RESPONSE SURFACE MODELLING OF MONTE-CARLO FIRE DATA

A thesis submitted in fulfilment of the requirement
for the degree of
Doctor of Philosophy

by

Jianguo Qu, BEng. MEng.

Centre for Environmental Safety And Risk Engineering
Victoria University of Technology, Australia

2003

CONTENTS

CHAPTER 6

CHAPTER 7

RELIABILITY ANALYSIS IN DESIGN VIA MAXIMUM TEMPERATURE

CHAPTER 8

CHAPTER 9

CHAPTER 10

CHAPTER 11

DECLARATION

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university and, to the best of this candidate's knowledge and belief, it contains no material previously published or written by another person except where due reference is made in the text of the thesis.

Signature: ______________________

Jianguo Qu

February 2003

ACKNOWLEDGEMENT

I wish to acknowledge and express my sincere gratitude to the following people:

Thanks to my principal supervisor, Professor A. M. Hasofer, for his continued guidance, supervision and help throughout this research work and at all stages during the preparation of this thesis.

Thanks to my second supervisor Professor Ian Thomas, for his encouragement and help when needed during the research.

Thanks also to staff who work in the Centre for Environmental Safety And Risk Engineering, Victoria University, for their kindly support during the study.

Thanks to my wife Ying Chen and my son Brian Qu whose love, faith and patience enabled me to complete this thesis.

NOMENCLATURE

A	Area of compartment opening (m^2)
A_v	Burn area at time t (m^2)
A_{vo}	Initial burn area (m^2)
A_w	Total wall and ceiling surface area in the compartment (m^2)
a	Molar stoichiometric coefficient for CO
b	Molar stoichiometric coefficient for CO_2
C	Flame heat transfer modulus (m$^{3/2}$s$^{1/2}$/W)
C_D	Orifice coefficient for compartment ventilation
c_p	Specific heat of gas or fuel (kJ/kg K)
c_W	Specific heat of wall (kJ/kg K)
g	Gravitational constant (m/s^2)
h_L	Height of interface between the hot and cool gas layers (m)
H_o	Height of the doorway or window opening (m)
h	Convective transfer coefficient
ΔH_c	Heat of complete combustion (J/kg fuel)
ΔH_v	Heat of vaporization (J/kg fuel)
K	Extinction coefficient (1/m)
k	Thermal conductivity (W/m K)
k_G	Gas absorption coefficient (1/m)
k_{GO}	Constant derived from experiments
L	Compartment length (m)
m	Mass flow rate (kg/s)
m_a	Mass flow of gases leaving the compartment (kg/s)
m_{ideal}	Free vaporization/pyrolysis rate of the fuel (kg/m^2 s)
m_o	Initial mass of the fuel (kg)
m_c	Mass of fuel that has been consumed by the fire (kg)
Q_c	Heat release rate of the fire (W)
Q_O	Heat loss rate through vent openings by radiation (W)
Q_r	Heat loss rate due to fuel vaporization and heating (W)
Q_v	Heat loss rate through vent openings by convection (W)
Q_W	Heat loss rate through the walls by conduction (W)
q	Heat transfer rate (W/m^2)

$q_{o,ig}$	Minimum external heat flux required to ignite the fuel (W/ m^2)
q_r	External heat flux to the fuel (W/ m^2)
R	Fuel vaporization rate, or mass loss rate (kg/s)
R_m	Maximum burning rate for smouldering fire (kg/s)
r	Radius of area (m)
Δr	Burning rate enhancement due to heat radiation (kg/m^2 s)
T	Temperature of effluent gases (K)
T_W	Wall temperature (K)
T_s	Fuel surface temperature (K)
T_{WI}	Inner wall temperature (K)
T_{WO}	Out wall temperature (K)
t	Time (s)
V	Compartment volume (m^3)
V_{fo}	Lateral flame speed produced by radiation (m/s)
V_f	Actual flame speed as limited by the available oxygen (m/s)
W_o	Width of the compartment opening (m)
X	Moles of water produced per mole of carbon burned
x	*x* dimension (m)
Y	Mass fraction of O_2, CO, CO_2 or vapour, identified by subscripts
Y^o	Mass fraction of O_2, CO, CO_2 or vapour from the previous timestep, identified by subscripts
δ	Wall thickness (m)
ε	Gas emissivity
ϕ	Compartment equivalence ratio, shape factor
γ	Stoichiometric air to fuel ratio
μ	Combustion efficiency
μ_o	Maximum possible combustion efficiency for fuel
ρ	Density of gas, wall or fuel material (kg/m^3)
ρ_o	Gas density at ambient conditions (kg/m^3)
ρ_W	Density of material of wall (kg/m^3)
σ	Stefan Boltzmann constant (W/m^2 K^4).

ABSTRACT

Deterministic computer fire models have progressed over recent years to the point of providing good predictions for some parameters of fire behaviour. However, input data are not always available, and many factors that affect the course of a fire are probabilistic in nature and cannot be determined from physics.

One way of surmounting the problem of unavailability of the values of the input parameters is to take them as random variables. By specifying an unsafe region in the output space and calculating its probability, we can obtain a figure for the reliability of the design being tested, in terms of the probability of the unsafe region. In practice, evaluation of the probability distribution of the output space cannot in general be carried out analytically because of the complexity of the computer fire models. An alternative method is to use Monte-Carlo simulation. But it usually requires a large amount of calculation to reach sufficient accuracy, particularly if the probability of the unsafe region is small, as it should be if the design is to be reasonably reliable. Also, if the probability distribution of the input is changed, the whole Monte-Carlo simulation must be redone *ab initio*.

An approach that has been recently advocated in the structural reliability context is that of the response surface method. It consists in representing each output parameter by a nonlinear function of the input parameters. Usually, a quadratic function of the input parameters turns out to be sufficient. Fitting of the response surface is carried out by regression. However, if the range of the input parameters is comparatively large, it is unlikely that one quadratic function will fit the whole range. It then becomes necessary to break up the full range of input parameters into smaller subranges and fit a quadratic function separately to each subrange.

In this thesis, the results of a large scale Monte-Carlo simulation of a computer fire model, CESARE-Risk model, are summarized in the form of a simple response surface for each of a number of subranges of the input parameters. The subranges are automatically determined through the use of a powerful modern regression methodology *before* the linear multiple (quadratic) regression. A brief summary of the

First Order Second Moment Reliability Index Method is given. It is shown that the particular form of the obtained response surface allows the reliability index to be easily calculated. The reliability index and corresponding probability of failure are obtained for particular examples and the results confirmed by Monte-Carlo simulation.

CHAPTER 1
INTRODUCTION

1.1 Project Background

Research into fires in buildings is a comparatively recent activity and only in the last few decades has a substantial effort been mounted. Accordingly, this area of research can be characterized as being broad and fertile for new research. Since research into fire and its effects in buildings involves many disciplines, the previous research effort can be characterized as being conducted in many disparate areas.

Fire safety design has been highly reliant on prescriptive rules in building codes. This is particularly the situation for occupant safety in the case of fire. Regulations usually state in detail what measures should be taken in order to accomplish a minimum occupant fire safety level. For a review of the history of prescriptive codes and references see [1].

However, there are some deficiencies associated with this type of regulations. They are rather inflexible if not applied to a standard type building. Prescriptive regulations could lead to a safety level that may be too low in some buildings, or it may lead to an unnecessarily expensive design [2][4].

Safety can be ensured either by comparing the proposed design with accepted solutions, or by using design values in the calculations that are based on a specified level of risk. Therefore, an advanced engineering methodology for the cost – effective design of fire safety and protection in buildings has been proposed and widely accepted [5]. The risk analysis should incorporate an uncertainty analysis because many variables of fire are associated with uncertainty[3][71][72][73]. However, a detailed methodology for implementing a realistic risk analysis, which can nevertheless be used by practising engineers, has not been developed to date. The proposed research is intended to provide a useful tool towards this goal.

1.2 Problem Definition

The aim of this research project is the identification some aspects of probability-based indices of safety for use by practising engineers in comparing competing designs.

This fundamental research is required to support the implementation of an advanced engineering methodology for the cost-effective design of fire safety and protection in buildings.

The proposed research will bring the methodology of risk analysis in the design of fire safety for apartment buildings in line with risk analysis as practised by the civil engineering profession at large, namely the beta reliability index[72][74][75]. To date, only small scale simplified models have been analyzed in this way. What is proposed here is to carry out a full - fledged analysis of the CESARE-Risk fire and smoke spread model with stochastic input, using advanced regression analysis methods, and to set out and test a detailed methodology for evaluating the reliability index for any set of limiting states required. This methodology will then be available for the analysis of any other computer models of fire spread developed for modelling specific fire safety situations.

1.3 Overview of this Thesis

The research presented in this thesis is mainly concerned with a simplified form of response surface for each of a number of subranges of the input parameters of a large scale Monte-Carlo simulation of a computer fire model, the CESARE-Risk model. Also, the particular form of the obtained response surfaces allows the reliability index to be easily calculated. Thus, an advanced engineering approach for the risk-based design of fire safety in buildings is developed, providing a feasible and cost-effective methodology, which can be used by practising engineers in comparing designs.

The existing computer fire models, their recent development and application, as well as their limitations are discussed in Chapter 2.

Chapter 3 gives the background of the CESARE-Risk model and provides a complete theoretical discussion of the CESARE-Risk model and a description of the model's assumptions. Details of the four scenarios, as well as input and output variables of the CESARE-Risk model are also introduced in Chapter 3.

Chapter 4 gives a brief overview of modern regression methods, and more details of the ACE (Alternating Conditional Expectations) and AVAS (Additive and Variance

Stabilizing Transformation) regression methodologies, which are used throughout the research.

Chapter 5 gives a theoretical discussion of response surface methods, the calculation of the reliability index and the probability of failure in fire engineering. A brief outline and a theoretical discussion of the Monte-Carlo method and applications in fire engineering are also given in Chapter 5.

In Chapter 6, through using the modern regression method, AVAS, we analyze and identify variable transformations for the maximum temperature reached under different events of the CESARE-Risk model. A simple response surface is derived which can be used for reliability design, for each of a number of subranges of the input parameters. The particular shape of the regression equation derived in Chapter 6 makes the task of finding the design point[75] and reliability index very simple.

Details of finding the design point and reliability index using Lagrange's method of undetermined multipliers, are given in Chapter 7. Also, the reliability index for maximum temperature reached for specific examples of the four scenarios are calculated. The corresponding probability of failure for each of scenarios is obtained by use of the First Order Second Moment (FOSM) Method[75] and results validated by Monte-Carlo simulation.

In Chapter 8, another output variable, time to untenable conditions, is analysed. Using the modern regression method, ACE, we analyze and identify variable transformations for the time to untenable conditions under different events of the CESARE-Risk model. A simple response surface is derived, which can be used for reliability design for each of a number of subranges of the input parameters.

In Chapter 9, the reliability index for the time to untenable conditions for specific examples of the four scenarios of the CESARE-Risk model are calculated. The corresponding probability of failure for each of the scenarios is obtained by the use of the First Order Second Moment Method and results validated by Monte-Carlo simulation.

In Chapter 10, we use a logarithmic fit to the time to untenable conditions. The reliability indexes of some specific examples for the four scenarios are calculated. Also, the corresponding probability of failure for each of them is derived using the FOSM Method and the result is confirmed by Monte-Carlo simulation. A comparison of the reliability index derived from the ordinary fit and the index derived from a logarithmic fit to the time to untenable conditions is carried out.

Chapter 11 presents the conclusion and further research directions.

In order to carry out the risk analysis procedures a number of S-Plus functions have been developed (see APPENDIX). These functions are described as follows:
Calculate the correlation between original outputs and the predict values of the output,
Calculate the coefficients for modern regression,
Calculate the reliability index and the probability of failure for design engineers,
Carry out Monte - Carlo simulations for the response surfaces that were derived in the research, and
Draw most figures through the whole thesis.

CHAPTER 2
LITERATURE REVIEW

2.1 Survey of Computer Models for fire and smoke

There are many models available that give estimates of fire growth and fire spread. They can be divided into two categories: deterministic models and non-deterministic models.

2.2 Deterministic Models

Deterministic Models are the models that give an output without considering the possibility that given the same situation the estimates could change. The major drawback of deterministic models is that they do not take into account the randomness of fire phenomena. Typical deterministic models are Zone Models, Field Models and Network models.

2.2.1 Zone Models

Table 2-1 lists 31 zone models relating to a fire in a compartment. These models come from 10 countries. Twenty of them deal with only a single vented compartment, and the other 11 treat multiple interconnected compartments. Two models emphasize post-flashover; the others generally present the history of the fire both before and after flashover. The user must be able to input a good deal of information about the heat release rate of the fire in all cases. Twenty-five of these models are designed to run on a personal computer. The underlying physical assumptions of most of these models have a great deal of similarity. Some of the models, notably Hazard I, go further than others in predicting the consequences of a fire, such as the survival of building occupants [41].

A well known Zone Model is the NRCC (NRCC1 in Table 2-1) (National Research Council of Canada) Model. The NRCC model was developed by Takeda and Yung in 1992 [6]. This one-zone fire growth model can be used to predict fire growth characteristics and species concentrations.

Model	Country of origin	Run on PC	Comments
ARGOS	Denmark	Yes	Multi-compartment
ASET /ASET-B	U.S.A.	Yes	One room/ (BASIC source code)
BRI-2	Japan	Yes	Multi-compartment
CCFM.VENTS	U.S.A.	Yes	Multi-compartment
CFAST	U.S.A.	Yes	Multi-compartment
CFIRE-X	Ger./Nor.	Yes	One room
CiFi	France	No	Multi-compartment
COMPBRN-III	U.S.A.	Yes	One room
COMPF2	U.S.A.	Yes	Post-flashover
DACFIR-3	U.S.A.	No	Aircraft cabin
DSLAYV	Sweden	Yes	One room
FAST	U.S.A.	Yes	Multi-compartment
FIRAC	U.S.A.	No	Uses FIRIN, complex vent. systems
FIRIN	U.S.A.	No	Many rooms, ducts, fans, filters
FIRST	U.S.A.	Yes	One room
FISBA	France	No	One room
FPETOOL	U.S.A.	Yes	One room
HarvardMarkVI	U.S.A.	Yes	Multi-compartment
Hazard I	U.S.A.	Yes	Includes FAST and other models
HEMFAST	U.S.A.	Yes	Furniture fire in room
IMFE	Poland	Yes	One room; multiple vents
MAGIC	France	No	Multi-compartment
NRCC1/ NRCC2	Canada	Yes	One room/For large office spaces
OSU	U.S.A.	Yes	One room
POGAR	Russia	Yes	One room
R-VENT	Norway	Yes	One room
SFIRE-4	Sweden	Yes	Post-flashover
WPI-2	U.S.A.	Yes	One room
ZMFE	Poland	Yes	One room

Table 2-1: Zone models for compartment fires

The NRCC fire growth model is a simplified one-zone model for single room fires. It treats the fire room as a well-stirred combustion chamber and assumes uniformly distributed quantities inside the room. Further, Victoria University of Technology has undertaken research in conjunction with the NRCC to develop recommendations for timber-framed apartment buildings based on risk assessment work [7]. This work has been accepted by the authorities and included in the Building Code of Australia.

Cooper and Yung [8] have improved the NRCC Fire Growth Model for Apartment buildings.

Zone models cannot provide detailed information on fluid flow but their simplicity, ability to run rapidly on computers, ease of transfer from one organization to another, and low cost make them attractive. Zone models can be used for multiple compartments.

2.2.2 Field Models

There are other models that describe phenomena that occur in two (or three) dimensional spaces, called Field models [9]. This kind of model involves dividing the enclosure by two (or three) dimensional grids into elements. Field models can model the differences in physical parameters throughout the grid. The physical parameters could be temperature, species concentrations, etc. Table 2-2 shows 10 field models for compartment fires.

Model	Country of origin	Comments
BF3D	U.S.A.	Treats buoyant heat-driven flow
FISCO-3L	Ger./Nor.	One room - run on PC
FLOW3D	U.K.	General fluid - dynamics code
JASMINE	U.K.	Uses PHOENICS - treats radiation
KAMELEON E-3D	Norway	One room
KAMELEON II	Norway	Multi-compartment
KOBRA-3D	Germany	One room - no turbulence-runs on PC
PHOENICS	U.K.	General fluid - dynamics code
RMFIRE	Canada	One room - 2D - B. F. C.
UNDSAFE	U.S.A./Japan	Treats buoyant, heat - driven flow

Table 2-2: Field models for compartment fires

In Table 2-2, two of these (FLOW3D and PHOENICS) are general fluid dynamics codes which are usable as basic elements of models treating fire specifically. All these models except two rather limited ones (FISCO-3L and KOBRA-3D) require a much more powerful computer than a PC, and indeed could effectively use the most powerful computer available. The various field models originate in U.K., Norway, Germany, U.S.A., Japan and Canada [41]. More recent development of field modeling are described in references [68][69].

Field models do not make simplifications like Zone models, and they solve for the governing flow equations in each cell. The advantage of a field model is that it can provide detailed information on the fluid motions. It is normally suitable for those problems where only one compartment is considered [10]. So it is often used in fundamental research to study some specific aspect of building fire.

Overall, both Zone and Field models are based on conservation equations for mass, momentum and energy for solving for the variables of interest, such as temperature and gas concentrations. Both of them adopt basic sub-models to model heat release rate, mass release rate, radiation and so on.

2.2.3 Network Models

Network modelling has been used to solve fire and smoke protection problems [12,13]. Networks are made up of nodes connected by links. The building is divided into compartments (nodes). The temperature, pressure and species concentration in each of the nodes is assumed to be uniform. The nodes represent space and also the smoke and/or fire conditions of the space. The nodes are connected by leakage opening (flow paths). The links are the possible movement of the fire/smoke from space to space. The mass flow rates and pressure differences are related by the orifice flow equation. The network modelling technique uses the mass balance and flow equations, and expressions for temperature and smoke concentration. Network and graph theories have been used successfully for studying multi-compartment buildings.

The reduction in cost of computers has encouraged the use of network modelling. Network models can predict conditions in many rooms and locations far away from the source of fire. They are most suitable for high-rise buildings.

2.3 Non-deterministic/Probabilistic Models

It is now generally accepted that the widely used deterministic approach to fire safety design is not cost effective, for the following reasons:

- The consideration of fire scenarios in isolation
- The built-in arbitrariness of the choice of safety factors

Probabilistic models are models that give estimates of the outcome of a fire phenomenon while considering the uncertainties in the process. This kind of model yields the relative frequency of occurrence of each pattern of growth of fire and spread of smoke in a large number of real fires. In the words of Ramachandran [11], probability modelling is concerned with final outcomes rather than the detailed knowledge of the processes that make it.

Ling and Williamson [12] have used network modelling to solve fire and smoke protection problems. They proposed the use of probabilistic networks in analyzing the spread of smoke and the egress of people in buildings. Calculation of the probability of occurrence for the fire scenarios was based on Mirchandani's algorithm. The NRCC smoke spread model can be classified as a network model too.

The work of researchers, such as Ling and Williamson [12] [13], Beck [5] [14], Takeda and Yung [6], Hasofer and Beck [15], Beard [16] and more recently the work at the Centre for Environmental Safety and Risk Engineering, Victoria University of Technology, has focussed on the development of an integrated system model to predict the performance of building fire safety. These researches have developed new concepts, which have enabled the performance of the building fire safety system to be analyzed and quantified. In addition, they have enabled previous research results to be used (where appropriate) to predict the performance of various sub-elements comprising the system model. However, the nature and structure of the system model has required, in many cases, that new research be conducted to enable predictions of the performance of various sub-elements of the system to be made.

Stochastic fire and smoke spread models have been developed by Beck [14], Hasofer and Beck [15] Ramachandran [17], Takeda and Yung [6], He [18]. But existing models are still extremely limited in their ability to accurately predict the levels of risk to life safety.

The current fire safety system model, called CESARE-Risk model, is based on the original paper by Takeda and Yung [6], a recent report by Cooper and Yung [8], and the modification to the model by researchers at the Centre for Environmental Safety and Risk Engineering, VUT (see draft report He [18] December, 1998, Centre for Environmental Safety and Risk Engineering). It has been used to generate temperature and smoke data for an integrated system model which incorporates many aspects of a building - fire situation, such as sprinkler and alarm response, smoke spread, human behavior and egress, fire brigade response, structural failure, etc. [18]. It is used to predict the performance of building fire safety systems and to identify cost-effective fire safety system designs for buildings.

The CESARE-Risk model is based on a Probabilistic Risk Analysis (PRA) foundation. PRA consists in using statistical analysis to estimate the relevant variability measures and then to use them within the framework of a stochastic model. The ultimate target is to choose the design that will fulfill the required reliability requirements at minimal total expected cost.

2.3.1 The use of random input parameters

The recent research on Fire Safety by Hasofer and Beck 1997 [15], introduced random input parameters to derive a stochastic model for compartment fire in buildings from basic physical laws. It consists of just three variables, which form a Markov vector satisfying a stochastic differential equation. The deterministic version of the model can be calibrated to closely mimic the results of the more elaborate models. In the paper [15], the model used as a basis for the physical background as well as for calibration is the growth model developed by NRCC and described in [6]. It uses a simplified one-zone approach and was developed for a risk-cost assessment model for apartment houses.

2.3.2 Early use of response surface in fire spread model

It is important to understand that performance requirements in the fire safety area are expressed in terms of risk; either explicitly or implicitly. Fundamental questions remain to be discussed and eventually decided upon such as: how do we evaluate risk? How do risk evaluation methods differ when we look at different levels of design such as the whole building level, the subsystem level and the one-component level? What is the link between risk calculation procedures and a deterministic design format based on safety factors or partial coefficients? Calculation of risk means calculations based on models and parameters characterised by uncertainty, usually described by statistical distributions. To what extent are necessary data available for well-defined classes of buildings? What are the differences in design procedures when we are considering on the one hand a well-defined class of building and, on the other hand, a single complex building with a unique design layout and unique fire safety solutions?

To answer the above questions, Magnusson et al. [24] and Frantzich [4] have analyzed evacuation life safety in a one-room public assembly building. Limit state equations have been defined, using response surface approximations of output from computer programs. The research made a first attempt to carry out an uncertainty analysis and safety checking.

In the paper [24], Magnusson et al. illustrated the various methods and approaches, which included the analytical first-order second moment method and the standard probability risk analysis method, by showing calculations and results for an actual design problem. They concentrated on risk assessment methods taken from the area of structural engineering, from the area of large-scale technological systems, and from environmental engineering. Input parameter distributions were subjectively quantified and classified with respect to category: knowledge or stochastic uncertainty. Risk assessment results comprised probability of failure, reliability index and complementary cumulative distribution function for evacuation time-margin deficit[3][71][72][73][74]. Of special interest is the calculation of confidence intervals for the distribution of complementary cumulative distribution functions obtained by the two-phase Monte Carlo sampling procedure, allowing a distinction between knowledge and stochastic uncertainty. The important analysis carried out analytically gives data of fundamental significance for an understanding of the practical design problem.

2.3.3 Reliability analysis in the room of fire origin

Hasofer and Beck [42] present a partial safety factor approach to the problem of evaluating competing designs for fire safety in the room of fire origin in a building. A partial safety factor is defined as the ratio of the design value to the characteristic value for a load type variable and its inverse for a resistance type variable. The safety criterion considered is the expected number of deaths in the room or, alternatively, the probability of any death [74]. A death is assumed to occur when the time between the occurrence of the alerting cue and the onset of untenable conditions is shorter than the time to evacuation.

First, a safety index is obtained, based on the means and standard deviations of the logarithms of time between the occurrence of the alerting cue and the onset of untenable conditions and the time required for evacuation.

It is further shown that there are theoretical reasons as well as empirical evidence for assuming that the time between the occurrence of the alerting cue and the onset of untenable conditions and the time required for evacuation both have approximately a lognormal distribution. There is then a direct connection between the probability of death and the safety index, which leads to a rationale for selecting appropriate values of the index.

Tat [21], Hasofer, Beck and Odigie [22], Hasofer and Odigie [23], show that it is possible to set up stochastic process models to carry out a risk analysis of fire safety systems.

This research project will build upon the work of Professor V. R. Beck and his colleagues in the context of an Australia Research Council (ARC) Grant entitled “Modelling Non-Stationary Stochastic Processes of Fire and Smoke Spread in Multi-storey Buildings for Cost - Effective Risk-Based Design”.

CHAPTER 3
THE CESARE-RISK MODEL

3.1 Overview of the CESARE-RISK Model

The stochastic behaviour of fire was analysed in collaboration with research workers who are currently undertaking projects dealing with fire at CESARE. This was achieved by feeding a stochastic fire load input into a deterministic model to obtain a probability distribution of outputs.

The description of this model and its general assumptions are briefly as follows:
The purpose of the apartment fire growth model is to simulate the ignition and growth of fires in an apartment unit in order to help assess the fire safety performance of apartment buildings. This assessment is done on the basis of the amount, temperature and concentration of the gases generated by the model fire, the response speed and effectiveness of various fire protection systems, and the behaviour of building occupants.

The apartment fire growth model calculates the characteristics of compartment fires that have the greatest impact on occupant safety and building damage. These characteristics fall into two categories: the smoke and fire hazard category and the detection category. The former category data include the composition, temperature and flow rate of compartment effluent gases, and this information can be used to estimate the potential for smoke spread and fire damage outside the compartment of fire origin. The latter category information includes the time of occurrence of specific fire-detection - related events, such as the time the person in the room of fire origin first notices the fire, the smoke detector activation time, the sprinkler activation time, the time of flashover and the time to fire burnout. These times can be used to determine the occupant response and evacuation time [18].

The model uses standard flexible polyurethane foams to represent the upholstered furniture and bedding typically found in an apartment. Flame spread and fire growth over other fuel materials can also be simulated. All physical parameters associated with foam may be changed to represent other fuels. However, the combustion chemistry is specifically formulated to describe the products of combustion produced by polyurethane under a range of enclosure conditions. The ventilation conditions simulated by the fire growth model include natural ventilation through door and window openings, which may be either open or closed, and forced ventilation from an air handling system, such as air conditioning or smoke extraction.

3.2 Assumptions of the CESARE-RISK Model

To assess the fire safety performance of a building, a practical fire growth model for this application needs to be relatively simple since a large number of calculations are required. The aim in fire growth modelling is to develop a model that is simple enough to have a practical execution time without making undue sacrifices in accuracy [6, 18]. Therefore, the following assumptions are introduced.

3.2.1 One - zone model

Compartment fires are characterised by a hot upper layer caused by buoyancy effects and a relatively cool lower gas layer. The height of the interface between these two layers is time-dependent and decreases as the fire progresses, as does the layer temperature difference. Therefore, two zone models, which treat the upper and lower gas layers separately, are often used to represent the compartment gas temperature. If the compartment is under-ventilated (such as in door closed scenario), these models have impractically long computation times for this application and often predict premature fire extinction. The latter is due to the rapid descent of the upper layer predicted by two-zone models under closed-door conditions, which suffocates the fire by reducing the inflow vent area [6]. An additional problem posed by using two-zone models is that the modelling of flame spread over fuel surfaces becomes very complicated. A two-zone model incorporating flame spread would require the development of a moving plume sub-model. Because the fire plume is a complex phenomenon the development of a moving plume sub-model would be difficult, time consuming and unnecessarily accurate.

In order to eliminate the difficulties presented by two-zone models, a one-zone model, which incorporates lateral flame spread over fuel surfaces to simulate the growth of a fire and calculates a single, transient gas temperature for the compartment, is employed. This significantly reduces computation time and allows a more conservative estimate of under-ventilated fires. The following conditions and assumptions are employed:

(1) The ceiling, walls and floor of the compartment are fire separations.

(2) The compartment is small (1 to 2 average-size residential rooms).

(3) The compartment gases are well mixed (at uniform temperature and pressure).

(4) Flow through multiple compartment openings is weighted by area.

(5) The compartment wall temperatures are uniform and equal.

As the tool (CESARE-Risk) is basically developed to calculate the risk to occupants it is therefore most relevant for the pre-flashover situation. For this case the two-zone model usually gives a better prediction of the conditions and even better predictions will have with a CFD. However, the statistical methodology developed in this thesis is not restricted to the one-zone model only. The method suits all models.

3.2.2 Heat transfer mechanisms

Heat transfer from both of the compartment to the rest of the building and from the compartment to the burning fuel occurs through radiation, convection, and conduction. In this model, the compartment floor is treated as an adiabatic boundary. Heat transfer to fuel is assumed to occur mainly by radiation. Heat losses through the compartment boundaries take place by radiation, convection and conduction through the compartment walls, openings and ceiling.

3.2.3 Furniture arrangement

The arrangement of furniture in the compartment can give rise to an infinite number of possible fire scenarios, the statistical occurrence of which would be quite difficult to model. Therefore, a worst-case arrangement is assumed and the furniture is modelled as a single mass in the compartment and results in a conservative (from the safety point of view) estimate of fire severity. In general, the upholstered surfaces of the furniture determine the progress of combustion; thus the combustion properties of flexible polyurethane foam are used as a benchmark to simulate the combustion behaviour of apartment furniture. Some of the parameters are made adjustable to cater for variations in flame-spread rate, heat of combustion, etc [18].

The furniture is assumed to exist as a single mass in the center of the room and to possess uniform properties. In the model, the size of the fuel mass reflects the amount of ignitable combustible in the room. Therefore, large fuel masses are used to simulate flashover fires, in which all room furnishings ignite, whereas smaller fuel masses are used to simulate the burning of isolated furnishings caused by flaming fires.

3.2.4 Material properties

Many material properties, such as heat capacity, thermal conductivity and density, vary with temperature. In the temperature range normally experienced by the materials in the fire compartment (20°C-1200°C), however, only the gas density changes significantly. Therefore, for the purposes of simplicity, all material properties are assumed to remain constant at their ambient values except the gas density. Since heat capacities and thermal conductivities rise with temperature, this assumption is expected to result in conservative predictions of fire severity.

3.2.5 Fire detection/suppression

The fire growth model does not calculate the effects of fire suppression since the activation of the fire devices is often difficult to predict as the time required for a given device to activate depends on its location and sensitivity.

3.2.6 Other assumptions

Other assumptions specific to the model are shown in the following modelling equations (Symbols and units are fully defined in section NOMENCLATURE).

3.2.6.1 Compartment ventilation

The compartment ventilation rate m_a is dependent on buoyant forces created by temperature differences across the compartment openings. During the ventilation controlled state of well developed fires, it determines the rates of combustion, species production and heat release.

The upper half of the compartment fills with hot gases and the temperature rises in this hot layer as the fire progresses, causing an increase in pressure that drives hot gases through the upper half of the compartment opening, while cool, dense air enters from below. The following relation from Steckler et al [49] models this mechanism for gas flow into the compartment (this formula is also used to calculate gas flow out of the compartment in the Cesare-Risk model because they are very close in practice even though they are not the same theoretically):

$$m_a = \frac{2}{3}\sqrt{2g}C_D\rho_o A_o\sqrt{H_o}\left[\left(1-\frac{T_o}{T}\right)\frac{T_o}{T}\right]^{0.5}\left(1-\frac{h_L}{H_0}\right)^{1.5} \qquad (3\text{-}1)$$

where C_D is the orifice coefficient for the compartment opening,, H_o is the total height of the opening and h_L is the height of the interface between the hot and cool gas layers, g is the gravitational constant, ρ_o is the gas density at ambient conditions, T_o is the temperature outside the compartment, T is the gas temperature inside the compartment (since the model is a one zone model employing a single room temperature), A_o is the area of the opening.

The interface between the hot and cool air masses passing through the compartment opening is assumed to be at 0.5 H_0 (halfway up the compartment opening). The Cesare-Risk model uses this assumption for simplicity even though the location of the neutral layer can be very easily calculated [70]. In the simplified model, an average flow coefficient of 0.7 is used for both inflow and outflow.

3.2.6.2 Flame spread

The rate of flame spread from the point of ignition directly affects the fuel mass loss and burning rates. The current compartment flame spread model assumes that the lateral flame spread rate depends on the net external radiative heat flux to the combustible and the oxygen (O_2) concentration in the compartment [6]:

$$V_f = \begin{cases} V_{fo}\sqrt{\dfrac{Y_{O2i}-0.11}{0.12}} & for Y_{O2i} > 0.11 \\ 0 & for Y_{O2i} < 0.11 \end{cases} \qquad (3\text{-}2)$$

where Y_{O2i} is the oxygen mass fraction in the compartment and V_f is the lateral flame spread velocity. V_{fo} is the radiation-dependent flame velocity, given by:

$$V_{fo} = \frac{C^{-2}}{(q_{o,ig}-q_r)^2}. \qquad (3\text{-}3)$$

where $q_{o,ig}$ is the minimum external heat flux required to ignite the fuel (W/ m^2), q_r is external heat flux to the fuel (W/m^2), C is the flame heat transfer modulus (m$^{3/2}$s$^{1/2}$/W).

Heat fed back to the fuel by the compartment enclosure can be expressed as:

$$q_r = \sigma[\varepsilon T^4 - (1-\varepsilon)T_W^4 - T_S^4] \qquad (3\text{-}4)$$

where σ is the Stefan Boltzmann constant (W/m^2 K^4), ε is the gas emissivity, and *T, T_W* and T_S are the gas, inner wall and fuel surface temperatures, respectively.

$$\varepsilon = 1 - \exp(-k_G L) \qquad (3\text{-}5)$$

Where L is the compartment length and k_G is the gas absorption coefficient, which is assumed to vary linearly with the product gas concentration

$$k_G = k_{GO} Y_{PRO}\,. \qquad (3\text{-}6)$$

Y_{PRO} is the mass fraction of product gases and k_{GO} is a constant derived from experiments.

The ignited area of the fuel is considered to be roughly circular and thus grows according to the relationship

$$A_v = \pi r^2 = \pi\left(\sqrt{A_{VO}/\pi} + \int V_f dt\right)^2 \qquad (3\text{-}7)$$

where A_{VO} is the initial burning area and r is radius.

3.2.6.3 Mass loss rate for flaming fires

The distinction between the fuel mass loss rate and the fuel burning rate is that the first refers to all of the vapour that is driven from the fuel, whereas the second refers only to the portion of evolved vapour that is converted into products. The mass loss rate R of the fuel for flaming fires depends on the ignited area, the concentration of oxygen in the burning environment and the radiative heat flux to the fuel [18].

$$R = \left(m_{ideal} \frac{Y_{O2i}}{0.23} + \Delta r \right) A_v \qquad (3\text{-}8)$$

where m_{ideal} is the free vaporization/pyrolysis rate of the fuel (kg/m^2 s) and Δr is the enhancement to the mass loss rate due to heat radiation by the compartment walls:

$$\Delta r = \frac{q_r}{\Delta H_v}, \qquad (3\text{-}9)$$

where ΔH_v is the heat of vaporization (J/kg fuel).

Cooper and Yung modified the above R by taking into account the reduction in mass loss rate due to oxygen concentration and fuel consumption as follows:

$$R = \left(m_{ideal} \frac{Y_{O2F}}{0.23} + \Delta r \right) A_v \left(\frac{m_o - m_c}{m_o} \right)^n \qquad (3\text{-}10)$$

where Y_{o2F} is the mass fraction of post-combustion oxygen, m_o is the initial mass of the fuel and m_c is the mass of fuel that has been consumed by the fire. $n = 1$ at CESARE-RISK model for CESARE-Risk's experimental building fire facility instead of Cooper and Yung's $n = 2$ (see [18]).

