partial h}\ \epsilon_{i+1} + \frac{\partial \sigma_y}{\partial h}\ (1-b) - \sigma_y\ \frac{\partial b}{\partial h}$

6. Correct the derivatives of the maximum/minimum stress if the strain is within the end points of the circular segment:

   (a) **if** $(B_x^{tension} < x < D_x^{tension})$

      i. $\left.\frac{\partial x}{\partial h}\right|_{\epsilon_{i+1}\ fixed} = \frac{\epsilon_{i+1}\frac{\partial E}{\partial h}\sigma_y\,\eta - \epsilon_{i+1}E\left(\frac{\partial \sigma_y}{\partial h}\,\eta + \sigma_y\,\frac{\partial \eta}{\partial h}\right)}{(\sigma_y\,\eta)^2}$

      ii. $\left.\frac{\partial \tau_1}{\partial h}\right|_{\epsilon_{i+1}\ fixed} = \frac{1}{\tau_1}\left(R\frac{\partial R}{\partial h} - (x - A_x^{tension})\left(\left.\frac{\partial x}{\partial h}\right|_{\epsilon_{i+1}\ fixed} - \frac{\partial A_x^{tension}}{\partial h}\right)\right)$

      iii. $\left.\frac{\partial \sigma_{max}}{\partial h}\right|_{\epsilon_{i+1}\ fixed} = \frac{\partial \sigma_y}{\partial h}\left(\tau_1 + A_y^{tension}\right) + \sigma_y\left(\left.\frac{\partial \tau_1}{\partial h}\right|_{\epsilon_{i+1}\ fixed} + \frac{\partial A_y^{tension}}{\partial h}\right)$

   (b) **else if** $(D_x^{compression} < x < B_x^{compression})$

      i. $\left.\frac{\partial x}{\partial h}\right|_{\epsilon_{i+1}\ fixed}$ = as item (a) above.

      ii. $\left.\frac{\partial \tau_2}{\partial h}\right|_{\epsilon_{i+1}\ fixed} = \frac{1}{\tau_2}\left(R\frac{\partial R}{\partial h} - (x - A_x^{compression})\left(\left.\frac{\partial x}{\partial h}\right|_{\epsilon_{i+1}\ fixed} - \frac{\partial A_x^{compression}}{\partial h}\right)\right)$

      iii. $\left.\frac{\partial \sigma_{min}}{\partial h}\right|_{\epsilon_{i+1}\ fixed} = \frac{\partial \sigma_y}{\partial h}\left(-\tau_2 + A_y^{compression}\right) + \sigma_y\left(-\left.\frac{\partial \tau_2}{\partial h}\right|_{\epsilon_{i+1}\ fixed} + \frac{\partial A_y^{compression}}{\partial h}\right)$

7. Set the stress derivative depending on the material state:

   (a) **if** $(\sigma_{max} < \sigma_{i+1}^{el})$
   $\left.\frac{\partial \sigma_{i+1}}{\partial h}\right|_{\epsilon_{i+1}\ fixed} = \left.\frac{\partial \sigma_{max}}{\partial h}\right|_{\epsilon_{i+1}\ fixed}$

   (b) **else if** $(\sigma_{min} > \sigma_{i+1}^{el})$
   $\left.\frac{\partial \sigma_{i+1}}{\partial h}\right|_{\epsilon_{i+1}\ fixed} = \left.\frac{\partial \sigma_{min}}{\partial h}\right|_{\epsilon_{i+1}\ fixed}$

   (c) **else**
   $\left.\frac{\partial \sigma_{i+1}}{\partial h}\right|_{\epsilon_{i+1}\ fixed} = \left.\frac{\partial \sigma_{i+1}^{el}}{\partial h}\right|_{\epsilon_{i+1}\ fixed}$

B.3 UNCONDITIONAL SENSITIVITY HISTORY VARIABLES FOR UNIAXIAL SMOOTHED BI-LINEAR STEEL MATERIAL

The algorithm referred to in Section 2.8.6 is presented here. From the expressions for the conditional stress derivative it is clear that the following sensitivity history variables are present: $\frac{\partial \sigma_i}{\partial h}$, $\frac{\partial \epsilon_i}{\partial h}$,

$\frac{\partial A_x^{tension}}{\partial h}$, $\frac{\partial A_x^{compression}}{\partial h}$, $\frac{\partial A_y^{tension}}{\partial h}$ and $\frac{\partial A_y^{compression}}{\partial h}$. These are updated in this "phase 2" of the sensitivity computations. The computations are now performed without the assumption of fixed strain. The needed quantity $\frac{\partial \epsilon_{i+1}}{\partial h}$ is passed from the element/section based on the displacement sensitivity vector $\mathbf{v}$, which is now available. It is assumed that the derivatives of the initial circle parameters R, $\tilde{A}_x$ and $\tilde{A}_y$ are available from the computations according to the previous section.

1. Store $\frac{\partial \epsilon_{i+1}}{\partial h}$

2. Compute the derivative of the elastic stress:
   $\frac{\partial \sigma_{i+1}^{el}}{\partial h} = \frac{\partial \sigma_i}{\partial h} + \frac{\partial E}{\partial h}\ (\epsilon_{i+1} - \epsilon_i) + E\ \left(\frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \epsilon_i}{\partial h}\right)$

3. Compute the derivative of the default minimum stress:
   $\frac{\partial \sigma_{min}}{\partial h} = \frac{\partial b}{\partial h}\ E\ \epsilon_{i+1} + b\ \frac{\partial E}{\partial h}\ \epsilon_{i+1} + bE\ \frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \sigma_y}{\partial h}\ (1-b) + \sigma_y\ \frac{\partial b}{\partial h}$

4. Compute the derivative of the default maximum stress:
   $\frac{\partial \sigma_{max}}{\partial h} = \frac{\partial b}{\partial h}\ E\ \epsilon_{i+1} + b\ \frac{\partial E}{\partial h}\ \epsilon_{i+1} + bE\ \frac{\partial \epsilon_{i+1}}{\partial h} + \frac{\partial \sigma_y}{\partial h}\ (1-b) - \sigma_y\ \frac{\partial b}{\partial h}$

5. Correct the derivatives of maximum/minimum stress if the strain is within the end points of the circular segment:

6. **if** $(B_x^{tension} < x < D_x^{tension})$

   (a) $\frac{\partial x}{\partial h} = \frac{\left(\frac{\partial \epsilon_{i+1}}{\partial h} E + \epsilon_{i+1} \frac{\partial E}{\partial h}\right)\sigma_y\,\eta - \epsilon_{i+1} E\left(\frac{\partial \sigma_y}{\partial h}\,\eta + \sigma_y\,\frac{\partial \eta}{\partial h}\right)}{(\sigma_y\,\eta)^2}$

   (b) $\frac{\partial \tau_1}{\partial h} = \frac{1}{\tau_1}\left(R\frac{\partial R}{\partial h} - (x - A_x^{tension})\left(\frac{\partial x}{\partial h} - \frac{\partial A_x^{tension}}{\partial h}\right)\right)$

   (c) $\frac{\partial \sigma_{max}}{\partial h} = \frac{\partial \sigma_y}{\partial h}\left(\tau_1 + A_y^{tension}\right) + \sigma_y\left(\frac{\partial \tau_1}{\partial h} + \frac{\partial A_y^{tension}}{\partial h}\right)$

7. **else if** $(D_x^{compression} < x < B_x^{compression})$

   (a) $\frac{\partial x}{\partial h}$ = as item (a) above.

