To the Graduate Council:

I am submitting herewith a dissertation written by Loong K. Yong entitled "Efficient Limit State Method for Uncertainty Analysis of Pollutant Transport Models ." I have examined the final copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Nuclear Engineering.

Laurence F. Miller, Major Professor

We have read this dissertation
and recommend its acceptance:

Accepted for the Council:

Associate Vice Chancellor and
Dean of The Graduate School

EFFICIENT LIMIT STATE METHOD FOR UNCERTAINTY ANALYSIS OF POLLUTANT TRANSPORT MODELS

A Dissertation
Presented For The
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Loong Kwet Yong
December 1998

DEDICATION

This dissertation is dedicated to my father,
the late Yong Tong Chai,
and my mother,
Tay Kah Hoon

ACKNOWLEDGEMENTS

The author would like thank Dr. Laurence F. Miller for his support, patience, and guidance over the many years it took to research and prepare this dissertation. In addition, the author would like to thank Dr. James S. Bogard, Dr. Peter G. Groer, Dr. Rafael B. Perez, and Dr. Lawrence W. Townsend for serving on the author's doctoral committee.

This work would not have been possible without the domestic and morale support provided by the author's wife and daughter, Janet L. Wagner and Dixy H. Yong. In addition, the author extends his appreciation to his family in Malaysia, colleagues, friends, and employers who have provided encouragement and assistance.

ABSTRACT

The purpose of this research is to improve the computational efficiency of the Limit State method for uncertainty analysis. The viability of using the improved Limit State method with radiological and environmental assessment computer codes is investigated. The uncertainty results are compared against that obtained using Monte Carlo methods.

There are several methods available for evaluating uncertainty using the probability-distribution approach. Of these methods, the more common approaches are to use response surface analysis, differential analysis, and Monte Carlo. The Limit State method has been introduced as an alternative approach for performing uncertainty analysis. One of the problems encountered with the Limit State method is the efficient generation of a Limit State function (approximating function) for computer codes that are implicit.

In this research, automatic differentiation is utilized to improve the computational efficiency of the Limit State method. An iteration routine based on automatic differentiation can also be used to obtain more accurate results with a first order function. A groundwater pollutant transport model was selected to test the efficiency improvements to the Limit State method.

Cumulative Distribution Functions (CDF) of groundwater transport test cases using Limit State were generated. These CDFs required approximately 20 computer model runs. These CDFs are more accurate than the CDFs generated using 50 runs from Latin Hypercube sampling. The Limit State CDFs compare favorably with the CDFs generated using 100, and 500 Latin Hypercube samples. A CDF computed using 10,000 Monte Carlo runs is used as the benchmark for accuracy. The computational efficiency of the automatic differentiation approach for Limit State uncertainty analysis was compared with that using conventional numerical differentiation schemes (for computing sensitivities). Computer time savings on the order of a factor of ten can be realized using the improved Limit State approach. Automatic differentiation requires a precompilation step, which requires some effort. However, the Monte Carlo method requires large amounts of data handling, which is time consuming as well.

This research shows that the Limit State method, augmented with automatic differentiation, is a viable approach for efficiently performing uncertainty analysis of computationally intensive computer models.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

TABLE OF CONTENTS (continued)

TABLE OF CONTENTS (continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (continued)

Page

LIST OF ACRONYMS

ADGEN	Adjoint generator
AMV	Advanced mean value
ANSI	American National Standards Institute
CDF	Cumulative distribution function
DUA	Deterministic uncertainty analysis
FORM	First order reliability method
GRESS	Gradient enhanced software system
MOC	Method of characteristics
MPP	Most probable point
MPPL	Most probable point locus
MV	Mean value
MVFOSM	Mean value first order second moment
PDF	Probability density function
SORM	Second order reliability method
SWSA	Solid Waste Storage Area

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

A key issue relating to the industrial use of radioactive and hazardous chemical materials is the impact on human health and on the ecology as a result of planned or inadvertent release of these materials into the environment. The quantitative prediction of these human health and ecological impacts is achieved through the use of mathematical models, manifest as computer models, which attempt to simulate the complex processes of material behavior, transport, fate, and effect in the environment (Till, 1983). These models are used to: evaluate compliance with environmental regulations, consequence of nuclear accidents, comparative risks associated with new technologies, and cost effectiveness of new studies and data. The models are also used to serve as guides to aid in policy and decision making (Rish, 1988).

The quality of the results generated by a computer model is dependent on the accuracy of the input data and on the adequacy of the model in terms of its representation of biological and physical phenomena. Another consideration that is related to the quality of results is "completeness" - whether all the significant phenomena, relationships, and random states have been considered (e.g., whether all significant scenarios have been identified). Given the uncertainties associated with input parameters, the models, and the issue of "completeness," effective interpretation of results can only be achieved if these

uncertainties are evaluated and quantified. The most idealized method for evaluating and quantifying uncertainties is the use of the probability-distribution approach. This approach models all the input and transfer function (model) uncertainties as probability distributions and calculates the uncertainties associated with the results. In practice, the probability-distribution approach is difficult to implement. There is usually insufficient data or knowledge for establishing distributions and quantifying model uncertainties. Thus, the probability-distribution approach is typically used in conjunction with other uncertainty evaluation methods such as bounding, expert judgement, and sensitivity analysis. Bounding and corresponding assurance levels are used to quantify input uncertainty for cases where there is not enough data to establish an uncertainty parameter. Expert judgement can be summoned to assist in the development of distributions and models. In many cases, expert judgement ends up being the only mechanism available for selecting scenarios, process models, and values of parameters (Wu, 1991).

There are several methods available for evaluating uncertainty using the probability-distribution approach. Most uncertainty analysis methods rely on repeated evaluation of the model's response for different combinations of input parameters. Of these methods, the more common approaches are to use response surface analysis, differential analysis, and Monte Carlo. Recently, the use of an alternative method derived from structural reliability analysis, called the Limit State method, has been proposed. The proponents of this method suggest that the Limit State method is more efficient computationally and in some cases, offers better accuracy in the estimation of uncertainties

for repository system performance assessment models (Wu, 1992). These performance assessments are characterized by large uncertainties, large numbers of variables, and complicated nonlinear performance functions. Another attribute of the Limit State method is that it focuses on a single response or probability level, thus allowing analysis in critical performance regions. Since the method is independent of the probability level, it is much suited for evaluating tail region probabilities (i.e., cumulative distribution function [CDF] values close to 0 or 1). At this point in time, much of the research associated with this method for use in the radiological arena has been theoretical. Limited documented applications of this method to computer codes used in radiological assessments exist in literature. Further, an area in this methodology that has been identified for further work is the generation of an approximating function[1] for codes that are implicit (i.e., black boxes) with large numbers of random input variables.

1.2 STATEMENT OF PURPOSE

There are two major goals in this proposed research. The first goal is to improve the efficiency of the Limit State method for uncertainty analysis for models where there are large numbers of random input variables, as often encountered in environmental assessment problems. This goal is met by using a computer calculus (also generally referred to as automatic differentiation) method (Oblow, 1983) to generate information

[1] The approximating function is a first or second order Taylor series approximation of the actual transfer function(s) in the computer model. It is used to replace the actual computer model in Limit State uncertainty analysis.

for functions used in the Limit State method. The second goal is to apply the improved Limit State method to actual problems.

In support of the development and application of techniques for enhancing the Limit State method, tests are performed by applying the methodology to problems that are representative of those encountered in radiological and environmental assessments. The testing should yield a better understanding of the advantages and limitations of the proposed method for actual or realistic applications.

1.3 RESEARCH OBJECTIVES

The research objectives accomplished by this proposed research are comprised of the following:

(1) To use an automatic differentiation method for evaluating the partial derivatives used in developing approximate performance functions for the Limit State.

(2) To test the viability of using the improved Limit State method with radiological and environmental assessment codes.

(3) To compare the uncertainty analysis results achieved with the Limit State method against the Monte Carlo and Latin Hypercube sampling method.

1.4 ORIGINAL CONTRIBUTIONS

The original contributions from this research are:

(1) Implementation and testing of a more efficient, automatic differentiation based, method for performing Limit State based uncertainty analysis. The use of automatic differentiation allows for the decoupling of the number of runs needed to perform Limit State uncertainty analysis from the number, n, of random input variables modeled. This significantly reduces the computational overhead incurred as n increases in size, since when n gets larger, the attractiveness of using Latin Hypercube sampling or Monte Carlo increases (if we ignore criticisms about the ability of the Latin Hypercube sampling method to properly handle correlated variables), relative to the conventional Limit State method.

(2) Development of a Most Probable Point (MPP) localized function approach to overcome limitations of using highly local derivative based Limit State functions in a more global capacity. This approach, made possible with the ability to compute partial derivatives of interest easily, is tested and presented in this research. The results show good agreement with the benchmark 10,000 run Monte Carlo derived CDF.

(3) Performance of Limit State uncertainty analysis using the Finite Difference Groundwater Transport model as a "test bed." Very few examples of the Limit

State method being used with an implicit (i.e., numerical) computer model are available in literature.

(4) Comparisons of uncertainty analysis results using Latin Hypercube sampling and Monte Carlo with the Limit State CDFs. This research shows the levels of accuracy obtained using different Latin Hypercube sample sizes and compared it to the Limit State results.

(5) Comparisons of the level of human effort and computer time required by the different uncertainty analysis methods studied. These comparisons are not readily available in literature.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

Over the course of the past ten to fifteen years, the study of methods for evaluating uncertainty and sensitivity associated with the use of mathematical models has achieved much attention. As such, there is a large and well founded body of work that exists in literature on the topic of uncertainty and sensitivity analysis. A survey of the literature reveals that while many methods have been proposed, developed, and used, there are only a few methods that are widely preferred for uncertainty and sensitivity analysis in the area of radiological and environmental assessments. These methods are Monte Carlo, the response surface method, and differential analysis (Iman, 1985). The Limit State method, which is the subject of this research, is introduced as an alternative method for uncertainty and sensitivity analysis for performance assessments of radioactive waste repositories (Wu, 1991). The following is a review of the technical literature that describe the development, theory, and application of the most widely applied uncertainty and sensitivity analysis methods.

2.2 RESPONSE SURFACE METHOD

The response surface method consists of approximations to computer models by a suitable response function (Myers, 1976). The approximating function, called the response surface, is then used to replace the computer model for evaluating the response.

The response function (usually representing a linear hyper-surface) is generated by first using an experimental design to select specific values and pairings of the input parameters that are used for making runs with the computer model. The output from the computer model and the input parameters are used in the least squares method to obtain the coefficients for a general model. The model is known as the fitted response surface and is used in place of the computer model itself.

The two most commonly used types of experimental design in response surface work are the factorial and the central composite design. In factorial designs, two or more fixed values (i.e., levels) are used to represent each input variable under consideration. Thus, in the case where two levels are used for k input variables, there are 2^k possible combinations of the k variables. Hence, for 3 levels, there are 3^k combinations and in general for n levels, n^k combinations.

A problem that arises in the use of factorial designs is that the number of combinations of the input variables becomes rather large as the number of input variables increases. In order to apply the factorial design method in a practical manner for computer models with large numbers of variables, an approach called fractional factorial design is used. This approach reduces the number of treatment combinations by using a fraction of the total number of treatment combinations. A drawback of this approach is that as the fraction of the total number of treatment combination decreases, the effects of some individual variables cannot be estimated because of confounding interactions among

the variables. Thus, the selection of the fraction size and treatment combinations must be done carefully. The significance and interaction of the variables in relation to the model response must be taken into consideration before selection of the actual design for use in the analysis can be made.

The class of central composite designs is an efficient class of experimental design for fitting second-order response surfaces (Box and Wilson, 1951). This design approach consists of a complete or fractional 2^k factorial design, where the factor values are coded to -1 and +1, augmented by $n_0 \geq 1$ center points and completed by 2k points placed at the coordinates $-\alpha$ and $+\alpha$ of the design center. The total number of design points is $N = 2^k + 2k + n_0$. The values of α and n_0 are chosen by the experimenter to achieve the desired design properties.

Since the selection procedure used with the input values to create a response surface is not random, the distribution function for an output variable cannot be estimated directly from the set of output values. In order for the surface response method to be used for uncertainty analysis, moment matching, or Monte Carlo simulation of the response surface are techniques typically used to approximate the probability density function (PDF).

The roots of the response surface method can be traced back to the early 1930s or even earlier (Khuri, 1987). However, it was in 1951 that Box and Wilson formally

developed the response surface method (Box and Wilson, 1951). Since then, the response surface method has been used and applied widely in the many diverse fields of science and engineering. In the area of nuclear engineering, Pike and Smith (1981), and Pike and Wetherill (1983) applied the response surface method to reactor safety studies. Faravelli (1989) used the response surface method in his development of a stochastic finite element method for computing the response uncertainty (reliability) of structural and mechanical systems (his numerical example was the reliability analysis of a reactor pressure vessel). A critical examination of the response surface method for uncertainty analysis in assessment models was performed by Downing et al. (1985). In this study, two techniques of uncertainty analysis (the response surface method and Latin Hypercube sampling) were applied to a mathematical model that estimates the dose-equivalent to man from the concentration of radioactivity in air, water, and food. The response surface technique employed consisted of data screening, response surface modeling, and fitting a Pearson or Johnson distribution to the calculated moments to estimate the output PDF. The Latin Hypercube sampling method was used on the same model as a comparison. The comparison indicated that it was often difficult to ascertain the adequacy or reliability of the response surface method and that Latin Hypercube sampling is a better technique for uncertainty analysis. However, it was noted that "both approaches offer different insights to the input-output relationship of the computer code." Furthermore, the Downing paper stated that "few studies exist that indicate the superiority of one approach over the other."

2.3 DIFFERENTIAL ANALYSIS

The differential analysis method is based on replacing a complex performance (or transfer function) with a Taylor's series expansion and the associated partial derivatives (similar in principle to the response surface method). The Taylor's series is expanded about a fixed base value, usually the mean of the input variables. The mean and variance of the output from the actual performance function is then estimated from the first-order Taylor's series approximation.

A higher order Taylor's series may be achieved by expanding the series to include terms with second order or higher derivatives. However, in most practical situations, the expansion is typically truncated after the first or second order derivatives. This is because of the complexities and additional computational effort needed to generate a second or higher order series. For instance, a second order Taylor expansion calls for second order partial derivatives of the performance function and moments of order 4 of the input random variables vector in the calculation of the variance (or second moment) of the performance function.

A key step in the differential analysis method is the generation of the partial derivatives required in the series. For cases where the actual performance function is relatively simple, the partial derivatives can be generated analytically or by simple differencing schemes. In cases where the performance function of interest in embedded in a complex computer code, numerical techniques such as the Runge-Kutta method,

perturbation analysis, or the adjoint method may be employed to obtain the partial derivatives. An automated systems procedure for computing partial derivatives for computer models using computer calculus was developed by Oblow (1985) and Worley (1989). These automated systems are the GRESS (Gradient-Enhanced Software System) and ADGEN (Adjoint Generator) methodologies, which respectively use direct and adjoint approaches for solving derivatives.

In order to perform uncertainty analysis, Monte Carlo simulation is often used in conjunction with a Taylor series to estimate distribution functions (Iman, 1985). Another approach is to use the deterministic uncertainty analysis method (DUA) introduced by Worley (1987). This method is an extension of the probability tree method. The multi-dimensional parameter space is discretized into a mesh that spans the entire parameter space. The probability of each section of the mesh occurring within the entire parameter space is calculated as well as the expected value of the response function within that particular section of the mesh. The probability is then assigned to the expected value of the response to obtain the probability of the expected value of response within the discrete space of expected values. The probability and expected value of response pairs are then reordered to form a probability distribution function of the response over the parameter space.

With the differential analysis approach, sensitivity analysis for quantifying the relative importance of input variables is often performed by normalizing the Taylor series

coefficients. One such normalization procedure, as explained in Draper and Smith (1981), involves normalizing the partial derivatives (Taylor series coefficients) with respect to the standard deviation. Thus, the normalized coefficients represent a measure of the fractional change in the results relative to its standard deviation when the input variables are changed by a fixed fraction of its standard deviation. Another method of normalization involves multiplying each of the Taylor series coefficients by an input variable specific factor. The factor consists of the value of each respective input variable divided by the output value (from the model or transfer function) for a chosen base case. According to Iman (1985), "the resultant coefficients indicate the effect on the dependent variable of equivalent fractional changes of base case values for the individual input variables."

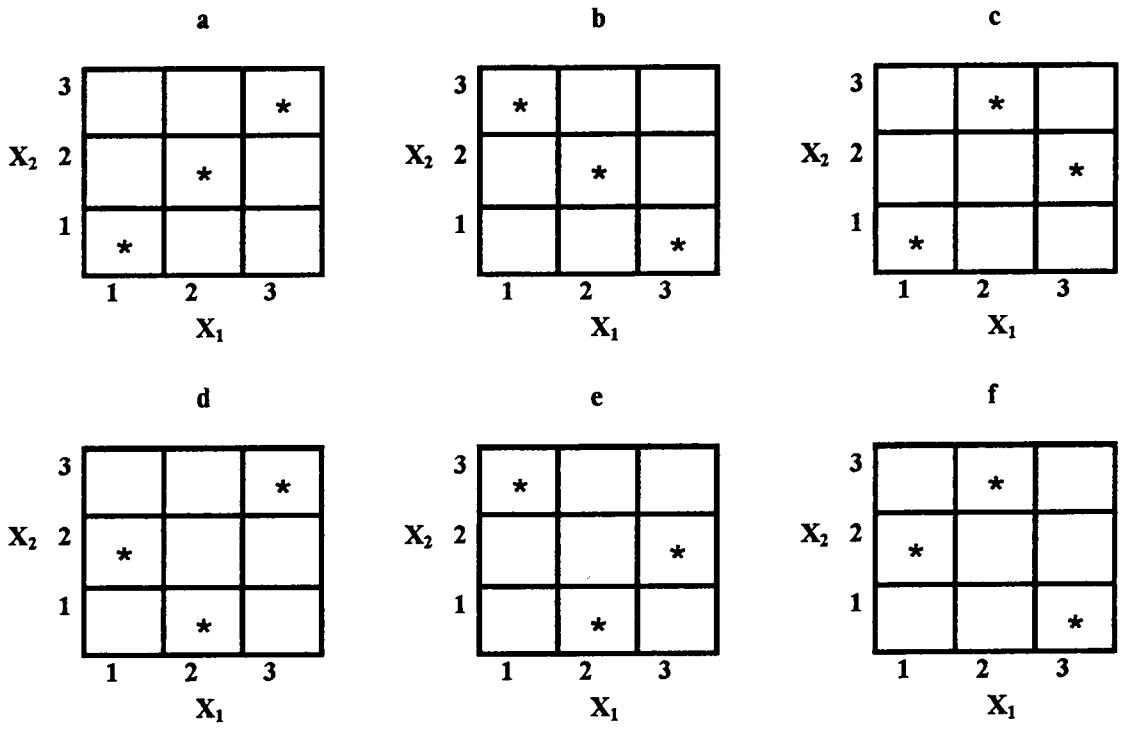
2.4 MONTE CARLO AND LATIN HYPERCUBE SAMPLING

In using the Monte Carlo method for uncertainty and sensitivity analysis, k realizations of n input variables, $X_1, \ldots, X_n$, are generated from given probabilistic distribution information. These k realizations are then used as input to generate k sets of output or responses from runs made with the computer model of interest. The response values are then used to generate probabilistic information, typically in the form of CDFs or complimentary CDFs. One of the types of sampling used in Monte Carlo is a form of stratified sampling called Latin Hypercube sampling.

In Latin Hypercube sampling, the range of each variable is divided into n non-overlapping intervals on the basis of equal probability. One value is then randomly

selected from each interval, based on the probability density in that interval. For a sample of size N, the range of each input variable X_i (i = 1, ... , L) is divided into nonoverlapping intervals of probability 1/N, and a value $x_{i,n}$ is chosen at random from the n^{th} interval (n = 1, ... , N). The N values thus obtained for X_i are rearranged according to an independent random permutation of {1, 2, ... , N}. The resulting permutations form an L x N matrix, with the l^{th} row consisting of the permuted values of X_i. The N columns of this matrix constitute the Latin Hypercube sample (Iman, 1984). As an example: n values obtained for X_1 are paired randomly with n values of X_2 in equally likely combinations. For given values of n and k, there exist $(n!)^{k-1}$ possible interval combinations for a Latin Hypercube sample. The six interval combinations are marked by an asterisk in the Figure 2.1. In Figure 2.1, diagram a shows a value of X_1 randomly selected in interval 1 and paired with a value of X_2 selected at random from the first interval of X_2. Similar pairings are selected randomly for intervals 2 and 3. Diagram b shows the value of X_1 from interval 1 being paired with the value of X_2 in interval 3. Thus, in each of the six diagrams, the entire range of both X_1 and X_2 are sampled, as opposed to a simple Monte Carlo scheme where all three pairs of values from the same sub-intervals of X_1 and X_2 could have been selected.

Latin Hypercube sampling covers the range of each of the input variables and results in a representative sample. However, a drawback of Latin Hypercube sampling is that it is not able to accommodate a general dependency structure for the input variables. Journel, in Appendix A of Wu (1991) states that "even though Latin Hypercube sampling



Source: Iman, R. L. and J. C. Helton, "A Comparison of Uncertainty and Sensitivity Analysis Techniques for Computer Models," NUREG/CR-3904, Sandia National Laboratories, NM, 1985.

Figure 2.1 Example of Latin Hypercube sampling

can be modified to accommodate some types of dependency, there is a danger that use of Latin Hypercube sampling can lead to biased estimates." For high level geologic waste repository modeling, Journel recommends using stratified sampling instead of Latin Hypercube sampling.

McKay et. al. (1979) presented Latin Hypercube sampling as one of the two alternatives to simple random sampling in Monte Carlo studies. They showed, relative to the performance of the respective sampling methods in a computer code, that Latin Hypercube sampling improves upon simple random sampling. For transfer functions which are monotonic functions of each of its arguments, Latin Hypercube sampling was shown to have a smaller variance compared to random sampling and stratified sampling for estimating the mean and the population distribution function. Note however, that the same number of samples were generated for each sampling method used in the McKay study (i.e., equal numbers of function evaluations). In practice, smaller Latin Hypercube samples are often used in place of large random samples (e.g., 100 to 300 Latin Hypercube samples in lieu of several thousand random samples). For these cases, the quality of Latin Hypercube sampling derived results versus random sampling are not as obvious. In addition, Latin Hypercube sampling has the disadvantage of having no theoretical basis for uncertainties.

The study performed by McKay et al in 1979 considered only the cases where the components of **X** are independent. Iman and Conover (1982) extended the usefulness of

Latin Hypercube sampling by introducing a method for inducing a desired rank correlation structure, on the input variables, that preserves the exact marginal distributions. Using an example, Iman and Conover showed that this technique is better than Latin Hypercube sampling, based on independent inputs. They also pointed out that their method should be used whenever input random variables are correlated, instead of just assuming independence for simplicity.

Latin Hypercube sampling has been used extensively in probabilistic risk assessments and uncertainty analysis of radioactive waste repository performance assessments (ORNL, 1994). In most cases, the input variables are generally assumed to be independent or have a simple dependency structure.

2.5 FIRST AND SECOND ORDER RELIABILITY METHOD AND THE LIMIT STATE METHOD

The origination of the reliability index (also known as the safety index) to describe structural reliability is attributed to Cornell (1969). Based on Mean Value First Order Second Moment (MVFOSM) theory, this method's main disadvantage is the lack of invariance with respect to the formulation of the failure function or limit state. This limitation is overcome by Hasofer and Lind when they proposed a method that is based on using the limit state surface as opposed to linearization at mean values, for the

determination of the reliability index (Hasofer and Lind, 1974). This reliability index is called the generalized reliability index.

Since Hasofer and Lind's method uses only information relating to the first and second moments of the input parameters, any distributional information related to higher moments that might be available is not used. This consideration lead to an algorithm proposed by Rackwitz and Fiessler that incorporates distributional information in the computation for the reliability index (Rackwitz and Fiessler, 1978). The method that they introduced makes use of the Rosenblatt transformation (Rosenblatt, 1952) to retain the distributional information associated with the random variables in the determination of the reliability index. The theory behind Rackwitz and Fiessler's work was later described by Ditlevsen (1981). Rackwitz and Fiessler's paper was considered difficult to understand and this lead to several other variations of the Rackwitz Fiessler algorithm, such as that proposed by Madsen (1986). Another algorithm that deserves mention is the Chen-Lind algorithm (Chen and Lind, 1983), which extends the methodology further by incorporating an additional scale parameter for the equivalent normal distribution. The methods used in these algorithms are collectively known as "fast probability integration." Fast probability integration refers to the analytical solutions for computing the CDF of performance functions. These analytical solutions are based on linear and quadratic approximations of the performance functions. Use of fast probability integration obviates the need to integrate joint PDFs for computing CDFs.

When the reliability analysis method described above makes use of a first order limit state, it is referred to as the First Order Reliability Method (FORM). Similarly, if a second order limit state is used, the method is called the Second Order Reliability Method (SORM). The results of research relating to the second order method can be found in Ditlevsen (1979), Fiessler et al. (1979), Der Kiureghian et al. (1987), Hohenbichler and Rackwitz (1988), and Tvedt (1990). In general, the second order techniques are complicated and computationally intensive.

An important step in the development of FORM and SORM was its application to problems where the performance functions are implicit and cannot be expressed in closed analytical forms. Such implicit performance functions are generally found in numerical computer models. Adaptation of FORM and SORM to problems with implicit performance functions typically consist of replacing the implicit function with a simpler approximation function. An early method was based on the use of multiple regression to generate the approximate function (Wu, 1984). A review of FORM methods (Wong, 1985) featured a point-estimates-for-probability-moments and a response surface based system. Both of these methods were found to be compatible with numerical modeling techniques. Currently, most techniques use numerical differentiation to generate coefficients for a first order polynomial approximation to the actual performance function (Wu, 1984). Recent efforts to improve the accuracy of the methodology has led to the introduction of the Advanced Mean Value (AMV) method (Wu, 1990) and the Most

Probable Point Locus (MPPL) tracking technique (Wu, 1991). In Wu (1991), the agglomeration of AMV and fast probability integration is called the Limit State method.

FORM, SORM, and the Limit State method were originally introduced as reliability analysis techniques for structural engineering applications. As such, the problems that were addressed using these methods primarily concerned probabilities of failure in mechanical structures and components. One of the first applications of these methods in the environmental arena was the uncertainty analysis of groundwater flow (Dettinger and Wilson, 1981). In a related study, stochastic analysis of subsurface flow and contaminant transport was performed (Sitar et al., 1987). This study used examples with both explicit and implicit performance functions. As far as radiological applications are concerned, probabilistic analysis of radioactive waste package performance (Song and Lee, 1989) and waste repository modeling (Song and Lee, 1992) have been conducted. However, in both cases, only performance functions in closed analytical forms were used. A recent study performed by Wu (1993) involved the uncertainty analysis of one-dimensional radionuclide transport through layered fractured rock. In this study, analytical solutions to the performance function were used as well.

CHAPTER 3

GENERAL THEORY

3.1 DEFINING THE LIMIT STATE

Let the result from a computer model be defined as

$$Z = Z (x_1 , x_2 , x_3 , . . . , x_n) , \qquad \textbf{(3-1)}$$

where x_i are the random variables describing the uncertain parameters of the problem.

In the limit state formulation, Z is approximated at a selected value, e.g. $Z = z$. The various combinations of x_i that result in the limit value z is defined by a "limit state surface." This limit state surface is similar to a response surface, but it is constrained by the condition that $Z = z$. The limit state surface partitions the *n*-dimensional space of **x** into two parts. When $g(\mathbf{x})$ is defined as

$$g(\mathbf{x}) = Z(\mathbf{x}) - z = 0 , \qquad \textbf{(3-2)}$$

the hyperplane $g(\mathbf{x})$, which represents the limit state surface, forms the boundary between "safe" and "unsafe" regions. Note that the terms "safe" and "unsafe" regions are conventions of the traditional applications of this method in structural reliability problems. In the case of environmental problems, the "safe" and "unsafe" regions correspond to domains of **x** that result in probabilities above or below a specified value, e.g. a regulatory

dose or pollutant concentration limit. Figure 3.1 illustrates the limit state concept for a two-dimensional case.

Given the joint PDF $f_x(\mathbf{x})$, the probability of failure (i.e., the CDF of the performance function), is given by

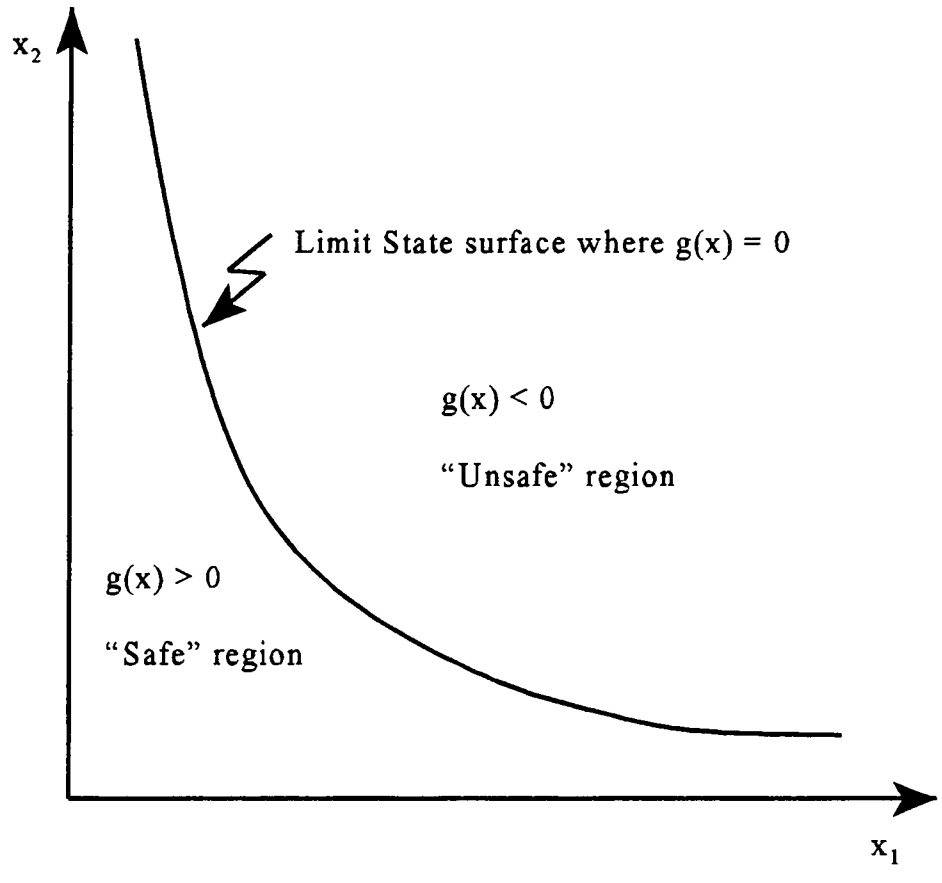
$$p_f = P[g(x) \le 0] = \int_{g(x)\le 0} \cdots \int f_x(x)\, dx \,. \qquad \textbf{(3-3)}$$

Note that reliability is conventionally defined as the complement of the probability of failure, $1 - p_f$.

