To the Graduate Council:

I am submitting herewith a dissertation written by Xiao,Shili entitled "Bayesian Estimation of the Relative Toxicity of ^{239}Pu and ^{226}Ra with Dependent Competing Risks." 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.

Dr. P. G. Groer, Major Professor

We have read this dissertation
and recommend its acceptance:

Accepted for the Council:

Associate Vice Chancellor
and Dean of The Graduate School

BAYESIAN ESTIMATION OF THE RELATIVE TOXICITY OF ^{239}Pu AND ^{226}Ra WITH DEPENDENT COMPETING RISKS

A Dissertation

Presented for the

Doctor of Philosophy Degree

The University of Tennessee, Knoxville

Shili Xiao

May 1998

To My Parents.

ACKNOWLEDGMENTS

I wish to express my deepest and most sincere gratitude to my major adviser, Dr. P. G. Groer. His support, technical advice and encouragement throughout the entire period of this research is invaluable to me. I further thank Dr. Groer for his continued guidance, both academic and personal.

I would also like to thank my dissertation committee members Dr. L. F. Miller, Dr. B.R.Upadhyaya of the Nuclear Engineering Department, and Dr. K. F. Eckerman and Dr. D. J. Downing of Oak Ridge National Laboratory (ORNL). Their suggestions and contributions to the present work have been most valuable. I would also like to thank Dr. J. E. Turner of ORNL for his guidance, both academic and personal. Many thanks to the previous Head of the Department of Nuclear Engineering, Dr. T. W. Kerlin, and current Head, Dr. Dodds, for their support, suggestions and their guidance.

I owe a special debt of thanks to Dr. P. N. Stevens, Dr. R. B. Perez, Dr. L. W. Townsend, Dr. R. E. Uhrig and Dr. J. W. Hines for teaching me several enjoyable classes. I have also to thank Mr. R. P. Osborne of Radiological Safety Department of UTK for his guidance and supports.

In the pursuit of this degree, I have also benefited from the help of many my fellow students. In particular, I wish to mention Yan Wu and Xu Xiao. Their help are very much appreciated.

Last, but not the least, I wish to thank my husband, my son, and my all family members for their loving supports and patience during my graduate study.

ABSTRACT

The purpose of this dissertation research is to compare the toxicity of the α-emitting, bone-seeking radionuclides ^{239}Pu and ^{226}Ra, develop a model for radiation induced osteosarcomas, and analyze the survival data of beagles exposed to these radionuclides. This research integrates the knowledge of radiation protection, survival theory and methods (competing risks, maximum likelihood estimation, and Bayesian techniques), numerical integration techniques (Monte Carlo, Lattice rule and Gauss-quadrature) and object-oriented programming in C++. The outline of this research is: (1) survival data preprocessing, (2) model identification and selection, (3) introduction of FGM model, the dependent competing risk model created by Farlie, Gumbel and Morgenstern , to the study of survival data with dependent competing risks: osteosarcomas and other diseases, development of the crude density of the FGM model and construction of the likelihood function for the FGM model, (4) Bayesian estimates of the posterior marginal density of the toxicity ratio in the FGM model using several numerical integration techniques (Monte Carlo, Lattice rule and Gaussian Quadrature), (5) construction of the likelihood function for the independent competing risk model, Bayesian estimate of the posterior marginal density of toxicity ratio in the model using Monte Carlo method, which is compared with the posterior marginal densities for the toxicity ratio obtained from the FGM model, (6) Bayesian estimates of all other parameters in the FGM model using Monte Carlo method, (7) Comparison of the cumulative hazard for ^{239}Pu calculated according to the model with Nelson's cumulative hazard plot under Bayesian point estimates of parameters and the mean activity in each injection level. (8) Comparison of the toxicity of plutonium in osteosarcoma with that of

radium under Bayesian point estimates of parameters and the selected activit of 0.85 μC_i, (7) discuss Bayesian prediction of the cumulative failure distribution.

It is the first time for the FGM model and the independent competing risk model to compare the toxicity between ^{239}Pu and ^{226}Ra. Bayesian parameter estimates for these models are also the original contribution of this research. The approach is novel in the field of radiation risk assessment, particularly, for toxicity studies.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE PAGE

LIST OF TABLES

TABLE PAGE

CHAPTER 1. INTRODUCTION

1.1. Motivation and Importance of the Research

The study of bone-seeking radionuclides has engaged scientists and physicians for over 75 years. Bone-seeking radionuclides are those that have the skeleton as a primary, or at least a major deposition site in the body [1, 4]. The induction of bone cancer is the primary risk from these radionuclides. Regardless of the intake route, once a bone seeker has entered the blood stream, it is quickly and efficiently transferred to bone and once deposited in bone, the radionuclides are retained for long times and induce bone cancers. Beginning with in-vivo measurements of radium and exhaled radon in radium workers in 1920's (Martland et al., 1925) to recent findings in workers exposed to transuranic elements (Toohey and Kathren, 1994), the study of the measurement, metabolism, dosimetry and health effects of bone seekers has been continuous.

Radium toxicity has been studied for more than a half-century, and it continues to be of great interest not only for its radioepidemiological significance but also because of radium's ubiquity in our environment. The history of radiation protection is intimately associated with epidemiological studies of radium toxicity [1]. The studies of radium workers (Evans, 1933) took on new importance in the early days of World War II when it was realized that the new radionuclides to be mass-produced, such as ^{239}Pu, may behave similarly in the body and have similar effects as radium [4].

As a result of nuclear weapons tests, measurable quantities of the man-made element, plutonium, are found in foods and in human tissues [2]. However, no data are

given in the Reference Man Report (ICRP Publication 23) for this element. Some data on plutonium are available for humans from occupationally exposed persons. A great deal of metabolic information comes from animal experiments. Generally, the data that were analyzed were obtained by exposing beagles. Beagles are favored because of their relatively long life span which would allow the expression of disease and because the skeleton of the beagle approximates that of the human in its structure and physiology [3].

Large-scale animal studies were begun in conjunction with the continued follow-up of the radium workers, based on the following hypothesis [4]:

$$Toxicity\ of\ plutonium\ in\ man = \frac{Toxicity\ of\ radium\ in\ man \times Toxicity\ of\ plutonium\ in\ dog}{Toxicity\ of\ Radium\ in\ dog} \tag{1.1}$$

These toxicity ratios, dubbed "Robley Ratios" in honor of Robley D. Evans, have been used for many bone seeking radionuclides and developed for many species. By determining the toxicity ratio for radium between dogs and the humans, the risk to humans for plutonium can be extrapolated from the animal data. This formula also indicates the importance of comparing toxicity in dog for plutonium and radium to predict the plutonium risk to humans. In this research, the toxicity in dog's skeleton from plutonium is compared with that from radium.

The data for this research are obtained from beagles exposed to ^{239}Pu and ^{226}Ra respectively. The experiments are performed at the University of Utah [3]. In this experiment, dogs are exposed to graded injection levels. There are nine injection levels from 0 to 5 for ^{239}Pu (0.00051-3.3 μCi/kg in 278 dogs) and eleven injection levels from 0

to 5 for ^{226}Ra (0.00577-11.9 μCi/kg in 140 dogs). At each level, about one half of the dogs are males. Uninjected, age-matched dogs served as controls. For each beagle the sex, the group number, the code letter for the radionuclide, the injection level, the age at time of exposure to ^{239}Pu or ^{226}Ra (in days), the weight, the time at risk (in days), the injected activity per kg body weight (in μCi /kg) and the cause of death are recorded. The injected activity per kg body weight is determined by the "retained" activities, the maximum permissible concentration of ^{226}Ra in man [3]. The time at risk is defined as the elapsed time (in days) from first exposure to the time of death from osteosarcoma or from an other disease [3]. There are therefore at least two causes of death: death from an osteosarcoma or from an other disease. For each individual beagle, the cause of death, the time at risk and total accepted activity are available.

Since bone cancer is the primarily biological response of the two radionuclides[4], the relative toxicity ratio is in terms of the bone cancer risk ratio. The biological effect of radiation can be divided into two general categories: stochastic and deterministic, or nonstochastic [2]. Cancer is a stochastic biological effect of ionizing radiation ,which can be analyzed by estimating the parameters of a survival model.

If the collected survival data include both the time to death and the corresponding cause of death for each individual, like the experimental data used in this research, then analyzing and interpreting this kind of survival data involve competing risks [5, 9]. Much work in survival theory is based on an assumption of independence, e.g. independent functioning of causes of death. However, this is an unverified assumption and often not likely to be true generally, where failure of one type affects the probability of failure of

another type. A large number of clinical experimental studies are analyzed assuming independence. In addition, there are many other examples for dependent competing risks in biometry, radiation risk analysis and reliability analysis of a technological system consisting of n components in series [6]. The theory of competing risks includes: (a) nonparametric approaches, (b) parametric approaches - independent competing risks, (c) parametric approaches - dependent competing risks. There are methods, such as, Nelson's hazard plots [7] and total-time -on-test plots [8], which provide simple means of checking the distributional assumption in the case of independent risks. There are also various parametric probability models, like the Exponential, Normal and the Gompertz model. Particularly the Weibull life distribution has been widely used in many fields such as life science, industrial statistics, physics, etc [9]. For the dependent competing risk problem, the Marshall-Olkin multivariate exponential distribution is one of the few simple multivariate life distributions admitting dependence [10]. A multivariate Weibull distribution has been espoused by several authors, and a rather general method for obtaining new multivariate life distributions was outlined by Lee and Thompson [11]. Furthermore, many models (e.g. the Farlie-Gumbel-Morgenstern model (FGM)) for association in bivariate failure data are reproduced and the details discussed [12]. Oakes [13] introduced the bivariate distribution with uniform marginals [14], and Frank's family of bivariate distributions [15]. Since Slud and Byar [16] showed that the dependent causes of death can make risk factors appear protective, it is important to study models for dependent causes of death.

However, most data analysis performed with the above models and methods are based on the assumption of independent competing risks. For dependent competing risks, a lot of theoretical work has been done. So far no work has been done on practical problems, like estimation of toxicity ratios where it could be necessary to use dependent competing risk models.

Over the years survival analysis methods based on sampling theory have been found to be extremely useful for a wide variety of problems. The basic idea is that the unknown parameters are assumed to be fixed constants or a constant vector. A point estimator is then derived according to some principle, such as maximum likelihood (ML), minimum variance, least squares, or the method of moments [17]. However, as what pointed by Bonis (1966,1975) and Schafer (1973) [18], there are two major disadvantages due to the nature of sampling theory: (a) It is inappropriate for analysis based on scarce data, (b) Information available before the current experimental data are collected can not be used. For reasons such as these, clauses in military and government reliability contracts permit other methods, such as Bayesian procedures, with the onus of justification put on the producer[18].

The scope of this research concerns primarily with the application of one particular dependent competing risk model, the FGM model, to compare the toxicity of ^{239}Pu and ^{226}Ra .

1.2. Objectives and Research Tasks

The primary goal of the research is to compare the toxicity in beagles of plutonium and radium on the formation of osteosarcomas. For this goal, the survival theory and its methods are applied to the toxicity analysis of these radionuclides. Different models are studied and different methods for parameter estimation are compared. This research integrates data preprocessing, nonparametric analysis, identification of life distribution models, application of independent and dependent competing risk models, maximum likelihood estimations of parameters (MLE), Bayesian parameter estimation and numerical integration methods.

The key steps in the survival data analysis include the following: (1) Summarize the experimental data. (2) Select life distribution model by using non-parametric diagnostic plots, maximum likelihood estimates and maximum likelihood ratio. (3) Derive the parametric cumulative hazard, survival function and the failure distribution for a relative risk model with a Weibull life distribution as baseline. (4) Derive the crude density from the joint density of FGM model and independent model. (5) Construct the Likelihood function respectively for an independent competing risk model and the dependent FGM model. (6) Bayesian estimates for the toxicity ratio with the dependent FGM model and three different numerical integration methods (Monte Carlo, Gauss Quadrature and Lattice method), and develop the corresponding computer programs. (7) Compare the toxicity ratios obtained by three different numerical integration methods. (8) Bayesian estimates for the toxicity ratio with the independent competing risk model. (9) Compare the toxicity ratios obtained from the independent

competing risk model and the dependent FGM competing risk model by normalizing their marginal posterior densities. (10) Estimate the parameters in the dependent FGM competing risk model by Bayesian techniques and develop corresponding computer programs. (11) Estimate the parameters in the independent competing risk model by Bayesian techniques and develop corresponding computer programs. (12) Compare these parameter estimetes for the independent competing risk model and the dependent FGM competing risk model. (13) Derive predictive probability distributions according to the Bayesian prediction technique. (14) Using the modes of the posterior densities as point estimates for the parameters in the FGM model, the cumulative hazard for ^{239}Pu is calculated with the mean activity for each injection level, and compared with Nelson's cumulative hazard plot. (15) Using the modes of the posterior densities as point estimates of parameters, the toxicity of plutonium in osteosarcoma is predicted and compared with that of radium by determining the cumulative hazard, the survival probability, the cumulative distribution and the failure density under the activity of 0.85 μC_i. (16) State the conclusions about the toxicity comparison of ^{239}Pu and ^{226}Ra and discussions and make suggestions about future research.

1.3. Original Contributions of the Dissertation.

The comparison of toxicity between ^{239}Pu and ^{226}Ra with the independent competing risk model, the dependent FGM competing risk model, and the use of Bayesian techniques are the original contributions of this research. This approach is novel in the field of radiation risk assessment, particularly, toxicity studies.

The original contributions of this dissertation are listed below:

1. Development of the crude density for the FGM model.
2. Development of the net density for the FGM model from the relative risk model based on Weibull distribution.
3. Construction of the likelihood function for the given survival data with the independent competing risk model and dependent FGM model.
4. Bayesian estimation of the toxicity ratio for plutonium and radium in the presence of dependent competing risks using three different numerical integration methods.
5. Comparison of the toxicity ratio obtained from the dependent and the independent competing risk model.
6. Bayesian estimation of other parameters for the relative risk model using the FGM model, and comparison of the results with that obtained from the independent competing risk model.
7. Comparison of the cumulative hazard for ^{239}Pu calculated according to the model with Nelson's cumulative hazard plot under Bayesian point estimates of parameters and the mean activity in each injection level.
8. Comparison of the toxicity of plutonium in osteosarcoma with that of radium under Bayesian point estimates of parameters and the selected activity of 0.85 μC_i.

1.4. Organization of the Dissertation

This dissertation is organized into four chapters and three appendixes. Following the introductory chapter, a literature review of the applied theory and methods are given in chapter 2, including life distributions, competing risk models, maximum likelihood estimate (MLE), Bayesian techniques and numerical integration methods. Chapter 3 gives a detail description of the modeling and analysis of the survival data applied in this research. This is the main part of the dissertation. A summary of the research is stated in chapter 4, including conclusions, discussions and suggestions for future research. Appendix A is the derivations of crude densities for FGM model, Appendix B is the collected survival data, and Appendix C lists computer codes for the data analysis

CHAPTER 2. REVIEW OF APPLIED THEORY AND METHODS

2.1. Survival Probability and Survival Data

Survival probability is defined as the probability that an item will live under stated conditions for a stated period of time[18]. When applied to human being or life test, it usually refers to person's or animal's ability to perform certain tasks according to a specified standard. By extension, survival probability is applied to a piece of equipment, or a component of a large system, to mean the ability of that equipment or component to fulfill what is required of it.

The important feature of survival data is the time at risk which assumes that aging occurs during operation. There are three distinctions to be made here concerning survival data [19]: 1. Noncensored versus censored data. It is customary to stop an experiment before all the units have failed, in which case only a lower bound is known for the failure load or failure time of the unfailed units. Such data are called right-censored. In other contexts, only an upper bound for the failure time may be known (left-censored), or it may be known only that failure occurred between two specified times (interval-censored data). 2. Single samples versus data with covariates. A single sample means the sample of lifetime of units tested under identical conditions. In many contexts, however, there are additional explanatory variables or covariates. For example, there may be several different samples conducted with slightly different materials or under different stresses or different ambient conditions. This leads us into the study of survival models which incorporate covariates. 3. Univariate and multivariate data. If the

variable being measured is a scalar, such as a single lifetime, this is called univariate data. Otherwise, the variable is a vector of observations, such as lifetimes of different components or failure stresses in different directions or lifetimes for different causes of death, relevant to the survival probability of a single unit, this kind of data are multivariate data.

According to the above discussion, the survival data in this research are multivariale data with a covariate, because of the lifetime for different causes of death and the different samples conducted under different injected activities. The lifetimes for different causes of death makes the survival data multivariate. The injected activity is the covariate.

2.2. Probability Distributions in Survival Analysis

2.2.1. Preliminaries on Life Distribution

In this part, some important concepts in survival analysis are defined .

Let T be the random time to failure of a unit under study. Here time is used in its most general sense. It may be real time or operational time or indeed any non-negative time metameter variable, such as breaking strain or number of revolutions to failure. Let

$$F(t) = \Pr(T < t) \tag{2.1}$$

be the distribution function of T and let

$$S(t) = \Pr(T \geq t) = 1 - F(t) \tag{2.2}$$

be the survivor function or reliability function of T. Assuming that T has a density function

be the survivor function or reliability function of T. Assuming that T has a density function

$$f(t) = \frac{dF(t)}{dt} = -\frac{dS(t)}{dt} \tag{2.3}$$

so that the probability that a unit fails in the short time interval [t, t + δt) is

$$\Pr(t \le T < t + \delta t) \cong f(t)\delta\, t \tag{2.4}$$

Consider now the same event, that is t ≤ T < t + δt, but this time conditional on the fact that the unit has not failed by time t. That is

$$\Pr(t \le T < t + \delta t | T \ge t) \cong \frac{f(t)\delta t}{S(t)} \tag{2.5}$$

This may be thought of as the probability of imminent failure at time t. The function h (t) given by

$$h(t) = \frac{f(t)}{S(t)} \tag{2.6}$$

is the hazard or failure rate function, and is a natural indicator of the 'proneness to failure' of a unit after t has elapsed [19]. The cumulative hazard function is

$$H(t) = \int_0^t h(u)du \tag{2.7}$$

and it is readily seen that

$$S(t) = \exp\{-H(t)\} \tag{2.8}$$

The f, F, S, h and H give mathematically equivalent descriptions of T in the sense that, given any one of these functions the other four functions may be deduced.

Some typical cases are briefly discussed below.

1. If $h(t) = \lambda$, a constant, the $H(t) = \lambda t$ and $S(t)=\exp(-\lambda t)$, the survivor function of exponential distribution with rate parameter λ. The corresponding density function is

$$f(t) = \lambda e^{-\lambda t} \tag{2.9}$$

Thus, an exponential life distribution corresponds to no aging, and is a natural starting point in survival modeling.

2. If h(t) is an increasing function of t, then T is said to have an increasing failure rate (IFR). This is appropriate when the unit is subject to aging, perhaps through wear, fatigue or cumulative damage.

3. If h(t) is a decreasing function of t, then T is said to have an decreasing failure rate (DFR). This may occur, for example, when a manufacturing process initially produces an appreciable proportion of low quality units which are likely to fail early. After a while, higher quality components remain, which have lower failure rates. This is a common situation with some electronic devices. In such cases, initial proof-testing of units is often carried out in order to eliminate the substandard units.

2.2.2. The Weibull Distribution

In survival analysis, generalizations of the exponential distribution, such as the gamma and Weibull distributions have proved to be valuable models. The Weibull distribution is probably the most widely used distribution in survival analysis [19]. It has been found to provide a reasonable model for lifetimes of many types of units, such as beagles exposed to radiation, vacuum tubes, ball bearings and composite materials and for many human cancers. The closed form of Weibull density function and the wide

variety of shapes exhibited by Weibull density functions make it a particularly convenient generalization of the exponential distribution.

A Weibull random variable has survival function

$$S(t) = \exp\{-(t/\alpha)^{\eta}\} \tag{2.10}$$

for t > 0 and where α and η are positive parameters, α being a scale parameter and η being a shape parameter. When $\eta = 1$, the exponential distribution with $\lambda = 1/\alpha$ is obtained.

The Weibull hazard function is

$$h(t) = \eta\, \alpha^{-\eta}\, t^{\eta-1} \tag{2.11}$$

This is DFR for $\eta < 1$, constant for $\eta = 1$(exponential) and IFR for $\eta > 1$. In particular, for $1<\eta<2$, the hazard function increases slower than linearly; for $\eta=2$ the hazard function is linear, and for $\eta>2$ the hazard function increases faster than linearly[19].

The Weibull density (for t > 0) is

$$f(t) = \eta\alpha^{-\eta}\, t^{\eta-1} \exp\{-(\frac{t}{\alpha})^{\eta}\} \tag{2.12}$$

The Weibull cumulative hazard is

$$H(t) = (\frac{t}{\alpha})^{\eta} \tag{2.13}$$

or
$$\ln(H(t)) = \eta \ln(t) - \eta \ln(\alpha) \tag{2.14}$$

Equation (2.13) and equation (2.14) show that the cumulative hazard plot should appear as convex or concave, and the plot of log cumulative hazard against log lifetime appears

as a straight line for Weibull distribution. These are important features of Weibull distributions which are used in the field of model identification and selection.

2.3. *Statistical Models and Methods in Survival Analysis*

2.3.1. Nonparametric Estimation

Nonparametric estimation does not depend on assumptions relating to any specific probability model [19]. It is natural to use the survival function and the cumulative hazard function as the basis for graphical displays [19]. They may suggest the form of a probability model for the data. After fitting a probability model, they may allow a simple assessment of adequacy of the fitted model. In addition, in certain circumstances crude parameter estimates for a model may be obtained graphically. Two separate techniques are employed. The first technique plots an estimate of the cumulative hazard versus time [7], while the second procedure plots a scaled total time on test versus fraction of endpoints of interest observations [8]. The second probability plotting method is used to verify the results of the first one.

Cumulative hazard plots

Assuming independence of the times of death, Nelson estimates $H(t_j, d)$, cumulative hazard at time, in the following manner:

$$H(t_j)_k = \sum_{i=1}^{j} \frac{1}{RR_{ik}} \tag{2.15}$$

where the RR_i is the reverse rank of the *i*th time to death for the *k*th cause of death with the times to death arranged in ascending order.

From equation (2.15) we have

$$\ln H(t_j)_k = \ln(\sum_{i=1}^{j} \frac{1}{RR_{ik}}) \tag{2.16}$$

According to equation (2.14), if a Weibull model is appropriate for the data, the plot of $\ln(\sum_{i=1}^{j} \frac{1}{RR_i})$ against $\ln(t_j)$ should be roughly linear. Further, if the slope and the intercept of the plot are a and b respectively, then rough estimates of η and α are a and exp(-b/a) respectively .

The standard deviation for the Nelson estimate is calculated in a similar manner. Let $\sigma(t, d)$ be the standard deviation then [7]

$$\sigma(t_j) = \left[\sum_{i=1}^{j} \frac{1}{(RR_i)^2}\right]^{1/2} \quad . \tag{2.17}$$

Total time on test plots

The total time on test procedure as well as the Nelson procedure makes the assumptions that the random times to the observation of interest are independent from the censoring times [8]. The scaled total time on test transform is given by the equation:

$$TTT_{scaled_i} = \frac{TTT_i}{TTTT} \tag{2.18}$$

where TTT_i is the total time on test up to time t_i. Let n is the total number of items up to the last item to die with the specified endpoint of interest, the TTT_i is expressed as following:

$$TTT_i = \sum_{j=1}^{i} t_j + (n-j)t_i \qquad (2,19)$$

TTTT is the sum of all the times up to the last failure caused by the specific endpoint of interest, the "total" total time on test.

The analysis is made by looking at the shape of the graph. If the graph is convex the hazard rate is increasing, while if concave the hazard rate is decreasing. If the hazard rate is constant in time, the total time on test plot is linear along the 45^0 line.

2.3.2. Regression Models for Survival Data

In many applications of survival analysis, our real interest is in determining the way in which strength or lifetime depends on other variables, some of which may be under the operator's control. For example, failure time depends on the load applied to a component. It is almost always the case that increasing the load reduces the time to failure, but it is of interest to have a specific model for this. In this context, load is a covariate or explanatory variable. The regression models are used to deal with survival data involving covariates.

Assuming the variable of interest is a failure time T, and the covariates are represented quantitatively by a vector of x, the purpose is to determine how T varies with x using an appropriate model for the dependence of T on x and relevant data (t_1, x_1), (t_2, x_2), ... The data may then be used to estimate the unknown parameters of the model to appraise the fit of the model and if necessary to suggest modifications of the model. The

regression model [19], including accelerated life model and proportional model, is often applied in survival analysis.

Accelerated Life Models

The accelerated life model holds when the survival function is of the form

$$S(t; x) = S_0(t\ \varphi_x) \tag{2.20}$$

where S_0 is some baseline survivor function and φ_x is a positive function of x. The interpretation is that, compared with some standardized system in which $\varphi_x = 1$, the lifetime is divided by a factor φ_x or equivalently, the decay of the system is accelerated by a factor φ_x.

Proportional Hazards Models

In the proportional hazards models the hazard function satisfies $h(t ; x) = h_0(t)\ \varphi_x$. Here h (t; x) is the hazard under factors x, $h_0(t)$ is some baseline hazard function, and φ_x is some positive function of x. Since $h(t ; x_1) / h(t ; x_2) = \varphi_{x1} / \varphi_{x2}$ is independent of t, the hazards at different x values are in constant proportion over time, hence the name. Provided $\varphi_0 > 0$, there is no loss of generality in assuming $\varphi_0 = 1$ since one can redefine $h(t; x) = h_0(t)\ \varphi_x / \varphi_0$. The survivor function satisfies $S(t ; x) = S_0(t)^{\varphi x}$ for this model, S_0 being the baseline survivor function corresponding to h_0.

Suppose that S (t ; x) is both an accelerated life and a proportional hazards model, so that

$$S(t;x) = S_{a0}(t\,\varphi_x) = S_{p0}(t)^{\varphi_x} \tag{2.21}$$

with the baseline functions standardized so that $\varphi_{p0} = \varphi_{a0} = 1$. Then

$$S_{a0}(t) = S_{p0}(t) = \exp(-t^{\eta}) \tag{2.22}$$

for some $\eta > 0$ (Weibull survivor function) and $\varphi_{px} = \varphi_{ax}^{\eta}$. For a proof see Cox and Oakes (1984, section 5.3) [19]. From equation (2.21), we have

$$\ln(S(t;x)) = \phi(x)\ln(S_{p0}(t;x)) \ . \tag{2.23}$$

According to equation(2.8), the above equation is rewritten as

$$H(t;x) = \phi(x)\, H_0(t;x) \tag{2.24}$$

or

$$\ln(H(t;x) = \ln(H_0(t;x)) + \ln(\phi(x)). \tag{2.25}$$

Suppose that the data are grouped according to different loads, and a regression model is appropriate. Then the plots of H(t; x) against t for the different groups should resemble multiples of baseline curve $H_0(t)$ against t. Alternatively, the plots of log(H(t; x)) against log (t) should resemble vertically shifted copies of log (H_0(t; x)) against log(t).

Models based on the Weibull distribution

One simple form of Weibull model uses the survivor function

$$S(t;x) = \exp\{-(t/\alpha_x)^{\eta}\} \tag{2.26}$$

where we may take log $\alpha_x = x^T \beta$ (loglinear model) and η independent of x. This is a full parametric model with parameters β (regression coefficient) and η (Weibull shape parameter). The hazard function is

$$h(t;x) = \eta\, t^{\eta-1}/\alpha_x^{\eta} \tag{2.27}$$

Let $S_0(t) = \exp(-t^{\eta})$, then S(t; x) can be expressed as $S_0(t\,\varphi_{ax})$ with $\varphi_{ax} = 1/\alpha_x$, an accelerated life model, or $S_0(t)^{\varphi x}$ with $\varphi_{px} = \alpha_x^{-\eta}$, a proportional hazards model[19]. The cumulative hazard is

$$\ln(H(t,x)) = \eta \ln(t) - \eta \ln(\alpha_x) \tag{2.28}$$

Replacing the second term on the right side of the above equation with parameter ξ:

$$\xi = -\eta \ln(\alpha_x) \tag{2.29}$$

the cumulative hazard is rewritten in the following form:

$$\ln(H(t,x)) = \eta \ln(t) + \xi \tag{2.30}$$

Suppose a regression model based on the Weibull distribution is appropriate for grouped data. Then the plots of log(H(t; x)) versus log(t) for different loads should resemble parallel straight lines, and the intercept ξ is related to the load dependence of the survival data.

2.3.3. Multivariate Models (Competing Risks)

In some cases there is more than a single failure time to be recorded for a system, so that the multivariate models need to be considered. There might be times to failure of different types, or of different components, or of partial failures or other associated events. Suppose that there are p potential failure times $T_1, T_2, \ldots\ldots$, T p associated with a system. If a system has components in parallel. The system continues to operate until all p components have failed. It may be possible to observe some or all of the individual component failure times before final failure, which occurs at time V = max ($T_1, T_2, \ldots\ldots$, T p) [19, 20]. If a system has components in series, the system fails as soon as any one

component fails, this being at time U = min (T_1, T_2, ... , T_p). One system is subject to competing risks when the system failures may be associated with one of p different causes, type of failure modes. T_j represents the time to system failure from cause j. The observations comprise (C_s , U_s), the cause and time of the sth failure, for s = 1, 2, ..., r. The T_j is defined as net life, and the U or V is called crude life [19, 20]. They are described respectively by net density and crude density (or subfailure density) [20].

Much work in survival analysis involving complex systems is based on an assumption of independence, e.g. independent functioning of components or causes of death. It means failure of one type does not affect the probability of failure of another type. For the situation with two independent competing risks, if t_1, t_2 represent the time since exposure until death because of cause one and cause two respectively, then $f_1(t_1)$, $f_2(t_2)$ are the corresponding marginal densities (net densities) then the joint density of net lives is defined as:

$$f_T(t_1, t_2) = f_1(t_1) * f_2(t_2). \tag{2.31}$$

The crude density for one item ,which died at time t because of the cause one, is

$$\int_t^\infty f_1(t) f_2(t_2)\, dt_2 = f_1(t) S_2(t) \tag{2.32}$$

where $S_2(t)$ is the net survival function for cause two.

However, independence of causes of death is often not likely to be true, particularly when there is load-sharing between components or randomly fluctuating loads, or generally when failures of one type affect the probability of failure of another type. There are a few distributions containing the possibility of dependence between the

T_j. Various hazard functions, failure rates, survival functions have found application in this literature. For example, Gumbel suggested three different bivariate exponential forms with joint survivor functions as following[19]:

1. $S(t_1,t_2) = \exp(-t_1 - t_2 - \delta\, t_1 t_2)$. The dependent parameter δ satisfies $0 \le \delta \le 1$, $\delta = 0$ yielding independence of T_1 and T_2.

2. $S(t) = e^{-t_1 - t_2}\{1 + \delta\,(1 - e^{-t_1})(1 - e^{-t_2})\}$. Here, $-1 \le \delta \le 1$ with $\delta = 0$ yielding independence.

3. $S(t) = \exp\{-(t_1^{\delta} + t_2^{\delta})^{1/\delta}\}$. Here $\delta \ge 1$ with $\delta = 1$ yielding independence.

These survivor functions have been given in $S(t) = \exp\{-\lambda_1 t_1 - \lambda_2 t_2 - \lambda_{12} \max(t_1,t_2)\}$ standard form. Marshall and Olkin derived the joint survivor function for bivariate exponential model

$$S(t) = \exp\{-\lambda_1 t_1 - \lambda_2 t_2 - \lambda_{12} \max(t_1,t_2)\} \tag{2.33}$$

to describe a two-component system subject to randomly occurring shocks.

FGM Model

The FGM model was created by Farlie, Gumbel and Morgenstern. A multivariate generalization of the Farlie-Gumbel-Morgenstern system of distributions and its properties are investigated by Johnson and Kotz [12].

Morgenstern (1956), Farlie (1960) and Gumbel (1958) have discussed families of bivariate distributions of the form

$$F_{\tilde{T}}(\tilde{t}) = F_{T_1,T_2}(t_1,t_2)$$

$$= F_1(t_1)F_2(t_2)[1+\delta\, S_1(t_1)S_2(t_2)] \qquad (0<\delta\le 1) \tag{2.34}$$

where T_1 and T_2 are time since exposure until death because of cause one and cause two; $F_T(t)$ is the joint cumulative distribution function of T_1 and T_2, $F_j(t_j) = F_{xj}(t_j)$, and $S_j(t_j) = 1 - F_j(t_j)$: j = 1, 2. Constraints on the values of the δ's are needed to ensure that the $F_T(t)$ is a nondecreasing function of each t_1, t_2 [12].

It is easy to see that

$$\begin{aligned}
&\lim_{t_i\to-\infty} F_{\tilde{T}}(\tilde{t}) = 0 \qquad i = 1,2.\\
&F_{\tilde{T}}(\infty,\infty) = 1\\
&\lim_{t_2\to\infty} F_{\tilde{T}}(t_1,t_2) = F_{T_1}(t_1)\\
&\lim_{t_1\to\infty} F_{\tilde{T}}(t_1,t_2) = F_{T_2}(t_2)
\end{aligned} \tag{2.35}$$

In the two-dimensional case we have the well-known symmetrical relationships

$$\begin{aligned}
F_{T_1,T_2}(t_1,t_2) &= F_1(t_1)F_2(t_2)[1+\alpha\,\{1-F_{X_1}(t_1)\}\{1-F_{X_2}(t_2)\}]\\
S_{T_1,T_2}(t_1,t_2) &= S_1(t_1)S_2(t_2)[1+\alpha\,\{1-S_1(t_1)\}\{1-S_2(t_2)\}]
\end{aligned} \tag{2.36}$$

The joint density of T_1 and T_2 is:

$$f_T(t_1,t_2) = f_1(t_1)f_2(t_2)\{1+\delta[1-2F_1(t_1)][1-2F_2(t_2)]\} \tag{2.37}$$

where $f_1(t_1)$, $f_2(t_2)$ are densities of T_1 and T_2 corresponding to $F_1(t_1)$, $F_2(t_2)$.