   (b) $\frac{\partial \tau_2}{\partial h} = \frac{1}{\tau_2}\left(R\frac{\partial R}{\partial h} - (x - A_x^{compression})\left(\frac{\partial x}{\partial h} - \frac{\partial A_x^{compression}}{\partial h}\right)\right)$

   (c) $\frac{\partial \sigma_{min}}{\partial h} = \frac{\partial \sigma_y}{\partial h}\left(-\tau_2 + A_y^{compression}\right) + \sigma_y\left(-\frac{\partial \tau_2}{\partial h} + \frac{\partial A_y^{compression}}{\partial h}\right)$

8. Set stress derivative depending on material state:

   (a) **if** $(\sigma_{max} < \sigma_{i+1}^{el})$
   $\frac{\partial \sigma_{i+1}}{\partial h} = \frac{\partial \sigma_{max}}{\partial h}$

(b) **else if** $(\sigma_{min} > \sigma_{i+1}^{el})$

$\frac{\partial \sigma_{i+1}}{\partial h} = \frac{\partial \sigma_{min}}{\partial h}$

(c) **else**

$\frac{\partial \sigma_{i+1}}{\partial h} = \frac{\partial \sigma_{i+1}^{el}}{\partial h}$

9. The derivative of the circle coordinates are updated based on the current material state (derivatives of the original center coordinates are available from the previous section):

(a) $\frac{\partial \epsilon_{shift}}{\partial h} = \frac{\left(\frac{\partial E}{\partial h}\epsilon_{i+1} + E\frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \sigma_{i+1}}{\partial h}\right)E(1-b) - (E\epsilon_{i+1} - \sigma_{i+1})\left(\frac{\partial E}{\partial h}(1-b) - E\frac{\partial b}{\partial h}\right)}{E^2\,(1-b)^2}$

(b) $\frac{\partial x_{shift}}{\partial h} = \frac{\left(\frac{\partial E}{\partial h}\epsilon_{shift} + E\frac{\epsilon_{shift}}{\partial h}\right)\sigma_y\,\eta - E\,\epsilon_{shift}\left(\frac{\partial \sigma_y}{\partial h}\eta + \sigma_y\frac{\partial \eta}{\partial h}\right)}{(\sigma_y\,\eta)^2}$

(c) **if** $(state = 1)$

i. $\frac{\partial A_x^{tension}}{\partial h} = \frac{\partial \tilde{A}_x^{tension}}{\partial h} + \frac{\partial x_{shift}}{\partial h}$

ii. $\frac{\partial A_y^{tension}}{\partial h} = \frac{\partial \tilde{A}_y^{tension}}{\partial h} + \frac{\partial b}{\partial h}\,\eta\,x_{shift} + b\,\frac{\partial \eta}{\partial h}\,x_{shift} + b\,\eta\,\frac{\partial x_{shift}}{\partial h}$

iii. $\frac{\partial A_x^{compression}}{\partial h} = \frac{\partial \tilde{A}_x^{compression}}{\partial h} + \frac{\partial x_{shift}}{\partial h}$

iv. $\frac{\partial A_y^{compression}}{\partial h} = \frac{\partial \tilde{A}_y^{compression}}{\partial h} + \frac{\partial b}{\partial h}\,\eta\,x_{shift} + b\,\frac{\partial \eta}{\partial h}\,x_{shift} + b\,\eta\,\frac{\partial x_{shift}}{\partial h}$

(d) **else if** $(state = 2a)$

i. $\frac{\partial A_x^{compression}}{\partial h} = \frac{\partial \tilde{A}_x^{compression}}{\partial h} + \frac{\partial x_{shift}}{\partial h}$

ii. $\frac{\partial A_y^{compression}}{\partial h} = \frac{\partial \tilde{A}_y^{compression}}{\partial h} + \frac{\partial b}{\partial h}\,\eta\,x_{shift} + b\,\frac{\partial \eta}{\partial h}\,x_{shift} + b\,\eta\,\frac{\partial x_{shift}}{\partial h}$

(e) **else if** $(state = 2b)$

i. $\frac{\partial A_x^{tension}}{\partial h} = \frac{\partial \tilde{A}_x^{tension}}{\partial h} + \frac{\partial x_{shift}}{\partial h}$

ii. $\frac{\partial A_y^{tension}}{\partial h} = \frac{\partial \tilde{A}_y^{tension}}{\partial h} + \frac{\partial b}{\partial h}\,\eta\,x_{shift} + b\,\frac{\partial \eta}{\partial h}\,x_{shift} + b\,\eta\,\frac{\partial x_{shift}}{\partial h}$

(f) **else if** $(state = 3a)$

i. $\frac{\partial A_x^{compression}}{\partial h} = \frac{\partial \tilde{A}_x^{compression}}{\partial h} + \frac{\partial x_{shift}}{\partial h}$

ii. $\frac{\partial A_y^{compression}}{\partial h} = \frac{\partial \tilde{A}_y^{compression}}{\partial h} + \frac{\partial b}{\partial h}\,\eta\,x_{shift} + b\,\frac{\partial \eta}{\partial h}\,x_{shift} + b\,\eta\,\frac{\partial x_{shift}}{\partial h}$

iii. $\frac{\partial x_1}{\partial h} = \frac{\left(\frac{\partial E}{\partial h}\,\epsilon_{i+1} + E\,\frac{\partial \epsilon_{i+1}}{\partial h}\right)\sigma_y\,\eta - E\,\epsilon_{i+1}\left(\frac{\partial \sigma_y}{\partial h}\,\eta + \sigma_y\,\frac{\partial \eta}{\partial h}\right)}{(\sigma_y\,\eta)^2}$

iv. $\frac{\partial y_1}{\partial h} = \frac{\frac{\partial \sigma_{i+1}}{\partial h}\sigma_y - \sigma_{i+1}\frac{\partial \sigma_y}{\partial h}}{\sigma_y^2}$