In practice, the joint PDF, $f_x(\mathbf{x})$, is often unknown since complete probability information on $\mathbf{x}$ is seldom available (Kiureghian, 1986). Even if $f_x(\mathbf{x})$ is known, an exact solution of the multiple integral is often not practical with analytical or direct numerical approaches for n greater than 5 or 6. Thus, alternative methods for evaluating the integral have been developed to enable p_f computations for practical engineering applications.

3.2 MEAN-VALUE FIRST-ORDER SECOND-MOMENT RELIABILITY METHOD

In many engineering applications, the available statistical information for the input parameters is limited to the first and second moments, i.e., the mean and the variance. In



Source:Sitar, N., Cawlfield, J. D., Kiureghian, A. D., "First-Order Reliability Approach to Stochastic Analysis of Subsurface Flow and Contaminant Transport," Water Resources Research, Vol. 23, No. 5. Pages 794-804, May 1987

Figure 3.1 Joint sample space in two-dimensions

most cases, there isn't enough information to characterize the distribution type. Since the propagation of the first two moments is a simpler task than the exact solution, first-order second-moment methods which entail the use of the means and variance and their propagation through functional relationships can be used. In the context of the first-order second-moment reliability method, an alternative measure of reliability, known as the reliability index is introduced (Cornell, 1969). This reliability index, β, is defined as

$$\beta = \frac{\mu}{\sigma} , \tag{3-4}$$

where

$$\mu = g(\boldsymbol{\mu}) ,$$

$$\sigma = \sqrt{\sum_{i=0}^{n} \left(\frac{\partial g}{\partial X_i}\right)_{\boldsymbol{\mu}}^2 \sigma_i^2} ,$$

and where $\boldsymbol{\mu}$ is a vector of mean values and σ_i is the standard deviation of the x_i variable. The basic idea behind this reliability index definition is that the location measure μ (relative to the limit state surface) provides a good measure of reliability. The distance is measured in units of the standard deviation σ.

The most popular first-order second moment method is the partial derivative or Taylor series expansion method. In this method, the function Z is expanded in a power series about a point, usually the mean value μ, thus yielding a Taylor series. The first and second moments of Z can then be computed directly from this polynomial in terms of the first and second moments of $\mathbf{x}$. Note that when the linearization point is the mean-value point, the reliability index is called a MVFOSM reliability index. The formulation for this first-order second-moment method is shown for a linear limit-state function and normally distributed random variables, x_i. The failure surface is a hyperplane and the linear failure function can be approximated as

$$\tilde{Z} = a_0 + \sum_{i=1}^{n} a_i x_i \,, \tag{3-5}$$

where a_0 and a_i are constants and $\tilde{Z}$ approximates Z. If all x_i are normal variables, $\tilde{Z}$ is also normally distributed with a mean

$$\mu_{\tilde{Z}} = a_0 + \sum_{i=1}^{n} a_i \mu_i \,, \tag{3-6}$$

and a standard deviation

$$\sigma_{\tilde{Z}} = \sqrt{\sum_{i=1}^{n} a_i^2 \sigma_i^2} \,, \tag{3-7}$$

The failure condition is $\tilde{Z} < 0$ and the probability of failure is

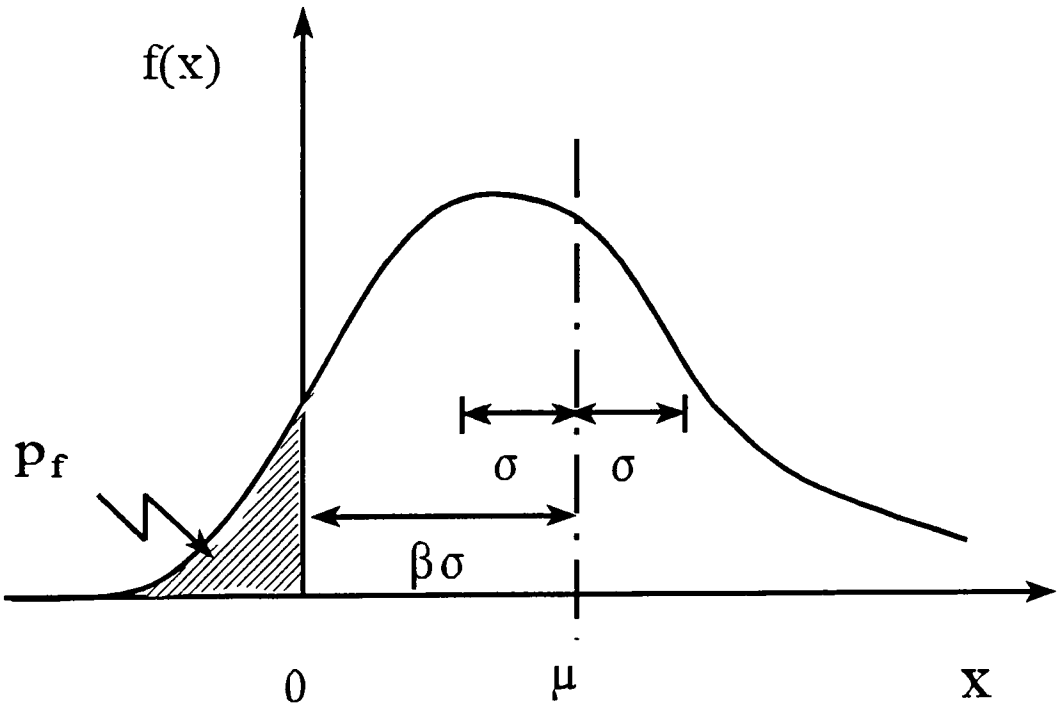
$$p_f = P(\tilde{Z} \leq 0) = P\left(\frac{\tilde{Z} - \mu_{\tilde{Z}}}{\sigma_{\tilde{Z}}} \leq -\frac{\mu_{\tilde{Z}}}{\sigma_{\tilde{Z}}}\right), \quad \textbf{(3-8)}$$

and

$$p_f = \Phi(-\beta) = P\left(-\frac{\mu_{\tilde{Z}}}{\sigma_{\tilde{Z}}}\right), \quad \textbf{(3-9)}$$

where Φ is the standard normal distribution (Wu, 1984). A graphical presentation of the probability of failure, p_f, and the reliability index is shown in Figure 3.2.

The MVFOSM method is simple and straightforward to apply in practice. However, since the method relies only on use of the first two moments, distributional information is limited. A severe limitation of this method lies in the fact that β may depend on the formulation of the limit state. For example, consider (a structural problem illustration) a simple tension member with cross-sectional area A has to withstand an



Source: Sitar, N., Cawlfield, J. D., Kiureghian, A. D., "First-Order Reliability Approach to Stochastic Analysis of Subsurface Flow and Contaminant Transport," Water Resources Research, Vol. 23, No. 5, pages 794-804, May 1987.

Figure 3.2 Probability of failure, p_f, and the reliability index

applied loading Q. The strength is R and the stress is S = Q / A. The limit state can be formulated in two ways:

$$Z = R - \frac{Q}{A} = 0 , \qquad \textbf{(3-10)}$$

or

$$Z = RA - Q = 0 . \qquad \textbf{(3-11)}$$

Q is deterministic and R and A are random variables. Although the limit state is the same for both cases, different values of β may be produced in each case. Thus, β is dependent upon the manner in which the limit state function is formulated. This problem, referred to as the lack of invariance, was overcome in the method that is described next.

3.3 HASOFER-LIND RELIABILITY INDEX

The lack of limit state function invariance was overcome when Hasofer and Lind (1974) proposed a reliability index that was shown to be invariant with respect to the choice of limit state function. Hasofer and Lind's method first calls for reducing the set of variables, $\mathbf{x} = (x_1, \ldots, x_n)$ to standard normal space (or u-space). This new set $\mathbf{u} = (u_1, \ldots, u_n)$ is defined by

$$u_i = \frac{x_i - \mu_i}{\sigma_i} , \qquad \textbf{(3-12)}$$

where μ_i and σ_i are the mean values and standard deviations. Note that $\mu_{ui} = 0$, $\sigma_{ui} = 1$, and $i = 1, \ldots, n$. Using linear mapping, the original limit state $g(x) = 0$, is transformed to the reduced limit state $g(u) = 0$. The reduced limit state, now mapped in u-space, is rotationally symmetric with respect to the second moment (standard deviation) distribution for the **x** variables (Madsen, 1986).

The Hasofer and Lind reliability index can be shown to be identical to the Cornell reliability index when the failure surface is linear (a hyperplane). Taking the case where all x_i are normally distributed and the failure surface is a hyperplane, the reduced limit state becomes

$$g_1(\mathbf{u}) = a_o + \sum_{i=1}^{n} a_i(u_i\sigma_i + \mu_i) \qquad \textbf{(3-13)}$$

$$= (a_o + \sum_{i=1}^{n} a_i\mu_i) + \sum_{i=1}^{n} a_i\sigma_i u_i \, .$$

Hence, from the theory of calculus, the minimum distance, β, from the origin to the "hyperplane" is

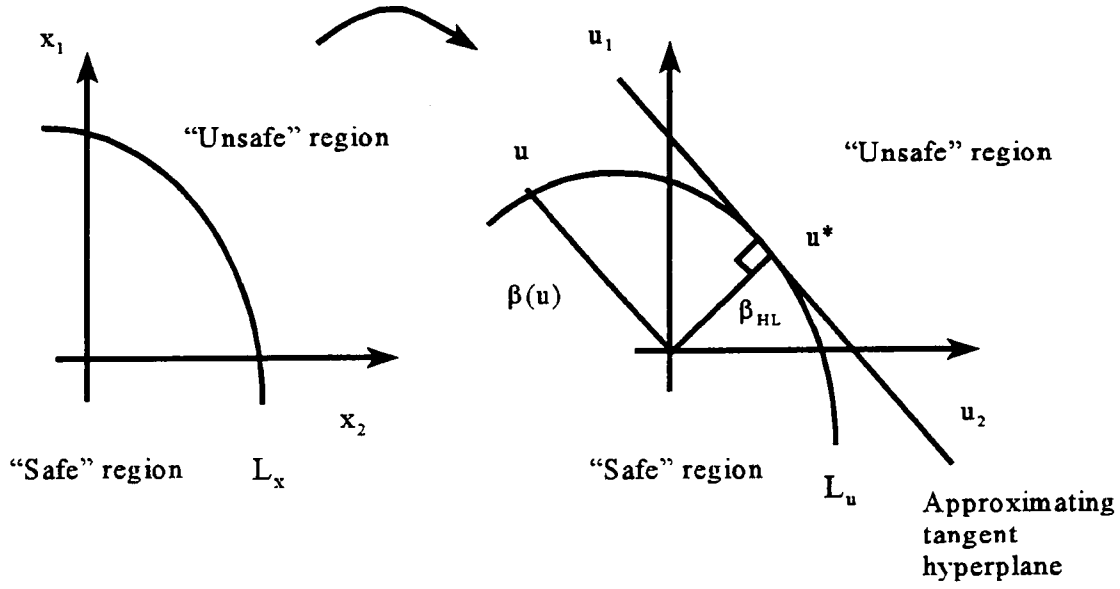
$$\beta = \frac{a_o + \sum_{i=1}^{n} a_i\mu_i}{\sqrt{\sum_{i=1}^{n} a_i^2\,\sigma_i^2}} \, . \qquad \textbf{(3-14)}$$

Since the numerator and denominator in this formulation are equal to μ_z and σ_z respectively, the β is the same as that defined using the MVFOSM method and p_f is related to β (Wu, 1984). A geometrical illustration of the Hasofer and Lind reliability index is shown in Figure 3.3.

The invariance of the Hasofer and Lind reliability index with respect to the limit state function stems from relating β to the limit state surface. As a result, all similar failure functions will result in the same limit state surface. This is as opposed to the MVFOSM method, where the limit state function is linearized using mean values.

For non-linear limit state surfaces, the MPPs point can be located iteratively using an optimization algorithm which minimizes the distance from the origin in the standard normal space to the limit state surface. A widely used algorithm introduced by Madsen (1986) determines the design point u^* as the limit of the sequence $u_0, u_1, u_2,\ldots,u_n$ according to the following equation

$$\mathbf{u}_{i+1} = \left[\alpha_i u_i + \frac{G(u_i)}{|\nabla u_i\ G(u_i)|}\right] \alpha_i^T \quad . \qquad \textbf{(3-15)}$$



Source: Madsen, H. O., Krenk, S., Lind, N. C., "Methods of Structural Safety," page 52, Prentice-Hall, Inc., Englewood Cliffs, NJ 07632, 1986.

Figure 3.3 Geometrical illustration of the Hasofer and Lind reliability index (β_{HL})

where

$$\nabla_u G(\mathbf{u}) = \left(\frac{\partial G(u)}{\partial u_i}, \ldots, \frac{\partial G(u)}{\partial u_n} \right) \tag{3-16}$$

is the gradient vector, and

$$\alpha = - \left[\frac{\nabla_u G(\mathbf{u})}{|\nabla_u G(\mathbf{u})|} \right] \tag{3-17}$$

is a unit vector pointing in the negative gradient direction. Using the chain rule, the gradient is computed in terms of the gradient vector in the original space $\nabla_x g(x)$

$$\nabla_u G(\mathbf{u}) = (\nabla_x g(\mathbf{x}))\mathbf{J}_{\mathbf{u},\mathbf{x}}^{-1} \tag{3-18}$$

where $\mathbf{J}_{\mathbf{u},\mathbf{x}}$ is the Jacobian of the linear transformation of the random variables $\mathbf{x}$ into a set of uncorrelated variates $\mathbf{u}$ having zero means and unit variances. Starting from an arbitrary point (e.g., u = 0), the sequence usually converges to u^* and α^* in just a few model evaluations.

The Hasofer and Lind reliability index is an improvement over the Cornell reliability index in that it is invariant relative to the limit state formulation. However, since only the mean and standard deviation for the variables are used, any higher moment distributional information is neglected. Thus, it was evident that the next improvement to

be made in the development of the Limit State method would make use of distributional information. This was accomplished by Rackwitz and Fiessler as described in the next section.

3.4 FAST PROBABILITY INTEGRATION

Rackwitz and Fiessler suggested a method which extended the Hasofer and Lind reliability index concept by incorporating the distributional information associated with the random variables. This is achieved by transforming non-normal distributions into "equivalent" normal distributions in such a way that the values of the original density functions f(x) and the original distribution function F(x) for the random variables x are equal to the corresponding values of the density function for a normally distributed variable at the MPP. The mean and standard deviation of the "equivalent" normal distribution are determined such that values of the distribution functions and the probability density functions are identical at the MPP, x^*, i.e.,

$$F_{x_i}(x_i^*) = \Phi\left(\frac{x_i^* - \mu_{x_i}}{\sigma_{x_i}}\right), \qquad \textbf{(3-19)}$$

$$f_{x_i}(x_i^*) = \frac{1}{\sigma_{x_i}}\,\phi\left(\frac{x_i^* - \mu_{x_i}}{\sigma_{x_i}}\right).$$

where ϕ is the standard normal PDF and Φ is the standard normal CDF.

Note that the MPP $x^* = (x_1^*, \ldots, x_i^*, \ldots, x_n^*)$ and μ_{xi}' and σ_{xi}' are the unknown mean and standard deviation of the equivalent normal distribution (not the standard normal distribution). Solving for μ_{xi}' and σ_{xi}' (at the MPP x^*) gives

$$\sigma_{x_i}' = \frac{\phi(\Phi^{-1}(F_{x_i}(x_i^*)))}{f_{x_i}(x_i^*)} \qquad \textbf{(3-20)}$$

$$\mu_{x_i}' = x_i^* - \Phi^{-1}(F_{x_i}(x_i^*))\sigma_{x_i}' .$$

In order to determine the MPP, the iterative algorithm proposed by Rackwitz and Fiessler (1978) first requires that the random variables be reduced to standard normal variables using

$$u_i = \frac{x_i - \mu_i}{\sigma_i} \qquad i = 1, \ldots n \qquad \textbf{(3-21)}$$

where x_i are the random variables, μ_i the mean of x_i, and σ_i the standard deviation of x_i respectively. Next, the approximate limit state surface is defined in reduced variables, i.e.,

$$g_1(\mathbf{u}) = 0 \qquad \textbf{(3-22)}$$

where $\mathbf{u} = (u_1, u_2, \ldots, u_n)$. An initial estimate of the reliability index is then made,

$$\beta = \min\sqrt{u_1^2 + u_2^2 + \ldots + u_n^2} \qquad \textbf{(3-23)}$$

subject to $g_1(\mathbf{u}) = 0$, where $g_1(\mathbf{u})$ is an approximation of the computer model. The MPP x^*, in standard space, is calculated using the following equation:

$$x_i^* = u_i^* \sigma_i + \mu_i \, . \qquad \textbf{(3-24)}$$

With the MPP x^*, new normal parameters μ_i and σ_i are calculated for each non-normal variable:

$$\sigma_{Ni} = \frac{\phi[\Phi^{-1}(F_i(x_i^*))]}{f_i(x_i^*)} \, , \qquad \textbf{(3-25)}$$

and

$$\mu_{Ni} = x_i^* - \Phi^{-1}[F_i(x_i^*)]\sigma_{Ni} \, , \qquad \textbf{(3-26)}$$

where ϕ is the standard normal PDF, and Φ is the standard normal CDF. New reduced variables are then defined as:

$$u_i' = \frac{x_i - \mu_{Ni}}{\sigma_{Ni}} \, , \qquad \textbf{(3-27)}$$

and a new estimate of the reliability index is calculated:

$$\beta_1 = \min \sqrt{(u_1')^2 + \ldots + (u_N')^2} \tag{3-28}$$

subject to $g_1(\mathbf{u}') = 0$. The calculations for μ_{N_i}, σ_{N_i}, and u_i' are repeated until convergence is achieved, e.g.,

$$|\beta_N - \beta_{N-1}| \leq \epsilon \tag{3-29}$$

where ϵ is the "error" (typically $\epsilon = 0.001$ is used). The probability of failure is finally calculated using $\beta = \beta_N$:

$$p_f = \Phi(-\beta) \; . \tag{3-30}$$

The Rackwitz-Fiessler algorithm outlined above is recognized as efficient because it does not require the knowledge of the exact limit state in the standard normal space.

3.5 ADVANCED MEAN VALUE METHOD

The AMV method is an iterative method for improving the CDF calculated using the mean value method. The mean value (MV) method, also known as differential analysis, involves the use of a first order Taylor's series to approximate the performance function as follows:

$$\tilde{Z}_{MV} = \tilde{Z}(\boldsymbol{\mu}) + \sum_{i=1}^{n} \left(\frac{\partial \tilde{Z}}{\partial x_i} \right) \cdot (x_i - \mu_i) \tag{3-31}$$

$$= a_o + \sum_{i=1}^{a} a_i x_i$$

where

$$\mu_i = E(x_i) \ . \qquad \textbf{(3-32)}$$

The approximate function is based on the mean values, i.e., the derivatives ($\Delta \tilde{Z} / \Delta x_i$) are evaluated at the mean values of x_i. There are several ways by which the derivatives can be obtained. In the case where the performance function is implicit, such as in numerical codes using finite differences or finite elements, the derivatives can be evaluated using numerical differentiation, the adjoint method, and the least squares method. Note that using numerical differentiation to solve for the derivatives requires n+1 computer code runs.

For a limit state value z, the probability of $\{ \tilde{Z}_{MV} < z \}$ can be computed using fast probability integration. This estimate is not very accurate when the performance function is highly nonlinear. The AMV method improves on the MV method by adding to the approximate performance function a term consisting of a deterministic function $H(\tilde{Z}_{MV})$. This is expressed as:

$$\tilde{Z}_{AMV} = \tilde{Z}_{MV} + H(\tilde{Z}_{MV}) = \left(a_o + \sum_{i=1}^{n} a_i x_i \right) + H(\tilde{Z}_{MV}) \qquad \textbf{(3-33)}$$

where the $H(\tilde{Z}_{MV})$ function is defined as the difference between $\tilde{Z}_{MV}$ and Z_{EXACT} (i.e., the model).

The stepwise AMV method is as follows:

- Obtain a linear approximation of the performance function at $\mathbf{x}$ (mean values of parameters).
- Compute the CDF of $\tilde{Z}_{MV}$ at selected limit states $\mathbf{z}$ using fast probability integration.
- Select several CDF values that are representative of the probability range of interest.
- For each selected CDF value, identify the MPP $\mathbf{u}^*$ in the transformed u-space based on the approximating function.
- Transform $\mathbf{u}^*$ to $\mathbf{x}^*$.
- Compute $\tilde{Z}_{EXACT}$ and determine $H(\tilde{Z}_{MV})$, i.e., the difference between $\tilde{Z}_{MV}$ and Z_{EXACT} at each value of CDF selected.
- Update $\tilde{Z}_{AMV}$ with $H(\tilde{Z}_{MV})$ and recompute the CDF.

The last step updates $\tilde{Z}$ at the MPP, thus the MPP, $\mathbf{x}^*$, defines a limit state of $Z_{EXACT} = Z(\mathbf{x}^*)$. This update requires one Z_{EXACT}-function calculation (Wu, 1992). The MPP and CDF can be improved by successively linearizing at the previous MPP and searching for the more "exact" MPP. An iteration algorithm can be used such that convergence is reached when the exact MPP is located.

3.6 AUTOMATIC DIFFERENTIATION

The determination of CDFs for implicit performance functions can be performed efficiently if the AMV method is performed with the aid of automatic differentiation. "Automatic differentiation is a technique for augmenting computer programs with derivative computations. It exploits the fact that every computer program, no matter how complicated, executes a sequence of elementary arithmetic operations such as additions or elementary functions such as EXP( )." (Berz, 1995).

The use of automatic differentiation to compute the derivatives in the first order Taylor's series can significantly reduce the number of actual computer model runs, as compared to other methods such as Runge-Kutta or finite differences. An automatic differentiation tool can accurately compute the derivatives of parameters of interest x, with respect to the performance function Z, i.e., dZ/dx, at various points of interest, with basically one computer model run. The General Sensitivity Methodology is a method that can be used to automate differentiation of large computer codes. This method has been proven to work on actual large scale computers model used in environmental transport and dose assessment (Worley, 1991). Specifically, GRESS and ADGEN has been used to enhance large-scale codes such as the 3-D finite difference transient geohydrology code SWENT (where derivatives of approximately 900 output parameters with respect to a selection of input variables) and the shallow-land waste disposal model PRESTO-II (where the derivatives and sensitivities for a few key parameters were calculated with respect to 69,000 data variables) (Worley, 1989). This method is described in the following Section, 3.6.1.

The AMV method is based on a first-order Taylor's series with an adjustment factor that accounts for the difference between Z_{MV} (the approximate performance function) and Z_{EXACT} (the actual performance function) at the MPP. This adjustment factor is not reliable when the actual performance function is non-linear. Since the factor is a deterministic (fixed) value, the updated approximate performance function might not be accurate within the domain of the new MPP. A more appropriate method is possible, whereby the difference between Z_{MV} and Z_{EXACT} is represented with the use of a second order term in the Taylor's series. The second order term can be obtained by using a combination of GRESS and the least squares method. The development of the second order term is described in Section 4.4.

3.6.1 General Sensitivity Methodology

Consider the following general set of non-linear equations

$$\mathbf{y} = F(\mathbf{y},\mathbf{c}) \qquad \textbf{(3-34)}$$

where **y** represents the dependent variable being solved for, **c** represents the user specified model data or parameter set, and **F** defines the model equations. This expression is generally used to represent equations in FORTRAN programs, where the calculated left-hand-side variables are a function of previously defined left-hand-side variables and data.

There are typically large numbers of components of the vector **y** in large computer models. However, the **y** vector components are generally used to calculate a much

smaller set of results. Such results can be defined as:

$$\mathbf{R} = \mathbf{h(y)} \tag{3-35}$$

where R is a single number which is a function of **y**. To calculate the derivatives of the results of interest with respect to any and all of the model data and input parameters, for a given computer model, the following analytical expression may be used,

$$\frac{dR}{d\alpha_i} = \frac{\partial h}{\partial \mathbf{y}} \frac{\mathbf{dy}}{d\alpha_i} \tag{3-36}$$

where the generic parameter α_i is used to denote any individual parameter. The total number of parameters is assumed to be M, such that the index on α_i will run from 1 to M.

The dependence of R on **y** through h(**y**) is defined analytically by the model user. Hence, only $d\mathbf{y}/d\alpha_i$ can be calculated by differentiating equation 3-37 as follows:

$$\frac{\mathbf{dy}}{d\alpha_i} = \frac{\partial \mathbf{F}}{\partial \mathbf{y}} \frac{\mathbf{dy}}{d\alpha_i} + \frac{\partial \mathbf{F}}{\partial \mathbf{c}} \frac{\mathbf{dc}}{d\alpha_i} \tag{3-37}$$

Equation 3-37 can be rearranged to yield the following set of coupled equations to solve for $d\mathbf{y}/d\alpha_i$,

$$\left(\mathbf{I} - \frac{\partial \mathbf{F}}{\partial \mathbf{y}}\right) \frac{\mathbf{dy}}{d\alpha_i} = \frac{\partial \mathbf{F}}{\partial \mathbf{c}} \frac{\mathbf{dc}}{d\alpha_i} \tag{3-38}$$

In a more compact form, equation 3-38 can be expressed as

$$\mathbf{A}\mathbf{y}_i' = \mathbf{s}_i, \qquad i = 1, \ldots, M \tag{3-39}$$

I is the identity matrix and A, $\mathbf{y}_i'$, and $\mathbf{s}_i$ are given by:

$$\mathbf{y}_i' \equiv \frac{\mathbf{dy}}{\mathbf{d}\alpha_i}$$

$$\mathbf{A} \equiv \mathbf{I} - \frac{\partial \mathbf{F}}{\partial \mathbf{y}}$$

and

$$\mathbf{s}_i \equiv \frac{\partial \mathbf{F}}{\partial \mathbf{c}} \frac{\mathbf{dc}}{\mathbf{d}\alpha_i} \tag{3-40}$$

The classical method known as the "direct" or "forward" approach involves solving equation 3-39 directly and using the results in equation 3-36 to evaluate $dR/d\alpha_i$. This requires that equation 3-39 be solved each time a new α_i is defined. Thus, the direct method is suitable for problems with relatively few input parameters of interest, or for problems where the solution of equation 3-39 is inexpensive when compared to the solution of the model itself.

For models where large numbers of input parameters of interest exist, the "adjoint" approach is a more efficient method for calculation $dR/d\alpha_i$ for many α_i. This method is based on the fact that equation 3-39 is linear in y_i' and appropriate adjoint equations can therefore be developed specifically to evaluate equation 3-36. If there are many output parameters relative to input parameters, the forward method may be preferable due to the large computer storage requirements.

The mathematical definition of an adjoint operator, where the matrix adjoint of A is A^*, is:

$$\int_v \mathbf{u}\ A\ \mathbf{v}\ dv = \int_v \mathbf{u}\ A^*\ \mathbf{v}\ dv \qquad \textbf{(3-41)}$$

or in inner product notation,

$$\langle\ \mathbf{u},\ A\ \mathbf{v}\ \rangle = \langle\ A^*\mathbf{u},\ \mathbf{v}\ \rangle \qquad \textbf{(3-42)}$$

or in vector notation,

$$\mathbf{u}^{tr}\ A\ \mathbf{v} = \mathbf{v}^{tr}\ A^*\ \mathbf{u} \qquad \textbf{(3-43)}$$

In the equation above, **u** and **v** are arbitrary vectors and A^* is defined as (when A is real):

$$A^* = A^{tr} \qquad \textbf{(3-44)}$$

Note that the tr superscript represents the transpose of the vector or matrix. The model

adjoint equation can be formulated as shown:

$$\mathbf{A}^{*}\mathbf{y}^{*} = \mathbf{s}^{*} \tag{3-45}$$

where

$$\mathbf{A}^{*} \equiv \mathbf{A}^{tr} = \left(I - \frac{\partial \mathbf{F}}{\partial \mathbf{y}} \right)^{tr} \tag{3-46}$$

Choosing $\mathbf{s}^{*}$ appropriately as

$$\mathbf{s}^{*} \equiv \left(\frac{dh}{dy} \right)^{tr} \tag{3-47}$$

then equation 3-36 can be evaluated as follows:

$$\frac{dR}{d\alpha_i} = \mathbf{y}^{*\,tr} \frac{\partial \mathbf{F}}{\partial \mathbf{c}} \frac{d\mathbf{c}}{d\alpha_i} \tag{3-48}$$

where $\mathbf{y}^{*}$ is now the solution to

$$\left(I - \frac{\partial \mathbf{F}}{\partial \mathbf{y}} \right)^{tr} \mathbf{y}^{*} = \left(\frac{dh}{dy} \right)^{tr} \tag{3-49}$$

or

$$\mathbf{y}^* = \left[\left(\mathbf{I} - \frac{\partial \mathbf{F}}{\partial \mathbf{y}}\right)^{tr}\right]^{-1}\left(\frac{\mathbf{dh}}{\mathbf{dy}}\right)^{tr} \qquad \textbf{(3-50)}$$

The adjoint method is both efficient and simple since equation 3-49 needs to be solved only once to obtain any and all model derivatives (or sensitivities). This is due to equation 3-49 being independent of the definition of α_i. The chosen data field parameters, α_i, are only used in equation 3-48, which involves simple vector products. Thus, it can be seen that the adjoint method significantly reduces the amount of computational effort required to evaluate $dR/d\alpha_i$ from solving many coupled linear equations to the evaluation of several vector products.