In this research, the FGM model is chosen to describe the dependent relationship between two competing risks. One cause of death for irradiated beagles is osteosarcoma, a kind of bone cancer, the other one represents all of other diseases and cancer. The reason for choosing this model is that it is an independent model plus a perturbation term

describing the dependence of risks, and the well-known symmetrical relationships (2.36) in the two-dimensional case.

2.4. Maximum Likelihood Estimation

2.4.1. Likelihood Function

There is a general method of estimation of parameters, called maximum likelihood estimation (MLE) [19]. Assuming t_i is lifetime of i'th item, and all of items can be regarded as observations with common density function $f(t; \theta_1, \theta_2, \ldots, \theta_m)$ where the form of f is known but where the parameters $\theta_1, \theta_2, \ldots, \theta_m$ are unknown. For brevity the dependence on $\theta_1, \theta_2, \ldots, \theta_m$ is denoted by θ. The likelihood function is defined by [19]

$$L(\theta) = \prod_{i=1}^{n} f(t_i; \theta). \tag{2.38}$$

More generally, suppose that some of observations are right-censored. Then the observations 1, 2, ..., n is split into two disjoint sets, one, U say, corresponding to observations that are uncensored, the other, C say, corresponding to right-censored observations. Then the likelihood function in this case is defined by [19]

$$L(\vartheta) = \{\prod_{i \in u} f(t_i; \vartheta)\}\{\prod_{i \in c} S(t_i; \vartheta)\} \tag{2.39}$$

Thus, for a right-censored observation the density has been replaced by the survivor function. In a similar manner for a left-censored observation the density should be replaced by the distribution function [19]. For an interval-censored observation the density should be replaced by the distribution function evaluated at the upper end-point

of the interval minus the distribution function evaluated at the lower end-point of the interval, thus yielding the probability of occurrence of a lifetime within the interval.

For the independent competing risk model, treating the lifetimes of the other causes of death as right censored data, the above likelihood is appropriate. For the survival data with dependent competing risks, the failure density in equation (2.38) is the crude density for each item, the likelihood is the product of all crude densities [19].

2.4.2. Maximum Likelihood Estimation (MLE)

It is almost always more convenient to work with the log-likelihood, let $\ell(\theta) = \log L(\theta)$, the maximum likelihood estimates(MLE) $\hat{\theta}_1, \hat{\theta}_2, ..., \hat{\theta}_m$ (mode of $\theta_1, \theta_2, ..., \theta_m$) are those values that maximize the likelihood, or equivalently, the log-likelihood, and the MLEs may be found by solving the likelihood equations:

$$\frac{\partial \ell}{\partial \vartheta_j} = 0 \qquad (j = 1,2,....,m) \tag{2.40}$$

Usually, Gauss-Newton or Newton-Raphson methods [21] are used to solve the equations (2.40) numerically.

To estimate the standard deviation of θ_i, consider the m x m information matrix **J** with entries

$$-\frac{\partial^2 l}{\partial \theta_j \partial \theta_k} \qquad (j = 1,2,...,m; \quad k = 1,2,...,m) \tag{2.41}$$

evaluated at $\hat{\theta}$. Then the inverse of **J** is the estimated variance-covariance matrix of

$\hat{\theta}_1, \hat{\theta}_2, ..., \hat{\theta}_m$. That is, if $V = J^{-1}$ has the entries v_{jk}, then v_{jk} is the estimated covariance between ϑ_j and ϑ_k. In particular an estimate for the standard deviation of ϑ_j (j=1, 2, ... m) is just $v_{jj}^{1/2}$.

Suppose that $\hat{\theta} = (\hat{\theta}_1, \hat{\theta}_2, ..., \hat{\theta}_m)$ has been calculated. Some function of unknown parameters such as $\phi = g(\vartheta)$, may be interested, where g is a specific one-to-one function. The MLE of ϕ is defined by

$$\hat{\phi} = g(\hat{\theta}). \tag{2.42}$$

Then the standard deviation of $\hat{\phi}$ may be estimated by Delta method [19]:

$$se(\hat{\phi}) = \{\sum_{j=1}^{m}\sum_{k=1}^{m}(\frac{\partial^2 g}{\partial \vartheta_j \partial \vartheta_k})_{\hat{\vartheta}} V_{jk}\}^{1/2} \tag{2.43}$$

where the partial derivatives are evaluated at $\hat{\theta}$.

2.4.3. Maximum Likelihood Ratio

To test between nested models M_0 and M_1, where M_0 with p_0 parameters is contained within M_1 which has $p_1 > p_0$ parameters, the likelihood ratio statistic $W = 2(l_1 - l_0)$ has an approximate chi-square distribution, $\chi^2_{p_1 - p_0}$. Here l_1 and l_0 are the maximized values of l under M_1 and M_0, and $p_1 - p_0$ is the degree of freedom. The value of W is compared with chi-square x^2_{α} with df = $p_1 - p_0$, where α is the probability (for example , 0.05). If $w > x^2_{\alpha}$, reject model M_1 and select model M_0.

2.4.4. Newton-Raphson Methods

Usually, Gauss-Newton or Newton-Raphson methods [21] are used to solve the

equation (41) numerically. In this research, this method is used by the software JMP [22] to obtain the MLEs of parameters. A typical problem gives N functional relations to be zeroed, involving variables x_i , i = 1, 2, ..., N:

$$F_i(x_1, x_2, \ldots, x_N) = 0 \qquad i = 1, 2, \ldots, N \tag{2.44}$$

Let x denote the entire vector of values x_i and F denote the entire vector of function F_i. In the neighborhood of x, each of the function F_i can be expanded in a Taylor series

$$F_i(\mathbf{x} + \delta\mathbf{x}) = F_i(\mathbf{x}) + \sum_{j=1}^{N} \frac{\partial F_i}{\partial x_j} \delta x_j + O(\delta \mathbf{x}^2). \tag{2.45}$$

The matrix of partial derivatives appearing in the above equation is the *Jacobian* matrix J:

$$J_{ij} \equiv \frac{\partial F_i}{\partial x_j}. \tag{2.46}$$

In matrix notation the equation is changed to

$$\mathbf{F}(\mathbf{x} + \delta\mathbf{x}) = \mathbf{F}(\mathbf{x}) + \mathbf{J} \cdot \delta\mathbf{x} + O(\delta\mathbf{x}^2). \tag{2.47}$$

By neglecting terms of order $\delta\mathbf{x}^2$ and higher and by setting $\mathbf{F}(\mathbf{x} + \delta\mathbf{x}) = 0$, we can obtained a set of linear equations for the correction $\delta\mathbf{x}$ which move each function closer to zero simultaneously, namely

$$\mathbf{J} \cdot \delta\mathbf{x} = -\mathbf{F} \tag{2.48}$$

that is

$$\delta\mathbf{x} = -\mathbf{J}^{-1} \cdot \mathbf{F} \tag{2.49}$$

The correction are then added to the solution vector,

$$\mathbf{x}_{new} = \mathbf{x}_{old} + \delta\mathbf{x} \tag{2.50}$$

and the process is iterated to convergence.

2.5. Bayesian Approach

From the point of view of survival analysis, the adoption of Bayesian approach has number of practical advantages. Many survival problems arise in situations where there is high level of uncertainty, either because of the influence of uncertain factors which are hard to quantify, or because survival prediction involves significant extrapolation from any available data. Also, in many survival problems, the data are simply inadequate for any predictions to be made with any reasonable degree of confidence. Thus it is recognized that practical decisions are very often, inevitably, strongly guided by subjective judgments. Bayesian statistics provides a means of quantifying subjective judgments and combining them in a rigorous way with information obtained from experimental data [23]. In this research Bayesian statistics is used to estimate the parameters in a competing risk model.

2.5.1. Bayes' Theorem:

If X is a continuous random variable, whose probability density depends on an unknown parametric vector variable θ, is given by $f(x|\theta)$, and if the prior probability density of θ is denoted by $P_0(\theta)$, then for every x such that $f(x|\theta) > 0$ exists, the posterior probability density of θ, given $X = x$ is [18]

$$P(\vartheta|x) = \frac{f(x|\vartheta)P_0(\vartheta)}{f(x)} \tag{2.51}$$

where

$$f(x) = \int f(x|\vartheta)P_0(\vartheta)d\vartheta \tag{2.52}$$

is the marginal probability density of X.

Suppose that experimental results $X_1, X_2, \ldots, X_n$ are available and that we desire to estimate the unknown parameter vector θ, the f(x|θ) term in the Bayes' theorem is replaced by Likelihood function L(X|θ) [23, 24, 25, 26] as follows:

$$P(\vartheta \mid X) = \frac{L(X \mid \vartheta) P_0(\vartheta)}{\int L(X \mid \vartheta) P_0(\vartheta) d\vartheta} \tag{2.53}$$

Since the θ is usually the hypothesis to be evaluated relative to empirical data x, this form of Bayes' theorem thus states that the probability of the hypothesis conditional on the data (or the posterior probability of the hypothesis) is equal to the probability of the data conditional on the hypothesis (or the likelihood of the hypothesis) times the probability (the so-called prior probability) of the hypothesis, all divided by the probability of the data [35].

Since the denominator in equation (54) is not a function of parameter θ , Bayes' theorem can be written as

$$P(\vartheta \mid X) \quad \propto \quad L(X \mid \vartheta) \cdot P_0(\vartheta) \tag{2.54}$$

The proportional constant is the normalization factor that assures that the integral of posterior density of θ over θ is unity.

Bayes' theorem incorporates the current observations and modifies prior knowledge of θ [25, 27]. The prior density $P_0(6)$ expresses the decision maker's degree of belief about the parameters, prior to the observing the data, based on previous practical experience and understanding. The posterior distribution $P(6 \mid X)$ is the conditional

distribution of θ after the data have been incorporated in the analysis. It contains all the available information about the parameters and some Bayesians assert that inference about any parameter is complete whenever its entire posterior is reported. Consequently, quantities of interest such as point or interval estimates can be readily derived from this distribution.

2.5.2. Bayesian Estimation of Parameters

In the above equation, if θ is n dimensional parameter vector : $\theta = (\theta_1, \theta_2, \ldots \ldots, \theta_n)$, the posterior distribution refers to the joint posterior for all of components of θ. A complete inference for the interested parameter, θ_i is described by the marginal posterior density of θ_i as follows :

$$P(\vartheta_i | X) = \int_{\vartheta_R} P(\vartheta | X) d\vartheta_R \tag{2.55}$$

where R is the complement of i in the set (1, 2, ... , n). The multiple integration can be performed analytically or numerically depending on the forms of the likelihood and the prior densities.

2.5.3. Bayesian Prediction

Bayesian prediction technique is used to predict the predictive distribution of X [Guttman and Tiao (1964)], which is the posterior distribution of a future observation X^* given the current sample statistics. Generally, let $Q(X, \theta)$ be a probability function of X depending on the parameter θ, such as failure density, hazard rate, survival function, ..., etc. Using the posterior density of θ obtained by a current sample data, the prediction for $Q(X^*)$ for a new observation X^* is [19]:

$$Q(X^*) = \int_{\vartheta} Q(X^*, \vartheta) P(\vartheta | \mathbf{X}) d\vartheta \qquad (2.56)$$

The predictive density is based on considering the uncertainty from all parameters [28]. It is independent of the parameters.

2.5.4. Noninformative Priors

Noninformative priors are used for random quantities whenever little prior information is available. A noninformative prior distribution of the parameter θ expresses little preference for some values of the parameter over others. Noninformative priors provide some degree of objectivity in Bayesian analysis. They are sometimes used whenever results from Bayesian and conventional analysis are to be compared. In this research, constant priors are used.

2.5.5. The Bayes Factor

One of Bayesian model choice criteria is the posterior odds ratio [29, 30] . If there are two possible models, M_1 and M_2, with different parameters θ_1 and θ_2 for a given data set X, and each with prior density $P(\theta_1)$ and $P(\theta_2)$, then the posterior odds on model 1 is

$$\frac{P(\vartheta_1 | X)}{P(\vartheta_2 | X)} = \frac{q_1}{q_2} \frac{P_0(\vartheta_1)}{P_0(\vartheta_2)} \qquad (2.57)$$

The Bayes factor is defined by

$$BF = \frac{q_1}{q_2} = \frac{\int_{\vartheta \subset M_1} L_1(\vartheta_1 | X) P_0(\vartheta_1) d\vartheta_1}{\int_{\vartheta \subset M_2} L_2(\vartheta_2 | X) P_0(\vartheta_2) d\vartheta_2} \qquad (2.58)$$

where the term

$$\int_{\vec{\vartheta} \subset M_i} L_i(\vec{\vartheta}_i \mid X) P_0(\vec{\vartheta}_i) d\vec{\vartheta}_i \qquad (2.59)$$

is the marginal probability of the data X [29] from model i.

The Bayes factor depending on the prior density provides the sample 'weight of evidence' for model 1 over model 2. If BF is about 1, we can say there is no big difference between model 1 and model 2. For multiple model selection, the one with the largest $P(\theta_i \mid X)$ is considered as the model that most likely to represent data X.

2.6. Numerical Integration

In Bayesian estimation of parameters, it is necessary to integrate the joint posterior density over one or more parameters to obtain the marginal posterior density for the interested parameter. However, for most problems, the integration will not be feasible analytically. Numerical integration techniques are required for the implementation of the Bayesian approach. In this research, three different numerical methods are employed and compared to verify the results. They are the Monte Carlo method, Gauss-Legendre quadrature and Lattice integration.

2.6.1. Gauss - Legendre Quadrature

The idea of Gauss quadrature [21, 31], is to approximate the value of an integration as a linear combination of values of the integrand evaluated at specific points, so called nodes x_i, multiplied by certain aptly chosen weighting coefficients w_i, i = 1, 2, ..., n. Thus

$$\int_a^b W(x) f(x) dx \cong \sum_{i=1}^{n} w_i f(x_i) \quad . \qquad (2.60)$$

where $W(x)$ is the weighting function.

It is shown the set of $(x_1, x_2, \ldots, x_n)$ is the n distinct roots of the orthogonal polynomial $P_n(x)$, which has the recurrence relation

$$\begin{aligned} &P_{-1}(x) = 0 \\ &P_0(x) = 1 \\ &P_{j+1} = (x - a_j)\, P_j(x) - b_j\, P_{j-1}(x) \qquad j = 1,2,\cdots \end{aligned} \tag{2.61}$$

where

$$\begin{aligned} a_j &= \frac{\langle x P_j \mid P_j \rangle}{\langle P_j \mid P_j \rangle} \qquad j = 0,1,\cdots \\ b_j &= \frac{\langle P_j \mid P_j \rangle}{\langle P_{j-1} \mid P_{j-1} \rangle} \qquad j = 1,2,\cdots \end{aligned} \tag{2.62}$$

the $< >$ is Dirac symbol, it is used to describe the scalar product of two functions over the weight function W(x) as

$$\langle f \mid g \rangle = \int_a^b W(x)\, f(x)\, g(x)\, dx \tag{2.63}$$

Two functions are said to be orthogonal if their scalar product is zero. A function is said to be normalized if its scalar product with itself is unity. A set of functions that all mutually orthogonal and also all individually normalized is called an orthogonal set. The weights can be estimated by the formula

$$w_i = \frac{\langle P_{N-1} \mid P_{N-1} \rangle}{P_{N-1}(x_j)\, P'_N(x_j)} \tag{2.64}$$

where the $P'_N(x_j)$ is the derivative of the orthogonal polynomial at its zero x_j.

If the weight function, intervals, and recurrence relations that generate the most commonly used orthogonal polynomials are respectively as

$$W(x) = 1 \qquad -1 \prec x \prec 1$$
$$(j+1)\,P_{j+1} = (2j+1)\,x\,P_j - j\,P_{j-1} \tag{2.65}$$

the corresponding weights are

$$w_j = \frac{2}{(1-x_j^{\,2})\,[P'_N(x_j)]^2} \tag{2.66}$$

and the Gaussian quadrature formula is

$$\int_a^b f(x)\,dx = \sum_1^N w_j\, f(x_j) \tag{2.67}$$

for multi- dimensional integrals, for example, for dimension k, it is

$$\int_{a_1}^{b_1}\cdots\int_{a_k}^{b_k} f(x_1, x_2, \cdots, x_k)\,dx_1\,dx_2\cdots dx_k \approx$$
$$\approx \sum_{j_1}^{n_1}\sum_{j_2}^{n_2}\cdots\sum_{j_k}^{n_k} w_{j_1}\,w_{j_2}\cdots w_{j_k}\,f(x_{j_1}, x_{j_2}, \cdots, x_{j_k}) \quad . \tag{2.68}$$

If $n_1 = n_2 = \ldots = n_k$, one has to evaluate the integrand at n^k points.

2.6.2. Lattice Integration

The lattice rules [31, 32] are quadrature rules designed to approximate the integral of function $f(x_1, x_2, \ldots, x_s)$ over the unit s-dimensional cube, in situations in which f is reasonably smooth and also one-periodic with respect to each component of x . This is an equal-weight rule whose quadrature points are all the points of an integration lattice that lie in the half-open unit cube. Some preliminary transform of the given integral may be needed to bring the problem into this form.

The embedded lattice rule is given by

$$
\begin{aligned}
If &= \int_{C^n} f(x_1,\cdots,x_n) = \int_0^1 \cdots \int_0^1 f(x_1,\cdots,x_n)\,dx_1 \cdots dx_n \\
&\approx Q_s f \\
&= \frac{1}{n^s m} \sum_{k_s=0}^{n-1} \cdots \sum_{k_1=0}^{n-1} \sum_{j=0}^{m-1} f\left(\left\{\frac{j}{nm}\mathbf{z} + \frac{(k_1,\cdots,k_s)}{n}\right\}\right).
\end{aligned}
\tag{2.69}
$$

where $C^S = [0, 1]^S$, n and m are relatively prime, and z is an integer vector having no nontrivial factor in common with m. Q_S is the lattice rules having $(n^S\ m)$ points.

A strong case can be made, that the best value of n is 2 [31]. Then the m must be odd if n =2, because of the constraint that m is relatively prime with n. For example, if n = 2, m = 5, z = (1, 2) and s = 2, the above formula is a sum of twenty terms, in which each term is the value of integrand f at one of twenty lattice points , divided by 20. For non periodic integrands, Niederreiter and Sloan (1993) proposed the following lattice rule [31]

$$
Bf := \sum \cdots \sum W_{i1,\ldots,is} f(i_1,\ldots,i_s) + \frac{1}{N}\sum_{j=0}^{N-1} f\left(\left\{\frac{j}{N}z(\ell)\right\}\right) \tag{2.70}
$$

They gave tables of Z obtained by finding a value of ℓ in

$$
\begin{aligned}
z &= (1,\ \ell,\ \ell^2 \bmod N,\ \cdots,\ \ell^{s-1} \bmod N), \\
&1 \le \ell \le \lfloor N/2 \rfloor,
\end{aligned}
\tag{2.71}
$$

such that the vertex variance

$$
v_B = \sum_{i_s=0}^{1} \cdots \sum_{i_1=0}^{1} W^2_{i_1 \cdots i_s} \tag{2.72}
$$

is minimized. Here $W_{i1 \ldots is}$, for $i_1, \ldots, i_s = 0$ or 1, are the weights in the 'optimal' quadrature rule equation (2.70). In this research, we use the Table 2.1 to

select the value of L (L = ℓ) for s = 6 [31]:

Table 2.1 Recommended choices of L for s = 6

N	79	157	313	619	1249	2503	5003	10007
L	15	22	59	132	143	347	708	357

2.6.3. Monte Carlo Method

In the classical Monte Carlo method [21, 31, 33] (Metropolis and Ulam 1949, Hammersley and Handscomb 1964) points $x_0, \ldots, x_{N-1}$ are chosen randomly in $[0, 1]^s$ and the integral

$$I f = \int_{C^n} f(x_1, \cdots, x_n) = \int_0^1 \cdots \int_0^1 f(x_1, \cdots, x_n)\, d x_1 \cdots d x_n \tag{2.73}$$

is then estimated by

$$Q_N f = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) \tag{2.74}$$

where x_j is the j 'th s-dimensional random vector .

Like the whole subject of gambling, the Monte Carlo method invites strong passions [21]. Its virtues are considerable. Convergence is guaranteed (almost surely) by the central limit theorem, even if f is merely continuous. Moreover, the rate of convergence is independent of the dimension: the error $| Q_N f - I f |$ is estimated by

$$\frac{\sigma(f)}{\sqrt{N}} \tag{2.75}$$

where

$$\sigma^2(f) = \int_{C^s} f^2(x)\, d x - (I f)^2 \tag{2.76}$$

is the variance of f. Finally, and most importantly, a free estimate of the error is available: the variance may be estimated by

$$Q_N f^2 - (Q_N f)^2 \quad , \tag{2.77}$$

and its square root then used to estimate the error (standard deviation).

Monte Carlo methods are sometimes the most convenient for evaluating certain integrals, particularly when they contain several independent variables (multiple integration) or involve functions that are not well behaved [36].

The negative aspects of the Monte Carlo method begin with the fact that truly random number sequences are not available: rather, one must make do with 'pseudo-random' sequences generated by a deterministic algorithm [31]. Of course, no deterministic sequence can be truly random, but it is possible to find 'random number generators' whose output appears random when subjected to simple statistical tests. Implementations of the Monte Carlo method often include some sort of "variance reduction" technique. Such techniques include stratified sampling, importance sampling, and correlated sampling[33].

2.6.4. Comparison of the Monte Carlo and Lattice Method

Let us calculate the six dimensional integral:

$$\begin{aligned} Q &= \int_0^1 \cdots \int_0^1 F(a,b,c,d,e,f)\, da\, db\, dc\, dd\, de\, df \\ &= \int_0^1 \cdots \int_0^1 a*b*c*d*e*f\, da\, db\, dc\, dd\, de\, df \end{aligned} \tag{2.78}$$

The exact value of this integration is 0.015625. Using Lattice rule and Monte Carlo method, the results are compared in Table 2.2.

Table 2.2 Comparison of Numerical Integration Methods.

	Q
Exact Value	0.015625
Monte Carlo (N = 10007)	0.01561469
Lattice Method (L = 357, N = 10007)	0.01557420

CHAPTER 3. SURVIVAL DATA ANALYSIS

In this chapter, the primary work in the research is described, the results are reported and analyzed.

3.1. Experimental Data Preprocessing

3.1.1. Experimental Data Summary

The experimental data described in section 1.1 are summarized in Table 3.1. The table 3.1 lists the mean activity, mean time since exposure and the osteosarcoma incidences in each injection level for Pu_Data and Ra_Data (see APPENDIX B).

For the comparison of the toxicity between plutonium and radium, the experimental data are displayed in Fig 3.1 to Fig 3.3. Fig 3.1 shows the mean activity in each injection level for ^{239}Pu as compared with ^{226}Ra. However, Fig 3.2 shows the much shorter mean lifetime since exposure to ^{239}Pu as compared with ^{226}Ra. Furthermore, Fig 3.3 shows the much greater osteosarcoma incidence in each injection level for Pu_Data as compared with Ra_Data. These plots indicate that the radionuclide ^{239}Pu is more dangerous than ^{226}Ra in radiation-induced bone cancer, osteosarcoma.

3.1.2. Scaling of the Survival Terms

In working with statistics we frequently work with transformations of our data rather than the actual data [18]. Some common transformations are "square roots", "logarithm" and "linear". The purpose of these transformations are to stabilize the variance, to provide better fitting distributions, and to calculate distributions of statistics. A common transformation of data is the linear one defined by

$$Y = A + BX . \tag{3.1}$$

Direct application of the definitions of expected value and variance gives

Table 3.1 Summary of the Experimental Data

injection level	mean(activity)	mean(time)	Osteosarcoma(%)
	[μC_i]	[day]	
	Pu_Data		
0	0	4204.3	1.85
0.1	0.0075	4131.07	0
0.2	0.019	4101.41	2.17
0.5	0.0567826	3398.72	6.52
0.7	0.1073158	4000.63	15.79
1	0.131675	3756.75	20
1.7	0.3901143	3076.86	31.43
2	0.852	2495.75	83.33
3	2.59	1649.42	100
4	7.8858	1319.67	100
5	25.213778	1338	77.78
	Ra_Data		
0	0	4202.76	0
0.1			
0.2	0.0785	3990.6	0
0.5	0.22808	4407.16	4
0.7			
1	0.6233478	3927.22	4.35
1.7	1.53023	3655.85	15.38
2	3.2111667	3948.08	41.67
3	10.8775	2234.92	91.67
4	32.9671562	1353.81	71.88
5	95.6029231	946.6923077	76.92

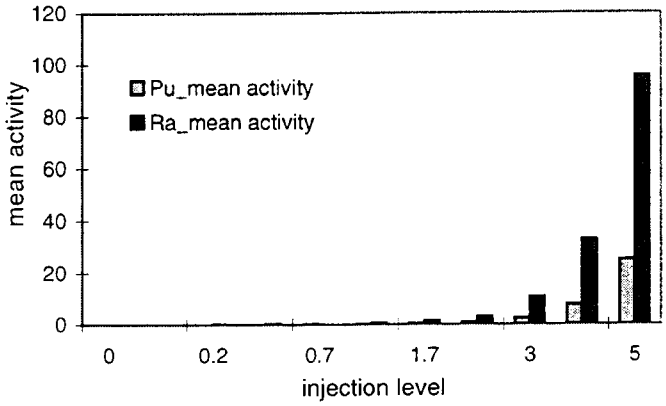

Figure 3.1 Comparison of mean activities

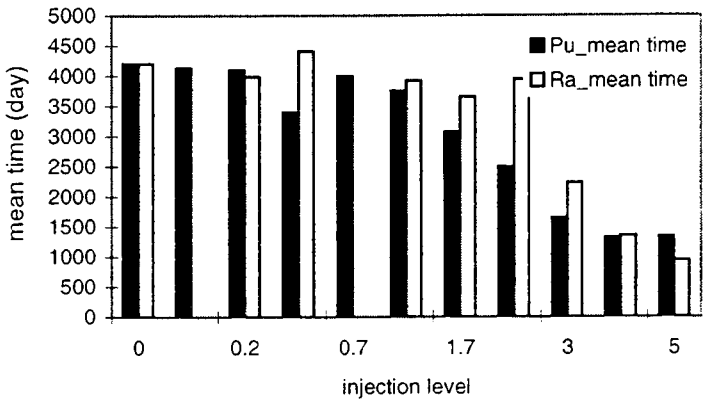

Figure 3.2 Comparison of mean lifetimes.

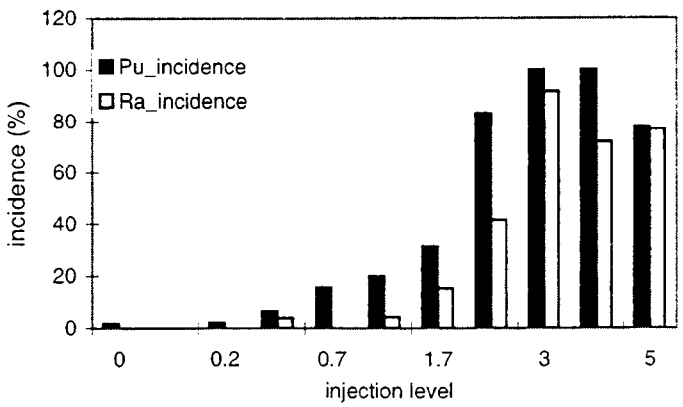

Figure 3.3 Comparison of osteosarcoma incidences

$$E(Y) = A + BE(X) \tag{3.2}$$

$$\text{Var}(Y) = B^2 \text{Var}(X) \tag{3.3}$$

Thus the mean of linear function of X is the same linear function of the mean of X. However, the variance of X is unchanged by the location factor A but changed by the square of the scale factor B.

The observed lifetimes in the research work are a few thousand days. They cause overflow problems in computations with the Weibull distribution described in section 2.2.2. To solve this problem, the experimental data are linearly transformed. Although the transformation changes the scale of data, the relative relationship among the data is remained. The lifetimes are transformed as follows:

1. Define the up limit and the lower limit for the scaled data as y_{max} and y_{min} ;
2. Take the longest observed lifetime as x_{max} , and the shortest as x_{min} ;
3. Define the linear transformation as:

$$y = a\,x + b \tag{3.4}$$

4. Determine the slope a and the intercept by using y_{max} , y_{min} , x_{max} and x_{min}

$$y_{max} = a\,x_{max} + b \tag{3.5}$$

$$y_{min} = a\,x_{min} + b \tag{3.6}$$

5. Linear transform all of lifetime using the determined slope and intercept.

The same method is applied to linearly transform the accepted activities in the data.

3.2. Model Identification and Selection of Life Distributions

3.2.1. Nonparametric Diagnostic Plots

Nonparametric diagnostic plot is one convenient way to select an appropriate life distribution model for the survival data. Fig 3.4 is the Nelson cumulative hazard plot for Pu_data, and Fig 3.5 gives the log cumulative hazard for different injection level at which

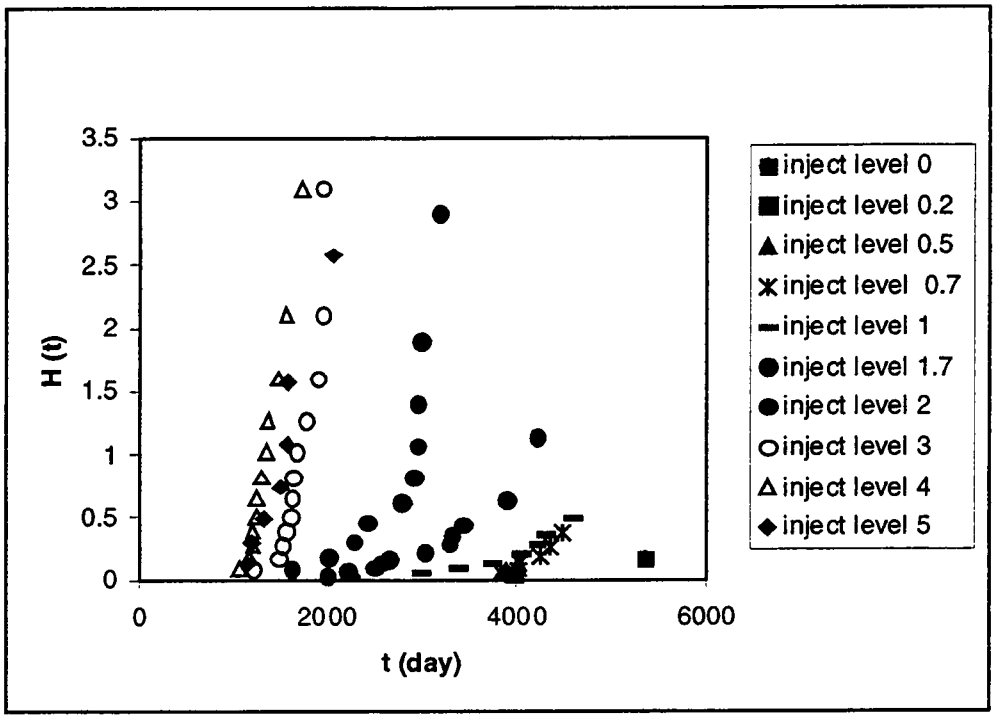

Figure 3.4 Cumulative Hazards for Pu –Data.

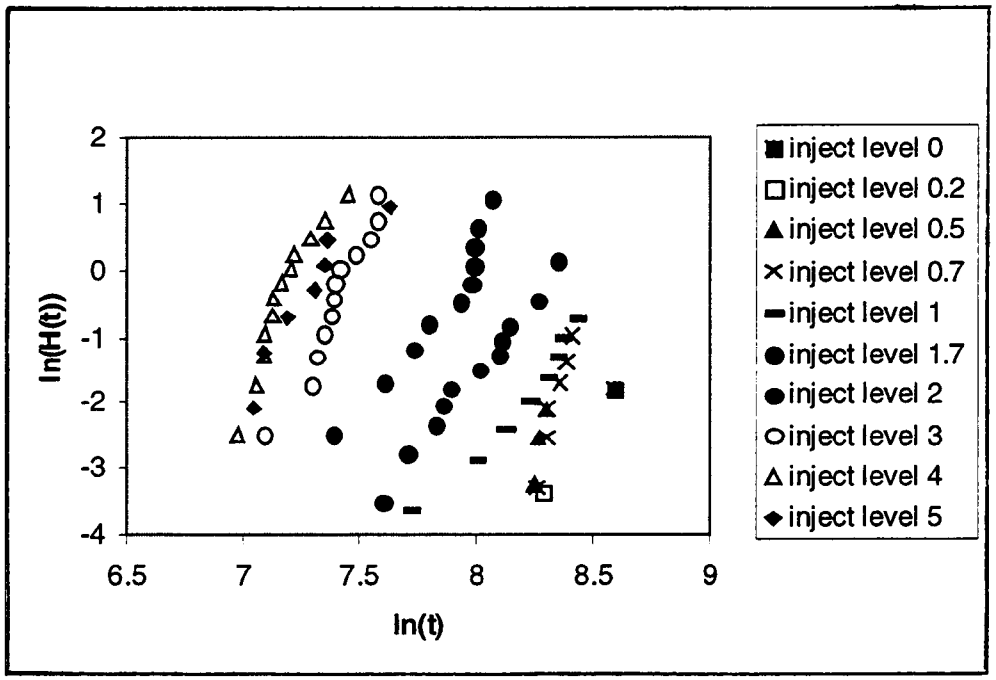

Figure 3.5 Log cumulative Hazards for Pu-Data

dogs were exposed to Plutonium. Fig 3.6 and Fig 3.7 are the plots for Ra-data.