v. $\frac{\partial x_2}{\partial h} = \frac{\left(\frac{\partial b}{\partial h}\eta\,y_1 + b\,\eta\frac{\partial \eta}{\partial h}y_1 + b\,\eta\frac{\partial y_1}{\partial h} + \frac{\partial x_1}{\partial h} - \frac{\partial b}{\partial h}\eta(1-b) - b\frac{\partial \eta}{\partial h}(1-b) + b\,\eta\frac{\partial b}{\partial h}\right)\left((b\,\eta)^2+1\right)}{\left((b\,\eta)^2+1\right)^2}$
$- \frac{(b\,\eta\,y_1 + x_1 - b\,\eta(1-b))\left(2\,b\,\eta\left(\frac{\partial b}{\partial h}\,\eta + b\,\frac{\partial \eta}{\partial h}\right)\right)}{\left((b\,\eta)^2+1\right)^2}$

vi. $\frac{\partial y_2}{\partial h} = \frac{\partial b}{\partial h}\,\eta\,x_2 + b\,\frac{\partial \eta}{\partial h}\,x_2 + b\,\eta\,\frac{\partial x_2}{\partial h} - \frac{\partial b}{\partial h}$

vii. $\frac{\partial l_{1,2}}{\partial h} = \frac{1}{\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}}\left((x_1-x_2)\left(\frac{\partial x_1}{\partial h} - \frac{\partial x_2}{\partial h}\right) + (y_1-y_2)\left(\frac{\partial y_1}{\partial h} - \frac{\partial y_2}{\partial h}\right)\right)$

viii. $\frac{\partial l_{2,3}}{\partial h} = \frac{1}{\sqrt{R^2-(l_{1,2}-R)^2}}\left(R\frac{\partial R}{\partial h} - (l_{1,2}-R)\left(\frac{\partial l_{1,2}}{\partial h} - \frac{\partial R}{\partial h}\right)\right)$

ix. $\frac{\partial l_{2,3,y}}{\partial h} = \frac{\left(\frac{\partial b}{\partial h}\,\eta\, l_{2,3} + b\,\frac{\partial \eta}{\partial h}\, l_{2,3} + b\,\eta\,\frac{\partial l_{2,3}}{\partial h}\right)\sqrt{1+b^2\eta^2} - b\,\eta\, l_{2,3}\frac{1}{\sqrt{1+b^2\eta^2}}b\,\eta\left(\frac{\partial b}{\partial h}\,\eta + b\,\frac{\partial \eta}{\partial h}\right)}{1+b^2\eta^2}$

x. $\frac{\partial l_{2,3,x}}{\partial h} = \frac{\frac{\partial l_{2,3}}{\partial h}\sqrt{1+b^2\eta^2} - l_{2,3}\frac{1}{\sqrt{1+b^2\eta^2}}b\,\eta\left(\frac{\partial b}{\partial h}\,\eta + b\,\frac{\partial \eta}{\partial h}\right)}{1+b^2\eta^2}$

xi. $\frac{\partial \Delta R_x}{\partial h} = \frac{\frac{\partial R}{\partial h}\sqrt{\frac{1}{b^2\eta^2}+1} - R\frac{1}{2\sqrt{\frac{1}{b^2\eta^2}+1}}\frac{1}{(b\,\eta)^3}\left(\frac{\partial b}{\partial h}\,\eta + b\,\frac{\partial \eta}{\partial h}\right)}{\frac{1}{(b\,\eta)^2}+1}$

xii. $\frac{\partial \Delta R_y}{\partial h} = \frac{\frac{\partial R}{\partial h}b\,\eta\sqrt{\frac{1}{b^2\eta^2}+1} - R\left(\frac{\partial b}{\partial h}\,\eta\sqrt{\frac{1}{b^2\eta^2}+1} + b\,\frac{\partial \eta}{\partial h}\sqrt{\frac{1}{b^2\eta^2}+1} + b\,\eta\frac{1}{2\sqrt{\frac{1}{b^2\eta^2}+1}}\frac{1}{(b\,\eta)^3}\left(\frac{\partial b}{\partial h}\,\eta + b\,\frac{\partial \eta}{\partial h}\right)\right)}{(b\,\eta)^2\left(\frac{1}{(b\,\eta)^2}+1\right)}$

xiii. $\frac{\partial A_x^{tension}}{\partial h} = \frac{\partial x_2}{\partial h} + \frac{\partial l_{2,3,x}}{\partial h} + \frac{\partial \Delta R_x}{\partial h}$

xiv. $\frac{\partial A_y^{tension}}{\partial h} = \frac{\partial y_2}{\partial h} + \frac{\partial l_{2,3,y}}{\partial h} - \frac{\partial \Delta R_y}{\partial h}$

(g) **else if** $(state = 3b)$

i. $\frac{\partial A_x^{tension}}{\partial h} = \frac{\partial \tilde{A}_x^{tension}}{\partial h} + \frac{\partial x_{shift}}{\partial h}$

ii. $\frac{\partial A_y^{tension}}{\partial h} = \frac{\partial \tilde{A}_y^{tension}}{\partial h} + \frac{\partial b}{\partial h}\,\eta\, x_{shift} + b\,\frac{\partial \eta}{\partial h}\, x_{shift} + b\,\eta\,\frac{\partial x_{shift}}{\partial h}$

iii. $\frac{\partial x_1}{\partial h} = \frac{\left(\frac{\partial E}{\partial h}\,\epsilon_{i+1} + E\,\frac{\partial \epsilon_{i+1}}{\partial h}\right)\sigma_y\,\eta - E\,\epsilon_{i+1}\left(\frac{\partial \sigma_y}{\partial h}\,\eta + \sigma_y\,\frac{\partial \eta}{\partial h}\right)}{(\sigma_y\,\eta)^2}$

iv. $\frac{\partial y_1}{\partial h} = \frac{\frac{\partial \sigma_{i+1}}{\partial h}\sigma_y - \sigma_{i+1}\frac{\partial \sigma_y}{\partial h}}{\sigma_y^2}$

v. $\frac{\partial x_2}{\partial h} = \frac{\left(\frac{\partial b}{\partial h}\eta\, y_1 + b\frac{\partial \eta}{\partial h}y_1 + b\,\eta\frac{\partial y_1}{\partial h} + \frac{\partial x_1}{\partial h} + \frac{\partial b}{\partial h}\eta(1-b) + b\frac{\partial \eta}{\partial h}(1-b) - b\,\eta\frac{\partial b}{\partial h}\right)\left((b\,\eta)^2+1\right)}{\left((b\,\eta)^2+1\right)^2}$
$- \frac{(b\,\eta\, y_1 + x_1 + b\,\eta(1-b))\left(2\,b\,\eta\left(\frac{\partial b}{\partial h}\,\eta + b\,\frac{\partial \eta}{\partial h}\right)\right)}{\left((b\,\eta)^2+1\right)^2}$

vi. $\frac{\partial y_2}{\partial h} = \frac{\partial b}{\partial h}\,\eta\, x_2 + b\,\frac{\partial \eta}{\partial h}\, x_2 + b\,\eta\,\frac{\partial x_2}{\partial h} + \frac{\partial b}{\partial h}$