It should also be noted that the direct (forward) and adjoint (backward) model equations (equations 3-39 and 3-49) are far easier to solve than the original model equations (i.e., equation 3-37). Both equations 3-39 and 3-49 are linear while equation 3-37 is nonlinear. Use of the direct and adjoint methods, however, require the availability of the results to the original model equation since the A matrix and the vectors **s** and $\mathbf{s}^*$ depend on **y**.

3.6.2 GRESS

The Gradient Enhanced Software System (GRESS) is a FORTRAN language compiler developed to automate the calculation of derivatives needed for sensitivity

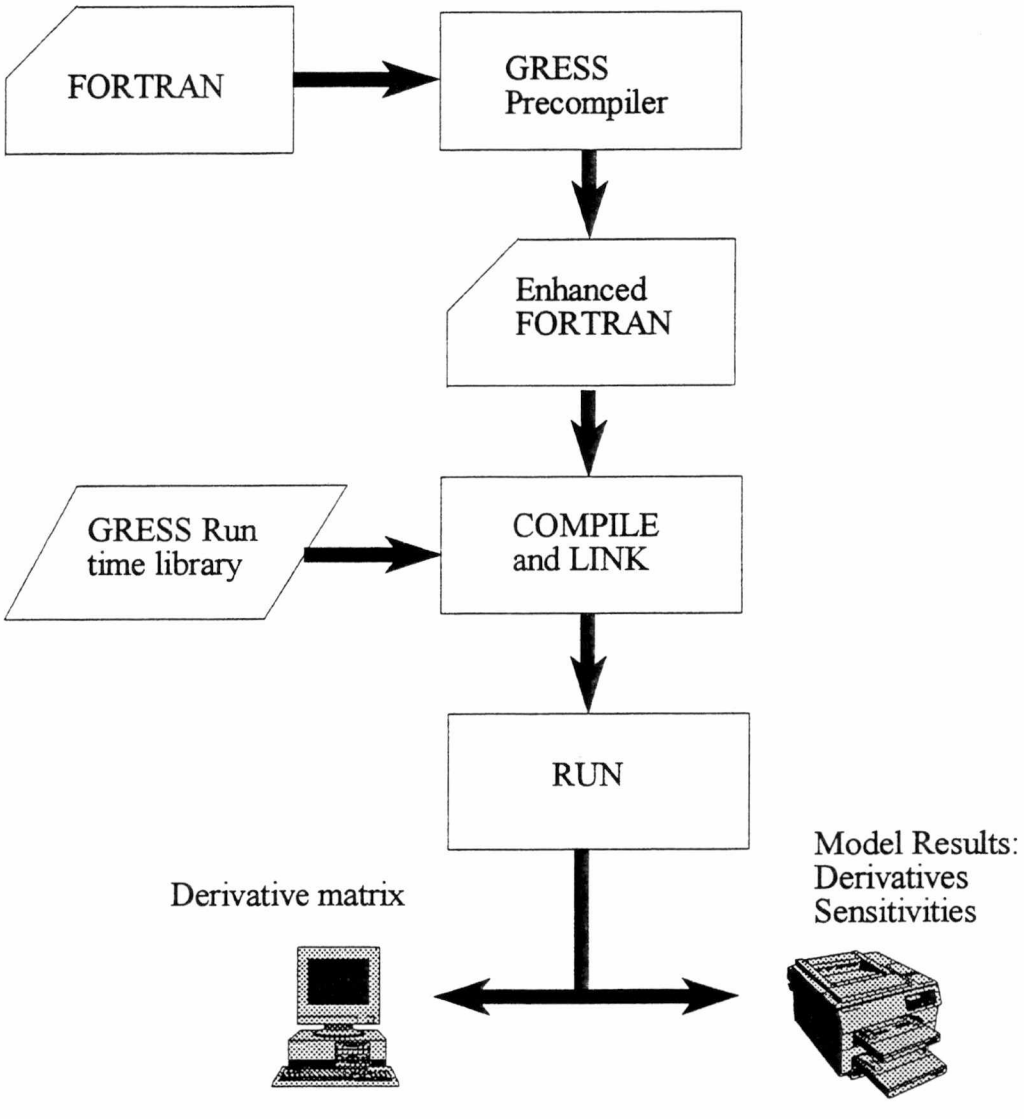
analysis of large scale computer modeling programs. In such programs, calculated variables are mathematical functions of previously defined variables and data. The mathematical functions are identified in FORTRAN programs by the "=" symbol and have a single dependent variable on the left-hand-side of the equation (i.e., in the form of equation 3.34). GRESS searches for and analyzes each "equation" in terms of its functional dependence on **y** and **c**. The derivatives $\partial\mathbf{F}/\partial\mathbf{c}$ and $\partial\mathbf{F}/\partial\mathbf{y}$ are computed for each equation (i.e., each component of **F**) and the results are stored in vector form for each component of **y**. The differentiation is carried out analytically using calculus software for all permissible FORTRAN functions and operators and the results are computed and stored numerically using the local values of the independent and dependent variables.

GRESS uses a FORTRAN precompiler (SYMG) to interpret FORTRAN statements and determine the mathematical operations embodied in them. As each arithmetic assignment statement in a program is interpreted, information necessary to allow the calculation of derivatives is generated. The result of the precompilation step is a new FORTRAN program that can produce derivatives for any REAL (i.e., single-or double-precision) variable calculated by the model. Consequently, GRESS enhances FORTRAN programs by adding the calculation of derivatives along with the original output. Derivatives from a GRESS- enhanced model can be used internally (e.g., for iteration acceleration) or externally (e.g., for sensitivity studies). GRESS accepts a majority of ANSI-X3.9 FORTRAN 77, including subroutines, common blocks, data

statements, read statements, user functions, intrinsic functions, block data subprograms, single-precision variables, double-precision variables, and equivalence statements. GRESS is available from the Radiation Shielding Information Center at Oak Ridge National Laboratory and is operational on VAX/VMS computer systems. Test versions of GRESS have been implemented on VAX/ULTRIX, CRAY/UNICOS, IBM RISC/6000 Workstations, and Sun Workstations; however, GRESS has only been rigorously tested in the VAX/VMS environment.

The steps used to process a code with GRESS are illustrated in Figure 3.4. A FORTRAN model is input to the GRESS precompiler to create an enhanced program. The enhanced model is compiled in the usual manner and then linked with a library of GRESS utility routines. When the enhanced model is executed, derivatives are calculated for each arithmetic assignment statement immediately before the statement is executed.

GRESS provides two methods for calculating derivatives. The CHAIN option calculates the derivatives of a variable with respect to a user-selected subset of the input data by repeated application of the chain rule in the forward mode. The CHAIN option calculates derivatives as the model is executing and is the recommended option when the user is only concerned with a very small number of input parameters. The ADGEN option incorporates the adjoint methods long used by nuclear engineers to calculate the derivatives of selected model responses with respect to thousands of input parameters.



Source: J. E. Horwedel, "GRESS Version 2.0 User's Manual," ORNL/TM-11951, Oak Ridge National Laboratory, November 1991, Oak Ridge, TN 37831.

Figure 3.4 Processing steps for a GRESS application

When the ADGEN option is chosen, partial derivatives for every floating-point assignment statement in the model are output to a data set or stored in memory. Matrix-solving routines are then used to calculate and report derivatives for selected results. The ADGEN option provides the user with the capability to calculate and report the derivatives of any calculated model result with respect to all data input to the model. An important advantage of the adjoint method over the forward method is that the derivatives of selected model results can be calculated with respect to thousands of input parameters at a cost comparable to that of executing only a few model runs. To approximate the same information by direct parameter perturbations would require separate model runs for each input parameter.

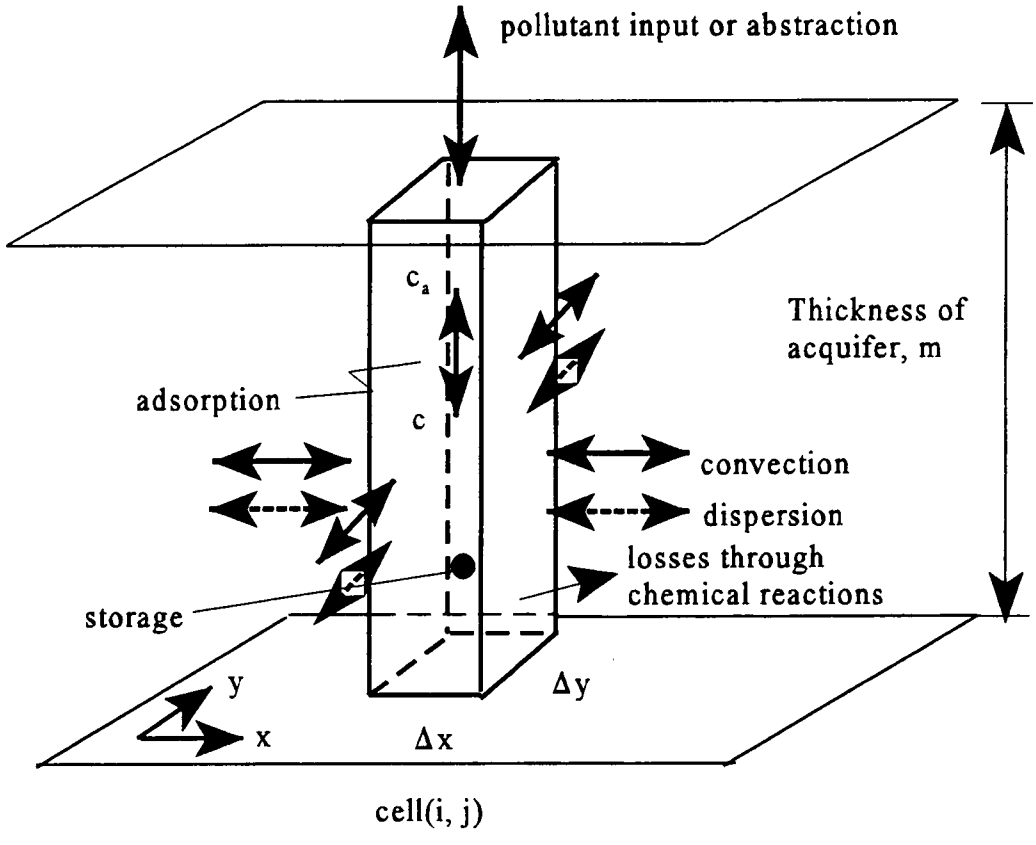
3.7 Finite Difference Method

Quantitative analysis of ground-water applications can be performed most generally using numerical models. Partial differential equations are used to mathematically express the concepts and laws (i.e., the concept of storage and Darcy's law) governing physical ground-water systems such as aquifers. Boundary conditions and initial conditions are specified for the partial differential equations. Numerical methods are then used to solve these partial differential equations. The numerical solutions normally involve approximating continuous (defined at every point) partial differential equations with a set of discrete equations in time and space. The region and time period of interest are divided up in some fashion, resulting in an equation or set of equations for each sub-region and time step. These discrete equations are combined to form a system of algebraic

equations that must be solved for each time step. The two major numerical techniques used in ground-water applications are the finite-difference and finite element methods. When numerical methods are used, three general characteristics are important to the solution procedure: (1) accuracy, (2) efficiency, and (3) stability. Accuracy deals with how well the discretized solution approximates the solution to the continuous problem it represents. Efficiency is a measure of how much computer work and computer resources are required to obtain a solution. Stability addresses the question of whether or not a solution is reliable with a particular choice of discretization parameters. While these definitions may be oversimplified, they are for most practical purposes sufficient. The reason many variations of numerical techniques exist is that each one was developed to improve at least one of these characteristics. No one combination of numerical technique and matrix solution can be considered superior for all applications. For most ground-water flow problems, the finite difference method is adequate. For sharp-front problems, the method of characteristics (MOC) or the finite element method will probably give better results. For deformation problems, the finite element method is better because of its treatment of tensorial parameters. For any given class of problems the choice of the best approach depends on the process being modeled, the accuracy desired, and the effort that can be expanded on obtaining a solution. For the purposes of testing the Limit State uncertainty method in this research, a finite difference method for solving ground-water contaminant transport is chosen. This choice is largely based on the simplicity of the model.

The finite difference method discretizes space and time in intervals. In two-dimensional space a grid of rectangular cells is laid over the modeled domain. For simplicity, only grids of constant grid-distances Δx and Δy in x- and y-directions are used. The center of a grid cell is again called a node. It is given indices (i, j) to indicate its position on the grid. As in the case of the flow equation, difference equations of transport can be obtained either by formal replacement of derivative by difference approximations on the grid or by taking a pollutant mass balance over each cell of the grid. Here, the latter procedure is preferred. The mass balance over the cell (i, j) (Figure 3.5) requires that over a time interval $[t, t+\Delta t]$ inputs and outputs due to storage, advection, dispersion diffusion and reaction add up to zero (Kinzelbach, 1986).

When computing the mass imported into the cell by advection, the velocity is combined with a concentration. This concentration can be approximated by a weighted average of the concentration of adjacent cells. Two choices of weight are common. If transport is dominated by convention, the weight is chosen on upstream side of the cell. This procedure is called up-winding. If dispersive transport dominates, symmetric weighting is preferred. The advective inputs into cell (i, j) over the time interval $[t, t+\Delta t]$



Source: Kinzelbach, Wolfgang, "Groundwater Modelling: An Introduction with Sample Programs in BASIC," page 258, Elsivier Science Publishers, N.Y., 1986.

Figure 3.5 Mass balance over cell c(i, j)

are then given by

$$\begin{aligned}
&\textbf{Convective input} = \\
&\quad u_{x,\,i-1,\,j}\,(\alpha c_{i-1,\,j} + (1 - \alpha)\, c_{ij})\,\Delta y\,(m_{i-1,j} + m_{ij})\,n_e\,\frac{\Delta t}{2} \\
&- u_{x,\,ij}\,(\beta c_{ij} + (1 - \beta)\, c_{i+1,\,j})\,\Delta y\,(m_{ij} + m_{i+1,j})\,n_e\,\frac{\Delta t}{2} \\
&+ u_{y,\,i,j-1}(\gamma c_{i,j-1} + (1 - \gamma) c_{i,\,j})\,\Delta x\,(m_{i,\,j-1} + m_{i,j})\,n_e\,\frac{\Delta t}{2} \\
&- u_{y,\,ij}(\delta c_{ij} + (1 - \delta)\, c_{i,j+1})\,\Delta x\,(m_{ij} + m_{i,\,j+1})\,n_e\,\frac{\Delta t}{2}
\end{aligned} \tag{3-51}$$

where m_{ij} is the saturated thickness of flow at node (i, j) and the upwind weights can be written in condensed form as:

$$\alpha = (1 + \text{sign}(u_{x,\,i-1,\,j})) / 2$$

$$\beta = (1 + \text{sign}(u_{x,\,ij})) / 2$$

$$\delta = (1 + \text{sign}(u_{y,\,ij})) / 2$$

$$\gamma = (1 + \text{sign}(u_{y,\,i,\,j-1})) / 2$$

while symmetric or central weighting is given by

$$\alpha = \beta = \gamma = \delta = 0.5$$

To discuss dispersive fluxes, first consider the case that flow is parallel to one of

the axes, e.g., to the x-axis. Then the net input into the cell by dispersion in x-direction can be calculated form concentrations $c_{i-1,j}$, $c_{i,j}$, and $c_{i+1,j}$. The net input is the difference of dispersive input at the boundaries between cells (i, j) and (i-1, j) and cells (i, j) and (i+1, j). Dispersive input in y-direction is calculated in a completely analogous fashion. The scheme is mass-conserving over the whole grid.

Dispersive input =

$$\begin{aligned} & D_{xx,\,i-1,\,j}\,(c_{i-1,\,j} - c_{ij})\,(\Delta y/\Delta x)\,(m_{i-1,j} + m_{ij})\,n_e\,\frac{\Delta t}{2} \\ & - D_{xx,\,ij}\,(c_{ij} + c_{i+1,\,j})\,(\Delta y/\Delta x)\,(m_{ij} + m_{i+1,j})\,n_e\,\frac{\Delta t}{2} \\ & + D_{yy,\,i,j-1}(c_{i,j-1} + c_{i,\,j})\,(\Delta x/\Delta y)\,(m_{i,\,j-1} + m_{i,j})\,n_e\,\frac{\Delta t}{2} \\ & - D_{yy,\,ij}(c_{ij} + c_{i,j+1})\,(\Delta x/\Delta y)\,(m_{i,\,j+1})\,n_e\,\frac{\Delta t}{2} \end{aligned} \qquad \textbf{(3-52)}$$

If flow is not parallel to any one of the coordinate axes, the concentrations at the four direct neighbors are not sufficient to calculate dispersive fluxes into cell (i, j). One possibility of formulating the two additional terms, which involves the off-diagonal elements of the dispersion tensor, is givenin Equation 3-53.

Dispersive input in the x-direction due to the concentration gradient in the y-direction =

$$\begin{aligned} & D_{xy,\,ij}\,(c_{i,j+1} - c_{i,j-1} + (c_{i+1,j+1} - c_{i+1,j-1})\,(m_{ij} + m_{i+1,j})\,n_e\,\frac{\Delta t}{8} \\ & - D_{xy,\,i-1,j}\,(c_{i-1,j+1} - c_{i-1,j-1} + (c_{i,j+1} - c_{i,j-1})\,(m_{i-1,\,j} + m_{ij})\,n_e\,\frac{\Delta t}{8} \end{aligned} \qquad \textbf{(3-53)}$$

Dispersive input in the y-direction due to the concentration gradient in the x-direction =

$$D_{yx,ij}\,(c_{i+1,j} - c_{i-1,j} + (c_{i+1,j+1} - c_{i-1,j+1})\,(m_{ij} + m_{i,j+1})\,n_e\,\frac{\Delta t}{8} \qquad \textbf{(3-54)}$$

$$- D_{yx,i,j-1}\,(c_{i+1,j-1} - c_{i-1,j-1} + (c_{i+1,j} - c_{i-1,j})\,(m_{i,j-1} + m_{ij})\,n_e\,\frac{\Delta t}{8}$$

In the calculation of dispersion coefficients on the grid, velocities between nodes or averages thereof are used. The calculations of coefficients $D_{xx,ij}$ and $D_{xy,ij}$ for dispersive transport between cell (i, j) and cell (i+1, j), for example, requires representative velocities U_x and U_y between nodes (i, j) and (i+1, j). For U_x, the internodal velocity $U_{x,ij}$ can be used directly. For U_y, a symmetric average over the four closet internodal velocities in y-direction is used. The dispersion coefficients in x-direction is obtained by:

$$D_{xx,ij} = (\alpha_L\,U_x^2 + \alpha_T\,U_y^2)/U$$

$$D_{xy,ij} = (\alpha_L + \alpha_T)\,U_x\,U_y/U \qquad \textbf{(3-55)}$$

where

$$U_x = u_{x,ij}$$

$$U_y = u_{y,ij} + u_{y,i+1,j} + u_{y,i,j-1} + u_{y,i+1,j-1}$$

$$U = \sqrt{U_x^2 + U_y^2}$$

The coefficients D_{yx} and D_{yy} for dispersive transport between cells (i, j) and

(i, j+1) are defined analogously:

$$D_{yy,ij} = (\alpha_T U_x^2 + \alpha_L U_y^2)/U$$

$$D_{yx,ij} = (\alpha_L - \alpha_T) U_x U_y / U \qquad \textbf{(3-56)}$$

where

$$U_x = u_{x,ij}$$

$$U_y = u_{y,ij} + u_{y,i+1,j} + u_{y,i,j-1} + u_{y,i+1,j-1}$$

$$U = \sqrt{U_x^2 + U_y^2}$$

The reaction term for a first order-reaction and the other source-sink terms do not contain derivatives. They convert directly into discrete values on the grid.

Input or loss sources, sinks =

$$-n_e \lambda \Delta x \Delta y m_{ij} c_{ij} \Delta t + q_{ij} \Delta x \Delta y c_{in,ij} \Delta t + S_{int,ij} \Delta x \Delta y \Delta t \qquad \textbf{(3-57)}$$

The storage term is given by

Stored pollutant mass =

$$n_e \Delta x \Delta y m_{ij} (c_{ij}(t + \Delta t) - c_{ij}(t)) \qquad \textbf{(3-58)}$$

Absorption according to a linear adsorption isothermal can be taken care of by dividing the pore velocities and the inflows by a retardation factor R. The sum of contributions from equations 3-51, 3-52, 3-53, 3-54, and 3-57 equal to storage (equation

3-58) for every cell (i, j). This results in the required system of linear equations for concentration $c_{ij}(t+\Delta t)$. These are only complete if initial and boundary conditions are specified. The time the concentrations appearing in equations 3-51, 3-52, 3-53, 3-54, and 3-57 are to be taken must also be specified. Setting

$$c_{ij} = \theta\,(c_{ij}\,(t + \Delta t) + (1 - \theta)\,c_{ij}(t) \qquad \textbf{(3-59)}$$

everywhere, fully implicit equations for $\theta = 1$, explicit equations for $\theta = 0$ and the central-in-time Crank-Nicholson-scheme for $\theta = 0.5$, are obtained.

Initial conditions do not pose any problems. Prescribed concentration boundaries lead to trivial nodal equations.

$$c_{ij}\,(t + \Delta t) = f(t + \Delta t) \qquad \textbf{(3-60)}$$

Impervious boundaries are introduced by setting dispersive boundary fluxes equal to zero. The convective flux will become zero due to vanishing normal velocity components at the boundary. Problems arise at boundaries where nonzero dispersive fluxes exist, as the concentration gradient at the boundary is usually not know. There are several possibilities to solve the problem: (1) Select a modeled domain so large that the concentrations at the boundary will not be influenced by the pollutant plume in the interior; then the boundary can be considered as a prescribed-concentration-boundary with concentration zero, (2) Neglect the dispersive flux at the boundary which is a reasonable approximation whenever the convective flux is larger than the dispersive flux, or (3) Using

the transmission-boundary-concept, approximate the concentration gradient across the boundary by the concentration gradient one grid distance further inside the modeled domain. This amounts to setting the dispersion coefficients along the relevant coordinated direction in the boundary cell equal zero. The convective flux at the boundaries is automatically taken care of by using an up-winding calculation procedure on the boundary as long as the velocity is directed towards the outside of the domain. If the convective flux is directed towards the inside of the domain, concentrations of the inflow must be specified.

If the explicit model equations ($\theta = 0$) are chosen, stability conditions must be satisfied. The first one is the Courant-criterion which requires that

$$Co_x = |\Delta t\, U_x / \Delta x| \leq 1 \qquad \textbf{(3-61)}$$

$$Co_y = |\Delta t\, U_y / \Delta x| \leq 1 \qquad \textbf{(3-62)}$$

or $Co_x + Co_y \leq 1$ in the case of upwind differences. Co is the Courant number. In physical terms the criterion says that not more pollutant mass can leave the cell via convention during the time interval $[t, t + \Delta t]$ than is inside it at the beginning of the time interval.

The second criterion is the Neumann-criterion. It takes care that concentration

gradients cannot be reversed by dispersion-diffusion fluxes alone. It has the form if one coordinate axis is parallel to flow.

$$D_{xx}\,\Delta t\,/\Delta x^2 + D_{yy}\,\Delta t\,/\Delta y^2 \le 0.5 \qquad \textbf{(3-63)}$$

A third criterion demands that a source and sink nodes not more pollutant mass can leave or enter the cell within a time interval than in contained in the cell originally. This requirement leads to the constraint.

$$\Delta t \le n_e\, m_{ij} / q_{ij} = m_{ij}\,\Delta x\,\Delta y / Q_{ij} \qquad \textbf{(3-64)}$$

A combination of the constraints (equations 3.61 through 3.64) decides upon the maximum length of time step that can be used. In the presence of reaction terms further stability criteria apply.

The equations of the explicit scheme can be solved one by one, while in implicit schemes a system of simultaneous equations must be solved. Implicit schemes are unconditionally stable. The equation system of the form

$$\begin{aligned} &A_{i,j}\,c_{i-1,j-1}(t+\Delta t) + B_{i,j}\,c_{i-1,j}(t+\Delta t) + C_{i,j}\,c_{i,j-1}(t+\Delta t) + \\ &D_{i,j}\,c_{i,j}(t+\Delta t) + E_{i,j}\,c_{i+1,j}(t+\Delta t) + F_{i,j}\,c_{i,j+1}(t+\Delta t) + \\ &G_{i,j}\,c_{i+1,j+1}(t+\Delta t) = H_{i,j} \end{aligned} \qquad \textbf{(3-65)}$$

with i = 1, ... , NX and j = 1, ... , NY, where the right hand side is a function of the concentrations at time t. The system can be solved by a direct equation solver such as the Gauss-Jordan method. An iterative method which can applied successfully is the IADI-procedure. Other iterative methods can be used as well, the convergence bing the better the more the dispersive terms dominate over the convective terms.

Although stability is unconditionally ful filled in implicit schemes, the accuracy of the results still depends (as in explicit schemes) on the choice of the discretization. Two types of numerical problems are common. One is overshoot (undershoot), the other is numerical dispersion.

The most accurate schemes use central differences in space and time. Central difference schemes tend, however, to oscillations. These can be avoided by sufficiently fine discretization. Up-wind-differences smooth out oscillations. They introduce, in turn, artificial dispersion.

CHAPTER 4

RESULTS

4.1 TRANSPORT MODEL AND PARAMETERS USED FOR TESTS

A two-dimensional groundwater pollutant transport model is selected to test the enhancements to the Limit State method using automatic differentiation and the least-squares method; hence the time required to solve the model equations is on the order of a few minutes. Use of the two-dimensional groundwater pollutant transport model allows for more comprehensive testing and development of the Limit State uncertainty analysis method for implicit performance functions. Owing to the model run times of a few minutes, results from a large number of runs can be assembled, thereby making it viable to compare the results obtained using the proposed methods versus results from the Monte Carlo and Latin Hypercube sampling method.

4.1.1 Finite Difference Groundwater Transport Model

The groundwater pollutant transport program uses an explicit difference method to solve the partial differential equations used in ground water transport. The weighting of the spatial difference scheme can alternatively be chosen as upwind or central. An initial concentration of zero is assumed everywhere. All pollution sources are modeled as permanent sources of constant source strength starting at time t = 0. Pollutant input does not influence the flow field. The ground water flow can be in the X or Y direction, or it can be a combination of both, depending on the values specified for the flow velocities,

UX and UY. Both velocity components, UX and UY, must be non-negative. For UX > 0 the southern boundary is a transmission boundary. For UY > 0 the southern boundary is a transmission boundary. For UX = 0 the southern and northern boundaries are assumed impervious. For UY = 0 the eastern and western boundaries are assumed impervious. Table 4.1 shows the allowable input parameters for the groundwater transport code. The output parameters are shown in Table 4.2. Figure 4.1 shows a 3-dimensional plot of the groundwater plume (X- and Y-axis versus pollutant concentration) at an arbitrarily chosen time step.

4.1.2 Test Parameters

For the tests conducted using the groundwater transport model, the parameters treated as random (uncertain) were the longitudinal and transverse dispersitivities, pore velocity in the x direction, thickness of flow, and the effective porosity. These parameters were treated as being lognormally distributed and were assigned values that are representative of conditions encountered on the U.S. Department of Energy (DOE) Oak Ridge reservation (Tauxe, 1997). The parameters and the mean and standard deviation are tabulated in Table 4.3.

The remaining parameters used in the groundwater transport model were treated as deterministic. These values are listed in Table 4.4.