3.2.6.4 Heat release rate for flaming fires

The heat release rate of the fuel is determined by the mass loss rate for fuel controlled fires, and by the ventilation rate for ventilation controlled fires. It is given by the equation:

$$Q_c = \begin{cases} \Delta H_c R\mu \quad , & \phi < 1, \\ \Delta H_c \min(R, m_a / \gamma)\mu, & \phi > 1, \end{cases} \tag{3-11}$$

where ΔH_c is the heat of complete combustion (J/kg fuel), γ is the stoichiometric air to fuel ratio. The combustion efficiency μ is estimated as follows:

$$\mu = \begin{cases} \mu_o, \; for \phi < 1, \\ \mu_o / \phi, for \phi > 1. \end{cases} \tag{3-12}$$

The compartment equivalence ratio ϕ is defined as the normal fuel vapour to oxygen mass ratio present in the compartment. This equivalence ratio gives a more conservative estimate of combustion efficiency, and prevents premature extinction of the fire when there are no openings, or only a small amount of leakage. This eliminates the dependency of ϕ on the ventilation rate in the early stages of the fire and is thus more likely to give low predictions of ϕ for fires in sealed compartments. ϕ is calculated by considering the mass ratio of vaporized fuel to oxygen present in the compartment:

$$\phi = \frac{0.23\gamma(Y^o_{VAP}\rho V + R\Delta t)}{Y^o_{O2F}\rho V + 0.23 m_a \Delta t} \tag{3-13}$$

where ρ is the gas density, V is the compartment volume, t is time and $Y^o{}_{VAP}$ and $Y^o{}_{O2F}$ are the mass fractions of vaporized fuel and oxygen, respectively.

3.2.6.5 Species concentrations

The species being considered in the fire growth model are oxygen, the product gases (mainly CO and CO_2) and unburned fuel vapour. The concentrations of these species depend on reaction stoichiometry and the ventilation, fuel mass loss and burn rates. Since the model consists of time-discretised equations, suitable average oxygen, product gas and fuel vapour concentration value for each time-step must be calculated. Treating the compartment as a well-stirred batch reactor allows the calculation to be carried out. The well-stirred batch reactor is filled at the beginning of each time step and emptied to its initial volume just before the end of each time-step, requiring that the concentrations be calculated on the basis of the total mass of gas contacting the compartment over a given time-step. The following sections give the detail of the calculations.

3.2.6.5.1 Oxygen concentration

The model calculates two oxygen concentrations. One is the pre-combustion mass fraction of oxygen in the compartment, which is the concentration of oxygen that would exist in the compartment if no chemical transformation of the vaporised fuel had taken place. This is calculated by adding the mass of oxygen already in the compartment to the mass injected over the current time-step and dividing by the total mass of all the gas that will have contacted the compartment during the current time-step:

$$Y_{O2i} = \frac{Y^{o}_{O2F}\rho V + 0.23 m_a \Delta t}{\rho V + (m_a + R)\Delta t} \tag{3-14}$$

where *Δt* is the time-step.

The other oxygen concentration calculated by the model includes the oxygen consumption term. This is the true oxygen concentration at the end of each time-step when both fuel vaporisation and chemical reaction have taken place:

$$Y_{O2F} = \frac{Y^{o}_{O2F}\rho V + 0.23(m_a - R\gamma\mu)\Delta t}{\rho V + (m_a + R)\Delta t} \tag{3-15}$$

3.2.6.5.2 Product gas concentration

The mass fraction of product gases in the compartment at the end of each time-step is:

$$Y_{PRO} = \frac{Y^{o}_{PRO}\rho V + (1.0 + 0.23\gamma)\mu R \Delta t}{\rho V + (m_a + R)\Delta t} \tag{3-16}$$

where Y^{o}_{PRO} represents the product gas mass fraction from the previous time-step. The denominator represents the mass of gas that will have contacted the compartment. The second term in the numerator represents the combination of oxygen and fuel to form product gases.

3.2.6.5.3 Fuel vapour concentration

Vaporized fuel that fails to burn, either because of insufficient oxygen or imperfect fuel/air mixing, accumulates in the compartment with the other gaseous species. The average concentration of fuel vapour in the compartment for a given time-step is calculated similarly to that of oxygen by including a production and depletion term:

$$Y_{VAP} = \frac{Y_{VAP}^{o}\rho V + R(1-\mu)\Delta t}{\rho V + (m_a + R)\Delta t} \tag{3-17}$$

where Y_{VAP}^{o} represents the product vapour mass fraction from the previous time-step and (1 - μ) represents the unburned fraction of vapour produced over the time period Δt.

3.2.6.5.4 Product gas composition

The calculation is carried out separately for flaming fires and smouldering fires.

3.2.6.5.4.1 Flaming fires

For flaming combustion of foams, plastics and other synthetic substances, the simplified product gas is assumed to consist of water vapour, CO and CO_2. The relative proportion of these is determined by the reaction stoichiometry for the fuel. This takes the general form:

$$\text{Fuel} + O_2 \rightarrow a\ CO + b\ CO_2 + X\ H_2O \tag{3-18}$$

where a is the molar stoichiometric coefficient for CO, b is the molar stoichiometric coefficient for CO_2 and X is the number of moles of water produced in the normalised stoichiometry. The equation above is normalised so that:

$$a + b = 1.0 \text{ mole} \tag{3-19}$$

The mass ratio of CO and CO_2 is assumed to be linear for the purposes of this analysis. The relationship that is assumed for this model is based on the fact that CO_2 production increases with the amount of oxygen available in the ambient air and relates the molar stoichiometric coefficients a and b by

$$44\ b = KY_{O2F}\,(28\ a) \tag{3-20}$$

where K is a tunable constant that depends on the fuel type (260 for polyurethane). Equation (3-20) is only a first order approximation and is deficient when oxygen concentration approaches zero. A value of 1 is inconsistent with experimental observations as this will lead to a zero value of predicted CO_2 concentration. To avoid this situation, a maximum value of 0.5 was set in the computation algorithm, which is consistent with experimental observations. This value will lead to a maximum CO to CO_2 mass ratio of 0.64.

The CO and CO_2 fractions are obtained through the following relationships:

$$Y_{CO} = Y_{PRO} \frac{28a}{28a + 44b + 18X} \tag{3-21}$$

$$Y_{CO2} = Y_{PRO} \frac{44b}{28a + 44b + 18X} \tag{3-22}$$

The coefficients *a* and *b* are obtained by simultaneously solving $\begin{cases} a + b = 1mol \\ 44b = KY_{O2F} 28a \end{cases}$.

3.2.6.5.4.2 Smouldering fires

For smouldering fires:

$$Y_{CO} = 0.05 Y_{PRO} \tag{3-23}$$

$$Y_{CO2} = 0.56 Y_{PRO} . \tag{3-24}$$

Note that the unaccounted fraction of Y_{PRO} consists of a wide variety of gases including H_2O and other compounds.

3.2.6.6 Compartment temperature

The compartment containing the fire is treated as well-mixed combustor in which a single transient energy balance can be written to obtain the room temperature *T*. The temperature in the compartment is based on the heat of combustion for the fuel involved instead of being linked to the production of individual species because of the limited knowledge of chemical kinetics for large-scale fires. The ventilation and fuel mass loss rates also affect the compartment temperature, as these quantities dictate the net energy flux across the compartment boundaries. Therefore, the energy balance for the compartment is given by:

$$c_p \rho V \frac{\Delta T}{\Delta t} = Q_c - Q_W - Q_v - Q_o - Q_r \tag{3-25}$$

where Q_c is the rate of heat release to the room by combustion, Q_W is the rate of heat loss through the compartment walls, Q_v is the rate of heat loss through the compartment opening by convection, Q_o is the rate of heat loss rate through the compartment opening by radiation and Q_r is the rate at which heat is transferred from the room to the fuel for vaporisation and heating, c_p is the gas specific heat, V is the compartment volume, ρ is the gas density.

The rate of heat loss through the compartment walls is:

$$Q_w = A_W[\varepsilon\sigma(T^4 - T_{WI}^4) + h(T - T_{WI})] \qquad (3\text{-}26)$$

where A_W is the total wall and ceiling surface area in the compartment, ε is the gas emissivity, σ is the Stefan Boltzmann constant (W/m^2 K^4), T_{WI} is the inner wall temperature and h is the convective transfer coefficient for the wall.

The convective heat-loss through the compartment opening is:

$$Q_v = c_p m_a(T - T_o) + Rc_p(T - T_S) \qquad (3\text{-}27)$$

where m_a is the compartment ventilation rate, R is the mass lost rate, c_p is the specific heat of wall. This energy term represents the heat loss from the room due to the convection of gases in and out of the room and heat lost to the vaporised fuel in heating it from T_s to T.

The radiative heat loss through the compartment opening is:

$$Q_o = A_o[\varepsilon\sigma T^4 + (1-\varepsilon)\sigma T_{WI}^4 - \sigma T_o^4] \qquad (3\text{-}28)$$

where T_{WI} is the inner wall temperature, T_o is the temperature outside the compartment, σ is the Stefan Boltzmann constant (W/m^{2}K^4) and A_o is the area of the compartment opening.

The heat requirement for solid fuel heating and vaporization is expressed as:

$$Q_r = \sigma\varepsilon(T^4 - T_S^4)A_v + R\Delta H_v \qquad (3\text{-}29)$$

where ΔH_v is the heat required to create one unit mass of vapour, A_v is the ignited area. The first term represents the energy conducted into the fuel in order to heat it, the second term represents the energy required to vaporise the fuel at T_s , T_s is the fuel surface temperature.

3.2.6.7 Wall temperature

The wall temperature T_W within each wall varies with distance from the heated surface and is calculated through the one dimensional heat conduction:

$$\rho_W c_W \frac{\partial T_W}{\partial t} = k\frac{\partial^2 T_W}{\partial x^2} \qquad (3\text{-}30)$$

where x is the wall thickness coordinate, ρ_W is the material density, c_W is the specific heat and k is the thermal conductivity.

The above equation (3-30) requires the definition of two boundary conditions, one for the inside surface of the wall and one for the outside wall surface (see 3-31).

$$
\begin{aligned}
-k\frac{\partial T_W}{\partial x} &= \delta\varepsilon(T^4 - T_{WI}^4) + h(T - T_{WI}) \\
-k\frac{\partial T_W}{\partial x} &= \sigma(T_{WO}^4 - T_O^4) + h(T_{WO} - T_O)
\end{aligned} \qquad (3\text{-}31)
$$

where *k* is the thermal conductivity of the wall material, δ is the wall thickness and *h* is the convective heat transfer coefficient. These boundary conditions represent the radiative and convective exchanges between the inner and outer wall surface and the surrounding gas. For the purpose of simplicity, the emissivities of the compartment walls and surfaces outside the compartment are assumed to be 1.0 (see [18]).

3.2.6.8 Fuel surface temperature

The temperature of the fuel surface is assumed to be controlled by the same mechanisms that control the temperature of surfaces in the lower portion of the compartment. This assumption is necessary in order to give a more conservative estimate of the heat flux to the fuel as mentioned previously. The fuel and lower compartment surfaces are thus modelled as a conductive body between radiative heat flux boundaries:

$$
\rho_f c_f \frac{\partial T_f}{\partial t} = k_f \frac{\partial^2 T_f}{\partial x^2} \qquad (3\text{-}32)
$$

Where k_f is the fuel conductivity, c_f is the fuel specific heat and ρ_f is the fuel density.

The fuel is assumed to radiate to a temperature equal to the ambient temperature T_o from its base and to a temperature equal to the compartment gas temperature *T* from its surface. The radiation boundaries of the fuel are thus similar to those for the walls, but without the convective heat transfer term:

$$
\begin{aligned}
-k_f\frac{\partial T_f}{\partial x}\bigg|_{x=0} &= \sigma\varepsilon(T^4 - T_{si}^4) \\
-k_f\frac{\partial T_f}{\partial x}\bigg|_{x=l} &= \sigma\varepsilon(T^4 - T_{so}^4)
\end{aligned} \qquad (3\text{-}33)
$$

where *l* is the fuel thickness, T_{si} and T_{so} are the temperatures of the fuel surfaces facing the ceiling and floor of the compartment, respectively. The above equations

(3.33) contain the assumption that the floor under the fuel remains close to the ambient temperature.

3.3 Scenarios of the CESARE-RISK Model

In the CESARE-RISK model, four scenarios are considered: Door open, Window open; Door open, Window closed; Door closed, Window open; Door closed, Window closed. They will be represented symbolically by DOWO; DOWC; DCWO; DCWC. The events are also shown in Figure 3-1.

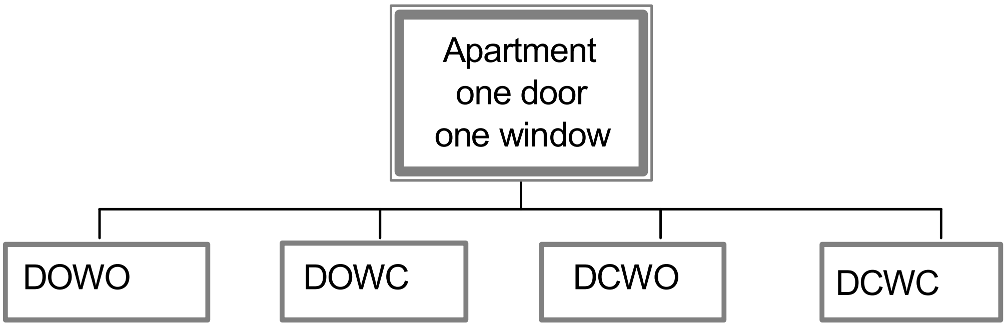


Figure 3-1: Scenarios of apartment fire

3.4 Input variables of the CESARE-RISK Model

The stochastic nature of the input for the CESARE-RISK model is described in the following Table 3-1.

Variables	Name of variables	Symbol	Unit	Distribution	Interval
x_1	Length of room	L	cm	Uniform	(300, 1000)
x_2	Width of room	W_r	cm	Uniform	(300, 1000)
x_3	Height of room	H_r	cm	Uniform	(240, 300)
x_4	Window width factor	f_W		Uniform	(0.5, 1.0)
x_5	Window height factor	f_H		Uniform	(0.4, 1.0)
x_6	Fuel density	ρ_f	kg/m^2	Uniform	(20, 60)
x_7	Fuel Area factor	f_A		Uniform	(0.3, 0.9)
x_8	Flame Spread Rate	R_f	m/sec	Uniform	(0.1, 2.0)

Table 3-1: Stochastic input variables

In the above Table 3-1:

$f_W = W_w / W_r$, where W_w is the window width in cm,

$f_H = H_w / H_r$, where H_w is the window height in cm,

$\rho_f = m_f \times 10^4 / W_r L$, where m_f is the fuel mass in kg,

$f_A = \pi r_f^2 / W_r L$, where r_f^2 is the fuel radius in cm.

The available data have been obtained by sampling the values of the eight input parameters given in the table 1 independently from the specified probability distributions. There are 10,000 simulation data sets available, 2,500 for each of the four scenarios.

3.5 Output variables of the CESARE-RISK Model

The output variables from running the model include: time to light smoke, time to medium smoke, time to heavy smoke, time to flare over, time to untenable conditions, maximum temperature reached and active time.

In this research, we shall concentrate on the analysis of just two output variables: the maximum temperature reached and the time to untenable conditions, for the four scenarios.

CHAPTER 4
MODERN REGRESSION METHODOLOGY

4.1 Introduction

Nonlinear transformation of variables is a commonly used practice in regression problems. Two common goals are stabilization of error variance and symmetrization /normalization of error distribution. A more comprehensive goal is to find those transformations that produce the best-fitting additive model. Knowledge of such transformations aids in the interpretation and understanding of the relationship between the response and predictors.

There are several modern regression methods, such as Alternating Conditional Expectations (ACE), Additive and Variance Stabilizing Transformation (AVAS), Fit Linear Regression (lm), Least Trimmed Squares Regression (ltsreg), Projection Pursuit Regression (ppreg) that could be used to do regression on the data. ACE and AVAS are suited for the data in this research.

4.2 The Alternating Conditional Expectations (ACE) methodology

ACE (Alternating Conditional Expectations) is an intuitively appealing technique introduced by Breiman and Friedman in 1985 [34]. The idea is to find nonlinear transformations $\theta(y)$, $\phi_1(x_1)$, $\phi_2(x_2)$, ..., $\phi_p(x_p)$ of the response y and carriers (or "independent variables") $x_1,x_2,...,x_p$, respectively, such that the additive model

$$\theta(y) = \phi_1(x_1) + \phi_2(x_2) + \ldots + \phi_p(x_p) + \varepsilon \tag{4-1}$$

is a good approximation for the data $y_i,x_{i1},x_{i2},...,x_{ip}$, $i = 1, \ldots, n$. Let y, x_1, x_2, …, x_p be random variables with joint distribution F, and let expectations be taken with respect to F. Consider the goodness-of-fit measure

$$e^2 = e^2(\theta,\phi_1,\ldots,\phi_p) = \frac{E\{[\theta(y) - \sum_{k=1}^{k=p}\phi_k(x_k)]^2\}}{E[\theta^2(y)]}. \tag{4-2}$$

The measure e^2 is the fraction of variance not explained by regressing $\theta(y)$, on $\phi_1(x_1)$, $\phi_2(x_2)$, ..., $\phi_p(x_p)$. The data-based version of e^2 is

$$\hat{e}^2 = \frac{\sum_{i=1}^{n}[\hat{\theta}(y_i) - \sum_{k=1}^{p}\hat{\phi}_k(x_{ik})]^2}{\sum_{i=1}^{n}\hat{\theta}^2(y_i)} \tag{4-3}$$

where $\hat{\theta}$ and the $\hat{\phi}_k$, estimates of θ and ϕ_k, are standardized so that $\hat{\theta}(y_i)$ and the $\hat{\phi}_k(x_{ik})$ have mean zero:

$$\sum_{i=1}^{n}\hat{\theta}(y_i) = 0 \text{ and } \sum_{i=1}^{n}\hat{\phi}_k(x_{ik}) = 0,\ k = 1, \ldots, p.$$

For the usual linear regression case, where $\hat{\theta}(y_i) = y_i - \bar{y}$ and $\hat{\phi}_1(x_{i1} - \bar{x}_1) = (x_{i1} - \bar{x}_1)\hat{\beta}_1, \ldots, \hat{\phi}_p(x_{ip} - \bar{x}_p) = (x_{ip} - \bar{x}_p)\hat{\beta}_p$ with $\hat{\beta}_1, \ldots, \hat{\beta}_p$ the least squares regression coefficients, we have

$$\hat{e}_{LS}^2 = \frac{RSS}{SSY} \equiv \frac{\sum_{i=1}^{n}[(y_i - \bar{y}) - \sum_{k=1}^{p}(x_{ik} - \bar{x}_k)\hat{\beta}_k]^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

and the squared multiple correlation coefficient is given by $R^2 = 1 - e_{LS}^2$.

4.2.1 Further Details of ACE

The ace transformations θ^* and $\phi_1^*, \phi_2^*, \ldots, \phi_p^*$ are the result of minimizing $e^2(\theta, \phi_1, \ldots, \phi_p) = e^2(\theta, \phi)$. Since we wish to minimize $e^2(\theta, \phi)$, the reason for dividing by $E\theta^2(y)$ becomes clear: we want to avoid obtaining the trivial and useless solution

$$\theta(y) = \phi_1(x_1) = \cdots = \phi_p(x_p) \equiv 0.$$

To see why the term "alternating conditional expectations" came into being, consider the case $p = 1$, where

$$e^2(\theta, \phi) = \frac{E[\theta(y) - \phi(x)]^2}{E\theta^2(y)}.$$

It may be shown that the solutions θ^*, ϕ^* must satisfy the simultaneous equations

$$\begin{aligned} \phi^*(x) &= E[\theta^*(y)|x] \\ \theta^*(y) &= E[\phi^*(x)|y] \end{aligned} \qquad (4\text{-}4)$$

where for arbitrary random variables z and w, $E[z|w]$ denotes the conditional expectation of z given w. Under reasonable assumptions this theoretical solution can be shown to be the limit of an iterated sequence of alternating conditional expectations:

$$\begin{aligned}
\phi^{(1)}(x) &= E[y|x] \\
\theta^{(1)}(y) &= E[\phi^{(1)}(x)|y] \\
\phi^{(2)}(x) &= E[\theta^{(1)}(y)|x] \\
&\vdots \\
\phi^{(j)}(x) &= E[\theta^{(j-1)}(y)|x] \\
\theta^{(j)}(y) &= E[\phi^{(j)}(x)|y] \\
&\vdots
\end{aligned} \tag{4-5}$$

For the data-based ACE algorithm one must estimate the above conditional expectations, using good estimates $\hat{\phi}^{(j)}(x_i)$ and $\hat{\theta}^{(j)}(y_i)$ at each iteration *j*.

At iteration *j*, one has available as bivariate data the values $(x_i, \hat{\theta}^{(j-1)}(y_i)) = (x_i, \tilde{y}_i)$, i = 1, …, n which may be used to estimate $\hat{\phi}^{(j)}(x\ )$ in the first half of the iteration step. One then has bivariate data $(y_i, \hat{\phi}^{(j)}(x_i)) = (y_i, \tilde{x}_i)$, i = 1, …, n, with which to estimate $\hat{\theta}^{(j)}(y\ )$ in the second half of the iteration step, thereby completing the *j*th iteration. In each of these half steps, the estimates $\hat{\phi}^{(j)}(x_i)$ and $\hat{\theta}^{(j)}(y_i)$ of the conditional expectations $\phi^{(j)}(x_i) = E[\theta^{(j-1)}(y_i)|x_i]$ and $\theta^{(j)}(y_i) = E[\phi^{(j)}(x_i)|y_i]$ are obtained using a sophisticated scatter plot smoother called *supersmoother* (see [56] for detailed information about *supersmoother*).

In order to deal with the case of $p \geq 2$, ACE uses an iterative technique called *backfitting* [56]. In the first half-step of the *j*th iteration, the backfitting procedure computes estimates $\hat{\phi}^{(j)}(x_i)$ for each carrier x_{i}, one at a time, treating

$\hat{\theta}^{(j-1)}(y) - \sum_{\substack{k=1 \\ k \neq i}}^{p} \hat{\phi}_k^{(j)}(x_k)$ as the response, and cycling through the x_{i}'s until convergence is achieved. Then the $\hat{\theta}^{(j)}(y\ )$ is computed as an estimate of $E[\sum_{k=1}^{p} \hat{\phi}_k^{(j)}(x_k)|y]$ to complete the second half-step of the *j*th iteration.

4.2.2 Key property of ACE

The key property of the ACE is the following: Suppose the true additive model is

$$\theta^0(y) = \sum_{l=1}^{p} \phi_l^0(x_l) + \varepsilon \tag{4-6}$$

where $\phi_1^0(x_1),\ldots,\phi_p^0(x_p)$ has a multivariate normal distribution, ε has a normal distribution with mean zero and ε is independent of x_1, x_2, …, x_p. Then the ACE iteration sequence $\theta^{(j)},\phi_1^{(j)},\ldots,\phi_p^{(j)}$, which is the generalization of equation (4-5) to multiple carriers by using backfitting, converges to $\theta^0,\phi_1^0,\ldots,\phi_p^0$ respectively. The corresponding data-based iteration sequence of estimates $\theta^{(j)},\phi_1^{(j)},\ldots,\phi_p^{(j)}$ will, at convergence, provide estimates $\hat{\theta},\hat{\phi}_1,\ldots,\hat{\phi}_p$ of the true model transformations $\theta^0,\phi_1^0,\ldots,\phi_p^0$. See Breiman and Friedman [34] for more detailed comments on ACE.

4.3 The Additive and Variance Stabilizing Transformation (AVAS) Regression Methodology

Like ACE, the AVAS regression methodology tries to find transformations $\theta(y)$, $\phi_1(x_1)$, $\phi_2(x_2)$, ..., $\phi_p(x_p)$ such that

$$\theta(y) = \phi_1(x_1)+ \phi_2(x_2)+ \ldots + \phi_p(x_p) + \varepsilon \tag{4-7}$$

provides a good additive model approximation for the data $y_i, x_{i1}, x_{i2}, \ldots, x_{ip}$, i= 1, 2, ..., n. However, AVAS differs from ACE in that it chooses $\theta(y)$ to achieve a special variance stabilizing feature. In particular the goal of AVAS is to estimate transformations $\theta,\phi_1,\phi_2,\ldots,\phi_p$, which have the properties

$$E[\theta(y)|x_1,\ldots,x_p] = \sum_{i=1}^{p}\phi_i(x_i) \tag{4-8}$$

and

$$\mathrm{var}[\theta(y)|\textstyle\sum_{i=1}^{p}\phi_i(x_i)] = \text{constant}. \tag{4-9}$$

Here $E[z|w]$ is the conditional expectation of z given w. The additivity structure (4-8) is the same as for ACE, and correspondingly the ϕ_1's are calculated by the *backfitting* algorithm

$$\phi_k(x_k) = E[\theta(y) - \sum_{i\neq k}\phi_i(x_i)|x_k] \tag{4-10}$$

cycling through k = 1, 2,…,p until convergence. The variance stabilizing aspect comes from (4-9). The conditional variance in equation (4-9) is estimated by a different scatter plot smoothing technique (for details see [56]). The equality (4-9) is approximately achieved by estimating the classic variance stabilizing transformation (See Section 4.3.2).

4.3.1 Key properties of AVAS

(a) Suppose that the true additive model is

$$\theta^0(y) = \sum_{i=1}^{p} \phi_i^0(x_i) + \varepsilon \tag{4-11}$$

with ε independent of $x_1, x_2, ..., x_p$, and var(ε) = constant. Then the iterative AVAS algorithm for (4-8) - (4-10), described below for the data versions of (4-8) to (4-10), yields a sequence of transformations $\theta^{(j)}, \phi_1^{(j)}, ..., \phi_p^{(j)}$ which converge to the true transformation $\theta^0, \phi_1^0, ..., \phi_p^0$, as the number of iterations j tends to infinity. Correspondingly, the data-based version of this iteration yields a sequence of transformations $\hat{\theta}^{(j)}, \hat{\phi}_1^{(j)}, ..., \hat{\phi}_p^{(j)}$ which, at convergence, provide estimates $\hat{\theta}, \hat{\phi}_1, ..., \hat{\phi}_p$ of the true model transformations $\theta^0, \phi_1^0, ..., \phi_p^0$.

(b) AVAS appears not to suffer from some of the anomalies of ACE, that is, not finding good estimates of a true additive model (equation 4-11) when normality of ε and joint normality of $\phi_1(x_1), \phi_2(x_2), ..., \phi_p(x_p)$ fail to hold.

(c) AVAS is a generalization of the Box-Cox (1964) [57] maximum-likelihood procedure for choosing a power transformation y^λ of the response. AVAS also generalizes the Box-Tidwell [58] procedure for choosing transformations of the carriers $x_1, x_2, ..., x_p$, and is much more convenient than the Box-Tidwell procedure (see also Weisberg [59]).

(d) $\hat{\theta}(y)$ is a monotone transformation, since it is the integral of a nonnegative function. This is important if one wants to predict y by inverting $\hat{\theta}$: monotone transformations are invertible, and hence we can predict y with $\hat{y} = \hat{\theta}^{-1}[\sum_{i=1}^{p} \hat{\phi}_i(x_i)]$. This predictor has no particular optimality property, but is simply one straightforward way to get a prediction of y once an AVAS model has been fitted.

4.3.2 Further details of AVAS

Let

$$v(u) = \mathrm{var}[\hat{\theta}(y) | \sum_{i=1}^{p} \phi_i(x_i) = u] \tag{4-12}$$

where $\hat{\theta}(y)$ is an arbitrary transformation of y. $\hat{\theta}(y)$ will be the "previous" estimate of θ(y) in the overall iterative procedure described below. Given the variance function $v(u)$, it is known that $\mathrm{var}[g(\hat{\theta}(y)) \mid \sum_{i=1}^{p} \phi_i(x_i) = u]$ will be constant if g is computed according to the rule

$$g(t) = \int_c^t \frac{du}{v^{1/2}(u)} \tag{4-13}$$

for an appropriate constant c. See Box and Cox [57].

The detailed steps in the population version of the AVAS algorithm are as follows:

1. Initialize:

   Set $\hat{\theta}(y) = (y - Ey)/[\mathrm{var}(y)]^{1/2}$ and backfit on $x_1, x_2, \ldots, x_\mathrm{p}$ to get $\hat{\phi}_1, \ldots, \hat{\phi}_p$, that is $\hat{\phi}(x) \leftarrow E(\theta(y)|x)$.

2. Get the new transformation of y:

   a. Compute the variance-stabilizing transformation

   $v(u) = \mathrm{var}[\hat{\theta}(y) \mid \sum_{i=1}^{p} \hat{\phi}_i(x_i) = u]$.

   b. Compute the variance-stabilizing transformation $g(t) = \int_c^t \frac{du}{v^{1/2}(u)}$ and

   c. Set $\hat{\theta}(y) = g(\hat{\theta}(y))$ and standardize $\hat{\theta}(y) = \dfrac{\hat{\theta}(y) - E\hat{\theta}(y)}{\mathrm{var}^{1/2}\hat{\theta}(y)}$.

3. Get the new $\hat{\phi}_i$'s:

   Backfit $\hat{\theta}(y)$ on $x_1, x_2, \ldots, x_p$ to obtain new estimates $\hat{\phi}_1, \ldots, \hat{\phi}_p$.

4. Iterate steps 2 and 3 until

   $$R^2 = 1 - \hat{e}^2 = 1 - E[\hat{\theta}(y) - \sum_{i=1}^{p} \hat{\phi}_i(x_i)]^2 \tag{4-14}$$

   does not change.

Of course, the above algorithm is actually carried out using the sample of data y_i, x_{i1}, …, x_{ip}, i = 1, …, n, with expectations replaced by sample averages, conditional expectations replaced by scatter plot smoothing techniques and population variances replaced by sample variances. See Tibshirani 1988 [35] for more detailed comments on AVAS.

CHAPTER 5
RESPONSE SURFACE METHODS AND RELIABILITY INDEX ANALYSIS

5.1 Response surface methods

In studying the structure of limit state functions, the following two problems are often encountered by researchers:

1. The functions are not known explicitly;
2. They have a complicated functional form.

Suppose that the state function is given in terms of a response variable Y with

$$Y = g\,(X_1 ,X_2 ,\ldots,X_n\ ) \qquad (5\text{-}1)$$

but the functional form of g is unknown. Here the $X_1\ ,X_2\ ,\ldots,X_n$ are called the independent or regressor variables and Y the dependent or response variable. The usual method of statistical inference to find such a relationship is the *response surface method* (see [40], [43], [44] and [45]). A brief review of response surface methods as they relate to this research will be given in the following sections.

5.1.1 Basic ideas of response surfaces

Suppose that the response variable Y depends on the input variables $X_1, X_2, \ldots\ldots, X_n$. Experiments are conducted with input variables $\boldsymbol{X} = (X_1, X_2, \ldots\ldots, X_n\ )$ a sufficient number of times to define the response surface to the level of accuracy desired. Each experiment can be represented by a point with coordinates $\boldsymbol{X}_j = (\ X_{1j}, X_{2j}, \ldots\ldots,X_{nj}\ )$ in an n-dimensional space. At each point, a value of y_j is observed. Although the actual response Y is a function of the input variables, that is $Y = g(X_1, X_2, \ldots\ldots, X_n)$, this function is generally unavailable in closed form. The classical response surface procedure is to approximate $g(\boldsymbol{X})$ by an nth order polynomial $\widetilde{g}\,(\boldsymbol{X})$ with undetermined coefficients. Statistical analysis is performed to determine the unknown coefficients in the polynomial $\widetilde{g}\,(\boldsymbol{X})$ such that the error of approximation is minimum in the region of interest. Normally, a log transformation is not considered, when there is no reason to believe that the deviations from the response surface are significantly non-normal. However, in certain conditions, such as output involving a non-negative response, such as a waiting time, a log transformation will improve the fit (see Chapter 10 in the thesis).

More generally, the response surface method consists of the following steps:

1. Choice of one of several families of functions, which appear to be suitable to approximate the unknown function $g(x_1, x_2, \ldots\ldots, x_n)$,
2. If possible, design of experiments, which will give optimal estimators of the parameters of the functions chosen in the last step,
3. Validation of the derived approximation model by statistical tests or other methods.

5.1.2 Selection of the order of the polynomial

The selection of the order of the approximating polynomial and points x_i for experimentation require careful consideration. Up to a certain degree, a higher order polynomial improves the accuracy of the approximation at the expense of additional computation. The rate of increase in accuracy reduces with increasing the degree of the polynomial but the computational costs increase exponentially. Moreover, higher order polynomials can exhibit erratic behaviour in the sub-domains not covered by the experiments [47].

For reliable estimates, one needs to have a good approximation to $g(\boldsymbol{X})$ around the design (or minimum norm) point, that is the region of the failure domain D_F that contributes most to the overall failure probability. Since we neither know the actual limit state function nor the actual design point, the accuracy of the estimate depends on the accuracy of the polynomial approximation in the region of the design point.

5.1.2.1 Quadratic approximation

A second order response surface for n input variables is described by a quadratic model,

$$\tilde{g}(\boldsymbol{X}) = A + \boldsymbol{X}^{\mathrm{T}}\boldsymbol{B} + \boldsymbol{X}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{X} \qquad (5\text{-}2)$$

where A, $\boldsymbol{B}^{\mathrm{T}} = [B_1, B_2, \ldots\ldots, B_n]$, and,

$$\boldsymbol{C} = \begin{bmatrix} C_{11} & \cdots & C_{1n} \\ \vdots & \ddots & \vdots \\ & \cdots & C_{nn} \end{bmatrix}$$

are the undetermined coefficients. Experiments are conducted as per the adopted design and the resulting system of equations may be put in the form.

$$\boldsymbol{G} = \boldsymbol{Dd} + \boldsymbol{e} \tag{5-3}$$

where d is a vector of constants A, B_i, C_{ij}, the matrix $\boldsymbol{D}$ contains constant, linear, quadratic and cross-combination functions of the X_j and $\boldsymbol{e}$ is the error vector, the components e_i of which consist of a lack of fit error resulting from approximating g by $\tilde{g}$, and a pure experimental error, assumed to be a zero mean random vector. The solution,

$$E(\boldsymbol{d}) = (\boldsymbol{D}^{\mathrm{T}}\boldsymbol{D})^{-1}\boldsymbol{D}^{\mathrm{T}}\boldsymbol{G} \tag{5-4}$$

provides the expected values of the unknown coefficients.

Other polynomial interpolation schemes using Lagrangian and Hermite polynomials are possible, although no specific examples of their application for reliability analysis could be located. At a higher level of sophistication, Ditlevsen and Madsen [39] have presented a random field model for stochastic interpolation between point by point measured values of a spatially distributed material property.