These plots provide some information about the dependence of the hazard rate on time and activity. (i). The curves describing cumulative hazard versus time in Fig 3.4 and Fig 3.6 appear as convex other than straight lines, and the plots of $\ln(H(t_I))$ against $\ln(t_i)$ Fig 3.5 and Fig 3.7, are roughly linear. According to the equation (2.13) and (2.14), a Weibull model is chosen to describe the dependence of the hazard function on time for the survival data used in this research. (ii). Comparing the cumulative hazard for different injection levels in these plots, we can see that the slope increases as the injection level increases, which means the hazard rate increases as the injected activity increases. This feature of the activity dependence of hazard rate is then added to the life distribution model, and these plots are used to select an appropriate relative risk model describing the activity dependence. (iii). The Fig 3.5, consisting of vertically shifted copies of the same curve indicates a proportional hazards model, and one consisting of roughly parallel straight lines implies a Weibull model as in equation (2.14). The same suggestion comes from Fig 3.7. Therefore, a proportional hazard model with baseline of Weibull distribution (relative risk model) is appropriate for the survival data. The survival function is chosen as

$$\begin{aligned} S(t,\eta,\xi) &= \exp(-(\frac{t}{\alpha})^{\eta}) \\ &= \exp(-t^{\eta}\, e^{\xi}) \end{aligned} \tag{3.7}$$

where the t is the time since exposure, parameter η, ξ have the relationship

$$\xi = -\eta \ln(\alpha) \tag{3.8}$$

The cumulative hazard of the model is

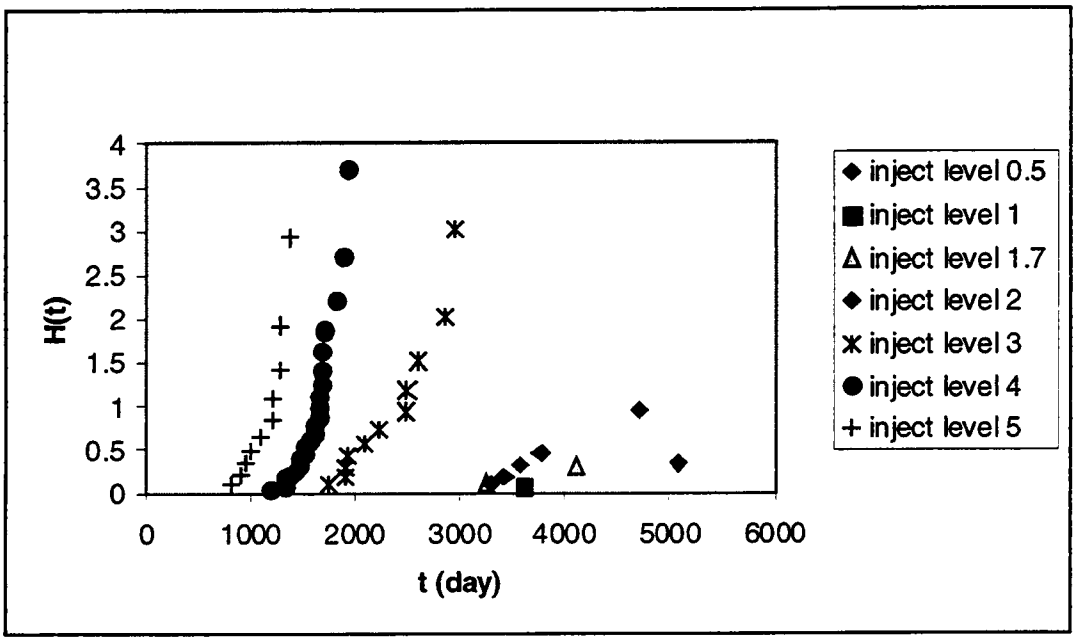

Figure 3.6 Cumulative Hazards for Ra-Data

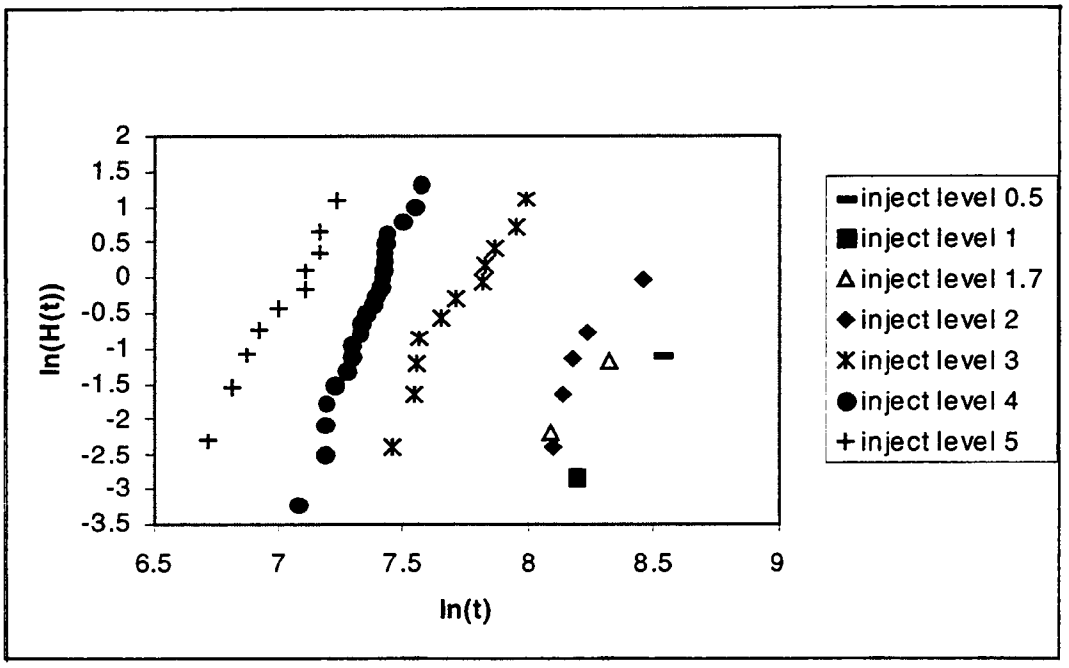

Figure 3.7 Log Cumulative Hazards for Ra-Data.

$$H(t,\eta,\xi) = t^{\eta} e^{\xi} \tag{3.9}$$

The relationship between log cumulative hazard and log time is

$$\ln(H(t,\eta,\xi)) = \eta \ln(t) + \xi \tag{3.10}$$

the parameter η is simply the slope and ξ is intercept of the log cumulative hazard plot. They are analyzed in the following section.

Nonparametric analysis also indicates the effects of competing risks. The effect of competing risks is shown in Fig 3.8 for the injection lever 2 of Pu_Data. The survival curve with two competing risks is displayed by the curve with legend "combined". The survival curve with only one cause of death, osteosarcoma or non-osteosarcoma, are shown respectively by the curves with legend "omitted 2" and "omitted 1". It is very clear that the values of survival function representing two causes of death are smaller than that omitting any one cause of death, which is just the result of competing risks. This point is reflected in the construction of the likelihood function.

3.2.2. Exposure Models

As what mentioned before, the parameters η, ξ are related to the injected activity. To establish the function relationship $\eta(d)$, $\xi(d)$ (d represents the activity), we fit the straight lines in Fig 3.5 and Fig 3.7. The regression equations for different injection level are obtained in Table 3.2 for Pu_data and Ra_data respectively.

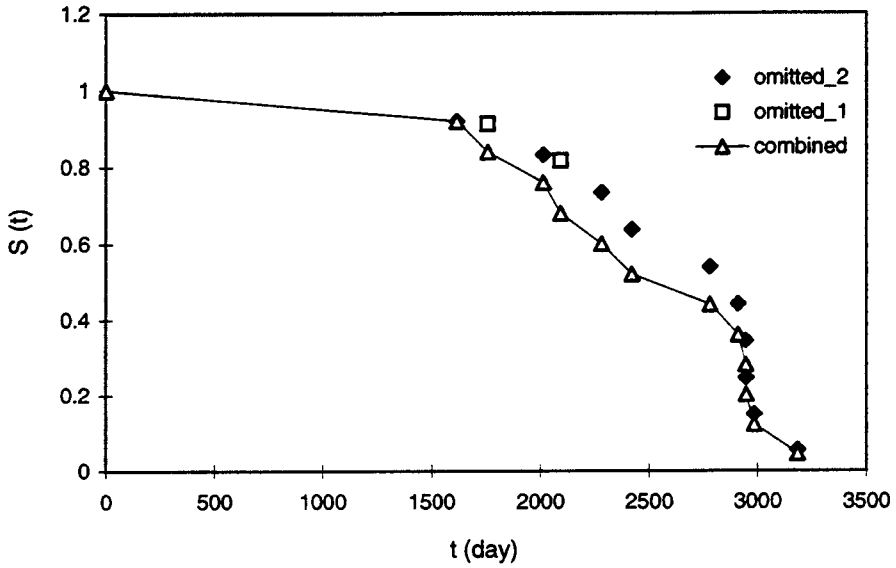

Fig 3.8. Comparison of survival probability for the data with competing risks.

Table 3.2 Regression Equations for Cumulative Hazard

Injection Level	Linear regression Pu-Data	Linear regression Ra-Data
1	Ln(H(t)) = -35.94+4.153*ln(t)	
1.7	Ln(H(t)) = -37.74+4.527*ln(t)	
2	Ln(H(t)) = -38.81+4.886*ln(t)	Ln(H(t))= - 44.05+5.649*ln(t)
3	Ln(H(t)) = -55.75+7.475*ln(t)	Ln(H(t))= - 49.28+5.847*ln(t)
4	Ln(H(t)) = -54.07+7.474*ln(t)	Ln(H(t))= - 68.74+9.281*ln(t)
5		Ln(H(T)) = -42.27+5.972*ln(t)

Since both the variety of slopes and that of intercepts come from different activity, taking the average activity for each injection level, a simple polynomial fit gives the relationship between slope and activity and that between intercept and activity. The plots of slope versus average activity and the intercept versus activity are developed in Fig 3.9 for Pu_data, Fig 3.10 for Ra_data.

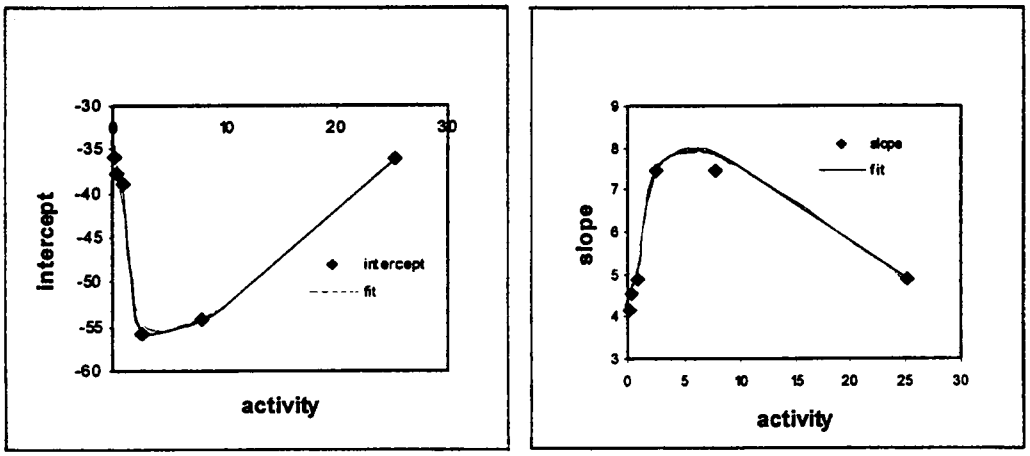

Figure 3.9. Intercept and slope vs. activity for Pu_Data

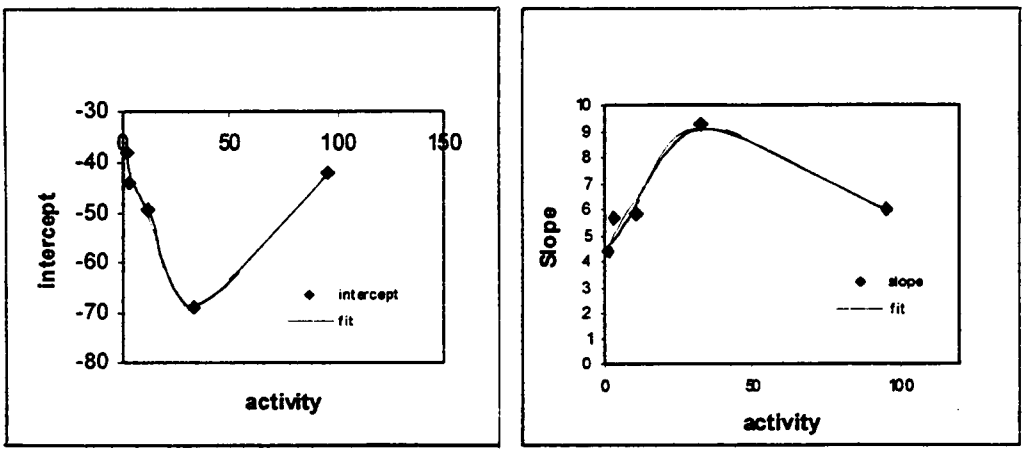

Figure 3.10 Intercept and slope vs. activity for Ra_Data

So the initial estimates of parameters in the model for Pu_data is

$$\begin{aligned}\eta &= b_1 + b_2 d + b_3 d^2 = 3.77 + 1.833d - 0.1728d^2 \\ \xi &= b_4 + b_5 d + b_6 d^2 = -33.06 - 11.12d + 1.071d^2\end{aligned} \tag{3.11}$$

and for Ra_data is

$$\begin{aligned}\eta &= b_1 + b_2 d + b_3 d^2 = 4.348 + 0.2113d - 0.00203d^2 \\ \xi &= b_4 + b_5 d + b_6 d^2 = -37.16 - 1.413 d + 0.01422 d^2\end{aligned} \tag{3.12}$$

The values of b_1 , ..., b_6 are taken as the initial values for maximum likelihood estimation with the Newton-Raphson method.

3.2.3. Construction of Likelihood Function

The Pu_data and Ra_data are taken as two independent sets with independent competing risks. Since we are only interested in one cause of death, osteosarcoma, the lifetimes with other causes of death are considered as right censored data. According to equation (2.39), the likelihood for Pu_Data and for Ra_Data are

$$\begin{aligned}L_{Pu}(\vartheta \mid t, d) &= \{\prod_{i \subset u} f_{p1}(t_i, d_i, \vartheta)\}\{\prod_{i \subset c} S_{p1}(t_i, d_i, \vartheta)\} \\ L_{Ra}(\vartheta \mid t, d) &= \{\prod_{i \subset u} f_{Ra1}(t_i, d_i, \vartheta)\}\{\prod_{i \subset c} S_{Ra1}(t_i, d_i, \vartheta)\}\end{aligned} \tag{3.13}$$

where u represents the failure data with the cause of osteosarcoma, c represents the data with the cause of other diseases, and ϑ represents parameter set: b_1 , b_2 , ... , b_6 .

Using the relationship in equation (3.8)

$$\xi = -\eta \ln(\alpha),$$

the survival function $S(t_j, d_i, \vartheta)$ and the failure density $f(t_i, d_i, \vartheta)$ in equation (3.13) are expressed as

$$S(t_i, d_i, \vartheta) = \exp(-t^{\eta} e^{\xi}) = \exp(-t^{\eta} / \alpha^{\eta})$$
$$f(t_i, d_i, \vartheta_i) = \eta\, t^{\eta-1} e^{\xi} \exp(t^{\eta} e^{\xi}) = \eta\, t^{\eta-1} \frac{1}{\alpha^{\eta}} \exp(t^{\eta} / \alpha^{\eta}) \qquad (3.14)$$

The relationship between parameter η and activity, and ξ and activity are expressed in equation (3.11) and (3.12) for Pu_data and Ra_data respectively.

3.2.4. Maximum Likelihood Estimation

The software JMP [22] is a good tool for survival analysis. To get the MLEs to the Weibull fits of the survival data, at first, transform the Weibull model with parameter η and α as Gumbell model (Extreme - value) [19], the survival function and failure density in Gumbell model are

$$S(x) = \exp\{-\exp[(x-\mu)/\sigma]\}$$
$$f(x) = \frac{1}{\sigma} \exp\{(x-\mu)/\sigma\}\, S(x) \qquad (3.15)$$

where $x = \ln(t)$, $\mu = \ln(\alpha)$ and $\sigma = 1/\eta$.

Let $c = 1$ for the failure data with the cause of other diseases

$c = 0$ for the failure data with the cause of osteosarcoma,

the likelihood function is rewritten as

$$L(\vartheta) = \prod_{i \subset (n+m)} \{\{f(t_i, \vartheta)\}^{(1-c)} \{S(t_i, \vartheta)\}^{c}\} \qquad (3.16)$$

Using equation (3.15), the - ln (L (θ)) is

$$-\ln(L(\vartheta)) = \sum_{i=1}^{n+m} \{(1-c)\{\ln\sigma + \frac{\mu - x_i}{\sigma}\} + \exp\{\frac{x_i - \mu}{\sigma}\}\} \qquad (3.17)$$

Input the -ln (L (θ)) and use the command " Estimate Weibull Parameters ", JMP gives the MLE of parameters μ , σ , η and α by Newton-Raphson method. Here we input the

-ln (L (θ)) with the six parameters b_1 , ... , b_6. The results are shown in Table 3.3 and Table 3.4 for Pu_data and Ra_data respectively.

Therefore, the slope and the intercept for the Pu-data are

$$\begin{aligned} \eta &= 3.798 + 3.705d - 0.4153d^2 \\ \xi &= -33.845 - 24.935\,d + 2.869\,d^2 \end{aligned} \tag{3.18}$$

and for the Ra-data they are

$$\begin{aligned} \eta &= 2.55 + 0.36\,d - 0.00325\,d^2 \\ \xi &= -23.38 - 2.497\,d + 0.023\,d^2 \end{aligned} \tag{3.19}$$

3.2.5. Maximum Likelihood Ratio

For further simplification, the above model is changed to a regression model:

$$\begin{aligned} \eta &= b_1 \\ \xi &= b_2 + b_3\,d \end{aligned} \tag{3.20}$$

In the same way as that in section 3.2.4, the initial values of these parameters are obtained in Table 3.5 and Table 3.6 for Pu_data and Ra_data respectively.

Let the simplified model as M_2 with three parameters, the original model as M_1 with six parameters, using the maximum likelihood ratio method, the chi-square test is shown in Table 3.7.

Since both w's for Pu_Data and Ra_Data are greater than $x^2_{df=3,\,\alpha=0.05}$, the result suggests model 2, the relative risk model with three parameters. In the following section, the model 2 is used to compare toxicity between ^{239}Pu and ^{226}Ra.

The selected model can be easily transformed to the standard relative risk model with a baseline given by a Weibull distribution. Let ρ represents the toxicity ratio between ^{239}Pu and ^{226}Ra, d represents the injected activity, then the hazard rate, the survival function, the failure density and the cumulative hazard for this model are expressed as follows:

Table 3.3 JMP Maximum Likelihood Estimate of Six Parameters for Pu-Data

Parameters	Estimate	ApproxStdErr	Lower CL	Upper CL
b1	3.7977915173	0.82944638	2.25814542	5.50201341
b2	3.7050170068	0.99448948	1.80939471	5.70746825
b3	-0.415322471	0.12397079	-0.6647668	-0.1788206
b4	-33.84451463	6.83959419	-47.965863	-21.216774
b5	-24.93477231	7.55138236	-40.113973	-10.508078
b6	2.8689112506	0.93509198	1.08387836	4.74694483

Table 3.4 JMP Maximum Likelihood Estimate of Six Parameters for Ra-Data

Parameter	Estimate	ApproxStdErr	Lower CL	Upper CL
b1	2.5498940671	0.87523782	0.96218993	4.37807641
b2	0.3598085708	0.07161022	0.22180923	0.50299436
b3	-0.003247967	0.00075214	-0.0047439	-0.0017882
b4	-23.37926207	7.21873001	-38.552287	-10.377218
b5	-2.49728632	0.54289328	-3.5487606	-1.4457512
b6	0.023049826	0.00561554	0.0121447	0.03421231

Table 3.5 JMP Maximum Likelihood Estimate of Parameters for Pu-Data

Parameter	Estimate	ApproxStdErr	Lower CL	Upper CL
b1	3.7455308622	0.41657182	2.97429738	4.6062852
b2	-31.72683393	3.461282	-38.892677	-25.330605
b3	0.636081249	0.06883016	0.50407942	0.77472179

Table 3.6 JMP Maximum Likelihood Estimate of Parameters for Ra-Data

Parameter	Estimate	ApproxStdErr	Lower CL	Upper CL
b1	3.1302774144	0.33481574	2..50976343	3.82269816
b2	-26.0714099	2.78123187	-31.83985	-20.931441
b3	0.0470378173	0.00559913	0.0362221	0.05826415

Table 3.7. Maximum Likelihood Ratio

	M1 6 parameters	M2 3 parameters	w	Chi Square Test df = 3; α = 0.05
log-likelihood Pu-Data	419	459	80	7.81
log-likelihood Ra-Data	370	422	104	7.81

for Pu_Data and osteosarcomas:

$$h_{p1}(t,d,\alpha_1,\lambda_1,\beta,\rho) = \alpha_1\lambda_1(\lambda_1 t)^{\alpha_1-1}\exp(\rho\beta d)$$

$$H_{p1}(t,d,\alpha_1,\lambda_1,\beta,\rho) = (\lambda_1 t)^{\alpha_1}\exp(\rho\beta d)$$

$$S_{p1}(t) = \exp\{-(\lambda_1 t)^{\alpha_1}\exp(\rho\beta d)\}, \qquad (3.21)$$

$$f_{p1}(t) = \alpha_1\lambda_1(\lambda_1 t)^{\alpha_1-1}\exp(\rho\beta d)\exp\{-(\lambda_1 t)^{\alpha_1}\exp(\rho\beta d)\}.$$

for Pu_Data and another cause of death:

$$h_{p2}(t,d,\alpha_2,\lambda_2) = \alpha_2\lambda_2(\lambda_2 t)^{\alpha_2-1}$$

$$H_{p2}(t,d,\alpha_2,\lambda_2) = (\lambda_2 t)^{\alpha_2}$$

$$S_{p2}(t) = \exp\{-(\lambda_2 t)^{\alpha_2}\}, \qquad (3.22)$$

$$f_{p2}(t) = \alpha_2\lambda_2(\lambda_2 t)^{\alpha_2-1}\exp\{-(\lambda_2 t)^{\alpha_2}\}.$$

for Ra_Data and osteosarcoma as the cause of death:

$$h_{R1}(t,d,\alpha_1,\lambda_1,\beta) = \alpha_1\lambda_1(\lambda_1 t)^{\alpha_1-1}\exp(\beta d)$$

$$H_{R1}(t,d,\alpha_1,\lambda_1,\beta) = (\lambda_1 t)^{\alpha_1}\exp(\beta d)$$

$$S_{R1}(t) = \exp\{-(\lambda_1 t)^{\alpha_1}\exp(\beta d)\}, \qquad (3.23)$$

$$f_{R1}(t) = \alpha_1\lambda_1(\lambda_1 t)^{\alpha_1-1}\exp(\beta d)\exp\{-(\lambda_1 t)^{\alpha_1}\exp(\beta d)\}.$$

for Ra_Data and another cause of death:

$$
\begin{aligned}
&h_{R2}(t,d,\alpha_2,\lambda_2) = \alpha_2 \lambda_2 (\lambda_2 t)^{\alpha_2 - 1} \\
&H_{R2}(t,d,\alpha_2,\lambda_2) = (\lambda_2 t)^{\alpha_2} \\
&S_{R2}(t) = \exp\{-(\lambda_2 t)^{\alpha_2}\}, \\
&f_{R2}(t) = \alpha_2 \lambda_2 (\lambda_2 t)^{\alpha_2 - 1} \exp\{-(\lambda_2 t)^{\alpha_2}\}.
\end{aligned}
\tag{3.24}
$$

The estimation of the toxicity ratio ρ and the other parameters are discussed in the following section.

3.3. *Crude Density of FGM model*

For the FGM model, if one item died at time t because of the cause one, its crude density, the probability density of the crude life, is as following:

$$
\int_t^{\infty} f_X(t,x_2)dx_2 = f_1(t)S_2(t)\ [\ 1-\delta\ F_2(t)(1-2F_1(t))\ ] \tag{3.25}
$$

otherwise, if one item died at time t because of the cause two, its crude density is

$$
\int_t^{\infty} f_X(x_1,t)dx_1 = f_2(t)S_1(t)\ [\ 1-\delta\ F_1(t)\ (1-2F_2(t))\ ] \tag{3.26}
$$

The equation (3.25) and (3.26) are derived in Appendix A. The derivations of the crude densities are an original contribution of this research. These formulas indicate the FGM model's well-known symmetrical relationships in the two-dimensional case and its form, which is an independent model with a perturbation term related to the dependence of risks. If $\delta = 0$, the FGM model becomes an independent competing risk model.

3.4. *Likelihood Function for the Survival Data*

In the following section, let f_{p1} and f_{p2} represent the probability density function (pdf) of time to death because of osteosarcoma and other diseases for individual exposed to ^{239}Pu respectively, f_{R1} and f_{R2} represent the pdf of time to death from osteosarcoma

and other diseases for individual exposed to ^{226}Ra respectively. $S_{p1}, S_{p2}, S_{R1}, S_{R2}$ are used to represent corresponding survival functions.

3.4.1. Likelihood Function for the Independent Competing Risk Model

The likelihood function of the independent competing risk model for the experimental data is

$$L(t_i, d_i, \vartheta) = L_{Pu}(t, d, \vartheta)\, L_{Ra}(t, d, \vartheta)$$
$$= \prod_{i=1}^{N1} f_{p1}(t_i, d_i, \vartheta) \prod_{j=1}^{N2} S_{p1}(t_j, d_j, \vartheta) \prod_{k=1}^{M1} f_{R1}(t_g, d_g, \vartheta) \prod_{L=1}^{M2} S_{R2}(t_q, d_q, \vartheta) \tag{3.27}$$

where f_{p1}, S_{p1} are defined in equation (3.21) and f_{R1}, S_{R1} are defined in equation (3.22), N_1 and M_1 are the number of beagles, which are exposed respectively to ^{239}Pu and ^{226}Ra and died because of osteosarcoma, N_2 and M_2 represent the number of beagles with other causes of deaths for Pu_Data and Ra_Data.

3.4.2. Likelihood Function for the FGM Model

For the FGM model, the crude density for a beagle exposed to ^{239}Pu which died with osteosarcoma is

$$\int_{t_{pi}}^{\infty} f_X(t_{p_i}, x_2)dx_2 = f_{p1}(t_{pi})S_{p2}(t_{pi})[1-\delta F_{p2}(t_{pi})(1-2F_{p1}(t_{pi}))] \ . \tag{3.28}$$

The crude density for the beagle exposed to ^{239}Pu which died because of another cause is

$$\int_{t_{pi}}^{\infty} f_X(x_1, t_{p_i})dx_1 = f_{p2}(t_{pi})S_{p1}(t_{pi})[1-\delta F_{p1}(t_{pi})(1-2F_{p2}(t_{pi}))]. \tag{3.29}$$

The crude density for the beagle exposed to ^{226}Ra which died because of osteosarcoma is

$$\int_{t_{Rii}}^{\infty} f_X(t_{Ri}, x_2)dx_2 = f_{R1}(t_{Ri})S_{R2}(t_{Ri})[1-\delta F_{R2}(t_{pi})(1-2F_{R1}(t_{Ri}))], \tag{3.30}$$

the crude density for the beagle exposed to ^{226}Ra which died from another cause is

$$\int_{t_{pi}}^{\infty} f_X(x_1, t_{Ri})dx_1 = f_{R2}(t_{Ri})S_{R1}(t_{Ri})[1-\delta F_{R1}(t_{Ri})(1-2F_{R2}(t_{Ri}))], \qquad (3.31)$$

where f_{p1}, S_{p1} are defined in equation (3.21), f_{p2}, S_{p2} are defined in equation (3.22), f_{R1}, S_{R1} are defined in equation (3.23), and f_{R2}, S_{R2} are defined in equation (3.24).

So the likelihood is:

$$L(\underline{\theta}|data) \propto \prod_{i=1}^{N1} \int_{t_i}^{\infty} f(t_i, x_2)dx_2 \prod_{j=1}^{N2} \int_{t_j}^{\infty} f(x_1, t_j)dx_1 \prod_{k=1}^{M1} \int_{t_k}^{\infty} f(t_k, x_2)dx_2 \prod_{l=1}^{M2} \int_{t_l}^{\infty} f(x_1, t_l)dx_1 \qquad (3.32)$$

where N1 is the number of beagles exposed to ^{239}Pu with death from an osteosarcoma,

N2 is the number of beagles exposed to ^{239}Pu with death from another cause,

M1 is the number of beagles exposed to ^{226}Ra with death from an osteosarcoma,

M2 is the number of beagles exposed to ^{226}Ra with death from another cause.

3.5. Bayesian Parameter Estimation

3.5.1. Posterior Densities

According to Bayes' theorem, the FGM model, the selected model and the likelihood function, the posterior density for all of parameters, including the toxicity ratio, ρ, is as following:

$$P(\rho, \delta, \alpha_1, \alpha_2, \lambda_1, \lambda_2, \beta, | t, d) \propto \propto L(t, d | \rho, \delta, \alpha_1, \alpha_2, \lambda_1, \lambda_2, \beta) * P_0(\rho, \delta, \alpha_1, \alpha_2, \lambda_1, \lambda_2, \beta) \qquad (3.33)$$

where the likelihood is expressed in equation (3.32) .

3.5.2. Bayesian Estimation of the Toxicity Ratio

For uniform prior, the marginal density for toxicity ratio is

$$P(\rho|t,d) \propto \int_0^1 \int_0^\infty \int_0^\infty \int_0^\infty \int_0^\infty \int_0^\infty L(t,d|\rho,\delta,\alpha_1,\alpha_2,\lambda_1,\lambda_2,\beta)\, d\delta\, d\alpha_1\, d\alpha_2\, d\lambda_1\, d\lambda_2\, d\beta \tag{3.34}$$

In the above equation, the parameter δ equals zero for an independent competing risk model.

1. Bayesian estimate of the toxicity ratio with the FGM model and three integration methods.

A few computer codes in C++ (see Appendix C) were developed to compute the marginal densities for the toxicity ratio with different numerical integration techniques (Lattice Rule, Monte Carlo method and Gaussian Quadrature Method) using a SunStation. Fig 3.11, Fig 3.12 and Fig 3.13 are flow charts of C++ programs respectively for each numerical integration method. For Monte Carlo method, thirty to sixty thousands of random vectors with six dimensions are chosen to perform the numerical integration. For Gaussian-Legendre quadrature, eight pairs of nodes and weights are taken from software Mathematica for each parameter, which lie in the range [0, 10]. There are 8^6 loops for parameters multiplied by the loops reading experimental data. For the Lattice rule, the recommended table (Table 2.1) [31] is used to determine N and L. After normalization, the marginal densities of the toxicity ratio, ρ, obtained from three numerical integration methods, are compared in Fig 3.14.

2. Comparison of marginal densities for toxicity ratio obtained by the FGM model and the independent competing risk model.