Table 4.1 Input data

NX and NY	Number of nodes in x- and y- direction
DX and DY	Grid distances in x- and y- direction, in meters
NS	Number of pollution sources
AL and AQ	Longitudinal and transverse dispersitivities, AL and AQ, in meters
UX and UY	x- and y- pore velocities, in meters/day
M	Thickness of aquifer, in meters
NE	Effective porosity
LA	Decay constant of pollutant, in 1/day
TM	Maximum time of simulation, in days
DS	Type of difference scheme used for the convection term, DS = 1 for upwind differences, DS = 2 for central differences
IS(i) and JS(i)	Location of pollution sources, characterized by nodal indices, i=1, ... , NS
MP(i, j)	Strength of pollution sources, stored in an NX - by NY - array as MP(IS(i), JS(i)), i = 1, ... , NS, in pCi/day

Table 4.2 Output data

t	Time, in d
C(i, j)	Concentration distribution at time t and location i, j, in pCi/ml

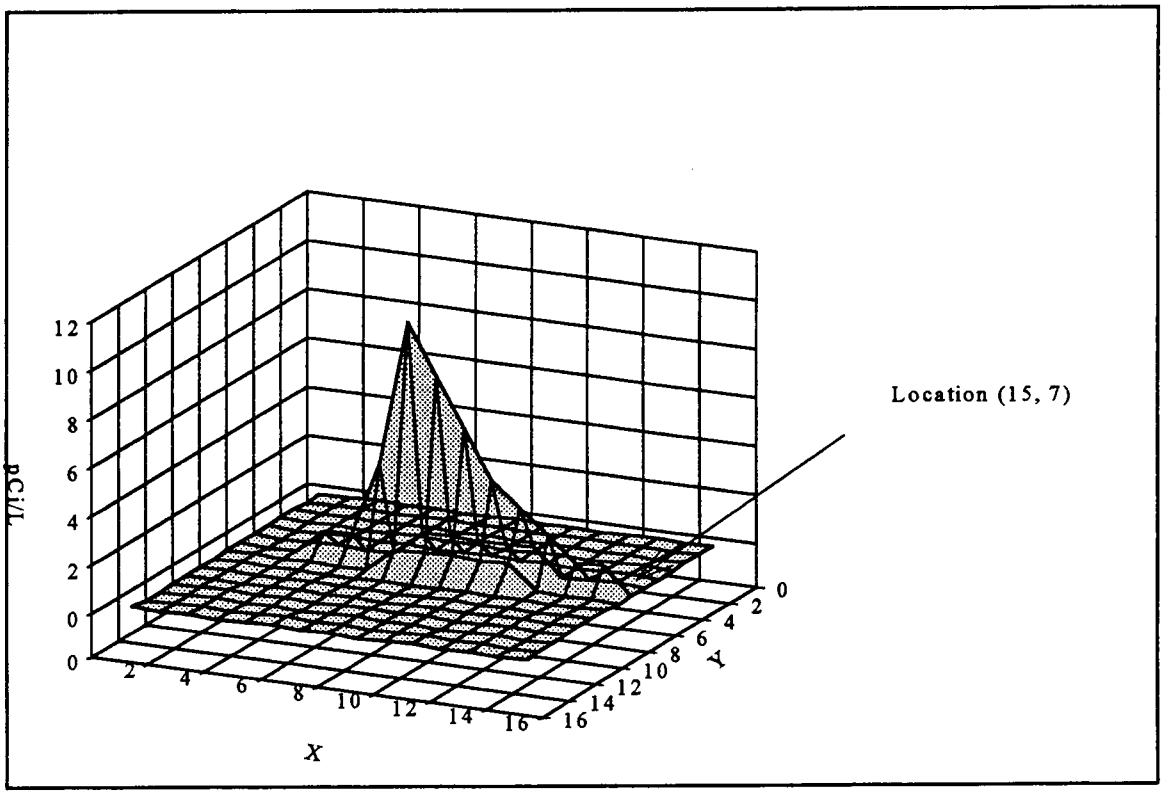


Figure 4.1 Concentration (pCi/L) gradient at t = 106 days

Table 4.3 Random test parameters used in the Groundwater Transport Model

Uncertain Parameter	Distribution	Geometric Mean	Standard Deviation
Longitudinal dispersitivity, AL	Lognormal	100 m	30 m
Transverse dispersitivity, AQ	Lognormal	10 m	3 m
Velocity, x direction, UX	Lognormal	3 m/day	1 m/day
Thickness of aquifer, M	Lognormal	3 m	3 m
Effective porosity, NE	Lognormal	0.2	0.04

Table 4.4 Deterministic parameters used in Groundwater Transport Model

Parameter	Value
Velocity, y direction, UY	0 m/day
Number of nodes in x - direction, NX	15
Number of nodes in y - direction, NY	15
Grid distances in x - direction, DX	100 m
Grid distances in y - direction, DY	100 m
Number of pollution sources, NS	1
Decay constant of pollutant, LA	0 day^{-1}
Maximum time of simulation, TM	600 days
Strength of pollution source MP(i, j)	10,000 pCi/L

4.2 UNCERTAINTY ANALYSIS USING A FIRST ORDER LIMIT STATE FUNCTION

4.2.1 Computation of First Order Limit State Function

A CDF for the Groundwater Transport model and the parameters described in Section 4.1 is computed. To exemplify the application of an automatic differentiation tool to reduce the number of computer model runs necessary to compute the CDF, GRESS is used.

A Limit State function is formulated by first computing the derivatives of the uncertain parameters at their mean values. This requires modifying the Groundwater Transport model to include the GRESS commands which specify which derivatives to compute and whether the direct or adjoint method is to be used. A precompilation step is then performed to generate the enhanced FORTRAN code. Next, the enhanced code is compiled and executed. With this one run of the Groundwater Transport model, the derivative information of interest is obtained for the mean input parameters. The first-order Limit State function that approximates the groundwater model at the region around the mean values of the uncertain variables is:

$$g(u) = a_0 + a_1 u_1 + a_2 u_2 + a_3 u_3 + a_4 u_4 + a_5 u_5 - z$$

The coefficients a_1, a_2, a_3, a_4, and a_5 are computed using GRESS. These coefficients are

the partial derivatives (or sensitivities) of the mean values of variables u_1, u_2, u_3, u_4, and u_5, with respect to the performance function computed at location C(15, 7) of the two-dimensional grid. Once the coefficients a_1 through a_5 have been computed, a_0 is determined by taking the difference between the sum of coefficients multiplied by the respective mean variable values with the value of C(15, 7) computed by the computer code. Inserting the coefficients a_0 through a_5 into the polynomial (and subtracting a limit state term, z) results in the following limit state function:

$$g(u) = 3.05E+02 - 1.84E-02\ u_1 - 3.06E-01\ u_2 - 3.36E+00\ u_3 - 5.17E-01\ u_4 - 2.59E+01\ u_5 - z$$

where:

u_1 is the standard normal variable for AL,

u_2 is the standard normal variable for AQ,

u_3 is the standard normal variable for UX,

u_4 is the standard normal variable for M,

u_5 is the standard normal variable for NE,

and,

z is the limit state value.

The probabilities of failure (p_f) based on this Limit State function at various limit state values (z) are then computed. A CDF is obtained by plotting the p_f values versus z.

4.2.2 Rackwitz Fiessler Algorithm for Limit State Uncertainty Analysis

A CDF is computed for the Groundwater Transport model using the Rackwitz Fiessler algorithm. The input parameters and their associated uncertainties are specified in Section 4.1.2. The Rackwitz Fiessler algorithm is used since it is a preferred technique for Limit State uncertainty analysis method. Probabilities of failure (p_f) are computed for limit state values ranging from 0 to 18 pCi/L, and intervals of 1 pCi/L, at location C(15, 7). The limit state values z, MPPs, the pollutant concentration at location C(15, 7), and the p_f (based on the Rackwitz Fiessler method) are tabulated in Table 4.5.

The CDF obtained using the Rackwitz Fiessler approach is shown in Figure 4.2. The CDF is constructed using the p_f values computed for each limit state (as shown in Table 4.5). One Groundwater Transport model run is required to setup the Limit State function used in the Rackwitz Fiessler approach. The setup time for running a GRESS enhanced FORTRAN code is approximately 30 minutes. No modifications are required to enable the use of GRESS with the Groundwater Transport Model. In contrast, standard methods used to compute the derivatives needed for formulating the Limit State function (e.g., numerical differentiation perturbation, or least squares) requires at least n+1 model runs. In this case, $n + 1 = 6$. As n gets larger, the other methods for computing derivatives becomes more computationally intensive, whereas only one model run would be required, regardless of n, in the GRESS approach.

Table 4.5 Most Probable Point and p_f values for the Rackwitz Fiessler approach

z	Most Probable Points					Concentration at C(15, 7) (pCi/L)	p_f
	NE	AL (m)	AQ (m)	UX (m/day)	M (m)		
0	0.2077	100.0114	10.3150	4.3072	10.9384	3.2781	0.0785
1	0.2065	99.5891	10.2381	4.0517	10.7836	3.5572	0.1116
2	0.2051	99.0850	10.1474	3.8033	10.6055	3.8797	0.1561
3	0.2035	98.4876	10.0413	3.5624	10.4028	4.2520	0.2143
4	0.2015	97.7848	9.9184	3.3297	10.1751	4.6817	0.2876
5	0.1993	96.9617	9.7767	3.1058	9.9208	5.1896	0.3759
6	0.1967	96.0063	9.6155	2.8909	9.6414	5.7584	0.4771
7	0.1938	94.9040	9.4335	2.6853	9.3376	6.4300	0.5858
8	0.1904	93.6408	9.2298	2.4891	9.0107	7.1935	0.6941
9	0.1867	92.2025	9.0038	2.3024	8.6625	8.1129	0.7929
10	0.1825	90.5737	8.7549	2.1253	8.2946	9.1744	0.8740
11	0.1779	88.7517	8.4853	1.9561	7.9153	10.3480	0.9326
12	0.1728	86.7040	8.1912	1.7974	7.5157	11.5987	0.9692
13	0.1672	84.4276	7.8750	1.6474	7.1041	12.7989	0.9884
14	0.1612	81.9512	7.5455	1.5009	6.7033	14.2601	0.9965
15	0.1547	79.1980	7.1900	1.3656	6.2804	15.5333	0.9992
16	0.1476	76.1473	6.8081	1.2434	5.8333	17.4924	0.9999
17	0.1400	72.8303	6.4085	1.1276	5.3853	17.3032	1.0000
18	0.1322	69.3343	6.0048	1.0068	4.9911	17.2550	1.0000

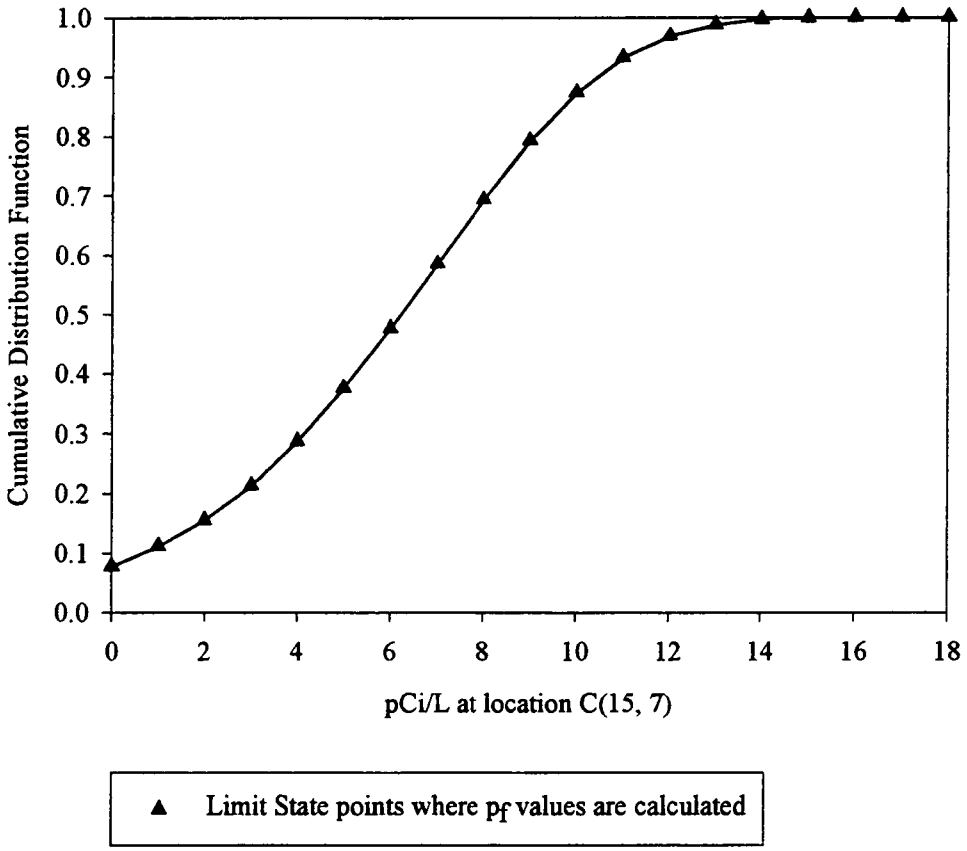


Figure 4.2 CDF of Groundwater Transport model using the Rackwitz Fiessler approach

4.2.3 Advanced Mean Value Approach for Limit State Uncertainty Analysis

In order to obtain a more accurate CDF, the AMV approach is tested. As described earlier, this approach improves on the Limit State function used by incorporating a "correction" term to account for the difference in results predicted by the Limit State function and the Groundwater Transport model at the MPPs. The derivatives used in the AMV approach are the ones computed using GRESS to formulate the Limit State function described in Section 4.2.1. Only one Groundwater Transport model run is required.

As with the CDF computation made with the Rackwitz Fiessler approach, p_f values were computed at selected limit state values. The AMV approach is performed with one iteration at each limit state value. Thus, for each p_f calculated using the one-iteration AMV approach, one run with the Groundwater Transport model is made to determine the difference in results computed by the actual model and the proxy Limit State function. The total number of model runs performed is 19. This total depends on the number of z points selected (i.e., total runs is m + 1, where m is the number of limit state points). The limit state values z, MPPs, the pollutant concentration at location C(15, 7), and the p_f (based on the one-iteration AMV method) are tabulated in Table 4.6.

Figure 4.3 presents the CDF computed from the one-interation AMV p_f results, plotted against the CDF obtained using the Rackwitz Fiessler approach.

Table 4.6 Most Probable Point and p_f values

for the Advanced Mean Value approach

z	Most Probable Points					Concentration at C(15, 7) (pCi/L)	p_f
	NE	AL (m)	AQ (m)	UX (m/day)	M (m)		
0	0.2077	100.0114	10.3150	4.3072	10.9384	3.2781	0.0223
1	0.2065	99.5891	10.2381	4.0517	10.7836	3.5573	0.0303
2	0.2051	99.0850	10.1474	3.8033	10.6055	3.8797	0.0820
3	0.2034	98.4876	10.0413	3.5624	10.4028	4.2520	0.1437
4	0.2015	97.7848	9.9184	3.3297	10.1751	4.6817	0.2359
5	0.1993	96.9617	9.7767	3.1059	9.9208	5.1896	0.3581
6	0.1967	96.0063	9.6155	2.8910	9.6414	5.7584	0.5029
7	0.1938	94.9040	9.4335	2.6853	9.3376	6.4300	0.6480
8	0.1904	93.6408	9.2298	2.4892	9.0107	7.1935	0.7750
9	0.1867	92.2025	9.0038	2.3024	8.6625	8.1129	0.8659
10	0.1825	93.6408	8.7549	2.1253	8.2947	9.1744	0.9241
11	0.1779	88.7517	8.4853	1.9561	7.9153	10.3480	0.9588
12	0.1728	86.7040	8.1920	1.7974	7.5157	11.5987	0.9787
13	0.1672	84.4276	7.8750	1.6474	7.1041	12.7989	0.9907
14	0.1612	81.9512	7.5455	1.5009	6.7033	14.2601	0.9976
15	0.1547	79.1980	7.1900	1.3660	6.2804	15.5333	0.9982
16	0.1476	76.1473	6.8081	1.2434	5.8333	17.4924	1.0000
17	0.1400	72.8303	6.4085	1.1276	5.3853	17.3032	1.0000
18	0.1322	69.3343	6.0048	1.0068	4.9911	17.2549	1.0000

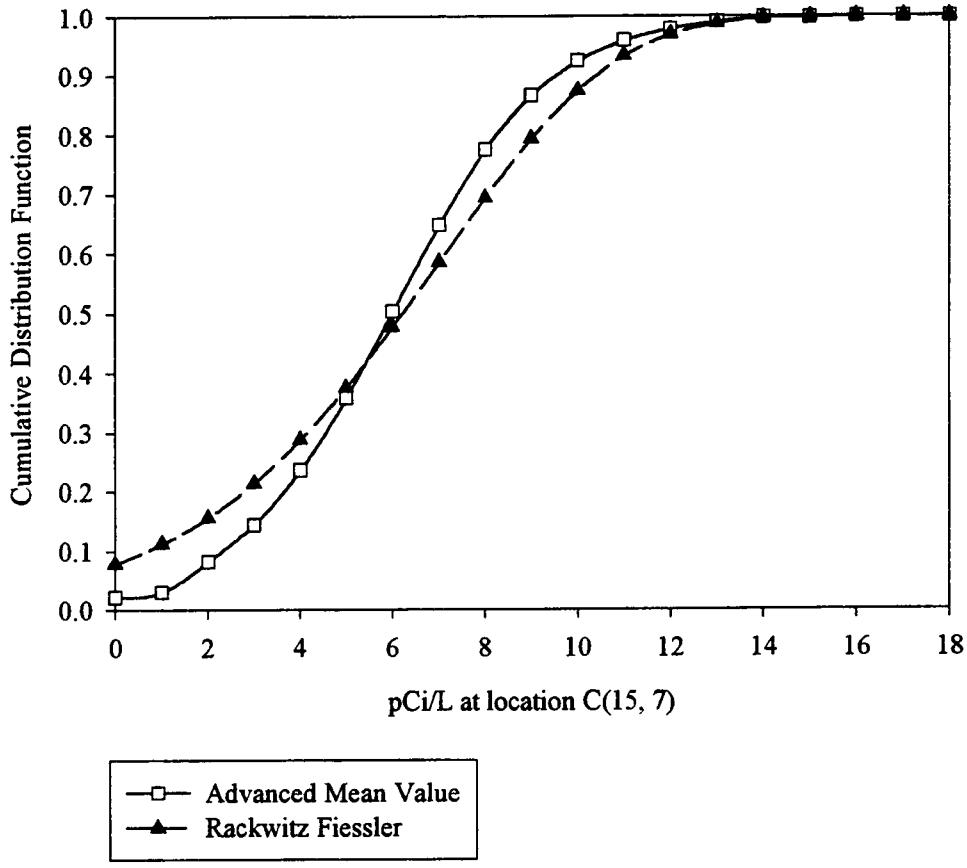


Figure 4.3 CDF of Groundwater Transport model using AMV

4.2.4 Most Probable Point Localized Function Approach for Limit State Uncertainty Analysis

The derivatives used to formulate the first-order Limit State function (in Section 4.2.1) were computed at the mean values for the uncertain input parameters. For a given set of MPPs, this Limit State function might not adequately model the actual performance function. In fact, a general concern with differential analysis methods for uncertainty analysis is that these methods are based on performance function behavior at a single point in the input variable space. Hence, the differential analysis methods are appropriate in cases where the input parameters are essentially fixed and the effects of small perturbations in the input parameters are being studied or the model is linear. However, in environmental risk assessment problems, this is usually not the case, since the uncertainties associated with the input values are often large.

Since the Limit State function formulated using partial derivatives computed for the mean input values is used in the Limit State method, the use of this performance function at MPPs located far away from the mean can lead to inaccurate CDF results. A proposed solution to this "local behavior" concern with the partial derivatives based Limit State performance function is to compute the partial derivatives at the MPPs. Given the ease of generating derivatives using GRESS, especially when the model FORTRAN code has been GRESS enhanced, derivatives for any set of MPPs can be generated with only one additional model run. Thus, the Limit State function formulated with mean based partial derivatives is used as the initial function. After the MPPs are identified, the Limit

State function is replaced with a new function valid for the set of MPPs. The method of "localizing" the Limit State function at the MPPs is outlined as follows:

1. Obtain the Limit State function, g(u), by using GRESS to compute the required derivative information.
2. Compute the CDF of g(u) at selected limit state points using the fast probability integration method, and assure that the CDF values cover a sufficiently wide probability range.
3. For each CDF value, identify the MPPs, x*.
4. GRESS the actual computer model at the MPPs to obtain a new set of derivatives.
5. Formulate a g(u) using the new derivative function.
6. Recompute the CDF of g(u) at selected limit state points using the fast probability integration method.

The preceding steps require one additional model run (during the GRESS phase) to compute a new set of derivatives for each limit state value. Hence, the total number of model runs is m + 1, where m is the number of limit state values used.

A CDF for the Groundwater Transport model is computed using the MPP localized function approach. The p_f values were computed at selected limit state values (at intervals of 2 pCi/L ranging from 0 to 18 pCi/L). The MPP localized function approach required one Groundwater Transport model run at each limit state value, in

order to reformulate the Limit State function. The total number of model runs performed is 19. This total depends on the number of z points selected (i.e., total runs is m + 1, where m is the number of limit state points). The limit state values z, MPPs, the pollutant concentration at location C(15, 7), and the p_f are tabulated in Table 4.7.

Figure 4.4 presents the CDF computed from the MPP localized function p_f results, plotted against the CDFs obtained using the Rackwitz Fiessler and AMV approach.

4.3 COMPARISONS WITH LATIN HYPERCUBE SAMPLING AND MONTE CARLO RESULTS

4.3.1 Generation of a CDF using Latin Hypercube sampling

Three CDFs are generated using the Latin Hypercube sampling method to be used for comparison with results obtained using the Limit State method. To generate the three CDFs, 3 sets of Latin Hypercube samples are generated. Each Latin Hypercube sample consisted of a n x k matrix, where n is the number of computer runs to be made and k is the number of random variables modeled. The value of k is 5, since there are five random variables modeled (i.e., AL, AQ, UX, M, NE). Three different values of n were used: 50, 100, and 500. The 50 computer runs Latin Hypercube sample is generated because it is the number of computer runs used in the uncertainty analysis portion of the ORNL Solid Waste Storage Area (SWSA) 6 Performance Assessment (Lee, 1997). The 100 and 500

Table 4.7 Most Probable Point and p_f values for the Most Probable Point localized function approach

z	Most Probable Points					Concentration at C(15, 7) (pCi/L)	p_f
	NE (unitless)	AL (m)	AQ (m)	UX (m/day)	M (m)		
0	0.2207	99.1070	11.1493	5.4263	12.7395	2.0285	0.0093
2	0.2134	99.0689	10.6766	4.4811	11.6546	2.8303	0.0491
4	0.2041	98.0363	10.0844	3.4973	10.4717	4.2701	0.2246
6	0.1961	95.7870	9.5789	2.8469	9.5794	5.8985	0.4996
8	0.1899	92.2705	9.1808	2.3847	8.9319	7.5087	0.7407
10	0.1851	87.7986	8.8631	2.0276	8.4361	8.9744	0.8912
12	0.1811	81.9715	8.5858	1.7110	8.0158	10.1595	0.9676
14	0.1772	74.7330	8.3259	1.4186	7.6125	10.6838	0.9944
16	0.1732	66.7872	8.0766	1.1629	7.1844	10.3760	0.9995
18	0.1687	56.4984	7.8278	0.9085	6.7390	6.9456	1.0000

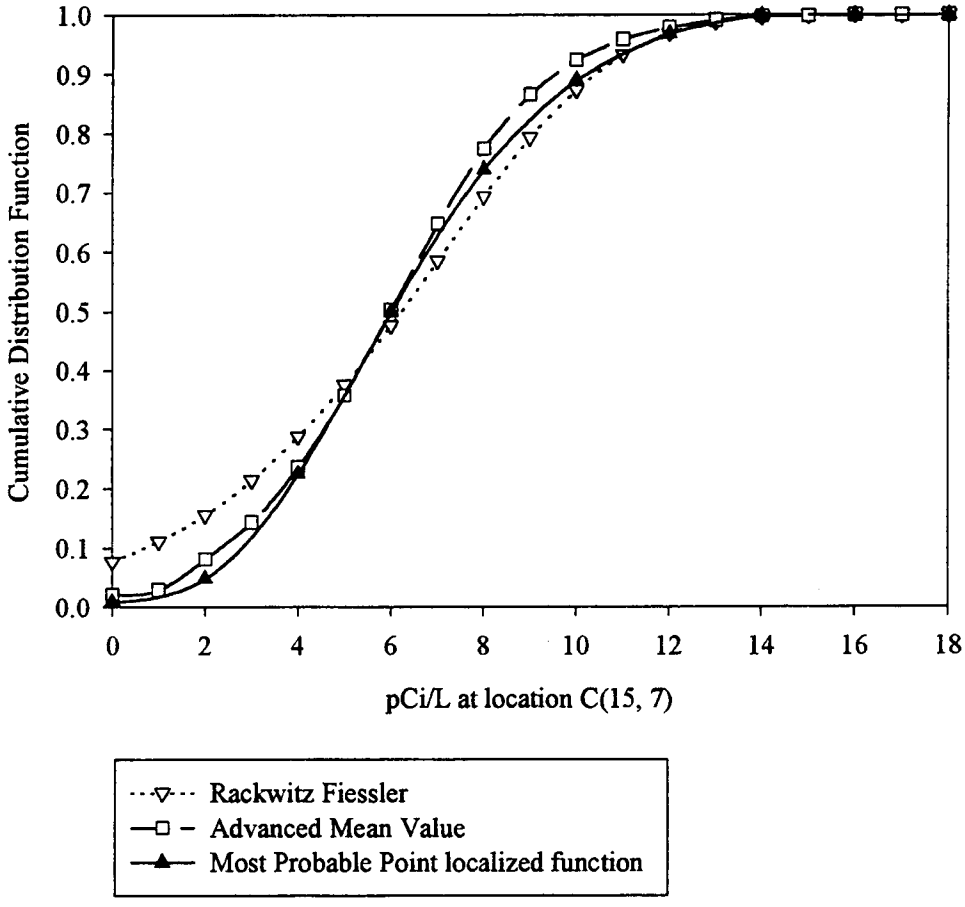


Figure 4.4 CDF of Groundwater Transport model using the MPP localized function

computer run Latin Hypercube samples were selected because they are commonly used sizes for Latin Hypercube samples.

The Latin Hypercube samples are generated using software developed by Ron Iman of Sandia National Laboratory (Iman, 1984). The three sample sets were then used as input to the Groundwater Transport model. The output from the Groundwater Transport model runs are then used to generate CDFs. The three CDFs are shown in Figure 4.5. The 500 sample Latin Hypercube curve is the smoothest and the most accurate of the three plots.

4.3.2 Generation of a CDF using Monte Carlo

A CDF of the output from the Groundwater Transport model is generated using a 10,000 run Monte Carlo simulation. This CDF is considered to be the best estimate of the actual CDF for the model output given the selected set of input variables. The methods available for computing the precision of an estimated CDF indicate that 10,000 runs are required to obtain a 95% confidence interval for the least precise estimated percentile (the 50^{th} percentile) to be plus or minus one estimated percentile (Morgan, 1990). The input for the Monte Carlo simulation is created using Crystal Ball. The 10,000 values for each random input variable are generated based on each variable's probability distribution, mean, and standard distribution. The 10,000 Monte Carlo run CDF is shown in Figure 4.6.

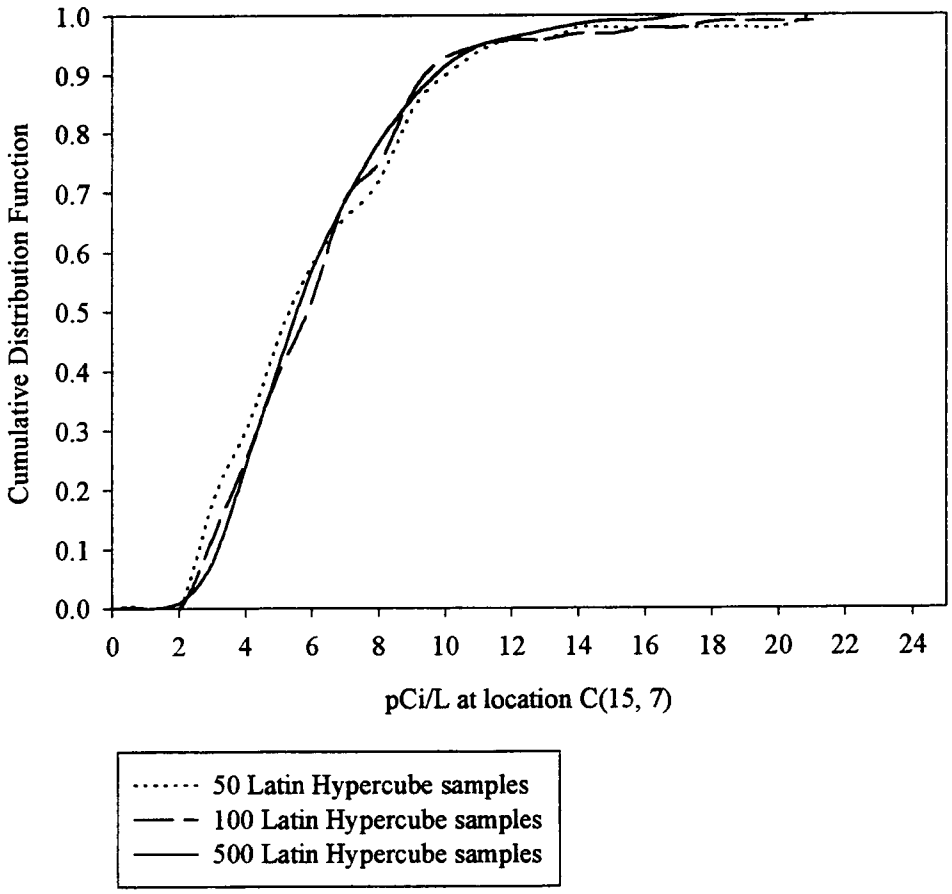


Figure 4.5 CDF of Groundwater Transport model using 50, 100, and 500 Latin Hypercube samples

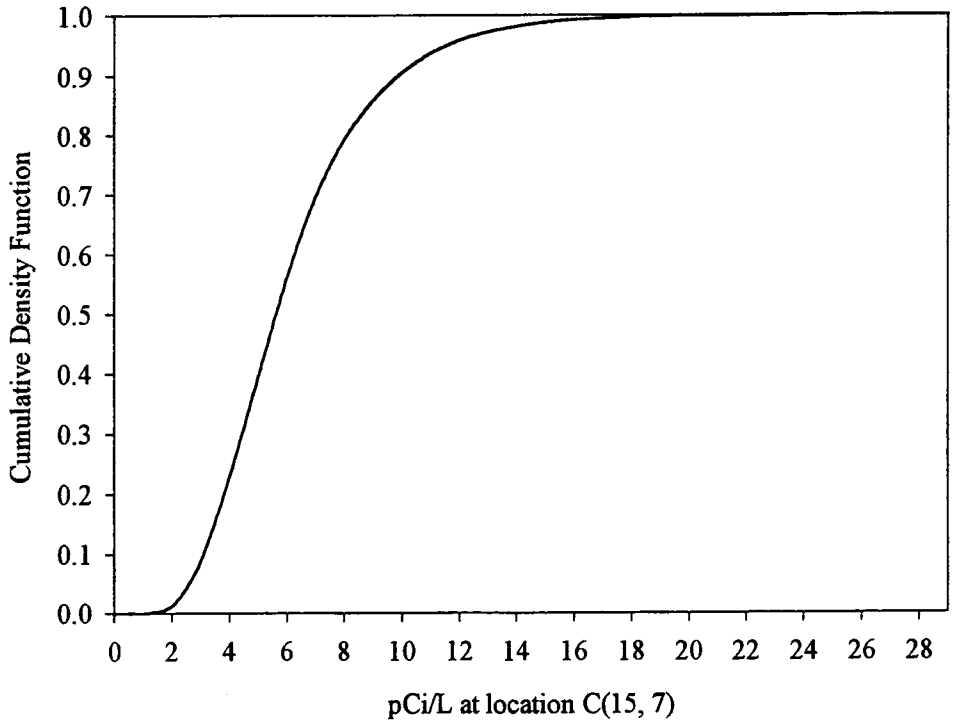


Figure 4.6 CDF of Groundwater Transport model using 10,000 Monte Carlo runs

4.3.3 Comparison of Results

The CDFs generated using different uncertainty analysis methods are compared in this section. Table 4.8, containing the pCi/L versus CDF data is provided to allow for comparison of CDF values obtained at selected limit state points (where p_f values were evaluated). Note that the CDF plots of the Latin Hypercube and Monte Carlo results were made using the complete set of CDF values, i.e., the plots were made using all 50, 100, 500, or 10,000 data points.