vii. $\frac{\partial l_{1,2}}{\partial h} = \frac{1}{\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}}\left((x_1-x_2)\left(\frac{\partial x_1}{\partial h} - \frac{\partial x_2}{\partial h}\right) + (y_1-y_2)\left(\frac{\partial y_1}{\partial h} - \frac{\partial y_2}{\partial h}\right)\right)$

viii. $\frac{\partial l_{2,3}}{\partial h} = \frac{1}{\sqrt{R^2-(l_{1,2}-R)^2}}\left(R\frac{\partial R}{\partial h} - (l_{1,2}-R)\left(\frac{\partial l_{1,2}}{\partial h} - \frac{\partial R}{\partial h}\right)\right)$

ix. $\frac{\partial l_{2,3,y}}{\partial h} = \frac{\left(\frac{\partial b}{\partial h}\,\eta\, l_{2,3} + b\,\frac{\partial \eta}{\partial h}\, l_{2,3} + b\,\eta\,\frac{\partial l_{2,3}}{\partial h}\right)\sqrt{1+b^2\eta^2} - b\,\eta\, l_{2,3}\frac{1}{\sqrt{1+b^2\eta^2}}b\,\eta\left(\frac{\partial b}{\partial h}\,\eta + b\,\frac{\partial \eta}{\partial h}\right)}{1+b^2\eta^2}$

x. $\frac{\partial l_{2,3,x}}{\partial h} = \frac{\frac{\partial l_{2,3}}{\partial h}\sqrt{1+b^2\eta^2} - l_{2,3}\frac{1}{\sqrt{1+b^2\eta^2}}b\,\eta\left(\frac{\partial b}{\partial h}\,\eta + b\,\frac{\partial \eta}{\partial h}\right)}{1+b^2\eta^2}$

xi. $\frac{\partial \Delta R_x}{\partial h} = \frac{\frac{\partial R}{\partial h}\sqrt{\frac{1}{b^2\eta^2}+1} - R\frac{1}{2\sqrt{\frac{1}{b^2\eta^2}+1}}\frac{1}{(b\,\eta)^3}\left(\frac{\partial b}{\partial h}\,\eta + b\,\frac{\partial \eta}{\partial h}\right)}{\frac{1}{(b\,\eta)^2}+1}$

xii. $\frac{\partial \Delta R_y}{\partial h} = \frac{\frac{\partial R}{\partial h}b\,\eta\sqrt{\frac{1}{b^2\eta^2}+1} + R\left(\frac{\partial b}{\partial h}\,\eta\sqrt{\frac{1}{b^2\eta^2}+1} + b\,\frac{\partial \eta}{\partial h}\sqrt{\frac{1}{b^2\eta^2}+1} + b\,\eta\frac{1}{2\sqrt{\frac{1}{b^2\eta^2}+1}}\frac{1}{(b\,\eta)^3}\left(\frac{\partial b}{\partial h}\,\eta + b\,\frac{\partial \eta}{\partial h}\right)\right)}{(b\,\eta)^2\left(\frac{1}{(b\,\eta)^2}+1\right)}$

xiii. $\frac{\partial A_x^{compression}}{\partial h} = \frac{\partial x_2}{\partial h} - \frac{\partial l_{2,3,x}}{\partial h} - \frac{\partial \Delta R_x}{\partial h}$

xiv. $\frac{\partial A_y^{compression}}{\partial h} = \frac{\partial y_2}{\partial h} - \frac{\partial l_{2,3,y}}{\partial h} + \frac{\partial \Delta R_y}{\partial h}$

(h) **else if** $(state = 4a)$

i. $\frac{\partial A_x^{compression}}{\partial h} = \frac{\partial \tilde{A}_x^{compression}}{\partial h} + \frac{\partial x_{shift}}{\partial h}$

ii. $\frac{\partial A_y^{compression}}{\partial h} = \frac{\partial \tilde{A}_y^{compression}}{\partial h} + \frac{\partial b}{\partial h}\,\eta\,x_{shift} + b\,\frac{\partial \eta}{\partial h}\,x_{shift} + b\,\eta\,\frac{\partial x_{shift}}{\partial h}$

(i) **else if** $(state = 4b)$

i. $\frac{\partial A_x^{tension}}{\partial h} = \frac{\partial \tilde{A}_x^{tension}}{\partial h} + \frac{\partial x_{shift}}{\partial h}$

ii. $\frac{\partial A_y^{tension}}{\partial h} = \frac{\partial \tilde{A}_y^{tension}}{\partial h} + \frac{\partial b}{\partial h}\,\eta\,x_{shift} + b\,\frac{\partial \eta}{\partial h}\,x_{shift} + b\,\eta\,\frac{\partial x_{shift}}{\partial h}$

B.4 HISTORY VARIABLES FOR UNIAXIAL SMOOTHED CONCRETE MATERIAL

The history variables of the smoothed concrete material are ϵ_{min}, ϵ_{end} and k_u. The following algorithm is found in the original "Concrete01" material in OpenSees:

1. $\epsilon_{min} = \epsilon_{i+1}$
2. **if** $(\epsilon_{i+1} < epscu)$
   (a) $\tilde{\epsilon} = \epsilon_{cu}$
3. **else**
   (a) $\tilde{\epsilon} = \epsilon_{i+1}$
4. $\eta = \frac{\tilde{\epsilon}}{epsc0}$
5. **if** $(\eta < 2)$
   (a) $ratio = 0.145\,\eta^2 + 0.13\,\eta$
6. **else**
   (a) $r = 0.707\,(\eta - 2) + 0.834$
7. $\tau_1 = \epsilon_{i+1} - r\,\epsilon_{c0}$
8. $\tau_2 = \frac{\sigma_{i+1}\,\epsilon_{c0}}{2\,fpc}$
9. **if** $(\tau_1 = 0)$
   (a) $k_u = \frac{2\,f'_c}{\epsilon_{c0}}$

10. **else if** $(\tau_1 < \tau_2)$

    (a) $\epsilon_{end} = \epsilon_{i+1} - \tau_1$

    (b) $k_u = \frac{\sigma_{i+1}}{\tau_1}$

11. **else**

    (a) $\epsilon_{end} = \epsilon_{i+1} - \tau_2$

    (b) $k_u = \frac{2\, f'_c}{\epsilon_{c0}}$

B.5 BACKBONE CURVE FOR THE UNIAXIAL SMOOTHED CONCRETE MATERIAL

The backbone curve employed in this material model is similar to the one used in the ordinary "Concrete01" material in OpenSees. It is, however, distinguished by having a smooth transition between the different regions of the backbone curve:

1. **if** $(\epsilon_{i+1} > \epsilon_{c0})$; point on the parabola:

    (a) $\sigma_{i+1} = f'_c \left(2\, \frac{\epsilon_{i+1}}{\epsilon_{c0}} - \left(\frac{\epsilon_{i+1}}{\epsilon_{c0}} \right)^2 \right)$

    (b) $k_T = 2\, \frac{f'_c}{\epsilon_{c0}} \left(1 - \frac{\epsilon_{i+1}}{\epsilon_{c0}} \right)$

2. **else if** $(\epsilon_{i+1} > \epsilon_{cu})$; point on the third-order polynomial used to smoothen the backbone curve:

    (a) $a = -2 \frac{f'_{cu} - f'_c}{(\epsilon_{cu} - \epsilon_{c0})^3}, \quad b = 3 \frac{f'_{cu} - f'_c}{(\epsilon_{cu} - \epsilon_{c0})^2}$

    (b) $\sigma_{i+1} = a\, (\epsilon_{i+1} - \epsilon_{c0})^3 + b\, (\epsilon_{i+1} - \epsilon_{c0})^2 + f'_c$

    (c) $k_T = 3\, a\, (\epsilon_{i+1} - \epsilon_{c0})^2 + 2\, b\, (\epsilon_{i+1} - \epsilon_{c0})$

3. **else**; on the horizontal line:

    (a) $\sigma_{i+1} = f'_{cu}$

    (b) $k_T = 0$

B.6 SMOOTHING LINE BETWEEN TWO POINTS FOR UNIAXIAL SMOOTHED CONCRETE MATERIAL

A general third-order interpolating polynomial between two points is established. This is used at several material states in the incremental algorithm presented for the uniaxial smoothed concrete material model.

1. Assume the strain, stress and tangent at points 1 and 2 are available (see Figure 2.26): ϵ_1, σ_1, k_1, ϵ_2, σ_2 and k_2.

2. Polynomial coefficient: $a = \frac{k_1\,\epsilon_2 - 2\,\sigma_2 + k_2\,\epsilon_2}{\epsilon_2^3}$

3. Polynomial coefficient: $b = \frac{3\sigma_2 - 2k_1\,\epsilon_2 - k_2\,\epsilon_2}{\epsilon_2^2}$

4. Polynomial coefficient: $c = k_1$

5. Smooth stress curve: $\sigma_{i+1} = a\,(\epsilon_{i+1} - \epsilon_1)^3 + b\,(\epsilon_{i+1} - \epsilon_1)^2 + c\,(\epsilon_{i+1} - \epsilon_1) + \sigma_1$

6. Corresponding tangent: $k_T = 3\,a\,(\epsilon_{i+1} - \epsilon_1)^2 + 2\,b\,(\epsilon_{i+1} - \epsilon_1) + c$

B.7 DERIVATIVE OF BACKBONE CURVE FOR UNIAXIAL SMOOTHED CONCRETE MATERIAL

The equations to obtain $\frac{\partial \sigma_{i+1}}{\partial h}$ when the stress state is found to be on the backbone curve are derived in this section. Note that the quantity $\frac{\partial \epsilon_{i+1}}{\partial h}$ should be set to zero when the conditional stress derivative is sought in "phase 1." When computing sensitivity history variables in "phase 2," this quantity is available to the material object from the preceding computation of the displacement sensitivity vector. The derivative of the stiffness is derived because it is used in the following section.

1. **if** $(\epsilon_{i+1} > \epsilon_{c0})$; point on the parabola:

   (a) $\frac{\partial \sigma_{i+1}}{\partial h} = \frac{2\left(\frac{\partial f_c'}{\partial h}\epsilon_{i+1} + f_c'\frac{\partial \epsilon_{i+1}}{\partial h}\right) - \frac{\partial \epsilon_{c0}}{\partial h}}{\epsilon_{c0}^2} - \frac{\partial f_c'}{\partial h}\left(\frac{\epsilon_{i+1}}{\epsilon_{c0}}\right)^2 - 2f_c'\left(\frac{\epsilon_{i+1}}{\epsilon_{c0}}\right)\frac{\frac{\partial \epsilon_{i+1}}{\partial h}\epsilon_{c0} - \epsilon_{i+1}\frac{\partial \epsilon_{c0}}{\partial h}}{\epsilon_{c0}^2}$

   (b) $\frac{\partial k_T}{\partial h} = 2\,\frac{\frac{\partial f_c'}{\partial h}\,\epsilon_{c0} - f_c'\,\frac{\partial \epsilon_{c0}}{\partial h}}{\epsilon_{c0}^2}\left(1 - \frac{\epsilon_{i+1}}{\epsilon_{c0}}\right) - 2\,\frac{f_c'}{\epsilon_{c0}}\left(\frac{\frac{\partial \epsilon_{i+1}}{\partial h}\,\epsilon_{c0} - \epsilon_{i+1}\,\frac{\partial \epsilon_{c0}}{\partial h}}{\epsilon_{c0}^2}\right)$

2. **else if** $(\epsilon_{i+1} > \epsilon_{cu})$; point on the third-order polynomial used to smoothen the backbone curve:

   (a) $\frac{\partial a}{\partial h} = -2\frac{\left(\frac{\partial f_{cu}'}{\partial h} - \frac{\partial f_c'}{\partial h}\right)(\epsilon_{cu} - \epsilon_{c0})^3 - 3\,(f_{cu}' - f_c')(\epsilon_{cu} - \epsilon_{c0})^2\left(\frac{\partial \epsilon_{cu}}{\partial h} - \frac{\partial \epsilon_{c0}}{\partial h}\right)}{(\epsilon_{cu} - \epsilon_{c0})^6}$

(b) $\frac{\partial b}{\partial h} = 3\frac{\left(\frac{\partial f'_{cu}}{\partial h} - \frac{\partial f'_c}{\partial h}\right)(\epsilon_{cu}-\epsilon_{c0})^2 - 2\,(f'_{cu}-f'_c)(\epsilon_{cu}-\epsilon_{c0})\left(\frac{\partial \epsilon_{cu}}{\partial h} - \frac{\partial \epsilon_{c0}}{\partial h}\right)}{(\epsilon_{cu}-\epsilon_{c0})^4}$

(c) $\frac{\partial \sigma_{i+1}}{\partial h} = \frac{\partial a}{\partial h}\left(\epsilon_{i+1}-\epsilon_{c0}\right)^3 + 3\,a\left(\epsilon_{i+1}-\epsilon_{c0}\right)^2\left(\frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \epsilon_{c0}}{\partial h}\right)$
$+ \frac{\partial b}{\partial h}\left(\epsilon_{i+1}-\epsilon_{c0}\right)^2 + 2\,b\left(\epsilon_{i+1}-\epsilon_{c0}\right)\left(\frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \epsilon_{c0}}{\partial h}\right) + f'_c$