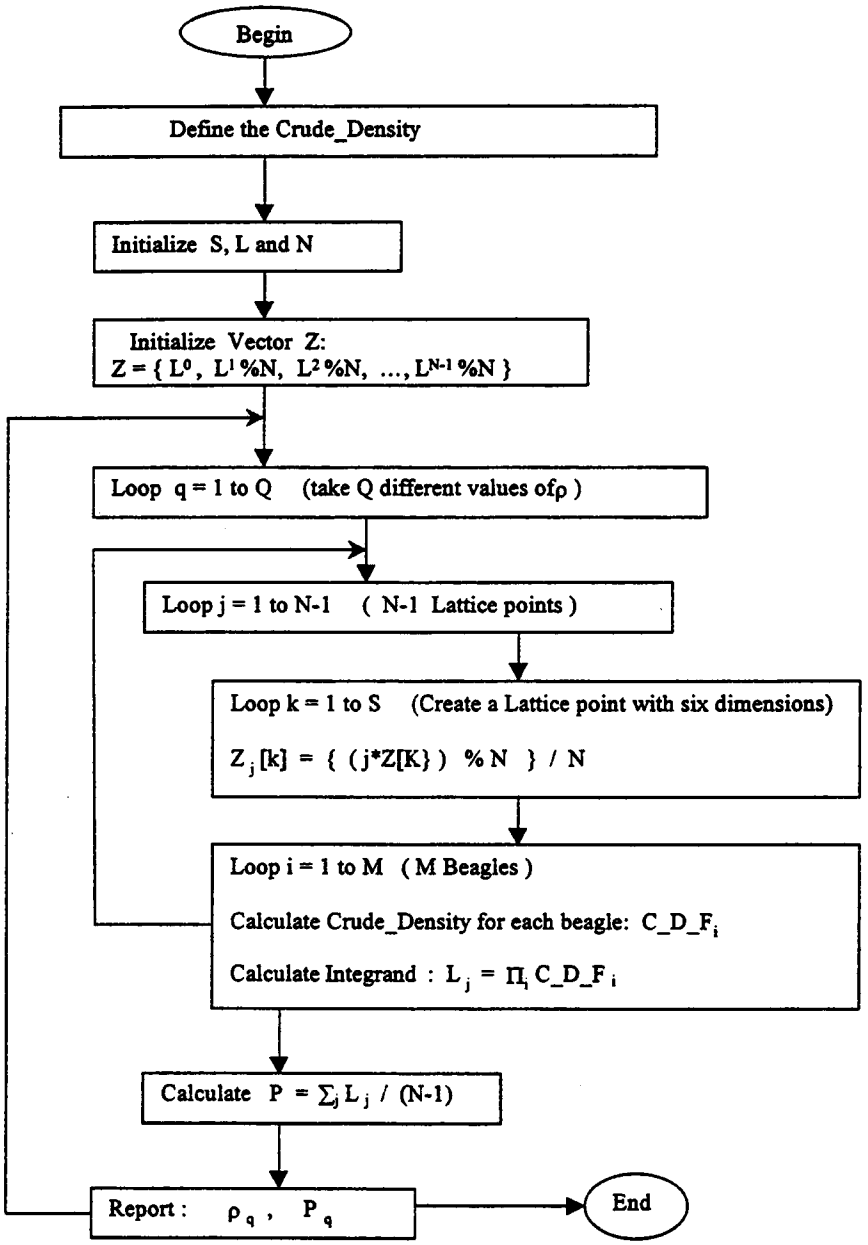

Figure 3.11. Flow-chart of the program to compute the marginal density of ρ using Lattice rules.

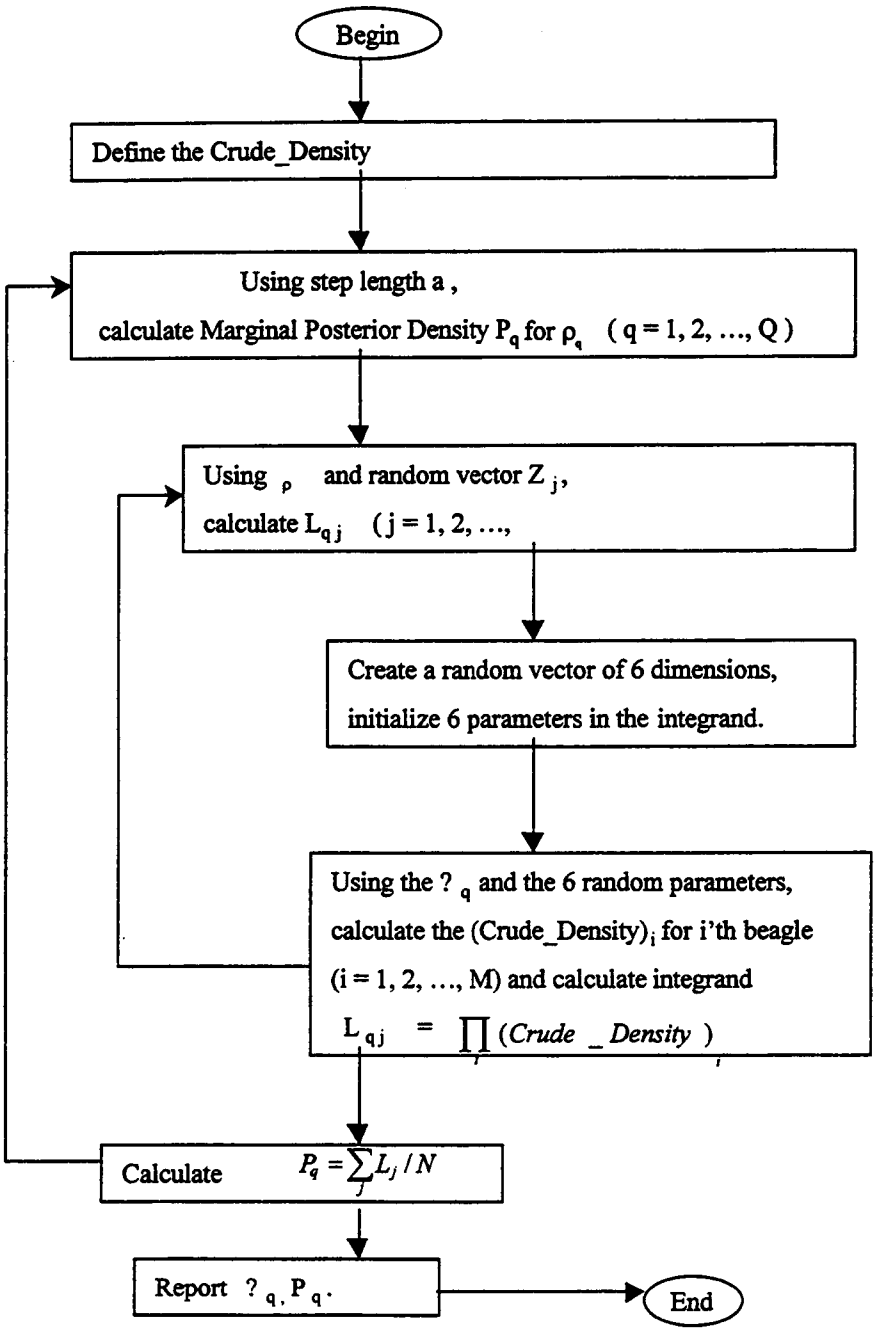

Figure 3.12. Flow chart of the program to compute the marginal density of ρ using Monte Carlo method

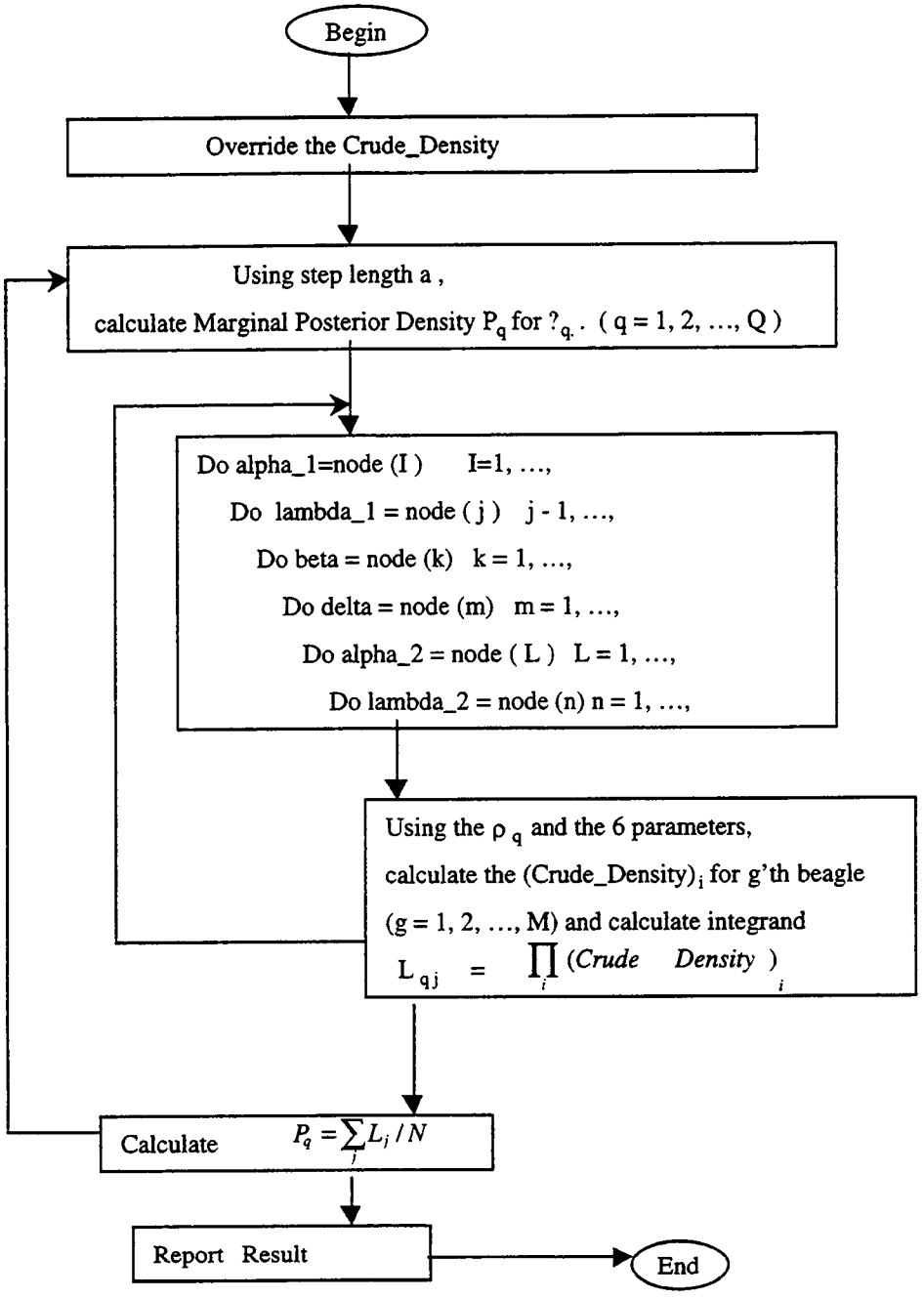

Figure 3.13. Flow-chart of the program to compute the marginal density of ρ using Gaussian Legendre quadrature.

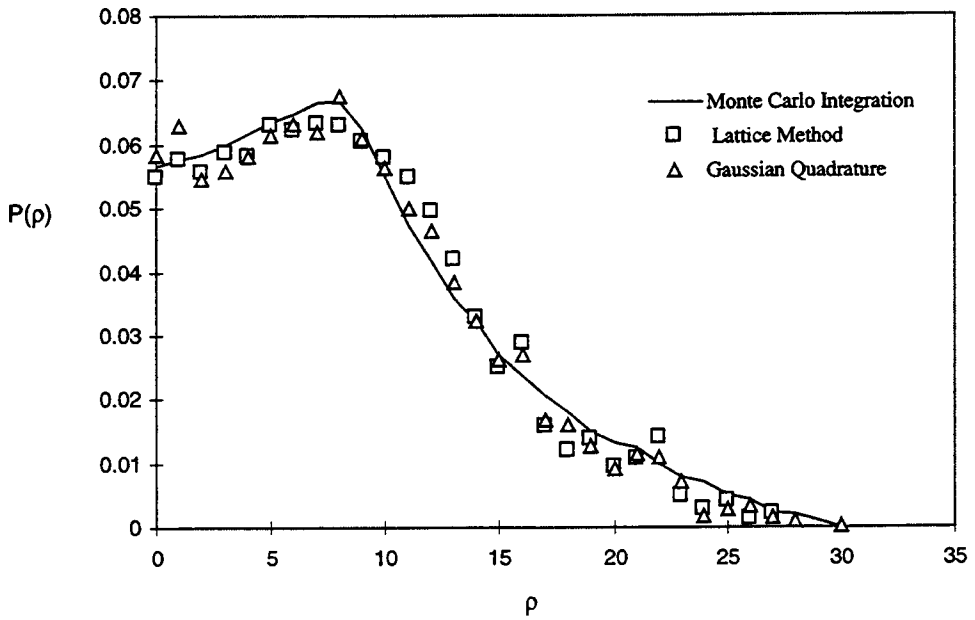

Fig 3.14. Comparison of posterior marginal densities of ρ

According to the likelihood function in equation (3.32), the marginal density of toxicity ratio is calculated with Monte Carlo method. The result is shown in Fig 3.15 and Fig 3.16 for the independent competing risk model and the FGM model respectively. Let $P_{dep}(\rho_i)$ and C_{dep} represent the marginal density of ρ and the normalization constant for the FGM model, P_{ind} and C_{ind} are used for the independent competing risk model. Then C_{dep} and C_{ind} are

$$C_{dep} = \frac{1}{\int_0^\infty P_{dep}(\rho)d\rho} \cong \frac{1}{\sum_1^N P_{dep}(\rho_i)} \quad , \tag{3.35}$$

$$C_{ind} = \frac{1}{\int_0^\infty P_{ind}(\rho)d\rho} \cong \frac{1}{\sum_1^N P_{ind}(\rho_i)} \quad . \tag{3.36}$$

The normalized posterior marginal densities of ρ are $C_{dep} * P_{dep}(\rho_i)$ and $C_{ind} * P_{ind}$ for the FGM model and the independent competing risk model respectively, which are compared in Fig 3.17. It is shown that the marginal density of ρ, obtained from the FGM model, appears very sharp, while the other one appears very flat. According to the equation (2.74), (2.76) and (2.77), the mean and standard deviation are computed respectively as

$$\bar{\rho} = \int_0^\infty \rho\, P(\rho) \cong \sum_1^N \rho_i P(\rho_i) \tag{3.37}$$

$$\begin{aligned} \sigma(\rho) &= \{ \int_0^\infty (\rho - \bar{\rho})^2 P(\rho) d\rho \}^{\frac{1}{2}} = \{ \int_0^\infty \rho^2 P(\rho) d\rho \; - \; \bar{\rho}^2 \}^{\frac{1}{2}} \\ &\cong \{ \sum_1^N (\rho_i)^2 P(\rho_i) \; - \; \bar{\rho}^2 \}^{\frac{1}{2}} \end{aligned} \tag{3.38}$$

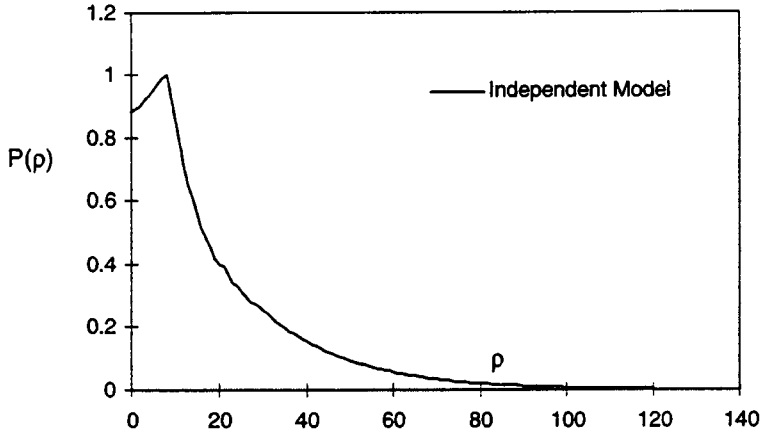

Fig 3.15. Bayesian estimation of the toxicity ratio ρ for the independent model

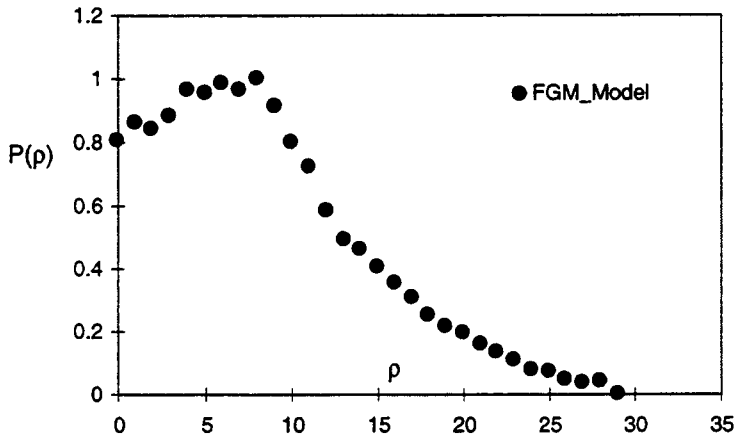

Fig 3.16. Bayesian estimation of the toxicity ratio ρ for FGM model

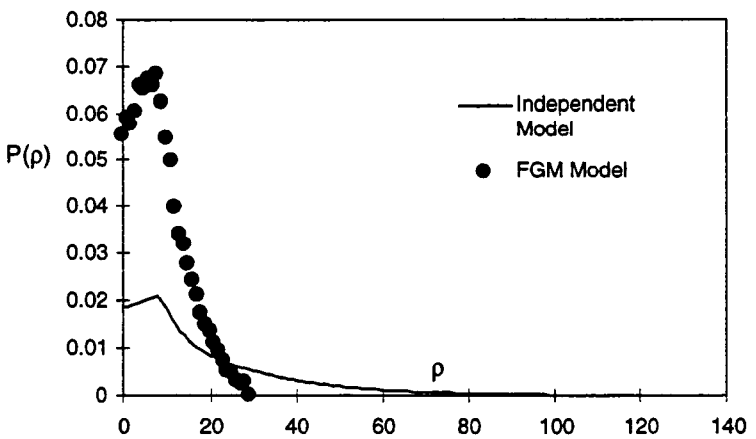

Fig 3.17. Comparison of toxicity ratio ρ

The computed modes, means and standard deviations for the FGM model and the independent competing risk model are compared in Table 3.8.

Table 3.8. Comparison of Toxicity Ratio Estimates

Toxicity Ratio ρ	FGM Model	Independent Model
Mode	7.87	7.87
Mean	8.55	19.54
STD	5.02	18.57

3.5.3. Bayesian Estimate of Other Parameters

The marginal density of α_1 is expressed as:

$$P(\alpha_1 | Y) \propto$$

$$\propto \int_0^\infty \int_0^1 \int_0^\infty \int_0^\infty \int_0^\infty \int_0^\infty L(t,d|\rho,\delta,\alpha_1,\beta,\lambda_1,\alpha_2,\lambda_2) * P_0(\rho,\delta,\alpha_1,\beta,\lambda_1,\alpha_2,\lambda_2) \quad (3.39)$$

$$d\rho\ d\delta\ d\beta\ d\lambda_1\ d\alpha_2\ d\lambda_2$$

The results obtained from the independent competing risk model and the FGM model are shown respectively in Fig 3.18 and Fig 3.19. They are compared in Fig 3.20.

The marginal density of β is expressed as:

$$P(\beta | Y) \propto$$

$$\propto \int_0^\infty \int_0^1 \int_0^\infty \int_0^\infty \int_0^\infty \int_0^\infty L(t,d|\rho,\delta,\alpha_1,\beta,\lambda_1,\alpha_2,\lambda_2) * P_0(\rho,\delta,\alpha_1,\beta,\lambda_1,\alpha_2,\lambda_2) \quad (3.40)$$

$$d\rho\ d\delta\ d\alpha_1\ d\lambda_1\ d\alpha_2\ d\lambda_2$$

The results obtained from the independent competing risk model and the FGM model are shown respectively in Fig 3.21 and Fig 3.22. They are compared in Fig 3.23.

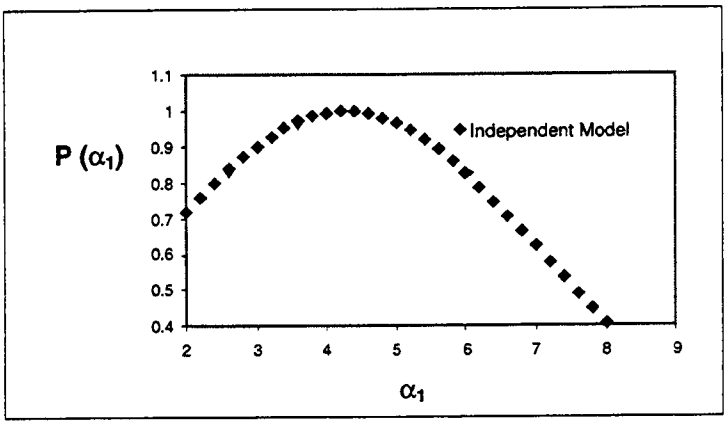

Figure 3.18. Bayesian estimate of α_1 for the independent model

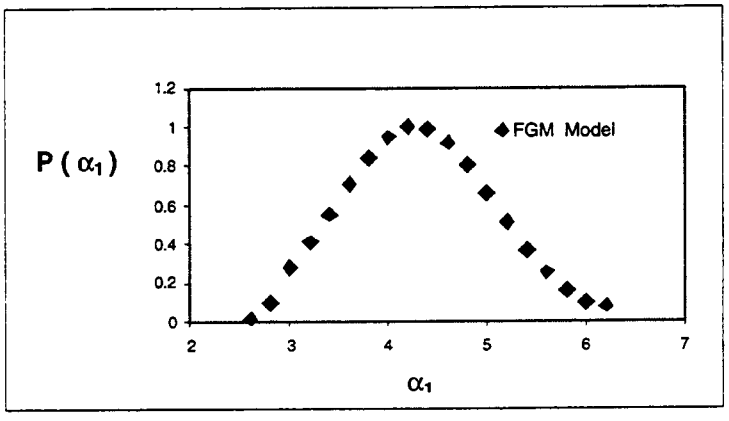

Figure 3.19. Bayesian estimate of α_1 for the FGM model

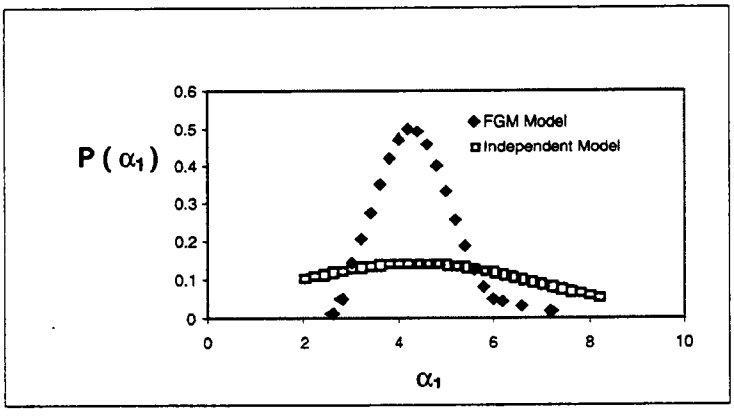

Figure 3.20. Comparison of posterior marginal densities of α_1

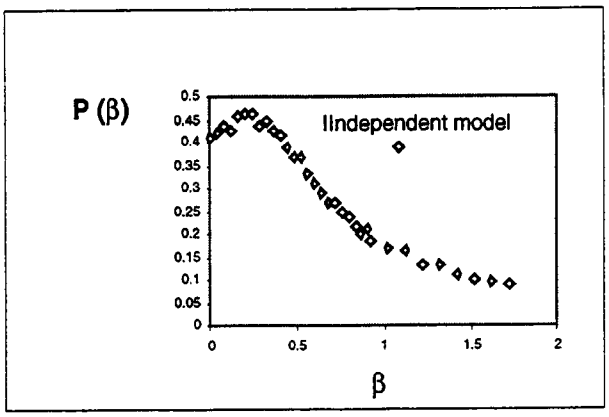

Figure 3.21. Bayesian estimate of β for the independent model

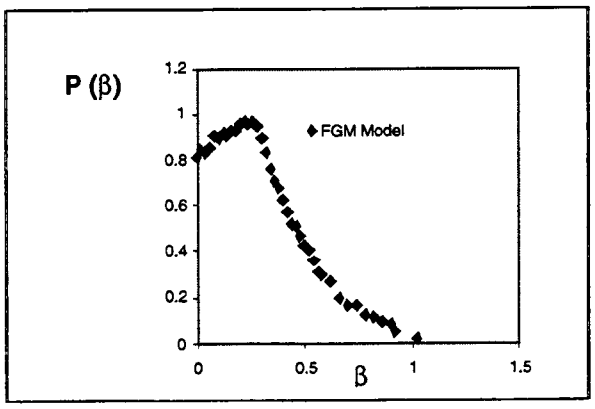

Figure 3.22. Bayesian estimate of β for the FGM model

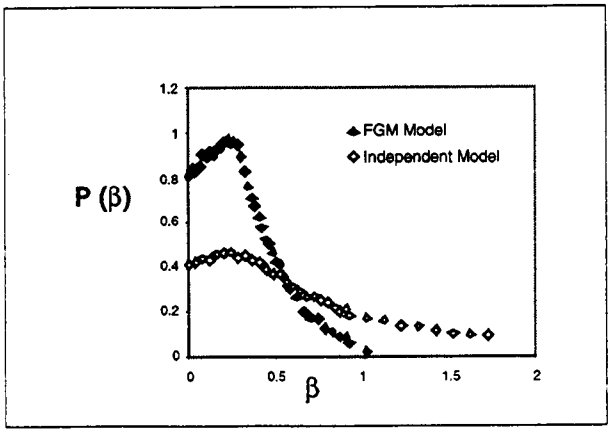

Figure 3.23. Comparison of the marginal densities of β

The marginal density of λ_1 is expressed as:

$$P(\lambda_1 | Y) \propto$$

$$\propto \int_0^\infty \int_0^1 \int_0^\infty \int_0^\infty \int_0^\infty \int_0^\infty L(t,d|\rho,\delta,\alpha_1,\beta,\lambda_1,\alpha_2,\lambda_2) * P_0(\rho,\delta,\alpha_1,\beta,\lambda_1,\alpha_2,\lambda_2)$$
$$d\rho\ d\delta\ d\beta\ d\alpha_1\ d\alpha_2\ d\lambda_2$$

(3.41)

The results obtained from the independent competing risk model and the FGM model are shown respectively in Fig 3.24 and Fig 3.25. They are compared in Fig 3.26.

The marginal density of λ_2 is expressed as :

$$P(\lambda_2 | Y) \propto$$

$$\propto \int_0^\infty \int_0^1 \int_0^\infty \int_0^\infty \int_0^\infty \int_0^\infty L(t,d|\rho,\delta,\alpha_1,\beta,\lambda_1,\alpha_2,\lambda_2) * P_0(\rho,\delta,\alpha_1,\beta,\lambda_1,\alpha_2,\lambda_2) \quad (3.42)$$
$$d\rho\ d\delta\ d\beta\ d\lambda_1\ d\alpha_2\ d\alpha_1$$

The result obtained from FGM model is shown in Fig 3.27.

The marginal density of α_2 is expressed as :

$$P(\alpha_2 | Y) \propto$$

$$\propto \int_0^\infty \int_0^1 \int_0^\infty \int_0^\infty \int_0^\infty \int_0^\infty L(t,d|\rho,\delta,\alpha_1,\beta,\lambda_1,\alpha_2,\lambda_2) * P_0(\rho,\delta,\alpha_1,\beta,\lambda_1,\alpha_2,\lambda_2) \quad (3.43)$$
$$d\rho\ d\delta\ d\beta\ d\lambda_1\ d\lambda_2\ d\alpha_1$$

The result obtained from FGM model is shown in Fig 3.28.

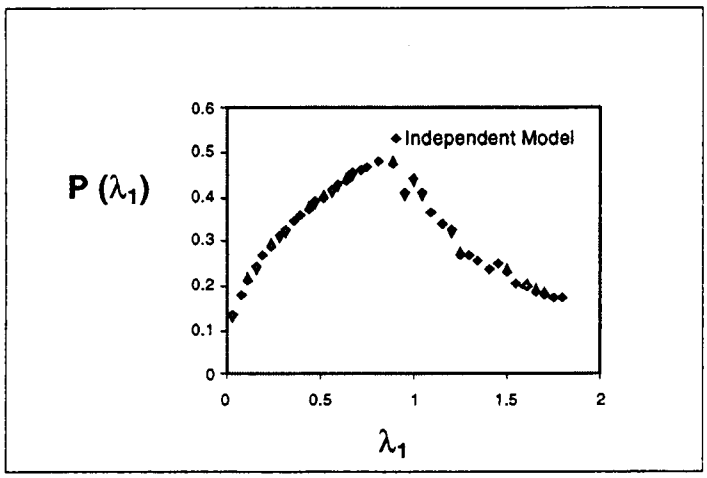

Figure 3.24. Bayesian estimate of λ_1 for the independent model

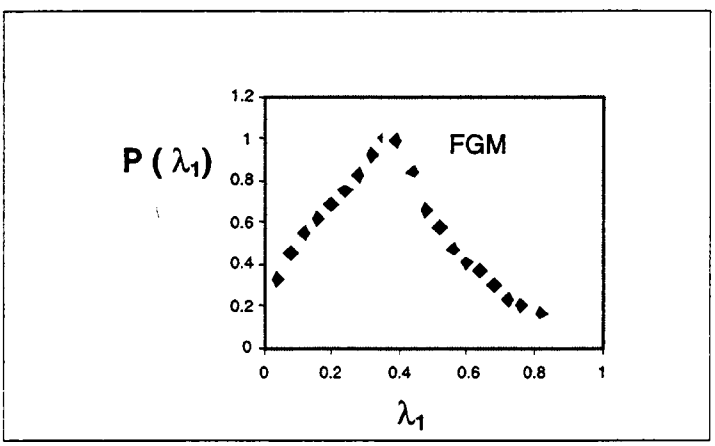

Figure 3.25. Bayesian estimate of λ_1 for the FGM model

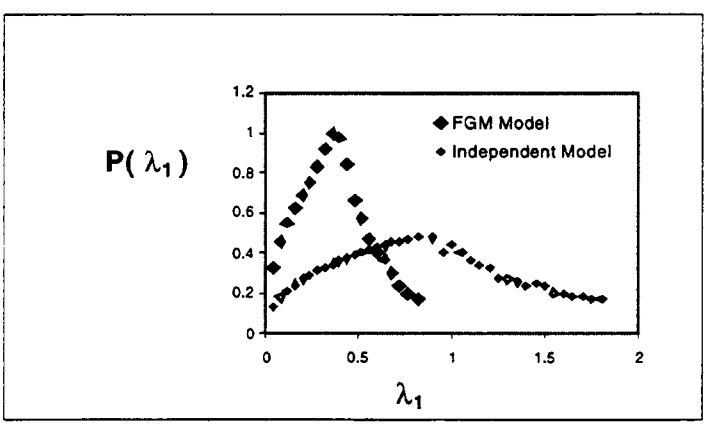

Figure 3.26. Comparison of marginal densities of λ_1

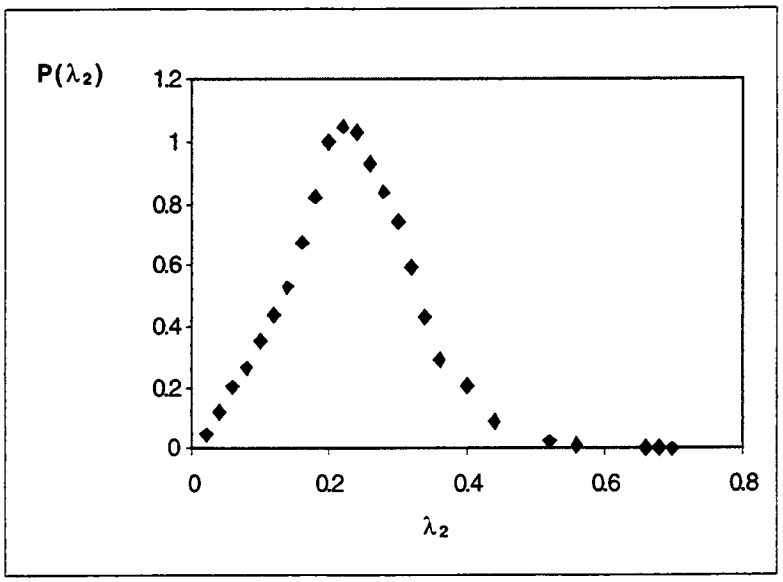

Figure 3.27. Bayesian estimate of the posterior marginal density of λ_2 in FGM model.

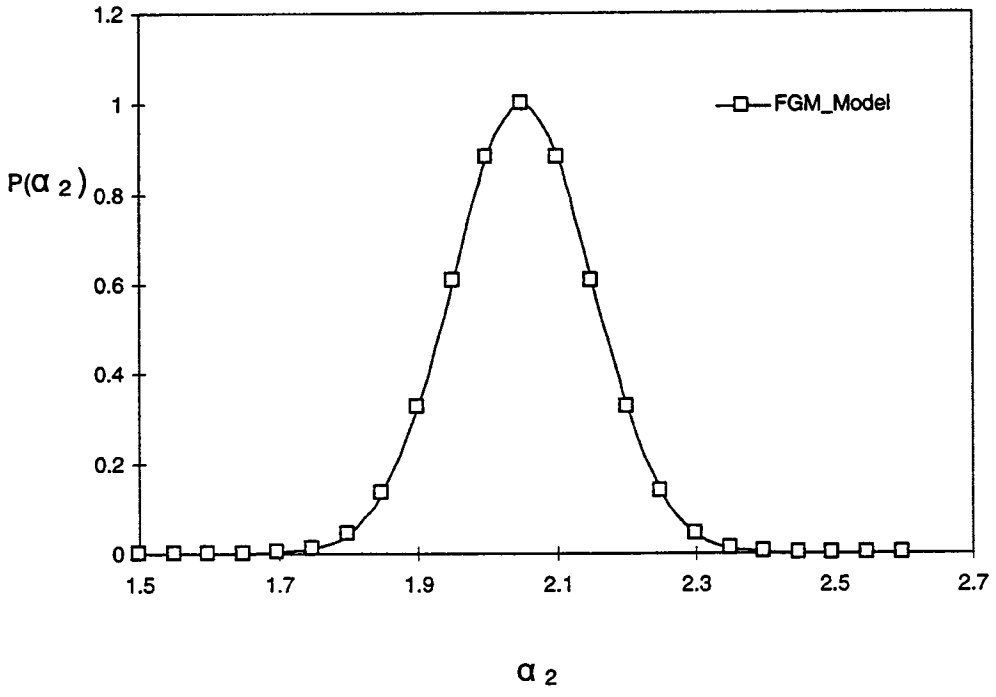

Fig 3.28. Bayesian estimation of α_2 for FGM model

3.6. Comparison of Cumulative Hazard of FGM model with Nelson's Method.

The modes of the parameters in FGM model estimated by Bayesian technique are shown in Table 3.9.