A possible further step in response surface methodology is to test the significance of the contribution of terms in the derived functional form, for example, in a quadratic polynomial, the significance of the square terms. If they are not significant, a simpler model without these terms might be used instead of the whole polynomial expression. If experiments are made, we have in general some random variability. If the random experiment is run m times for the same values of $x_1, x_2, \ldots\ldots, x_n$, the resulting values $y_1, y_2, \ldots\ldots, y_m$ of the response variable will be different. This means that there is inherently a pure random error. Therefore the model should be put in the form

$$y = g(x_1, x_2, \ldots\ldots, x_n) + \varepsilon(x_1, x_2, \ldots\ldots, x_n) \tag{5-5}$$

with $\varepsilon(x_1, x_2, \ldots\ldots, x_n)$ representing the error term. To judge the quality of a model, estimates of the magnitude of this error term are important. Since in such an

experimental design we have different responses for the identical set of regressor variables, it will not be possible in general to fit a model without any error term.
If the experiments are of a numerical nature, contribution to this pure error arises from the impossibility of working in the space of all the influencing variables. Usually, a projection in a space of reduced dimension is introduced, where then the influence of the neglected variables results in a random effect.

The commonly used orthogonal experimental designs are 2^n and 3^n factorial designs [50, 51, 52, 53]

These factorial designs, though efficient, lead to unacceptably high computational efforts with the increase in number of variables for complex systems and may become more time consuming than simulation.

An iterative response surface approach for reliability analysis was presented by Bucher and Bourgund [54]. The experimental design in each iteration consists of as many locations as the total number of undetermined coefficients in the polynomial

$$\tilde{g}(x) = a + \sum_{i=1}^{n} b_i x_i + \sum_{i=1}^{n} c_i x_i^2 \qquad (5\text{-}6)$$

in which x_i, i = 1, 2,…, n are basic variables and the parameters a, b_i, c_i are to be determined.

The constants are to be determined by using (2n+1) values of $g(\boldsymbol{x})$ at the mean values μ_i of the random variables X_i, and at $x_i = \mu_i \pm h_i \sigma_i$, in which h_i is an arbitrary factor and σ_i is the standard deviation of X_i.

The new center point for interpolation is chosen on a straight line from mean vector to the minimum norm point [54]. The total number of evaluations of $g(x)$ is $(4n+1)$.
To improve the accuracy of the response surface, Rajashekhar and Ellingwood [55] added the cross terms in the polynomial developed by Bucher and Bourgund [54].

$$\tilde{g}(x) = a + \sum_{i=1}^{n} b_i x_i + \sum_{i=1}^{n} \sum_{j \geq i=1}^{n} c_{ij} x_i x_j \,. \qquad (5\text{-}7)$$

The total number of experiments to be conducted for each approximation would then increase to $(n+1)(n+2)/2$. This method does not appear to have any particular advantage with regard to computational effort and accuracy as compared to more conventional methods for selecting experimental points.

5.2 The methodology adopted in this thesis

5.2.1 Steps in the procedure

1. For each scenario we apply a modern regression method (AVAS or ACE depending on which appears most successful). Visual assessment of the plots of transformed inputs against the transformed output indicates different modes of fire growth for different ranges of the inputs. The data set is then split into subsets covering the ranges of input identified.
2. For each subset we carry out a polynomial regression of a special type, namely one for which each input appears as a polynomial, but there are no product terms between the inputs because this will considerably simplify the reliability calculation. As will be shown in the following chapters, it is enough to use quadratic polynomials.
3. For each subset we carry out, where appropriate, a cubic transformation of the predicted output so as to further improve the fit of the predicted output values to the observed output values.

Throughout the procedure, we measure the goodness of fit of the regression by evaluating the correlation between the predicted values and the observed values. The square of this correlation measures the amount of variation in the input that is explained by the regression.

The use of correlation here carries no implication that there is a random element in the computer model. That model is in fact purely deterministic. Here the use of correlation is as described by R.A.Fisher (quoted by Searle 1987 p.13 [66]): “a simple method of arranging arithmetical facts so as to isolate and display the essential features of a body of data with the utmost simplicity”. Correlation here simply measures the departure of the regression approximation from the full computer model.

5.2.2 Outliers and their treatment

In the course of applying the procedure outlined in subsection 5.2.1 there were usually some data points that were clearly “outliers”.

An outlier is defined as “an observation that is so different from the bulk of the observations that it stands out”. It is customary, when such outliers are detected, to reject them, as they tend to grossly distort the results of the analysis. ([67] Staudte and Sheather 1990). Outliers in the analysis of the CESARE-RISK computer model arise because certain particular combinations of inputs lead to unstable outputs where small errors in the computations can lead to large variations in the output.

In an engineering context, two conditions are required to make rejection of outliers acceptable:

1. The proportion of outliers in the data set must be small.
2. Omitting them, which is equivalent to replacing the corresponding outputs with their predicted values in the design calculations, should be overwhelmingly conservative, i.e. should make the design safer.

As will be seen at the end of Chapter 6, those two conditions were fulfilled in the study of maximum temperature.

For the study of time to untenable conditions (chapters 8 and 10) no outliers were detected.

5.3 Reliability index and the failure probability in fire engineering

The aim of probabilistic design in fire engineering is to ensure that the probability that the design is safe is greater than 1-p_F, where p_F is an acceptable level of failure probability. Safety is defined as follows: We consider a set of physical variables representing the model: $\boldsymbol{X} = X_1, \ldots, X_n$. In the n-dimensional space of these physical variables we define a limit hypersurface $G(X)$ which divides the space into a safe region and a failure region. The variables of the fire model are stochastic and therefore induce a probability distribution in the physical variable space. The reliability of the design is the probability of the safe region. Calculating the reliability of a design directly from the fire model requires a very large amount of computation. The purpose of the regression analysis described in the preceding section is to make it possible to use a simplified approach which can be used by practising engineers to obtain a reliability index for the considered design.

We propose to apply the well-known First Order Second Moment Method (Hasofer and Lind 1974 [36]) to obtain what is known as the β reliability index. This method has been widely used in structural engineering (Melchers 1987 [37]) and has recently been advocated in fire engineering by Frantzich et al 1997 [38] and Ramachandran 1998 [75].

The procedure is as follows:

1) Carry out a linear transformation on the vector ***X*** of physical variables so as to obtain a vector ***U*** of uncorrelated variables, each having zero mean and unit standard deviation.
2) Find the image $G^*(\boldsymbol{U})$ of the limit state function $G(\boldsymbol{X})$ in the ***U*** space.
3) Find the distance from the origin to the limit state function in the ***U*** space. This is the β reliability index.

This can be illustrated in two dimensions by Figure 5-1. Clearly, if we draw a series of concentric circles around the origin, the β index will simply be the distance of the origin to the point ***D*** at which one circle just touches the limit curve. This point is known as the *design point*.

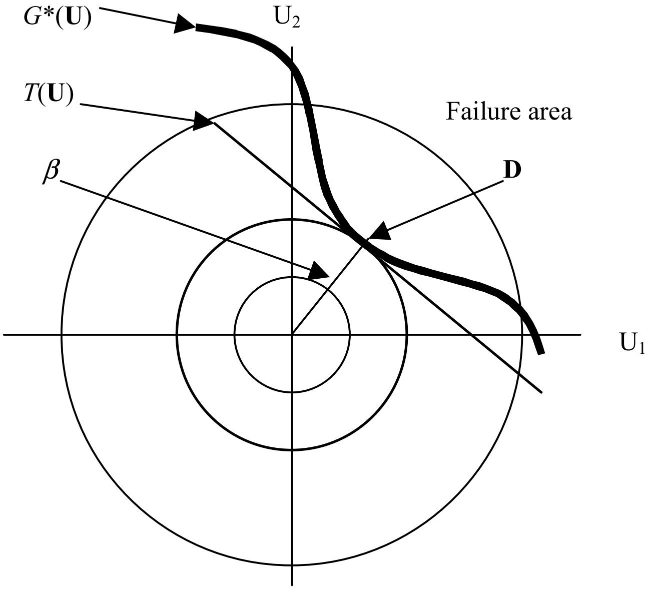


Figure 5-1: Illustration of the β index in two dimensions

It is to be noted that as long as the limit surface is relatively smooth it is well approximated by the tangent hyperplane to the limit state surface $T(\boldsymbol{U})$ at $\boldsymbol{D}$. If, in addition, the new variables $\boldsymbol{U}$ have approximately a multivariate normal distribution, which is often the case, it is shown in the reliability literature that the reliability of the design, i.e. the probability of the safe region, is approximately equal to $\Phi(\beta)$ where Φ is the distribution function of the standard normal distribution. The probability of failure p_F is given by $\Phi(-\beta)$.

To estimate risk in fire, the probability of failure must be found. The final step of the research is the development and application of a model that can be used to estimate the probability of failure in the fire model.

The first order second moment analysis is illustrated in Figure 5-1. There the limit state function $G(\boldsymbol{X})$ consists of two basic random variables, x_1 and x_2 . But the ideas are readily extended to more than two basic random variables.

In Figure 5-1. $T(\boldsymbol{U}) = 0$ is the tangent hyperplane. Therefore, point $\boldsymbol{D}$ is the checking point of failure (also called design point). The safety index β can calculated as follows:

$$\beta = \min(u_1^2 + u_2^2)^{\frac{1}{2}} . \tag{5-8}$$

In an n-dimensional space with a hyperplane limit state function, the shortest distance, the safety index β, is then

$$\beta = \min\left(\sum_i^n u_i^2\right)^{\frac{1}{2}} \tag{5-9}$$

where the u_i represent the coordinates of any point on the limit state surface.

5.4 The Monte Carlo method and applications

5.4.1 The Monte Carlo method

There are many situations in probabilistic risk analysis when there is no analytic algorithm that will evaluate the required probabilities. Alternatively, the available algorithm is extremely complex and can only be carried out at great expense of effort and computer time.

An alternative method is known as Monte Carlo simulation. It depends on the fact that the histogram of a large random sample approximates the probability function of the underlying random variable.

Suppose that the output variable required to carry out the risk analysis, denoted by Y, is given as a function of a vector $\boldsymbol{X}$ of underlying variables: $\boldsymbol{X} = X_1, X_2, ..., X_n$, in the form

$$Y = f(\boldsymbol{X}). \tag{5-10}$$

In the Monte Carlo method, a random sample of size N of the vector of underlying variables, $\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_N$ is generated. Each such vector is called a realization of the vector $\boldsymbol{X}$. To each realization there corresponds a value of the output variable Y. Thus we obtain a sample of size N from the output variable Y. Provided N is chosen appropriately large, the histogram of Y will approximate its distribution as closely as required.

5.4.2 The confidence interval for a Monte Carlo simulation

Suppose that a Monte Carlo simulation of size N is carried out to determine the probability of some subset A of the output space. Suppose that the output of n simulations is in A. The Monte Carlo estimator of p_A, the probability of A, is $\hat{p}_A$ = n/N. Now each realization of the input can be thought of as a Bernoulli trial with probability of success p_A. It is not difficult to calculate a confidence interval for the estimator (see [20], [27]), for example the 95% confidence interval for the estimator $\hat{p}_A$ is

$$\hat{p}_A \mp 1.96\sqrt{\frac{\hat{p}_A(1-\hat{p}_A)}{N}} . \tag{5-11}$$

Clearly, the larger N, the shorter the confidence interval and the greater the precision of the estimator. In fact, the length of the confidence interval varies inversely as the square root of the number of simulations.

5.4.3 Confirmation of reliability by Monte Carlo

Monte Carlo simulation can be used to confirm the probability of failure obtained from the reliability index. The easiest method is to work in the $\boldsymbol{U}$ space. A set of N realizations of the standardized vector $\boldsymbol{U}$ is generated and each $\boldsymbol{U}$ is tested to determine whether it falls in the safe region or failure region. The probability of the failure region is then estimated by n/N, where n is the number of vectors $\boldsymbol{U}$ that fall in the failure region.

The details of the application of reliability index and Monte Carlo simulations for fire engineering in this research will be further discussed in Chapter 6 and Chapter 7.

CHAPTER 6
MODERN REGRESSION ANALYSIS OF MAXIMUM TEMPERATURE

In this chapter, through using the modern regression method, AVAS, a simple response surface will be derived for the maximum temperature reached under different events of the CESARE-Risk Model, for each of a number of subranges of the input parameters.

6.1 Modern regression analysis of DOWO scenario

6.1.1 The stochastic nature of DOWO scenario

6.1.1.1 The stochastic nature of the input for DOWO scenario

The stochastic nature of the input is described in the following Table 6-1-1.

Variables	Name of variables	Symbol	Unit	Distribution	Interval
x_1	Length of Room	L	[cm]	Uniform	(300, 1000)
x_2	Width of Room	W_r	[cm]	Uniform	(300, 1000)
x_3	Height of Room	H_r	[cm]	Uniform	(240, 300)
x_4	Window Width Factor	f_W		Uniform	(0.5, 1.0)
x_5	Window Height Factor	f_H		Uniform	(0.4, 1.0)
x_6	Fuel Density	ρ_f	[kg/ m^2]	Uniform	(20, 60)
x_7	Fuel Area Factor	f_A		Uniform	(0.3, 0.9)
x_8	Flame Spread Rate	R_f	[m/sec]	Uniform	(0.1, 2.0)

Table 6-1-1: the stochastic input parameters of DOWO scenario

In Table 6-1-1: $W_w = f_W \cdot W_r$, where W_w is window width; $H_w = f_H \cdot H_r$, where H_w is window height; $m_f = 10^4 \times W_r \cdot L \cdot \rho_f$, where m_f is fuel mass (kg); $r_f = (W_r \cdot L \cdot f_A / \pi)^{1/2}$,where r_f is fuel radius.

The available data have been obtained by sampling the values of the eight input parameters given in the table independently from the specified probability distributions. There are 2,500 simulation data available for DOWO scenario.

6.1.1.2 The stochastic output for DOWO scenario

In this section, we shall concentrate on the analysis of just one output variable: the maximum temperature reached, denoted by y.

6.1.2 AVAS regression analysis for DOWO scenario

By applying the AVAS regression on the DOWO data (2,500 data sets), we obtained eight transformed inputs and one transformed output. Plots of the transformed data against the original data are shown in Figure 6-1-1.

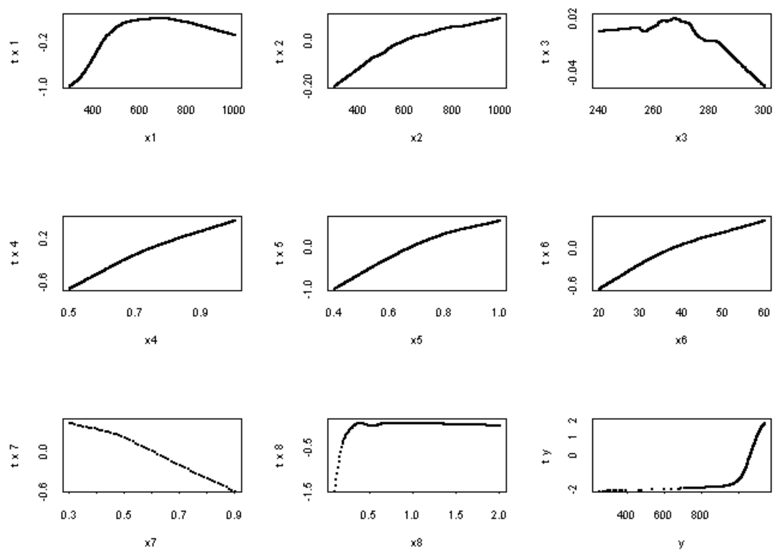


Figure 6-1-1: Plots of the transformed variables against the original data (X2)

When there are two different rates of grow in the transformed data curve, this corresponds to a different mode of fire growth. Analyzing Figure 6-1-1leads to the following conclusions:

1. From the plot of variable x_1, which is the length of the room, it is clear that there were different modes of fire growth for room length *L*, less than 600 cm and room length greater than 600 cm.
2. There were also different fire growth modes of variable x_8 , which is the flame spread rate R_f. It is clear that there is a change of behavior when the flame spread rate is greater than 0.455 m/sec and when the flame spread rate is less than 0.455 m/sec.

and also the transformed value of variable x_3 is comparatively small and thus can be neglected.Therefore, I have created two sub-range data sets for $L > 600$ cm ($L < 600$ cm will be discussed later) as follows:

X233: $L > 600$ cm, $R_f > 0.455$ m/sec; X232: $L > 600$ cm, $R_f < 0.455$ m/sec.

6.1.2.1 AVAS regression analysis for X233

There were 1176 data points satisfying the constraints $L > 600$ cm and $R_f > 0.455$ (i.e.X233). By using AVAS regression analysis on X233 data set, it is clear that seven data points were outliers, as shown in Figure 6-1-2, which is a scatter plot of the transformed y against its predicted value from the regression. Deleting them left us with 1169 data points (X2331). It also turned out that the room height, x_3, and the flame spread rate x_8, can be ignored in the regression calculations under the applied constraints because their effect is very small. So the set of indices used was just i = 1,2, 4,5,6,7.

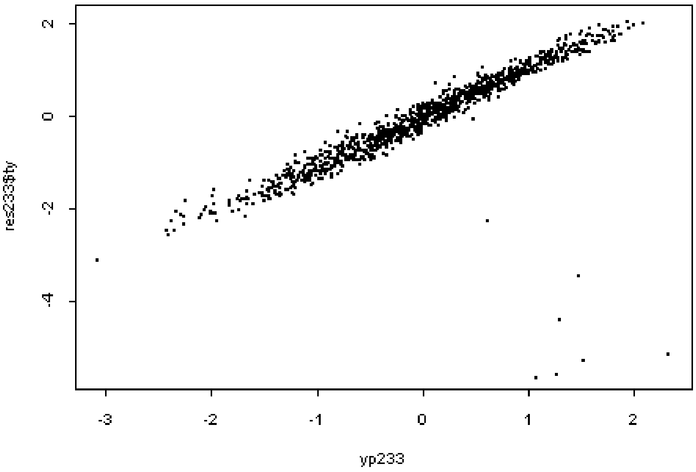


Figure 6-1-2: Scatter plot of transformed y against y_{pred}

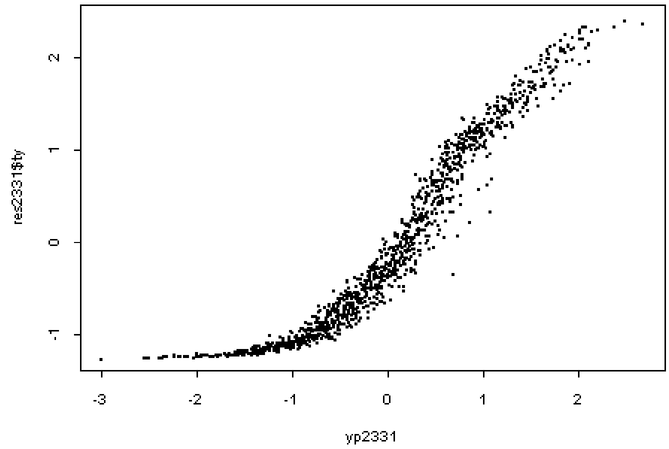


Figure 6-1-3: Scatter plot of transformed y against y_{pred} (outliers removed)

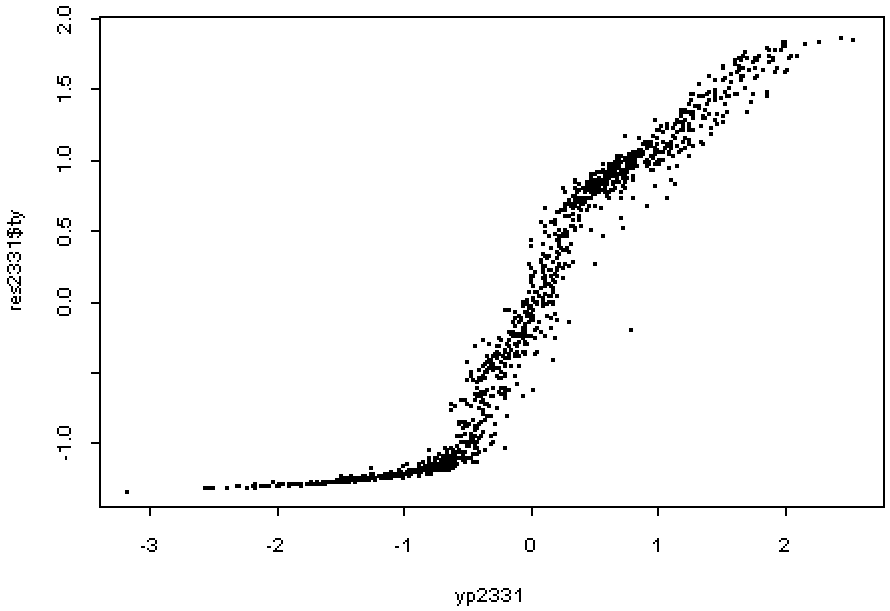


Figure 6-1-4: Scatter plot of transformed *y* against y_{pred} (including x_8)

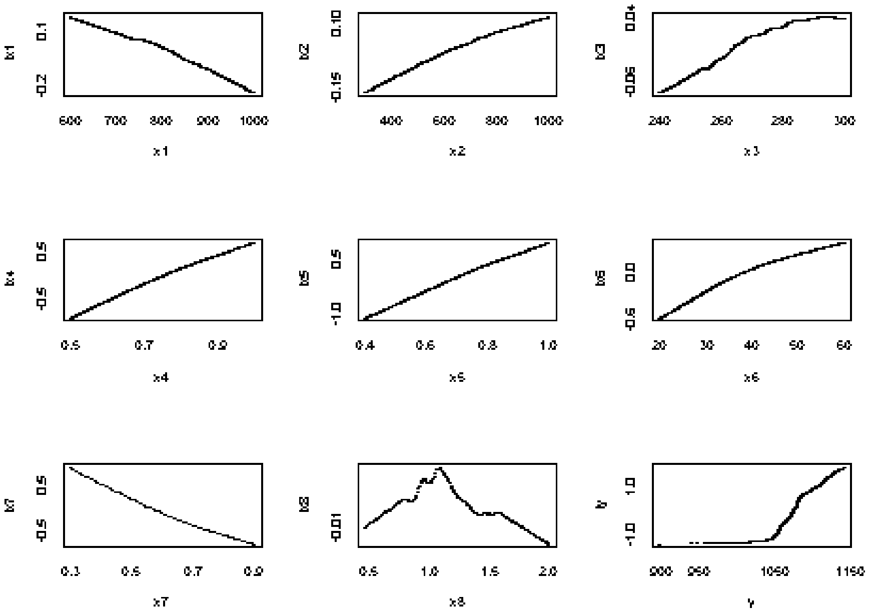


Figure 6-1-5: Plots of the transformed variables against the original data (X2331)

To the 1169 data points we fitted by linear regression a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon . \qquad (6\text{-}1\text{-}1)$$

The coefficient c = 682.4. The a_i and b_i were as in Table 6-1-2.

i	1	2	3	4	5	6	7
b_i	0.0867	0.0511	0.3892	271. 3	309.7	2.4186	-155.3
a_i	-0.0001	-0.0000	-0.0006	-97.27	-123.4	-0.0181	34.65

Table 6-1-2: Values of quadratic regression coefficients for X2331

Letting

$$y_t = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}1\text{-}2)$$

It was found that the correlation between y and y_t was 0.988. The scatter plot of y against y_t is shown in Figure 6-1-6.

The second step in the fitting is to improve the fit of y_t to y by using a cubic regression formula of the form

$$y = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 + \varepsilon \qquad (6\text{-}1\text{-}3)$$

The coefficients turned out to be:

C_0 = 1898.7; C_1 = -6.459; C_2 = 0.008978; C_3 = -3.423e-006.

Setting

$$y_{pred} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 \qquad (6\text{-}1\text{-}4)$$

The correlation achieved between y and y_{pred} is now 0.992. A scatter plot of y against y_{pred} is shown in Figure 6-1-7.

Note: When we ignored x_3 and x_8.

The coefficient c was 743.2. The b_{i} and a_{i} are as in Table 6-1-2a.

i	1	2	4	5	6	7
b_i	0.0910	0.0459	273.6	311.6	2.435	-154.1
a_i	-0.0001	0.000	-98.83	-125.1	-0.0183	33.80

Table 6-1-2a: Values of quadratic regression coefficients (ignored x_3 and x_8)

It was found that the correlation between y and y_t was 0.969. More detailed discussion of this issue were given by Hasofer and Qu [65].

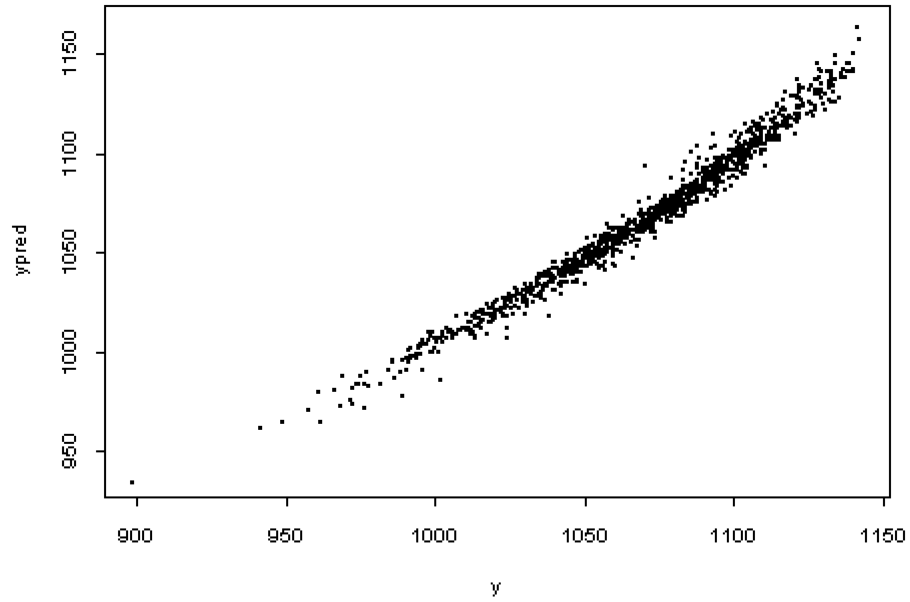


Figure 6-1-6: Scatter plot of y against y_{pred} (quadratic fitted)

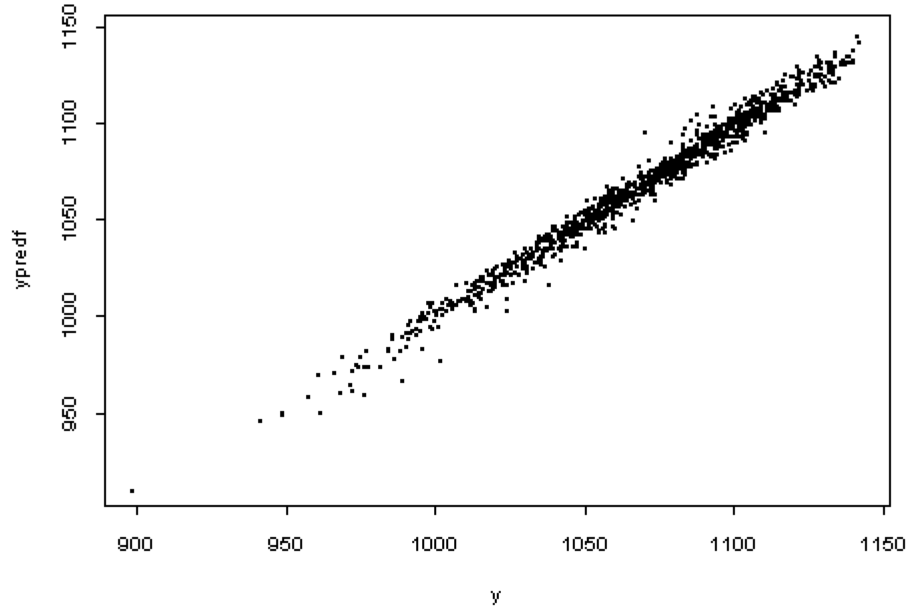


Figure 6-1-7: Scatter plot of y against y_{predf} (cubic fitted to quadratic values)

6.1.2.2 AVAS regression analysis of X232

There were 261 data sets satisfying the constraints L >600 cm and R_f < 0.455 m/sec. By using AVAS regression analysis on X232 data set, (see Figure 6-1-8), it is clear that there were two different modes of fire growth for room width W_r, that is x_2. When W_r < 700 cm, we have a new sub-range data set X2322 (159 observations), and when W_r > 700 cm, data set X2323 (102 observations).

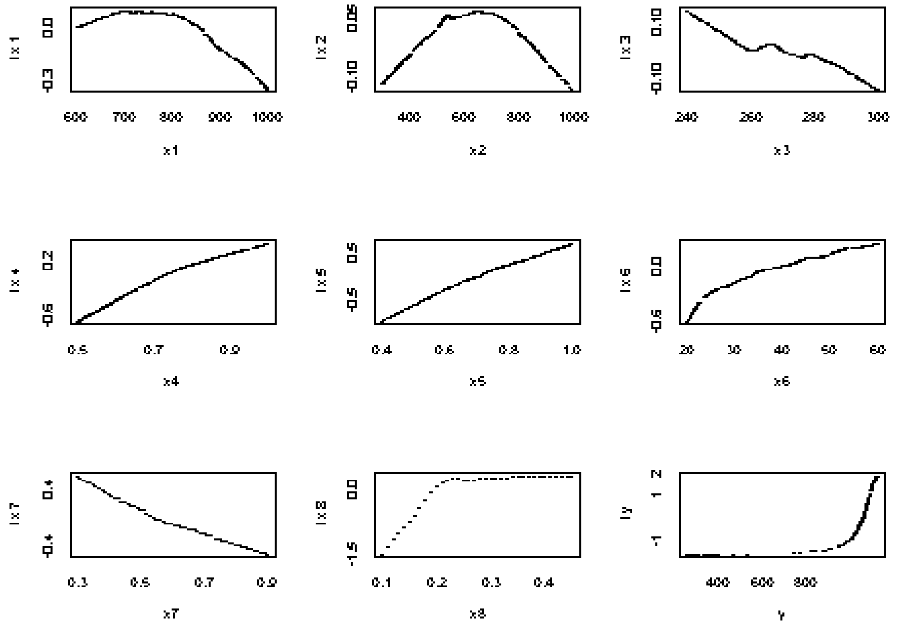


Figure 6-1-8: Plots of transformed variables against original data of X232

6.1.2.2.1 AVAS regression analysis for X2322

Through using AVAS regression analysis on X2322 data set , it is clear that there are several data points that are outliers, as shown in Figure 6-1-9. Deleting them left us with 148 data points, which we call it X2322f. and the AVAS correlation rose from 0.395 (X2322) to 0.954 (X2322f).

We apply again the AVAS algorithm and plot the predicted transformed variables against the original data of X2322f (see Figure 6-1-10).

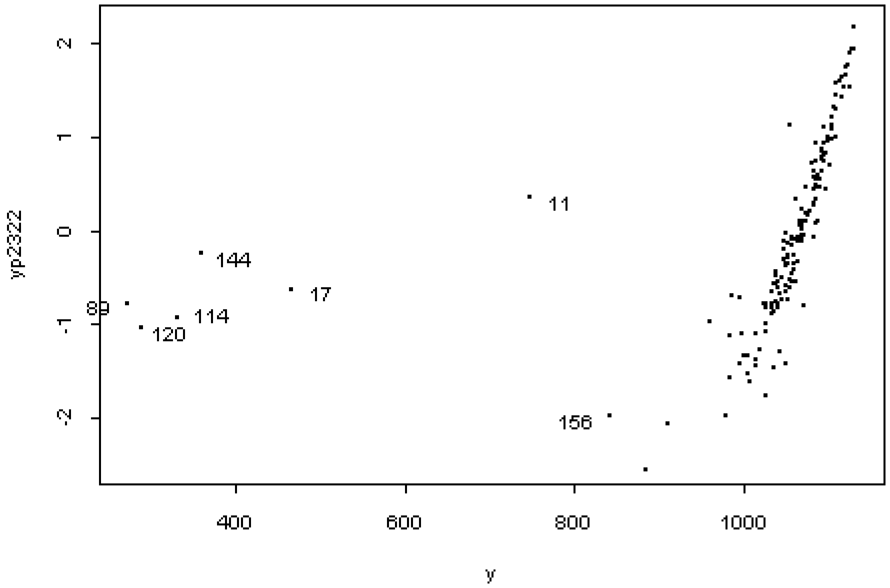


Figure 6-1-9: Scatter plot of y against AVAS y_{pred} of X2322

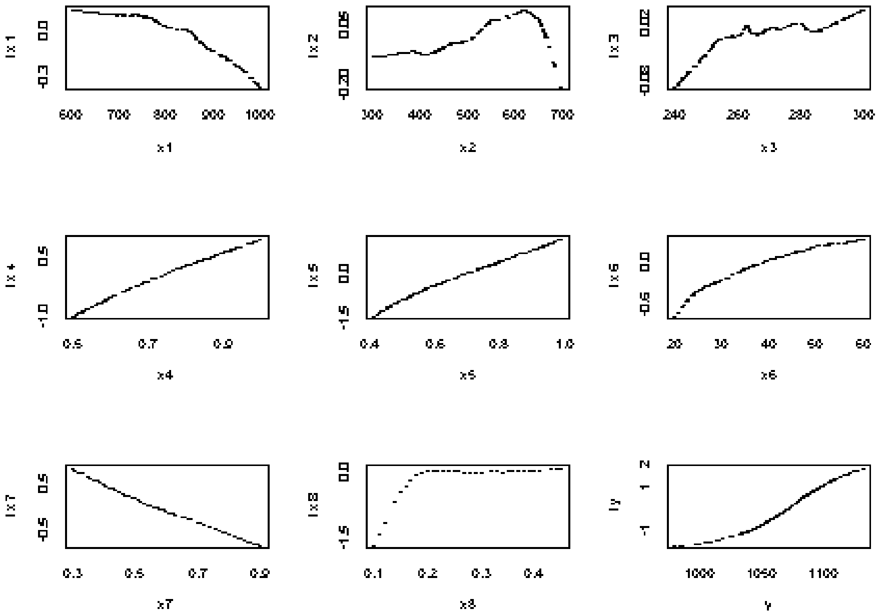


Figure 6-1-10: Plots transformed variables against original data of X2322f

To the 148 data sets we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon \qquad (6\text{-}1\text{-}5)$$

The coefficient c = 570.0. The a_i and b_i were as in Table 6-1-3.

i	1	2	3	4	5	6	7	8
b_i	0.1058	0.1415	0.4145	238.3	257.8	2.799	-59.32	439.5
a_i	-0.00010	-0.00012	-0.00054	-72.18	-88.69	-0.02461	-17.15	-685.9

Table 6-1-3: Values of quadratic regression coefficients for X2322f

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}1\text{-}6)$$

It was found that the correlation between y and y_{pred} (quadratic fitted) is 0.941. (cubic fitted 0.944), and a scatter plot of y against y_{pred} is shown in Figure 6-1-11.