(d) $\frac{\partial k_T}{\partial h} = 3\,\frac{\partial a}{\partial h}\left(\epsilon_{i+1}-\epsilon_{c0}\right)^2 + 6\,a\left(\epsilon_{i+1}-\epsilon_{c0}\right)\left(\frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \epsilon_{c0}}{\partial h}\right) + 2\,\frac{\partial b}{\partial h}\left(\epsilon_{i+1}-\epsilon_{c0}\right) + 2\,b\left(\frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \epsilon_{c0}}{\partial h}\right)$

3. **else**; on the horizontal line:

(a) $\frac{\partial \sigma_{i+1}}{\partial h} = \frac{\partial f'_{cu}}{\partial h}$

(b) $\frac{\partial k_T}{\partial h} = 0$

B.8 DERIVATIVE OF SMOOTHING LINE BETWEEN TWO POINTS FOR UNIAXIAL SMOOTHED CONCRETE MATERIAL

As in the algorithm to determine the smoothing line itself, we assume that the derivatives of the interpolation points (and the respective stiffnesses), namely ϵ_1, σ_1, k_1, ϵ_2, σ_2 and k_2, are available as history variables.

1. Derivative of the polynomial coefficient:
$\frac{\partial a}{\partial h} = \frac{\left(\frac{\partial k_1}{\partial h}\,\epsilon_2 + k_1\,\frac{\partial \epsilon_2}{\partial h} - 2\,\frac{\partial \sigma_2}{\partial h} + \frac{\partial k_2}{\partial h}\,\epsilon_2 + k_2\,\frac{\partial \epsilon_2}{\partial h}\right)\epsilon_2^3 - 3\,(k_1\,\epsilon_2 - 2\,\sigma_2 + k_2\,\epsilon_2)\epsilon_2^2\,\frac{\partial \epsilon_2}{\partial h}}{\epsilon_2^6}$

2. Derivative of the polynomial coefficient:
$\frac{\partial b}{\partial h} = \frac{\left(3\,\frac{\partial \sigma_2}{\partial h} - 2\,\frac{\partial k_1}{\partial h}\epsilon_2 - 2k_1\,\frac{\partial \epsilon_2}{\partial h} - \frac{\partial k_2}{\partial h}\epsilon_2 - k_2\,\frac{\partial \epsilon_2}{\partial h}\right)\epsilon_2^2 - 2\,(3\sigma_2 - 2k_1\,\epsilon_2 - k_2\,\epsilon_2)\epsilon_2\,\frac{\partial \epsilon_2}{\partial h}}{\epsilon_2^4}$

3. Derivative of the polynomial coefficient:
$\frac{\partial c}{\partial h} = \frac{\partial k_1}{\partial h}$

4. Derivative of the smooth stress curve:
$\frac{\partial \sigma_{i+1}}{\partial h} = \frac{\partial a}{\partial h}\left(\epsilon_{i+1}-\epsilon_1\right)^3$
$+ \quad 3\,a\left(\epsilon_{i+1}-\epsilon_1\right)^2\left(\frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \epsilon_1}{\partial h}\right) \quad + \quad \frac{\partial b}{\partial h}\left(\epsilon_{i+1}-\epsilon_1\right)^2 \quad + \quad 2\,b\left(\epsilon_{i+1}-\epsilon_1\right)\left(\frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \epsilon_1}{\partial h}\right)$
$+ \frac{\partial c}{\partial h}\left(\epsilon_{i+1}-\epsilon_1\right) + c\left(\frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \epsilon_1}{\partial h}\right) + \frac{\partial \sigma_1}{\partial h}$

5. Derivative of the corresponding tangent:
$\frac{\partial k_T}{\partial h} = 3\,\frac{\partial a}{\partial h}\left(\epsilon_{i+1}-\epsilon_1\right)^2 + 6\,a\left(\epsilon_{i+1}-\epsilon_1\right)\left(\frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \epsilon_1}{\partial h}\right)$
$+ 2\,\frac{\partial b}{\partial h}\left(\epsilon_{i+1}-\epsilon_1\right) + 2\,b\left(\frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \epsilon_1}{\partial h}\right) + \frac{\partial c}{\partial h}$

B.9 SENSITIVITY HISTORY VARIABLES FOR UNIAXIAL SMOOTHED CONCRETE MATERIAL

Computation of the sensitivity history variables $\frac{\partial \epsilon_{min}}{\partial h}$, $\frac{\partial \epsilon_{end}}{\partial h}$ and $\frac{\partial k_u}{\partial h}$ is as follows:

1. $\frac{\partial \epsilon_{min}}{\partial h} = \frac{\partial \epsilon_{i+1}}{\partial h}$
2. **if** $(\epsilon_{i+1} < \epsilon_{cu})$
   (a) $\tilde{\epsilon} = \epsilon_{cu}$
   (b) $\frac{\partial \tilde{\epsilon}}{\partial h} = \frac{\partial \epsilon_{cu}}{\partial h}$
3. **else**
   (a) $\tilde{\epsilon} = \epsilon_{i+1}$
   (b) $\frac{\partial \tilde{\epsilon}}{\partial h} = \frac{\partial \epsilon_{i+1}}{\partial h}$
4. $\eta = \frac{\tilde{\epsilon}}{\epsilon_{c0}}$
5. $\frac{\partial \eta}{\partial h} = \frac{\frac{\partial \tilde{\epsilon}}{\partial h}\epsilon_{c0} - \tilde{\epsilon}\frac{\partial \epsilon_{c0}}{\partial h}}{\epsilon_{c0}^2}$
6. **if** $(\eta < 2)$
   (a) $r = 0.145\,\eta^2 + 0.13\,\eta$
   (b) $\frac{\partial r}{\partial h} = 0.29\ \eta\ \frac{\partial \eta}{\partial h} + 0.13\frac{\partial \eta}{\partial h}$
7. **else**
   (a) $r = 0.707\ (\eta - 2) + 0.834$
   (b) $\frac{\partial r}{\partial h} = 0.707\ \frac{\partial \eta}{\partial h}$
8. $\tau_1 = \epsilon_{i+1} - r\ \epsilon_{c0}$
9. $\tau_2 = \frac{\sigma_{i+1}\,\epsilon_{c0}}{2\,f_c'}$
10. $\frac{\partial \tau_1}{\partial h} = \frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial r}{\partial h}\ \epsilon_{c0} - r\ \frac{\partial \epsilon_{c0}}{\partial h}$
11. $\frac{\partial \tau_2}{\partial h} = \frac{\left(\frac{\partial \sigma_{i+1}}{\partial h}\epsilon_{c0} + \sigma_{i+1}\frac{\partial \epsilon_{c0}}{\partial h}\right) 2\,f_c' - 2\,\sigma_{i+1}\epsilon_{c0}\frac{\partial f_c'}{\partial h}}{4\,f_c'^2}$
12. **if** $(\tau_1 = 0)$