Table 3.9. Modes of Parameters

Parameter	α_1	β	λ_1	ρ	α_2	λ_2
Mode	4.45	0.32	0.94	7.87	2.05	0.22

Using the modes of the posterior densities as point estimates of parameters, the cumulative hazard for the ^{239}Pu-induced osteosarcoma is calculated by equation (3.21), where the lifetime comes from Pu_Data, the activity is set equal to the mean activity of the corresponding injection level. Fig 3.29 shows the calculated cumulative hazard compared with the Nelson's cumulative hazard plot.

3.7. Radiation-induced Osteosarcoma From ^{239}Pu in Comparison With ^{226}Ra

Using the modes of the posterior densities as point estimates of parameters, equations (3.21) are used to calculate the net cumulative hazard, the net survival probability, the net cumulative distribution, and the net failure density for ^{239}Pu-induced osteosarcoma, while equations (3.23) are used to calculate these functions for ^{226}Ra-induced osteosarcoma.

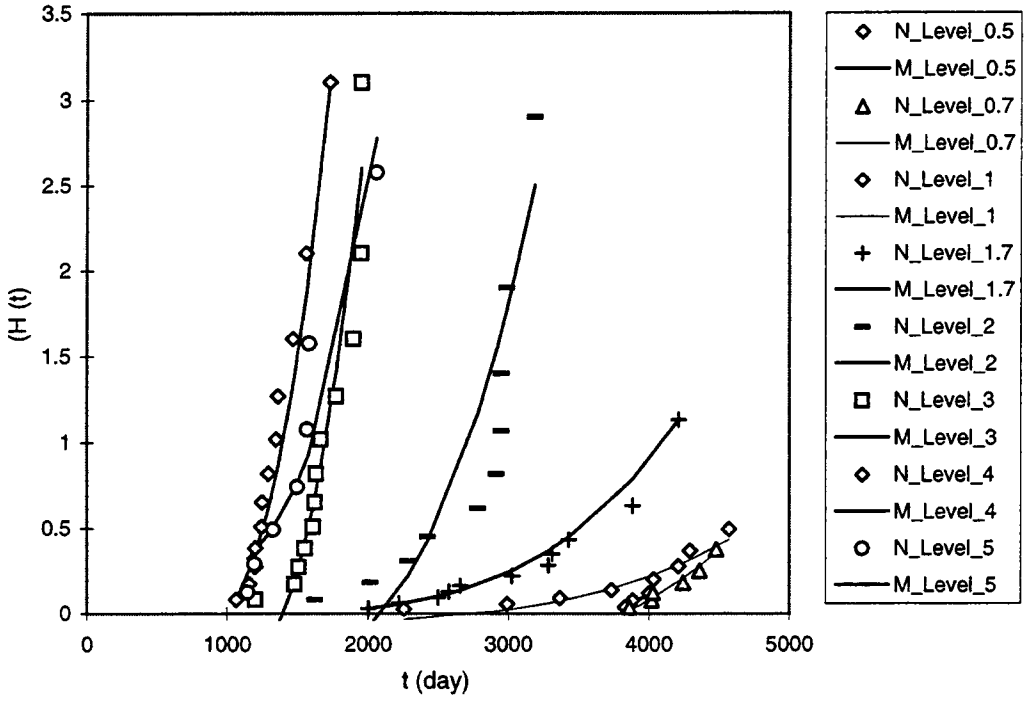

Fig 3.29. Comparison of cumulative hazard between FGM Model and Nelson's method

In these calculations, a value of $0.85\,\mu C_i$ is taken as the activity, and the time ranges from 1000 days to 4000 days. The value of $0.85\,\mu C_i$ is the mean activity of the injection level 2 for Pu_data, which is approximately the mean activity of the injection level 1 for Ra_Data. Under the almost same activities, the mean lifetime is 2495 days for Pu_Data, but 3927 days for Ra_data.

The results are compared in Fig 3.30 to Fig 3.33.

Fig 3.30 shows the much faster increase of the cumulative hazard for ^{239}Pu-induced osteosarcoma as compared with the ^{226}Ra-induced osteosarcoma. The relative risk coefficient, $\exp(\rho\beta d) / \exp(\beta d)$, is approximately 5.7 with the injection activity 0.85 μC_i, while it is 9.0 for the injection activity 1.0 μC_i.

Fig 3.31 and Fig 3.32 show the much faster decrease of the survival probability and the much faster grows of the failure distribution for ^{239}Pu, as compared with ^{226}Ra.

The failure densities are compared in Fig 3.33. The failure density of ^{239}Pu-induced osteosarcoma has a smaller mode at about 1200 days as compared with the mode of 1500 days for ^{226}Ra.

These figures show that the ^{239}Pu is more effective than ^{226}Ra for inducing osteosarcomas.

3.8 Bayesian Prediction for Crude Distribution

Bayesian prediction technique, illustrated by equation (2.52). can be used to predict the crude distribution. If the function $Q(x^*, \theta)$ equals the crude distribution for osteosarcoma, the prediction of the crude distribution for osteosarcoma is expressed as:

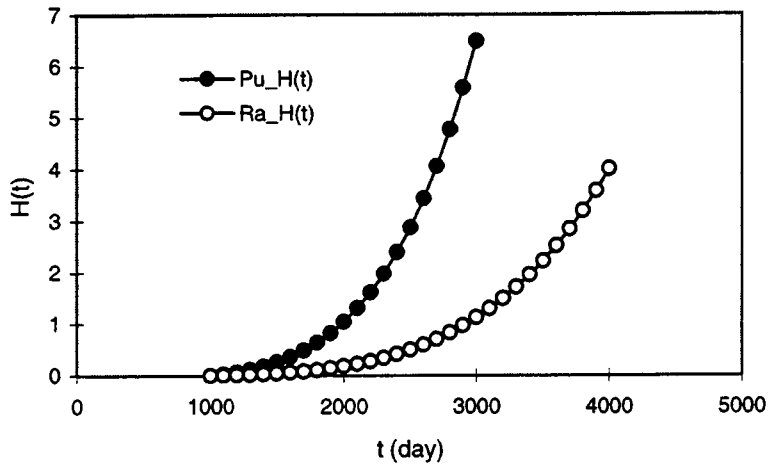

Fig 3.30. Comparison of cumulative hazard between ^{239}Pu and ^{226}Ra

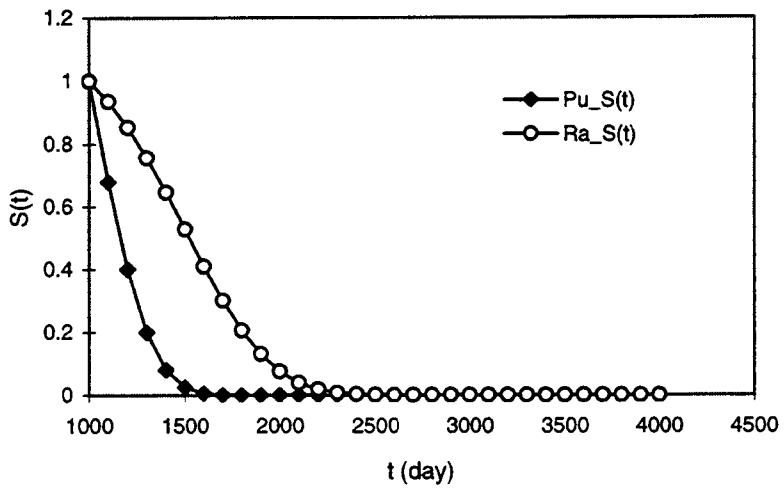

Fig 3.31. Comparison of survival probabilities between ^{239}Pu and ^{226}Ra

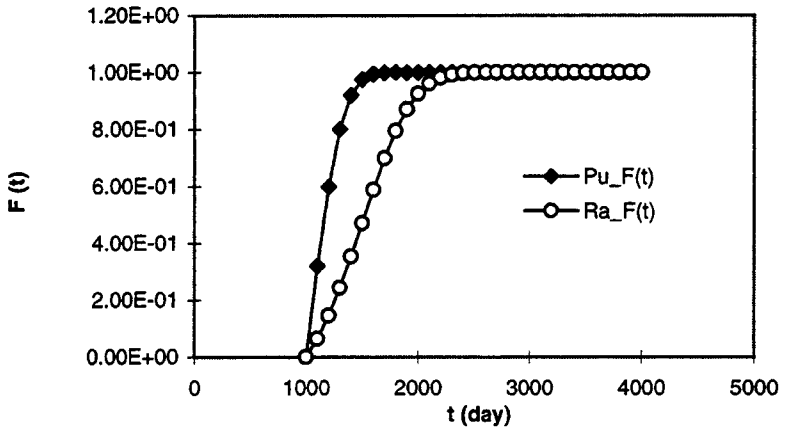

Fig 3.32. Comparison of cumulative distribution between 239Pu and 226Ra

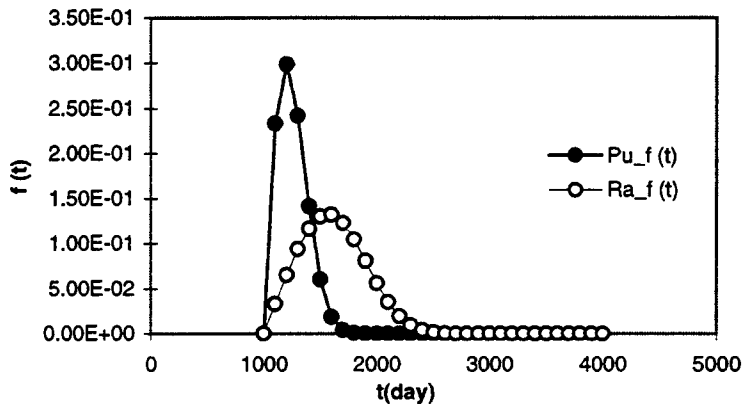

Fig 3.33. Comparison of failure density between ^{239}Pu and ^{226}Ra

$$F_1(t_i) = \int_{\vartheta} F_1(t_i|\vartheta) P(\vartheta|X) d\vartheta \tag{3.44}$$

Since

$$F_1(t_i) = \int_0^t f_1(t)\, dt \tag{3.45}$$

and the crude density is

$$f_1(t) = \int_t^{\infty} f_X(t, x_2)\, d x_2 \tag{3.46}$$

the Bayesian prediction of the cumulative distribution for osteosarcoma is

$$\begin{aligned} F_1(t_i) &= \int_{\vartheta} \{\int_0^t \int_t^{\infty} f_X(t, x_2|\vartheta) dx_2\, dt]\ P(\vartheta|X)\ d\vartheta \\ &= \int_0^t \int_{\vartheta} f_{p1} S_{p2} (1 - \delta F_{p2}(1 - 2F_{p1}))\ P(\vartheta|X)\, d\vartheta\ dt \end{aligned} \tag{3.47}$$

where the posterior density P (θ | X) is given by equation (3.29).

The purpose of failure time analysis is to extract some information about net life distribution from the observations. If the Q (x^* , θ) in equation (2.52) is taken as net survival function S_{p1} (x^*, θ), the predictive net survival function from equation (2.52) will describe the time dependence of ^{239}Pu -induced osteosarcomas. If the Q (x^* , θ) in equation (2.52) is taken as S_{R1} (x^*, θ), then the predicted net survival probability will describe the time dependence of ^{226}Ra -induced osteosarcomas.

CHAPTER 4. CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE RESEARCH

4.1 Conclusions

The present work shows that ^{239}Pu is more dangerous than ^{226}Ra in radiation-induced osteosarcoma for dogs. The mode of toxicity ratio between ^{239}Pu and ^{226}Ra is estimated as about 7.87. Using the modes of the posterior densities as point estimates of parameters, the toxicity of ^{239}Pu in osteosarcoma is predicted and compared with that of ^{226}Ra by calculating the cumulative hazard, the survival probability, the cumulative distribution and the failure density for an injected activity of 0.85 μC_i. The relative risk coefficient, $\exp(\rho\beta d) / \exp(\beta d)$, depends on the injected activity. For example in this study, the cumulative hazard of plutonium is approximately 5.7 times of that of radium for the injected activity of 0.85 μC_i, while it is 9.0 times is for the activity of 1.0 μC_i.

The relative risk model with the Weibull baseline is studied and used to describe the radiation-induced osteosarcoma due to ^{239}Pu and ^{226}Ra. Using the modes of the posterior densities as point estimates for the parameters in the model, the cumulative hazard for ^{239}Pu is calculated for the mean activity of each injection level, and compared with Nelson's cumulative hazard estimate. These results are comparable, which indicates that this model is suitable for analysis of these survival data.

Both the dependent competing risk model, the FGM model, and the independent competing risk model are applied to construct likelihood functions. In comparison with

the independent competing risk model, the FGM model produces different estimates of the posterior marginal densities of the parameters. This demonstrates that dependent competing risks influence the parameter estimates. This demonstration is original and significant for the analysis of survival data with dependent competing risks, where failures of one type affect the probability of failure of another type.

In this work, Bayesian techniques are introduced to estimate all parameters in terms of posterior marginal densities, which fully describe the uncertainty about these parameters and contain all necessary information about them. When new data become available, the posterior densities can be updated and so will the relative risk model. Also, Bayesian prediction techniques are able to predict probability for radiation-induced osteosarcoma under consideration of parameter uncertainty, given available survival data and selected future times and injected activities.

Three numerical integration methods (Monte Carlo, Gaussian Quadrature and Lattice rule) are applied to carry out Bayesian parameter estimation. They produce comparable results for the marginal posterior density of the toxicity ratio ρ. The codes to carry out these computations were developed. There is no doubt the great advances in computer technologies will pave the road to a much more widespread applications of Bayesian method to more complex systems.

4.2 Contributions

We performed a comparison of the relative toxicity of ^{239}Pu and ^{226}Ra with the relative risk model, the FGM model and the independent competing risk model. The

Bayesian techniques and application of several numerical integration methods especially the Lattice method, are the original contributions of this work. The Bayesian approach used here is novel in the field of radiation risk assessment, particularly, for toxicity studies.

4.3 Recommendations for Future Research

The theory and methods described in this work is by no means limited to the study of radiation-induced osteosarcoma. They are suitable for modeling and analysis of a wide variety of reliability data, particularly, data with dependent competing failure modes.

The comparison of the toxicity of plutonium and radium is achieved by using the modes of the posterior densities as point estimates for the parameters of the relative risk model. In this study a set of equations are derived according to Bayesian prediction techniques. In further work these equations could be applied to predict a variety of functions useful for the modeling and the analysis of the radiation-induced osteosarcoma by ^{239}Pu and ^{226}Ra.

The current work compared the toxicity of ^{239}Pu and ^{226}Ra in beagles. Together with the ^{226}Ra data, the risk to humans from ^{239}Pu can be extrapolated using the results of this work. This is one of the final goals in the field of radiation protection. Such extrapolation of ^{239}Pu-risk from a wide variety animal to humans with a simple model was already done by DuMouchol [37]. Also, more work and different applications are essential to evaluate the impact of dependent competing risks on estimation. The

application of other dependent competing risk models would provide interesting comparisons with the present result.

REFERENCES

REFERENCES

1. Evans, R. D. "Radium in man", Health Physics 27, pp. 497-519, 1974,.

2. Turner, James R. "Atoms, Radiation, and Radiation Protection", second edition, John Wiley & Sons, Inc. pp. 516, 1995.

3. Daynes, Raymond A. "Research in Radiobiology (Annual Report of Work in Progress in the internal irradiation Program)", A30-36, Radiobiology Division Department of Pharmacology, University of Utah School of Medicine, pp. 1, A15-27, December 1986.

4. Toohey, R. E. "Biokinetics of bone-seeking radionuclides", Internal Radiation Dosimetry, Health Physics Society, 1994 Summer School, pp. 198,. Medical Physics Publishing, Madison, Wisconsin.

5. Chiang, C. L. "Competing risks and conditional probabilities". Biometrics 26, 767-76, 1970.

6. Moeschberger, M. L. and David, H. A. "Life tests under competing causes of failure and the theory of competing risks", Biometrics 27, pp. 909-33, 1971.

7. Nelson, W. "Hazard plotting for incomplete failure data", Journal of Quality Technology 1, pp. 27-52, 1969.

8. Barlow, R.E. and Campo, R.A. "Total time on test processes and applications to failure data analysis", Reliability and failure tree analysis, pp. 451-481, SIAM, Philadelphia. 1975.

9. Herman, R. J. and Patell, R.K.N. "Maximum Likelihood Estimation for Multi-risk Model", Technometrics 13:385-396. 1971.

10. Marshall, A.W. and Olkin, I., "A multivariate exponential distribution" J. Am. Sta. Assoc.62:30-44, 1967.

11. Musa, J. D. A theory of software reliability and its applications. IEEE Trans. Software Engng., SE-1, 312-27, 1975.

12. Johnson, N. I. and Kotz, Samuel, "On some generalized Farlie-Gumbel-Morgenstern Distributions", Communications in Statistics, 4(5), pp.415-427, 1975.

13. David Oakes, "A Model for Association in Bivariate Survival Data", J. R.

Statistics. Soc. B(1982), 44. No. 3. pp 412-422.

14. Christian Genest and Jock MacKey, The Joy of Copulas: "Bivariate Distributions With Uniform Marginals", The American Statistician, Vol.40. No.4. pp 280-282, November 1986.

15. Christian Genest, "Frank's family of Bivariate distributions", Biometrika 74.3.pp.549-555, 1987.

16. Slud, E. and Byar, D. "How Dependent Causes of Death Can Make Risk Factors Appear Protective", Biometrics 44, 265-269, March 1988.

17. George Casella and Berger, Roger L. Statistical Inference, Wadsworth & Books/Cole Advanced Books & Software, Pacific Grove, California, pp.284 - 328, 1990.

18. Martz, H. F. and Waller, R. A., Bayesian Reliability Analysis, pp 171-172, John Wiley & Sons, 1982.

19. Crowder, M. J., Kimber, A. C., Smith, R. L. and Sweeting, T. J. Statistical Analysis of Reliability Data, Chapman & Hall, NY. pp 2-3. 1988.

20. Kimball, A. W., Models for the estimation of competing risks from grouped data, Biometrics 25, 329-37, 1969.

21. Press, William H., Teukolsky, Saul A., Vetterling, William T. and Flannery, Brian P. (1992). Numerical Recipes in C, The art of scientific computing, Second edition, Press Syndicate of University of Cambridge, pp. 147-165.

22. JMP Start Statistics, SAS Institute Inc, 1990.

23. Lee, P. M. , Bayesian Statistics: An Introduction, Oxford University. Press., New York, 1989.

24. Smith, A. F. M. "An Overview of the Bayesian Approach" , in Sander, P., and Badoux, R., Eds., Bayesian Methods in Reliability, Kluwar Academic Publishers, Netherlands, pp.15-79, 1991..

25. Box, G. E. P., and Tiao, G. C., Bayesian Inference in Statistical Analysis, Addison-Wesley, New York, 1973.

26. Lindley, D. V., "The Bayesian Approach", Scand. J. Statistics, 5, 1, 1978.

27. Bagchi, P., "Assessment of Prior Distributions based on Information Sciences,

7, 117-128, 1986.

28. Press, S. J., Bayesian Statistics: Principles, Models, and Applications, John Wiley and Sons, New York, 1989.

29. Aitkin, Murray, "Posterior Bayes Factors", J. R. Statist. Soc. B 53, 111-142, 1991.

30. Atkinson, A. C., "Posterior Probabilities for Choosing a Regression Model", Biometrika, 65, 39-48, 1978.

31. Sloan, I. H. and S, Joe., Lattice Methods for Multiple Integration, Clarendon Press, Oxford, pp.11, 24, 132-133, 165-173, 1994.

32. Sloan, I. H., and Kachoyan, P. J., "Lattice Methods for Multiple Integrations: Theory, Error Analysis and Examples", SIAM Journal on Numerical Analysis, 24, 116-128, 1987.

33. Ivan Lux & Laszlo Koblinger, "Monte Carlo Particle Transport Methods: Neutron and Photon Calculations" , pp. 5-30, CRC Press Inc, Boston, 1991.

34. Van Dijk, H. K., "Some Advances in Bayesian Estimation Models Using Monte Carlo Integration", in Fomby, Thomas B., and Rhodes, George F., Edis., Advances in Econometrics, JAI Press Inc., 6, 215-261, 1987.

35. Howson Colin and Urbach Peter, Scientific Reasoning: The Bayesian Approach, Open Court Publishing Company, pp.15-31, 1989.

36. Turner, J. E., Wright, H. A. and Hamm, R. N., "Review Article A Monte Carlo Primer For Health Physicists", Health Physics, Vol 48, No. 6.(June), pp. 717 - 733, 1985.

37. DuMouchel, W. H. , Groer, P. G. , "A Bayesian methodology for scaling radiation studies from animals to man". Health Physics, 57, Supplement 1, 411-418, 1989.

APPENDICES

APPENDIX A

DERIVATIONS

APPENDIX A. DERIVATIONS

1. Relative Risk Model

For Pu_Data with Osteosarcoma:

Pu_1_hazard_Rate :

$$h_{p1}(t,d,\alpha_1,\lambda_1,\beta,\rho) = \alpha_1\lambda_1(\lambda_1 t)^{\alpha_1-1}\exp(\rho\beta d) \tag{A.1}$$

Pu_1_Cumulative hazard :

$$H_{p1}(t,d,\alpha_1,\lambda_1,\beta,\rho)=(\lambda_1 t)^{\alpha_1}\exp(\rho\beta d) \tag{A.2}$$

Pu_1_Survival function

$$S_{p1}(t) = \exp\{-(\lambda_1 t)^{\alpha_1}\exp(\rho\beta d)\}, \tag{A.3}$$

Pu_1_Failure density

$$f_{p1}(t) =\alpha_1\lambda_1(\lambda_1 t)^{\alpha_1-1}\exp(\rho\beta d)\exp\{-(\lambda_1 t)^{\alpha_1}\exp(\rho\beta d)\}. \tag{A.4}$$

For Pu_data without Osteosarcoma :

Pu_2_Hazard rate

$$h_{p2}(t,d,\alpha_2,\lambda_2) = \alpha_2\lambda_2(\lambda_2 t)^{\alpha_2-1} \tag{A.5}$$

Pu_2_Cumulative hazard

$$H_{p2}(t,d,\alpha_2,\lambda_2) = (\lambda_2 t)^{\alpha_2} \tag{A.6}$$

Pu_2_Survival Function

$$S_{p2}(t) = \exp\{-(\lambda_2 t)^{\alpha_2}\}, \tag{A.7}$$

Pu_2_Failure density

$$f_{p2}(t) =\alpha_2\lambda_2(\lambda_2 t)^{\alpha_2-1}\exp\{-(\lambda_2 t)^{\alpha_2}\}. \tag{A.8}$$

for Ra_Data with Osteosarcoma :

Ra_1_Hazard rate

$$h_{R1}(t,d,\alpha_1,\lambda_1,\beta) = \alpha_1 \lambda_1 (\lambda_1 t)^{\alpha_1 - 1} \exp(\beta d) \tag{A.9}$$

Ra_1_Cumulative hazard

$$H_{R1}(t,d,\alpha_1,\lambda_1,\beta) = (\lambda_1 t)^{\alpha_1} \exp(\beta d) \tag{A.10}$$

Ra_1_Survival function

$$S_{R1}(t) = \exp\{-(\lambda_1 t)^{\alpha_1} \exp(\beta d)\}, \tag{A.11}$$

Ra_1_Failure density

$$f_{R1}(t) = \alpha_1 \lambda_1 (\lambda_1 t)^{\alpha_1 - 1} \exp(\beta d) \exp\{-(\lambda_1 t)^{\alpha_1} \exp(\beta d)\}. \tag{A.12}$$

for Ra_data without Osteosarcoma :

Ra_2_Hazard rate

$$h_{R2}(t,d,\alpha_2,\lambda_2) = \alpha_2 \lambda_2 (\lambda_2 t)^{\alpha_2 - 1} \tag{A.13}$$

Ra_2_Cumulative hazard

$$H_{R2}(t,d,\alpha_2,\lambda_2) = (\lambda_2 t)^{\alpha_2} \tag{A.14}$$

Ra_2_Survival function

$$S_{R2}(t) = \exp\{-(\lambda_2 t)^{\alpha_2}\}, \tag{A.15}$$

Ra_2_Filure density

$$f_{R2}(t) = \alpha_2 \lambda_2 (\lambda_2 t)^{\alpha_2 - 1} \exp\{-(\lambda_2 t)^{\alpha_2}\}. \tag{A.16}$$

2. Crude Density for Independent Competing Risk Model

Joint density of Independent Competing Risk Model

Joint failure density

$$f(t_1, t_2) = f_1(t_1) * f_2(t_2) \tag{A.17}$$

Crude Density of Independent Competing Risk Model

For the cause of death is osteosarcoma, let $t_1 = t$.

Crude density for independent competing model

$$\begin{aligned} f(t) &= \int_t^\infty f_1(t_1 = t) f_2(t_2) dt_2 \\ &= f_1(t) \int_t^\infty f_2(t_2) dt_2 \\ &= f_1(t) S_2(t) \end{aligned} \tag{A.18}$$

For the cause of death is other diseases, let $t_2 = t$, the crude density is

$$f(t) = f_2(t) S_1(t) \tag{A.19}$$

So for the item irradiated by Pu-239 died at time t and the cause of death is osteosarcoma, its crude density is

$$\begin{aligned} f_{p1} &= f_1(t) * S_2(t) \\ &= \alpha_1 \lambda_1 (\lambda_1 t)^{\alpha_1 - 1} \exp(\rho \beta d) \exp\{-(\lambda_1 t)^{\alpha_1} \exp(\rho \beta d) \\ &\quad * \exp\{-(\lambda_2 t)^{\alpha_2}\} \end{aligned} \tag{A.20}$$

For the item irradiated by Pu-239 died at time t and the cause of death is other diseases, its crude density is

$$\begin{aligned} f_{p2}(t) &= f_2(t) S_1(t) \\ &= \alpha_2 \lambda_2 (\lambda_2 t)^{\alpha_2 - 1} \exp\{-(\lambda_2 t)^{\alpha_2}\} \\ &\quad * \alpha_1 \lambda_1 (\lambda_1 t)^{\alpha_1 - 1} \exp(\rho \beta d) \exp\{-(\lambda_1 t)^{\alpha_1} \exp(\rho \beta d)\} \end{aligned} \tag{A.21}$$

For the item irradiated by Ra-226 died at time t and the cause of death is other diseases, its crude density is

$$\begin{aligned} f_{R1} &= f_1(t) * S_2(t) \\ &= \alpha_1 \lambda_1 (\lambda_1 t)^{\alpha_1 - 1} \exp(\beta d) \exp\{-(\lambda_1 t)^{\alpha_1} \exp(\beta d)\} \\ &\quad * \exp\{-(\lambda_2 t)^{\alpha_2} \end{aligned} \tag{A.22}$$

For the item irradiated by Ra-226 died at time t and the cause of death is not ostiosarcoma, its crude density is

$$\begin{aligned} f_{R2}(t) &= f_2(t) S_1(t) \\ &= \alpha_2 \lambda_2 (\lambda_2 t)^{\alpha_2 - 1} \exp\{-(\lambda_2 t)^{\alpha_2}\} \\ &\quad * \alpha_1 \lambda_1 (\lambda_1 t)^{\alpha_1 - 1} \exp(\beta d) \exp\{-(\lambda_1 t)^{\alpha_1} \exp(\beta d)\} \end{aligned} \tag{A.23}$$

3. Crude Density for Dependent Competing Risk Model

Joint density of Dependent Competing Risk Model

Joint density for FGM model

$$f(t_1, t_2) = f_1(t_1) f_2(t_2) \{1 + \delta [1 - 2F_1(t_1)][1 - 2F_2(t_2)]\} \tag{A.24}$$

Crude Density of Dependent Competing Risk Model

For the cause of death is osteosarcoma, let t1 = t, the crude density is

$$f_{C\_1}(t) = \int_t^\infty f(t_1, t_2)\, dt_2$$

$$= \int_t^\infty f_1(t_1) f_2(t_2) \{1 + \delta\, [1 - 2F_1(t_1)][1 - 2F_2(t_2)]\}\quad dt_2$$

$$= f_1(t) \int_t^\infty f_2(t_2) \{1 + \delta\, [1 - 2F_1(t_1)][1 - 2F_2(t_2)]\}\, dt_2$$

$$= f_1(t) \int_t^\infty f_2(t_2)\, \{1 + \delta\ [1 - 2F_2(t_2) - 2F_1(t\ ) + 4F_1(t) F_2(t_2)]\}\, dt_2$$

$$= f_1(t) \int_t^\infty f_2(t_2) \{1 + \delta\ -2\delta\, F_2(t_2) - 2\delta\, F_1(t) + 4\delta\, F_1(t) F_2(t_2)\}\, dt_2$$

$$= f_1(t) \{ \int_t^\infty (1 + \delta\ ) f_2(t_2) dt_2 - 2\delta \int_t^\infty f_2(t_2) F_2(t_2)\, dt_2 - 2\delta\, F_1(t) \int_t^\infty f_2(t_2) dt_2$$

$$+ 4\delta\, F_1(t) \int_t^\infty f_2(t_2) F_2(t_2)\, d\, t_2 \} \qquad \text{(A.25)}$$

where

$$\int_t^\infty (1 + \delta\ ) f_2(t_2) d\, t_2 = (1 + \delta\ ) S_2(t) \qquad \text{(A.26)}$$

$$2\delta\, F_1(t) \int_t^\infty f_2(t_2)\, d\, t_2 = 2\delta\, F_1(t) S_2(t)\,. \qquad \text{(A.27)}$$

Since:

$$\int_t^{\infty} f_2(t_2)F_2(t_2)\,d\,t_2 = \int_t^{\infty} F_2(t_2)\,dF_2(t_2)$$

$$= \frac{1}{2}F_2{}^2(t_2)\Big|_t^{\infty}$$

$$= \frac{1}{2}[F_2{}^2(\infty) - F_2{}^2(t)] \qquad \text{(A.28)}$$

$$= \frac{1}{2}[1 - F_2{}^2(t)]$$

$$= \frac{1}{2}S_2(t)\,[\,1 + F_2(t)\,]$$

(here we use $S_2(t) = 1 - F_2(t)$ and $F(\infty) = 1$)

and

$$-2\delta \int_t^{\infty} f_2(t_2)F(t_2)\,dt_2 + 4\delta\, F_1(t)\int_t^{\infty} f_2(t_2)F_2(t_2)\,dt_2$$

$$= [-2\delta + 4\delta\, F_1(t\ )] * \frac{1}{2}S_2(t)[1 + F_2(t\ )] \qquad \text{(A.29)}$$

$$= \delta\,[-1 + 2F_1(t)] * S_2(t)\,[1 + F_2(t\ )]$$

substitute equation(A.26), (A.27), and (A.29) into equation(A.25) , we obtain

Crude density in FGM model for the data with the osteosarcoma as

$$\begin{aligned} f_{C\_1}(t) &= f_1(t) * \{(1+\delta\ )S_2(t) - 2\delta\ F_1(t)S_2(t) + \delta\,[-1 + 2F_1(t)] * S_2(t)\,[1 + F_2(t\ )]\} \\ &= f_1(t)\,S_2(t) * \{\,1 + \delta - 2\delta\,F_1(t) + \delta\,(-1 - F_2(t) + 2F_1(t) + 2F_1(t)F_2(t)\,)\} \\ &= f_1(t)\,S_2(t) * \{1 + \delta - 2\delta\,F_1(t) - \delta - \delta\,F_2(t) + 2\delta\,F_1(t) + 2\delta\,F_1(t)F_2(t)\,\} \\ &= f_1(t)\,S_2(t) * [1 - \delta\,F_2(t) + 2\delta\,F_1(t)F_2(t)] \\ &= f_1(t)\,S_2(t) * \{1 - \delta\,F_2(t)\,[1 - 2F_1(t)]\} \\ &= f_1(t)\,S_2(t) * \{1 - \delta\,F_2(t)\,[S_1(t) - F_1(t)]\,\} \end{aligned} \qquad \text{(A.30)}$$

By the same derivation, the crude density for data with other diseases is:

$$f_{C\_2} = f_2(t)S_1(t) * \{1 - \delta\,F_1(t)\,[S_2(t) - F_2(t)]\} \qquad \text{(A.31)}$$