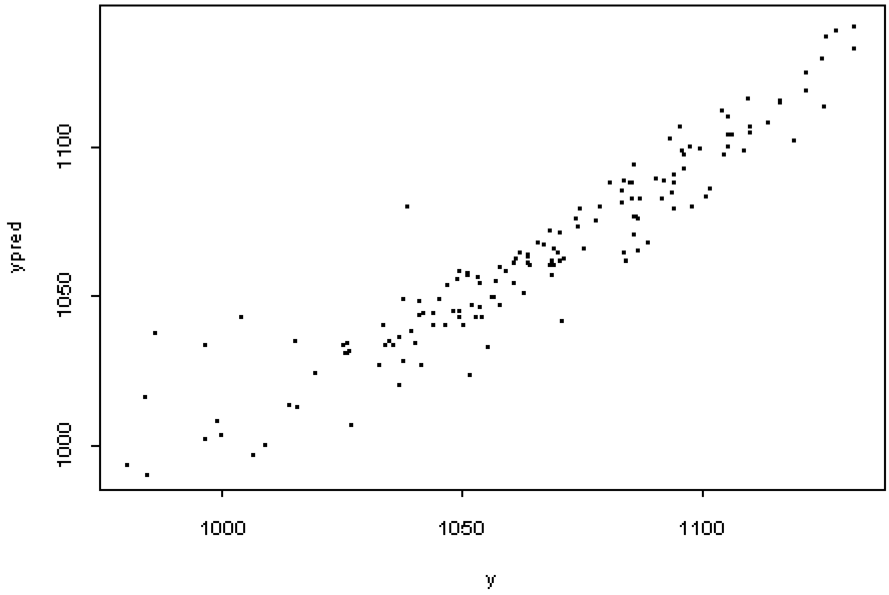


Figure 6-1-11: Scatter plot of y against $\bar{y}_{pred}$ for X2322f

6.1.2.2.2 AVAS regression analysis for X2323 (102 obs)

We apply the AVAS algorithm to the data set X2323 and plot the predicted transformed maximum temperature against maximum temperature as shown in Figure 6-12. Deleting the outliers in Figure 6-1-12 left us with 83 observations (data set X2323f) and the correlation was increased from 0.690 to 0.924. The plots of transformed variables against the original data of X2323f are shown as in Figure 6-1-13.

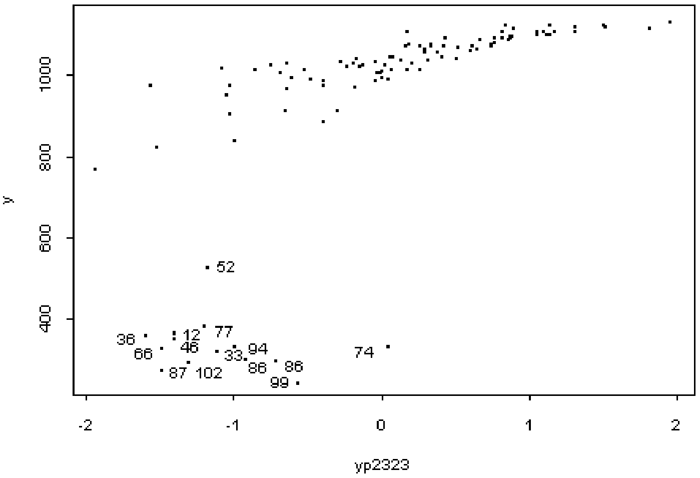


Figure 6-1-12: Scatter plot of y against y_{p2323} of X2323

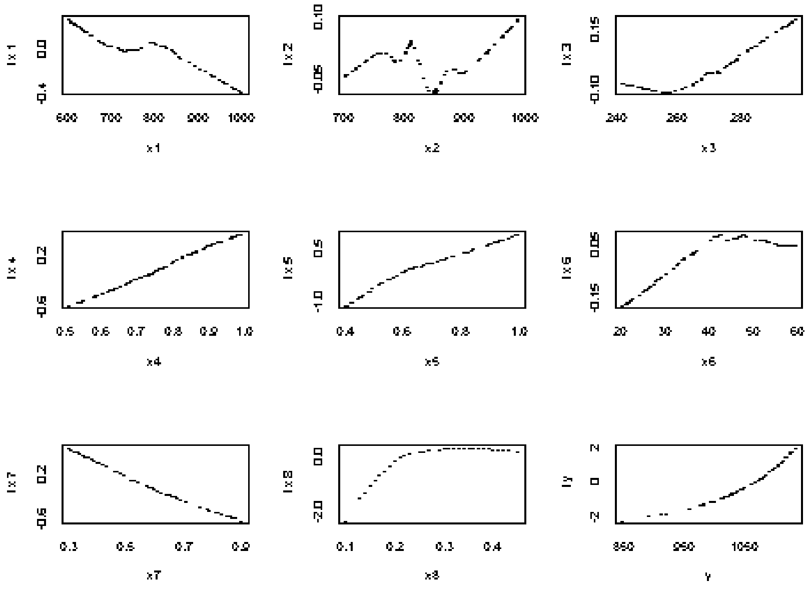


Figure 6-1-13: Plots transformed variables against original data of X2323f

To the 83 data points we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon \qquad (6\text{-}1\text{-}7)$$

The coefficient c = 1148.0. The a_i and b_i were as in Table 6-1-4.

i	1	2	3	4	5	6	7	8
b_i	0.1505	-0.3401	-5.549	309.5	479.2	3.491	-18.99	1860.9
a_i	-0.00014	0.00021	0.01119	-115.5	-232.5	-0.04117	-61.08	-2802.1

Table 6-1-4: Values of quadratic regression coefficients for X2323f

Letting

$$y_t = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}1\text{-}8)$$

The correlation between y and y_t (quadratic) is 0.908.

We improve the fit by using a formula of the form:

$$y = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 + \varepsilon \qquad (6\text{-}1\text{-}9)$$

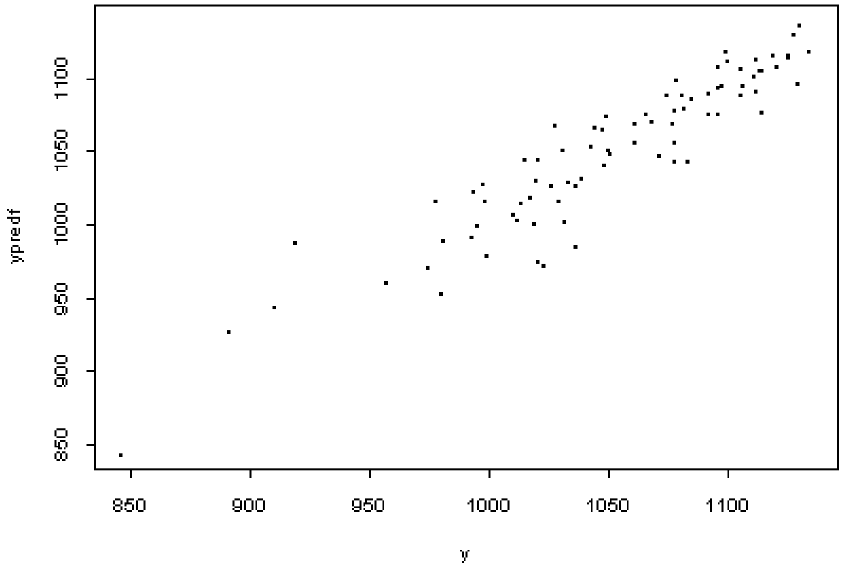


Figure 6-1-14: Scatter plot of y against y_{pred} for X2323f (cubic)

The coefficients turned out to be:

C_0 = -23098.4; C_1 = 63.95; C_2 = -0.05693; C_3 = 0.00001708.

Letting

$$y_{pred} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 \qquad (6\text{-}1\text{-}10)$$

the correlation achieved between y and y_{pred} is now 0.929. The scatter plot of y against y_{predf} is shown in Figure 6-1-14.

6.1.3 AVAS regression analysis for X22 (room length L < 600 cm)

We apply the AVAS algorithm and the plots of the transformed variables against original data are shown in Figure 6-1-15.

Analysing Figure 6-1-15, from the plot of x_8, which is the flame spread rate, it is clear that there is a change of behaviour at 0.66 m/sec. It turns out that there are 752 observations satisfying the constraints of length of room L < 600 cm and flame spread rate $R_f \geq 0.66$ m/sec, which are referred to as X223. And there are 311 observations satisfying the constraints L < 600 cm, R_f < 0.66 m/sec, which we call X222.

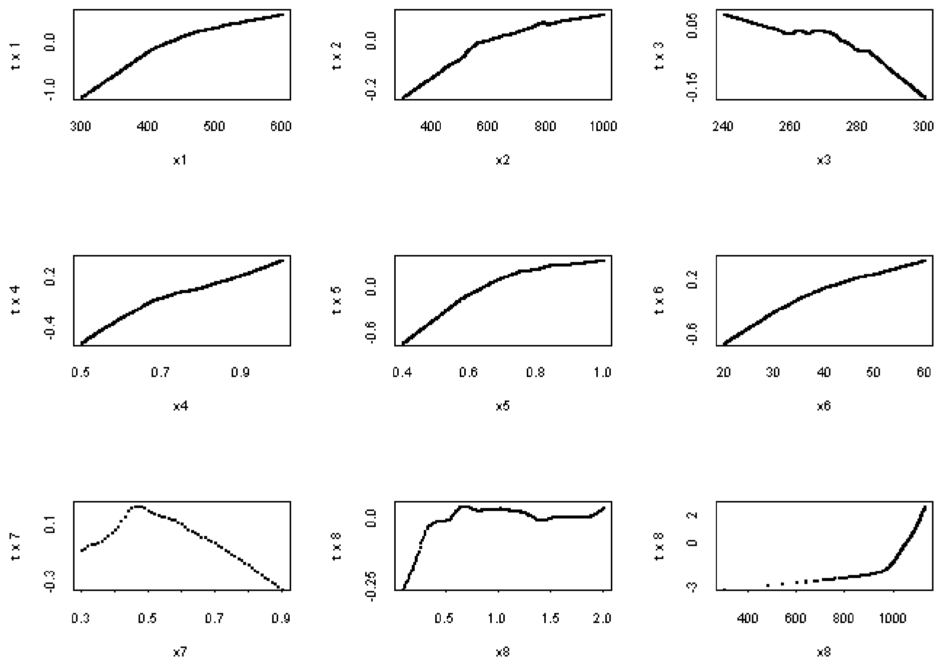


Figure 6-1-15: Plots of transformed variables against original data of X22

6.1.3.1 AVAS regression analysis for X223

We apply the AVAS algorithm and plot the transformed variables against the original data. The result of transformed data against the original data is shown in Figure 6-1-16. The correlation coefficient was 0.514.

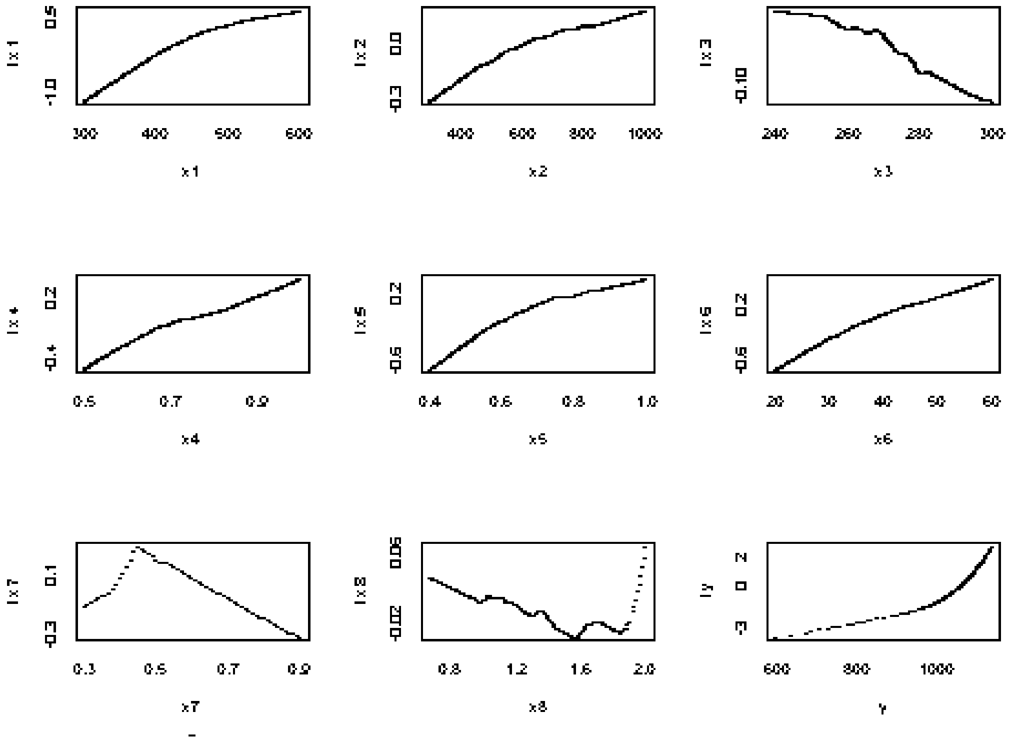


Figure 6-1-16: Plots transformed variables against original data for X223

From the plot of x_7, which is the fuel area factor f_A, it is clear that there is a change of behaviour at 0.455. This value, x_7 = 0.455, separated the data set into two new sub-sub-range data sets. It turns out that there are 202 observations satisfying the constraints of the length of the room L < 600 cm, flame spread rate R_f > 0.66 m/sec and f_A < 0.455, which is referred as X2232. And there are 550 observations satisfying the constraints of the L < 600 cm , $R_f \geq$ 0.66 m/sec and f_A > 0.455, which we call it X2233.

6.1.3.1.1 AVAS regression analysis for X2233

By using the AVAS regression analysis on X2233 data set we obtain the result of transformed maximum temperature against predicted value of AVAS shown in Figure 6-1-17. It is clear that there were seven data points outliers (17, 65, 41, 79, 27,171,112). Deleting them left us with 543 data points which are referred as X2233f. We apply the AVAS algorithm to X2233f and the plots of transformed variables against original data of X2233f. The result is shown in Figure 6-1-18, and the correlation coefficient is increased from 0.795 to 0.981.

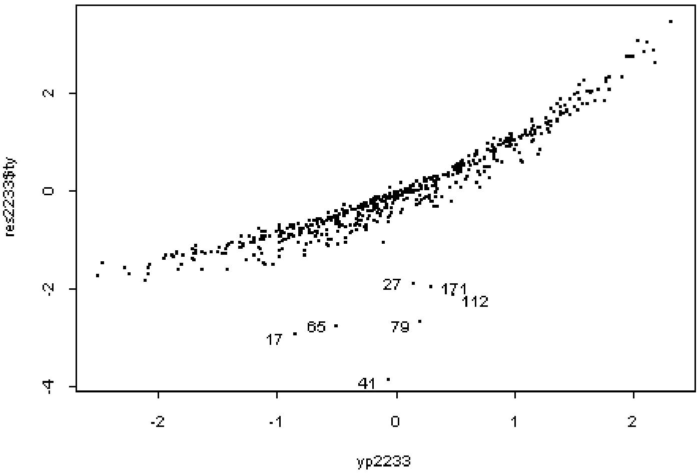


Figure 6-1-17: Scatter plot of transformed y against y_{p2233} for X2233

To the 543 data points we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon \qquad (6\text{-}1\text{-}11)$$

The coefficient c = 461.6. The a_i and b_i were as in Table 6-1-5.

i	1	2	3	4	5	6	7	8
b_i	1.033	0.03170	0.07184	248.6	287.8	2.234	-29.11	7.609
a_i	-0.00095	-0.00002	-0.00006	-96.62	-124. 2	-0.01720	-29.63	-3.086

Table 6-1-5: Values of quadratic coefficients for X2233f

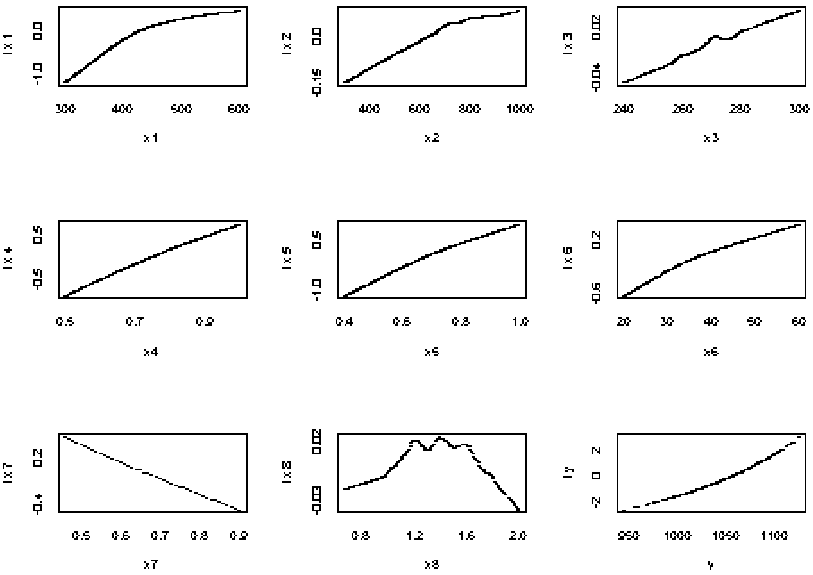


Figure 6-1-18: Plots of transformed variables against original data for X2233f

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}1\text{-}12)$$

the correlation between y and y_{pred} (quadratic fit) is 0.984. A scatter plot of y against y_{pred} for X2233f (quadratic fit) is shown in Figure 6-1-19.

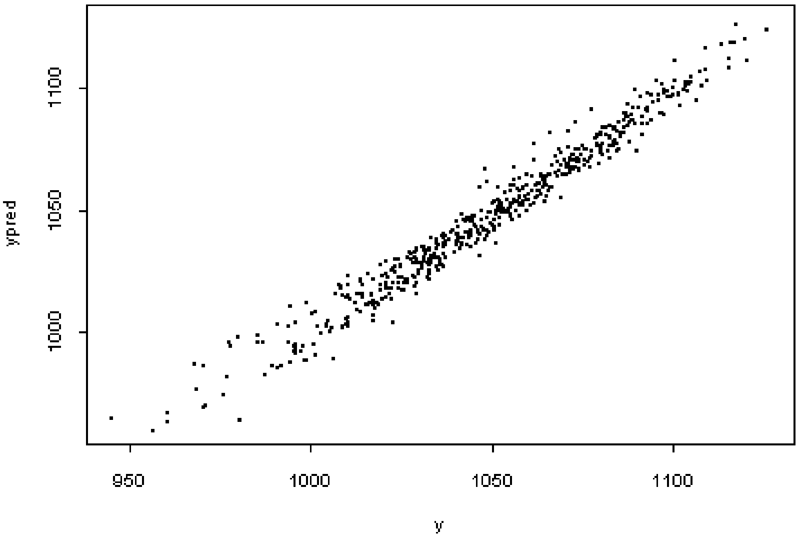


Figure 6-1-19: Scatter plot of y against y_{pred} for X2233f (quadratic fit)

When x_8 which is flame spread rate is ignored i = 1 to 7 instead of i = 1 to 8.

Again, to the 543 data points we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon . \tag{6-1-13}$$

The coefficient c = 458.8. The a_i and b_i were as in Table 6-1-6.

i	1	2	3	4	5	6	7
b_i	1.028	0.03217	0.1252	250.5	285.7	2.237	-27.85
a_i	-0.00095	-0.00002	-0.00015	-97.76	-122.7	-0.01722	-30.70

Table 6-1-6: Values of quadratic coefficients for X2233f(x_8 is ignored)

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \tag{6-1-14}$$

the correlation between y and y_{pred} (quadratic fit) is now 0.984. A scatter plot of y against y_{pred} for X2233f (quadratic fit) is shown in Figure 6-1-20.

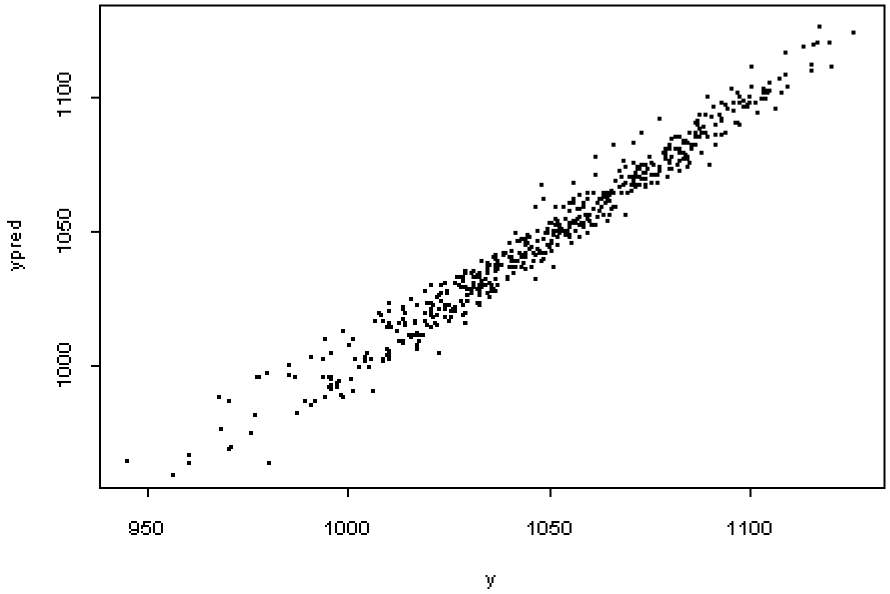


Figure 6-1-20: Scatter plot of y against y_{pred} for X2233f(x_8 is ignored)

6.1.3.1.2 AVAS regression analysis for X2232

By using the AVAS algorithm for X2232, the correlation coefficient was 0.598. Let us plot the transformed variables against the original data of X2232. The result is shown as in Figure 6-1-21. From the plot of variable x_7, which is the fuel area factor f_A, it is clear that there is a change of behaviour at 0.345. Therefore, this value (x_7 = 0.345) separates X2232 into two new sub-sub-range data sets. We have, when $x_7 <$ 0.345, X22321 with 69 observations and when $x_7 >$ 0.345 , X22322 with 133 observations.

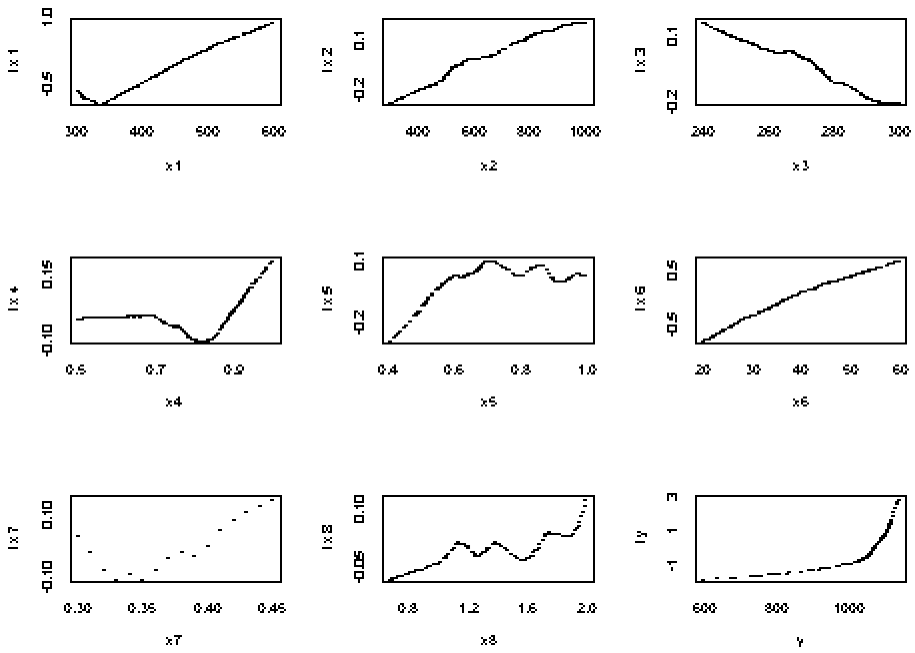


Figure 6-1-21: Plots transformed variables against original data for X2232

6.1.3.1.2.1 AVAS regression analysis for X22322

We use the AVAS analysis on data X22322. A scatter plot of transformed y against y_{pred} for X22322 is shown Figure 6-1-22. From the scatter plot, it is clear that there are some outlier data (11, 14, 15, 21, 22, 23, 24, 28, 29, 30, 32, 36, 37, 43, 57, 78, 88, 102, 109). Deleting them leaves us with X22322f which has 114 observations. And now the correlation coefficient achieved is 0.977.

The plot of the transformed variables against the original data of X22322f is shown in Figure 6-1-23.

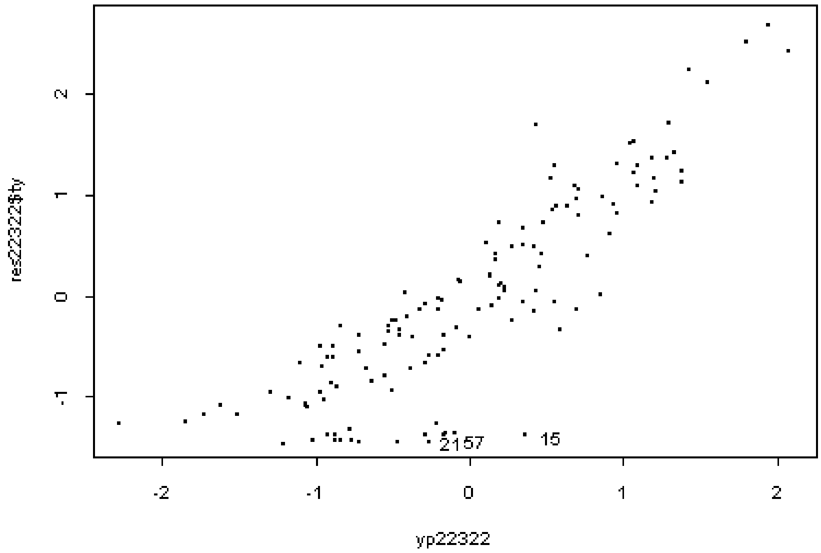


Figure 6-1-22: Scatter plot of transformed y against y_{pred} for X22322

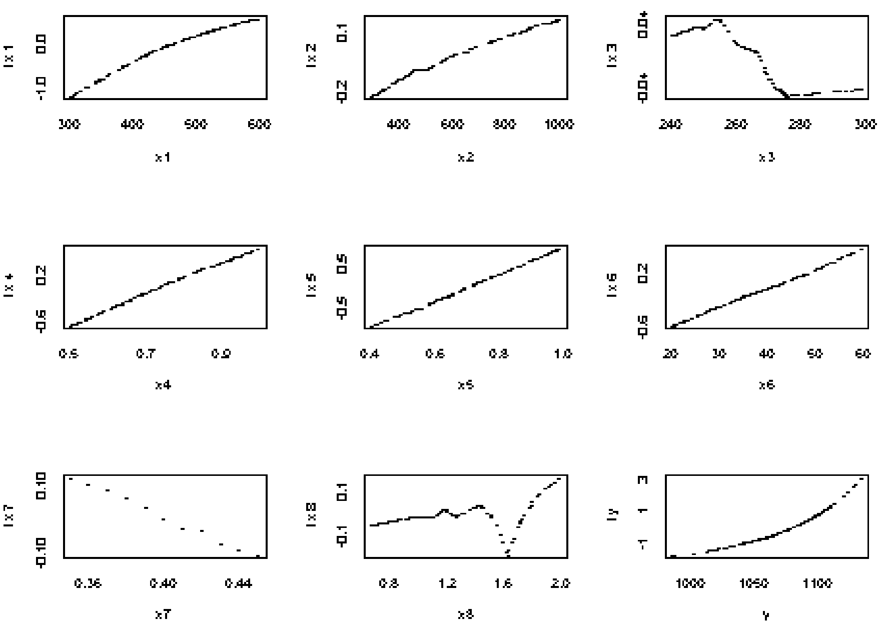


Figure 6-1-23: Plots transformed variables against original data for X22322f

To the 114 data points we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon . \tag{6-1-15}$$

The coefficient $c = 650.0$. The a_i and b_i were as in Table 6-1-7.

i	1	2	3	4	5	6	7	8
b_i	1.137	0.04865	-1.075	229.1	186.1	2.687	-81.40	1.573
a_i	-0.00101	-0.00002	0.00193	-100.1	-71.56	-0.02164	25.80	-0.9917

Table 6-1-7: Values of quadratic coefficients for X22322f

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \tag{6-1-16}$$

The correlation between y and y_{pred} (quadratic fitted) is 0.995. A scatter plot of y against y_{pred} is shown in Figure 6-1-24.

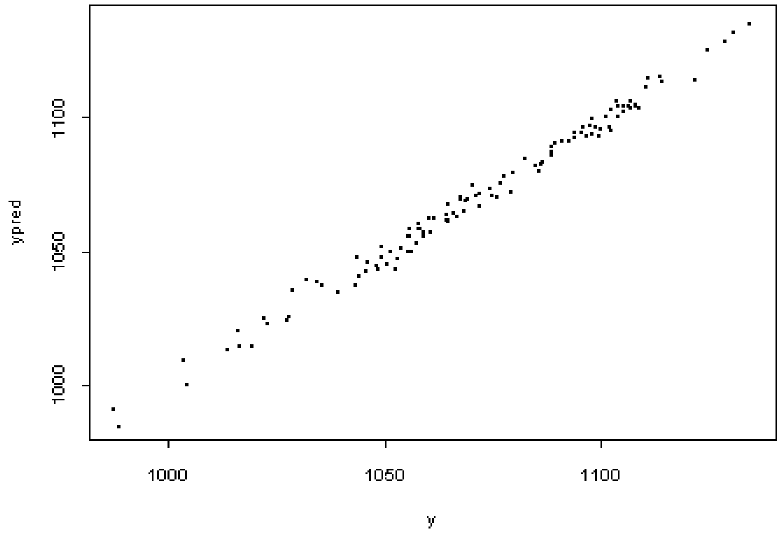


Figure 6-1-24: Scatter plot of y against y_{pred} for X22322f

6.1.3.1.2.2 AVAS regression analysis for X22321

By using the AVAS regression analysis on the data set X22321, the transformed variables against the original data are shown in Figure 6-1-25.

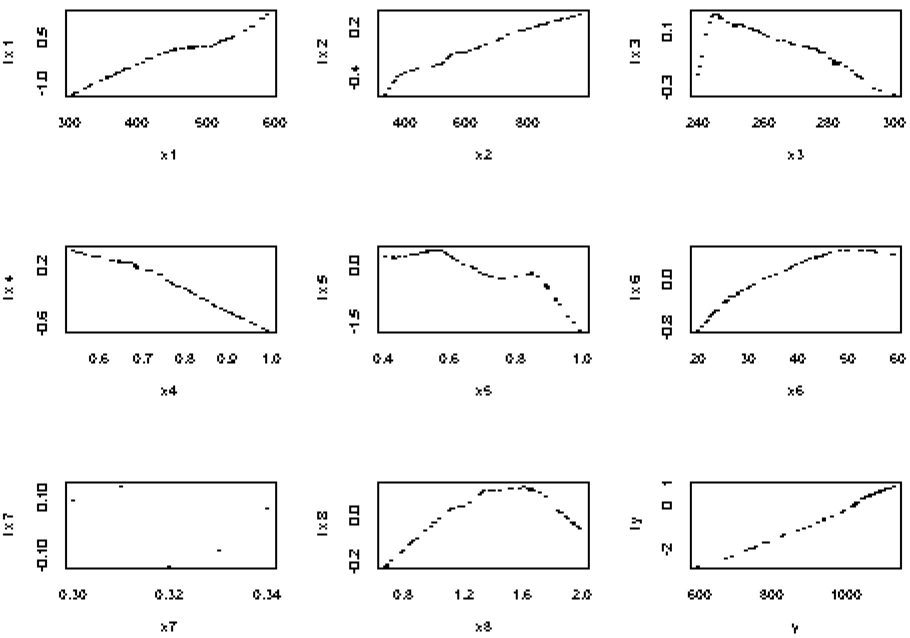


Figure 6-1-25: Plots of transformed variables against original data of X22321

To the data set X22321 we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon . \qquad (6\text{-}1\text{-}17)$$

The coefficient $c = 650.0$. The a_i and b_i were as in Table 6-1-8.

i	1	2	3	4	5	6	7	8
b_i	2.267	0.5354	12.11	-213.7	554.1	19.50	-24734	243.2
a_i	-0.00153	-0.00026	-0.02482	-59.53	-667.5	-0.1942	39029	-84.68

Table 6-1-8: Values of quadratic coefficients for X22321

Letting

$$y_t = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}1\text{-}18)$$

the correlation between y and y_t (quadratic) is 0.873.

We improve the fit by using a formula of the form:

$$y = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 + \varepsilon . \qquad (6\text{-}1\text{-}19)$$

The coefficients turned out to be:

C_0 = 2733.3; C_1 = -10.46; C_2 = 0.01484; C_3 = -6.072e-006.

Letting

$$y_{predf} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 \qquad (6\text{-}1\text{-}20)$$

the correlation between y and y_{predf} is now 0.938.

The scatter plot of y against y_{predf} is shown Figure 6-1-26.

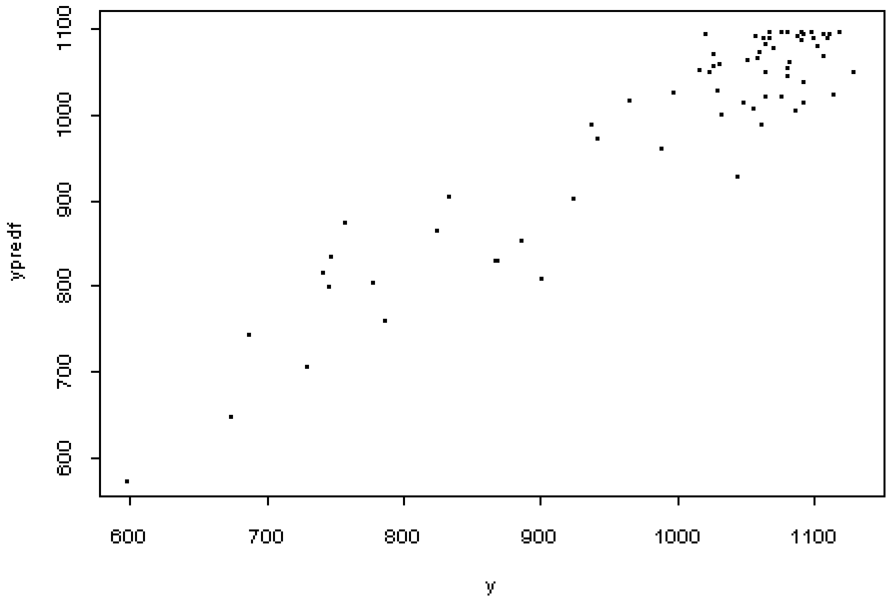


Figure 6-1-26: Scatter plot of y against y_{predf} for X22321

6.1.3.2 AVAS regression analysis for X222

To X222 (311 observations), we apply the AVAS algorithm and plot the AVAS predicted maximum temperature against transformed values. The result is shown in Figure 6-1-27. And the correlation coefficient was 0.407.

It is clear that there were some outliers data point. Deleting the outliers (5, 167, 7, 311,74, 240, 211, 10, 48, 2, 40, 250, 118, 49, 176, 257, 153, 301, 221, 186, 235, 4, 19, 6, 62) left us with X222f (282 observations).

Applying the AVAS algorithm to data set X222f (282 obs), the plots of transformed variables against original data is shown in Figure 6-1-28. And the correlation coefficient is now 0.979.