(a) $\frac{\partial k_u}{\partial h} = \frac{2\,\epsilon_{c0}\,\frac{\partial f_c'}{\partial h} - 2\,f_c'\,\frac{\partial \epsilon_{c0}}{\partial h}}{\epsilon_{c0}^2}$

13. **else if** $(\tau_1 < \tau_2)$

(a) $\frac{\partial \epsilon_{end}}{\partial h} = \frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \tau_1}{\partial h}$

(b) $\frac{\partial k_u}{\partial h} = \frac{\frac{\partial \sigma_{i+1}}{\partial h}\,\tau_1 - \sigma_{i+1}\,\frac{\partial \tau_1}{\partial h}}{\tau_1^2}$

14. **else**

(a) $\frac{\partial \epsilon_{end}}{\partial h} = \frac{\partial \epsilon_{i+1}}{\partial h} - \frac{\partial \tau_2}{\partial h}$

(b) $\frac{\partial k_u}{\partial h} = \frac{2\,\epsilon_{c0}\,\frac{\partial f_c'}{\partial h} - 2\,f_c'\,\frac{\partial \epsilon_{c0}}{\partial h}}{\epsilon_{c0}^2}$

It is seen in the above equations that $\frac{\partial \sigma_{i+1}}{\partial h}$ is needed. Hence, as a part of the procedure of computing sensitivity history variables, the unconditional stress sensitivity must be computed. This is done according to the equations outlined in the previous two sections, except that the quantity $\frac{\partial \epsilon_{i+1}}{\partial h}$ now is non-zero.

Appendix C: Probability Distributions

Table C.1: The normal probability distribution. The PDF for the standard normal distribution is usually denoted $\varphi(x)$, while the corresponding CDF is denoted $\Phi(x)$.

Normal $N(\mu, \sigma)$	
Probability Density Function (PDF)	$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$
Variable and parameter restrictions	$\sigma > 0$
Cumulative Distribution Function (CDF)	$F(x) = \Phi\left(\frac{x-\mu}{\sigma}\right)$
Inverse CDF	$x(F) = \mu + \sigma\,\Phi(F)^{-1}$
Mean and standard deviation in terms of the distribution parameters	Mean $= \mu$ Standard deviation $= \sigma$

Table C.2: The lognormal probability distribution.

Lognormal $LN(\lambda, \zeta)$	
Probability Density Function (PDF)	$f(x) = \frac{1}{\sqrt{2\pi}\zeta x} \exp\left[-\frac{1}{2}\left(\frac{\ln x-\lambda}{\zeta}\right)^2\right]$
Variable and parameter restrictions	$x > 0, \quad \zeta > 0$
Cumulative Distribution Function (CDF)	$F(x) = \Phi\left(\frac{\ln x-\lambda}{\zeta}\right)$
Inverse CDF	$x(F) = \exp\left[\Phi^{-1}(F)\zeta + \lambda\right]$
Mean and standard deviation in terms of distribution parameters	Mean $= \exp\left(\lambda + 0.5\,\zeta^2\right)$ St.dev. $= \exp\left(\lambda + 0.5\,\zeta^2\right)\sqrt{\exp\left(\zeta^2\right) - 1}$

Table C.3: The negative lognormal probability distribution.

Negative lognormal $NLN(\lambda, \zeta)$	
Probability Density Function (PDF)	$f(x) = \frac{1}{\sqrt{2\pi}\zeta x} \exp\left[-\frac{1}{2}\left(\frac{\ln(\lvert x\rvert)-\lambda}{\zeta}\right)^2\right]$
Variable and parameter restrictions	$x < 0, \quad \zeta > 0$
Cumulative Distribution Function (CDF)	$F(x) = 1 - \Phi\left(\frac{\ln(\lvert x\rvert)-\lambda}{\zeta}\right)$
Inverse CDF	$x(F) = -\exp\left[\Phi^{-1}(1-F)\zeta + \lambda\right]$
Mean and standard deviation in terms of distribution parameters	Mean $= \exp(\lambda + 0.5\,\zeta^2)$ St.dev. $= \exp(\lambda + 0.5\,\zeta^2)\sqrt{\exp(\zeta^2) - 1}$

Table C.4: The exponential probability distribution.

Exponential $Exp(\lambda)$	
Probability Density Function (PDF)	$f(x) = \lambda \exp(-\lambda x)$
Variable and parameter restrictions	$x > 0, \quad \lambda > 0$
Cumulative Distribution Function (CDF)	$F(x) = 1 - \exp(-\lambda x)$
Inverse CDF	$x(F) = \frac{-\ln(1-F)}{\lambda}$
Mean and standard deviation in terms of distribution parameters	Mean $= 1/\lambda$ Standard deviation $= 1/\lambda$

Table C.5: The shifted exponential probability distribution.

Shifted Exponential $Exp(\lambda, x_0)$	
Probability Density Function (PDF)	$f(x) = \lambda \exp(\lambda(x - x_0))$
Variable and parameter restrictions	$\lambda > 0, \quad x > x_0$
Cumulative Distribution Function (CDF)	$F(x) = 1 - \exp(-\lambda(x - x_0))$
Inverse CDF	$x(F) = \frac{\lambda x_0 - \ln(1-F)}{\lambda}$
Mean and standard deviation in terms of distribution parameters	Mean $= x_0 + 1/\lambda$ Standard deviation $= 1/\lambda$

Table C.6: The Rayleigh probability distribution.

Rayleigh $Ray(u)$	
Probability Density Function (PDF)	$f(x) = \frac{2x}{u^2} \exp\left[-\left(\frac{x}{u}\right)^2\right]$
Variable and parameter restrictions	$x > 0, \quad u > 0$
Cumulative Distribution Function (CDF)	$F(x) = 1 - \exp\left[-\left(\frac{x}{u}\right)^2\right]$
Inverse CDF	$x(F) = u\sqrt{-\ln(1-F)}$
Mean and standard deviation in terms of distribution parameters	Mean $= \frac{u}{2}\sqrt{\pi}$ Standard deviation $= \frac{u}{2}\sqrt{4-\pi}$

Table C.7: The shifted Rayleigh probability distribution.

Shifted Rayleigh $Ray(u, x_0)$	
Probability Density Function (PDF)	$f(x) = \frac{2(x-x_0)}{u^2} \exp\left[-\left(\frac{(x-x_0)}{u}\right)^2\right]$
Variable and parameter restrictions	$x > x_0, \quad u > 0$
Cumulative Distribution Function (CDF)	$F(x) = 1 - \exp\left[-\left(\frac{(x-x_0)}{u}\right)^2\right]$
Inverse CDF	$x(F) = x_0 + u\sqrt{-\ln(1-F)}$
Mean and standard deviation in terms of distribution parameters	Mean $= x_0 + 0.5u\sqrt{\pi}$ Standard deviation $= 0.5u\sqrt{4-\pi}$

Table C.8: The uniform probability distribution.