Figure 4.7 shows the CDFs obtained using the RF, AMV and MPP localized function approaches plotted against the 10,000 run Monte Carlo CDF. The Rackwitz Fiessler CDF does not compare as well as either the AMV or MPP localized function CDFs. The level of agreement between the Rackwitz Fiessler CDF and the other two Limit Sate CDFs is poorest at both tails of the CDF. This is because the Rackwitz Fiessler CDF is generated from a first order Limit State function based on partial derivatives computed at the means of the random input variables. The use of the partial derivatives results in "local behavior" and exemplifies the weakness of using the partial derivatives based Limit State function for global computation of the CDF. In contrast, the AMV and MPP localized function CDFs correct for the truncation error resulting from the use of a first order function. This correction, which represents a simple replacement for the higher order terms neglected by the first order function, results in better accuracy at the the tails of the CDFs as compared to the 10,000 run Monte Carlo CDF.

Table 4.8 CDF values computed using various uncertainty analysis methods

pCi/L	LS-RF CDF	LS-AMV CDF	LS-MPPLF CDF	50 LHS CDF	100 LHS CDF	500 LHS CDF	10,000 run MC CDF
0	0.078	0.022	0.009	0.00	0.00	0	0.000
1	0.112	0.030		0.00	0.00	0.000	0.000
2	0.156	0.082	0.049	0.00	0.00	0.008	0.013
3	0.214	0.144		0.18	0.12	0.054	0.090
4	0.288	0.236	0.225	0.30	0.25	0.210	0.231
5	0.376	0.358		0.46	0.40	0.368	0.398
6	0.477	0.503	0.500	0.58	0.52	0.535	0.564
7	0.586	0.648		0.66	0.69	0.648	0.698
8	0.694	0.775	0.740	0.72	0.75	0.759	0.794
9	0.793	0.866		0.84	0.87	0.836	0.860
10	0.874	0.924	0.890	0.90	0.93	0.896	0.907
11	0.933	0.959		0.94	0.95	0.935	0.939
12	0.969	0.979	0.968	0.96	0.96	0.959	0.960
13	0.988	0.991		0.96	0.96	0.971	0.973
14	0.997	0.998	0.994	0.98	0.97	0.979	0.982
15	0.999	0.998		0.98	0.97	0.990	0.989
16	1.000	1.000	0.999	0.98	0.98	0.993	0.993
17	1.000	1.000		0.98	0.98	0.998	0.995
18	1.000	1.000	1.000	0.98	0.99	1.000	0.997

CDF = Cumulative Distribution Function, LS = Limit State, RF = Rackwitz Fiessler, AMV = Advanced Mean Value, MPPLF = Most Probable Point localized function, LHS = Latin Hypercube sampling, MC = Monte Carlo

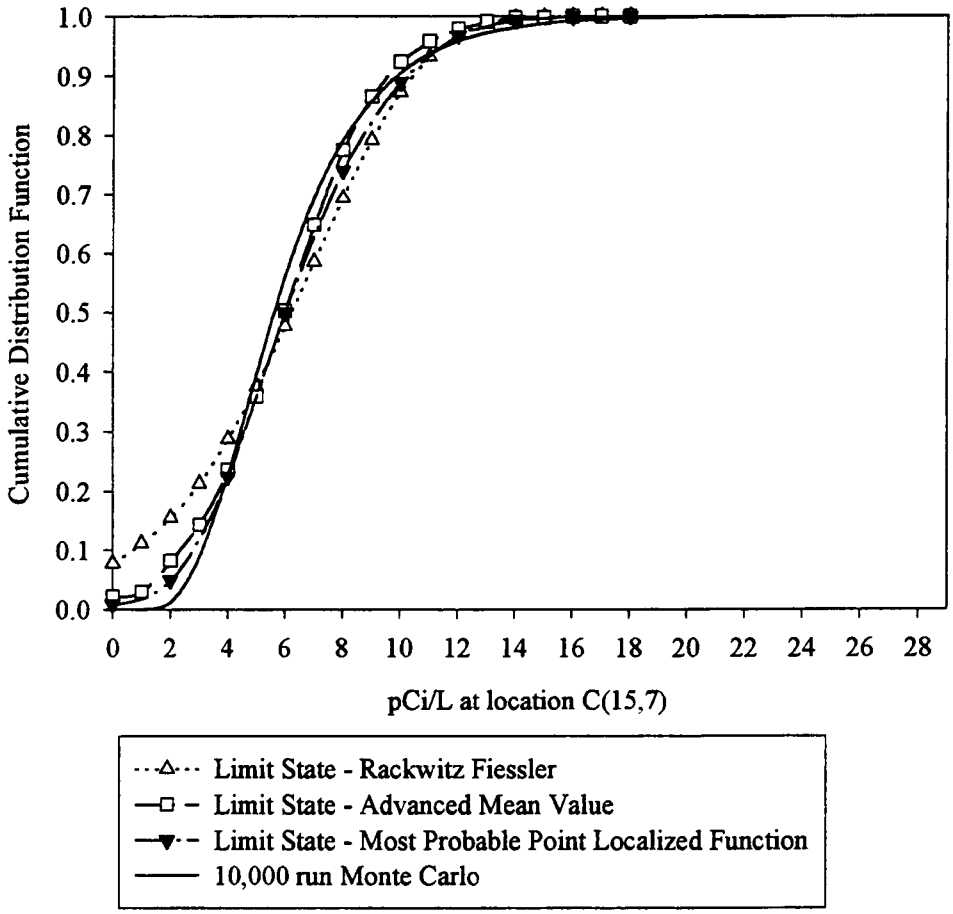


Figure 4.7 RF, AMV and MPP localized function CDFs versus the 10,000 run Monte Carlo CDF

Figure 4.8 shows the CDFs obtained using AMV and MPP plotted against the 50 run Latin Hypercube sample CDFs and the 10,000 run Monte Carlo CDF. From the plot, it can be seen that the 50 run CDF is not very accurate in the tail regions. The reason for this inaccuracy is illustrated in Figure 4.9, which is a probability density function (PDF) of the output from the Groundwater Transport model. From this plot, it can be seen that most of the output is clustered between 3 pCi/L and 11 pCi/L. Outside of this range, there is one "hit" at 1, 2 and 16 pCi/L respectively. This is the reason why the CDF cuts-off suddenly at both tails, suggesting that a Latin Hypercube sample with n = 50 is too small. When compared to the 10,000 run Monte Carlo CDF, the CDFs from the AMV and MPP localized function show better agreement than the 50 run Latin Hypercube sample CDF (especially at the tail regions), despite only requiring 19 runs of the actual computer model.

Figure 4.10 shows the CDFs obtained using AMV and MPP localized function plotted against the 100 run Latin Hypercube sample CDF and the 10,000 run Monte Carlo CDF. The 100 run Latin Hypercube sample CDF appears to be more accurate than the 50 run Latin Hypercube sample CDF (as compared to the 10,000 run Monte Carlo CDF). The tail region of this CDF does not cut off as sharply as in the case where n = 50. The AMV and MPP localized function CDFs compare favorably with the 100 run Latin Hypercube sample CDF. The PDF of the output from 100 runs of the Groundwater Transport model is shown in Figure 4.11. This PDF appears to be more lognormally distributed than the PDF of the 50 Latin Hypercube runs shown in Figure 4.9. There is a

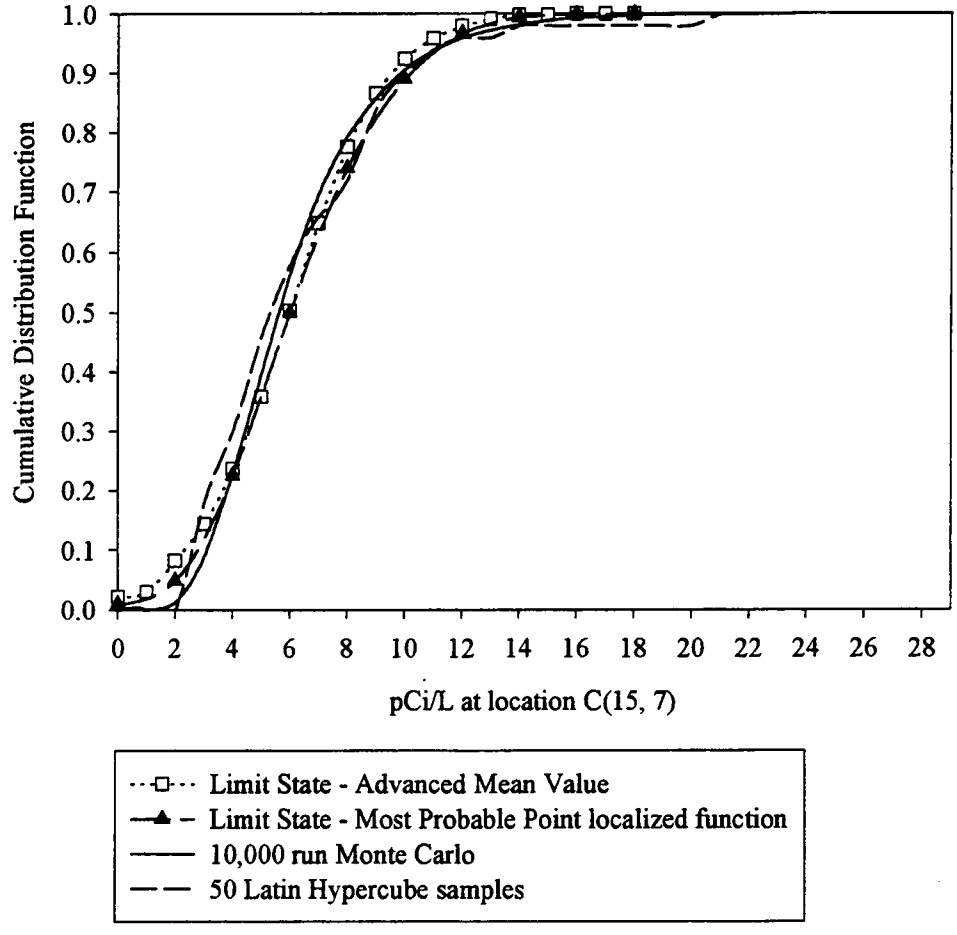


Figure 4.8 AMV and MPP localized function CDFs versus the 50 Latin Hypercube sample and 10,000 run Monte Carlo CDF

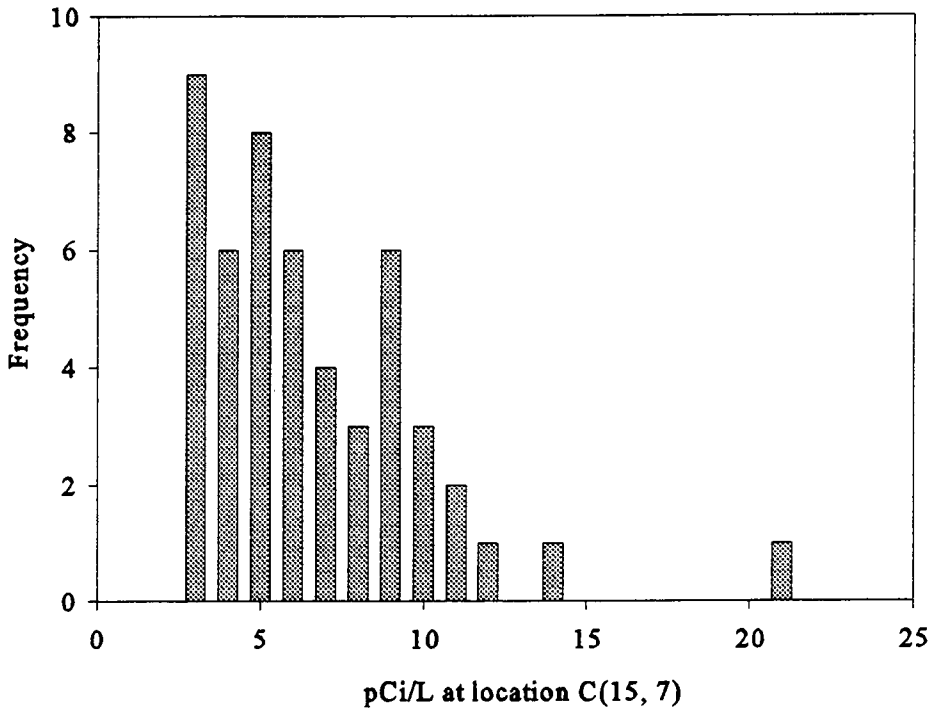


Figure 4.9 PDF of Groundwater Model output from 50 Latin Hypercube sample runs

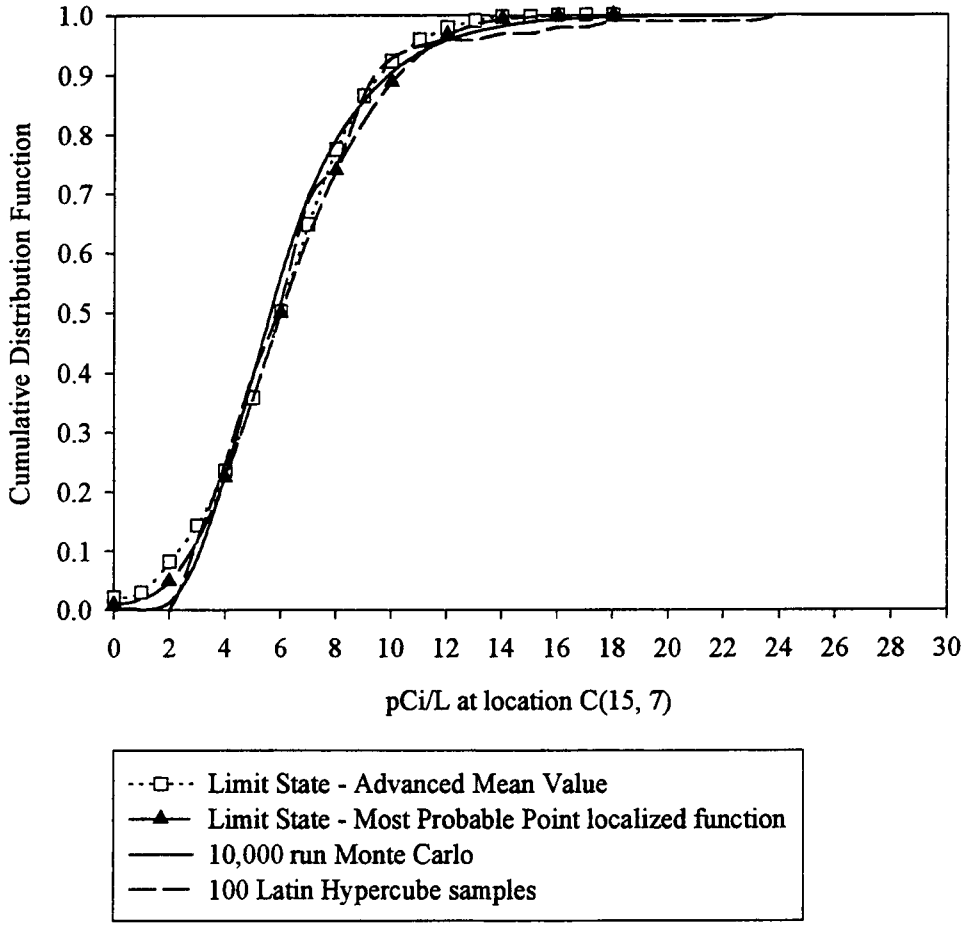


Figure 4.10 AMV and MPP localized function CDFs versus the 100 Latin Hypercube sample and 10,000 run Monte Carlo CDF

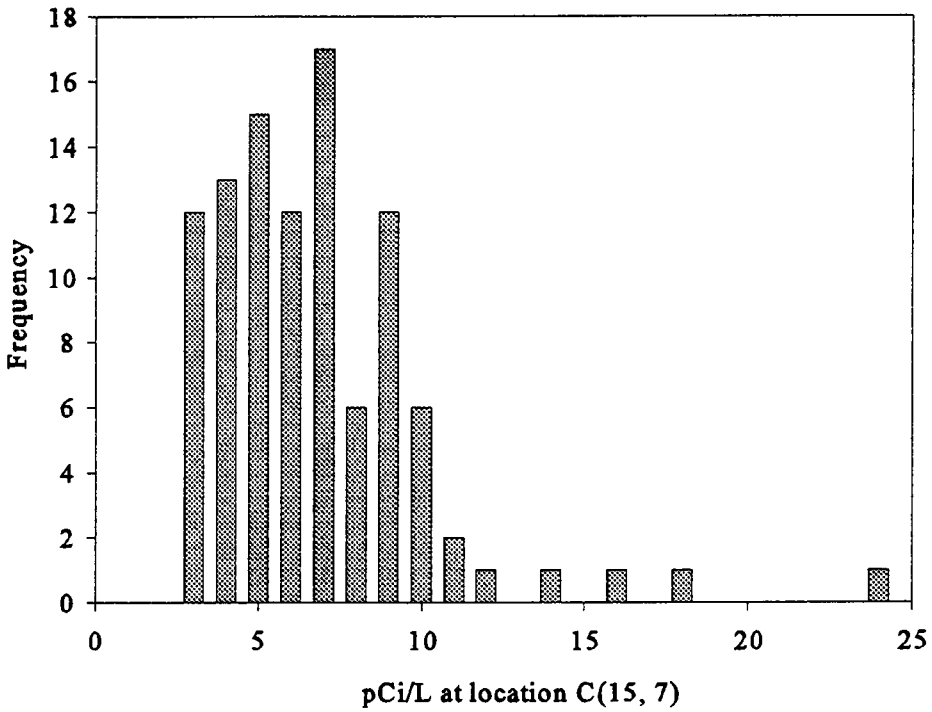


Figure 4.11 PDF of Groundwater Model output from 100 Latin Hypercube Sample runs

sharp cutoff below 3 pCi/L. Beyond the 11 pCi/L level, the histogram consists of sporadic single frequency results ranging up to 24 pCi/L.

Figure 4.12 shows the CDFs obtained using AMV and MPP plotted against the 500 run Latin Hypercube sample CDFs and the 10,000 run Monte Carlo CDF. The 500 run Latin Hypercube sample CDF is in close agreement with the 10,000 run Monte Carlo CDF and is more accurate than the AMV and Rackwitz Fiessler CDFs. The PDF of the output from 500 runs of the Groundwater Transport model is shown in Figure 4.13. This PDF is lognormally distributed, ranging from 2 pCi/L to 17 pCi/L with a maximum frequency occuring at 5 pCi/L. As shown in Figure 4.12, this PDF results in a more accurate CDF relative to the results obtained using Latin Hypercube samples of 50 and 100 runs respectively.

4.4 Second Order Limit State Function

An approach using a second order Limit State function is described in this section. As an extension of the AMV method, the intent is to replace the deterministic function $H(Z_{MV})$ (see Section 3.5) with a function consisting of second-order terms for the Limit State function (without the mixed terms). The replacement function is:

$$H(Z_{MV}) = \frac{1}{2} \sum_{i=1}^{n} b_i X_i^2 , \qquad (4\text{-}1)$$

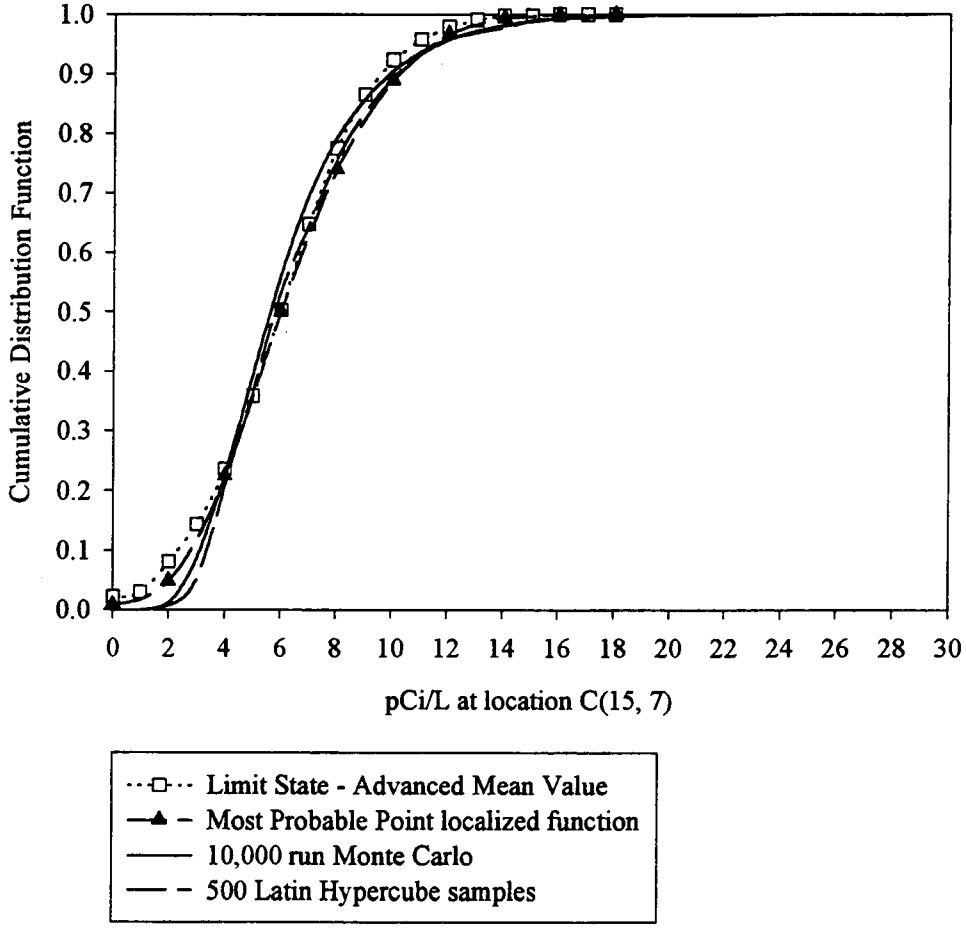


Figure 4.12 AMV and MPP localized function CDFs versus the 500 Latin Hypercube sample and 10,000 run Monte Carlo CDF

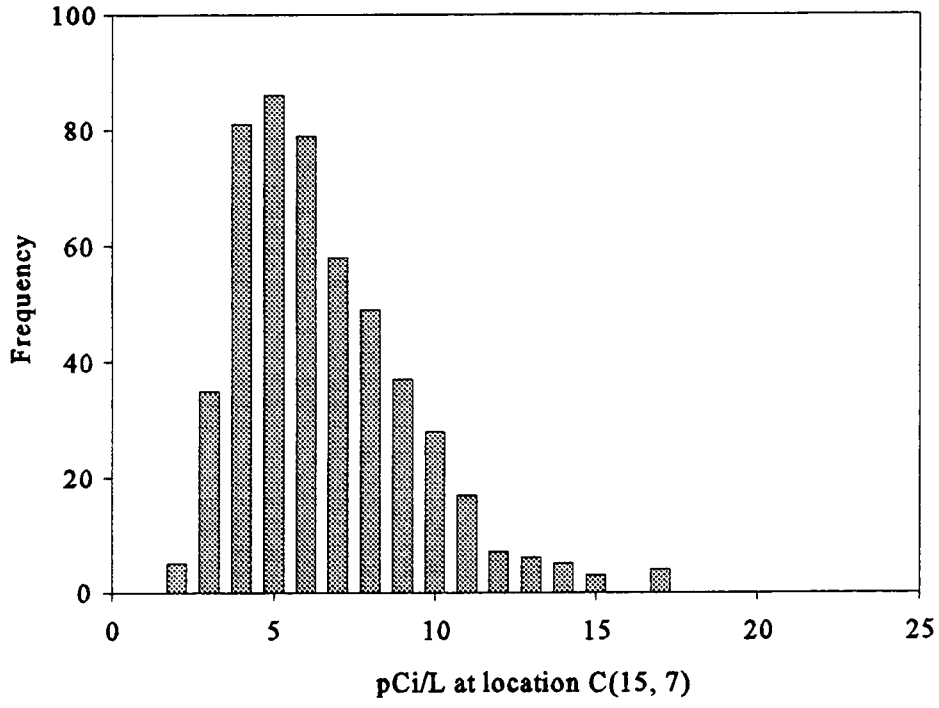


Figure 4.13 PDF of Groundwater Model output from 500 Latin Hypercube Sample runs

where

$$\mathbf{b_i} = \frac{\partial^2 Z}{\partial X_i^2} \cdot \tag{4-2}$$

The coefficient b_i can be solved by setting Equation 4-1 equal to $H(Z_{MV})$, which is defined as the difference between Z_{MV} and Z_{EXACT}. If the number of coefficients b_i is n, at least n+1 equations will have to be solved to compute the values of b_i. Since five random variables are used with the Finite Difference Groundwater Transport model, n = 5 and at least six equations will be required to solve for the b_i values.

The six equations needed to solve for b_i are obtained by perturbing the MPPs derived using the conventional Rackwitz-Fiessler algorithm. The MPP value for each random variable is perturbed by an arbitrary 10%, as shown in the Table 4.9.

Table 4.9 Example of Most Probable Point Data Perturbation

	Random Variables				
	AL (m)	AQ (m)	UX (m/day)	M (m)	NE (unitless)
Most Probable Points	93.641	9.230	2.489	9.011	0.190
AL increased by 10%	**103.005**	9.230	2.489	9.011	0.190
AQ increased by 10%	93.641	**10.153**	2.489	9.011	0.190
UX increased by 10%	93.641	9.230	**2.738**	9.011	0.190
M increased by 10%	93.641	9.230	2.489	**9.912**	0.190
NE increased by 10%	93.641	9.230	2.489	9.011	**0.209**

A run is made with the Finite Difference Groundwater Transport model for each of the six perturbed MPP data sets. This results in six values of Z_{EXACT}. The six $H(Z_{MV})$ values are computed by subtracting the Z_{MV} (computed using the first-order Limit State function) from the Z_{EXACT}. Next, the five b_i coefficients are solved by inserting the perturbed data set and $H(Z_{MV})$ values into Equation 4-1 and using the method of least-squares.

Once the b_i coefficients are known, the second-order Limit State function can be expressed as:

$$Z = a_o + \sum_{i=1}^{n} a_i X_i + \frac{1}{2} \sum_{i=1}^{n} b_i X_i^2 \qquad (4\text{-}3)$$

The CDF of Z can then be computed using fast probability integration at selected limit states. The CDF results, computed using the same input set and test functions used in Section 4.2, are shown compared to that obtained using a 10,000 run Monte Carlo CDF in Figure 4.14. The CDF is not as accurate as the ones derived using the first-order Limit State functions. Beyond the concentration of 11 pCi/L, the CDF curve begins to behave erratically. The reason for the lack of accuracy and instability of the second-order Limit State result is not known at present. Further investigation and research is needed in this matter.

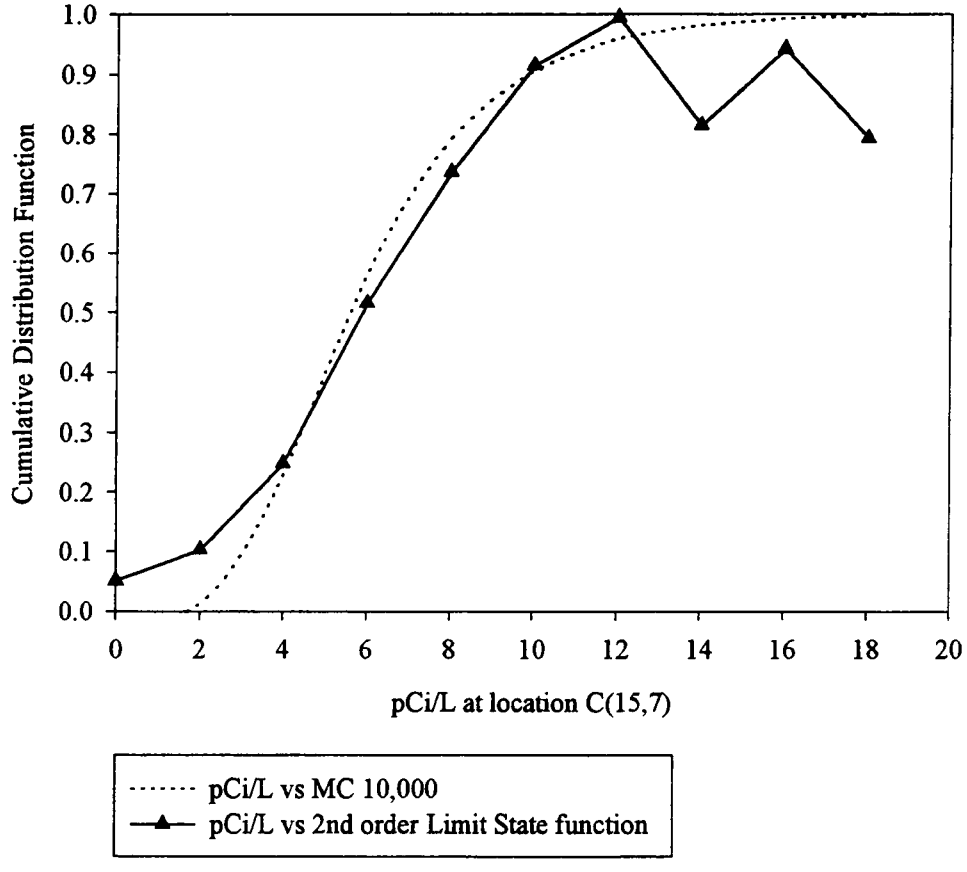


Figure 4.14 Second Order Limit State Function CDF versus Monte Carlo CDF

4.5 Discussion of Computation and Human Time Needed for the Limit State Method

The computer and human time required to perform uncertainty analysis using the Limit State method is discussed in this section. The results in the preceeding sections have demonstrated that comparably accurate results with significantly less computational effort can be achieved using the Limit State method as opposed to the Latin Hypercube method. The CDF results obtained using the AMV and MPP localized function Limit State approaches are shown to be more accurate than the 50 Latin Hypercube samples CDF and comparable with the 100 and 500 Latin Hypercube samples CDFs. The number of computer runs for each approach is presented in Table 4.10.