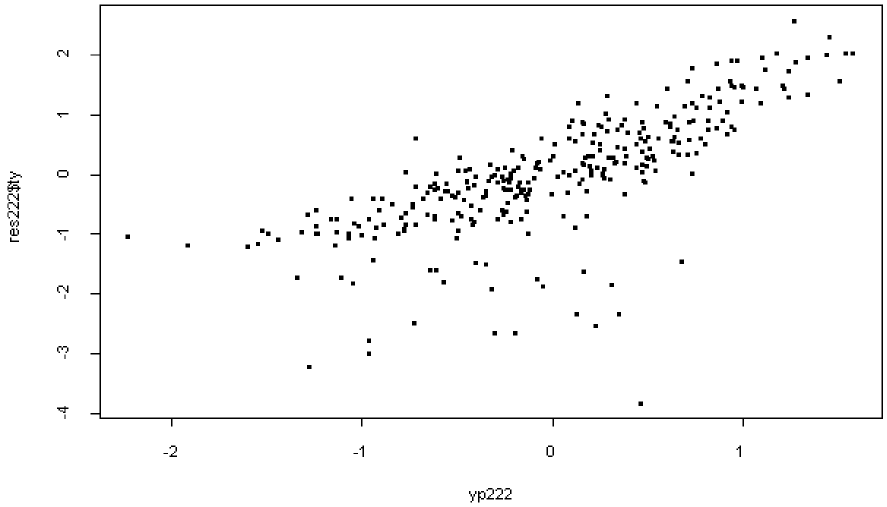


Figure 6-1-27: Scatter plot of y_{p222} against transformed y of X222

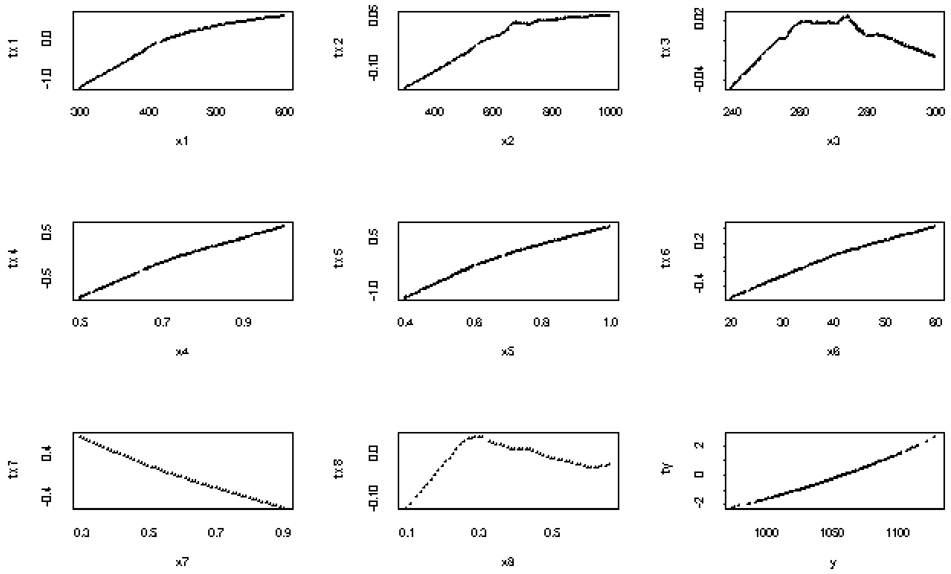


Figure 6-1-28: Plots transformed variables against original data of X222f

To the data set X222f we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon . \qquad (6\text{-}1\text{-}21)$$

The coefficient c = 465.2. The a_i and b_i were as in Table 6-1-9.

i	1	2	3	4	5	6	7	8
b_i	0.9427	0.03459	0.4844	237.9	292.9	1.623	-104.5	43.17
a_i	-0.00083	-0.00002	-0.00086	-94.74	-134.9	-0.01030	29.05	-54.58

Table 6-1-9: Values of quadratic coefficients for X222f

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}1\text{-}22)$$

the correlation between y and y_{pred} is 0.979.

And a scatter plot of y and y_{pred} is shown in Figure 6-1-29.

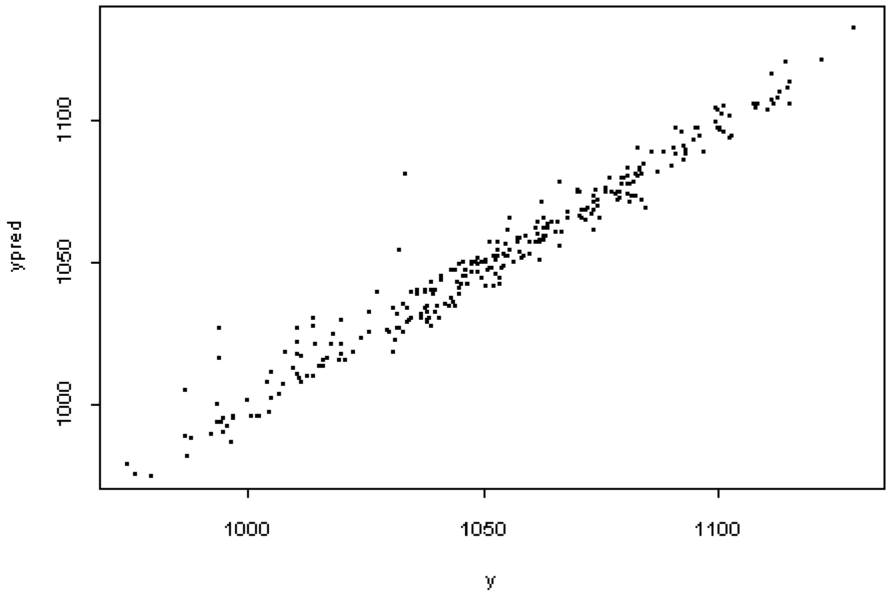


Figure 6-1-29: Scatter plot of y against y_{pred} for X222f

6.2 AVAS regression analysis for DOWC scenario

We apply the AVAS algorithm and plot the predicted transformed maximum temperature y_{pred} against the transformed maximum temperature as shown in Figure 6-2-1. The correlation coefficient was 0.415. Through applying the AVAS regression algorithm to the DOWC data, we obtain eight transformed inputs and one transformed output. Plots of the transformed data against the original data are shown in Figure 6-2-2.

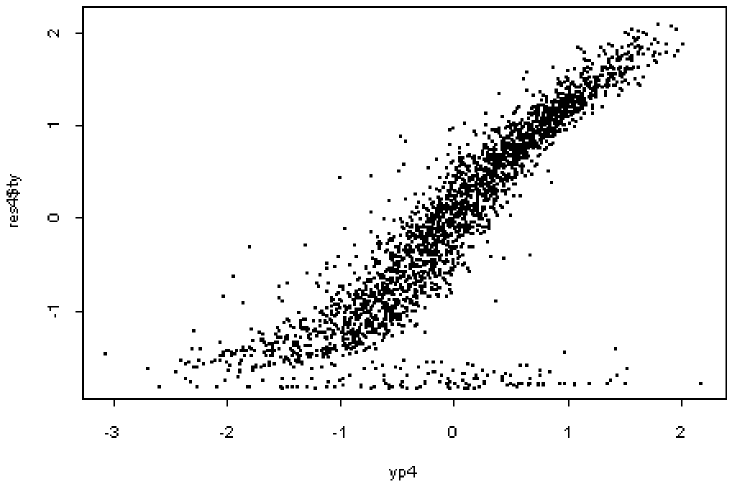


Figure 6-2-1: The predicted transformed output y_{pred} against the transformed output

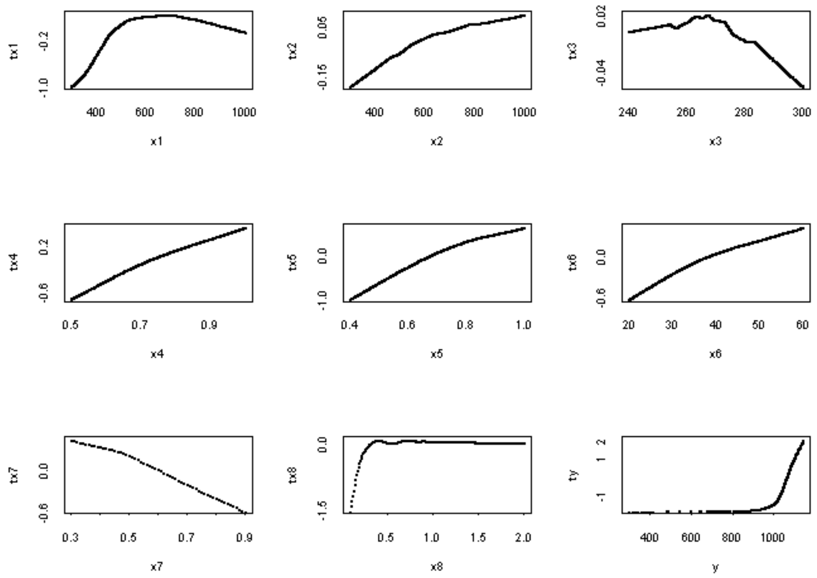


Figure 6-2-2: Plots of the transformed variables against the original data

Figure 6-2-2 leads to the following conclusions:

- From the plot of variable x_1, which is the length of the room, it is clear that there is a change of behaviour at a width of 700 cm.
- It is also clear that for variable x_8, which is the flame spread rate, there is a change of behaviour at 0.5 m/sec.

Therefore, we have obtained four sub-range data sets as follow:

X433: L > 700cm, $R_f \geq$ 0.5 m/sec; X432: L > 700cm, $R_f <$ 0.5 m/sec;

X423: L < 700cm, $R_f \geq$ 0.5 m/sec; X422: L < 700cm, $R_f <$ 0.5 m/sec.

6.2.1 AVAS regression analysis for X433

It turns out that there are 850 observations satisfying the constraints of the length of room L > 700cm, flame spread rate $R_f \geq$ 0.5 m/sec. We apply the AVAS algorithm to the data X433 and plot the transformed maximum temperature *ty* against the predicted transformed y_{pred} values. The result is shown in Figure 6-2-3. And the correlation coefficient achieved is 0.980.

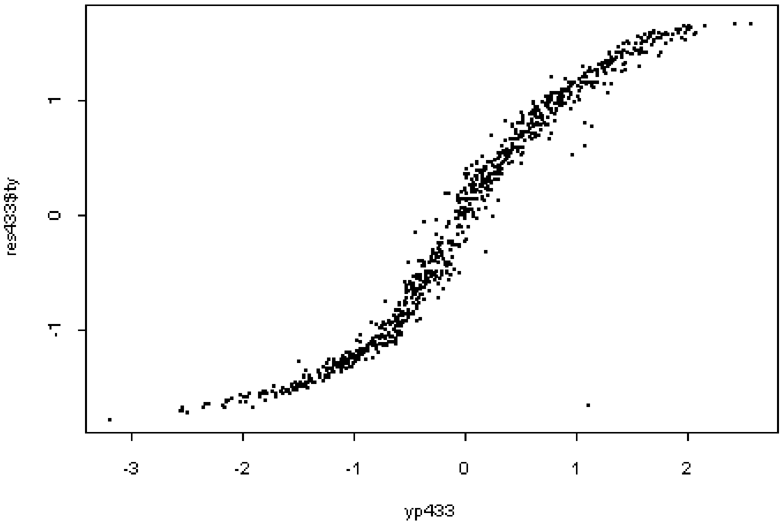


Figure 6-2-3: Plot the predicted transformed maximum temperature against the transformed values

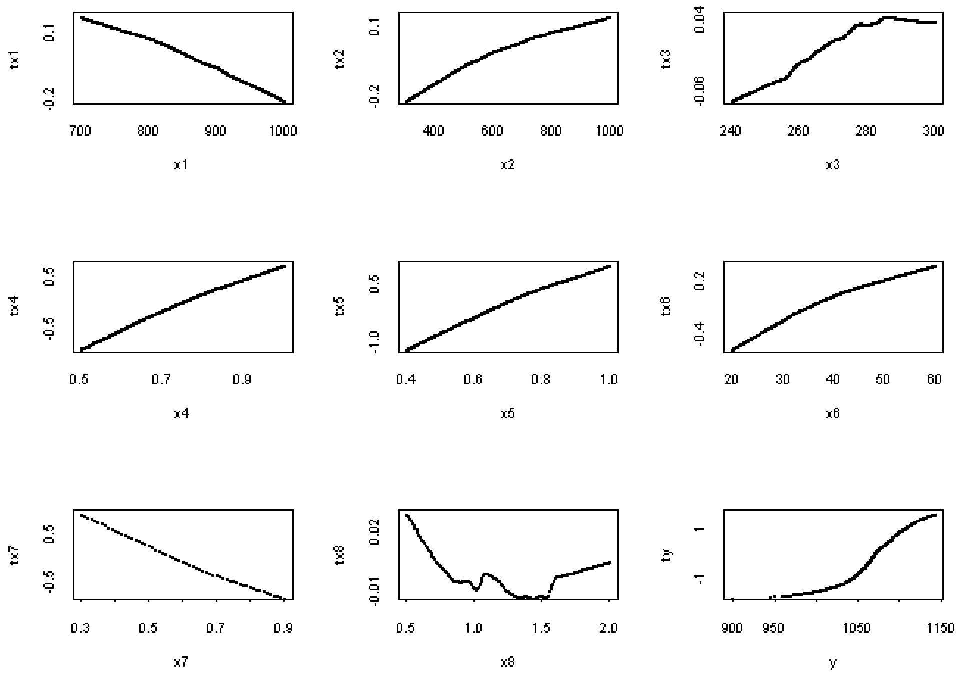


Figure 6-2-4: Plots of the transformed variables against the original data

To the 850 data points we fitted a quadratic regression formula of the form

$$y = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 + \varepsilon \text{, } I = 1 \text{ to } 8. \qquad (6\text{-}2\text{-}1)$$

The coefficient c = 566.0, and coefficients b_i and a_i were as in Table 6-2-1.

i	1	2	3	4	5	6	7	8
b_i	0.0786	0.0649	1.047	315.8	322.6	2.580	-134.3	-6.703
a_i	-0.0001	0.0000	-0.0018	-125.6	-130.4	-0.0198	14.50	2.310

Table 6-2-1: Values of quadratic regression coefficients for X433

Letting

$$y_t = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 \qquad (6\text{-}2\text{-}2)$$

the correlation between y and y_t is 0.952. A scatter plot of y and y_t is shown in Figure 6-2-5.

The second step in the fitting is to improve the fit of y_t to y by using a cubic regression formula of the form

$$y_{ppred} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 + \varepsilon^* . \qquad (6\text{-}2\text{-}3)$$

By using Linear Least-Squares Fit the coefficients turned out to be:
C_0 = 1123.2; C_1 = 0.6684; C_2 = 0.002190; C_3 = -3.7057e-006.

Setting

$$y_{ppred} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 \qquad (6\text{-}2\text{-}4)$$

the correlation between y and y_{ppred} is now 0.955.
A scatter plot of y and y_{pred} is shown in Figure 6-2-6.

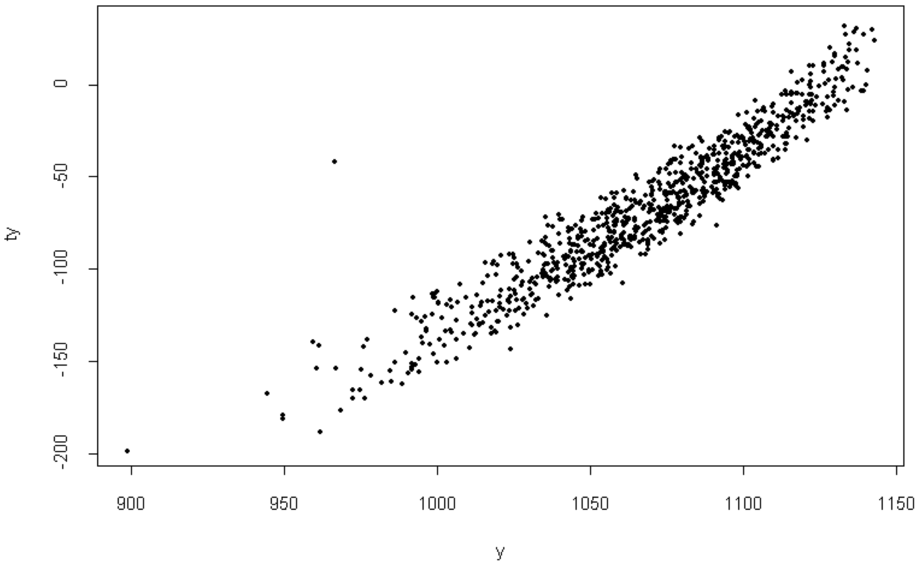


Figure 6-2-5: A scatter plot of y_t against original maximum temperature y

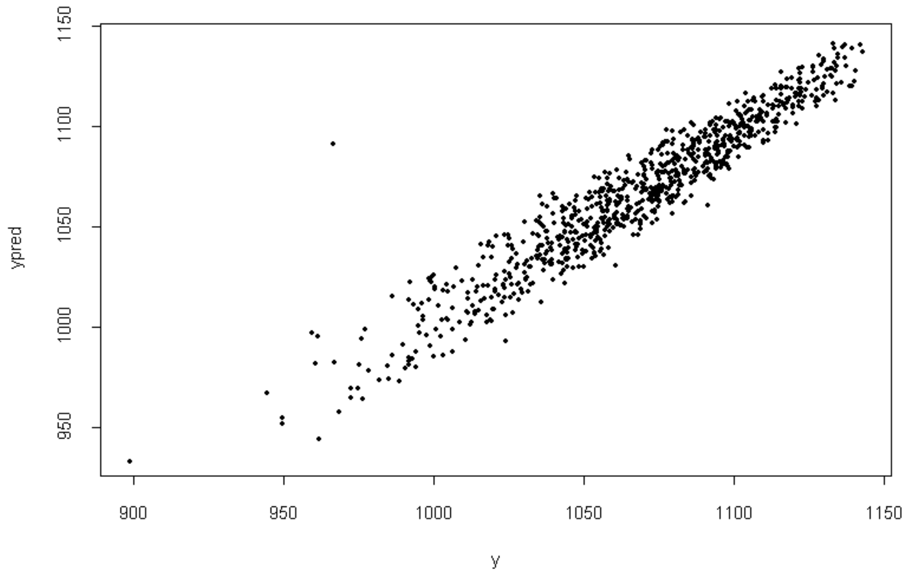


Figure 6-2-6: A scatter plot of y_{ppred} against original maximum temperature y

6.2.2 AVAS regression analysis for X432

There are 202 data sets satisfying the constraints of L > 700 cm, and R_f < 0.5 m/sec. By using the AVAS regression analysis on X432, the correlation coefficient is 0.577. A scatter plot of transformed output against predicted value in the AVAS is shown in Figure 6-2-7. Plots of transformed variables against the original data are shown in Figure 6-2-8. It is clear that there are different modes of fire growth for room width W_r, that is x_2, below and above 700 cm. So we separated X432 into two new sub-sub-range data sets: when W_r < 700cm, we have X4322 with 121 observations; when W_r > 700cm, we have X4323 with 81 observations.

6.2.2.1 AVAS regression analysis for X4322

It turns out that there are 121 observations satisfying the constraints of X4322. Through using the AVAS regression analysis on X4322 data set, the result of predicted maximum temperature against transformed maximum temperature is shown in Figure 6-2-9. Deleting the outliers data (109, 83, 88, 119, 113, 43, 82) left us with 114 observations which we call X4322f. and the correlation coefficient achieved is 0.972. We apply the AVAS algorithm and the plots of the transformed variables against original data of X4322f are shown as in Figure 6-2-10.

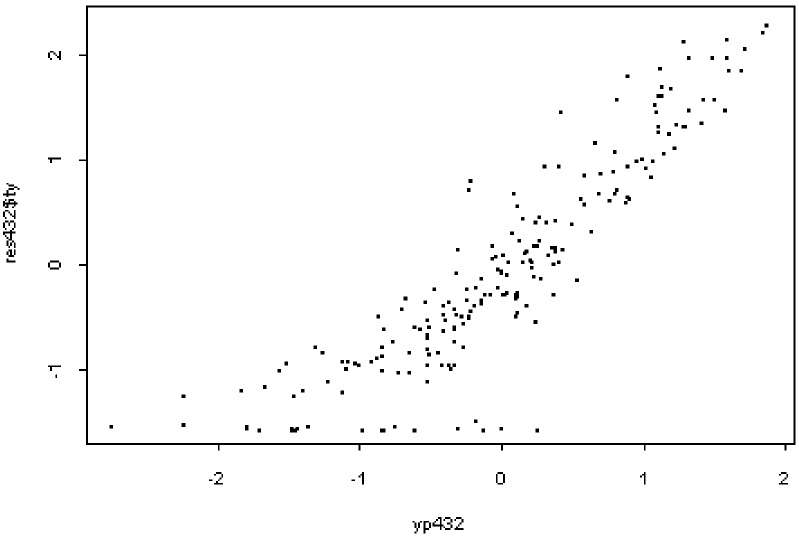


Figure 6-2-7: Scatter plot of transformed *y* against y_{p432} for X432

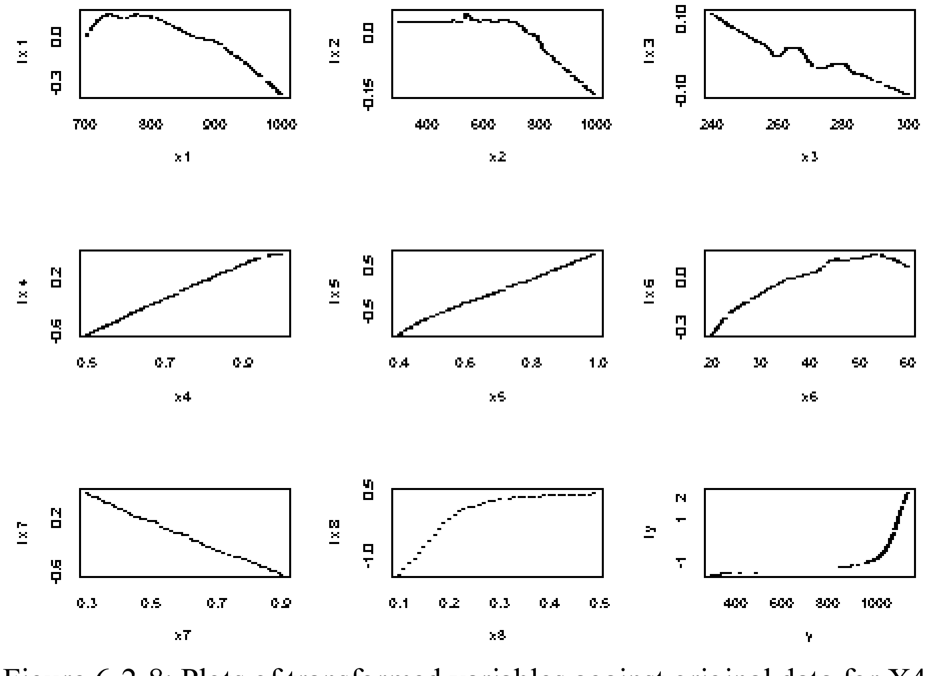


Figure 6-2-8: Plots of transformed variables against original data for X432

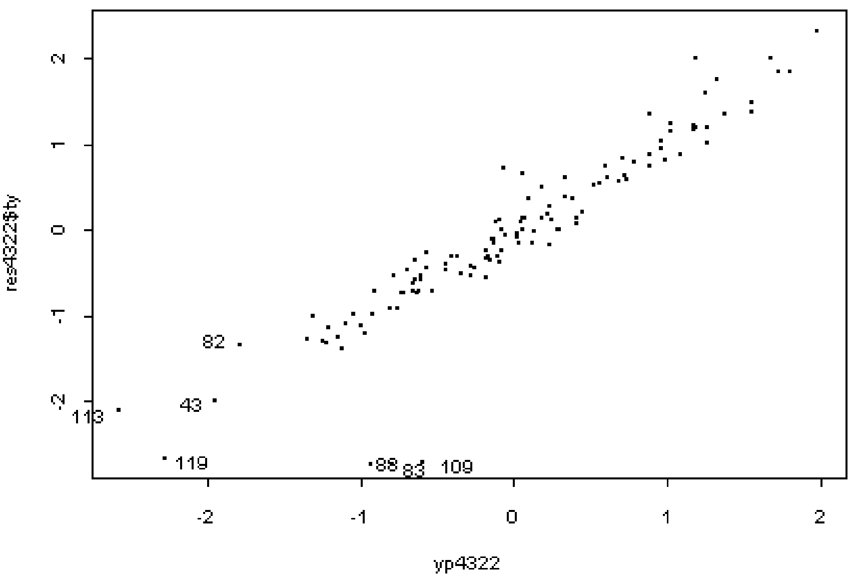


Figure 6-2-9: Scatter plot of transformed *y* against y_{p4322} for X4322

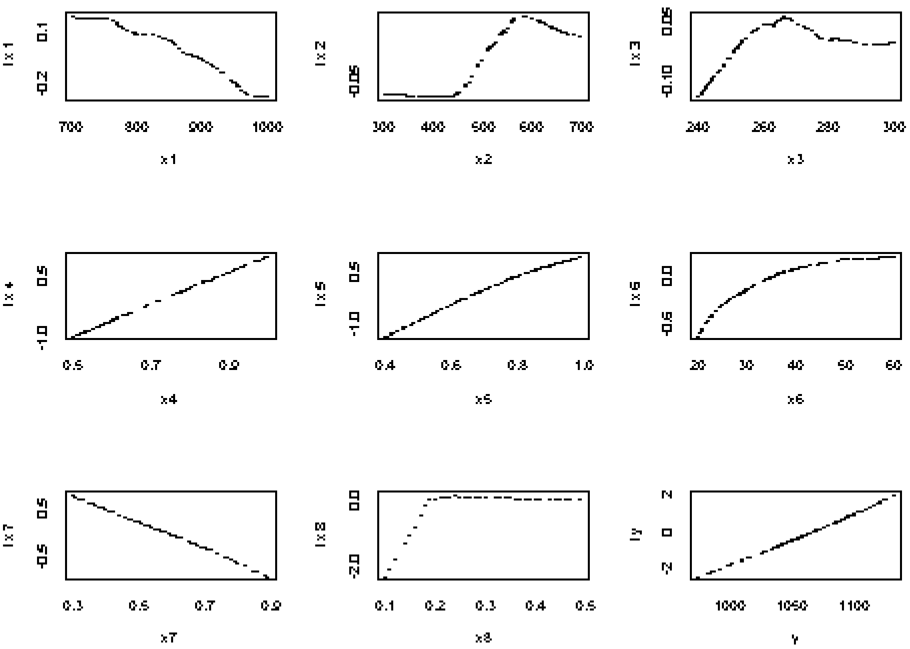


Figure 6-2-10: Plots transformed variables against original data of X4322f

To the 114 data points we fitted a quadratic regression formula of the form (6-2-1)

.

The coefficient *c* was 33.43 and b_i and a_I were as in Table 6-2-2.

i	1	2	3	4	5	6	7	8
b_i	0.2645	0.09921	3.791	146.7	247.4	3.676	-26.89	627.4
a_i	-0.00019	-0.00007	-0.00681	-5.327	-75.42	-0.03627	-53.79	-896.8

Table 6-2-2: Values of quadratic regression coefficients for X4322f

Letting

$$y_t = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}2\text{-}5)$$

it was found that the correlation between *y* and y_t is 0.937. And a scatter plot of *y* against y_t is shown in Figure 6-2-11.

The second step in the fitting is to improve the fit of y_t to *y* by using a cubic regression formula of the form

$$y = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 + \varepsilon . \qquad (6\text{-}2\text{-}6)$$

The coefficients turned out to be:
C_0 = 34496.4; C_1 = -99.02; C_2 =0.09651; C_3 = -0.00003099.

Letting

$$y_{predf} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 \qquad (6\text{-}2\text{-}7)$$

the correlation achieved between *y* and y_{predf} is now 0.942. And a scatter plot of *y* against y_{predf} is shown in Figure 6-2-12.

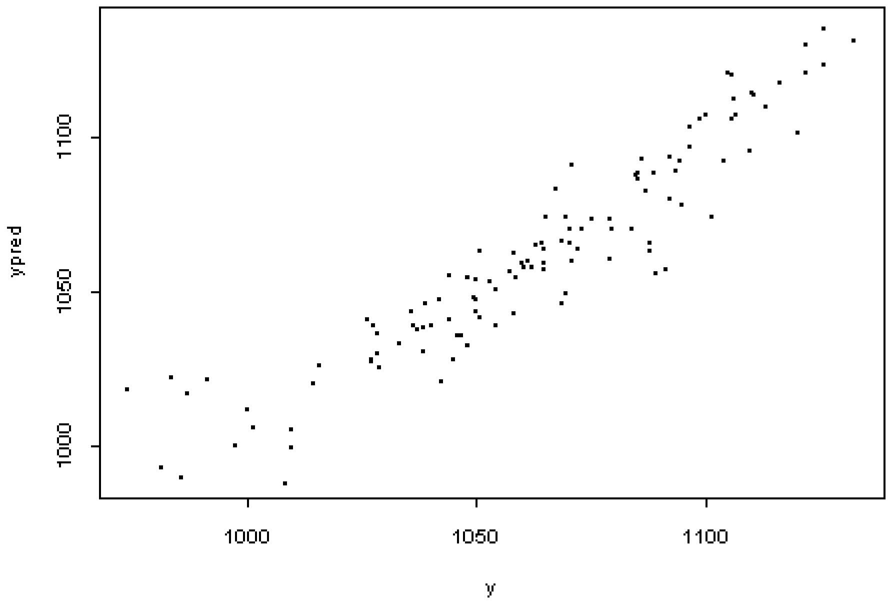


Figure 6-2-11: Scatter plot of *y* against y_t (quadratic) for X4322f

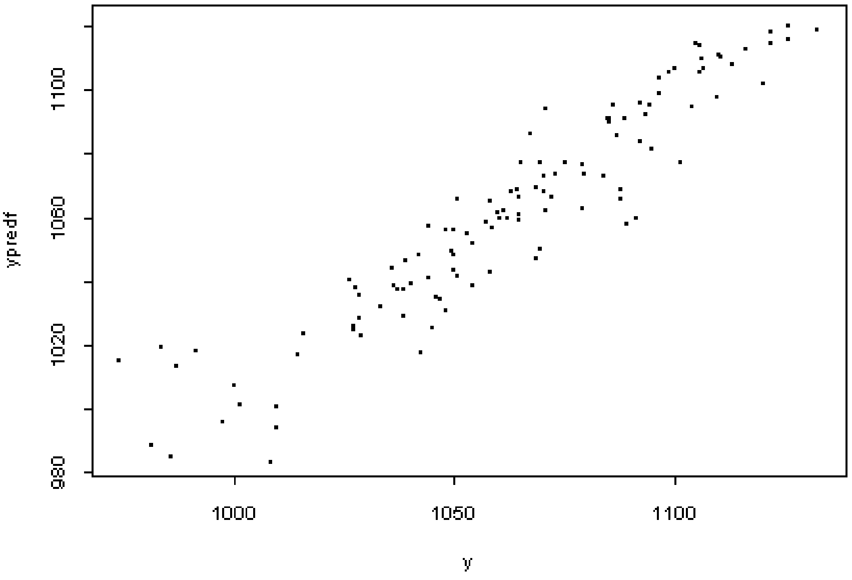


Figure 6-2-12: Scatter plot of *y* against y_{predf} (cubic fit) for X4322f

6.2.2.2 AVAS regression analysis for X4323(81 obs)

Through using AVAS regression analysis on the X4323 data set it is easy to see that some points are outliers (10, 22, 30, 35, 37, 39, 49, 54, 56, 57, 64, 65, 66, 74, 78, 81). Deleting them left us with 65 observations, which we call X4323f. And the correlation coefficient achieved is 0.969.

We apply the AVAS algorithm to X4323f. Plots of the transformed variables against original data of X4323f are shown in Figure 6-2-13.

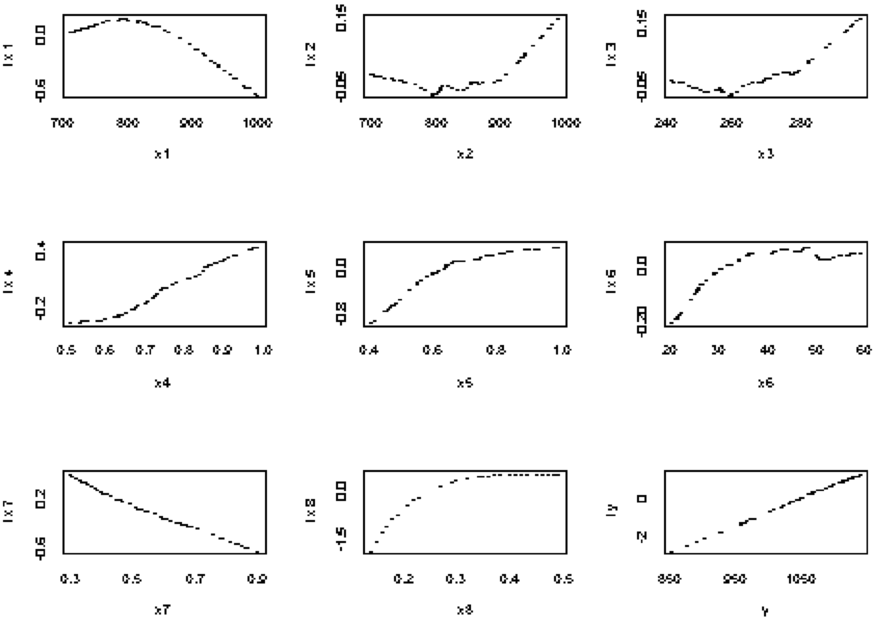


Figure 6-2-13: Plots of transformed variables against original data of X4323f

To the 65 data points (X4323f), we fitted a quadratic regression formula of the form (6-2-1).

.

The coefficients are c = 772.5 and b_i and a_I in Table 6-2-3.

i	1	2	3	4	5	6	7	8
b_i	2.842	-0.1562	-10.30	-316.9	836.6	4.246	-331.4	1913.7
a_i	-0.00177	0.00011	0.01958	301.6	-491.5	-0.04846	166.4	-2477.7

Table 6-2-3: Values of quadratic regression coefficients for X4323f

Letting

$$y_t = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}2\text{-}8)$$

it was found that the correlation between y and y_t is 0.964. And a scatter plot of y against y_t is shown in Figure 6-2-14.

The second step in the fitting is to improve the fit of y_t to y by using a cubic regression formula of the form

$$y = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 + \varepsilon \,. \qquad (6\text{-}2\text{-}9)$$

The coefficients turned out to be:

C_0 = -3286.1; C_1 = 8.634; C_2 = -0.005496; C_3 = 1.156e-006.

Letting

$$y_{pred} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 \qquad (6\text{-}2\text{-}10)$$

the correlation achieved between y and y_{pred} is now 0.973. A scatter plot of y against y_{pred} is shown in Figure 6-2-15.