Uniform $U(a, b)$	
Probability Density Function (PDF)	$f(x) = \frac{1}{b-a}$
Variable and parameter restrictions	$a < x < b, \quad a < b$
Cumulative Distribution Function (CDF)	$F(x) = \frac{x-a}{b-a}$
Inverse CDF	$x(F) = F\ b - F\ a + a$
Mean and standard deviation in terms of distribution parameters	Mean $= (a+b)/2$ Standard deviation $= \left(\sqrt{3}(b-a)\right)/6$

Table C.9: The gamma probability distribution. $\Gamma(\,)$ is the gamma function and $\Gamma(\ ,\ )$ is the so-called incomplete gamma function.

Gamma $Gam(k, \lambda)$	
Probability Density Function (PDF)	$f(x) = \frac{\lambda(\lambda\, x)^{k-1} \exp(\lambda\, x)}{\Gamma(k)}$
Variable and parameter restrictions	$x > 0, \quad \lambda > 0, \quad k > 0$
Cumulative Distribution Function (CDF)	$F(x) = \frac{\Gamma(k, \lambda\, x)}{\Gamma(k)}$
Inverse CDF	$x(F) = (noclosedform)$
Mean and standard deviation in terms of distribution parameters	Mean $= k/\lambda$ Standard deviation $= \sqrt{k}/\lambda$

Table C.10: The beta probability distribution. $B(q, r)$ is the beta function defined as $\Gamma(q)\,\Gamma(r)/\Gamma(q+r)$.

Beta $Bet(a, b, q, r)$	
Probability Density Function (PDF)	$f(x) = \frac{(x-a)^{q-1}\,(b-x)^{r-1}}{B(q,r)\,(b-a)^{q+r-1}}$
Variable and parameter restrictions	$a < x < b, \quad q > 0, \quad r > 0$
Cumulative Distribution Function (CDF)	$F(x) = (noclosedform)$
Inverse CDF	$x(F) = (noclosedform)$
Mean and standard deviation in terms of distribution parameters	Mean $= (a\, r + b\, q)/(q + r)$ Standard deviation $= \frac{b-a}{q+r}\sqrt{\frac{q\, r}{q+r+1}}$

Table C.11: The type I largest value probability distribution (identical to the Gumbel distribution).

Type I Largest Value and Gumbel $Gum(u, \alpha)$	
Probability Density Function (PDF)	$f(x) = \alpha \exp\left[-\alpha(x-u) - \exp\left[-\alpha(x-u)\right]\right]$
Variable and parameter restrictions	$\alpha > 0$
Cumulative Distribution Function (CDF)	$F(x) = \exp\left[-\exp\left(-\alpha(x-u)\right)\right]$
Inverse CDF	$x(F) = \frac{\alpha u - \ln(-\ln(F))}{\alpha}$
Mean and standard deviation in terms of distribution parameters	Mean $= u + \gamma/\alpha$ where $\gamma = 0.5772156649$ Standard deviation $= \pi/\left(\alpha\sqrt{6}\right)$

Table C.12: The type I smallest value probability distribution.

Type I Smallest Value $TypeISmallestValue(u, \alpha)$	
Probability Density Function (PDF)	$f(x) = \alpha \exp(\alpha(x-u) - exp(\alpha(x-u)))$
Variable and parameter restrictions	$\alpha > 0$
Cumulative Distribution Function (CDF)	$F(x) = 1 - \exp(-\exp(\alpha(x-u)))$
Inverse CDF	$x(F) = \frac{\alpha u + \ln(-\ln(1-F))}{\alpha}$
Mean and standard deviation in terms of distribution parameters	Mean $= u - \gamma/\alpha$ where $\gamma = 0.5772156649$ Standard deviation $= \pi/\left(\alpha\sqrt{6}\right)$

Table C.13: The type II largest value probability distribution.

Type II Largest Value $TypeIILargestValue(u, \alpha)$	
Probability Density Function (PDF)	$f(x) = \frac{k}{u}\left(\frac{u}{x}\right)^{k+1} \exp\left(-\left(\frac{u}{x}\right)^{k}\right)$
Variable and parameter restrictions	$x > 0, \quad u > 0, \quad k > 0$
Cumulative Distribution Function (CDF)	$F(x) = \exp\left(-\left(\frac{u}{x}\right)^{k}\right)$
Inverse CDF	$x(F) = u\left(-\ln(F)\right)^{-\frac{1}{k}}$
Mean and standard deviation in terms of distribution parameters	Mean $= u\Gamma\left(1 - 1/k\right)$ St.dev. $= u\sqrt{\Gamma\left(1-2/k\right) - \Gamma\left(1-1/k\right)^2}$

Table C.14: The type III smallest value probability distribution.

Type III Smallest Value $TypeIIISmallestValue(\epsilon, u, k)$	
Probability Density Function (PDF)	$f(x) = \frac{k}{u-\epsilon}\left(\frac{x-\epsilon}{u-\epsilon}\right)^{k-1} \exp\left(-\left(\frac{x-\epsilon}{u-\epsilon}\right)^{k}\right)$
Variable and parameter restrictions	$x > \epsilon, \quad u > 0, \quad k > 0, \quad u \neq \epsilon$
Cumulative Distribution Function (CDF)	$F(x) = 1 - \exp\left(-\left(\frac{x-\epsilon}{u-\epsilon}\right)^{k}\right)$
Inverse CDF	$x(F) = (u-\epsilon)\left(\frac{\epsilon}{u-\epsilon} + (-\ln(1-F))^{1/k}\right)$
Mean and standard deviation in terms of distribution parameters	Mean $= \epsilon + (u-\epsilon)\Gamma\left(1+1/k\right)$ St.dev. $= (u-\epsilon)\sqrt{\Gamma\left(1+2/k\right) - \Gamma\left(1+1/k\right)^2}$

Table C.15: The Weibull probability distribution.

Weibull $Wbl(u, k)$	
Probability Density Function (PDF)	$f(x) = \frac{k}{u}\left(\frac{x}{u}\right)^{k-1} \exp\left(-\left(\frac{x}{u}\right)^{k}\right)$
Variable and parameter restrictions	$x > 0, \quad k > 0, \quad u > 0$
Cumulative Distribution Function (CDF)	$F(x) = 1 - \exp\left(-\left(\frac{x}{u}\right)^{k}\right)$
Inverse CDF	$x(F) = u\ (-\ln(1-F))^{1/k}$
Mean and standard deviation in terms of distribution parameters	Mean $= u\,\Gamma\left(1+1/k\right)$ St.dev. $= u\sqrt{\Gamma\left(1+2/k\right) - \Gamma\left(1+1/k\right)^2}$