Table 4.10 Comparison of computer time requirements

Method	Number of runs[1]	Estimated computer time per run[2]	Total computer time
AMV	20	1 minutes	20 minutes
MPP localized function	20	1 minutes	20 minutes
Latinhypercube	100	1 minutes	100 minutes
Latinhypercube	500	1 minutes	500 minutes

[1] Limit State method runs are based on 19 CDF "limit state" points

[2] A personal computer with a 200 MHz Pentium microprocessor is used to perform the calculations.

A comparison of the human effort required to perform uncertainty analysis with each of the different approaches is a little more difficult. Using GRESS to compute partial derivatives requires some front-end preparation of the FORTRAN code. At a minimum, GRESS commands have to be inserted into the FORTRAN code and the enhanced code has to be precompiled using a GRESS precompiler. In some cases, code modifications have to be performed. A typical modification is to shorten FORTRAN lines since GRESS can only handle lines of a specified length. This is in itself a simple and straightforward process. The good news is that any modifications that need to be made to the FORTRAN program itself only have to be made once. No modifications were required for the Groundwater Transport model and editing the code to include the GRESS commands took less than 15 minutes. From there on, partial derivatives are computed by running the code once. Interviews with users of GRESS have indicated no major problems with preparing computer codes for use with GRESS.

While no code modifications are required when using the Latin Hypercube sampling method for uncertainty analysis, human effort is required in preparing and handling the input and output files and data from the numerous computer runs. To increase efficiency and to save time, modifications to a code's input/output functions have to be made or additional programs have to be written. This requires a lot of human effort. Another aspect of using the Latin Hypercube sampling approach is that, oftentimes, several sets of runs have to be made before desirable results are achieved. These sets of runs are typically made when variables or information used in a computer code are

changed or when mistakes are made. The implications of changes or mistakes are that the actual number of runs made is increased several fold.

Finally, a note regarding the Monte Carlo method. Uncertainty analyses performed using Monte Carlo require large numbers of runs, usually in the range of thousands of runs. The precision of estimates of the output distribution can be estimated using standard statistical techniques. The precision of the estimated CDF can also be calculated. Aside from the computer time incurred by making large numbers of runs (especially for time consuming computer codes), an examination of the code SAMPLE, developed for the Reactor Safety Study, showed that most of the effort was consumed in sorting the numerous output values to produce the CDF (Rasmussen, 1975).

CHAPTER 5

CONCLUSIONS

5.1 IMPLICATIONS OF THIS RESEARCH

The objectives of this research are: (1) To use an automatic differentiation method for evaluating the partial derivatives used in developing approximate performance functions for the Limit State, (2) To test the viability of using the improved Limit State method with a radiological and environmental assessment code, and (3) To compare the results achieved with the Limit State method against a Monte Carlo method.

The first objective is met through the use of an automatic differentiation tool, GRESS. Automatic differentiation provides a highly efficient method for obtaining partial derivatives of interest from a computer model. Through using GRESS, a minimal number of computer model runs are needed to obtain partial derivatives, independent of the number of partial derivatives desired. The partial derivatives obtained using GRESS are then used to formulate Limit State functions.

The second objective is realized through uncertainty analysis of a groundwater transport model. This computer model is implicit, meaning it is a numerical method for solving a series or set of partial differential equations. In this case, the method employed by the groundwater transport code to solve the partial differential equations is the finite difference method. Since the computer model itself is implicit, a proxy equation is used as

the Limit State function. A Limit State function is formulated with partial derivatives obtained using GRESS (for given mean random input values for a given output result). The Limit State functions are used to replace the actual computer model in the Limit State method. The results obtained are evaluated and the computational efficiency of the methods is evaluated.

The third objective is met by generating several CDFs for comparison with Limit State uncertainty analysis results. These comparison CDFs are generated using Latin Hypercube sampling and Monte Carlo. A 10,000 run Monte Carlo CDF served as the best estimate of the actual CDF and is used a s a benchmark for accuracy. Uncertainty analysis is performed using the Limit State method and the results are evaluated and compared for different approaches of the Limit State method and against results obtained using Monte Carlo and Latin Hypercube sampling.

5.2 LIMITATIONS OF THIS RESEARCH

This research examined first-order Limit State functions (an attempt at evaluating second-order Limit State functions was inconclusive and needs additional work). This is a limitation in cases of high nonlinearity in the computer model itself. In order to simplify the research and to maintain the focus on improving computational efficiencies, only log normal probability distributions are used in the research. In addition, no correlated variables are tested (since the main objective of the research is to improve and test a more computationally efficient Limit State method). Testing with other distribution functions

would be possible, but would have involved examining the efficacy of using the Rackwitz Fiessler algorithm for fast probability integration and would have complicated comparing the results against that derived using other methods. It was decided that these issues are outside the scope of this research. However, much research has been performed on this issue and many papers have been written.

Another limitation is that the groundwater transport code used is not as complex as some of the current models being used in actual performance assessments. However, even a simple computer code can demonstrate strong nonlinear behavior. For the purpose of testing the viability of using automatic differentiation with the Limit State method, it was decided that the Finite Difference Groundwater Transport model was adequate.

5.3 RECOMMENDATIONS FOR FURTHER RESEARCH

Recommendations for further research in improving and testing the Limit State method to make it a viable method for performing uncertainty analysis include development, application, and testing of automatic differentiation codes for deriving second order partial derivatives from implicit computer models. Investigating the advantages and disadvantages of using a second order Limit State function will also be useful. Documentation of the time taken to perform and obtain comparable CDFs (from an accuracy standpoint) using different methods with a computer code such as MOC (Konikow, 1994) would also be useful.

REFERENCES

REFERENCES

Berz, M., et al., "A Collection of Automatic Differentiation Tools", Presented at the 1995 International Conference in Industrial and Applied Mathematics, http://www.mcs.anl.gov/autodiff/AD_Tools, Hamburg, 1995.

Box, G. E. P. and K. B. Wilson, "On the Experimental Attainment of Optimum Conditions," *Journal of the Royal Statistical Society*, Series B 13, pp. 1-34, 1951.

Chen, X. and N. C. Lind, "Fast Probability Integration by Three Parameter Normal Tail Approximation," *Structural Safety*, Vol. 1, pp. 269-276, 1983.

Cornell, C. A., "A Probability-Based Structural Code," *Journal of the American Concrete Institute*, Vol. 66, No. 12, 1969, pp. 974-985, 1969.

Der Kiureghian, A. and P-L. Liu, "Structural Reliability Under Incomplete Probability Information," *Journal of Engineering Mechanics*, ASCE, Vol. 112, No. 1, pp. 85-104, 1986.

Der Kiureghian, A., H-Z. Lin. and S-J. Hwang, "Second Order Reliability Approximations," *Journal of Engineering Mechanics*, ASCE, Vol. 113, No.8, pp. 1208-1225, 1987.

Dettinger, M. D. and J. L. Wilson, "First-Order Analysis of Uncertainty in Numerical Models of Groundwater Flow, Part 1, Mathematical Development," *Water Resources Research*, Vol. 17, pp. 149-161 , 1981.

Ditlevsen, O., "Generalized Second Moment Reliability Index," *Journal of Structural Mechanics*, Vol. 7, No. 4, 1979, pp 435-451.

Ditlevsen, O., "Principle of Normal Tail Approximation," *Journal of the Engineering Mechanics Division*, ASCE, Vol. 107, No. 12, pp. 1191-1208, 1981.

Downing, D. J., R. H. Gardner, and F. O. Hoffman, "An Examination of Response-Surface Methodologies for Uncertainty Analysis in Assessment Models," *Technometrics*, Vol. 27, No. 2, pp. 151-163, 1985.

Draper, N. R. and H. Smith, *Applied Regression Analysis*, John Wiley & Sons, Inc., New York, 1981.

Faravelli, L., "Response-Surface Approach for Reliability Analysis," *Journal of Engineering Mechanics*, ASCE, Vol. 115, No. 12, pp. 2763-2781, 1989.

Fiessler, B., H. J. Neumann, and R. Rackwitz, "Quadratic Limit States in Structural Reliability," *Journal of the Engineering Mechanics Division*, ASCE,

Vol. 105, No. 8, pp. 661-676, 1979.

Hasofer A. M. and N. Lind, "Exact and Invariant Second-Moment Code Format," *Journal of Engineering Mechanics*, ASCE, Vol. 100, pp. 111-121, 1974.

Hohenbichler, M., and R. Rackwitz, "Improvement of Second-Order Reliability Estimates by Importance Sampling," *Journal of Engineering Mechanics*, ASCE, Vol. 114, No. 12, pp. 2195-2199, 1988.

Khuri, A. I. and J. A. Cornell, *Response Surfaces*, Marcel Dekker Inc., New York, 1987.

Iman, R. L. and W. J. Conover, "A Distribution-Free Approach to Inducing Rank Correlation Among Input Variable," *Communication in Statistics*, Vol. B11, No. 3, pp. 311-334, 1982.

Iman, R. L. and M. J. Shortencarier, *A FORTRAN 77 Program and User's Guide for the Generation of Latin Hypercube and Random Samples for Use with Computer Models*, NUREG/CR-3624, U.S. Nuclear Regulatory Commission, Washington D.C., 1984.

Iman, R. L. and J. C. Helton, *A Comparison of Uncertainty and Sensitivity*

Analysis Techniques for Computer Models, NUREG/CR-3904, Sandia National Laboratories, NM, 1985.

Kinzelbach, Wolfgang, "Groundwater Modelling: An Introduction with Sample Programs in BASIC," Elsevier Science Publishers, New York, NY, 1986.

Konikow, L. F., Granato, G. E., and Hornberger, G. Z., "User's Guide to Revised Method-of-Characteristics Solute-Transport Model," MOC Version 3.1, U.S. Geological Survey Water-Resources Investigations Report 94-4115, Reston, VA, 1994.

Lee, D. W., et. al., *Performance Assessment for Continuing and Future Operations at Solid Waste Storage Area 6*, ORNL-6783/R1, Martin Marietta Energy Systems, Inc., Oak Ridge National Laboratory, Oak Ridge, TN, 1997.

Madsen, H. O., Krenk. S., and Lind, N. C., *Methods of Structural Safety*, Prentice-Hall, Englewood Cliffs, NJ, 1986.

McKay, M. D., W. J. Conover, and R. J. Beckman, " Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output From a Computer Code," *Technometrics*, Vol. 2, pp. 239-245, 1979.

Myers, R. H., *Response Surface Methodology*, Allyn and Bacon, Boston, 1976.

Morgan, M. Granger, "Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis," Cambridge University Press, Cambridge, England, 1990.

Oblow, E. M., *GRESS-GRadient-Enhanced Software System*, ORNL/TM-9658, Oak Ridge National Laboratory, Oak Ridge, TN, 1985.

Pike, D. J. and J. R. Smith, "Response Surface Methodology in Simulation Studies of Nuclear Reactor Safety," *Proceedings of the First International Conference on Applied Modeling and Simulation*, Vol. 3, pp. 130-132, 1981.

Pike, D. J., G. Wetherill, "Comparative Experimental Design for Response Surface Analysis of reactor safety Codes," paper presented at the Annual Meeting of the American Statistical Association, 1983.

Rackwitz, R., and B. Fiessler, "Structural Reliability Under Combined Load Sequences," *Journal of Computers and Structures*, Vol. 9, pp. 489-494, 1978.

Rasmussen, Norman C., *Reactor Safety Study: An Assessment of Accident Risks in U.S. Commercial Nuclear Power Plants*, WASH-1400 (NUREG-75/-14), Nuclear

Regulatory Commission, Springfield, VA, 1975.

Rish, W. R., and R. J. Marnicio, *Review of Studies Related to Uncertainty in Risk Analysis*, ORNL/TM-10776, Oak Ridge National Laboratory, Oak Ridge, TN, 1988.

Rosenblatt, M., "Remarks on Multivariate Transformation," *Annals of Mathematical Statistics*, Institute of Mathematical Statistics, Baltimore, MD, Vol. 23, No.3, Sep., pp. 470-472, 1952.

Sitar, N., J. D. Cawlfield, and A. Der Kiureghian, "First-Order Reliability Approach to Stochastic Analysis of Subsurface Flow and Contaminant Transport," *Water Resources Research*, Vol. 23, No.5, pp. 794-804, 1987

Song, J. S., Lee, K. J., "Stochastic Analysis of Radioactive Waste Package Performance Using First-Order Reliability Method," *Waste Management*. Vol. 9, pp. 211-218, 1989.

Song, J. S. and Lee, K. J., "System Performance Assessment of Final Repository for Radioactive Wastes," *Waste Management*, Vol. 12, pp. 323-335, 1992.

Tauxe, J., Personal communication, Oak Ridge, Tennessee, August 25, 1997.

Till, J. E., and H. R. Meyer, *Radiological Assessment*, NUREG/CR-3332, U.S. Nuclear Regulatory Commission, Washington D.C., 1983.

Tvedt, L., "Distribution of Quadratic Forms in Normal Space - Application to Structural Reliability," *Journal of Engineering Mechanics*, ASCE, Vol. 116, No. 6, pp. 1183-1197, 1990.

Wong, F. S., "First-Order, Second-Moment Methods," *Computers & Structures*, Vol. 20, No. 4, pp. 779-791, 1985.

Worley, B. A., *Deterministic Uncertainty Analysis*, ORNL/TM-6428, Oak Ridge National Laboratory, Oak Ridge, TN, 1987.

Worley, B. A., et. al., *ADGEN-ADjoint GENerator for Computer Models*, ORNL/TM-11037, Oak Ridge National Laboratory, TN, 1989.

Worley, B. A., "Experience with the Forward and Reverse Mode of GRESS in Contaminant Transport Modeling and Other Applications," *Proceedings of SIAM Workshop on Automatic Differentiation of Algorithms: Theory, Implementation, and Application*, pp. 307 - 314, Breckenridge, Colorado, January 6- 8, 1991.

Wu, Y.-T., *Efficient Methods for Mechanical and Structural Reliability Analysis and Design*, Doctoral Dissertation, The University of Arizona, 1984.

Wu, Y.-T., H. R. Millwater, and T. A. Cruse, "An Advanced Probabilistic Structural Analysis Method for Implicit Performance Functions," *American Institute of Aerospace and Astronautics (AIAA) Journal*, Vol. 28, No. 9, pp. 1663-1669, 1990.

Wu, Y.-T., et. al., *Uncertainty Evaluation Methods for Waste Package Assessment*, NUREG/CR-5639, U.S. Nuclear Regulatory Commission, Washington D.C., 1991.

Wu, Y.-T., A. B. Gureghian, and B. Sagar, *Sensitivity and Uncertainty Analyses Applied to One-Dimensional Radionuclide Transport in a Layered Fractured Rock*, NUREG/CR-5917, Vol. 2, Center for Nuclear Waste Regulatory Analyses, San Antonio, TX, 1992.

Wu, Y.-T., A. B. Gureghian, B. Sagar, and R. B. Codell, "Sensitivity and Uncertainty Analyses Applied to One-Dimensional Radionuclide Transport in a Layered Fractured Rock, Part II: Probabilistic Methods Based on the Limit-State Approach," *Nuclear Technology*, Vol. 104, pp. 297-308, 1993.

APPENDICES

APPENDIX A

Rackwitz Fiessler Algorithm

```
'Rackwitz Fiessler Algorithm - 1996
'Prepared by Loong Yong
'Programmed in Quick Basic

CLS

20 NUMPAR = 5
21 itermax = 10
22 beta = 0
23 numl = 5

x(2, 1) = .2: sigma(2, 1) = .04
x(2, 2) = 100!: sigma(2, 2) = 30!
x(2, 3) = 10!: sigma(2, 3) = 3!
x(2, 4) = 3!: sigma(2, 4) = 1!
x(2, 5) = 10!: sigma(2, 5) = 3!

'x(2, 1) = 134.9: sigma(2, 1) = 13.49
'x(2, 2) = 134.9: sigma(2, 2) = 13.49
'x(2, 3) = 134.9: sigma(2, 3) = 13.49
'x(2, 4) = 134.9: sigma(2, 4) = 13.49
'x(2, 5) = 50: sigma(2, 5) = 15
'x(2, 6) = 40: sigma(2, 6) = 12

mean(2, 1) = x(2, 1)
mean(2, 2) = x(2, 2)
mean(2, 3) = x(2, 3)
mean(2, 4) = x(2, 4)
mean(2, 5) = x(2, 5)
'mean(2, 6) = x(2, 6)

'coeff(1) = 1
'coeff(2) = 2
'coeff(3) = 2
'coeff(4) = 1
'coeff(5) = -5
'coeff(6) = -5

95 FOR j = 1 TO numl
97 e(2, j) = LOG(mean(2, j)) - .5 * LOG((sigma(2, j) ^ 2 / mean(2, j) ^ 2) + 1)
98 d(2, j) = SQR(LOG((sigma(2, j) ^ 2 / mean(2, j) ^ 2 + 1)))
109 NEXT j
```

```
'DEFINE PARTIAL DERIVATIVES OF g(x)
coeff(1) = -25.86
coeff(2) = -.018425
coeff(3) = -.30632
coeff(4) = -3.359227
coeff(5) = -.5171939

'LOGNORMAL EQUIVALENT NORMAL
600 iteration = iteration + 1: PRINT "iter="; iteration
650 FOR j = 1 TO numl
655    normlogval(j) = (LOG(x(2, j)) - e(2, j)) / d(2, j)
660    sigma(2, j) = x(2, j) * d(2, j)
665    mean(2, j) = x(2, j) - normlogval(j) * sigma(2, j)
PRINT mean(2, j); sigma(2, j)
672 NEXT j

'SPECIFY g(x)
sumcs = 0
sumcm = 0

FOR i = 1 TO NUMPAR
cs(i) = (coeff(i) * sigma(2, i))
cm(i) = (coeff(i) * mean(2, i))
sumcs = sumcs + cs(i) ^ 2
sumcm = sumcm + cm(i)
NEXT i

normfactor = sumcs ^ .5

FOR i = 1 TO NUMPAR
csn(i) = cs(i) / normfactor
cmn(i) = cm(i) / normfactor
NEXT i

'a1 = cmn(1) + csn(1) * mean(2, 1)
'a2 = cmn(2) + csn(2) * mean(2, 2)
'a3 = cmn(3) + csn(3) * mean(2, 3)
'a4 = cmn(4) + csn(4) * mean(2, 4)
'a5 = cmn(5) + csn(5) * mean(2, 5)
'a6 = cmn(6) + csn(6) * mean(2, 6)

a1 = cm(1) + cs(1) * mean(2, 1)
a2 = cm(2) + cs(2) * mean(2, 2)
```

```
a3 = cm(3) + cs(3) * mean(2, 3)
a4 = cm(4) + cs(4) * mean(2, 4)
a5 = cm(5) + cs(5) * mean(2, 5)
'a6 = cm(6) + cs(6) * mean(2, 6)


'gx = (sumcm + 29.99) / normfactor
'gx = 6 - (36.79 + (a1 + a2 + a3 + a4 + a5))
gx = (a1 + a2 + a3 + a4 + a5 + a6 + 30.49924-8)

PRINT "normfactor"; normfactor
PRINT "gx="; gx

FOR i = 1 TO NUMPAR
'gp(i) = csn(i)
gp(i) = cs(i)
NEXT i

gpsqrsum = 0
FOR j = 1 TO NUMPAR
gpsqrsum = gp(j) ^ 2 + gpsqrsum
NEXT j

norm = SQR(gpsqrsum)

'Calculate alpha(i)
FOR i = 1 TO NUMPAR
alpha(i) = -gp(i) / norm
NEXT i

'Calculate next value of x
xalphasum = 0
FOR k = 1 TO NUMPAR
xalphasum = mean(2, k) * alpha(k) + xalphasum
NEXT k

FOR i = 1 TO NUMPAR
nextx(i) = xalphasum * alpha(i) + (gx / norm) * alpha(i)
PRINT "nextx="; nextx(i)
NEXT i

'3410 betanew = SQR(nextx(1) ^ 2 + nextx(2) ^ 2 + nextx(3) ^ 2 + nextx(4) ^ 2 +
nextx(5) ^ 2)
```

```
betanew = nextx(1) / alpha(1)
3412 betadif = betanew - beta
PRINT "beta="; betanew
PRINT "abs(betadif)", ABS(betadif)

3420 x(2, 1) = nextx(1) * sigma(2, 1) + mean(2, 1)
3425 x(2, 2) = nextx(2) * sigma(2, 2) + mean(2, 2)
3430 x(2, 3) = nextx(3) * sigma(2, 3) + mean(2, 3)
3435 x(2, 4) = nextx(4) * sigma(2, 4) + mean(2, 4)
3440 x(2, 5) = nextx(5) * sigma(2, 5) + mean(2, 5)
'3445 x(2, 6) = nextx(6) * sigma(2, 6) + mean(2, 6)

'FOR i = 1 TO numpar
'IF x(2, i) <= 0 THEN x(2, i) = .001
'NEXT i

PRINT "x=", x(2, 1)
PRINT "x=", x(2, 2)
PRINT "x=", x(2, 3)
PRINT "x=", x(2, 4)
PRINT "x=", x(2, 5)
'PRINT "x=", x(2, 6)

3914 IF ABS(betadif) < .0001 THEN GOTO 4600
IF iteration = itermax THEN GOTO 4600

4190 beta = betanew
4200 GOTO 600

4510 PRINT "beta=", betanew

4600 a = -5
4700 b = betanew
4800 n = 100
4900 h = (b - a) / n
5000 xi0 = EXP(-a ^ 2 / 2) + EXP(-b ^ 2 / 2)
5010 xi1 = 0
5020 xi2 = 0
5030 j = n - 1
5040 FOR i = 1 TO j
5050 x = a + i * h
5060 k = i MOD 2
5070 IF k = 0 THEN xi2 = xi2 + EXP(-x ^ 2 / 2)
```

```
5080 IF k > 0 THEN xi1 = xi1 + EXP(-x ^ 2 / 2)
5100 NEXT i
5200 xi = h * (xi0 + 2 * xi2 + 4 * xi1) / 3
5250 phi = .39894 * xi
5255 pf = 1 - phi
5300 PRINT "Pf="; pf

6100 END
```

APPENDIX B

Finite Difference Groundwater Transport Program

```
10   'Transport Model By Explicit Difference Method
11   'Input Parameters
12   CLS
15   max = 50
17   DIM IR(15, 15), CO(16, 16), c(15, 15), MP(15, 15), FL(15), FLA(15), al(max),
aq(max), ux(max), m(max), ne(max)

18   'OPEN "m9750out.50" FOR INPUT AS #1
19   'OPEN "m9750.run" FOR OUTPUT AS #2'

21   'PRINT #2, TIME$
22   'FOR i = 1 TO max: PRINT i
23   al(1) = 94.12358: aq(1) = 8.856795: ux(1) = 2.64072: m(1) = 8.380379
     ne(1) = .1839341

24   'FOR k = 1 TO max
25   'INPUT #1, AL(k), AQ(k), UX(k), M(k), NE(k)
26   'PRINT k; AL(k); AQ(k); UX(k); M(k); NE(k)
28   'NEXT k
29   'CLOSE #1

31   'PRINT #2, TIME$
32   FOR i = 1 TO max: PRINT i; TIME$

33   'AL(i) = ALIN(i)
34   'AQ(i) = AQIN(i)
35   'UX(i) = UXIN(i)
36   'M(i) = MIN(i)
37   'NE(i) = NEIN(i)

40   'Grid Parameters
50   NX = 15
60   NY = 15
70   DX = 100
80   DY = 100
90   '
100  'Number of Pollution Sources
110  NS = 1
120  '
150  'Dispersivities
160  'AL = 100
170  'AQ = 10
180  '
```

```
190  'Velocity
200  'UX = 1
210  UY = 0!
220  '
230  'USE ZERO FOR INITIAL CO
240  FOR m = 0 TO 15
250  FOR N = 0 TO 15
260  CO(m, N) = 0
270  NEXT N
275  NEXT m
280  '
290  'USE ZERO FOR INITIAL MP
300  FOR m = 1 TO 15
310  FOR N = 1 TO 15
320  MP(m, N) = 0
330  NEXT N
340  NEXT m
350  '
360  'USE ZERO FOR INITIAL IR
370  FOR m = 1 TO 15
380  FOR N = 1 TO 15
390  IR(m, N) = 0
400  NEXT N
410  NEXT m
420  '
430  'USE ZERO FOR INITIAL C
440  FOR m = 1 TO 15
450  FOR N = 1 TO 15

460  c(m, N) = 0
470  NEXT N
480  NEXT m
490  '
500  'Thickness of Flow, Effective Porosity, Decay Constant
510  'M = 10
520  'NE = .15
530  LA = 0
540  '
550  U = SQR(ux(i) * ux(i) + UY * UY)
560  DM = .000001
570  XX = al(i) * ux(i) * ux(i) / U + aq(i) * UY * UY / U + DM
580  XY = (al(i) - aq(i)) * ux(i) * UY / U
590  YY = al(i) * UY * UY / U + aq(i) * ux(i) * ux(i) / U + DM
```

```
600  '
620  'Time Parameters
630  T = 0
640  DT = .5 / (XX / DX / DX + YY / DY / DY)
650  TM = 600
660  '
670  IF U = 0 THEN GOTO 750
700  TD = DX * DY / (ux(i) * DY + UY * DX)
710  IF TD < DT THEN DT = TD
730  DT = DT / 2
750  NT = INT(TM / DT + .5)
760  '
780  'Difference Scheme
790  DS = 1
800  '
810  'Labelling of Constant Concentration Nodes
820  FOR j = 1 TO NY
830  IF ux(i) > 0 THEN IR(1, j) = 1
840  NEXT j
850  FOR j = 1 TO NX
860  IF UY > 0 THEN IR(j, 1) = 1
870  NEXT j
880  '
890  'Sources
900  'IS(1) = 7
910  'JS(1) = 7
920  MP(7, 7) = 10000
930  '
1000 'Explicit Solution Procedure
1010 'Weighting of Difference Scheme
1020 'SGNUX = 1
1030 'SGNUY = 0
1040 '
1050 A = (1 + SGN(ux(i))) / 2
1060 B = A
1070 G = (1 + SGN(UY)) / 2
1080 D = G
1090 'GOSUB 1752
1110 '
1120 'Time Loop
1130 FOR IT = 1 TO NT
1140 FOR l = 1 TO NX
1150 FOR j = 1 TO NY
```

```
1160  IF IR(l, j) = 1 THEN GOTO 1640
1170  'END IF
1180  C1 = ux(i) * (A * CO(l - 1, j) + (1 - A) * CO(l, j)) / DX
1190  C2 = -ux(i) * (B * CO(l, j) + (1 - B) * CO(l + 1, j)) / DX
1200  C3 = UY * (G * CO(l, j - 1) + (1 - G) * CO(l, j)) / DY
1210  C4 = -UY * (D * CO(l, j) + (1 - D) * CO(l, j + 1)) / DY
1220  D1 = XX * (CO(l - 1, j) - CO(l, j)) / DX / DX
1230  D2 = -XX * (CO(l, j) - CO(l + 1, j)) / DX / DX
1240  D3 = YY * (CO(l, j - 1) - CO(l, j)) / DY / DY
1250  D4 = -YY * (CO(l, j) - CO(l, j + 1)) / DY / DY
1260  D5 = XY * (CO(l, j + 1) - CO(l, j - 1) + CO(l + 1, j + 1) - CO(l + 1, j - 1)) / 4 / DX
/ DY
1270  D6 = -XY * (CO(l - 1, j + 1) - CO(l - 1, j - 1) + CO(l, j + 1) - CO(l, j - 1)) / 4 / DX
/ DY
1280  D7 = XY * (CO(l + 1, j) - CO(l - 1, j) + CO(l + 1, j + 1) - CO(l - 1, j + 1)) / 4 / DX
/ DY
1290  D8 = -XY * (CO(l + 1, j - 1) - CO(l - 1, j - 1) + CO(l + 1, j) - CO(l - 1, j)) / 4 / DX
/ DY
1310  'Modifications at Transmission Boundaries
1320  IF l = NX AND ux(i) > 0 THEN D1 = 0
      IF l = NX AND ux(i) > 0 THEN D2 = 0
      IF l = NX AND ux(i) > 0 THEN D5 = 0
      IF l = NX AND ux(i) > 0 THEN D6 = 0
      IF l = NX AND ux(i) > 0 THEN D7 = XY * (CO(l, j) - CO(l - 1, j) + CO(l, j + 1) -
CO(l - 1, j + 1)) / 2 / DX / DY
      IF l = NX AND ux(i) > 0 THEN D8 = -XY * (CO(l, j - 1) - CO(l - 1, j - 1) + CO(l, j)
- CO(l - 1, j)) / 2 / DX / DY
1390  IF j = NY AND UY > 0 THEN D3 = 0
      IF j = NY AND UY > 0 THEN D4 = 0
      IF j = NY AND UY > 0 THEN D5 = XY * (CO(l, j) - CO(l, j - 1) + CO(l + 1, j) -
CO(l + 1, j - 1)) / 2 / DX / DY
      IF j = NY AND UY > 0 THEN D6 = -XY * (CO(l - 1, j) - CO(l - 1, j - 1) + CO(l, j) -
CO(l, j - 1)) / 2 / DX / DY
      IF j = NY AND UY > 0 THEN D7 = 0
      IF j = NY AND UY > 0 THEN D8 = 0
1460  '
1470  'Modifications at Impervious Boundaries
1480  IF ux(i) = 0 AND l = 1 THEN D1 = 0
1490  IF ux(i) = 0 AND l = 1 THEN D6 = 0
1510  IF ux(i) = 0 AND l = NX THEN D2 = 0
1520  IF ux(i) = 0 AND l = NX THEN D5 = 0
1540  IF UY = 0 AND j = 1 THEN D3 = 0
1550  IF UY = 0 AND j = 1 THEN D8 = 0
```

```
1570  IF UY = 0 AND j = NY THEN D4 = 0
1580  IF UY = 0 AND j = NY THEN D7 = 0
1600  BS1 = C1 + C2 + C3 + C4
1610  BS2 = D1 + D2 + D3 + D4 + D5 + D6 + D7 + D8 - LA * CO(l, j) + MP(l, j) / DX
/ DY / m(i) / ne(i)
1620  BS3 = BS1 + BS2
1630  c(l, j) = CO(l, j) + DT * BS3
1640  NEXT j, l
1660  T = T + DT
1670  '
1680  'Updating of Concentrations
1690  FOR j = 1 TO NY
1700  FOR l = 1 TO NX
1710  CO(l, j) = c(l, j)
1720  NEXT l
1730  NEXT j
1740  '
1741  'GOSUB 1754
1742  NEXT IT
1743  PRINT  C(15, 7)
1744  PRINT c(15, 7)
1745  NEXT i
1747  'PRINT #2, TIME$
1748  'CLOSE #1
1752  'CLOSE #2

1753  END
1754  'Subroutine for Output of Results
1755  HOME: PRINT "CONCENTRATION (MG/L)"
1756  PRINT "TIME (D) : "; INT(T + .5)
1760  FOR j = 1 TO 15
1770  FOR l = 1 TO 15
1779  FLA(l) = c(l, j): FL(l) = INT(FLA(l) + .5)
1790  NEXT l
1795  PRINT USING " ## "; FL(1); FL(2); FL(3); FL(4); FL(5); FL(6); FL(7); FL(8);
FL(9); FL(10); FL(11); FL(12); FL(13); FL(14); FL(15)
1800  NEXT j
1805  RETURN
```