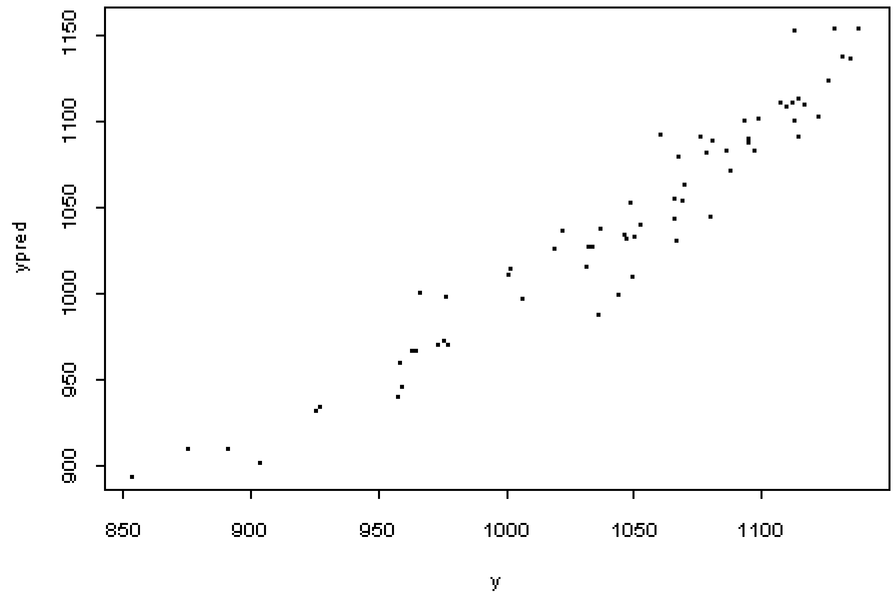


Figure 6-2-14: Scatter plot of y against y_t for X4323f

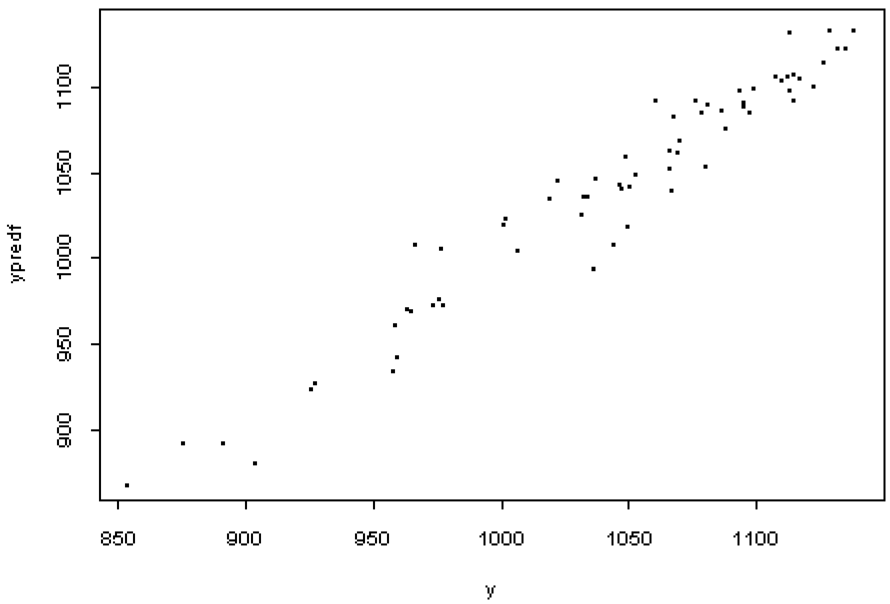


Figure 6-2-15: Scatter plot of y against y_{pred} for X4323f

6.2.3 AVAS regression analysis for X422

There are 310 data points satisfying the constraints of L < 700 cm, and R_f < 0.5 m/sec. By using the AVAS regression analysis on X422, the correlation coefficient was 0.4418. The result of plots of transformed variables against the original data is shown in Figure 6-2-16. It is clear that there are different modes of fire growth for room width W_r, that is x_2, for below and above 760 cm. So X422 is split into two new sub-sub-range data sets: when room width W_{r} < 760 cm , we have X4222 with 200 observations; when W_{r} > 760 cm , data set X4223 with 110 observations.

6.2.3.1 AVAS regression analysis for X4222

Through using the AVAS analysis on X4222 data set, the result of predicted maximum temperature against transformed maximum temperature is shown in Figure 6-2-17. Deleting the outliers data (59, 120, 72, 116, 155, 193, 9, 16, 57, 152, 42, 180, 19, 5, 185, 102, 156, 123, 109, 8) left us with 180 observations which we call X4222f. and the correlation coefficient achieved is 0.960.

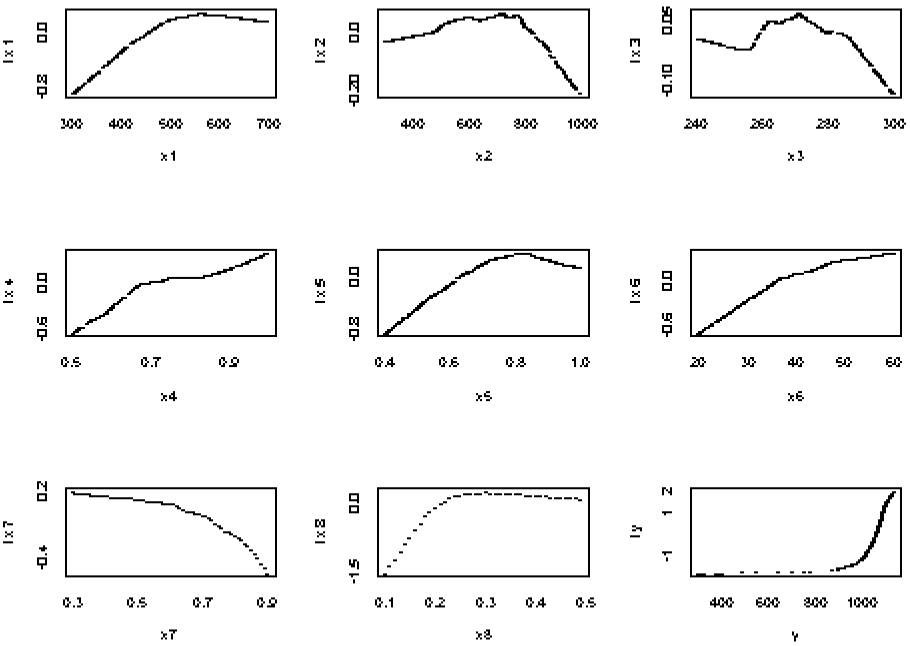


Figure 6-2-16: Plots of transformed variables against original data for X422

We apply the AVAS algorithm to X4222f. The plots of transformed variables against original data of X4222f are shown in Figure 6-2-18.

To the 180 data points we fitted a quadratic regression formula of the form (6-2-1)

The coefficient *c* was 509.9. The coefficients a_i and b_i were as in Table 6-2-4.

i	1	2	3	4	5	6	7	8
b_i	0.8394	0.1048	-0.08281	218.6	322.9	1.940	-112.1	228.6
c_i	-0.00073	-0.00009	0.00027	-81.44	-152.1	-0.01333	30.65	-345.4

Table 6-2-4: Values of quadratic regression coefficients for X4222f

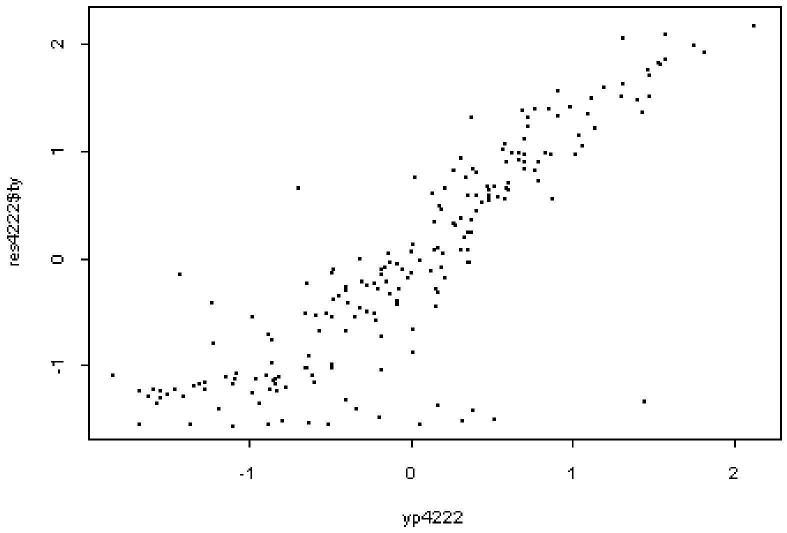


Figure 6-2-17: Scatter plot of transformed *y* against predicted *y* for X4222

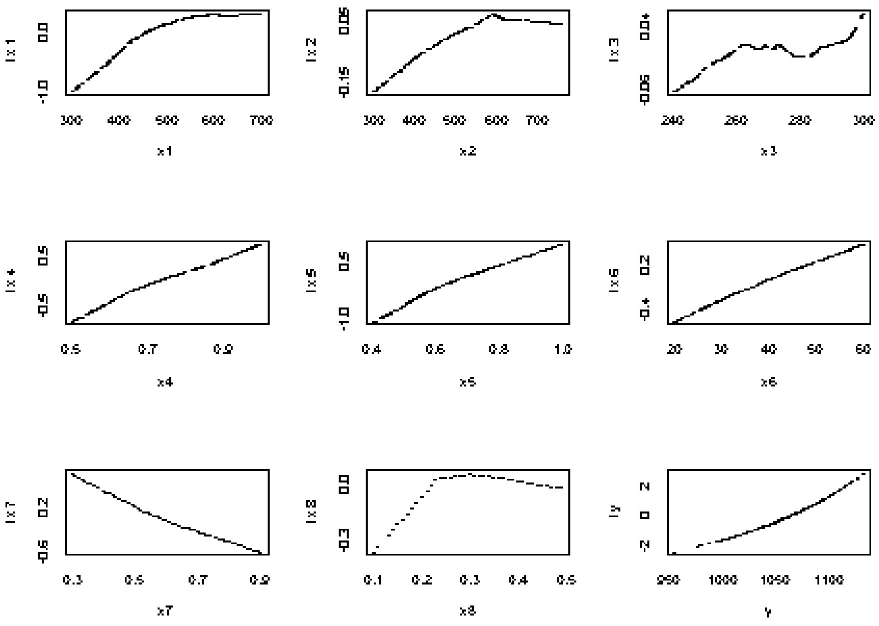


Figure 6-2-18: Plots of transformed variables against original data for X4222f

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}2\text{-}11)$$

it was found that the correlation between y and y_{pred} was 0.964.

A scatter plot of y against y_{pred} is shown in Figure 6-2-19.

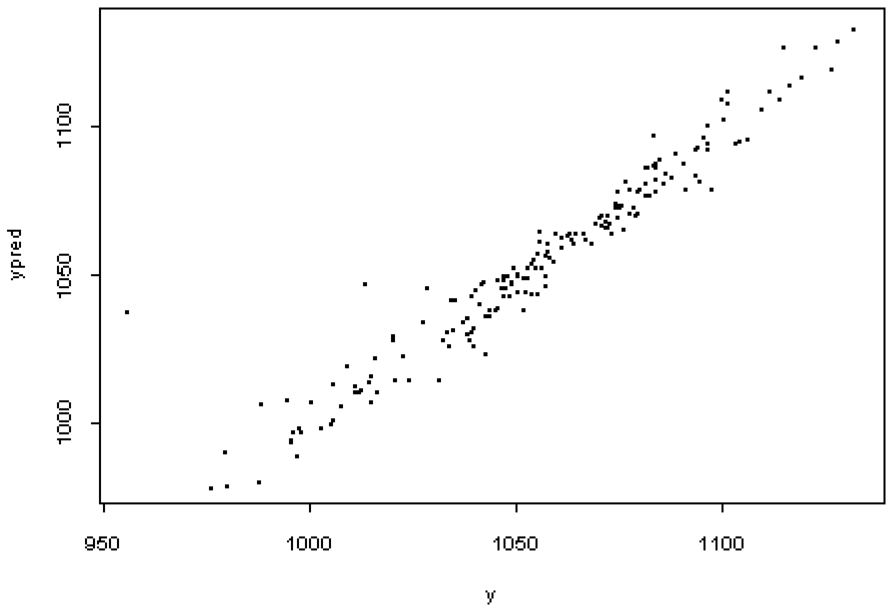


Figure 6-2-19: Scatter plot of y against y_{pred} for X4222f (quadric fit)

6.2.3.2 AVAS regression analysis for X4223

A plot of predicted maximum temperature against transformed maximum temperature using AVAS regression analysis on X4223 data set is shown in Figure 6-2-20. Deleting the outliers (2, 5, 6, 8, 13, 15, 22, 23, 26, 37, 45, 57, 62, 67, 87, 89, 93, 96, 100, 110) left us with 90 observations which are called X4223f. We apply the AVAS analysis to X4223f. The correlation coefficient achieved rises from 0.479 to 0.963. The plots of transformed variables against original data of X4223f are shown in Figure 6-2-21.

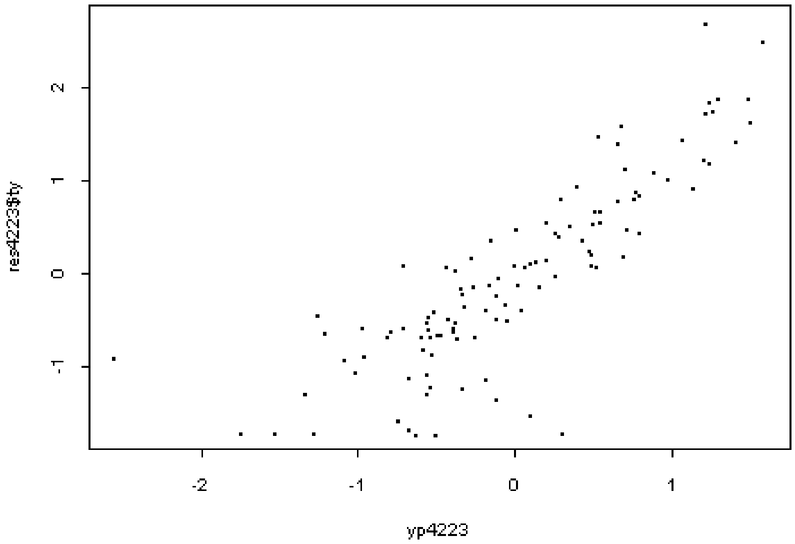


Figure 6-2-20: Scatter plot of transformed y against y_{p4223} for X4223

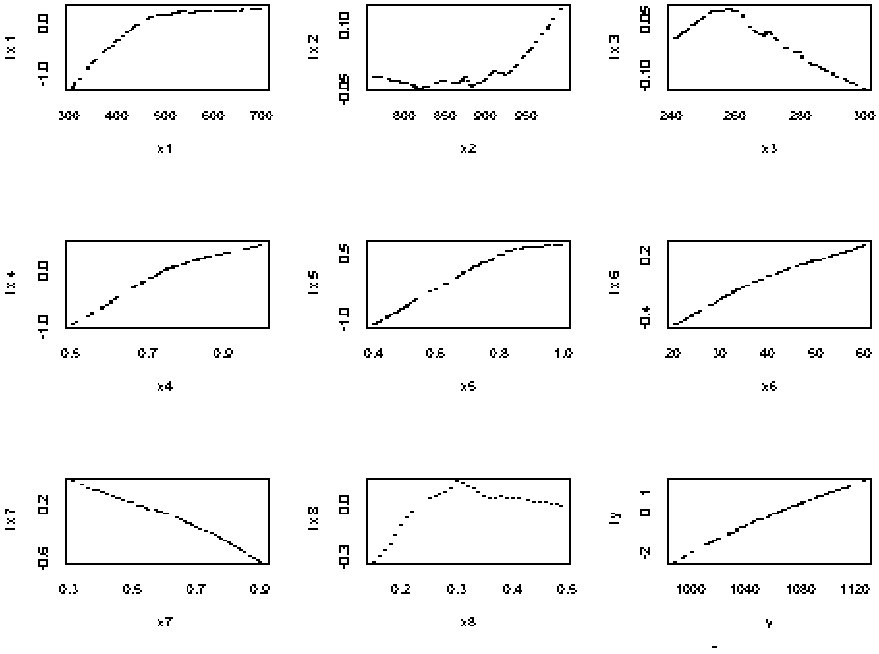


Figure 6-2-21: Plots of transformed variables against original data for X4223f

To data set X4223f, we fitted a quadratic regression formula of the form (6-2-1)

.

The coefficient *c* was 580.6. The a_i and b_i were as in Table 6-2-5.

i	1	2	3	4	5	6	7	8
b_i	0.8521	-0.6834	1.614	296.8	315.6	1.058	-5.492	235.3
a_i	-0.00074	0.00039	-0.00312	-132.5	-139.9	-0.00601	-49.96	-313.5

Table 6-2-5: Values of quadratic coefficients for X4223f

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}2\text{-}12)$$

it was found that the correlation between y and y_{pred} was 0.959. The scatter plot of y against y_{pred} is shown in Figure 6-2-22.

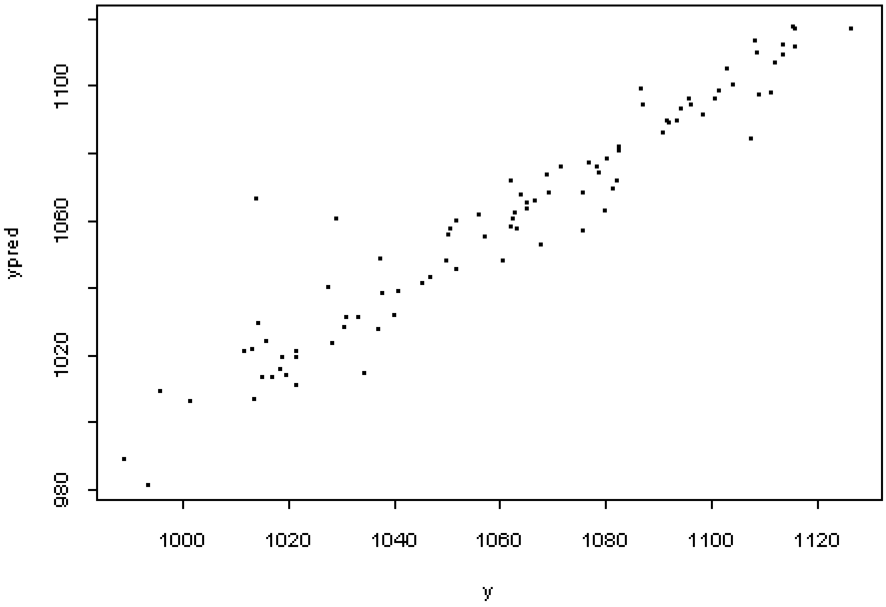


Figure 6-2-22: Scatter plot of y against y_{pred} (quadratic) for X4223f

6.2.4 AVAS regression analysis for X423

There are 1138 simulation data points in X423 (satisfying the constraints of $L < 700$ cm , $R_f > 0.5$m/sec). Using the AVAS algorithm analysis on the data set X423, the correlation coefficient was 0.5182. The plots of transformed variables against the original data are shown in Figure 6-2-23.

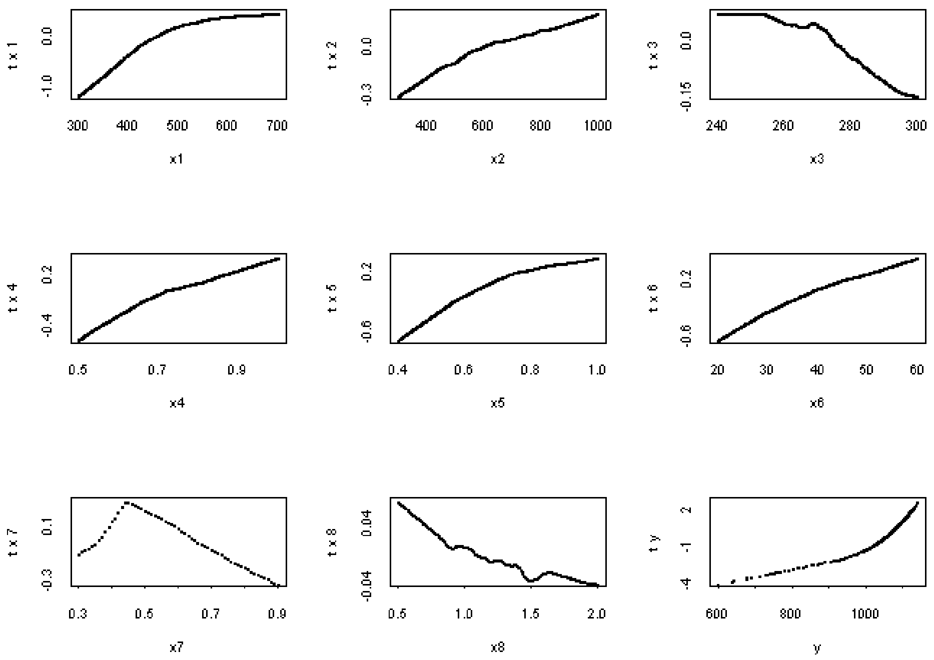


Figure 6-2-23: Plots of the transformed variables against the original data (X423)

In Figure 6-2-23, from the plot of variable x_7, which is the fuel area factor f_A, it is clear that there is a change of behaviour at a fuel area factor of 0.45. So we separated the data set X423 into two new sub-sub-range data sets. We call them X4233 and X4232

6.2.4.1 AVAS regression analysis for X4233

It turns out that there are 849 observations satisfying the constraints of fuel area factor $f_A > 0.45$. We apply the AVAS algorithm to X4233 and the correlation coefficient is 0.853. The scatter plot of the AVAS predicted output against transformed values is shown in Figure 6-2-24. It is obvious that there are 7 outliers, 47th, 122th, 144th, 189th, 285th, 63th, 88th. Deleting them left us with 842 observations which we denoted by X4233f. By applying the AVAS algorithm to X4233f, the correlation coefficient achieved is 0.9825. The plots of transformed variables against original data are shown in Figure 6-2-25.

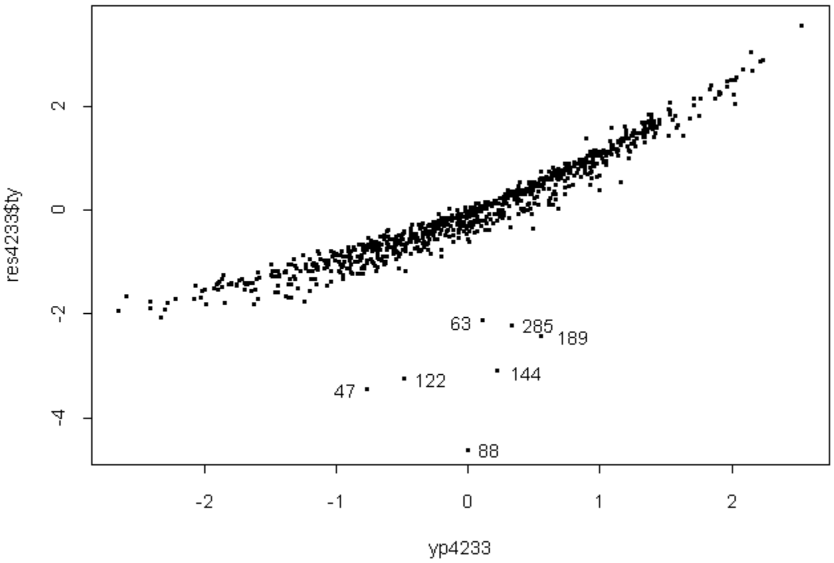


Figure 6-2-24: Scatter plot of y_{pred} against transformed y for X4233

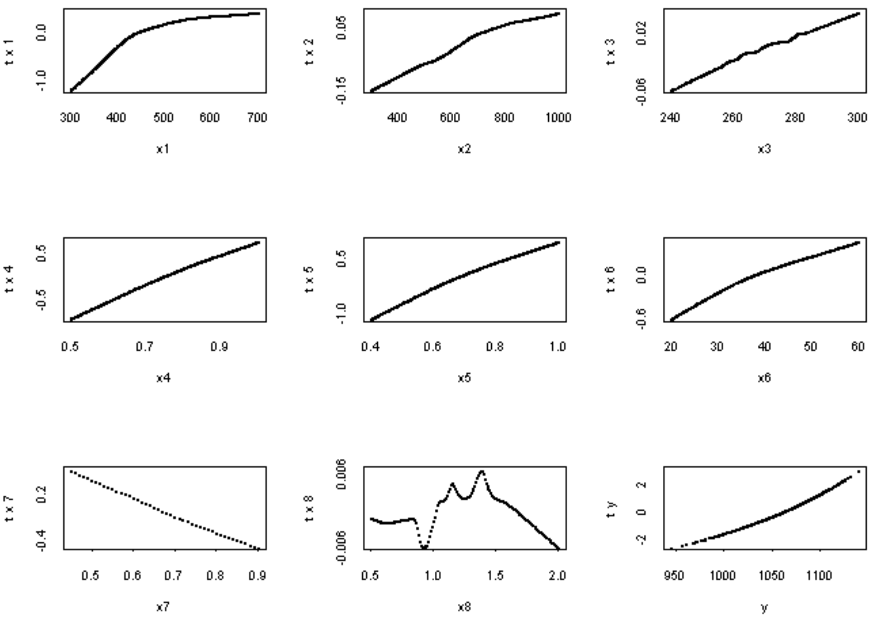


Figure 6-2-25: Plots of the transformed variables against the original data (X4233f)

To data set X4233f, we fitted a quadratic regression formula of the form (6-2-1)

The coefficient c was 481.8, coefficients b_i and a_I were in Table 6-2-6.

i	1	2	3	4	5	6	7	8
b_i	0.7527	0.02394	0.5466	244.4	288.3	2.040	-79.87	-0.1892
a_i	-0.00063	-0.00001	-0.00092	-90.59	-121.5	-0.01490	3.638	-0.1799

Table 6-2-6: Values of quadratic regression coefficients for X4233f

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}2\text{-}13)$$

it was found that the correlation between y and y_{pred} is 0.9826. And a scatter plot of y against y_{pred} is shown in Figure 6-2-26.

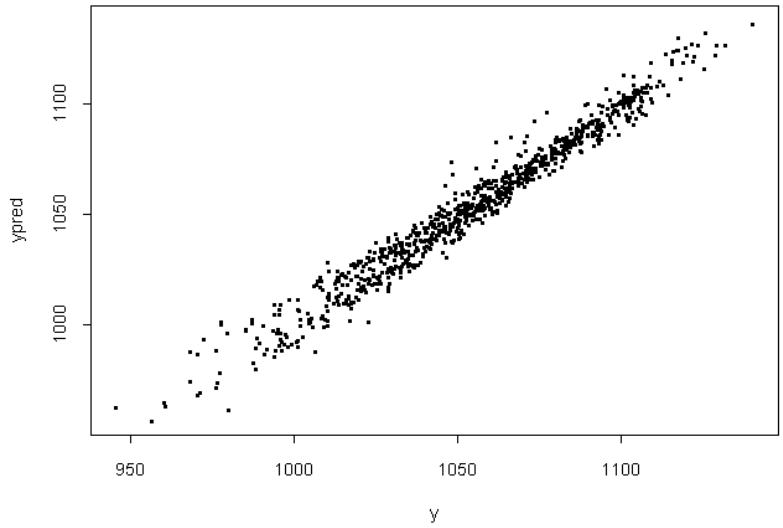


Figure 6-2-26: Scatter plot of y against y_{pred}

6.2.2.4.1* Ignoring x_8 and x_3

From analysing Figure 6-2-25, it is clear that the scale of variable x_8 is extraordinary smaller than others. And it is also obvious that variable x_3 is much smaller in comparison with others (excepted x_8), so x_8 and x_3 can be ignored in the additive model.

To the data set X4233f, we fitted a quadratic regression formula of the form (x_8 is ignored)

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon . \qquad (6\text{-}2\text{-}14)$$

The coefficients are c = 483.2 and b_i and a_i given by Table 6-2-7.

i	1	2	3	4	5	6	7
b_i	0.7512	0.02427	0.5345	244.9	287.7	2.048	-80.42
c_i	-0.000631	-0.000011	-0.000898	-90.88	-121.1	-0.01501	3.971

Table 6-2-7: Values of quadratic coefficients for X4233f

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}2\text{-}15)$$

it was found that the correlation between y and y_{pred} is 0.9825 (x_8 is ignored). A scatter plot of y against y_{pred} is shown in Figure 6-2-27.

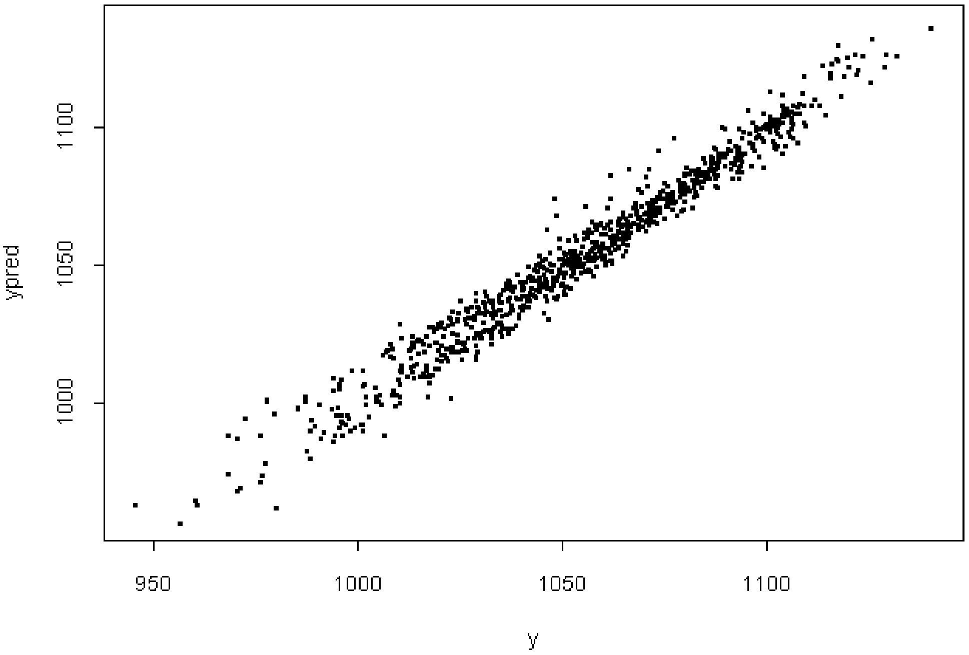


Figure 6-2-27: Scatter plot of y against y_{pred} (x_1 to x_7)

To the data set X4233f, we fitted a quadratic regression formula of the form (x_3 and x_8 are ignored)

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon . \qquad (6\text{-}2\text{-}16)$$

The coefficients c = 559.8, b_i and a_i are as in Table 6-2-8.

i	1	2	4	5	6	7
b_i	0.7526	0.02502	249.7	287.4	2.038	-80.79
a_i	-0.000633	-0.000011	-94.05	-120.9	-0.01484	4.421

Table 6-2-8: Values of quadratic regression coefficients for X4233f

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}2\text{-}17)$$

it was found that the correlation between y and y_{pred} is 0.9822. A scatter plot of y against y_{pred} for this case is shown in Figure 6-2-28.

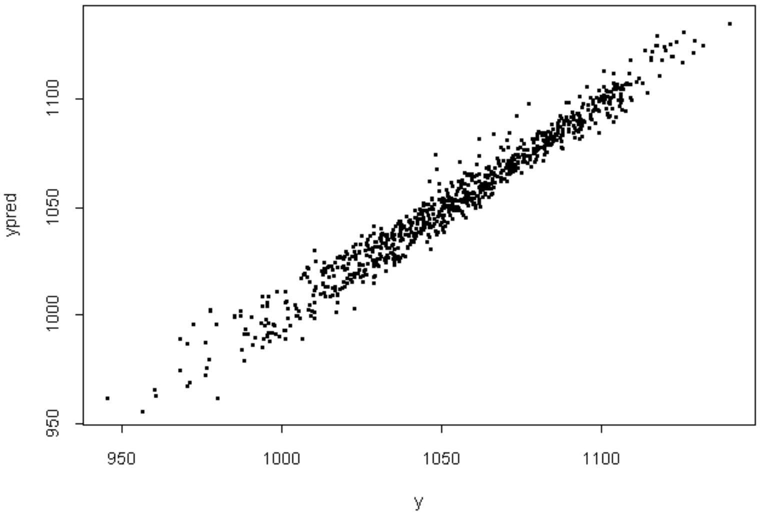


Figure 6-2-28: Scatter plot of y against y_{pred} for X4233f (x_3, x_8 are ignored)

Comparison of the above three values of correlation between y and y_{pred}: 0.9826, 0.9825 (x_8 is ignored) and 0.9822 (x_8 and x_3 are ignored) leads us to the conclusion: input parameter x_8 and x_3 can be ignored in the additive model in X4233f.

6.2.4.2 AVAS regression analysis for X4232

It turns out that there are 289 observations satisfying the constraint on the fuel area factor $f_A < 0.45$. On applying the AVAS algorithm regression to the data set X4232 the correlation coefficient was 0.575. The plots of the transformed variables against original data of X4232 are shown in Figure 6-2-29.

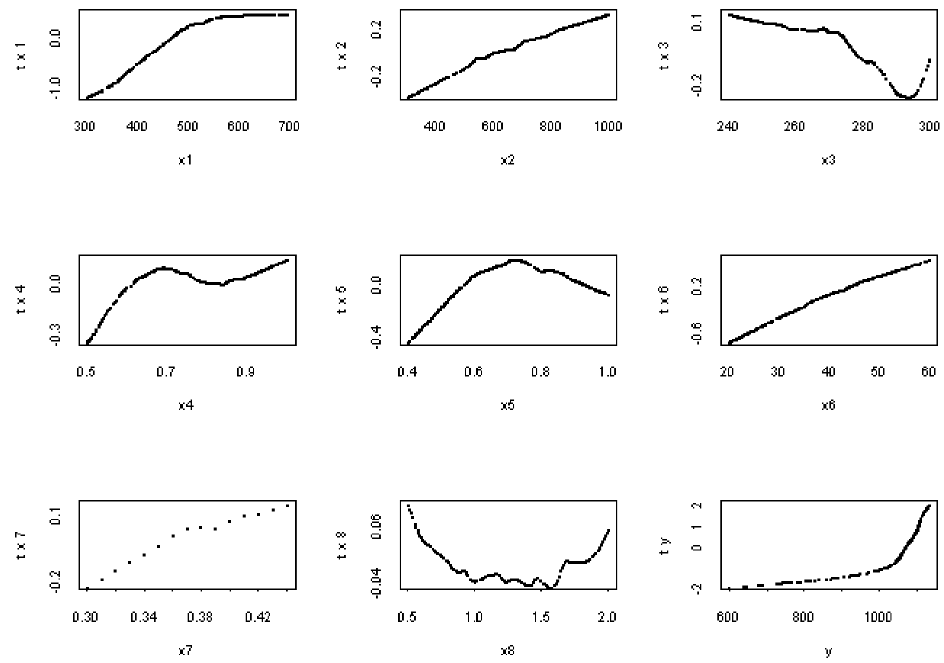


Figure 6-2-29: Plots of the transformed variables against the original data X4232

From the plot of variable x_5, which is the window width factor f_W, it is clear that there is a change of behaviour at a window width factor of 0.72. This point separated the data X4232 into two sub-sub-ranges: when $f_W < 0.72$ we have X42322 with 162 observations; when $f_W > 0.72$ we have X42323 with 127 observations.

6.2.4.2.1 AVAS regression analysis for X42322

After deleting outliers (149, 138, 98, 13, 12, 89, 160, 141, 137) from X42322 we are left with 153 observations, called X42322f. By using AVAS algorithm on data set X42322f, the correlation coefficient achieved is 0.9812. The plots of the transformed variables against original data are shown in Figure 6-2-30. It turns out that x_8 can be ignored since the transformed scale range is very small compared with the others.