According equation(A.1) to (A.16), the crude density for Pu_data with osteosarcoma is

$$
\begin{aligned}
f_{PC\_1}(t) &= f_1(t)\, S_2(t) * \{1-\delta\,[\,1-S_2(t)][\,2S_1(t)-1]\,\} \\
&= \alpha_1\lambda_1(\lambda_1 t)^{\alpha_1-1}\exp(\rho\beta d)\exp\{-(\lambda_1 t)^{\alpha_1}\exp(\rho\beta d) \\
&\quad * \ \exp\{-(\lambda_2 t)^{\alpha_2}\} \\
&\quad * \{1-\delta\,[1-\exp\{-(\lambda_2 t)^{\alpha_2}\}] * [2\exp\{-(\lambda_1 t)^{\alpha_1}\exp(\rho\beta d)\}-1]\}
\end{aligned}
\tag{A.32}
$$

the crude density for Pu_data without osteosarcoma is

$$
\begin{aligned}
f_{PC\_2}(t) &= f_2(t)\, S_1(t) * \{1-\delta\,[\,1-S_1(t)][\,2S_2(t)-1]\,\} \\
&= \alpha_2\lambda_2(\lambda_2 t)^{\alpha_2-1}\exp\{-(\lambda_2 t)^{\alpha_2} * \exp\{-(\lambda_1 t)^{\alpha_1}\exp(\rho\beta d)\} \\
&\quad * \{1-\delta\ [1-\exp\{-(\lambda_2 t)^{\alpha_2}][2\exp\{-(\lambda_1 t)^{\alpha_1}\exp(\rho\beta d)\}-1]\}
\end{aligned}
\tag{A.33}
$$

the crude density for Ra_data with osteosarcoma is

$$
\begin{aligned}
f_{RC\_1}(t) &= f_1(t)\, S_2(t) * \{1-\delta\,[\,1-S_2(t)][\,2S_1(t)-1]\,\} \\
&= \alpha_1\lambda_1(\lambda_1 t)^{\alpha_1-1}\exp(\beta d)\exp\{-(\lambda_1 t)^{\alpha_1}\exp(\beta d)\} \\
&\quad * \ \exp\{-(\lambda_2 t)^{\alpha_2}\} \\
&\quad * \{1-\delta\,[1-\exp\{-(\lambda_2 t)^{\alpha_2}\}] * [2\exp\{-(\lambda_1 t)^{\alpha_1}\exp(\beta d)\}-1]\}
\end{aligned}
\tag{A.34}
$$

the crude density for Ra_data without osteosarcoma is

$$
\begin{aligned}
f_{RC\_2}(t) &= f_2(t)\, S_1(t) * \{1-\delta\,[\,1-S_1(t)][\,2S_2(t)-1]\,\} \\
&= \alpha_2\lambda_2(\lambda_2 t)^{\alpha_2-1}\exp\{-(\lambda_2 t)^{\alpha_2}\} * \exp\{-(\lambda_1 t)^{\alpha_1}\exp(\beta d)\} \\
&\quad * \{1-\delta\ [1-\exp\{-(\lambda_1 t)^{\alpha_1}\exp(\beta d)\}][2\exp\{-(\lambda_2 t)^{\alpha_2}-1]\}
\end{aligned}
\tag{A.35}
$$