APPENDIX C

Second Order Limit State Test Algorithm

```
Second Order Limit State Function Algorithm - 1997
'This prog tests the FDGW runs using a 2nd order Limit State Function
'Loong Yong

'INPUT DATA
10 n=5
20 itermax=100
betaold=0

'INITIAL APPROXIMATION POINT
'1-ne, 2-al, 3-aq, 4-ux, 5-m
30 coeff(1)=-25.86
35 coeff(2)=-.018425
40 coeff(3)=-.30632
45 coeff(4)=-3.359227
50 coeff(5)=-.517194
coeff2(1) =-1.2479e2
coeff2(2) =2.7843e-4
coeff2(3) =1.7804e-1
coeff2(4) =-2.0019
coeff2(5) =-9.0939e-2

'coeff2(1) =0
'coeff2(2) =0
'coeff2(3) =0
'coeff2(4) =0
'coeff2(5) =0


70 z(1)=.2: dx(1)=.04
90 z(2)=100: dx(2)=30
100 z(3)=10: dx(3)=3
110 z(4)=3: dx(4)=1
112 z(5)=10: dx(5)=3

120 for i=1 to n
122 mx(i)=log(z(i))-0.5*log(dx(i)^2/z(i)^2+1)
124 sx(i)=(log(dx(i)^2/z(i)^2+1))^0.5
126 next i

u(1)=(log(z(1))-mx(1))/sx(1)
u(2)=(log(z(2))-mx(2))/sx(2)
u(3)=(log(z(3))-mx(3))/sx(3)
```

```
u(4)=(log(z(4))-mx(4))/sx(4)
u(5)=(log(z(5))-mx(5))/sx(5)

'START OF ITERATION
140 iter=0
145 iter=iter+1
155 if iter=itermax then end

160 w(1)=exp(sx(1)*u(1)+mx(1))
161 w(2)=exp(sx(2)*u(2)+mx(2))
162 w(3)=exp(sx(3)*u(3)+mx(3))
163 w(4)=exp(sx(4)*u(4)+mx(4))
164 w(5)=exp(sx(5)*u(5)+mx(5))
165 ww(1)=(exp(sx(1)*u(1)+mx(1)))^2
166 ww(2)=(exp(sx(2)*u(2)+mx(2)))^2
167 ww(3)=(exp(sx(3)*u(3)+mx(3)))^2
168 ww(4)=(exp(sx(4)*u(4)+mx(4)))^2
169 ww(5)=(exp(sx(5)*u(5)+mx(5)))^2

'DEFINE PARTIAL DERIVATIVES OF g(x)
170 gp(1)=coeff(1)*sx(1)*w(1)+coeff2(1)*2*sx(1)*ww(1)
180 gp(2)=coeff(2)*sx(2)*w(2)+coeff2(2)*2*sx(2)*ww(2)
190 gp(3)=coeff(3)*sx(3)*w(3)+coeff2(3)*2*sx(3)*ww(3)
200 gp(4)=coeff(4)*sx(4)*w(4)+coeff2(4)*2*sx(4)*ww(4)
210 gp(5)=coeff(5)*sx(5)*w(5)+coeff2(5)*2*sx(5)*ww(5)


'CALCULATE g(x)
stuff=coeff2(1)*ww(1)+coeff2(2)*ww(2)+coeff2(3)*ww(3)+coeff2(4)*ww(4)+coeff2(5)*
ww(5)
240
gu=coeff(1)*w(1)+coeff(2)*w(2)+coeff(3)*w(3)+coeff(4)*w(4)+coeff(5)*w(5)+0.5*stuff
+30.5-16


'CALCULATE NORM grad g(x)
250 norm=sqr(gp(1)^2+gp(2)^2+gp(3)^2+gp(4)^2+gp(5)^2)

'CALCULATE ALPHA(i)
260 for i=1 to n
270 alpha(i)=-gp(i)/norm
280 next i
```

```
'CALCULATE NEXT VALUE OF u
290 for j=1 to n
300
nextu(j)=(u(1)*alpha(1)+u(2)*alpha(2)+u(3)*alpha(3)+u(4)*alpha(4)+u(5)*alpha(5))*alp
ha(j)+(gu/norm)*alpha(j)
305 print "nextu=";nextu(j)
310 next j

325 for m=1 to n
326 u(m)=nextu(m)
327 print "u(m)=";u(m)
328 next m
'329 goto 145

'CALCULATE BETA
420 beta=(nextu(1)^2+nextu(2)^2+nextu(3)^2+nextu(4)^2+nextu(5)^2)^0.5
dif=beta-betaold
print "dif=";dif
print "beta=";beta
betaold=beta
if dif>0.001 goto 145

'PRINT RESULTS
430 print chr$(14);"  <<<<<<  Iteration No.  >>>>>>",iter
440 print chr$(14);" "
450 for l = 1 to n
460 print "       u";l
470 print using "    +#.####^^^^";nextu(l)
480 next l
490 print " "
500 print "    Beta ="; beta
510 print " "

600 a=-5
700 b=beta
800 n=100
900 h=(b-a)/n
1000 xi0=exp(-a^2/2)+exp(-b^2/2)
1010 xi1=0
1020 xi2=0
1030 j=n-1
1040 for i=1 to j
1050 x=a+i*h
```

```
1060 k=i mod 2
1070 if k=0 then xi2=xi2+exp(-x^2/2)
1080 if k>0 then xi1=xi1+exp(-x^2/2)
1100 next i
1200 xi=h*(xi0+2*xi2+4*xi1)/3
1250 phi=0.39894*xi
'note:  if>50%, use phi, <50%, use 1-phi
1300 print "PHI="; phi

1325 for m=1 to 5
1326 q(m)=nextu(m)*sx(m)+mx(m)
1327 print "z(m)=";exp(q(m))
1328 next m
```

APPENDIX D

Enhanced FORTRAN
Finite Difference Groundwater Transport Code

```
*
*  Code enhanced for sensitivity calculations
*        by SymG  Version 1.0
*
      DOUBLE PRECISION DXQQ01(5)
      PARAMETER (MAXROW=50)
      PARAMETER (IWSP = 20000)
      COMMON/ZZZZZZ/LIMIT(IWSP+3)
      REAL SQRTXX,ASINXX,ACOSXX,ALOGXX,EXPXX,RDXPXX,ATANXX
      DOUBLE PRECISION DSQRTXX,DASINXX,DACOSXX,DLOGXX
      DOUBLE PRECISION DEXPXX,DRXPXX,DATANXX
      REAL DX(MAXROW)
      COMMON /ZZZZQ1/ DX
      REAL RXQQ01(5)
      INTEGER IXQQ01(5)
       DIMENSION LF(15),IR(15,15), CO(16,16), C(15,15), XMP(15,15)
       REAL M,NE,LA
      CALL PREPXX(IWSP)
       CALL AUTOXX(-1,3,0,0)
       NX = 15
       NY = 15
       gdx = 100
      CALL ZERRXX(1,gdx)
       gdy = 100
      CALL ZERRXX(2,gdy)
       NS = 1
       AL = 100
      CALL ZERRXX(3,AL)
       AQ = 10
      CALL ZERRXX(4,AQ)
       UX = 1
      CALL ZERRXX(5,UX)
       UY = 0
      CALL ZERRXX(6,UY)
       M = 10
      CALL ZERRXX(7,M)
       NE = 0.15
      CALL ZERRXX(8,NE)
       LA = 0.0
      CALL ZERRXX(9,LA)
      DX(1)=SQRTXX(UX*UX+UY*UY)*UX+SQRTXX(UX*UX+UY*UY)*UX
      DX(2)=SQRTXX(UX*UX+UY*UY)*UY+SQRTXX(UX*UX+UY*UY)*UY
       U = SQRT(UX*UX+UY*UY)
```

```
CALL LOCCXX(10,3,U,UX,UY)
 DM = 1.0E-06
CALL ZERRXX(11,DM)
 CALL DEFIXX(AL)
 CALL DEFIXX(AQ)
 CALL DEFIXX(UX)
DX(1)=1./U*UX*UX
DX(2)=1./U*UX*AL+1./U*(AL*UX)
DX(3)=(-((AL*UX)*UX)/U/U)+(-((AQ*UY)*UY)/U/U)
DX(4)=1./U*UY*UY
DX(5)=1./U*UY*AQ+1./U*(AQ*UY)
DX(6)=1.0D00
 XX = AL*UX*UX/U+AQ*UY*UY/U+DM
CALL LOCCXX(12,7,XX,AL,UX,U,AQ,UY,DM)
DX(1)=1./U*UY*UX
DX(2)=1./U*UY*UX*(-1.0D00)
DX(3)=1./U*UY*(AL-AQ)
DX(4)=1./U*((AL-AQ)*UX)
DX(5)=-(((AL-AQ)*UX)*UY)/U/U
 XY = (AL-AQ)*UX*UY/U
CALL LOCCXX(13,6,XY,AL,AQ,UX,UY,U)
DX(1)=1./U*UY*UY
DX(2)=1./U*UY*AL+1./U*(AL*UY)
DX(3)=(-((AL*UY)*UY)/U/U)+(-((AQ*UX)*UX)/U/U)
DX(4)=1./U*UX*UX
DX(5)=1./U*UX*AQ+1./U*(AQ*UX)
DX(6)=1.0D00
 YY = AL*UY*UY/U+AQ*UX*UX/U+DM
CALL LOCCXX(14,7,YY,AL,UY,U,AQ,UX,DM)
 T = 0
CALL ZERRXX(15,T)
DX(1)=(-0.5/((XX/gdx)/gdx+(YY/gdy)/gdy)/((XX/gdx)/gdx+(YY/gdy)/gdy
*))*1./gdx*1./gdx
DX(2)=(-0.5/((XX/gdx)/gdx+(YY/gdy)/gdy)/((XX/gdx)/gdx+(YY/gdy)/gdy
*))*1./gdx*(-XX/gdx/gdx)+(-0.5/((XX/gdx)/gdx+(YY/gdy)/gdy)/((XX/gdx
*)/gdx+(YY/gdy)/gdy))*(-(XX/gdx)/gdx/gdx)
DX(3)=(-0.5/((XX/gdx)/gdx+(YY/gdy)/gdy)/((XX/gdx)/gdx+(YY/gdy)/gdy
*))*1./gdy*1./gdy
DX(4)=(-0.5/((XX/gdx)/gdx+(YY/gdy)/gdy)/((XX/gdx)/gdx+(YY/gdy)/gdy
*))*1./gdy*(-YY/gdy/gdy)+(-0.5/((XX/gdx)/gdx+(YY/gdy)/gdy)/((XX/gdx
*)/gdx+(YY/gdy)/gdy))*(-(YY/gdy)/gdy/gdy)
 DT = 0.5/(XX/gdx/gdx+YY/gdy/gdy)
CALL LOCCXX(16,5,DT,XX,gdx,YY,gdy)
```

```
      TM = 600
      CALL ZERRXX(17,TM)
      DX(1)=1./(UX*gdy+UY*gdx)*gdy+(-(gdx*gdy)/(UX*gdy+UY*gdx)/(UX*gdy+U
     *Y*gdx))*UY
      DX(2)=1./(UX*gdy+UY*gdx)*gdx+(-(gdx*gdy)/(UX*gdy+UY*gdx)/(UX*gdy+U
     *Y*gdx))*UX
      DX(3)=(-(gdx*gdy)/(UX*gdy+UY*gdx)/(UX*gdy+UY*gdx))*gdy
      DX(4)=(-(gdx*gdy)/(UX*gdy+UY*gdx)/(UX*gdy+UY*gdx))*gdx
      TD = gdx*gdy/(UX*gdy+UY*gdx)
      CALL LOCCXX(18,5,TD,gdx,gdy,UX,UY)
      IF (TD.LT.DT) THEN
      DX(1)=1.
      DT = TD
      CALL LOCCXX(19,2,DT,TD)
      END IF
      DX(1)=1./2
      DT = DT/2
      CALL LOCCXX(20,2,DT,DT)
      NT = INT(TM/DT+0.5)
      DS = 1
      CALL ZERRXX(21,DS)
      DO 90001 I = 1, NX
      IR(I,1) = 1
    1 CONTINUE
90001 CONTINUE
      DO 90002 J = 1, NY
      IR(1,J) = 1
    2 CONTINUE
90002 CONTINUE
      DO 90003 I=1,16
      DO 90004 J=1,16
      CO(I,J)=0
      CALL ZERRXX(22,CO(I,J))
   16 CONTINUE
90004 CONTINUE
   15 CONTINUE
90003 CONTINUE
      XMP(7,7) = 10000
      CALL ZERRXX(23,XMP(7,7))
      A = 1
      CALL ZERRXX(24,A)
      B = 1
      CALL ZERRXX(25,B)
```

```
 G = 0.5
CALL ZERRXX(26,G)
 D = 0.5
CALL ZERRXX(27,D)
 DO 3, IT = 1, NT
 DO 4, I = 1, NX
 DO 5, J = 1, NY
 IF (IR(I,J).EQ.1) THEN
 GOTO 5
 END IF
DX(1)=1./gdx*(A*CO(i-1,j)+(1-A)*CO(i,j))
DX(2)=1./gdx*UX*CO(i-1,j)+1./gdx*UX*CO(i,j)*(-1.0D00)
DX(3)=1./gdx*UX*A
DX(4)=1./gdx*UX*(1-A)
DX(5)=-(UX*(A*CO(i-1,j)+(1-A)*CO(i,j)))/gdx/gdx
 C1=UX*(A*CO(i-1,j)+(1-A)*CO(i,j))/gdx
CALL LOCCXX(28,6,C1,UX,A,CO(i-1,j),CO(i,j),gdx)
DX(1)=(-1.)*1./gdx*(B*CO(i,j)+(1-B)*CO(i+1,j))
DX(2)=(-1.)*1./gdx*UX*CO(i,j)+(-1.)*1./gdx*UX*CO(i+1,j)*(-1.0D00)
DX(3)=(-1.)*1./gdx*UX*B
DX(4)=(-1.)*1./gdx*UX*(1-B)
DX(5)=(-1.)*(-(UX*(B*CO(i,j)+(1-B)*CO(i+1,j)))/gdx/gdx)
 C2=-UX*(B*CO(i,j)+(1-B)*CO(i+1,j))/gdx
CALL LOCCXX(29,6,C2,UX,B,CO(i,j),CO(i+1,j),gdx)
DX(1)=1./gdy*(G*CO(i,j-1)+(1-G)*CO(i,j))
DX(2)=1./gdy*UY*CO(i,j-1)+1./gdy*UY*CO(i,j)*(-1.0D00)
DX(3)=1./gdy*UY*G
DX(4)=1./gdy*UY*(1-G)
DX(5)=-(UY*(G*CO(i,j-1)+(1-G)*CO(i,j)))/gdy/gdy
 C3=UY*(G*CO(i,j-1)+(1-G)*CO(i,j))/gdy
CALL LOCCXX(30,6,C3,UY,G,CO(i,j-1),CO(i,j),gdy)
DX(1)=(-1.)*1./gdy*(D*CO(i,j)+(1-D)*CO(i,j+1))
DX(2)=(-1.)*1./gdy*UY*CO(i,j)+(-1.)*1./gdy*UY*CO(i,j+1)*(-1.0D00)
DX(3)=(-1.)*1./gdy*UY*D
DX(4)=(-1.)*1./gdy*UY*(1-D)
DX(5)=(-1.)*(-(UY*(D*CO(i,j)+(1-D)*CO(i,j+1)))/gdy/gdy)
 C4=-UY*(D*CO(i,j)+(1-D)*CO(i,j+1))/gdy
CALL LOCCXX(31,6,C4,UY,D,CO(i,j),CO(i,j+1),gdy)
DX(1)=1./gdx*1./gdx*(CO(i-1,j)-CO(i,j))
DX(2)=1./gdx*1./gdx*XX
DX(3)=1./gdx*1./gdx*XX*(-1.0D00)
DX(4)=1./gdx*(-(XX*(CO(i-1,j)-CO(i,j)))/gdx/gdx)-((XX*(CO(i-1,j)-C
*O(i,j)))/gdx)/gdx/gdx
```

```
 D1=XX*(CO(i-1,j)-CO(i,j))/gdx/gdx
 CALL LOCCXX(32,5,D1,XX,CO(i-1,j),CO(i,j),gdx)
 DX(1)=(-1.)*1./gdx*1./gdx*(CO(i,j)-CO(i+1,j))
 DX(2)=(-1.)*1./gdx*1./gdx*XX
 DX(3)=(-1.)*1./gdx*1./gdx*XX*(-1.0D00)
 DX(4)=(-1.)*1./gdx*(-(XX*(CO(i,j)-CO(i+1,j)))/gdx/gdx)+(-1.)*(-((X
*X*(CO(i,j)-CO(i+1,j)))/gdx)/gdx/gdx)
 D2=-XX*(CO(i,j)-CO(i+1,j))/gdx/gdx
 CALL LOCCXX(33,5,D2,XX,CO(i,j),CO(i+1,j),gdx)
 DX(1)=1./gdy*1./gdy*(CO(i,j-1)-CO(i,j))
 DX(2)=1./gdy*1./gdy*YY
 DX(3)=1./gdy*1./gdy*YY*(-1.0D00)
 DX(4)=1./gdy*(-(YY*(CO(i,j-1)-CO(i,j)))/gdy/gdy)-((YY*(CO(i,j-1)-C
*O(i,j)))/gdy)/gdy/gdy
 D3=YY*(CO(i,j-1)-CO(i,j))/gdy/gdy
 CALL LOCCXX(34,5,D3,YY,CO(i,j-1),CO(i,j),gdy)
 DX(1)=(-1.)*1./gdy*1./gdy*(CO(i,j)-CO(i,j+1))
 DX(2)=(-1.)*1./gdy*1./gdy*YY
 DX(3)=(-1.)*1./gdy*1./gdy*YY*(-1.0D00)
 DX(4)=(-1.)*1./gdy*(-(YY*(CO(i,j)-CO(i,j+1)))/gdy/gdy)+(-1.)*(-((Y
*Y*(CO(i,j)-CO(i,j+1)))/gdy)/gdy/gdy)
 D4=-YY*(CO(i,j)-CO(i,j+1))/gdy/gdy
 CALL LOCCXX(35,5,D4,YY,CO(i,j),CO(i,j+1),gdy)
 DX(1)=1./gdy*1./gdx*1./4*(CO(i,j+1)-CO(i,j-1)+CO(i+1,j+1)-CO(i+1,j
*-1))
 DX(2)=1./gdy*1./gdx*1./4*XY
 DX(3)=1./gdy*1./gdx*1./4*XY*(-1.0D00)
 DX(4)=1./gdy*1./gdx*1./4*XY
 DX(5)=1./gdy*1./gdx*1./4*XY*(-1.0D00)
 DX(6)=1./gdy*(-((XY*(CO(i,j+1)-CO(i,j-1)+CO(i+1,j+1)-CO(i+1,j-1)))
*/4)/gdx/gdx)
 DX(7)=-(((XY*(CO(i,j+1)-CO(i,j-1)+CO(i+1,j+1)-CO(i+1,j-1)))/4)/gdx
*)/gdy/gdy
 D5=XY*(CO(i,j+1)-CO(i,j-1)+CO(i+1,j+1)-CO(i+1,j-1))/4/gdx/gdy
 CALL LOCCXX(36,8,D5,XY,CO(i,j+1),CO(i,j-1),CO(i+1,j+1),CO(i+1,j-1)
*,gdx,gdy)
 DX(1)=(-1.)*1./gdy*1./gdx*1./4*(CO(i-1,j+1)-CO(i-1,j-1)+CO(i,j+1)-
*CO(i,j-1))
 DX(2)=(-1.)*1./gdy*1./gdx*1./4*XY
 DX(3)=(-1.)*1./gdy*1./gdx*1./4*XY*(-1.0D00)
 DX(4)=(-1.)*1./gdy*1./gdx*1./4*XY
 DX(5)=(-1.)*1./gdy*1./gdx*1./4*XY*(-1.0D00)
 DX(6)=(-1.)*1./gdy*(-((XY*(CO(i-1,j+1)-CO(i-1,j-1)+CO(i,j+1)-CO(i,
```

```
*j-1)))/4)/gdx/gdx)
 DX(7)=(-1.)*(-(((XY*(CO(i-1,j+1)-CO(i-1,j-1)+CO(i,j+1)-CO(i,j-1)))
*/4)/gdx)/gdy/gdy)
  D6=-XY*(CO(i-1,j+1)-CO(i-1,j-1)+CO(i,j+1)-CO(i,j-1))/4/gdx/gdy
 CALL LOCCXX(37,8,D6,XY,CO(i-1,j+1),CO(i-1,j-1),CO(i,j+1),CO(i,j-1)
*,gdx,gdy)
 DX(1)=1./gdy*1./gdx*1./4*(CO(i+1,j)-CO(i-1,j)+CO(i+1,j+1)-CO(i-1,j
*+1))
 DX(2)=1./gdy*1./gdx*1./4*XY
 DX(3)=1./gdy*1./gdx*1./4*XY*(-1.0D00)
 DX(4)=1./gdy*1./gdx*1./4*XY
 DX(5)=1./gdy*1./gdx*1./4*XY*(-1.0D00)
 DX(6)=1./gdy*(-((XY*(CO(i+1,j)-CO(i-1,j)+CO(i+1,j+1)-CO(i-1,j+1)))
*/4)/gdx/gdx)
 DX(7)=-(((XY*(CO(i+1,j)-CO(i-1,j)+CO(i+1,j+1)-CO(i-1,j+1)))/4)/gdx
*)/gdy/gdy
  D7=XY*(CO(i+1,j)-CO(i-1,j)+CO(i+1,j+1)-CO(i-1,j+1))/4/gdx/gdy
 CALL LOCCXX(38,8,D7,XY,CO(i+1,j),CO(i-1,j),CO(i+1,j+1),CO(i-1,j+1)
*,gdx,gdy)
 DX(1)=(-1.)*1./gdy*1./gdx*1./4*(CO(i+1,j-1)-CO(i-1,j-1)+CO(i+1,j)-
*CO(i-1,j))
 DX(2)=(-1.)*1./gdy*1./gdx*1./4*XY
 DX(3)=(-1.)*1./gdy*1./gdx*1./4*XY*(-1.0D00)
 DX(4)=(-1.)*1./gdy*1./gdx*1./4*XY
 DX(5)=(-1.)*1./gdy*1./gdx*1./4*XY*(-1.0D00)
 DX(6)=(-1.)*1./gdy*(-((XY*(CO(i+1,j-1)-CO(i-1,j-1)+CO(i+1,j)-CO(i-
*1,j)))/4)/gdx/gdx)
 DX(7)=(-1.)*(-(((XY*(CO(i+1,j-1)-CO(i-1,j-1)+CO(i+1,j)-CO(i-1,j)))
*/4)/gdx)/gdy/gdy)
  D8=-XY*(CO(i+1,j-1)-CO(i-1,j-1)+CO(i+1,j)-CO(i-1,j))/4/gdx/gdy
 CALL LOCCXX(39,8,D8,XY,CO(i+1,j-1),CO(i-1,j-1),CO(i+1,j),CO(i-1,j)
*,gdx,gdy)
  IF (I.EQ.NX.AND.UX.GT.0) THEN
  D1=0
 CALL ZERRXX(40,D1)
  D2=0
 CALL ZERRXX(41,D2)
  D5=0
 CALL ZERRXX(42,D5)
  D6=0
 CALL ZERRXX(43,D6)
 DX(1)=1./gdy*1./gdx*1./2*(CO(i,j)-CO(i-1,j)+CO(i,j+1)-CO(i-1,j+1))
 DX(2)=1./gdy*1./gdx*1./2*XY
```

```
 DX(3)=1./gdy*1./gdx*1./2*XY*(-1.0D00)
 DX(4)=1./gdy*1./gdx*1./2*XY
 DX(5)=1./gdy*1./gdx*1./2*XY*(-1.0D00)
 DX(6)=1./gdy*(-((XY*(CO(i,j)-CO(i-1,j)+CO(i,j+1)-CO(i-1,j+1)))/2)/
*gdx/gdx)
 DX(7)=-(((XY*(CO(i,j)-CO(i-1,j)+CO(i,j+1)-CO(i-1,j+1)))/2)/gdx)/gd
*y/gdy
  D7=XY*(CO(i,j)-CO(i-1,j)+CO(i,j+1)-CO(i-1,j+1))/2/gdx/gdy
 CALL LOCCXX(44,8,D7,XY,CO(i,j),CO(i-1,j),CO(i,j+1),CO(i-1,j+1),gdx
*,gdy)
 DX(1)=(-1.)*1./gdy*1./gdx*1./2*(CO(i,j-1)-CO(i-1,j-1)+CO(i,j)-CO(i
*-1,j))
 DX(2)=(-1.)*1./gdy*1./gdx*1./2*XY
 DX(3)=(-1.)*1./gdy*1./gdx*1./2*XY*(-1.0D00)
 DX(4)=(-1.)*1./gdy*1./gdx*1./2*XY
 DX(5)=(-1.)*1./gdy*1./gdx*1./2*XY*(-1.0D00)
 DX(6)=(-1.)*1./gdy*(-((XY*(CO(i,j-1)-CO(i-1,j-1)+CO(i,j)-CO(i-1,j)
*))/2)/gdx/gdx)
 DX(7)=(-1.)*(-(((XY*(CO(i,j-1)-CO(i-1,j-1)+CO(i,j)-CO(i-1,j)))/2)/
*gdx)/gdy/gdy)
  D8=-XY*(CO(i,j-1)-CO(i-1,j-1)+CO(i,j)-CO(i-1,j))/2/gdx/gdy
 CALL LOCCXX(45,8,D8,XY,CO(i,j-1),CO(i-1,j-1),CO(i,j),CO(i-1,j),gdx
*,gdy)
  END IF
  IF (J.EQ.NY.AND.UY.GT.0) THEN
  D3=0
 CALL ZERRXX(46,D3)
  D4=0
 CALL ZERRXX(47,D4)
 DX(1)=1./gdy*1./gdx*1./2*(CO(i,j)-CO(i,j-1)+CO(i+1,j)-CO(i+1,j-1))
 DX(2)=1./gdy*1./gdx*1./2*XY
 DX(3)=1./gdy*1./gdx*1./2*XY*(-1.0D00)
 DX(4)=1./gdy*1./gdx*1./2*XY
 DX(5)=1./gdy*1./gdx*1./2*XY*(-1.0D00)
 DX(6)=1./gdy*(-((XY*(CO(i,j)-CO(i,j-1)+CO(i+1,j)-CO(i+1,j-1)))/2)/
*gdx/gdx)
 DX(7)=-(((XY*(CO(i,j)-CO(i,j-1)+CO(i+1,j)-CO(i+1,j-1)))/2)/gdx)/gd
*y/gdy
  D5=XY*(CO(i,j)-CO(i,j-1)+CO(i+1,j)-CO(i+1,j-1))/2/gdx/gdy
 CALL LOCCXX(48,8,D5,XY,CO(i,j),CO(i,j-1),CO(i+1,j),CO(i+1,j-1),gdx
*,gdy)
 DX(1)=(-1.)*1./gdy*1./gdx*1./2*(CO(i-1,j)-CO(i-1,j-1)+CO(i,j)-CO(i
*,j-1))
```

```
 DX(2)=(-1.)*1./gdy*1./gdx*1./2*XY
 DX(3)=(-1.)*1./gdy*1./gdx*1./2*XY*(-1.0D00)
 DX(4)=(-1.)*1./gdy*1./gdx*1./2*XY
 DX(5)=(-1.)*1./gdy*1./gdx*1./2*XY*(-1.0D00)
 DX(6)=(-1.)*1./gdy*(-((XY*(CO(i-1,j)-CO(i-1,j-1)+CO(i,j)-CO(i,j-1)
*))/2)/gdx/gdx)
 DX(7)=(-1.)*(-(((XY*(CO(i-1,j)-CO(i-1,j-1)+CO(i,j)-CO(i,j-1)))/2)/
*gdx)/gdy/gdy)
  D6=-XY*(CO(i-1,j)-CO(i-1,j-1)+CO(i,j)-CO(i,j-1))/2/gdx/gdy
 CALL LOCCXX(49,8,D6,XY,CO(i-1,j),CO(i-1,j-1),CO(i,j),CO(i,j-1),gdx
*,gdy)
  D7=0
 CALL ZERRXX(50,D7)
  D8=0
 CALL ZERRXX(51,D8)
  END IF
  IF (UX.EQ.0.AND.I.EQ.1) THEN
  D1=0
 CALL ZERRXX(52,D1)
  D6=0
 CALL ZERRXX(53,D6)
  END IF
  IF (UX.EQ.0.AND.I.EQ.NX) THEN
  D2=0
 CALL ZERRXX(54,D2)
  D5=0
 CALL ZERRXX(55,D5)
  END IF
  IF (UY.EQ.0.AND.J.EQ.1) THEN
  D3=0
 CALL ZERRXX(56,D3)
  D8=0
 CALL ZERRXX(57,D8)
  END IF
  IF (UY.EQ.0.AND.J.EQ.NY) THEN
  D4=0
 CALL ZERRXX(58,D4)
  D7=0
 CALL ZERRXX(59,D7)
  END IF
 DX(1)=1.0D00
 DX(2)=1.0D00
 DX(3)=1.0D00
```

```
      DX(4)=1.0D00
       BS1=C1+C2+C3+C4
      CALL LOCCXX(60,5,BS1,C1,C2,C3,C4)
      DX(1)=1.0D00
      DX(2)=1.0D00
      DX(3)=1.0D00
      DX(4)=1.0D00
      DX(5)=1.0D00
      DX(6)=1.0D00
      DX(7)=1.0D00
      DX(8)=1.0D00
      DX(9)=(-1.0D00)*CO(I,J)
      DX(10)=(-1.0D00)*LA
      DX(11)=1./NE*1./M*1./gdy*1./gdx
      DX(12)=1./NE*1./M*1./gdy*(-XMP(I,J)/gdx/gdx)
      DX(13)=1./NE*1./M*(-(XMP(I,J)/gdx)/gdy/gdy)
      DX(14)=1./NE*(-((XMP(I,J)/gdx)/gdy)/M/M)
      DX(15)=(-(((XMP(I,J)/gdx)/gdy)/M)/NE/NE)
       BS2=D1+D2+D3+D4+D5+D6+D7+D8-LA*CO(I,J)+XMP(I,J)/gdx/gdy/M/NE
      CALL LOCCXX(61,16,BS2,D1,D2,D3,D4,D5,D6,D7,D8,LA,CO(I,J),XMP(I,J),
     *gdx,gdy,M,NE)
      DX(1)=1.0D00
      DX(2)=1.0D00
       BS3=BS1+BS2
      CALL LOCCXX(62,3,BS3,BS1,BS2)
      DX(1)=1.0D00
      DX(2)=BS3
      DX(3)=Dt
       C(i,j)=CO(i,j)+Dt*BS3
      CALL LOCCXX(63,4,C(i,j),CO(i,j),Dt,BS3)
    5 CONTINUE
    4 CONTINUE
      DX(1)=1.0D00
      DX(2)=1.0D00
       T=T+DT
      CALL LOCCXX(64,3,T,T,DT)
       DO 6, J = 1, NY
       DO 7, I = 1, NX
      DX(1)=1.
       CO(I,J)=C(I,J)
      CALL LOCCXX(65,2,CO(I,J),C(I,J))
       CALL PRNTXX(c(12,7))
    7 CONTINUE
```

```
  6  CONTINUE
  3  CONTINUE
     DO 20, J=1,15
     DO 30, I=1,15
     LF(I)=INT(C(I,J)+0.5)
  30 CONTINUE
  20 CONTINUE
     END
     BLOCK DATA GRESS
c ADGEN control parameters
     PARAMETER (LSTTOT=2)
     PARAMETER (LOCROWS= 2)
     PARAMETER (LOCTOT= 2)
     PARAMETER (MAXROWS= 256)
     PARAMETER (MAXRES=2)
     PARAMETER (MAXPAR= 2)
c CHAIN control parameters
     PARAMETER (ITABLE = 10)
c      common blocks
     COMMON/ZZZZAA/ICSEG,ICTAB,LIMRES,LIMROW,MAXP,
    *MAXLOC,LSTMAX,LIMLOC,LIMSEG,ITABSEG,PLIST(MAXPAR)
     COMMON/ZZZZQD/ NNQQ,RDX(2*MAXROWS)
     COMMON/ZZZZQ2/NUMROW,NPBUF0,NPBUF(MAXROWS)
     COMMON/ZZZZQ3/IBUF(2*MAXROWS)
     COMMON/ZZZZQ5/HTABLE(8192,ITABLE)
     COMMON /ZZZZQB/ LOCROW,LOCBUF(LOCROWS)
     COMMON /ZZZZQC/ DBUF(LOCROWS)
     CHARACTER*28 PNAME
     COMMON/ZZZZQE/PNAME(MAXPAR)
     COMMON/ZZZZQF/PVALUE(MAXPAR)
     INTEGER PLIST,LIMRES,LIMROW,ISPV
     COMMON/ZZZZQH/IPNTR,ISPV(LSTTOT)
     DATA IPNTR/LOCTOT/
     DATA MAXLOC/LOCTOT/
     DATA LIMRES/MAXRES/
     DATA LIMROW/MAXROWS/
     DATA MAXP/MAXPAR/
     DATA LSTMAX/LSTTOT/
     DATA LIMLOC/LOCROWS/
     DATA LIMSEG/ITABLE/
     DATA ITABSEG/8192/
     END
```