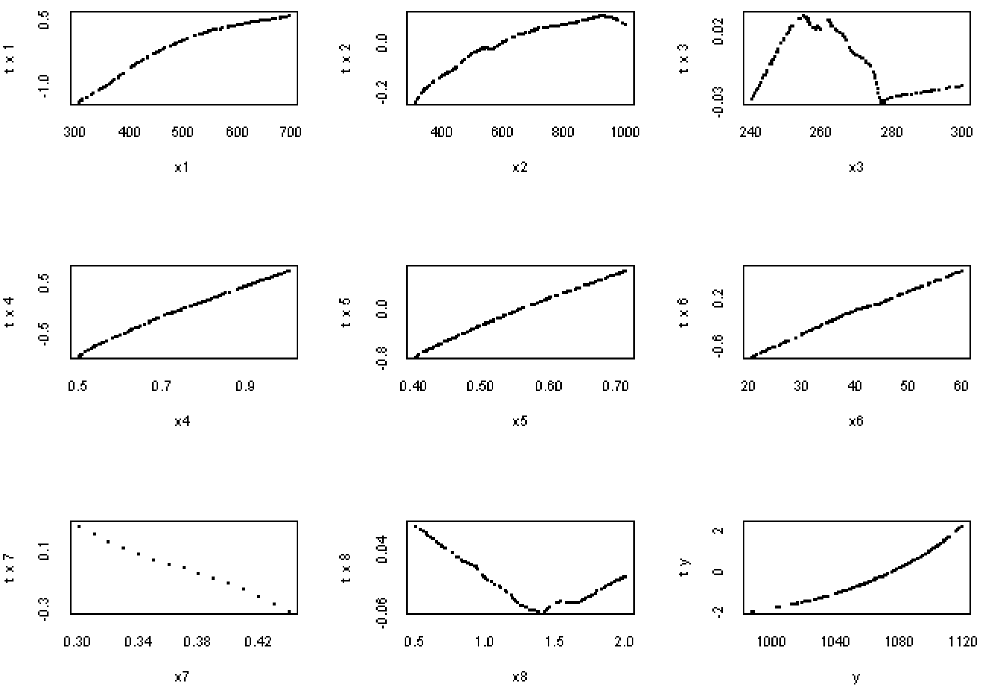


Figure 6-2-30: Plots of the transformed variables against the original data (X42322f)

To the 153 data points, we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i \ + c + \varepsilon \text{, } i = 1 \text{ to } 7. \qquad (6\text{-}2\text{-}18)$$

The coefficient c was 633.3 and b_i and a_I were as in Table 6-2-9.

i	1	2	3	4	5	6	7
b_i	0.8127	0.03726	-0.7869	247.7	228.1	2.590	120.6
a_i	-0.00066	-0.00002	0.00139	-105.2	-98.25	-0.01952	-297.0

Table 6-2-9: Values of quadratic coefficients for X42322f (x_8 are ignored)

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i \ + c \qquad (6\text{-}2\text{-}19)$$

it is found that the correlation between y and y_{pred} is 0.990.

A scatter plot of y against y_{pred} is shown in Figure 6-2-31.

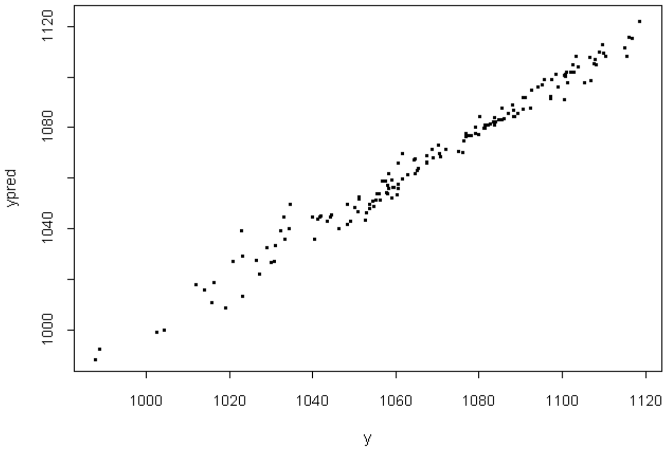


Figure 6-2-31: Scatter plot of y against y_{pred} for X42322f

6.2.4.2.2 AVAS regression analysis for X42323

There are 127 observations satisfying constraints: L < 700 cm, R_f > 0.5 m/sec, f_A < 0.45 and $f_H \geq 0.72$. By applying AVAS analysis to this data set X42323, the correlation coefficient is now 0.827. The plots of the transformed variables against original data are shown in Figure 6-2-32. It turns out that x_8 can be ignored since the transformed scale range is very small compared with the others.

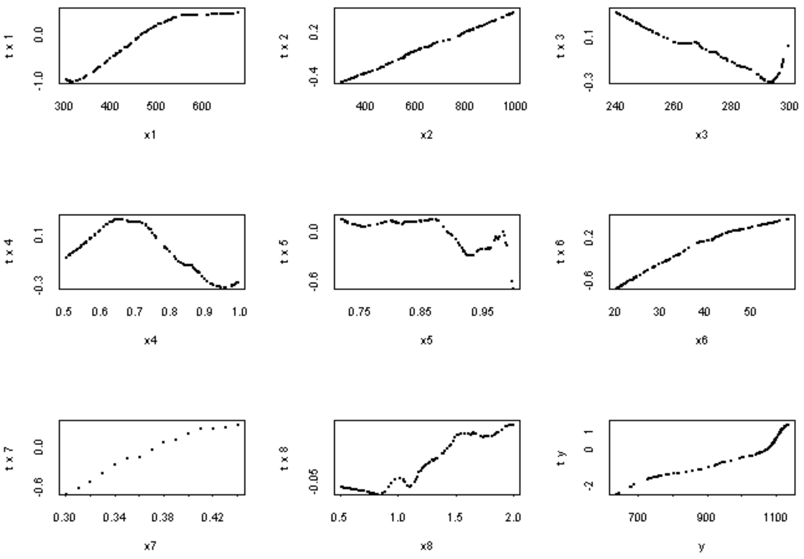


Figure 6-2-32: Plots of transformed variables against original data of X42323

To the 127 data points we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon . \tag{6-2-20}$$

The coefficient *c* was -4089.4 and coefficients b_i and a_I were as in Table 6-2-10.

i	1	2	3	4	5	6	7
b_i	3.301	0.3134	11.04	654.4	2471.1	10.93	6603.7
a_i	-0.00276	-0.00015	-0.02328	-642.8	-1618.0	-0.09240	-7215.0

Table 6-2-10: Values of quadratic regression coefficients for X42323 (x_8 is ignored)

Letting

$$y_t = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \tag{6-2-21}$$

it was found that the correlation between *y* and y_t was 0.8316.

The second step in the fitting is to improve the fit of y_t to *y* by using a cubic regression formula of the form

$$y = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 + \varepsilon . \tag{6-2-22}$$

The coefficients turned out to be:

C_0 = 5042.0; C_I = -18.21; C_2 = 0.02324; C_3 = -9.026e-006.

Letting

$$y_{predf} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 \tag{6-2-23}$$

the correlation between *y* and y_{predf} is now 0.9119. And a scatter plot of *y* against y_{predf} is shown in Figure 6-2-33.

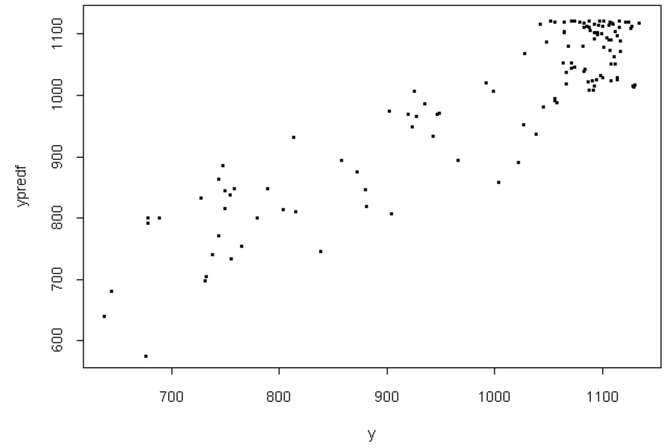


Figure 6-2-33: Scatter plot of *y* against y_{predf} of X42323

6.3 AVAS regression analysis for DCWO scenario

Applying the AVAS algorithm to the scenario the scatter plot of the predicted transformed maximum temperature against the transformed maximum temperature is shown in Figure 6-3-1. The correlation coefficient was 0.591.

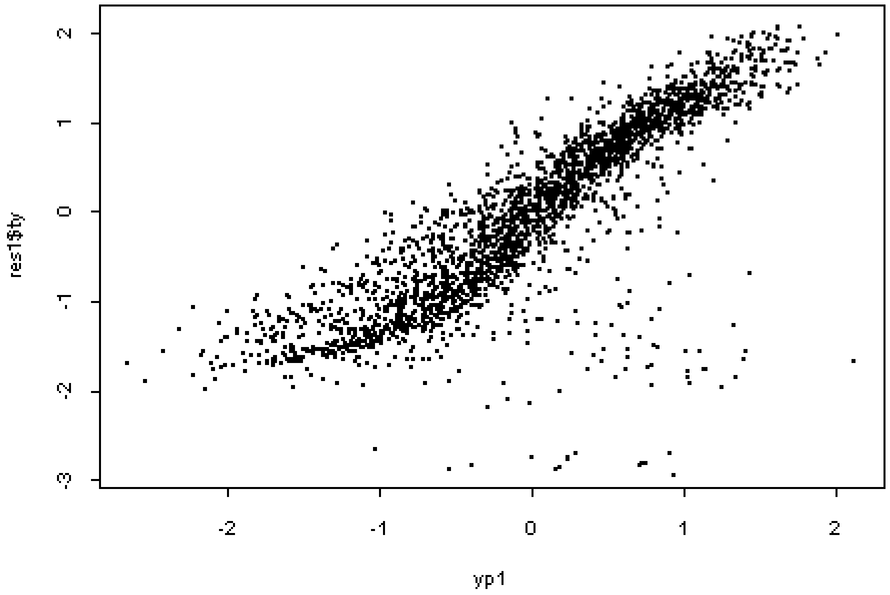


Figure 6-3-1: The predicted values against the transformed maximum temperature

Plots of the transformed variables against the original data in DCWO scenario were shown in Figure 6-3-2.

Figure 6-3-2 leads to the following conclusions:

1. From the plot of variable x_1, which is the length of the room L, it is clear that there is a change of behaviour at a width of 600 cm.
2. It is also clear that for variable x_8, which is the flame spread rate R_f, there is a change of behaviour at 0.45.

Therefore, we have obtained four sub-range data sets as follows:

X133: $L > 600$ cm, $R_f \geq 0.45$ m/sec; X132: $L > 600$ cm, $R_f < 0.45$ m/sec;

X123: $L < 600$ cm, $R_f \geq 0.45$ m/sec; X122: $L < 600$ cm, $R_f < 0.45$ m/sec.

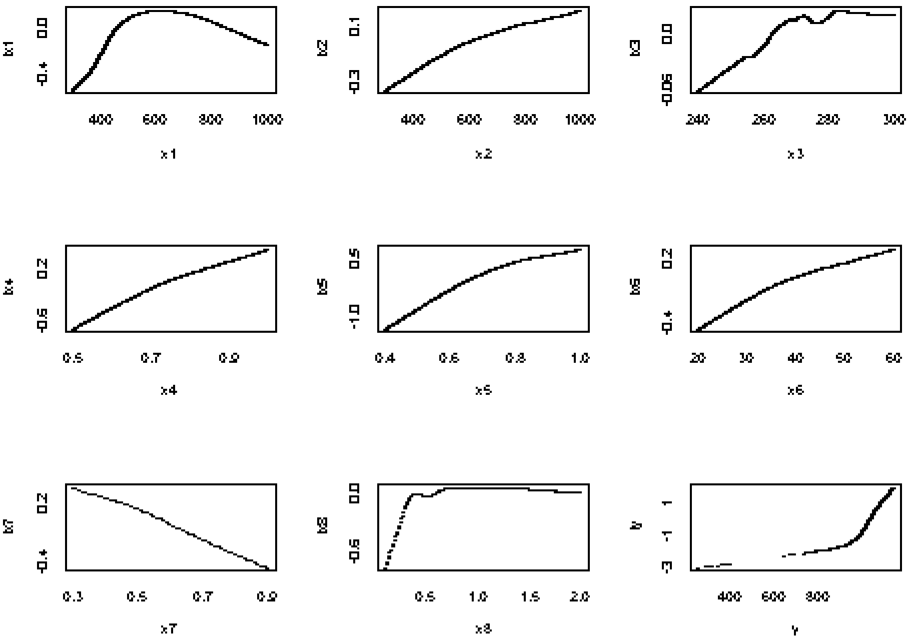


Figure 6-3-2 Plots of the transformed variables against the original data

6.3.1 AVAS regression analysis for X133:

It turns out that there are 1181 observations satisfying the constraints of length of room L > 600 cm, flame spread rate $R_f \geq 0.45$ m/sec. We apply the AVAS regression algorithm to this data set and the correlation coefficient obtained is 0.9452. The scatter plot of the predicted values of maximum temperature against the transformed values is shown in Figure 6-3-3.

From Figure 6-3-3, it is clear that there are 5 outliers. Deleting them left us with 1176 data points, which we named X133f. By applying AVAS to data set X133f the correlation coefficient achieved is 0.9840. The predicted values of maximum temperature against the transformed values are shown in Figure 6-3-4. And the plots of the transformed variables against the original data for X133f are shown in Figure 6-3-5.

It turns out that x_8 can be ignored since the transformed scale range is very small compared with the others.

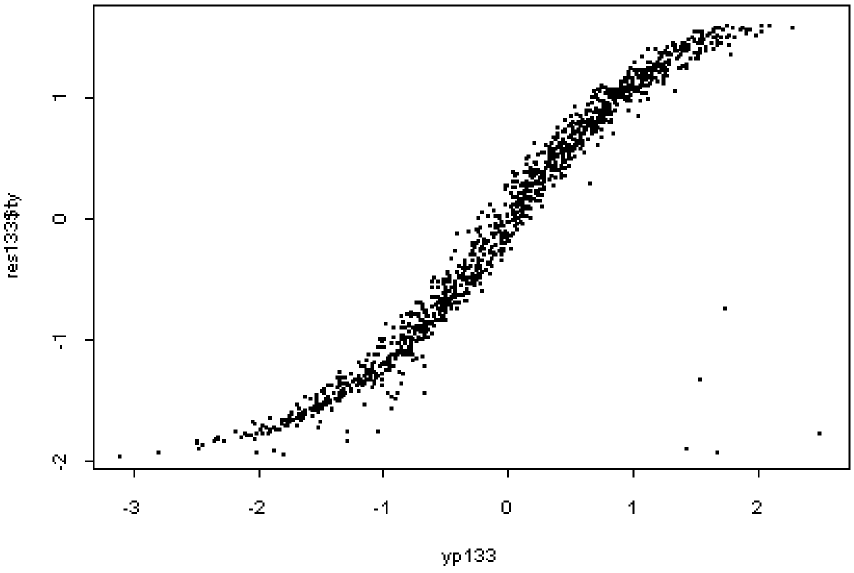


Figure 6-3-3: The predicted values of maximum temperature against the transformed values

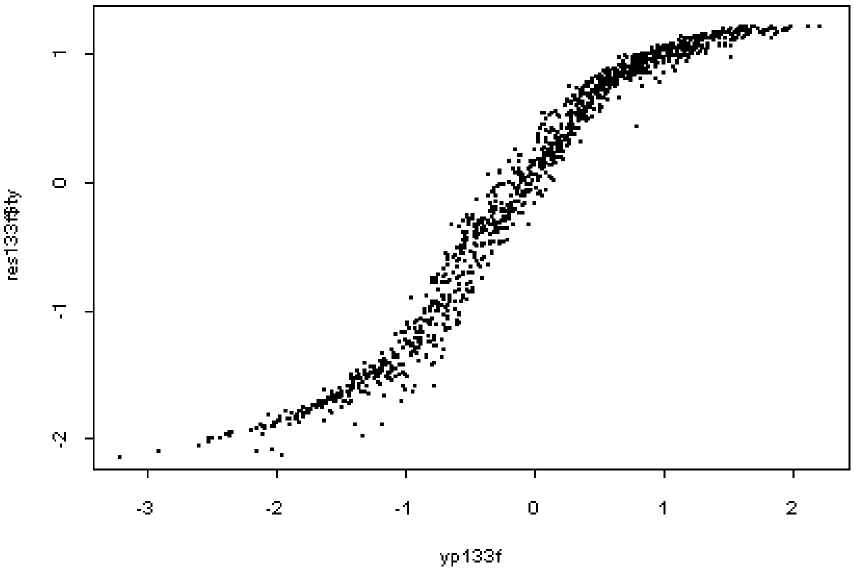


Figure 6-3-4: The predicted values of maximum temperature against the transformed values (X133f)

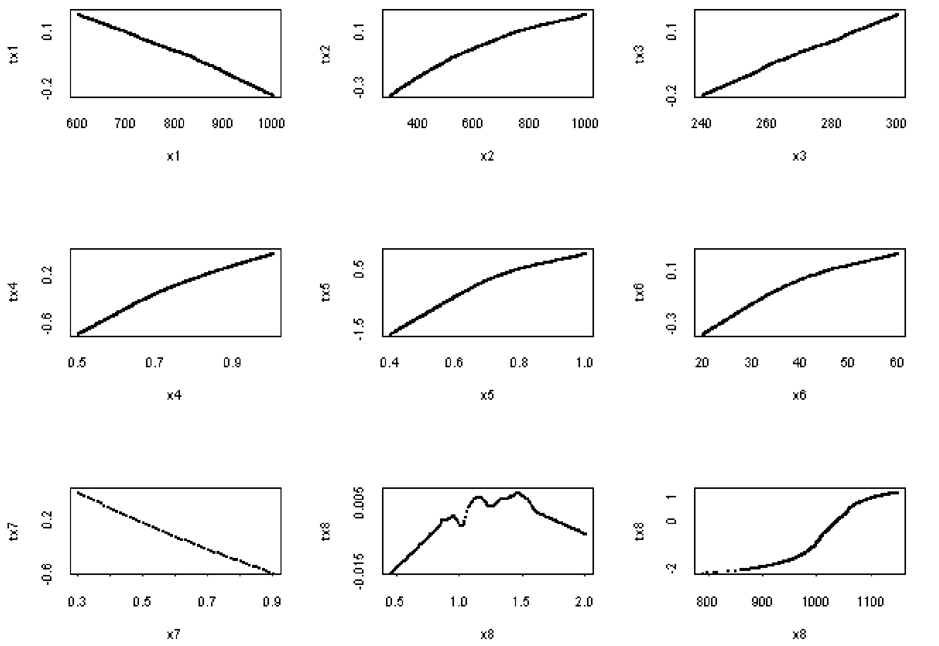


Figure 6-3-5: Plots of the transformed variables against the original data (for X133f)

To the data set X133f, we fitted a quadratic regression formula of the form

$$y = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 + \varepsilon \text{ , } I = 1,2,3,4,5,6,7. \qquad (6\text{-}3\text{-}1)$$

The intercept *c* was 177.5. And coefficients b_i and a_i were as in Table 6-3-2.

i	1	2	3	4	5	6	7
b_i	0.1544	0.1597	0.9302	423.9	804.6	2.832	-178.2
a_i	-0.0001	0.0001	-0.0009	-166.7	-373.8	-0.0228	44.14

Table 6-3-2: Values of quadratic regression coefficients for X133f

Letting

$$y_{pred} = c + \sum_{i=I} b_i x_i + \sum_{i=I} a_i x_i^2 \qquad (6\text{-}3\text{-}2)$$

it was found that the correlation between y and y_{pred} is 0.9859.

A scatter plot of y against y_{pred} is shown in Figure 6-3- 6.

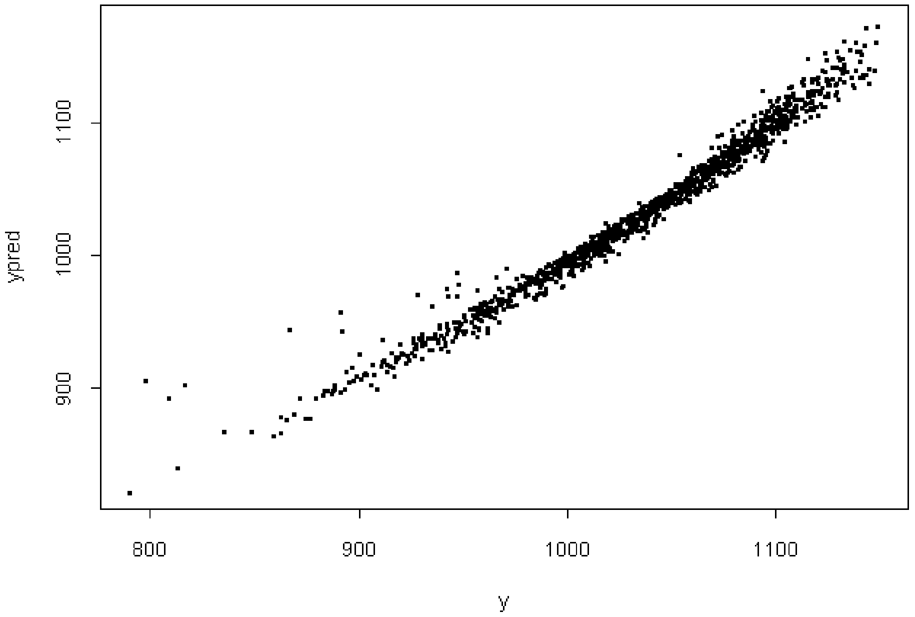


Figure 6-3-6: Scatter plot of y against y_{predf} for DCWO scenario

6.3.2 AVAS regression analysis for X132

There were 256 data points satisfying the constraints $L > 600$ cm and $R_f < 0.45$ m/sec. We apply the AVAS regression analysis to the X132 data set. The scatter plot of transformed maximum temperature against predicted values of X132 is shown in Figure 6-3-7.

It is clear that there are 14 outlier data points (3, 6, 7, 17, 19, 20, 21, 22, 24, 25, 27, 37, 56, 62). Deleting them left us with 242 observations, which are named X132f.

By applying the AVAS algorithm to X132f the correlation coefficient achieved is increased from 0.508 to 0.906. The plots of transformed variables against original data of X132f are shown in Figure 6-3-8.

To the data set X132f, we fitted a quadratic regression formula of the form

$$y = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 + \varepsilon \,, I = 1 \text{ to } 8. \qquad (6\text{-}3\text{-}3)$$

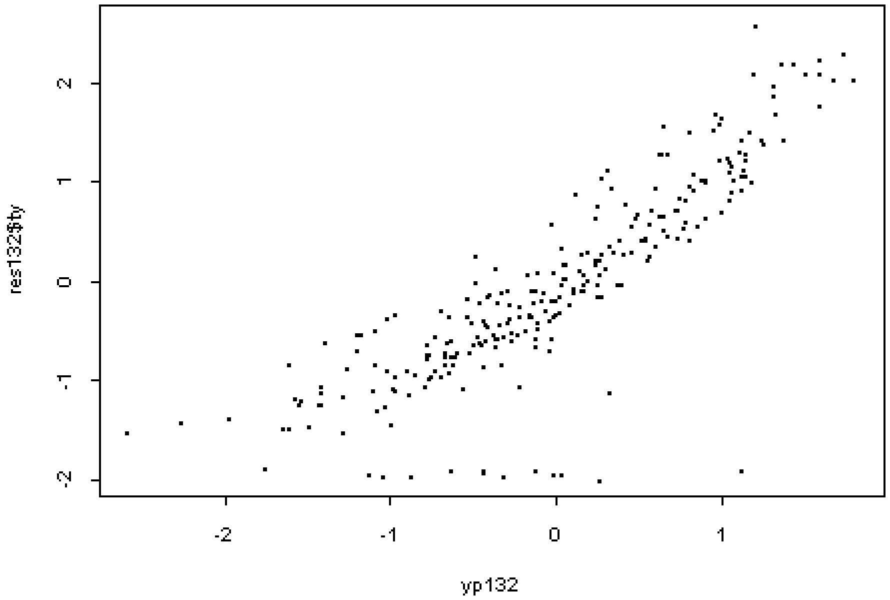


Figure 6-3-7: Scatter plot of transformed output against predicted values of X132

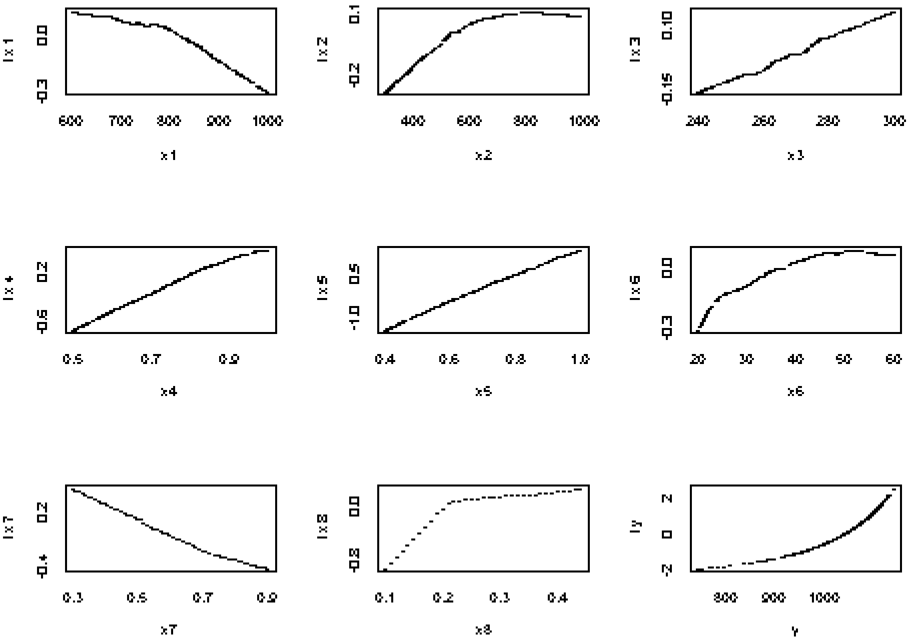


Figure 6-3-8: Plots of transformed variables against original data (X132f)

The intercept c was 403.3, and coefficients b_i and a_i were as in Table 6-3-3.

i	1	2	3	4	5	6	7	8
b_i	0.1423	0.1480	-2.971	483.8	821.0	4.074	-151.2	1628.7
a_i	-0.00015	-0.00011	0.00641	-178.1	-366.5	-0.04616	42.03	-2504.5

Table 6-3-3: Values of quadratic regression coefficients for X132f

Letting

$$y_t = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 \qquad (6\text{-}3\text{-}4)$$

it was found that the correlation between y and y_t is 0.921.

A scatter plot of y against y_t (= y_{pred}) is shown in Figure 6-3- 9

To improve the fit of y_t to y we use a cubic regression formula of the form

$$y_{predf} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 + \varepsilon^* . \qquad (6\text{-}3\text{-}5)$$

The coefficients are:

C_0 = -1231.7, C_1= 2.976, C_2 = -0.0002542, C_3 = -4.807e-007.

Letting

$$y_{predf} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 \qquad (6\text{-}3\text{-}6)$$

the correlation achieved between y and y_{predf} is now 0.932.

And the scatter plot of y against y_{predf} is now shown in 6-3-10.

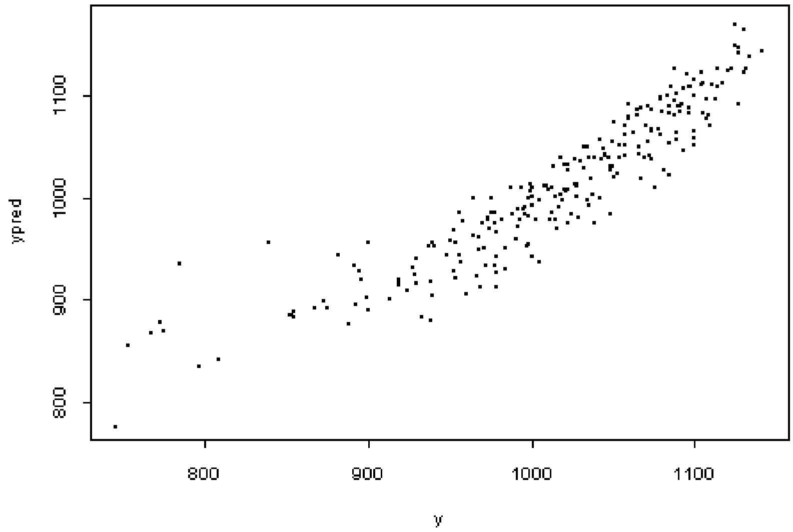


Figure 6-3-9: Scatter plot of y against y_{pred} (quadratic fitted for X132f)

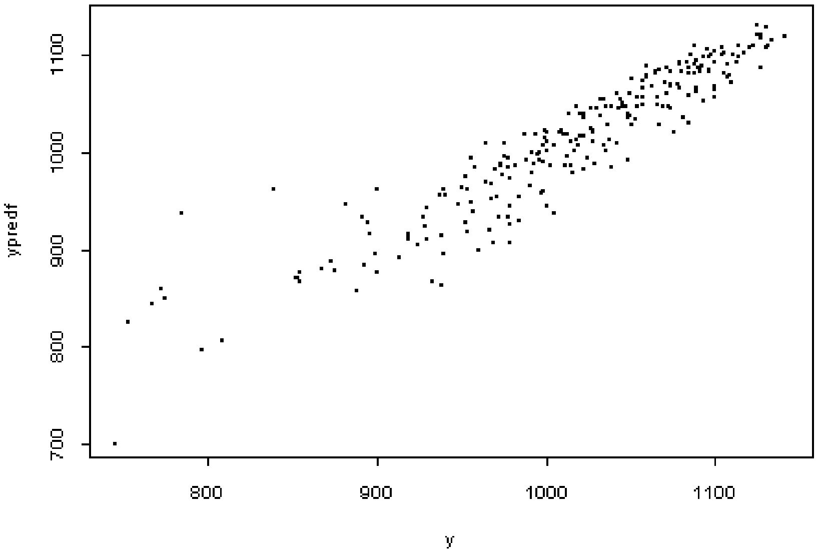


Figure 6-3-10: Scatter plot of y against y_{predf} (cubic fitted)

6.3.3 AVAS regression analysis for X123:

There were 855 data points satisfying the constraints $L < 600$ cm and $R_f \geq 0.45$ m/sec. By using the AVAS regression analysis on X132 data set, the correlation coefficient was 0.702. The plots of transformed variables against original data of X132 are shown in Figure 6-3-11.

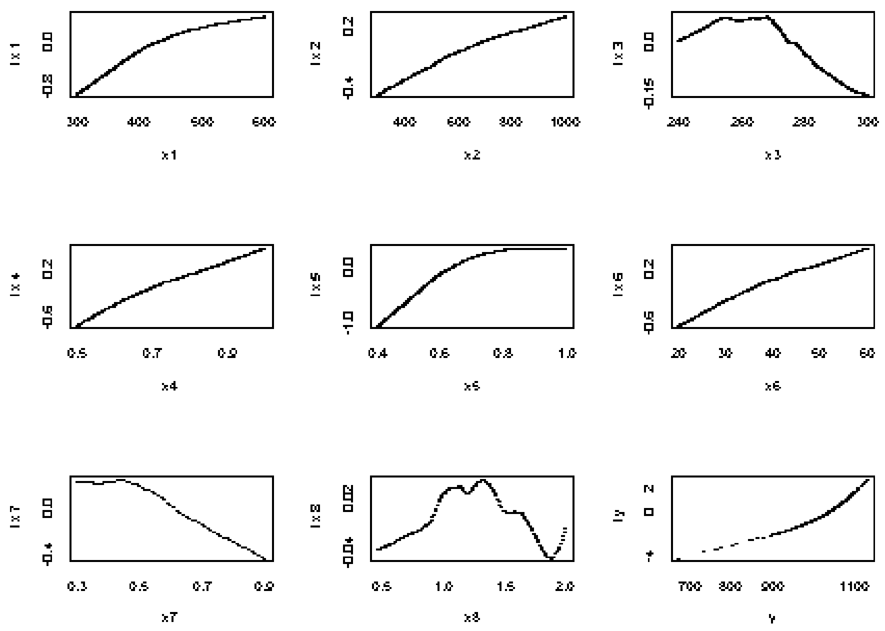


Figure 6-3-11: Plots of transformed variables against original data for X123

It is clear that there are different modes of fire growth for room height H_r, which is x_3, above and below x_3 = 270 cm. When H_r < 270 cm, we have a new data set X1232 with 439 observations; When H_r > 270, we have another data set X1233 with 416 observations.

6.3.3.1 AVAS regression analysis for X1232

We apply the AVAS algorithm regression analysis to data set X1232. The correlation coefficient is 0.861. A plot of transformed maximum temperature against predicted values in AVAS is shown in Figure 6-3-12. It was obvious that there were six outliers (162, 173, 69, 8, 104, 354). Deleting them left us with data set X1232f that has 434 observations.

By applying the AVAS algorithm to X1232f, the correlation coefficient achieved is 0.9425. Plots of the transformed variables against original data of X1232f are shown in Figure 6-3-13.

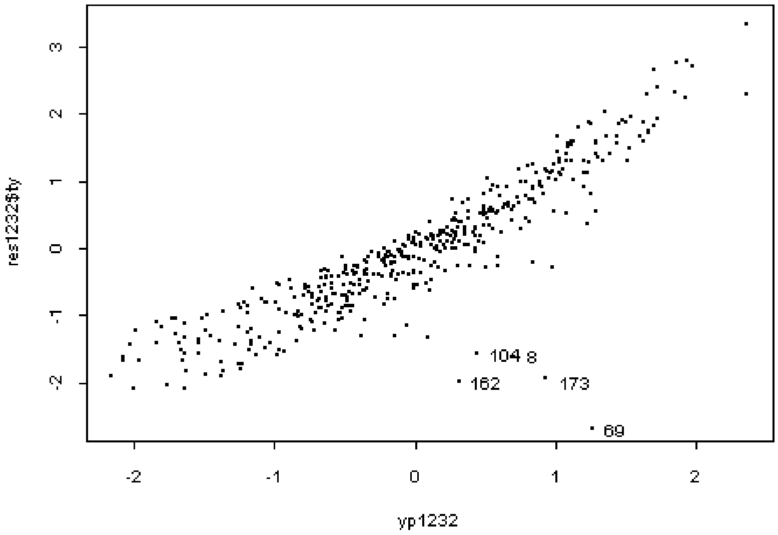


Figure 6-3-12: Scatter plot of transformed output against predicted values of AVAS

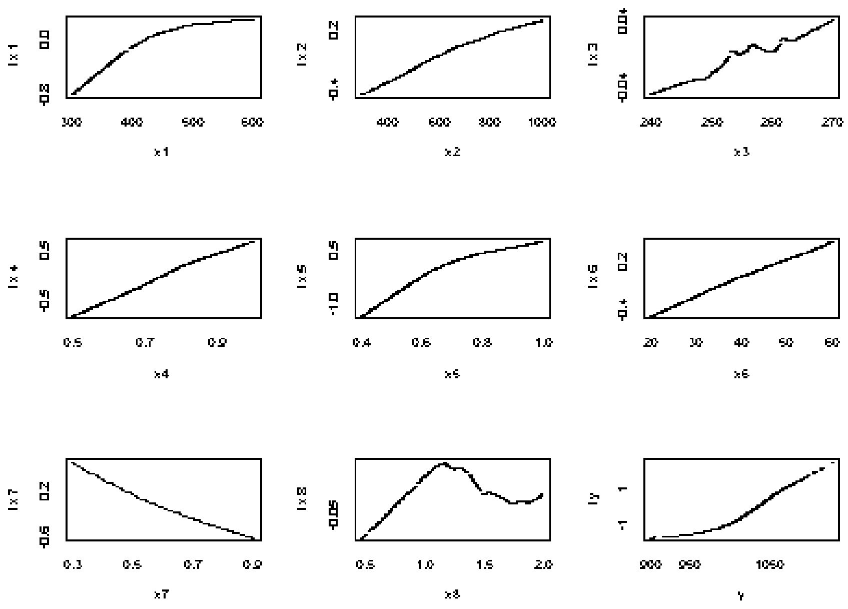


Figure 6-3-13: Plots of transformed variables against original data

To the data set X1232f, we fitted a quadratic regression formula of the form

$$y = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 + \varepsilon \text{ , } I = 1 \text{ to } 8. \tag{6-3-7}$$

The intercept $c = 351.9$, the coefficients b_i and a_i were as in Table 6-3-4.

i	1	2	3	4	5	6	7	8
b_i	0.8503	0.1017	-0.2871	338.3	658.1	2.013	-147.3	16.84
a_i	-0.00082	-0.00004	0.00082	-134.1	-362.5	-0.01361	45.58	-7.510

Table 6-3-4: Values of quadratic regression coefficients for X1232f

Letting

$$y_t = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 \qquad (6\text{-}3\text{-}8)$$

it is found that the correlation between y and y_t ($= y_{pred}$) is 0.950.