APPENDIX B

COMPUTER CODE

```
// Pu_Cancer.h
#ifndef PU_CANCER_H
#define PU_CANCER_H
class Pu_Cancer{
protected:
                float time;
                float activity;
                float alpha1,lambda1, beta, rou;
                float alpha2, lambda2, delta;
public :
Pu_Cancer(float t, float d, float a,float b, float c, float g,
                                        float l, float m, float n );
virtual float cancer_hazard_rate(float t, float d,float a,float b, float c, float g );
          float cancer_cumu_hazard (float t, float d,float a,float b,  float c, float g);
          float cancer_survival(float t, float d, float a,float b, float c, float g);
          float cancer_failure(float t, float d, float a,float b, float c, float g);
          float non_hazard_rate(float t, float d,float l, float m);
          float non_cumu_hazard (float t, float d,float l, float m);
          float non_survival(float t, float d, float l, float m);
          float non_failure(float t, float d, float l, float m);
          virtual float crude_density(float t, float d, float a,float b,float c, float g,float l,
float m, float n );
};
#endif
**************************************************************************
// Pu_Non_Cancer.h
#ifndef PU_NON_CANCER_H
#define PU_NON_CANCER_H
#include"Pu_Cancer.h"
class Pu_Non_Cancer : public Pu_Cancer {
public:
                Pu_Non_Cancer(float t, float d, float a,float b,
                        float c, float g,float l, float m, float n )
                        :Pu_Cancer(t,d,a,b,c,g,l,m,n)
                {
                        time=t; activity=d;   alpha1=a;   lambda1=b;
        beta=c; rou=g; alpha2=l; lambda2=m; delta=n;
                }
        virtual float crude_density(float t, float d,float a,float b,
                                float c, float g,float l, float m, float n);
} ;
#endif
**************************************************************************
// Ra_Cancer.h
#ifndef RA_CANCER_H
```

```
#define RA_CANCER_H
#include"Pu_Cancer.h"
class Ra_Cancer:public Pu_Cancer{
public:
      Ra_Cancer(float t, float d, float a,float b, float c, float g,
                                    float l, float m, float n ):Pu_Cancer(t,d,a,b,c,g,l,m,n)
                                    {
                              time=t; activity=d;   alpha1=a;   lambda1=b;
                  beta=c; rou=g; alpha2=l; lambda2=m; delta=n;
       };
  virtual float cancer_hazard_rate(float t, float d, float a, float b,
                                                          float c, float g);
};
#endif
*************************************************************************
// Ra_Non_Cancer.h
#ifndef RA_NON_CANCER_H
#define RA_NON_CANCER_H
#include"Ra_Cancer.h"
class Ra_Non_Cancer:public Ra_Cancer{
public:
                Ra_Non_Cancer(float t, float d, float a, float b, float c,
                        float g, float l, float m, float n):
                                Ra_Cancer(t,d,a,b,c,g,l,m,n){
                   time=t; activity=d;   alpha1=a;   lambda1=b;
                   beta=c; rou=g; alpha2=l; lambda2=m; delta=n;
        };
        virtual float crude_density( float t, float d, float a,
                float b, float c,float g, float l, float m, float n);
};
#endif
*************************************************************************
//source file name: Pu_Cancer.C
#include<math.h>
#include<stdlib.h>
#include<string.h>
#include"Pu_Cancer.h"
Pu_Cancer::Pu_Cancer(float t, float d, float a,float b,
                        float c, float g,float l, float m, float n){
        time=t; activity=d; alpha1=a;lambda1=b;
        beta=c;rou=g; alpha2=l; lambda2=m;delta=n;
};
 float Pu_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                                                          float c, float g){
                        float Pu_C_h;
```

```
                    Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(g*c*d);
                    return Pu_C_h;
};
float Pu_Cancer::cancer_cumu_hazard(float t, float d, float a,float b,
                                                    float c, float g ){
        float Pu_C_H;
                    Pu_C_H=cancer_hazard_rate(t,d,a,b,c,g)*t/a;
                    return Pu_C_H;
};
float Pu_Cancer::cancer_survival(float t, float d, float a,float b,
                                                    float c, float g ){
                    float Pu_C_S;
                    Pu_C_S=exp(-cancer_cumu_hazard(t,d,a,b,c,g)*exp(-20));
                    return Pu_C_S;
};
float Pu_Cancer::cancer_failure(float t, float d, float a,float b,
                                                    float c, float g){
                    float Pu_C_f;
                    Pu_C_f=cancer_hazard_rate(t,d,a,b,c,g)*
                            cancer_survival(t,d,a,b,c,g);
                    return Pu_C_f;
};
float Pu_Cancer::non_hazard_rate(float t, float d,float l, float m ){
                    float Pu_NC_h;
                    Pu_NC_h=l*pow(m,l)*pow(t,l-1);
                    return Pu_NC_h;
};
float Pu_Cancer::non_cumu_hazard(float t, float d,float l, float m ){
                    float Pu_NC_H;
                    Pu_NC_H=non_hazard_rate(t,d,l,m)*t/l;
                    return Pu_NC_H;
};
float Pu_Cancer::non_survival(float t, float d,float l, float m ){
                    float Pu_NC_S;
                    Pu_NC_S=exp(-non_cumu_hazard(t,d,l,m));
                    return Pu_NC_S;
};
float Pu_Cancer::non_failure(float t, float d,float l, float m ){
                    float Pu_NC_f;
                    Pu_NC_f=non_hazard_rate(t,d,l,m)*non_survival(t,d,l,m);
                    return Pu_NC_f;
};
float Pu_Cancer :: crude_density(float t, float d,float a,float b,
                    float c, float g,float l, float m, float n ){
        float Pu_C_Density;
```

```
            Pu_C_Density =cancer_failure(t,d,a,b,c,g)*non_survival(t,d,l,m)*
            (1-n*(1-non_survival(t,d,l,m))*(2*cancer_survival(t,d,a,b,c,g)-1)
        return  Pu_C_Density;
}
*************************************************************************
//source file name: Pu_Non_Cancer.C
#include<math.h>
#include<stdlib.h>
#include<string.h>
#include"Pu_Cancer.h"
#include"Pu_Non_Cancer.h"
float Pu_Non_Cancer :: crude_density(float t, float d,float a,float b,
                        float c, float g,float l, float m, float n ){
        float Pu_NC_Density;
Pu_NC_Density =non_failure(t,d,l,m)*cancer_survival(t,d,a,b,c,g)*
 (1-n*(1-cancer_survival(t,d,a,b,c,g))*(2*non_survival(t,d,l,m)-1))
        return  Pu_NC_Density;
 }
*************************************************************************
//source file name: Ra_Cancer.C
#include<math.h>
#include<stdlib.h>
#include<string.h>
#include"Pu_Cancer.h"
#include"Ra_Cancer.h"
float Ra_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                                                    float c, float g){
                        float Pu_C_h;
                        g/g;
                        Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(c*d);
                        return Pu_C_h;
};
*************************************************************************
//source file name: Ra_Non_Cancer.C
#include<math.h>
#include<stdlib.h>
#include<string.h>
#include"Ra_Cancer.h"
#include"Ra_Non_Cancer.h"
float Ra_Non_Cancer :: crude_density(float t, float d,float a,float b,
                        float c, float g,float l, float m, float n ){
        float Ra_NC_Density;
        g/g;
Ra_NC_Density =non_failure(t,d,l,m)*cancer_survival(t,d,a,b,c,g)*
(1-n*(1-cancer_survival(t,d,a,b,c,g))*(2*non_survival(t,d,l,m)-1))
```

```
return Ra_NC_Density;
};
**************************************************************************
//source file name: pr23.C
// This is the explain file for all of function in dependent model
#include<math.h>
#include<stdlib.h>
#include<string.h>
#include"Pu_Cancer.h"
#include"Pu_Non_Cancer.h"
#include"Ra_Cancer.h"
#include"Ra_Non_Cancer.h"
Pu_Cancer::Pu_Cancer(float t, float d, float a,float b,
                     float c, float g,float l, float m, float n){
        time=t; activity=d; alpha1=a;lambda1=b;
        beta=c;rou=g; alpha2=l; lambda2=m;delta=n;
};
float Pu_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                                                    float c, float g){
                     float Pu_C_h;
                     Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(g*c*d);
                     return Pu_C_h;
};
float Pu_Cancer::cancer_cumu_hazard(float t, float d, float a,float b,
                                                    float c, float g ){
        float Pu_C_H;
                     Pu_C_H=cancer_hazard_rate(t,d,a,b,c,g)*t/a;
                     return Pu_C_H;
};
float Pu_Cancer::cancer_survival(float t, float d, float a,float b,
                                                    float c, float g ){
                     float Pu_C_S;
               Pu_C_S=exp(-cancer_cumu_hazard(t,d,a,b,c,g)*exp(-20));
                     return Pu_C_S;
};
float Pu_Cancer::cancer_failure(float t, float d, float a,float b,
                                                    float c, float g){
                     float Pu_C_f;
                     Pu_C_f=cancer_hazard_rate(t,d,a,b,c,g)*
                               cancer_survival(t,d,a,b,c,g);
                     return Pu_C_f;
};
float Pu_Cancer::non_hazard_rate(float t, float d,float l, float m ){
                     float Pu_NC_h;
                     Pu_NC_h=l*pow(m,l)*pow(t,l-1);
```

```
                    return Pu_NC_h;
};
float Pu_Cancer::non_cumu_hazard(float t, float d,float l, float m ){
                    float Pu_NC_H;
                    Pu_NC_H=non_hazard_rate(t,d,l,m)*t/l;
                    return Pu_NC_H;
};
float Pu_Cancer::non_survival(float t, float d,float l, float m ){
                    float Pu_NC_S;
                    Pu_NC_S=exp(-non_cumu_hazard(t,d,l,m));
                    return Pu_NC_S;
};
float Pu_Cancer::non_failure(float t, float d,float l, float m ){
                    float Pu_NC_f;
                    Pu_NC_f=non_hazard_rate(t,d,l,m)*non_survival(t,d,l,m);
                    return Pu_NC_f;
};
float Pu_Cancer :: crude_density(float t, float d,float a,float b,
                    float c, float g,float l, float m, float n ){
        float Pu_C_Density;
Pu_C_Density =cancer_failure(t,d,a,b,c,g)*non_survival(t,d,l,m)*
(1-n*(1-non_survival(t,d,l,m))*(2*cancer_survival(t,d,a,b,c,g)-1));
return  Pu_C_Density;
}
float Pu_Non_Cancer :: crude_density(float t, float d,float a,float b,
                    float c, float g,float l, float m, float n ){
        float Pu_NC_Density;

Pu_NC_Density =non_failure(t,d,l,m)*cancer_survival(t,d,a,b,c,g)*
(1-n*(1-cancer_survival(t,d,a,b,c,g))*(2*non_survival(t,d,l,m)-1));
return  Pu_NC_Density;
}
float Ra_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                                                    float c, float g){
                    float Pu_C_h;
                    g/g;
                    Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(c*d);
                    return Pu_C_h;
};
float Ra_Non_Cancer :: crude_density(float t, float d,float a,float b,
                    float c, float g,float l, float m, float n ){
        float Ra_NC_Density;
        g/g;
Ra_NC_Density =non_failure(t,d,l,m)*cancer_survival(t,d,a,b,c,g)*
(1-n*(1-cancer_survival(t,d,a,b,c,g))*(2*non_survival(t,d,l,m)-1));
```

```
                    return Ra_NC_Density;
};

*************************************************************************
// Title: dep_Rho_Mont.C
//input:    data:   pu_can.C; pu_non.C;  ra_can.C; ra_non.C;
//          include:  Pu_cancer.h, Pu-Non_Cancer.h, Ra_Cancer.h,
//                    Ra-Non_Cancer.h
//output:   marginal density of toxicity ratio rho
//description:
//          Dependent Competing Risk Model
//          Monte Carlo Method
#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
#include"Pu_Cancer.h"
#include"Pu_Non_Cancer.h"
#include"Ra_Cancer.h"
#include"Ra_Non_Cancer.h"
// time=t; activity=d;  alpha1=a;  lambda1=b;
// beta=c; rho=g; alpha2=l; lambda2=m; delta=n;
main()
{  int  tt, i, j, k, num_rand, rr, qq1, qq2, qq3, qq4, qq5, qq6;
   int  pp1,pp2,pp3,pp4,pp5,pp6;
   float mm, nn, total;
   float t,d, total1, total2,total3, total4, sum;
   float a,b,c,g,l,m,n;
   float ini_g, increa_g, qq;
   int  num_g;
   unsigned int seed;
        ofstream fout("dep_rho_monte_dat");
        Pu_Cancer *p;
        cout<<"\nEnter a positive integer as seed:";
        cin>>seed;
        srand(seed);
        cout<<"\nEnter a integer rr as number of random parameter vectors\n";
        cin>>rr;
        cout<<"\nEnter two integers as scale factor\n";
        cout<<"of t and d: (t/mm, d/nn)\n";
        cin>>mm>>nn;
        cout<<"\nEnter the initial value and increament(float) and the
          number of points(int) for g\n";
        cin>>ini_g>>increa_g>>num_g;
        cout<<"\nEnter the 6 integers (qq) for ranges of parameters :"<<"\n";
```

```
cin>>qq1 >>qq2>>qq3>>qq4>>qq5>>qq6;
cout<<"\nEnter the 6 integers (pp) for ranges of parameters :"<<"\n";
cin>>pp1>> pp 2>> pp 3>> pp 4>> pp5>> pp6;
 g=ini_g;
for (k=0; k<num_g; k++){
        num_rand = 0;
        sum=0.0;
for(j=0; j<rr; j++){
        a = qq1*float(rand())/RAND_MAX+pp1;
        b = qq2*float(rand())/RAND_MAX+pp2;
        c = qq3*float(rand())/RAND_MAX+pp3;
        l = qq4*float(rand())/RAND_MAX+pp4;
        m = qq5*float(rand())/RAND_MAX+pp5;
        n = qq6*float(rand())/RAND_MAX+pp6;
        // in "limits.h" file, the RAND_MAX=INT_MAX=32767
        // to get random number between [0, 1].
ifstream fin1("pu_can.C");
    ifstream fin2("pu_non.C");
    ifstream fin3("ra_can.C");
    ifstream fin4("ra_non.C");
total1=0.0;
        for(i=0; i<71; i++){
                fin1>>d>>tt;
                t=tt/mm;   d=d/nn;
                Pu_Cancer  PC(t,d,a,b,c,g,l,m,n);
                p=&PC;
                total1 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
        total2=0.0;
        for(i=0; i<258; i++){
                fin2>>d>>tt;
                t=tt/mm;   d=d/nn;
                Pu_Non_Cancer PNC(t,d,a,b,c,g,l,m,n);
                p=&PNC;
                total2 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
         total3=0.0;
        for(i=0; i<53; i++){
                fin3>>d>>tt;
                t=tt/mm;   d=d/nn;
                Ra_Cancer  RC(t,d,a,b,c,g,l,m,n);
                p=&RC;
                total3 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
         total4=0.0;
```

```
                for(i=0; i<132; i++){
                        fin4>>d>>tt;
                        t=tt/mm;   d=d/nn;
                        Ra_Non_Cancer  RNC(t,d,a,b,c,g,l,m,n);
                        p=&RNC;
                        total4 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
                }
              total=total1+total2+total3+total4;
                sum += exp(total/rr);
                num_rand += 1;
        fin1.close();
            fin2.close();
            fin3.close();
            fin4.close();
                }
         fout<<g<<" "<<sum/num_rand<<"\n";
cout<<"\n\n\n g=  "<<rho <<"   Sum=  "<<     sum/num_rand<<"\n\n"<<endl;
        g+=increa_g;
        }
        fout.close();
        return 0;
}
*************************************************************************//
Title: dep_rho_Lattice.C
//input:     data:    pu_can.C; pu_non.C;  ra_can.C; ra_non.C;
//          include:  Pu_cancer.h, Pu-Non_Cancer.h, Ra_Cancer.h,
//                    Ra-Non_Cancer.h
//output:    marginal density of toxicity ratio rho
//description:
//          Dependent Competing Risk Model
//          Lattice Rule
#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
#include"Pu_Cancer.h"
#include"Pu_Non_Cancer.h"
#include"Ra_Cancer.h"
#include"Ra_Non_Cancer.h"
// time=t; activity=d;   alpha1=a;   lambda1=b;
// beta=c; rho=g; alpha2=l; lambda2=m; delta=n;
main()
{  int   tt, i, j, k, num_rand, rr, qq1, qq2, qq3, qq4, qq5, qq6;
   int   pp1, pp2, pp3, pp4, pp5, pp6;
  float  mm, nn, total;
```

```
float  t,d, total1, total2,total3, total4, sum;
float  a,b,c,g,l,m,n;
float  ini_g, increa_g, qq;
int    num_g;
       ofstream fout("dep_rho_Lattice_dat");
       Pu_Cancer *p;
       cout<<"\nEnter two integers as scale factor\n";
       cout<<"of t and d: (t/mm, d/nn)\n";
       cin>>mm>>nn;
       cout<<"\nEnter the initial value and increament(float) and the
         number of points(int) for g\n";
       cin>>ini_g>>increa_g>>num_g;
      cout<<"\nEnter the 6 integers (qq) for ranges of parameters :"<<"\n";
       cin>>qq1 >>qq2>>qq3>>qq4>>qq5>>qq6;
       cout<<"\nEnter the 6 integers (pp) for ranges of parameters :"<<"\n";
       cin>>pp1>> pp 2>> pp 3>> pp 4>> pp5>> pp6;
//Lattice Rule
       int LS=6;
       cout<<"\n Enter two integers for N and L \n";
       cin>>LN>>LL;
int Za, Zb, Zc, Zl, Zm, Zn;
 Za = 1;
 Zb = LL;
 Zc = int(LL**2.0) % LN;
 Zl = int(LL**3.0) % LN;
 Zm = int(LL**4.0) % LN;
 Zn = int(LL**5.0) % LN;
       g=ini_g;
      for (k=0; k<num_g; k++){
              sum=0.0;
        for(j=0; j<LN; j++){
              a = qq1*j/LN*1.0*Za+pp1;
              b = qq2*j/LN*1.0*Za+pp2;
              c = qq3*j/LN*1.0*Za+pp3;
              l = qq4*j/LN*1.0*Za+pp4;
              m = qq5*j/LN*1.0*Za+pp5;
              n = qq6*j/LN*1.0*Za+pp6;
      ifstream fin1("pu_can.C");
          ifstream fin2("pu_non.C");
          ifstream fin3("ra_can.C");
          ifstream fin4("ra_non.C");
      total1=0.0;
              for(i=0; i<71; i++){
                      fin1>>d>>tt;
                      t=tt/mm;   d=d/nn;
```

```
                Pu_Cancer PC(t,d,a,b,c,g,l,m,n);
                    p=&PC;
                    total1 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
                }
                total2=0.0;
                for(i=0; i<258; i++){
                        fin2>>d>>tt;
                        t=tt/mm;  d=d/nn;
                        Pu_Non_Cancer PNC(t,d,a,b,c,g,l,m,n);
                        p=&PNC;
                        total2 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
                }
                 total3=0.0;
                for(i=0; i<53; i++){
                        fin3>>d>>tt;
                        t=tt/mm;  d=d/nn;
                        Ra_Cancer RC(t,d,a,b,c,g,l,m,n);
                        p=&RC;
                        total3 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
                }
                 total4=0.0;
                for(i=0; i<132; i++){
                        fin4>>d>>tt;
                        t=tt/mm;  d=d/nn;
                        Ra_Non_Cancer RNC(t,d,a,b,c,g,l,m,n);
                        p=&RNC;
                        total4 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
                }
              total=total1+total2+total3+total4;
                sum += exp(total);
        j += 1;
           fin1.close();
           fin2.close();
           fin3.close();
           fin4.close();
                } // end loop of j
        fout<<g<<" "<<sum/LN<<"\n";
        cout<<"\n\n\n g=  "<<g  <<"   Sum=  "<< sum/LN <<"\n\n"<<endl;
        g += increa_g;
        }
        fout.close();
        return 0;
}
           fin1.close();
           fin2.close();
```

```
            fin3.close();
            fin4.close();
                }
        fout<<g<<" "<<sum/num_rand<<"\n";
        cout<<"\n\n\n g=  "<<g  <<"   Sum=  "<< sum/num_rand<<"\n\n"<<endl;
        g+=increa_g;
        }
        fout.close();
        return 0;
}
*************************************************************************
// Title: dep_rho_Gaus.C
//input:     data:    pu_can.C; pu_non.C;  ra_can.C; ra_non.C; Gauss.C
//          include:  Pu_cancer.h, Pu-Non_Cancer.h, Ra_Cancer.h,
//                    Ra-Non_Cancer.h
//output:    marginal density of toxicity ratio rho
//description:
//          Dependent Competing Risk Model
//          Gaussian Quadrature
#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
#include"Pu_Cancer.h"
#include"Pu_Non_Cancer.h"
#include"Ra_Cancer.h"
#include"Ra_Non_Cancer.h"
main()
{  int   tt, i, k, num_rand, rr;
   int   j, jj, jjj, jjjj, jjjjj, jjjjjj
   float mm, nn, total;
   float t, d, total1, total2,total3, total4, sum;
   float a, b, c, g, l, m, n;
   float wa, wb, wc, wl, wm, wn, ww;
   float ini_g, increa_g, qq;
   int   num_g;
   ofstream fout("dep_rho_Gaus_dat");
        Pu_Cancer *p;
    cout<<"\nEnter two integers as scale factor\n";
        cout<<"of t and d: (t/mm, d/nn)\n";
        cin>>mm>>nn;
        cout<<"\nEnter the initial value and increament(float) and the
          number of points(int) for g\n";
        cin>>ini_g>>increa_g>>num_g;
         g=ini_g;
```

```
 for (k=0; k<num_g; k++){
        sum=0.0;
ifstream fin("Gauss.C");
   for(j=0; j<8; j++){
  fin>>a>>wa;
 for(jj=0; jj<8; j++){
   fin>>b>>wb;
  for(jjj=0; jjj<8; j++){
    fin>>c>>wc;
   for(jjjj=0; jjjj<8; j++){
     fin>>l>>wl;
    for(jjjjj=0; jjjjj<8; j++){
      fin>>m>>wm;
     for(jjjjjj=0; jjjjjj<8; j++){
       fin>>n>>wn;
      ww = wa*wb*wc*wl*wm*wn;
 ifstream fin1("pu_can.C");
     ifstream fin2("pu_non.C");
     ifstream fin3("ra_can.C");
     ifstream fin4("ra_non.C");
 total1=0.0;
        for(i=0; i<71; i++){
               fin1>>d>>tt;
               t=tt/mm;  d=d/nn;
               Pu_Cancer  PC(t,d,a,b,c,g,l,m,n);
               p=&PC;
               total1 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
        total2=0.0;
        for(i=0; i<258; i++){
               fin2>>d>>tt;
               t=tt/mm;  d=d/nn;
               Pu_Non_Cancer PNC(t,d,a,b,c,g,l,m,n);
               p=&PNC;
               total2 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
         total3=0.0;
        for(i=0; i<53; i++){
               fin3>>d>>tt;
               t=tt/mm;  d=d/nn;
               Ra_Cancer  RC(t,d,a,b,c,g,l,m,n);
               p=&RC;
               total3 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
         total4=0.0;
```

```
                for(i=0; i<132; i++){
                        fin4>>d>>tt;
                        t=tt/mm;  d=d/nn;
                        Ra_Non_Cancer  RNC(t,d,a,b,c,g,l,m,n);
                        p=&RNC;
                        total4 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
                }
                total=log(ww)+(total1+total2+total3+total4);
                sum += tatol;
                        jjjjjj += 1;
                         fin1.close();
                             fin2.close();
                             fin3.close();
                             fin4.close();
                         } //end loop of jjjjjj
                        jjjjj += 1;
                        } //end loop of jjjjj
          jjjj += 1;
          }     //end loop of jjjj
                    jjj += 1;
                     } //end loop of jjj
                   jj += 1;
                   } //end loop of jj
                  j += 1;
                  } //end loop of j
                    sum = exp(sum);
        fout<<g<<" "<<sum <<"\n";
        cout<<"\n\n\n g=  "<<g  <<"   Sum=  "<< sum <<"\n\n"<<endl;
        g += increa_g;
        }
        fout.close();
        return 0;
}
*************************************************************************
//      Title:  ind_rho_mont.C
// input:       pu_can.C; pu_non.C; ra_can.C; ra_non.C
// output:      marginal density of toxicity ratio rho
// description:  independent competing risk model
//            Monte carlo Method
#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
class Pu_Cancer{
```

```
protected:
                float time;
                float activity;
                float alpha1,lambda1, beta, rho;
public :
Pu_Cancer(float t, float d, float a, float b, float c, float g);
virtual float cancer_hazard_rate(float t, float d,float a,float b,
                                                        float c, float g );
        float cancer_cumu_hazard (float t, float d,float a,float b,
                                                        float c, float g);
        float cancer_survival(float t, float d, float a,float b,
                                                        float c, float g);
        float cancer_failure(float t, float d, float a,float b,
                                                        float c, float g);
        virtual float crude_density(float t, float d, float a,float b,
                                        float c, float g);
};
class Pu_Non_Cancer : public Pu_Cancer {
public:
                Pu_Non_Cancer(float t, float d, float a,float b,
                        float c, float g ):Pu_Cancer(t,d,a,b,c,g)
                {       time=t; activity=d;  alpha1=a;  lambda1=b;
            beta=c; rou=g;
                }
        virtual float crude_density(float t, float d,float a,float b,
                                float c, float g);
} ;
class Ra_Cancer:public Pu_Cancer{
public:
      Ra_Cancer(float t, float d, float a,float b, float c,
            float g):Pu_Cancer(t,d,a,b,c,g)
                        {time=t; activity=d;  alpha1=a;  lambda1=b;
                          beta=c; rou=g;  };
 virtual float cancer_hazard_rate(float t, float d, float a, float b,
                                                        float c, float g);
};
class Ra_Non_Cancer:public Ra_Cancer{
public:
        Ra_Non_Cancer(float t, float d, float a, float b, float c,
                        float g):Ra_Cancer(t,d,a,b,c,g){
                time=t; activity=d;  alpha1=a;  lambda1=b;
                        beta=c; rou=g;};
        virtual float crude_density( float t, float d, float a,
                        float b, float c,float g);
};
```

```
Pu_Cancer::Pu_Cancer(float t,float d, float a,float b,float c,float g){
        time=t; activity=d; alpha1=a;lambda1=b;
        beta=c;rou=g;
};
float Pu_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                        float c, float g)
{                       float Pu_C_h;
                        Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(g*c*d);
                        return Pu_C_h;
};
float Pu_Cancer::cancer_cumu_hazard(float t, float d, float a,float b,
                                                        float c, float g ){
            float Pu_C_H;
                        Pu_C_H=cancer_hazard_rate(t,d,a,b,c,g)*t/a;
                        return Pu_C_H;
};
float Pu_Cancer::cancer_survival(float t, float d, float a,float b,
                                                        float c, float g ){
                        float Pu_C_S;
                Pu_C_S=exp(-cancer_cumu_hazard(t,d,a,b,c,g)*exp(-20));
                        return Pu_C_S;
};
float Pu_Cancer::cancer_failure(float t, float d, float a,float b,
                                                        float c, float g){
                        float Pu_C_f;
                        Pu_C_f=cancer_hazard_rate(t,d,a,b,c,g)*
                                cancer_survival(t,d,a,b,c,g);
                        return Pu_C_f;
};
float Pu_Cancer :: crude_density(float t, float d,float a,float b,
                        float c, float g ){
            float Pu_C_Density;
                    Pu_C_Density =cancer_failure(t,d,a,b,c,g);
                        return  Pu_C_Density;
}
float Pu_Non_Cancer :: crude_density(float t, float d,float a,float b,
                        float c, float g ){
            float Pu_NC_Density;
                    Pu_NC_Density =cancer_survival(t,d,a,b,c,g);
                        return  Pu_NC_Density;
}
float Ra_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                                                        float c, float g){
                        float Pu_C_h;
                        g/g;
```

```
                    Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(c*d);
                    return Pu_C_h;
};
float Ra_Non_Cancer :: crude_density(float t, float d,float a,float b,
                    float c, float g ){
        float Ra_NC_Density;
                Ra_NC_Density =cancer_survival(t,d,a,b,c,g);
                    return  Ra_NC_Density;
};
main(){
  int  tt, i, j, k, num_rand, rr, qq1, qq2, qq3;
  int  pp1, pp2, pp3;
  float mm, nn, total;
  float t,d, total1, total2,total3, total4, sum;
  float a,b,c,g,qq;
  float ini_g, increa_g;
  int   num_g;
  unsigned int seed;
        ofstream fout("ind_rho_mont_dat");
        Pu_Cancer *p;
        cout<<"\nEnter a positive integer as seed:";
        cin>>seed;
        srand(seed);
        cout<<"\nEnter a integer rr as number of random parameter vectors\n";
        cin>>rr;
        cout<<"\nEnter two integers as scale factor\n";
        cout<<"of t and d\n";
        cin>>mm>>nn;
        cout<<"\nEnter the initial value and increament(float) and the number
        of points(int) for g\n";
        cin>>ini_g>>increa_g>>num_g;
        cout<<"\nEnter the 3 integers (qq) for ranges of parameters :"<<"\n";
        cin>>qq1>>qq2>>qq3;
        cout<<"\nEnter the 4 integers (pp) for ranges of parameters :"<<"\n";
        cin>>pp1>> pp 2>> pp 3;
        g = ini_g;
        for (k=0; k<num_g; k++){
                num_rand = 0;
                sum=0.0;
        for(j=0; j<rr; j++){
                a = qq1*float(rand())/RAND_MAX+pp1;
                b = qq2*float(rand())/RAND_MAX+pp2;
                c = qq3*float(rand())/RAND_MAX+pp3;
                // in "limits.h" file, the RAND_MAX=INT_MAX=32767
                // to get random number between [0, 1].
```

```
ifstream fin1("pu_can.C");
    ifstream fin2("pu_non.C");
    ifstream fin3("ra_can.C");
    ifstream fin4("ra_non.C");
total1=0.0;
        for(i=0; i<71; i++){
                fin1>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Cancer  PC(t,d,a,b,c,g);
                p=&PC;
                total1 += log(p->crude_density(t,d,a,b,c,g));
        }
        total2=0.0;
        for(i=0; i<258; i++){
                fin2>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Non_Cancer PNC(t,d,a,b,c,g);
                p=&PNC;
                total2 += log(p->crude_density(t,d,a,b,c,g));
        }
         total3=0.0;
        for(i=0; i<53; i++){
                fin3>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Cancer  RC(t,d,a,b,c,g);
                p=&RC;
                total3 += log(p->crude_density(t,d,a,b,c,g));
        }
         total4=0.0;
        for(i=0; i<132; i++){
                fin4>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Non_Cancer  RNC(t,d,a,b,c,g);
                p=&RNC;
                total4 += log(p->crude_density(t,d,a,b,c,g));
        }
      total=total1+total2+total3+total4;
        sum += exp(total/1000.0);
        num_rand += 1;
    fin1.close();
    fin2.close();
    fin3.close();
    fin4.close();
}
fout<<g<<" "<<sum/num_rand<<"\n";
```

```
        cout<<"\n\n\n g=  "<<g  <<"   Sum=  "<< sum/num_rand<<"\n\n"<<endl;
        g += increa_g;
        }
        fout.close();
        return 0;
}
*********************************************************************
// Title: dep_alpha_1_Mont.C
//input:     data:    pu_can.C; pu_non.C;  ra_can.C; ra_non.C;
//          include:  Pu_cancer.h, Pu-Non_Cancer.h, Ra_Cancer.h,
//                    Ra-Non_Cancer.h
//output:    marginal density of alpha_1
//
//description:
//          Dependent Competing Risk Model
//          Monte Carlo Method
#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
#include"Pu_Cancer.h"
#include"Pu_Non_Cancer.h"
#include"Ra_Cancer.h"
#include"Ra_Non_Cancer.h"
// time=t; activity=d;   alpha1=a;   lambda1=b;
// beta=c; rou=g; alpha2=l; lambda2=m; delta=n;
main()
{
   int   tt, i, j, k, num_rand, rr;
   int qq1, qq2, qq3, qq4, qq5, qq6;
   int pp1, pp2, pp3, pp4, pp5, pp6;
   float mm, nn, total;
   float t,d, total1, total2,total3, total4, sum;
   float a,b,c,g,l,m,n;
   float ini_a, increa_a, qq;
   int   num_a;
   unsigned int seed;
        ofstream fout("dep_alpha_1_monte_dat");
        Pu_Cancer *p;
        cout<<"\nEnter a positive integer as seed:";
        cin>>seed;
        srand(seed);
        cout<<"\nEnter a integer rr as number of random
        parameter vectors\n";
        cin>>rr;
```

```
cout<<"\nEnter two integers as scale factor\n";
cout<<"of t and d: (t/mm, d/nn)\n";
cin>>mm>>nn;
cout<<"\nEnter the initial value and increament(float) and the
 number of points(int) for a\n";
cin>>ini_a>>increa_a>>num_a;
cout<<"\nEnter the 6 integers (qq) for ranges of parameters :"<<"\n";
cin>>qq1>>qq2>>qq3>>qq4>>qq5>>qq6;
cout<<"\nEnter the 6 integers (pp) for ranges of parameters :"<<"\n";
cin>>pp1>> pp 2>> pp 3>> pp 4>> pp5>> pp6;
a = ini_a;
for (k=0; k<num_a; k++){
        num_rand = 0;
        sum=0.0;
for(j=0; j<rr; j++){
        q = qq1*float(rand())/RAND_MAX+pp1;
        b = qq2*float(rand())/RAND_MAX+pp2;
        c = qq3*float(rand())/RAND_MAX+pp3;
        l = qq4*float(rand())/RAND_MAX+pp4;
        m = qq5*float(rand())/RAND_MAX+pp5;
        n = qq6*float(rand())/RAND_MAX+pp6;
        // in "limits.h" file, the RAND_MAX=INT_MAX=32767
        // to get random number between [0, 1].
ifstream fin1("pu_can.C");
    ifstream fin2("pu_non.C");
    ifstream fin3("ra_can.C");
    ifstream fin4("ra_non.C");
        total1=0.0;
        for(i=0; i<71; i++){
                fin1>>d>>tt;
                t=tt/mm;   d=d/nn;
                Pu_Cancer  PC(t,d,a,b,c,g,l,m,n);
                p=&PC;
                total1 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
        total2=0.0;
        for(i=0; i<258; i++){
                fin2>>d>>tt;
                t=tt/mm;   d=d/nn;
                Pu_Non_Cancer PNC(t,d,a,b,c,g,l,m,n);
                p=&PNC;
                total2 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
        total3=0.0;
        for(i=0; i<53; i++){
```

```
                    fin3>>d>>tt;
                    t=tt/mm;  d=d/nn;
                    Ra_Cancer  RC(t,d,a,b,c,g,l,m,n);
                    p=&RC;
                    total3 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
                }
                 total4=0.0;
                for(i=0; i<132; i++){
                    fin4>>d>>tt;
                    t=tt/mm;  d=d/nn;
                    Ra_Non_Cancer  RNC(t,d,a,b,c,g,l,m,n);
                    p=&RNC;
                    total4 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
                }
              total=total1+total2+total3+total4;
                 sum += exp(total/rr);
                num_rand += 1;
            fin1.close();
            fin2.close();
            fin3.close();
            fin4.close();
                }
        fout<<a<<" "<<sum/num_rand<<"\n";
        cout<<"\n\n\n a_1=  "<<a <<"
         Sum=  "<< sum/num_rand<<"\n\n"<<endl;
        a += increa_a;
        }
        fout.close();
        return 0;
}
*************************************************************************
//      Title:  ind_alpha_1_mont.C
// input:     pu_can.C; pu_non.C; ra_can.C; ra_non.C
// output:    marginal density of alpha_1
// description: independent competing risk model
//          Monte carlo Method
#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
class Pu_Cancer{
protected:
                float time;
                float activity;
                float alpha1,lambda1, beta, rho;
```

```
public :
Pu_Cancer(float t, float d, float a, float b, float c, float g);
        virtual float cancer_hazard_rate(float t, float d,float a,float b,
        float c, float g );
         float cancer_cumu_hazard (float t, float d,float a,float b,
                                        float c, float g);
          float cancer_survival(float t, float d, float a,float b,
                                        float c, float g);
          float cancer_failure(float t, float d, float a,float b,
                                        float c, float g);
          virtual float crude_density(float t, float d, float a,float b,
                                        float c, float g);
};
class Pu_Non_Cancer : public Pu_Cancer {
public:
                Pu_Non_Cancer(float t, float d, float a,float b,
                        float c, float g ):Pu_Cancer(t,d,a,b,c,g)
                {       time=t; activity=d;  alpha1=a;  lambda1=b;
                        beta=c; rou=g;
                }
        virtual float crude_density(float t, float d,float a,float b,
                                float c, float g);
} ;
class Ra_Cancer:public Pu_Cancer{
public:
      Ra_Cancer(float t, float d, float a,float b, float c,
            float g):Pu_Cancer(t,d,a,b,c,g)
                                { time=t; activity=d;  alpha1=a;  lambda1=b;
                              beta=c; rou=g;  };
  virtual float cancer_hazard_rate(float t, float d, float a, float b,
                                float c, float g);
};
class Ra_Non_Cancer:public Ra_Cancer{
public:
                Ra_Non_Cancer(float t, float d, float a, float b, float c,
                                        float g):Ra_Cancer(t,d,a,b,c,g){
                        time=t; activity=d;  alpha1=a;  lambda1=b;
                          beta=c; rou=g;};
          virtual float crude_density( float t, float d, float a,
                        float b, float c,float g);
};
Pu_Cancer::Pu_Cancer(float t,float d, float a,float b,float c,float g){
        time=t; activity=d; alpha1=a;lambda1=b;
        beta=c;rou=g;
};
```

```
float Pu_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                        float c, float g)
{                       float Pu_C_h;
                        Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(g*c*d);
                        return Pu_C_h;
};
float Pu_Cancer::cancer_cumu_hazard(float t, float d, float a,float b,
                                                        float c, float g ){
            float Pu_C_H;
                        Pu_C_H=cancer_hazard_rate(t,d,a,b,c,g)*t/a;
                        return Pu_C_H;
};
float Pu_Cancer::cancer_survival(float t, float d, float a,float b,
                                                        float c, float g ){
                        float Pu_C_S;
                Pu_C_S=exp(-cancer_cumu_hazard(t,d,a,b,c,g)*exp(-20));
                        return Pu_C_S;
};
float Pu_Cancer::cancer_failure(float t, float d, float a,float b,
                                                        float c, float g){
                        float Pu_C_f;
                        Pu_C_f=cancer_hazard_rate(t,d,a,b,c,g)*
                                cancer_survival(t,d,a,b,c,g);
                        return Pu_C_f;
};
float Pu_Cancer :: crude_density(float t, float d,float a,float b,
                        float c, float g ){
        float Pu_C_Density;
                Pu_C_Density =cancer_failure(t,d,a,b,c,g);
                        return  Pu_C_Density;
}
float Pu_Non_Cancer :: crude_density(float t, float d,float a,float b,
                        float c, float g ){
            float Pu_NC_Density;
                Pu_NC_Density =cancer_survival(t,d,a,b,c,g);
                        return  Pu_NC_Density;
}
float Ra_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                                                        float c, float g){
                        float Pu_C_h;
                        g/g;
                        Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(c*d);
                        return Pu_C_h;
};
float Ra_Non_Cancer :: crude_density(float t, float d, float a,float b,
```

```
                    float c, float g ){
        float Ra_NC_Density;
                Ra_NC_Density =cancer_survival(t,d,a,b,c,g);
                    return  Ra_NC_Density;
};
main(){
   int tt, i, j, k, num_rand, rr, qq1, qq2, qq3;
   int pp1, pp2, pp3;
   float mm, nn, total;
   float t,d, total1, total2,total3, total4, sum;
   float a,b,c,g,qq;
   float ini_a, increa_a;
   int  num_a;
   unsigned int seed;
        ofstream fout("ind_alpha_1_mont_dat");
        Pu_Cancer *p;
        cout<<"\nEnter a positive integer as seed:";
        cin>>seed;
        srand(seed);
        cout<<"\nEnter a integer rr as number of random
        parameter vectors\n";
        cin>>rr;
        cout<<"\nEnter two integers as scale factor\n";
        cout<<"of t and d\n";
        cin>>mm>>nn;
        cout<<"\nEnter the initial value and increament(float)
        and the number of points(int) for a\n";
        cin>>ini_a>>increa_a>>num_a;
        cout<<"\nEnter 3 integers (qq) for parameters: \n";
        cin>>qq1>>qq2>>qq3;
        cout<<"\nEnter the 3 integers (pp) for parameters: \n";
        cin>>pp1>>pp2>>pp3;
        a = ini_a;
        for (k=0; k<num_a; k++){
                num_rand = 0;
                sum=0.0;
        for(j=0; j<rr; j++){
                q = qq1*float(rand())/RAND_MAX+pp1;
                b = qq2*float(rand())/RAND_MAX+pp2;
                c = qq3*float(rand())/RAND_MAX+pp3;
                // in "limits.h" file, the RAND_MAX=INT_MAX=32767
                // to get random number between [0, 1].
        ifstream fin1("pu_can.C");
            ifstream fin2("pu_non.C");
```

```
    ifstream fin3("ra_can.C");
    ifstream fin4("ra_non.C");
total1=0.0;
        for(i=0; i<71; i++){
                fin1>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Cancer  PC(t,d,a,b,c,g);
                p=&PC;
                total1 += log(p->crude_density(t,d,a,b,c,g));
        }
        total2=0.0;
        for(i=0; i<258; i++){
                fin2>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Non_Cancer PNC(t,d,a,b,c,g);
                p=&PNC;
                total2 += log(p->crude_density(t,d,a,b,c,g));
        }
         total3=0.0;
        for(i=0; i<53; i++){
                fin3>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Cancer  RC(t,d,a,b,c,g);
                p=&RC;
                total3 += log(p->crude_density(t,d,a,b,c,g));
        }
         total4=0.0;
        for(i=0; i<132; i++){
                fin4>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Non_Cancer  RNC(t,d,a,b,c,g);
                p=&RNC;
                total4 += log(p->crude_density(t,d,a,b,c,g));
        }
      total=total1+total2+total3+total4;
        sum += exp(total/1000.0);
        num_rand += 1;
    fin1.close();
    fin2.close();
    fin3.close();
    fin4.close();
}
fout<<a<<" "<<sum/num_rand<<"\n";
cout<<"\n\n\n a=  "<< a <<"   Sum=  "<< sum/num_rand<<"\n\n"<<endl;
```

```
        a += increa_a;
        }
        fout.close();
        return 0;
}
*************************************************************************
```

```
// Title: dep_Beta_Mont.C

//input:    data:    pu_can.C; pu_non.C;  ra_can.C; ra_non.C;
//          include:  Pu_cancer.h, Pu-Non_Cancer.h, Ra_Cancer.h,
//                    Ra-Non_Cancer.h
//output:   marginal density of beta
//description:
//          Dependent Competing Risk Model
//          Monte Carlo Method

#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
#include"Pu_Cancer.h"
#include"Pu_Non_Cancer.h"
#include"Ra_Cancer.h"
#include"Ra_Non_Cancer.h"
// time=t; activity=d;   alpha1=a;   lambda1=b;
// beta=c; rou=g; alpha2=l; lambda2=m; delta=n;
main()
{
   int   tt, i, j, k, num_rand, rr;
   int  qq1, qq2, qq3, qq4, qq5, qq6;
   int pp1, pp2, pp3, pp4, pp5, pp6;
   float mm, nn, total;
   float t,d, total1, total2,total3, total4, sum;
   float a,b,c,g,l,m,n;
   float ini_c, increa_c;
   int   num_c;
   unsigned int seed;
        ofstream fout("dep_beta_monte_dat");
        Pu_Cancer *p;
        cout<<"\nEnter a positive integer as seed:";
        cin>>seed;
        srand(seed);
        cout<<"\nEnter a integer rr as number of random parameter vectors\n";
        cin>>rr;
        cout<<"\nEnter two integers as scale factor\n";
        cout<<"of t and d: (t/mm, d/nn)\n";
        cin>>mm>>nn;
        cout<<"\nEnter the initial value and increament(float) and the
          number of points(int) for c\n";
        cin>>ini_c>>increa_c>>num_c;
        cout<<"\nEnter the 6 integers (qq) for parameters :"<<"\n";
```

```
cin>>qq1>>qq2>>qq3>>qq4>>qq5>>qq6;
cout<<"\nEnter the 6 integers (pp) for parameters :"<<"\n";
cin>>pp1>> pp 2>> pp 3>> pp 4>> pp5>> pp6;
c = ini_c;
for (k=0; k<num_c; k++){
        num_rand = 0;
        sum=0.0;
for(j=0; j<rr; j++){
        a = qq1*float(rand())/RAND_MAX+pp1;
        b = qq2*float(rand())/RAND_MAX+pp2;
        q = qq3*float(rand())/RAND_MAX+pp3;
        l = qq4*float(rand())/RAND_MAX+pp4;
        m = qq5*float(rand())/RAND_MAX+pp5;
        n = qq6*float(rand())/RAND_MAX+pp6;
        // in "limits.h" file, the RAND_MAX=INT_MAX=32767
        // to get random number between [0, 1].
ifstream fin1("pu_can.C");
    ifstream fin2("pu_non.C");
    ifstream fin3("ra_can.C");
    ifstream fin4("ra_non.C");
total1=0.0;
        for(i=0; i<71; i++){
                fin1>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Cancer  PC(t,d,a,b,c,g,l,m,n);
                p=&PC;
                total1 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
        total2=0.0;
        for(i=0; i<258; i++){
                fin2>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Non_Cancer PNC(t,d,a,b,c,g,l,m,n);
                p=&PNC;
                total2 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
         total3=0.0;
        for(i=0; i<53; i++){
                fin3>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Cancer  RC(t,d,a,b,c,g,l,m,n);
                p=&RC;
                total3 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
         total4=0.0;
```

```
            for(i=0; i<132; i++){
                fin4>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Non_Cancer  RNC(t,d,a,b,c,g,l,m,n);
                p=&RNC;
                total4 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
            }
          total=total1+total2+total3+total4;
            sum += exp(total/rr);
          num_rand += 1;
        fin1.close();
        fin2.close();
        fin3.close();
        fin4.close();
            }
      fout<<c<<" "<<sum/num_rand<<"\n";
      cout<<"\n\n\n beta=  "<<c <<"   Sum=  "<< sum/num_rand<<"\n\n"<<endl;

c += increa_c;
      }
      fout.close();
      return 0;
}
```

```
// Title: dep_lamda_1_Mont.C

//input:    data:   pu_can.C; pu_non.C;  ra_can.C; ra_non.C;
//          include:  Pu_cancer.h, Pu-Non_Cancer.h, Ra_Cancer.h,
//                Ra-Non_Cancer.h
//output:   marginal density of lambda_1
//description:
//          Dependent Competing Risk Model
//          Monte Carlo Method

#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
#include"Pu_Cancer.h"
#include"Pu_Non_Cancer.h"
#include"Ra_Cancer.h"
#include"Ra_Non_Cancer.h"
// time=t; activity=d;  alpha1=a;  lambda1=b;
// beta=c; rou=g; alpha2=l; lambda2=m; delta=n;
```

```
main()
{
   int  tt, i, j, k, num_rand, rr;
   int  qq1,  qq2, qq3, qq4, qq5, qq6;
   int  pp1,  pp2, pp 3, pp 4, pp 5, pp6;
   float mm, nn, total;
   float t,d, total1, total2,total3, total4, sum;
   float a,b,c,g,l,m,n;
   float ini_b, increa_b;
   int  num_b;
   unsigned int seed;
        ofstream fout("dep_lamba_1_monte_dat");
        Pu_Cancer *p;
        cout<<"\nEnter a positive integer as seed:";
        cin>>seed;
        srand(seed);
        cout<<"\nEnter a integer rr as number of random parameter vectors\n";
        cin>>rr;
        cout<<"\nEnter two integers as scale factor\n";
        cout<<"of t and d: (t/mm, d/nn)\n";
        cin>>mm>>nn;
        cout<<"\nEnter the initial value and increament(float) and the
          number of points(int) for b\n";
        cin>>ini_g>>increa_g>>num_g;
        cout<<"\nEnter the 6 integers (qq) for parameters :"<<"\n";
        cin>>qq1>>qq2>>qq3>>qq4>>qq5>>qq6;
        cout<<"\nEnter the 6 integers (pp) for parameters :"<<"\n";
        cin>>pp1>> pp 2>> pp 3>> pp 4>> pp5>> pp6;
         b =ini_b ;
        for (k=0; k<num_b; k++){
                num_rand = 0;
                sum=0.0;
        for(j=0; j<rr; j++){
                a = qq1*float(rand())/RAND_MAX+pp1;
                q = qq1*float(rand())/RAND_MAX+pp1;
                c = qq1*float(rand())/RAND_MAX+pp1;
                l = qq1*float(rand())/RAND_MAX+pp1;
                m = qq1*float(rand())/RAND_MAX+pp1;
                n = qq1*float(rand())/RAND_MAX+pp1;
                // in "limits.h" file, the RAND_MAX=INT_MAX=32767
                // to get random number between [0, 1].
        ifstream fin1("pu_can.C");
            ifstream fin2("pu_non.C");
            ifstream fin3("ra_can.C");
```

```
    ifstream fin4("ra_non.C");
total1=0.0;
        for(i=0; i<71; I++)
        {
                fin1>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Cancer  PC(t,d,a,b,c,g,l,m,n);
                p=&PC;
                total1 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
        total2=0.0;
        for(i=0; i<258; i++){
                fin2>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Non_Cancer PNC(t,d,a,b,c,g,l,m,n);
                p=&PNC;
                total2 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
         total3=0.0;
        for(i=0; i<53; i++){
                fin3>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Cancer  RC(t,d,a,b,c,g,l,m,n);
                p=&RC;
                total3 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
         total4=0.0;
        for(i=0; i<132; i++){
                fin4>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Non_Cancer  RNC(t,d,a,b,c,g,l,m,n);
                p=&RNC;
                total4 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
     total=total1+total2+total3+total4;
        sum += exp(total/rr);
        num_rand += 1;
   fin1.close();
   fin2.close();
   fin3.close();
   fin4.close();
        }
fout<<b<<" "<<sum/num_rand<<"\n";
cout<<"\n\n\n beta =  "<< b  <<"   Sum=  "<< sum/num_rand<<"\n\n"<<endl;
b += increa_b;
```

```
        }
        fout.close();
        return 0;
}
```

```
// Title:    dep_alpha_2_Mont.C

//input:     data:    pu_can.C; pu_non.C;  ra_can.C; ra_non.C;
//            include:  Pu_cancer.h, Pu-Non_Cancer.h, Ra_Cancer.h,
//                      Ra-Non_Cancer.h
//output:    marginal density of alpha_2
//
//description:
//         Dependent Competing Risk Model
//         Monte Carlo Method

#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
#include"Pu_Cancer.h"
#include"Pu_Non_Cancer.h"
#include"Ra_Cancer.h"
#include"Ra_Non_Cancer.h"
// time=t; activity=d;   alpha1=a;   lambda1=b;
// beta=c; rho=g; alpha2=l; lambda2=m; delta=n;
main()
{
   int   tt, i, j, k, num_rand, rr;
   int qq1, qq2, qq3, qq4, qq5, qq6;
   int pp1, pp2, pp3, pp4, pp5, pp6;
   float mm, nn, total;
   float t,d, total1, total2,total3, total4, sum;
   float a,b,c,g,l,m,n;
   float ini_L, increa_L, qq;
   int   num_L;
   unsigned int seed;
        ofstream fout("dep_alpha_2_monte_dat");
        Pu_Cancer *p;
        cout<<"\nEnter a positive integer as seed:";
        cin>>seed;
        srand(seed);
        cout<<"\nEnter a integer rr as number of random parameter vectors\n";
        cin>>rr;
```

```
cout<<"\nEnter two integers as scale factor\n";
cout<<"of t and d: (t/mm, d/nn)\n";
cin>>mm>>nn;
cout<<"\nEnter the initial value and increament(float) and the
 number of points(int) for L\n";
cin>>ini_L>>increa_L>>num_L;
cout<<"\nEnter the 6 integers (qq) for parameters :"<<"\n";
cin>>qq1>>qq2>>qq3>>qq4>>qq5>>qq6;
cout<<"\nEnter the 6 integers (pp) for parameters :"<<"\n";
cin>>pp1>> pp 2>> pp 3>> pp 4>> pp5>> pp6;
 L =ini_L;
for (k=0; k<num_L; k++){
        num_rand = 0;
        sum=0.0;
for(j=0; j<rr; j++){
        a = qq1*float(rand())/RAND_MAX+pp1;
        b = qq2*float(rand())/RAND_MAX+pp2;
        c = qq3*float(rand())/RAND_MAX+pp3;
        q = qq4*float(rand())/RAND_MAX+pp4;
        m = qq5*float(rand())/RAND_MAX+pp5;
        n = qq6*float(rand())/RAND_MAX+pp6;
        // in "limits.h" file, the RAND_MAX=INT_MAX=32767
        // to get random number between [0, 1].
    ifstream fin1("pu_can.C");
    ifstream fin2("pu_non.C");
    ifstream fin3("ra_can.C");
    ifstream fin4("ra_non.C");
        total1=0.0;
        for(i=0; i<71; i++){
                fin1>>d>>tt;
                t=tt/mm;   d=d/nn;
                Pu_Cancer  PC(t,d,a,b,c,g,l,m,n);
                p=&PC;
                total1 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
        total2=0.0;
        for(i=0; i<258; i++){
                fin2>>d>>tt;
                t=tt/mm;   d=d/nn;
                Pu_Non_Cancer PNC(t,d,a,b,c,g,l,m,n);
                p=&PNC;
                total2 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
         total3=0.0;
        for(i=0; i<53; i++){
```

```
                    fin3>>d>>tt;
                    t=tt/mm;  d=d/nn;
                    Ra_Cancer RC(t,d,a,b,c,g,l,m,n);
                    p=&RC;
                    total3 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
                }
                 total4=0.0;
                for(i=0; i<132; i++){
                    fin4>>d>>tt;
                    t=tt/mm;  d=d/nn;
                    Ra_Non_Cancer RNC(t,d,a,b,c,g,l,m,n);
                    p=&RNC;
                    total4 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
                }
              total=total1+total2+total3+total4;
                sum += exp(total/rr);
                num_rand += 1;
            fin1.close();
            fin2.close();
            fin3.close();
            fin4.close();
                }
        fout<<L<<" "<<sum/num_rand<<"\n";
        cout<<"\n\n\n a_2=  "<<L  <<"   Sum=  "<< sum/num_rand<<"\n\n"<<endl;
        L +=increa_L;
        }
        fout.close();
        return 0;
}

**************************************************************************

// Title: dep_lambda_2_Mont.C

//input:    data:    pu_can.C; pu_non.C;  ra_can.C; ra_non.C;
//          include:  Pu_cancer.h, Pu-Non_Cancer.h, Ra_Cancer.h,
//                    Ra-Non_Cancer.h
//output:   marginal density of  lambda_2
//description:
//          Dependent Competing Risk Model
//          Monte Carlo Method

#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
```

```
#include"Pu_Cancer.h"
#include"Pu_Non_Cancer.h"
#include"Ra_Cancer.h"
#include"Ra_Non_Cancer.h"
// time=t; activity=d;  alpha1=a;  lambda1=b;
// beta=c; rho=g; alpha2=l; lambda2=m; delta=n;
main()
{
  int  tt, i, j, k, num_rand, rr;
  int qq1, qq2, qq3, qq4, qq5, qq6;
  int pp1, pp2, pp3, pp4, pp5, pp6;
  float mm, nn, total;
  float t,d, total1, total2,total3, total4, sum;
  float a,b,c,g,l,m,n;
  float ini_m, increa_m, qq;
  int  num_m;
  unsigned int seed;
      ofstream fout("dep_lambda_2_monte_dat");
      Pu_Cancer *p;
      cout<<"\nEnter a positive integer as seed:";
      cin>>seed;
      srand(seed);
      cout<<"\nEnter a integer rr as number of random parameter vectors\n";
      cin>>rr;
      cout<<"\nEnter two integers as scale factor\n";
      cout<<"of t and d: (t/mm, d/nn)\n";
      cin>>mm>>nn;
      cout<<"\nEnter the initial value and increament(float) and the
       number of points(int) for g\n";
      cin>>ini_m>>increa_m>>num_m;
      cout<<"\nEnter the 6 integers (qq) for parameters :"<<"\n";
      cin>>qq1>>qq2>>qq3>>qq4>>qq5>>qq6;
      cout<<"\nEnter the 6 integers (pp) for parameters :"<<"\n";
      cin>>pp1>> pp 2>> pp 3>> pp 4>> pp5>> pp6;
       m = ini_m;
      for (k=0; k<num_m; k++){
            num_rand = 0;
            sum=0.0;
      for(j=0; j<rr; j++){
            a = qq1*float(rand())/RAND_MAX+pp1;
            b = qq2*float(rand())/RAND_MAX+pp2;
            c = qq3*float(rand())/RAND_MAX+pp3;
            l = qq4*float(rand())/RAND_MAX+pp4;
            q = qq5*float(rand())/RAND_MAX+pp5;
            n = qq6*float(rand())/RAND_MAX+pp6;
```

```
        // in "limits.h" file, the RAND_MAX=INT_MAX=32767
        // to get random number between [0, 1].
ifstream fin1("pu_can.C");
    ifstream fin2("pu_non.C");
    ifstream fin3("ra_can.C");
    ifstream fin4("ra_non.C");
        total1=0.0;
        for(i=0; i<71; i++){
                fin1>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Cancer  PC(t,d,a,b,c,g,l,m,n);
                p=&PC;
                total1 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
        total2=0.0;
        for(i=0; i<258; i++){
                fin2>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Non_Cancer PNC(t,d,a,b,c,g,l,m,n);
                p=&PNC;
                total2 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
         total3=0.0;
        for(i=0; i<53; i++){
                fin3>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Cancer  RC(t,d,a,b,c,g,l,m,n);
                p=&RC;
                total3 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
         total4=0.0;
        for(i=0; i<132; i++){
                fin4>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Non_Cancer  RNC(t,d,a,b,c,g,l,m,n);
                p=&RNC;
                total4 += log(p->crude_density(t,d,a,b,c,g,l,m,n));
        }
      total=total1+total2+total3+total4;
        sum += exp(total/rr);
        num_rand += 1;
    fin1.close();
    fin2.close();
    fin3.close();
    fin4.close();
```

```
                }
        fout<<m<<" "<<sum/num_rand<<"\n";
        cout<<"\n\n\n lambda_2=  "<< m  <<" Sum= "<<sum/num_rand<<"\n\n"<<endl;
        m += increa_m;
        }
        fout.close();
        return 0;
}

**********************************************************************

//      Title:  ind_beta_mont.C

// input:      pu_can.C; pu_non.C; ra_can.C; ra_non.C
// output:     marginal density of beta
// description: independent competing risk model
//           Monte carlo Method

#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
class Pu_Cancer{
protected:
                float time;
                float activity;
                float alpha1,lambda1, beta, rho;
public :
Pu_Cancer(float t, float d, float a, float b, float c, float g);
        virtual float cancer_hazard_rate(float t, float d,float a,float b,
        float c, float g );
         float cancer_cumu_hazard (float t, float d,float a,float b,
                                        float c, float g);
          float cancer_survival(float t, float d, float a,float b,
                                        float c, float g);
          float cancer_failure(float t, float d, float a,float b,
                                        float c, float g);
          virtual float crude_density(float t, float d, float a,float b,
                                        float c, float g);
};
class Pu_Non_Cancer : public Pu_Cancer {
public:
                Pu_Non_Cancer(float t, float d, float a,float b,
                        float c, float g ):Pu_Cancer(t,d,a,b,c,g)
```

```
                {       time=t; activity=d;  alpha1=a;  lambda1=b;
                        beta=c; rou=g;
                }
        virtual float crude_density(float t, float d,float a,float b,
                        float c, float g);
} ;
class Ra_Cancer:public Pu_Cancer{
public:
      Ra_Cancer(float t, float d, float a,float b, float c,
            float g):Pu_Cancer(t,d,a,b,c,g)
                        { time=t; activity=d;  alpha1=a;  lambda1=b;
                      beta=c; rou=g;  };
  virtual float cancer_hazard_rate(float t, float d, float a, float b,
                        float c, float g);
};
class Ra_Non_Cancer:public Ra_Cancer{
public:
                Ra_Non_Cancer(float t, float d, float a, float b, float c,
                                float g):Ra_Cancer(t,d,a,b,c,g){
                      time=t; activity=d;  alpha1=a;  lambda1=b;
                        beta=c; rou=g;};
          virtual float crude_density( float t, float d, float a,
                      float b, float c,float g);
};
Pu_Cancer::Pu_Cancer(float t,float d, float a,float b,float c,float g){
        time=t; activity=d; alpha1=a;lambda1=b;
        beta=c;rou=g;
};
float Pu_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                      float c, float g)
{                     float Pu_C_h;
                      Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(g*c*d);
                      return Pu_C_h;
};
float Pu_Cancer::cancer_cumu_hazard(float t, float d, float a,float b,
                                                float c, float g ){
            float Pu_C_H;
                      Pu_C_H=cancer_hazard_rate(t,d,a,b,c,g)*t/a;
                      return Pu_C_H;
};
float Pu_Cancer::cancer_survival(float t, float d, float a,float b,
                                                float c, float g ){
                      float Pu_C_S;
                Pu_C_S=exp(-cancer_cumu_hazard(t,d,a,b,c,g)*exp(-20));
                      return Pu_C_S;
```

```
};
float Pu_Cancer::cancer_failure(float t, float d, float a,float b,
                                                        float c, float g){
                        float Pu_C_f;
                        Pu_C_f=cancer_hazard_rate(t,d,a,b,c,g)*
                                cancer_survival(t,d,a,b,c,g);
                        return Pu_C_f;
};
float Pu_Cancer :: crude_density(float t, float d,float a,float b,
                        float c, float g ){
        float Pu_C_Density;
                Pu_C_Density =cancer_failure(t,d,a,b,c,g);
                        return  Pu_C_Density;
}
float Pu_Non_Cancer :: crude_density(float t, float d,float a,float b,
                        float c, float g ){
            float Pu_NC_Density;
                Pu_NC_Density =cancer_survival(t,d,a,b,c,g);
                        return  Pu_NC_Density;
}
float Ra_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                                                        float c, float g){
                        float Pu_C_h;
                        g/g;
                        Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(c*d);
                        return Pu_C_h;
};
float Ra_Non_Cancer :: crude_density(float t, float d, float a,float b,
                        float c, float g ){
        float Ra_NC_Density;
                Ra_NC_Density =cancer_survival(t,d,a,b,c,g);
                        return  Ra_NC_Density;
};
main(){
   int tt, i, j, k, num_rand, rr, qq1, qq2, qq3;
   int pp1, pp2, pp3;
   float mm, nn, total;
   float t,d, total1, total2,total3, total4, sum;
   float a,b,c,g,qq;
   float ini_c, increa_c;
   int  num_c;
   unsigned int seed;
        ofstream fout("ind_beta_mont_dat");
        Pu_Cancer *p;
        cout<<"\nEnter a positive integer as seed:";
```

```
cin>>seed;
srand(seed);
cout<<"\nEnter a integer rr as number of random
parameter vectors\n";
cin>>rr;
cout<<"\nEnter two integers as scale factor\n";
cout<<"of t and d\n";
cin>>mm>>nn;
cout<<"\nEnter the initial value and increament(float)
and the number of points(int) for c\n";
cin>>ini_c>>increa_c>>num_c;
cout<<"\nEnter 3 integers (qq) for parameters: \n";
cin>>qq1>>qq2>>qq3;
cout<<"\nEnter the 3 integers (pp) for parameters: \n";
cin>>pp1>>pp2>>pp3;
c = ini_c;
for (k=0; k<num_c; k++){
        num_rand = 0;
        sum=0.0;
for(j=0; j<rr; j++){
        q = qq1*float(rand())/RAND_MAX+pp1;
        a = qq2*float(rand())/RAND_MAX+pp2;
        b = qq3*float(rand())/RAND_MAX+pp3;
        // in "limits.h" file, the RAND_MAX=INT_MAX=32767
        // to get random number between [0, 1].
ifstream fin1("pu_can.C");
    ifstream fin2("pu_non.C");
    ifstream fin3("ra_can.C");
    ifstream fin4("ra_non.C");
        total1=0.0;
        for(i=0; i<71; i++){
                fin1>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Cancer  PC(t,d,a,b,c,g);
                p=&PC;
                total1 += log(p->crude_density(t,d,a,b,c,g));
        }
        total2=0.0;
        for(i=0; i<258; i++){
                fin2>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Non_Cancer PNC(t,d,a,b,c,g);
                p=&PNC;
                total2 += log(p->crude_density(t,d,a,b,c,g));
        }
```

```
                total3=0.0;
                for(i=0; i<53; i++){
                        fin3>>d>>tt;
                        t=tt/mm;  d=d/nn;
                        Ra_Cancer RC(t,d,a,b,c,g);
                        p=&RC;
                        total3 += log(p->crude_density(t,d,a,b,c,g));
                }
                total4=0.0;
                for(i=0; i<132; i++){
                        fin4>>d>>tt;
                        t=tt/mm;  d=d/nn;
                        Ra_Non_Cancer RNC(t,d,a,b,c,g);
                        p=&RNC;
                        total4 += log(p->crude_density(t,d,a,b,c,g));
                }
              total=total1+total2+total3+total4;
                sum += exp(total/1000.0);
                num_rand += 1;
            fin1.close();
            fin2.close();
            fin3.close();
            fin4.close();
        }
        fout<<c<<" "<<sum/num_rand<<"\n";
        cout<<"\n\n\n beta=  "<< c  <<"   Sum=  "<< sum/num_rand<<"\n\n"<<endl;
        c += increa_c;
        }
        fout.close();
        return 0;
}

**********************************************************************

//      Title:  ind_lambda_1_mont.C

// input:     pu_can.C; pu_non.C; ra_can.C; ra_non.C
// output:    marginal density of lambda_1
// description: independent competing risk model
//          Monte carlo Method

#include<iostream.h>
#include<fstream.h>
#include<math.h>
#include<stdlib.h>
class Pu_Cancer{
```

```
protected:
                float time;
                float activity;
                float alpha1,lambda1, beta, rho;
public :
Pu_Cancer(float t, float d, float a, float b, float c, float g);
        virtual float cancer_hazard_rate(float t, float d,float a,float b,
        float c, float g );
         float cancer_cumu_hazard (float t, float d,float a,float b,
                                        float c, float g);
          float cancer_survival(float t, float d, float a,float b,
                                        float c, float g);
          float cancer_failure(float t, float d, float a,float b,
                                        float c, float g);
          virtual float crude_density(float t, float d, float a,float b,
                                        float c, float g);
};
class Pu_Non_Cancer : public Pu_Cancer {
public:
                Pu_Non_Cancer(float t, float d, float a,float b,
                        float c, float g ):Pu_Cancer(t,d,a,b,c,g)
                {       time=t; activity=d;  alpha1=a;  lambda1=b;
                        beta=c; rou=g;
                }
        virtual float crude_density(float t, float d,float a,float b,
                                float c, float g);
} ;
class Ra_Cancer:public Pu_Cancer{
public:
      Ra_Cancer(float t, float d, float a,float b, float c,
            float g):Pu_Cancer(t,d,a,b,c,g)
                                { time=t; activity=d;  alpha1=a;  lambda1=b;
                              beta=c; rou=g;  };
  virtual float cancer_hazard_rate(float t, float d, float a, float b,
                                float c, float g);
};
class Ra_Non_Cancer:public Ra_Cancer{
public:
                 Ra_Non_Cancer(float t, float d, float a, float b, float c,
                                        float g):Ra_Cancer(t,d,a,b,c,g){
                        time=t; activity=d;  alpha1=a;  lambda1=b;
                          beta=c; rou=g;};
          virtual float crude_density( float t, float d, float a,
                        float b, float c,float g);
};
```

```
Pu_Cancer::Pu_Cancer(float t,float d, float a,float b,float c,float g){
        time=t; activity=d; alpha1=a;lambda1=b;
        beta=c;rou=g;
};
float Pu_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                        float c, float g)
{                       float Pu_C_h;
                        Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(g*c*d);
                        return Pu_C_h;
};
float Pu_Cancer::cancer_cumu_hazard(float t, float d, float a,float b,
                                                        float c, float g ){
            float Pu_C_H;
                        Pu_C_H=cancer_hazard_rate(t,d,a,b,c,g)*t/a;
                        return Pu_C_H;
};
float Pu_Cancer::cancer_survival(float t, float d, float a,float b,
                                                        float c, float g ){
                        float Pu_C_S;
                Pu_C_S=exp(-cancer_cumu_hazard(t,d,a,b,c,g)*exp(-20));
                        return Pu_C_S;
};
float Pu_Cancer::cancer_failure(float t, float d, float a,float b,
                                                        float c, float g){
                        float Pu_C_f;
                        Pu_C_f=cancer_hazard_rate(t,d,a,b,c,g)*
                                cancer_survival(t,d,a,b,c,g);
                        return Pu_C_f;
};
float Pu_Cancer :: crude_density(float t, float d,float a,float b,
                        float c, float g ){
        float Pu_C_Density;
                Pu_C_Density =cancer_failure(t,d,a,b,c,g);
                        return  Pu_C_Density;
}
float Pu_Non_Cancer :: crude_density(float t, float d,float a,float b,
                        float c, float g ){
            float Pu_NC_Density;
                Pu_NC_Density =cancer_survival(t,d,a,b,c,g);
                        return  Pu_NC_Density;
}
float Ra_Cancer::cancer_hazard_rate(float t, float d, float a,float b,
                                                        float c, float g){
                        float Pu_C_h;
                        g/g;
```

```
                    Pu_C_h=a*pow(b,a)*pow((float)t,(a-1))*exp(c*d);
                    return Pu_C_h;
};
float Ra_Non_Cancer :: crude_density(float t, float d, float a,float b,
                    float c, float g ){
        float Ra_NC_Density;
                Ra_NC_Density =cancer_survival(t,d,a,b,c,g);
                    return  Ra_NC_Density;
};
main(){
   int tt, i, j, k, num_rand, rr, qq1, qq2, qq3;
   int pp1, pp2, pp3;
   float mm, nn, total;
   float t,d, total1, total2,total3, total4, sum;
   float a,b,c,g,qq;
   float ini_b, increa_b;
   int  num_b;
   unsigned int seed;
        ofstream fout("ind_lambda_1_mont_dat");
        Pu_Cancer *p;
        cout<<"\nEnter a positive integer as seed:";
        cin>>seed;
        srand(seed);
        cout<<"\nEnter a integer rr as number of random
        parameter vectors\n";
        cin>>rr;
        cout<<"\nEnter two integers as scale factor\n";
        cout<<"of t and d\n";
        cin>>mm>>nn;
        cout<<"\nEnter the initial value and increament(float)
       and the number of points(int) for b\n";
        cin>>ini_b>>increa_b>>num_b;
        cout<<"\nEnter 3 integers (qq) for parameters: \n";
        cin>>qq1>>qq2>>qq3;
        cout<<"\nEnter the 3 integers (pp) for parameters: \n";
        cin>>pp1>>pp2>>pp3;
        b = ini_b;
        for (k=0; k<num_c; k++){
                num_rand = 0;
                sum=0.0;
        for(j=0; j<rr; j++){
                q = qq1*float(rand())/RAND_MAX+pp1;
                a = qq2*float(rand())/RAND_MAX+pp2;
                c = qq3*float(rand())/RAND_MAX+pp3;
                // in "limits.h" file, the RAND_MAX=INT_MAX=32767
```

```
        // to get random number between [0, 1].
ifstream fin1("pu_can.C");
    ifstream fin2("pu_non.C");
    ifstream fin3("ra_can.C");
    ifstream fin4("ra_non.C");
        total1=0.0;
        for(i=0; i<71; i++){
                fin1>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Cancer  PC(t,d,a,b,c,g);
                p=&PC;
                total1 += log(p->crude_density(t,d,a,b,c,g));
        }
        total2=0.0;
        for(i=0; i<258; i++){
                fin2>>d>>tt;
                t=tt/mm;  d=d/nn;
                Pu_Non_Cancer PNC(t,d,a,b,c,g);
                p=&PNC;
                total2 += log(p->crude_density(t,d,a,b,c,g));
        }
         total3=0.0;
        for(i=0; i<53; i++){
                fin3>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Cancer  RC(t,d,a,b,c,g);
                p=&RC;
                total3 += log(p->crude_density(t,d,a,b,c,g));
        }
         total4=0.0;
        for(i=0; i<132; i++){
                fin4>>d>>tt;
                t=tt/mm;  d=d/nn;
                Ra_Non_Cancer  RNC(t,d,a,b,c,g);
                p=&RNC;
                total4 += log(p->crude_density(t,d,a,b,c,g));
        }
      total=total1+total2+total3+total4;
        sum += exp(total/1000.0);
        num_rand += 1;
    fin1.close();
    fin2.close();
    fin3.close();
    fin4.close();
```

```
        }
        fout<<c<<" "<<sum/num_rand<<"\n";
        cout<<"\n\n\n b=  "<< b  <<"   Sum=  "<< sum/num_rand<<"\n\n"<<endl;
        b += increa_b;
        }
        fout.close();
        return 0;
}
*************************************************************************
```