APPENDIX E

Second Order Limit State p_f Calculations

Z=0

AL	AQ	UX	M	NE
1.000E+02	1.031E+01	4.307E+00	1.094E+01	2.077E-01
1.100E+02	1.031E+01	4.307E+00	1.094E+01	2.077E-01
1.000E+02	1.135E+01	4.307E+00	1.094E+01	2.077E-01
1.000E+02	1.031E+01	4.738E+00	1.094E+01	2.077E-01
1.000E+02	1.031E+01	4.307E+00	1.203E+01	2.077E-01
1.000E+02	1.031E+01	4.307E+00	1.094E+01	2.285E-01
1.000E+04	1.064E+02	1.855E+01	1.196E+02	4.313E-02
1.210E+04	1.064E+02	1.855E+01	1.196E+02	4.313E-02
1.000E+04	1.287E+02	1.855E+01	1.196E+02	4.313E-02
1.000E+04	1.064E+02	2.245E+01	1.196E+02	4.313E-02
1.000E+04	1.064E+02	1.855E+01	1.448E+02	4.313E-02
1.000E+04	1.064E+02	1.855E+01	1.196E+02	5.219E-02
-1.843E-02	-3.063E-01	-3.359E+00	-5.172E-01	-2.586E+01
-1.843E+00	-3.160E+00	-1.447E+01	-5.657E+00	-5.371E+00
-2.027E+00	-3.160E+00	-1.447E+01	-5.657E+00	-5.371E+00
-1.843E+00	-3.476E+00	-1.447E+01	-5.657E+00	-5.371E+00
-1.843E+00	-3.160E+00	-1.592E+01	-5.657E+00	-5.371E+00
-1.843E+00	-3.160E+00	-1.447E+01	-6.223E+00	-5.371E+00
-1.843E+00	-3.160E+00	-1.447E+01	-5.657E+00	-5.908E+00

Polynomial	FDGW	Epsilon
1.343E-05	3.278E+00	3.278E+00
-1.843E-01	3.249E+00	3.433E+00
-3.160E-01	3.105E+00	3.421E+00
-1.447E+00	2.993E+00	4.440E+00
-5.657E-01	2.980E+00	3.546E+00
-5.371E-01	2.980E+00	3.517E+00

ai	-4.010E-05	-5.489E-03	1.860E-01	-1.299E-03	-5.530E+00
pf	2.278E-02				

Z=2

AL	AQ	UX	M	NE
9.909E+01	1.015E+01	3.803E+00	1.061E+01	2.051E-01
1.090E+02	1.015E+01	3.803E+00	1.061E+01	2.051E-01
9.909E+01	1.116E+01	3.803E+00	1.061E+01	2.051E-01
9.909E+01	1.015E+01	4.184E+00	1.061E+01	2.051E-01
9.909E+01	1.015E+01	3.803E+00	1.167E+01	2.051E-01
9.909E+01	1.015E+01	3.803E+00	1.061E+01	2.256E-01
9.818E+03	1.030E+02	1.447E+01	1.125E+02	4.207E-02
1.188E+04	1.030E+02	1.447E+01	1.125E+02	4.207E-02
9.818E+03	1.246E+02	1.447E+01	1.125E+02	4.207E-02
9.818E+03	1.030E+02	1.750E+01	1.125E+02	4.207E-02
9.818E+03	1.030E+02	1.447E+01	1.361E+02	4.207E-02
9.818E+03	1.030E+02	1.447E+01	1.125E+02	5.090E-02
-1.843E-02	-3.063E-01	-3.359E+00	-5.172E-01	-2.586E+01
-1.826E+00	-3.108E+00	-1.278E+01	-5.485E+00	-5.304E+00
-2.008E+00	-3.108E+00	-1.278E+01	-5.485E+00	-5.304E+00
-1.826E+00	-3.419E+00	-1.278E+01	-5.485E+00	-5.304E+00
-1.826E+00	-3.108E+00	-1.405E+01	-5.485E+00	-5.304E+00
-1.826E+00	-3.108E+00	-1.278E+01	-6.034E+00	-5.304E+00
-1.826E+00	-3.108E+00	-1.278E+01	-5.485E+00	-5.834E+00

Polynomial	FDGW	Epsilon
2.000E+00	3.880E+00	1.880E+00
1.818E+00	3.841E+00	2.024E+00
1.689E+00	3.675E+00	1.985E+00
7.225E-01	3.555E+00	2.832E+00
1.452E+00	3.527E+00	2.075E+00
1.470E+00	3.527E+00	2.057E+00

ai	-4.081E-05	-5.667E-03	2.385E-01	-1.377E-03	-5.712E+00
pf	6.385E-02				

Z=4

AL	AQ	UX	M	NE
9.778E+01	9.918E+00	3.330E+00	1.018E+01	2.015E-01
1.076E+02	9.918E+00	3.330E+00	1.018E+01	2.015E-01
9.778E+01	1.091E+01	3.330E+00	1.018E+01	2.015E-01
9.778E+01	9.918E+00	3.663E+00	1.018E+01	2.015E-01
9.778E+01	9.918E+00	3.330E+00	1.119E+01	2.015E-01
9.778E+01	9.918E+00	3.330E+00	1.018E+01	2.217E-01
9.562E+03	9.837E+01	1.109E+01	1.035E+02	4.061E-02
1.157E+04	9.837E+01	1.109E+01	1.035E+02	4.061E-02
9.562E+03	1.190E+02	1.109E+01	1.035E+02	4.061E-02
9.562E+03	9.837E+01	1.342E+01	1.035E+02	4.061E-02
9.562E+03	9.837E+01	1.109E+01	1.253E+02	4.061E-02
9.562E+03	9.837E+01	1.109E+01	1.035E+02	4.914E-02
-1.843E-02	-3.063E-01	-3.359E+00	-5.172E-01	-2.586E+01
-1.802E+00	-3.038E+00	-1.119E+01	-5.262E+00	-5.212E+00
-1.982E+00	-3.038E+00	-1.119E+01	-5.262E+00	-5.212E+00
-1.802E+00	-3.342E+00	-1.119E+01	-5.262E+00	-5.212E+00
-1.802E+00	-3.038E+00	-1.230E+01	-5.262E+00	-5.212E+00
-1.802E+00	-3.038E+00	-1.119E+01	-5.789E+00	-5.212E+00
-1.802E+00	-3.038E+00	-1.119E+01	-5.262E+00	-5.733E+00

Polynomial	FDGW	Epsilon
4.000E+00	0.000E+00	-4.000E+00
3.820E+00	0.000E+00	-3.820E+00
3.696E+00	0.000E+00	-3.696E+00
2.882E+00	0.000E+00	-2.882E+00
3.474E+00	0.000E+00	-3.474E+00
3.479E+00	0.000E+00	-3.479E+00

ai -3.088E-05 -6.491E-03 2.415E-01 -4.199E-03 -1.146E+01
pf 2.221E-01

Z=6

AL	AQ	UX	M	NE
9.601E+01	9.616E+00	2.891E+00	9.641E+00	1.967E-01
1.056E+02	9.616E+00	2.891E+00	9.641E+00	1.967E-01
9.601E+01	1.058E+01	2.891E+00	9.641E+00	1.967E-01
9.601E+01	9.616E+00	3.180E+00	9.641E+00	1.967E-01
9.601E+01	9.616E+00	2.891E+00	1.061E+01	1.967E-01
9.601E+01	9.616E+00	2.891E+00	9.641E+00	2.164E-01
9.217E+03	9.246E+01	8.358E+00	9.296E+01	3.870E-02
1.115E+04	9.246E+01	8.358E+00	9.296E+01	3.870E-02
9.217E+03	1.119E+02	8.358E+00	9.296E+01	3.870E-02
9.217E+03	9.246E+01	1.011E+01	9.296E+01	3.870E-02
9.217E+03	9.246E+01	8.358E+00	1.125E+02	3.870E-02
9.217E+03	9.246E+01	8.358E+00	9.296E+01	4.682E-02
-1.843E-02	-3.063E-01	-3.359E+00	-5.172E-01	-2.586E+01
-1.769E+00	-2.945E+00	-9.711E+00	-4.986E+00	-5.087E+00
-1.946E+00	-2.945E+00	-9.711E+00	-4.986E+00	-5.087E+00
-1.769E+00	-3.240E+00	-9.711E+00	-4.986E+00	-5.087E+00
-1.769E+00	-2.945E+00	-1.068E+01	-4.986E+00	-5.087E+00
-1.769E+00	-2.945E+00	-9.711E+00	-5.485E+00	-5.087E+00
-1.769E+00	-2.945E+00	-9.711E+00	-4.986E+00	-5.596E+00

Polynomial	FDGW	Epsilon
6.000E+00	5.759E+00	-2.410E-01
5.823E+00	5.694E+00	-1.294E-01
5.705E+00	5.461E+00	-2.444E-01
5.029E+00	5.366E+00	3.368E-01
5.501E+00	5.235E+00	-2.659E-01
5.491E+00	5.235E+00	-2.559E-01

ai	-1.183E-05	-7.103E-03	2.526E-01	-8.166E-03	-1.845E+01
pf	5.052E-01				

Z=8

AL	AQ	UX	M	NE
9.364E+01	9.230E+00	2.489E+00	9.011E+00	1.904E-01
1.030E+02	9.230E+00	2.489E+00	9.011E+00	1.904E-01
9.364E+01	1.015E+01	2.489E+00	9.011E+00	1.904E-01
9.364E+01	9.230E+00	2.738E+00	9.011E+00	1.904E-01
9.364E+01	9.230E+00	2.489E+00	9.912E+00	1.904E-01
9.364E+01	9.230E+00	2.489E+00	9.011E+00	2.094E-01
8.769E+03	8.519E+01	6.196E+00	8.119E+01	3.625E-02
1.061E+04	8.519E+01	6.196E+00	8.119E+01	3.625E-02
8.769E+03	1.031E+02	6.196E+00	8.119E+01	3.625E-02
8.769E+03	8.519E+01	7.497E+00	8.119E+01	3.625E-02
8.769E+03	8.519E+01	6.196E+00	9.824E+01	3.625E-02
8.769E+03	8.519E+01	6.196E+00	8.119E+01	4.387E-02
-1.843E-02	-3.063E-01	-3.359E+00	-5.172E-01	-2.586E+01
-1.725E+00	-2.827E+00	-8.362E+00	-4.660E+00	-4.924E+00
-1.898E+00	-2.827E+00	-8.362E+00	-4.660E+00	-4.924E+00
-1.725E+00	-3.110E+00	-8.362E+00	-4.660E+00	-4.924E+00
-1.725E+00	-2.827E+00	-9.198E+00	-4.660E+00	-4.924E+00
-1.725E+00	-2.827E+00	-8.362E+00	-5.126E+00	-4.924E+00
-1.725E+00	-2.827E+00	-8.362E+00	-4.660E+00	-5.416E+00

Polynomial	FDGW	Epsilon
8.001E+00	0.000E+00	-8.001E+00
7.828E+00	0.000E+00	-7.828E+00
7.718E+00	0.000E+00	-7.718E+00
7.165E+00	0.000E+00	-7.165E+00
7.535E+00	0.000E+00	-7.535E+00
7.509E+00	0.000E+00	-7.509E+00

ai 1.969E-05 -7.308E-03 3.046E-01 -1.430E-02 -2.857E+01

pf 7.372E-01

Z=10

AL	AQ	UX	M	NE
9.06E+01	8.75E+00	2.13E+00	8.29E+00	1.82E-01
9.96E+01	8.75E+00	2.13E+00	8.29E+00	1.82E-01
9.06E+01	9.63E+00	2.13E+00	8.29E+00	1.82E-01
9.06E+01	8.75E+00	2.34E+00	8.29E+00	1.82E-01
9.06E+01	8.75E+00	2.13E+00	9.12E+00	1.82E-01
9.06E+01	8.75E+00	2.13E+00	8.29E+00	2.01E-01
8.20E+03	7.66E+01	4.52E+00	6.88E+01	3.33E-02
9.93E+03	7.66E+01	4.52E+00	6.88E+01	3.33E-02
8.20E+03	9.27E+01	4.52E+00	6.88E+01	3.33E-02
8.20E+03	7.66E+01	5.47E+00	6.88E+01	3.33E-02
8.20E+03	7.66E+01	4.52E+00	8.32E+01	3.33E-02
8.20E+03	7.66E+01	4.52E+00	6.88E+01	4.03E-02
-1.84E-02	-3.06E-01	-3.36E+00	-5.17E-01	-2.59E+01
-1.67E+00	-2.68E+00	-7.14E+00	-4.29E+00	-4.72E+00
-1.84E+00	-2.68E+00	-7.14E+00	-4.29E+00	-4.72E+00
-1.67E+00	-2.95E+00	-7.14E+00	-4.29E+00	-4.72E+00
-1.67E+00	-2.68E+00	-7.85E+00	-4.29E+00	-4.72E+00
-1.67E+00	-2.68E+00	-7.14E+00	-4.72E+00	-4.72E+00
-1.67E+00	-2.68E+00	-7.14E+00	-4.29E+00	-5.19E+00

Polynomial	FDGW	Epsilon
1.00E+01	9.17E+00	-8.26E-01
9.83E+00	8.97E+00	-8.65E-01
9.73E+00	8.65E+00	-1.09E+00
9.29E+00	8.78E+00	-5.04E-01
9.57E+00	8.34E+00	-1.23E+00
9.53E+00	8.34E+00	-1.19E+00

ai	4.12E-05	-9.34E-03	4.55E-01	-2.04E-02	-3.57E+01
pf	9.01E-01				

Z=11

AL	AQ	UX	M	NE
8.875E+01	8.485E+00	1.956E+00	7.915E+00	1.779E-01
9.763E+01	8.485E+00	1.956E+00	7.915E+00	1.779E-01
8.875E+01	9.334E+00	1.956E+00	7.915E+00	1.779E-01
8.875E+01	8.485E+00	2.152E+00	7.915E+00	1.779E-01
8.875E+01	8.485E+00	1.956E+00	8.707E+00	1.779E-01
8.875E+01	8.485E+00	1.956E+00	7.915E+00	1.957E-01
7.877E+03	7.200E+01	3.826E+00	6.265E+01	3.164E-02
9.531E+03	7.200E+01	3.826E+00	6.265E+01	3.164E-02
7.877E+03	8.712E+01	3.826E+00	6.265E+01	3.164E-02
7.877E+03	7.200E+01	4.630E+00	6.265E+01	3.164E-02
7.877E+03	7.200E+01	3.826E+00	7.581E+01	3.164E-02
7.877E+03	7.200E+01	3.826E+00	6.265E+01	3.829E-02
-1.843E-02	-3.063E-01	-3.359E+00	-5.172E-01	-2.586E+01
-1.635E+00	-2.599E+00	-6.571E+00	-4.094E+00	-4.600E+00
-1.799E+00	-2.599E+00	-6.571E+00	-4.094E+00	-4.600E+00
-1.635E+00	-2.859E+00	-6.571E+00	-4.094E+00	-4.600E+00
-1.635E+00	-2.599E+00	-7.228E+00	-4.094E+00	-4.600E+00
-1.635E+00	-2.599E+00	-6.571E+00	-4.503E+00	-4.600E+00
-1.635E+00	-2.599E+00	-6.571E+00	-4.094E+00	-5.060E+00

Polynomial	FDGW	Epsilon
1.100E+01	0.000E+00	-1.100E+01
1.084E+01	0.000E+00	-1.084E+01
1.074E+01	0.000E+00	-1.074E+01
1.034E+01	0.000E+00	-1.034E+01
1.059E+01	0.000E+00	-1.059E+01
1.054E+01	0.000E+00	-1.054E+01

ai	9.850E-05	-1.014E-02	5.629E-01	-2.605E-02	-4.364E+01
pf	9.234E-01				

Z=12

AL	AQ	UX	M	NE
8.670E+01	8.191E+00	1.797E+00	7.516E+00	1.728E-01
9.537E+01	8.191E+00	1.797E+00	7.516E+00	1.728E-01
8.670E+01	9.010E+00	1.797E+00	7.516E+00	1.728E-01
8.670E+01	8.191E+00	1.977E+00	7.516E+00	1.728E-01
8.670E+01	8.191E+00	1.797E+00	8.267E+00	1.728E-01
8.670E+01	8.191E+00	1.797E+00	7.516E+00	1.900E-01
7.518E+03	6.710E+01	3.230E+00	5.649E+01	2.985E-02
9.096E+03	6.710E+01	3.230E+00	5.649E+01	2.985E-02
7.518E+03	8.119E+01	3.230E+00	5.649E+01	2.985E-02
7.518E+03	6.710E+01	3.909E+00	5.649E+01	2.985E-02
7.518E+03	6.710E+01	3.230E+00	6.835E+01	2.985E-02
7.518E+03	6.710E+01	3.230E+00	5.649E+01	3.612E-02
-1.843E-02	-3.063E-01	-3.359E+00	-5.172E-01	-2.586E+01
-1.598E+00	-2.509E+00	-6.038E+00	-3.887E+00	-4.468E+00
-1.757E+00	-2.509E+00	-6.038E+00	-3.887E+00	-4.468E+00
-1.598E+00	-2.760E+00	-6.038E+00	-3.887E+00	-4.468E+00
-1.598E+00	-2.509E+00	-6.642E+00	-3.887E+00	-4.468E+00
-1.598E+00	-2.509E+00	-6.038E+00	-4.276E+00	-4.468E+00
-1.598E+00	-2.509E+00	-6.038E+00	-3.887E+00	-4.914E+00

Polynomial	FDGW	Epsilon
1.200E+01	1.160E+01	-4.038E-01
1.184E+01	1.148E+01	-3.619E-01
1.175E+01	1.092E+01	-8.339E-01
1.140E+01	1.138E+01	-1.572E-02
1.161E+01	1.054E+01	-1.069E+00
1.155E+01	1.054E+01	-1.011E+00

ai	1.717E-04	-1.426E-02	9.098E-01	-3.679E-02	-6.009E+01
pf	8.027E-01				

Z=14

AL	AQ	UX	M	NE
8.195E+01	7.545E+00	1.501E+00	6.703E+00	1.612E-01
9.015E+01	7.545E+00	1.501E+00	6.703E+00	1.612E-01
8.195E+01	8.300E+00	1.501E+00	6.703E+00	1.612E-01
8.195E+01	7.545E+00	1.651E+00	6.703E+00	1.612E-01
8.195E+01	7.545E+00	1.501E+00	7.374E+00	1.612E-01
8.195E+01	7.545E+00	1.501E+00	6.703E+00	1.773E-01
6.716E+03	5.693E+01	2.253E+00	4.493E+01	2.599E-02
8.126E+03	5.693E+01	2.253E+00	4.493E+01	2.599E-02
6.716E+03	6.889E+01	2.253E+00	4.493E+01	2.599E-02
6.716E+03	5.693E+01	2.726E+00	4.493E+01	2.599E-02
6.716E+03	5.693E+01	2.253E+00	5.437E+01	2.599E-02
6.716E+03	5.693E+01	2.253E+00	4.493E+01	3.145E-02
-1.843E-02	-3.063E-01	-3.359E+00	-5.172E-01	-2.586E+01
-1.510E+00	-2.311E+00	-5.042E+00	-3.467E+00	-4.169E+00
-1.661E+00	-2.311E+00	-5.042E+00	-3.467E+00	-4.169E+00
-1.510E+00	-2.542E+00	-5.042E+00	-3.467E+00	-4.169E+00
-1.510E+00	-2.311E+00	-5.546E+00	-3.467E+00	-4.169E+00
-1.510E+00	-2.311E+00	-5.042E+00	-3.814E+00	-4.169E+00
-1.510E+00	-2.311E+00	-5.042E+00	-3.467E+00	-4.586E+00

Polynomial	FDGW	Epsilon
1.400E+01	1.426E+01	2.623E-01
1.385E+01	1.412E+01	2.713E-01
1.377E+01	1.396E+01	1.939E-01
1.350E+01	1.446E+01	9.692E-01
1.365E+01	1.297E+01	-6.875E-01
1.358E+01	1.297E+01	-6.159E-01

ai	1.750E-04	1.417E-02	1.997E+00	-7.545E-02	-1.165E+02
pf	5.042E-01				

Z=16

AL	AQ	UX	M	NE
7.615E+01	6.808E+00	1.243E+00	5.833E+00	1.476E-01
8.376E+01	6.808E+00	1.243E+00	5.833E+00	1.476E-01
7.615E+01	7.489E+00	1.243E+00	5.833E+00	1.476E-01
7.615E+01	6.808E+00	1.368E+00	5.833E+00	1.476E-01
7.615E+01	6.808E+00	1.243E+00	6.417E+00	1.476E-01
7.615E+01	6.808E+00	1.243E+00	5.833E+00	1.624E-01
5.798E+03	4.635E+01	1.546E+00	3.403E+01	2.179E-02
7.016E+03	4.635E+01	1.546E+00	3.403E+01	2.179E-02
5.798E+03	5.608E+01	1.546E+00	3.403E+01	2.179E-02
5.798E+03	4.635E+01	1.871E+00	3.403E+01	2.179E-02
5.798E+03	4.635E+01	1.546E+00	4.117E+01	2.179E-02
5.798E+03	4.635E+01	1.546E+00	3.403E+01	2.636E-02
-1.843E-02	-3.063E-01	-3.359E+00	-5.172E-01	-2.586E+01
-1.403E+00	-2.085E+00	-4.177E+00	-3.017E+00	-3.817E+00
-1.543E+00	-2.085E+00	-4.177E+00	-3.017E+00	-3.817E+00
-1.403E+00	-2.294E+00	-4.177E+00	-3.017E+00	-3.817E+00
-1.403E+00	-2.085E+00	-4.594E+00	-3.017E+00	-3.817E+00
-1.403E+00	-2.085E+00	-4.177E+00	-3.319E+00	-3.817E+00
-1.403E+00	-2.085E+00	-4.177E+00	-3.017E+00	-4.199E+00

Polynomial	FDGW	Epsilon
1.600E+01	0.000E+00	-1.600E+01
1.586E+01	0.000E+00	-1.586E+01
1.579E+01	0.000E+00	-1.579E+01
1.558E+01	0.000E+00	-1.558E+01
1.570E+01	0.000E+00	-1.570E+01
1.562E+01	0.000E+00	-1.562E+01

ai	2.784E-04	1.780E-01	-2.002E+00	-9.094E-02	-1.248E+02
pf	8.778E-01				

Z=18

AL	AQ	UX	M	NE
6.933E+01	6.005E+00	1.007E+00	4.991E+00	1.322E-01
7.627E+01	6.005E+00	1.007E+00	4.991E+00	1.322E-01
6.933E+01	6.605E+00	1.007E+00	4.991E+00	1.322E-01
6.933E+01	6.005E+00	1.107E+00	4.991E+00	1.322E-01
6.933E+01	6.005E+00	1.007E+00	5.490E+00	1.322E-01
6.933E+01	6.005E+00	1.007E+00	4.991E+00	1.454E-01
4.807E+03	3.606E+01	1.014E+00	2.491E+01	1.748E-02
5.817E+03	3.606E+01	1.014E+00	2.491E+01	1.748E-02
4.807E+03	4.363E+01	1.014E+00	2.491E+01	1.748E-02
4.807E+03	3.606E+01	1.227E+00	2.491E+01	1.748E-02
4.807E+03	3.606E+01	1.014E+00	3.014E+01	1.748E-02
4.807E+03	3.606E+01	1.014E+00	2.491E+01	2.115E-02
-1.843E-02	-3.063E-01	-3.359E+00	-5.172E-01	-2.586E+01
-1.278E+00	-1.839E+00	-3.382E+00	-2.581E+00	-3.419E+00
-1.405E+00	-1.839E+00	-3.382E+00	-2.581E+00	-3.419E+00
-1.278E+00	-2.023E+00	-3.382E+00	-2.581E+00	-3.419E+00
-1.278E+00	-1.839E+00	-3.720E+00	-2.581E+00	-3.419E+00
-1.278E+00	-1.839E+00	-3.382E+00	-2.839E+00	-3.419E+00
-1.278E+00	-1.839E+00	-3.382E+00	-2.581E+00	-3.761E+00

Polynomial	FDGW	Epsilon
1.800E+01	1.726E+01	-7.439E-01
1.787E+01	1.839E+01	5.153E-01
1.782E+01	1.627E+01	-1.548E+00
1.766E+01	2.071E+01	3.050E+00
1.774E+01	1.569E+01	-2.055E+00
1.766E+01	1.569E+01	-1.971E+00

ai	8.879E-04	-1.542E-01	1.612E+01	-3.199E-01	-4.296E+02
pf	8.443E-01				

VITA

Yong Kwet Loong was born in Johor Baru, Malaysia, on May 17, 1962. He attended primary school at Sekolah Temenggong Abdul Rahman Dua and obtained his secondary school education at Maktab Sultan Abu Bakar. In August 1980, he came to the United States to study at Iowa State University of Science and Technology, in Ames, Iowa. He graduated with a B.S. in Nuclear Engineering in May 1984 and enrolled at the Georgia Institute of Technology, Atlanta, Georgia the following September. He graduated with a M.S. Nuclear Engineering in August 1985 and began working as a radiological engineer for Atlan-Tech, Inc., supporting nuclear power plant start-up and operation activities around the U.S. In the summer of 1988, he was assigned to provide technical support at Oak Ridge National Laboratory (ORNL). He enrolled in the University of Tennessee evening school program in August 1989. In 1994, he went on to work for Science Applications International Corporation (SAIC). He left SAIC in 1995 to work as a consultant, teaming up with Advanced Integrated Management Services, Inc. (AIMSI). He joined AIMSI as a staff member in April 1997 to manage the company's projects in ^{233}U safe storage and disposition. Currently, he is involved with technical activities at the ORNL Radiochemical Development Facility (RDF) and the Waste Examination and Assay Facility (WEAF).