APPENDIX C

SURVIVAL DATA

Pu_Data

sex	injection level	weight [kg]	activity density [μCi/kg]	time [day]	cause of death
1	0	9.7	0	4003	0
2	0	6.36	0	2755	0
1	0	10.8	0	5362	0
1	0	10.7	0	5138	0
2	0	9.75	0	4088	0
2	0	5.59	0	4490	0
2	0	6.9	0	5344	0
1	0	10.9	0	4072	0
2	0	11	0	3032	0
2	0	11	0	3971	0
1	0	10.3	0	3821	0
1	0	10.9	0	4143	0
2	0	9.47	0	5361	1
2	0	9.89	0	4105	0
1	0	12.1	0	3750	0
1	0	13.9	0	4756	0
1	0	12.2	0	5535	0
2	0	11.4	0	4849	0
1	0	13.1	0	5203	0
2	0	8.5	0	3748	0
1	0	13.3	0	4157	0
2	0	10.6	0	4403	0
1	0	11.8	0	1763	0
1	0	12.6	0	3629	0
1	0	11.2	0	4840	0
1	0	10.3	0	5046	0
1	0	12.1	0	4923	0
1	0	11.7	0	4164	0
1	0	13.5	0	5487	0
1	0	12.7	0	4139	0
1	0	12.5	0	5654	0
1	0	13.1	0	3501	0
1	0	11	0	2525	0
1	0	12.2	0	5244	0
1	0	11.7	0	4179	0
1	0	10.4	0	4485	0
1	0	10.7	0	3623	0
1	0	12.2	0	3382	0
1	0	10.7	0	6061	0
1	0	10.1	0	5113	0
1	0	10.3	0	4957	0
1	0	11.4	0	3377	0
1	0	9.49	0	3875	0
1	0	11.6	0	32	0
1	0	14	0	3242	0
1	0	13.8	0	4216	0
1	0	12.7	0	4359	0
1	0	11.5	0	1537	0
1	0	11.5	0	4232	0
1	0	11.3	0	3715	0

Sex	inject level	weight [kg]	activity density [μCi/kg]	time [day]	cause of death
1	0	11.7	0	4232	0
1	0.1	9.67	0.00071	4319	0
1	0.1	12	0.00059	4146	0
1	0.1	12.2	0.00057	4346	0
2	0.1	9.28	0.0007	4221	0
1	0.1	11.6	0.00063	5519	0
2	0.1	9.8	0.00075	3939	0
1	0.1	11.3	0.00059	4676	0
2	0.1	9.8	0.00059	2968	0
1	0.1	12.2	0.00088	2760	0
2	0.1	10.4	0.00059	5272	0
1	0.1	10.8	0.00079	4156	0
2	0.1	11.1	0.00058	3292	0
1	0.1	10.3	0.00059	5036	0
2	0.1	9.79	0.0006	3600	0
1	0.1	11.3	0.00059	5072	0
2	0.1	9.52	0.00057	1979	0
1	0.1	10.5	0.00058	4466	0
1	0.1	10.9	0.00057	4412	0
2	0.1	8.34	0.00072	4025	0
1	0.1	13.5	0.00122	3762	0
2	0.1	9	0.00051	4618	0
1	0.1	13.2	0.0012	3630	0
2	0.1	10.2	0.00057	4409	0
2	0.1	9.34	0.00051	4507	0
2	0.1	10.9	0.0012	4323	0
2	0.1	9.71	0.00124	3222	0
2	0.2	9.44	0.0026	3221	0
2	0.2	7.44	0.00173	3983	0
1	0.2	10.9	0.0021	4803	0
1	0.2	11.4	0.00163	2841	0
1	0.2	11.8	0.00171	4391	0
2	0.2	9.46	0.002	5319	0
1	0.2	12.1	0.00198	4392	0
2	0.2	8.3	0.00224	4299	0
1	0.2	12.1	0.00181	4708	0
2	0.2	8.3	0.00176	4080	0
1	0.2	10.7	0.00185	2640	0
2	0.2	11.9	0.00169	4971	0
2	0.2	9.35	0.00186	5378	0
1	0.2	13.6	0.00178	3591	0
2	0.2	10.1	0.00183	3881	0
2	0.2	8.04	0.00193	5241	0
1	0.2	14.5	0.00178	2776	0
2	0.2	12.5	0.00176	4615	0
2	0.2	12.7	0.00178	5068	0
1	0.2	12.7	0.0017	3934	0
2	0.2	11.5	0.00172	4515	0
2	0.2	9.92	0.00167	4552	0
1	0.2	11.2	0.00133	4359	0

Sex	inject level	weight [kg]	activity density [µCi/kg]	time [day]	cause of death
2	0.2	10.5	0.00175	2593	0
2	0.2	9.1	0.00175	4330	0
1	0.2	10.6	0.00149	2804	0
2	0.2	10.1	0.0015	4829	0
2	0.2	7.14	0.00153	4787	0
1	0.2	10	0.00152	3546	0
2	0.2	7.95	0.00211	5006	0
2	0.2	9.08	0.00176	2880	0
1	0.2	11.6	0.00214	2916	0
2	0.2	9.46	0.00173	4911	0
2	0.2	9.34	0.00176	3401	0
1	0.2	9.42	0.00182	4618	0
2	0.2	9.55	0.00176	3966	1
1	0.2	11	0.00179	4359	0
2	0.2	9.5	0.00239	4942	0
1	0.2	12	0.00188	4359	0
1	0.2	11.6	0.00184	4359	0
1	0.2	11.6	0.00184	3631	0
1	0.2	11.4	0.00188	4359	0
1	0.2	11.1	0.00179	2624	0
2	0.5	9.93	0.0054	2388	0
2	0.5	11.2	0.00459	3498	0
2	0.5	9.98	0.00493	4537	0
1	0.5	8.41	0.00627	4588	0
1	0.5	12.6	0.00521	4062	0
1	0.5	13.4	0.0056	4564	0
2	0.5	8.98	0.00594	4333	0
1	0.5	11.9	0.00645	3829	1
2	0.5	9.3	0.00553	3490	0
1	0.5	9.8	0.00526	4954	0
2	0.5	8.1	0.00525	5203	0
1	0.5	9.7	0.00539	4443	0
1	0.5	9.96	0.00536	4062	0
1	0.5	10.5	0.00549	1648	0
2	0.5	8.44	0.00572	2546	0
1	0.5	13.6	0.00546	2275	0
2	0.5	10.1	0.00559	4509	0
1	0.5	12.5	0.00642	1981	0
2	0.5	9.54	0.0052	4502	0
1	0.5	11.5	0.00527	3885	1
2	0.5	8.39	0.00454	4956	0
1	0.5	10.5	0.00448	3498	0
2	0.5	10.9	0.00528	4350	0
1	0.5	13.1	0.00675	4194	0
1	0.5	12.2	0.00668	4778	0
2	0.5	9.71	0.00668	3618	0
2	0.5	11.6	0.00484	4177	0
2	0.5	10.7	0.0044	4393	0
1	0.5	12	0.00546	4618	0
1	0.5	11.5	0.0036	99	0
2	0.5	8.48	0.00634	4279	0

sex	inject level	weight [kg]	activity density [μCi/kg]	time [day]	cause of death
2	0.5	10	0.00516	35	0
1	0.5	11.8	0.00336	7	0
2	0.5	9.15	0.00516	7	0
1	0.5	11.9	0.00524	2302	0
1	0.5	12.1	0.00546	4253	0
2	0.5	9.1	0.00537	3240	0
2	0.5	12.8	0.00541	4253	0
1	0.5	11.9	0.00552	4232	0
1	0.5	10.4	0.00549	3205	0
1	0.5	13.2	0.00515	4004	1
1	0.5	10.8	0.00551	2478	0
2	0.5	9.65	0.00547	2943	0
2	0.5	9.56	0.00552	2917	0
2	0.5	8.14	0.00524	4166	0
2	0.7	8.98	0.00947	4746	0
1	0.7	10.3	0.00941	3471	0
1	0.7	11.9	0.0102	3575	0
1	0.7	8.04	0.0103	3938	0
2	0.7	9.66	0.00942	4823	0
1	0.7	11.6	0.0104	1737	0
2	0.7	9.18	0.00926	3718	0
1	0.7	11.1	0.0104	5212	0
2	0.7	9.69	0.0108	4657	0
2	0.7	9.56	0.0108	4481	1
2	0.7	8.9	0.011	3861	1
1	0.7	10.9	0.0117	2042	0
1	0.7	10.2	0.0112	4023	1
1	0.7	11.9	0.0116	4575	0
2	0.7	10.6	0.011	4512	0
2	0.7	8.63	0.0113	4210	0
2	0.7	9.91	0.0113	4246	1
1	0.7	13.2	0.00956	4253	0
1	0.7	9.46	0.00967	3496	0
1	0.7	10.7	0.0103	4443	0
1	0.7	10.1	0.00979	3711	0
2	0.7	10.6	0.00981	3661	0
2	0.7	11.1	0.00999	3951	0
1	0.7	11.6	0.00999	2836	0
2	0.7	11.4	0.00995	4235	0
2	0.7	8.24	0.00969	4500	0
1	0.7	10.7	0.0099	4074	0
1	0.7	11.9	0.0105	2750	0
1	0.7	11.6	0.0106	4235	0
1	0.7	10.1	0.0106	4443	0
2	0.7	10.2	0.0105	4285	0
1	0.7	10.8	0.0104	4362	1
2	0.7	10.2	0.0105	4443	0
2	0.7	8.39	0.0103	4443	0
1	0.7	10.3	0.0091	4409	0

sex	inject level	weight [kg]	activity density [μCi/kg]	time [day]	cause of death
1	0.7	12.3	0.0099	3405	0
2	0.7	9.33	0.0112	4030	1
1	1	9.41	0.015	4572	1
2	1	6.85	0.0163	4810	0
1	1	8	0.0165	4292	1
1	1	9.97	0.0139	4549	0
2	1	8.8	0.0142	1539	0
2	1	11	0.0168	3764	0
2	1	7.38	0.014	4292	0
2	1	6.36	0.0167	3981	0
1	1	10.6	0.0172	3367	1
2	1	7.87	0.0168	2257	1
2	1	12	0.0152	3649	0
1	1	8.9	0.0157	5161	0
1	1	9.67	0.0167	2374	0
1	1	12.7	0.0153	5277	0
2	1	10.4	0.0141	4185	0
1	1	12.8	0.0159	3596	0
1	1	10.6	0.0165	4211	1
1	1	10.9	0.0151	4690	0
1	1	9.89	0.014	4788	0
1	1	9.82	0.0159	5217	0
2	1	10.4	0.0141	3482	0
1	1	10	0.0156	3734	1
2	1	9.04	0.0139	4035	1
2	1	11.2	0.0163	4508	0
2	1	10.4	0.0163	3308	0
1	1	11	0.0168	4618	0
2	1	2.23	0.0171	4211	0
1	1	2.83	0.0171	3549	0
2	1	3.56	0.0137	1929	0
1	1	5.14	0.0158	3789	0
1	1	5.19	0.0143	3681	0
2	1	4.36	0.0142	3681	0
2	1	4.46	0.0194	3461	0
2	1	2.52	0.0146	3392	0
1	1	3.78	0.0155	2944	0
1	1	3.77	0.0158	2869	0
2	1	9.54	0.0158	3752	0
2	1	9.54	0.0174	2990	1
2	1	9.76	0.0163	2883	0
1	1	13.3	0.0158	2883	0
1	1.7	8.72	0.0475	3025	1
2	1.7	8.62	0.0431	3430	1
1	1.7	8.63	0.0495	3430	0
1	1.7	8.37	0.0484	3312	1
2	1.7	11.6	0.0493	2659	1
2	1.7	10.3	0.0459	2221	1
2	1.7	9.73	0.0481	3353	0
1	1.7	13.6	0.0479	3282	1
2	1.7	9.72	0.0485	2500	1
2	1.7	10.6	0.0495	467	0

sex	inject level	weight [kg]	activity density [μCi/kg]	time [day]	cause of death
2	1.7	8.07	0.0457	4214	1
1	1.7	9.41	0.0491	2973	0
1	1.7	10.6	0.0473	4375	0
2	1.7	2.34	0.0543	4211	0
1	1.7	2.97	0.0545	3604	0
2	1.7	3.84	0.0453	3912	0
1	1.7	3.4	0.0488	3625	0
1	1.7	4.06	0.0485	3625	0
2	1.7	4.2	0.0529	3885	1
2	1.7	3.91	0.0477	2999	0
1	1.7	4.32	0.0473	3461	0
2	1.7	3.06	0.051	2902	0
1	1.7	3.07	0.0464	2820	0
2	1.7	3.02	0.0471	2820	0
2	1.7	10	0.0456	2163	0
2	1.7	10	0.0416	3758	0
2	1.7	10.2	0.0519	3502	0
2	1.7	11.2	0.0527	3190	0
2	1.7	9.96	0.0441	2577	1
2	1.7	9.56	0.0449	2883	0
1	1.7	10.7	0.0458	2811	0
1	1.7	12.6	0.043	2762	0
1	1.7	12.7	0.0498	2160	0
1	1.7	11.5	0.0503	2002	1
1	2	7.61	0.0853	2985	1
2	2	7.73	0.112	2780	1
1	2	10.5	0.094	3185	1
1	2	9.84	0.0862	2948	1
2	2	8.12	0.0846	2423	1
2	2	7.54	0.0902	2947	1
2	2	8.4	0.0996	2093	0
1	2	9.73	0.0957	1761	0
2	2	9.72	0.101	2014	1
2	2	7.94	0.0968	2912	1
1	2	10.3	0.0961	1617	1
1	2	9.98	0.1	2284	1
1	3	8	0.261	1476	1
2	3	6.85	0.312	1947	1
1	3	8.74	0.291	1604	1
1	3	8.51	0.292	1950	1
2	3	8.22	0.288	1504	1
2	3	8.38	0.282	1617	1
2	3	9	0.314	1627	1
1	3	9.73	0.3	1771	1
2	3	7.67	0.3	1894	1
2	3	8.94	0.298	1546	1
1	3	10.5	0.309	1198	1
1	3	10.2	0.308	1659	1
1	4	7.61	0.823	1724	1
2	4	8.65	1.03	1556	1
1	4	9.36	0.929	1198	1
1	4	8.74	0.974	1066	1

sex	inject level	weight [kg]	activity density [μCi/kg]	time [day]	cause of death
2	4	9.26	0.811	1357	1
2	4	8.45	0.963	1198	1
1	4	9.22	0.887	1157	1
2	4	8.58	0.96	1343	1
2	4	6.48	0.868	1241	1
1	4	9.56	0.927	1288	1
1	4	11.4	0.838	1463	1
1	5	8.86	2.67	1324	1
2	5	8.75	3.3	1576	1
1	5	8.1	3	499	0
1	5	9.18	3.17	1562	1
2	5	8.77	2.77	2059	1
2	5	7.9	2.57	1194	1
2	5	8.33	2.99	1491	1
1	5	9.55	2.69	1192	0
2	5	9.45	2.73	1145	1

Ra_Data

sex	Inject Level	weight [kg]	activity density [μCi/kg]	time [day]	cause
1	0	8.03	0	3116	0
1	0	14.6	0	3675	0
2	0	11.4	0	2139	0
1	0	11	0	5284	0
1	0	6.57	0	4018	0
2	0	8.43	0	3182	0
1	0	11	0	3360	0
2	0	8.21	0	3361	0
2	0	11.7	0	1550	0
1	0	10.9	0	4698	0
2	0	10.2	0	4575	0
2	0	8.68	0	4283	0
1	0	12.3	0	4752	0
2	0	10.8	0	5725	0
1	0	12.8	0	4372	0
2	0	10	0	3677	0
1	0	12.5	0	5042	0
2	0	12	0	5321	0
2	0	8.42	0	4726	0
1	0	9.7	0	4890	0
2	0	9.9	0	4234	0
1	0	12.1	0	3907	0
2	0	10.6	0	4458	0
2	0	9.88	0	4690	0
2	0	9.9	0	4889	0
2	0	7.8	0	4657	0
2	0	11.7	0	4784	0
2	0	9.7	0	4828	0
2	0	9.8	0	3670	0
2	0	9.5	0	4509	0
2	0	11.8	0	3916	0
2	0	8.2	0	4929	0
2	0	8.9	0	4605	0
2	0	9.9	0	3185	0
2	0	9.41	0	5349	0
2	0	9.38	0	5066	0
2	0	8.86	0	5124	0
2	0	10.1	0	5281	0
2	0	9.17	0	4538	0
2	0	9.08	0	4403	0
2	0	11.1	0	4093	0
2	0	10.4	0	2788	0
2	0	10.9	0	5621	0
2	0	11.5	0	3821	0
2	0	8	0	33	0
1	0.2	9.77	0.00577	4518	0
2	0.2	8.1	0.00836	3448	0
1	0.2	10.8	0.00873	4102	0
2	0.2	8.9	0.00665	4190	0
1	0.2	11.8	0.00711	5056	0

sex	Inject Level	weight [kg]	activity density [μCi/kg]	time [day]	cause
2	0.2	11.5	0.00687	3101	0
1	0.2	12.9	0.00961	5000	0
1	0.5	11	0.0171	3676	0
2	0.5	9.75	0.022	5079	1
1	0.5	10.4	0.0263	4297	0
2	0.5	11.4	0.0205	4141	0
1	0.5	9.2	0.0215	4052	0
2	0.5	9.1	0.0197	4900	0
2	0.5	10	0.023	5059	0
1	0.5	13.2	0.0206	4526	0
2	0.5	8.8	0.0208	4281	0
1	0.5	12.3	0.029	3192	0
1	0.5	11.4	0.021	4310	0
2	0.5	9.4	0.0235	4393	0
2	0.5	11.7	0.0238	4797	0
1	0.5	13.4	0.0196	3219	0
2	0.5	10.1	0.0239	4180	0
2	0.5	10.1	0.024	3870	0
1	0.5	12.9	0.0194	3848	0
2	0.5	10.6	0.0212	4935	0
2	0.5	8.7	0.0217	4697	0
1	0.5	10.5	0.0196	5249	0
2	0.5	9.9	0.0215	4322	0
2	0.5	9.7	0.0212	4628	0
1	0.5	10.4	0.0205	4367	0
2	0.5	9	0.0201	4990	0
2	0.5	10.2	0.0202	5171	0
1	1	8.48	0.0618	5727	0
1	1	10	0.0876	4054	0
2	1	8.68	0.0576	3860	0
1	1	8.6	0.0642	2038	0
1	1	11.7	0.0436	3780	0
2	1	7.23	0.0584	5260	0
1	1	11.4	0.0651	3544	0
2	1	8.98	0.0559	2988	0
2	1	9.88	0.0521	4399	0
1	1	11.5	0.0573	4003	0
2	1	11.2	0.0522	5636	0
2	1	9.71	0.0444	3978	0
1	1	11.7	0.0527	3739	0
2	1	10.5	0.0701	1729	0
1	1	8.88	0.0797	893	0
2	1	8.99	0.0611	4557	0
1	1	11.4	0.0639	5601	0
2	1	10	0.0589	3625	0
2	1	11.6	0.0682	3612	1
1	1	10	0.061	4260	0
2	1	8.1	0.0633	5189	0
1	1	10.9	0.0861	4845	0
2	1	10.4	0.0712	3009	0
1	1.7	9.98	0.137	4438	0

sex	Inject Level	weight [kg]	activity density [μCi/kg]	time [day]	cause
2	1.7	13.1	0.165	3268	0
1	1.7	6.2	0.163	5495	0
1	1.7	10.1	0.151	4107	1
2	1.7	7.9	0.152	3432	0
1	1.7	7.17	0.163	3142	0
2	1.7	9.5	0.154	2577	0
2	1.7	7.55	0.168	3914	0
1	1.7	10.6	0.183	4903	0
2	1.7	8.17	0.165	5324	0
2	1.7	8.95	0.167	2399	0
1	2	8.74	0.382	3440	0
1	2	8.21	0.387	3775	1
2	2	8.53	0.347	4459	0
1	2	10.6	0.306	4368	0
1	2	11.5	0.267	4703	1
2	2	10.6	0.36	4615	0
1	2	11.1	0.413	3424	1
2	2	6.95	0.311	4781	0
2	2	9.38	0.317	3998	0
1	2	9.95	0.345	3569	1
2	2	9.3	0.31	3297	1
2	2	10.3	0.281	2948	0
1	3	8.91	1.2	2850	1
1	3	9.02	1.21	2226	1
2	3	7.74	1.11	2497	1
1	3	11.7	1.16	1917	1
1	3	13	0.846	2955	1
2	3	9.75	1.14	1932	1
1	3	12.3	1.29	2099	1
2	3	7.76	1.03	2612	1
2	3	8.02	0.987	2487	1
1	3	10.1	1.06	1737	1
2	3	12.9	0.938	1610	0
2	3	11.4	0.883	1897	1
1	4	9.08	3.51	1606	1
1	4	9.53	3.55	1884	1
2	4	7.2	3.1	1614	1
1	4	8.83	3.47	1518	1
1	4	13.2	2.42	1659	1
2	4	8.55	3.44	1939	1
1	4	9.55	3.88	1647	1
2	4	8.94	3.14	1324	1
2	4	8.53	3.02	1471	1
1	4	10.8	3.28	1553	1
2	4	10.4	2.84	1469	1
2	4	9.61	2.81	1435	1
2	4	10.5	2.5	303	0
2	4	9.09	2.96	1674	1
2	4	10.9	3.44	1521	1
2	4	7.93	1.98	636	0

sex	Inject Level	weight [kg]	activity density [µCi/kg]	time [day]	cause
1	4	11.2	3.04	1462	0
1	4	10.5	2.99	1365	1
1	4	13.2	3.65	952	0
1	4	12.2	3.61	516	0
2	4	10.7	3.88	1318	1
1	4	13.7	3.03	1680	1
2	4	10.9	3.2	1675	1
2	4	11.4	3.2	1182	1
1	4	10	3.21	702	0
1	4	13.2	3.21	1808	1
1	4	12.6	3.17	674	0
2	4	9.87	3.18	678	0
2	4	10.5	3.21	1662	1
2	4	9.43	3.25	1323	1
1	5	9.87	10.5	908	1
1	5	8.85	10.8	1380	1
1	5	8.9	10.6	1091	1
1	5	10.9	10.1	1220	1
2	5	9.66	10.2	1015	1
1	5	8.85	11.9	1288	1
2	5	7.76	9.68	968	1
2	5	9.16	9.48	1288	1
1	5	10.7	10.2	825	1
2	5	12.8	10.2	266	0
2	5	9.65	9.97	1219	1
2	5	10.8	6.31	419	0
2	5	7.52	9.22	420	0

VITA

Shili Xiao was born in Wuhan, Hubei Province, China on April 12, 1949. After she graduated from high school, she worked as a science teacher in a high school for eight years in Qianjiang, Hubei, China. In September 1978, she enrolled in the Physics Department of Huazhong Normal University, Wuhan, China. Soon after she obtained B. Sc. degree, she worked as a teacher in the Physics Department, South-Central University for Nationalities, Wuhan, China. In September 1984, she enrolled in the Physics Department of Huazhong Normal University, China, to pursue a Master's degree in Theoretical Physics. After graduating she continued to work as an assistant professor in the Physics Department of South-Central Universities for Nationalities, Wuhan, China. In May 1992, she joined the Department of Nuclear Engineering at the University of Tennessee, Knoxville, to study for her Ph.D. degree in Radiation Protection Engineering.