A scatter plot of y against y_{pred} is shown in Figure 6-3-14.

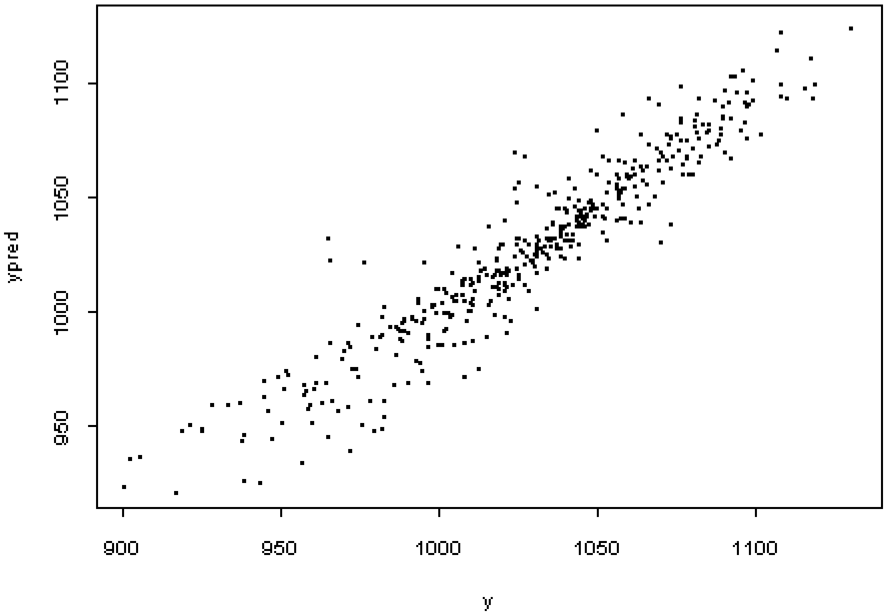


Figure 6-3-14: Scatter plot of y against y_{pred} for X1232f

6.3.3.2 AVAS regression analysis for X1233

By using the AVAS algorithm regression analysis on data X1233, the correlation coefficient was 0.650. The transformed maximum temperature against predicted values in AVAS is shown in Figure 6-3-15. Deleting 20 outliers left us with X1233ff which contains 395 data points. We apply the AVAS to X1233ff. The correlation coefficient achieved is 0.922. Plots of the transformed variables against original data of X1233ff are shown in Figure 6-3-16.

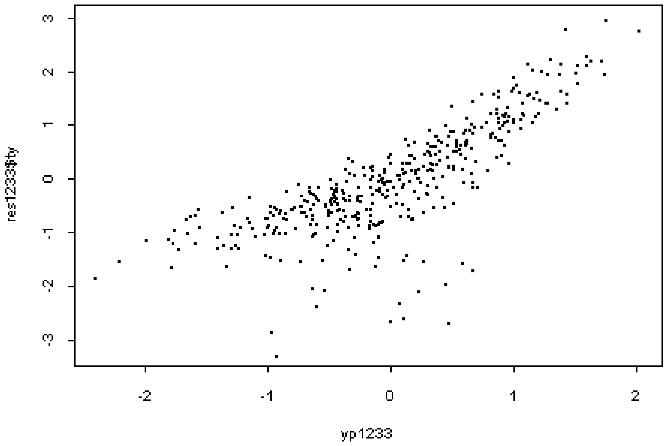


Figure 6-3-15: Transformed output against predicted values in AVAS

To the data set X1233ff, we fitted a quadratic regression formula of the form

$$y = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 + \varepsilon\,,\; I = 1,2,4,5,6,7. \qquad (6\text{-}3\text{-}9)$$

The intercept *c* was 194.5, the coefficients b_i and c_i were as in Table 6-3-5.

i	1	2	4	5	6	7
b_i	1.214	0.1498	402.8	685.5	1.948	-108.9
a_i	-0.001130	-0.000084	-176.9	-407.3	-0.01104	18.57

Table 6-3-5: Values of quadratic fit coefficients for X1233ff

Letting

$$y_{pred} = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 \qquad (6\text{-}3\text{-}10)$$

it is found that the correlation between y and y_{pred} (quadratic fit) is 0.932.

A scatter plot of y against y_{pred} (quadratic for X1233ff, x_3, x_8 are ignored because their effect is very small) is shown in Figure 6-3-17.

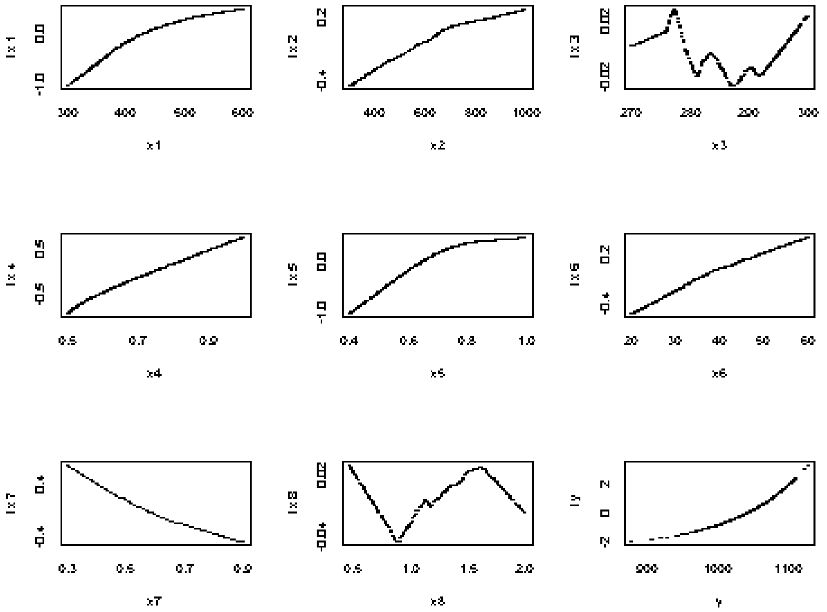


Figure 6-3-16: Plots of transformed variables against original data of X1233ff

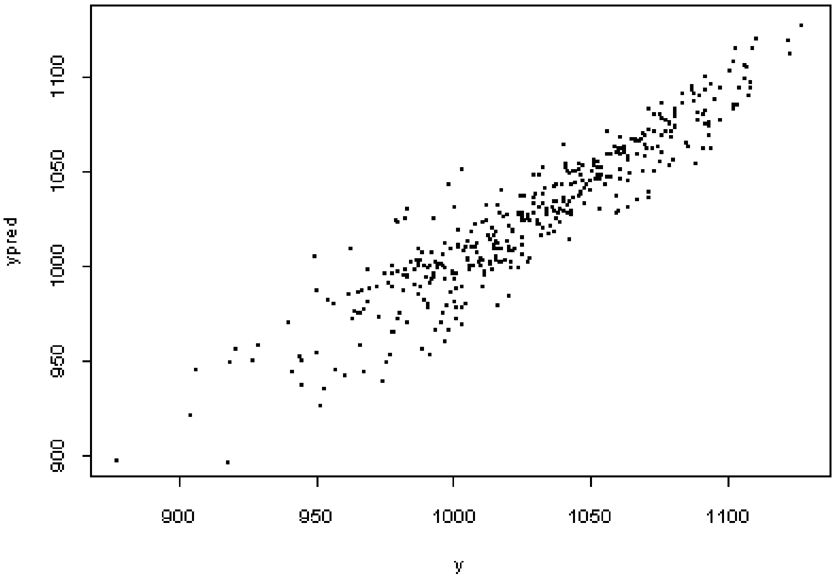


Figure 6-3-17: Scatter plot of y against y_{pred}

6.3.4 AVAS regression analysis for X122

There were 208 data sets satisfying the constraints $L < 600$ cm, $R_f < 0.45$ m/sec. By using AVAS regression analysis on X122, the correlation coefficient was 0.498. A scatter plot of transformed output against predicted values in AVAS is shown in Figure 6-3-18. From Figure 6-3-18, it is clear that there were ten outliers (5, 6, 8, 10,14, 40,48,74, 113,118). Deleting them left us with 198 observations that we named X122ff. We apply the AVAS algorithm to X122ff. The correlation coefficient achieved is 0.9309. Plots of the transformed variables against original data X122ff are shown in Figure 6-3-19.

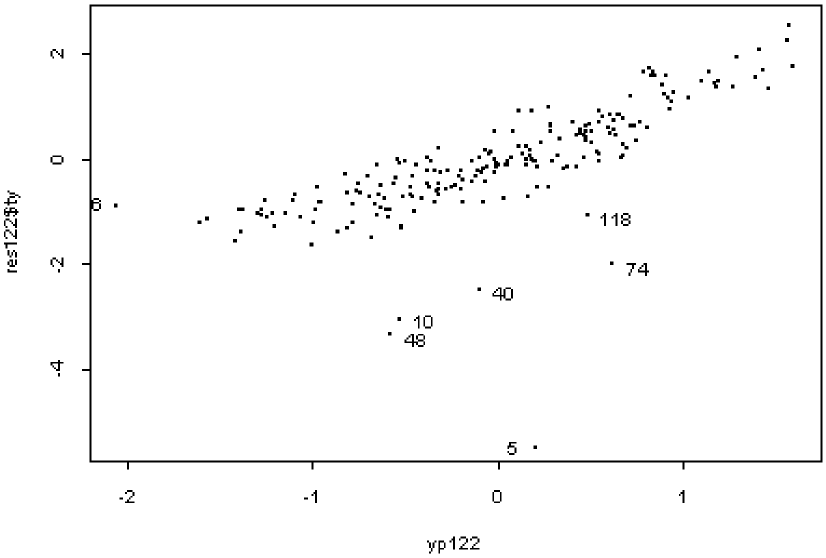


Figure 6-3-18: Scatter plot AVAS transformed output against predicted values

To the data set X122ff, we fitted a quadratic regression formula,

$$y = c + \sum_{i\in I} b_i x_i + \sum_{i\in I} a_i x_i^2 + \varepsilon \text{, I = 1,2,3,4,5,6,7,8.} \qquad (6\text{-}3\text{-}11)$$

The coefficient c was 45.17, coefficients b_i and a_i were as in Table 6-3-6.

i	1	2	3	4	5	6	7	8
b_i	0.7308	0.1041	1.881	346.8	683.0	1.448	-72.46	382.4
a_i	-0.00064	-0.00006	-0.00346	-149.9	-391.2	-0.00845	-5.835	-613.6

Table 6-3-6: Values of quadratic coefficients for X122ff (198 obs)

Letting

$$y_{pred} = c + \sum_{i=I} b_i x_i + \sum_{i=I} a_i x_i^2 \qquad (6\text{-}3\text{-}12)$$

it is found that the correlation between y and y_{pred} is 0.9287.

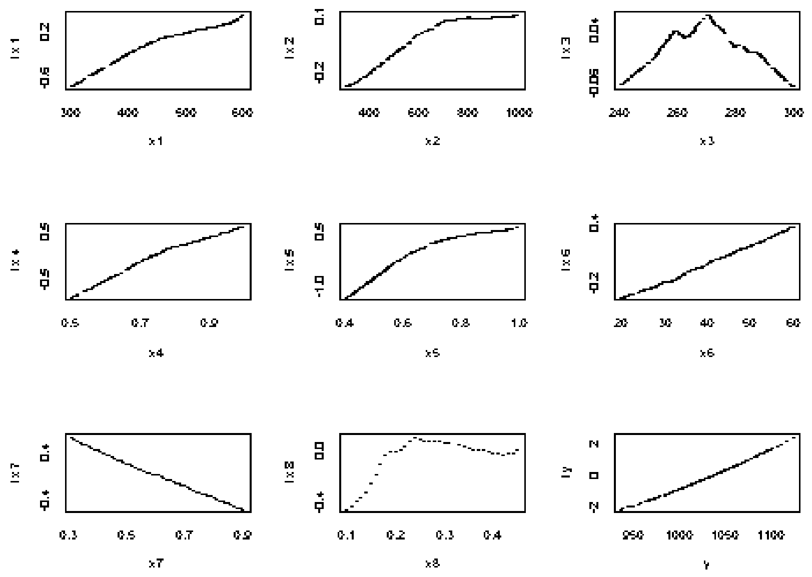


Figure 6-3-19: Transformed variables against original data of X122ff

From the Figure 6-3-19, it is clear that fire changed its growth behavior at x_3 = 270 cm, (x_3 is the height of room H_r). This separates the data set X122 into two new sub-sub-range data sets: X1222 when H_r < 270 cm, and X1223 when H_r > 270 cm.

6.3.4.1 AVAS regression analysis for X1222

We apply the AVAS algorithm on data X1222. Plots of transformed variable against original data are shown in Figure 6-3-20.

To data X1222, we fitted a quadratic regression formula

$$y = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 + \varepsilon \qquad (6\text{-}3\text{-}13)$$

The coefficient c_i was -335.4, coefficients b_i and a_i were in Table 6-3-7:

i	1	2	3	4	5	6	7	8
b_I	0.5945	0.04071	4.170	561.9	683.5	2.468	-65.22	559.0
a_I	-0.00056	-0.00001	-0.00771	-292.4	-367.2	-0.01889	-23.11	-881.9

Table 6-3-7: Values of quadratic fit coefficients for X1222 (111 obs)

Letting

$$y_{pred} = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 \qquad (6\text{-}3\text{-}14)$$

it is found that the correlation between y and y_{pred} is 0.9385.

And a scatter plot of y against y_{pred} is shown in Figure 6-3-21.

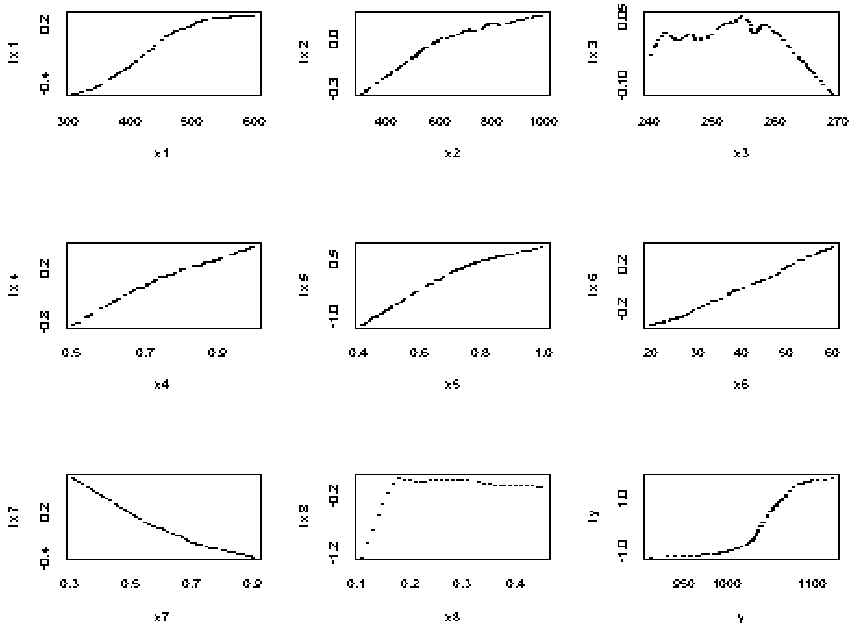


Figure 6-3-20: Plots transformed variables against original data for X1222

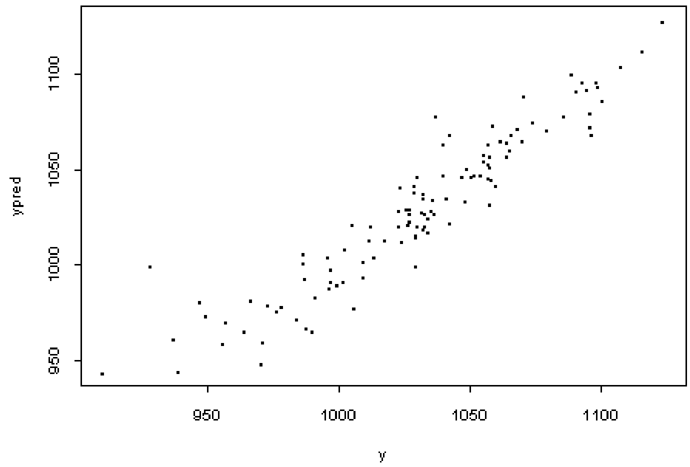


Figure 6-3-21: Scatter plot of y against y_{pred} of X1222

6.3.4.2 AVAS regression analysis for X1223

We apply the AVAS algorithm on data X1223. Plots of transformed variables against original data are shown in Figure 6-3-22.

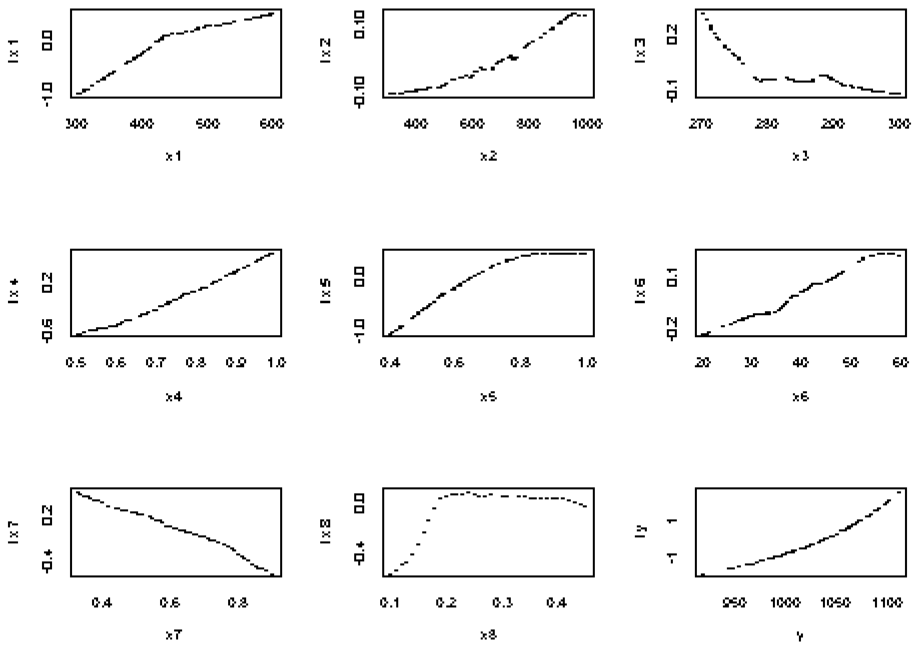


Figure 6-3-22: Plots transformed variables against original data for X1223

To data X1223, we fitted a quadratic regression formula

$$y = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 + \varepsilon \text{, I = 1 to 8.} \qquad (6\text{-}3\text{-}15)$$

The coefficients are c = 7923.7, b_i and a_i as in Table 6-3-8.

i	1	2	3	4	5	6	7	8
b_i	1.087	-0.00909	-53.70	98.00	820.3	1.340	80.41	546.2
a_i	-0.00099	0.00001	0.09388	0.3936	-504.2	-0.01083	-126.2	-840.6

Table 6-3-8: Values of quadratic fit coefficients for X1223

Letting

$$y_{pred} = c + \sum_{i \in I} b_i x_i + \sum_{i \in I} a_i x_i^2 \qquad (6\text{-}3\text{-}16)$$

it is found that the correlation between y and y_{pred} is 0.9403. A scatter plot of y against y_{pred} is shown in Figure 6-3-23.

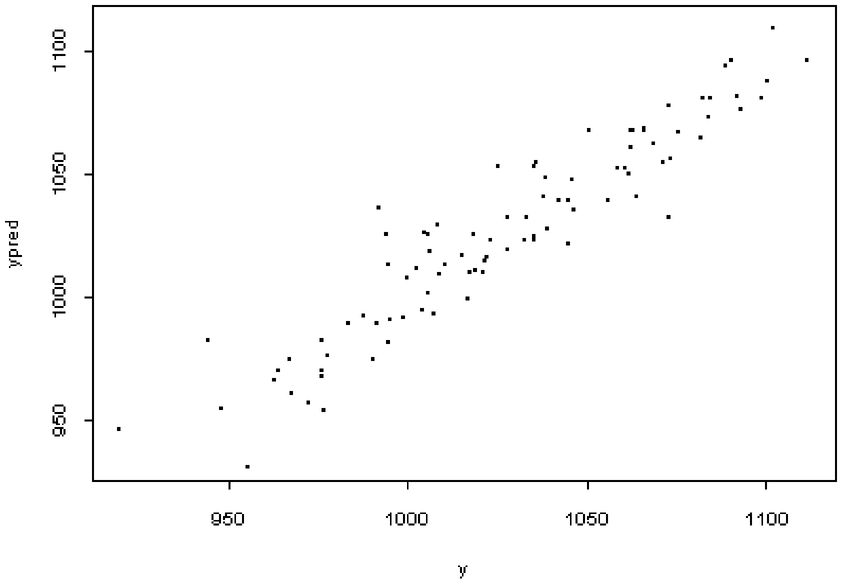


Figure 6-3-23: Scatter plot of y against y_{pred} (quadratic fit values) of X1223

6.4 AVAS Regression analysis for DCWC scenario

We apply the AVAS regression to the 2500 data sets of DCWC (denoted by X35) scenario. The scatter plot of transformed y against predicted values in AVAS is shown in Figure 6-4-1.

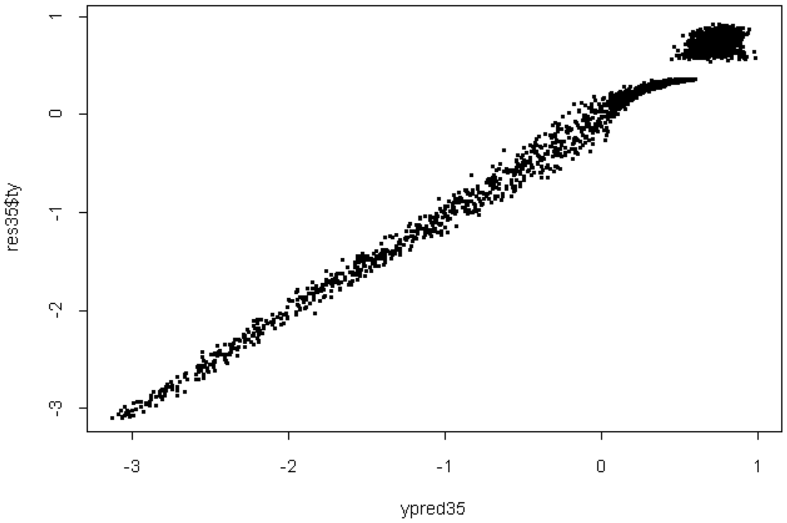


Figure 6-4-1: Scatter plot of transformed y (res35$ty) against y_{pred}

It is clear that some special things happened that caused the maximum temperature to change suddenly. Plots of the transformed variables against the original data X35 are shown in Figure 6-4-2.

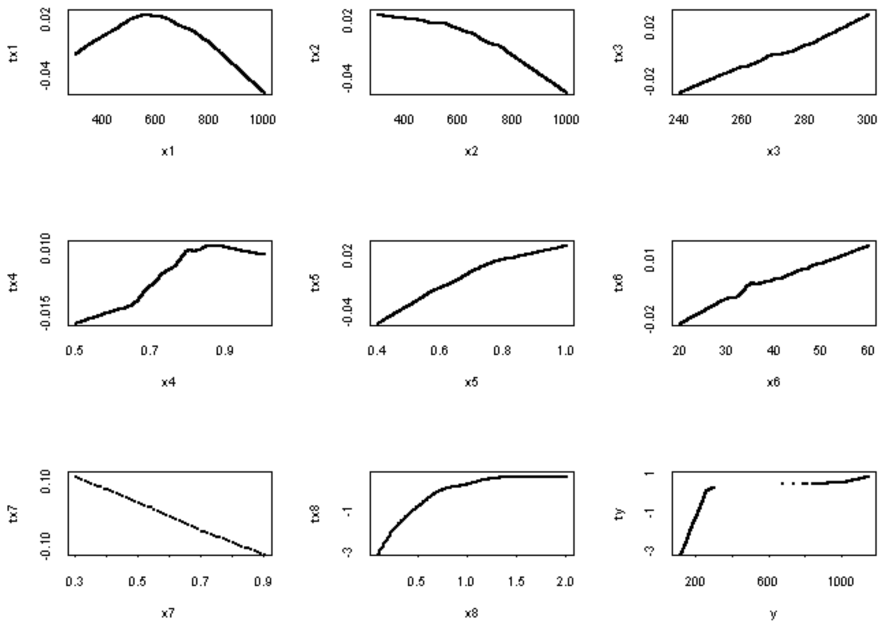


Figure 6-4-2: Plots of the transformed (AVAS) variables against the original data X35

By analyzing Figure 6-4-2. It is clear that x_8, which is the flame spread rate R_f , is the input that correlates best with the output, the maximum temperature, in this scenario. The correlation between x_8 and y (the output) is 0.8799. Further analysis shows that there are behavior changes at $x_8 = 0.96$ m/sec and at $x_8 = 1.32$ m/sec.

Therefore, we separated the data X35 into three new sub-range data sets:
X351 with 1166 observations, with $x_8 < 0.96$ m/sec;
X352 with 467 observations, with 0.96 m/sec $< x_8 <$1.32 m/sec;
X353 with 867 observations, with 1.32 m/sec $< x_8$.

6.4.1 AVAS regression analysis of X351

By using the AVAS regression analysis on data X351, a scatter plot of transformed y against y_{pred} in the AVAS is shown in Figure 6-4-3. And plots of the transformed variables against original data are shown in Figure 6-4-4. The correlation coefficient is 0.9929.

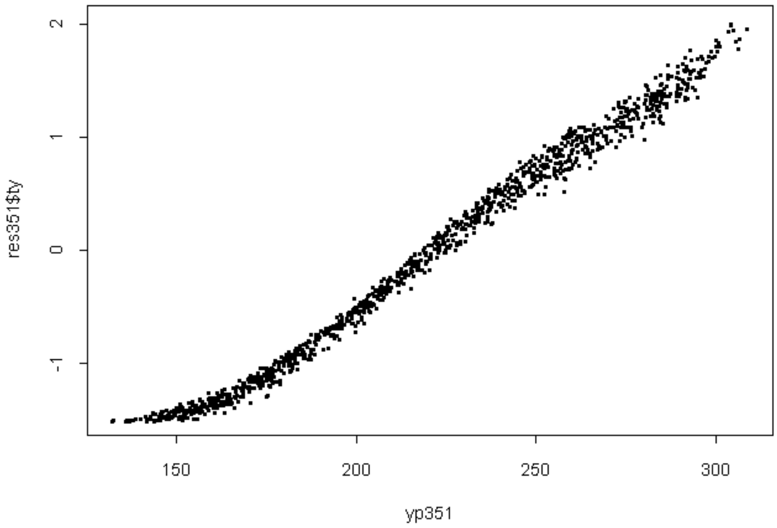


Figure 6-4-3: Scatter plot of transformed output against predicted values

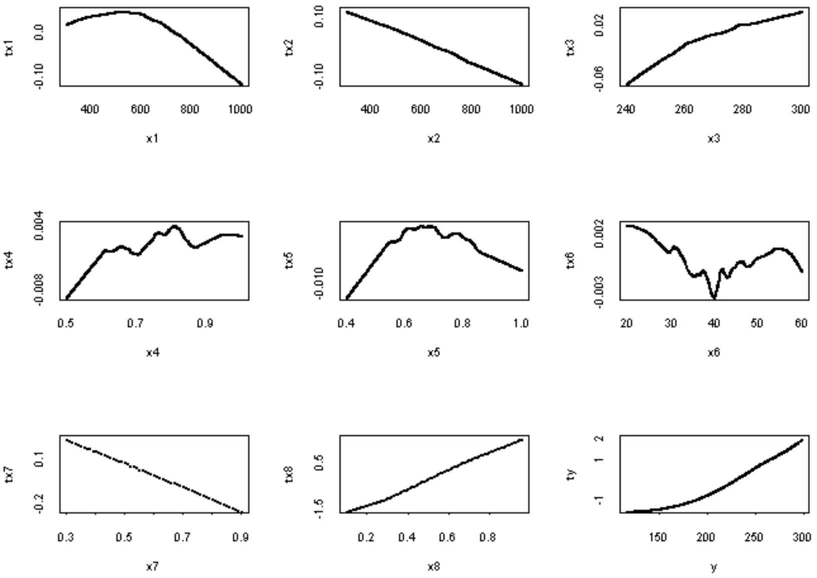


Figure 6-4-4: Plots of the transformed variables against original data of X351

From analyzing Figure 6-4-4, it is clear that the influence of input variables x_4, x_5 and x_6 on the output in X351 is not significant, so they can be ignored in the regression formula.

To the data X351 we fitted a formula of the form:

$$y = \sum_{i \in I} b_i x_i + c + \varepsilon . \qquad (6\text{-}4\text{-}1)$$

where I = 1,2,3,7,8.

The coefficients turned out to be:
c =140.0, b_1=-0.0091, b_2=-0.0162, b_3=-0.0762, b_7=-32.03, and b_8=177.9.

Setting

$$y_{pred} = \sum_{i \in I} b_i x_i + c \text{ , } I = 1,2,3,7,8 \qquad (6\text{-}4\text{-}2)$$

it was found that the correlation between y and y_{pred} is 0.9861. A scatter plot of y against y_{pred} is shown in Figure 6-4-5.

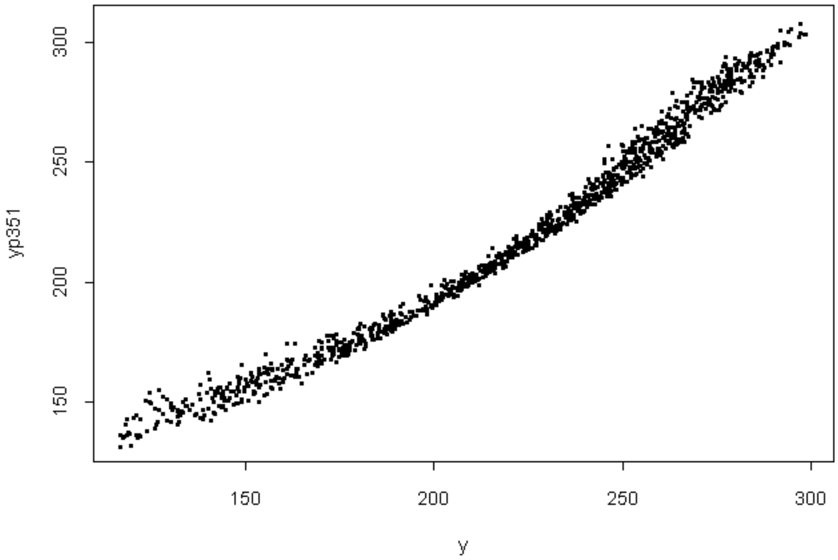


Figure 6-4-5: Scatter plot of y against y_{pred} (X351)

6.4.2 AVAS regression analysis for X353 (867 observations)

By using the AVAS regression on X353, the correlation coefficient was 0.8244. Plots of the transformed variables against original data are shown in Figure 6-4-6.

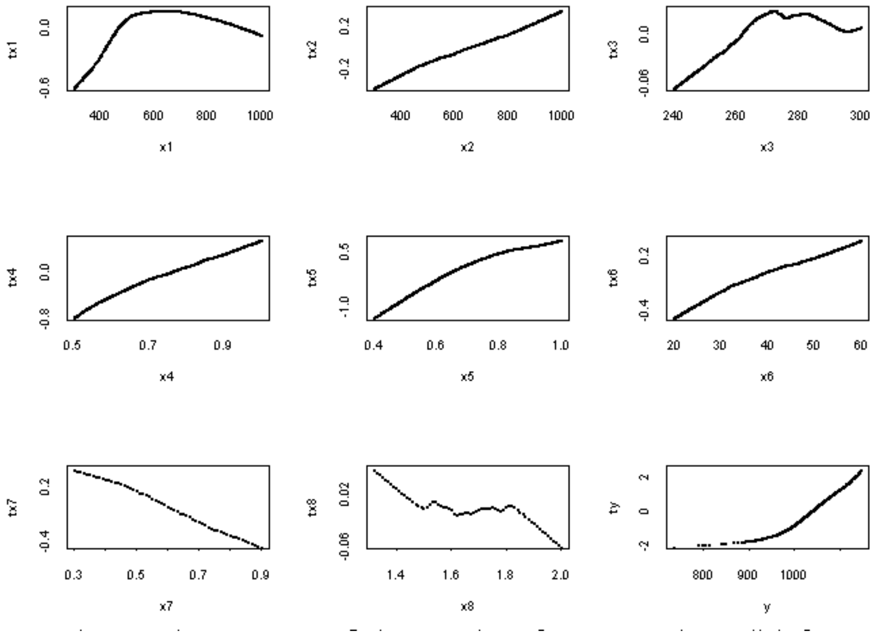


Figure 6-4-6: Plots of the transformed variables against the original data X353

From Figure 6-4-6, it is clear from the plot of x_1, which is the length of room L, that there is a behavior change at $x_1 = 600$ cm. Therefore, this point separates X353 into two new sub-sub-range data sets: X3532 when $x_1 < 600$ cm, and X3533 when $x_1 >$ 600 cm.

6.4.2.1 AVAS regression analysis for X3533

We use AVAS regression analysis on X3533 (488 points). The scatter plot of output against predicted values in AVAS is shown in Figure 6-4-7. Deleting three outliers left us with data X35333 (485 points). Plots of transformed variables against the original data are shown in Figure 6-4-8.

To the X35333 data set, we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon \,. \qquad (6\text{-}4\text{-}3)$$

The coefficient c was 284.4. Coefficients a_i and b_i were as in Table 6-4-1.

i	1	2	3	4	5	6	7	8
b_i	0.1765	0.1531	0.1807	416.6	816.1	2.781	-162.3	-22.45
a_i	-0.000149	-0.00008	0.0004786	-162.0	-383.1	-0.02195	30.52	6.416

Table 6-4-1: Values of quadratic regression coefficients for X35333

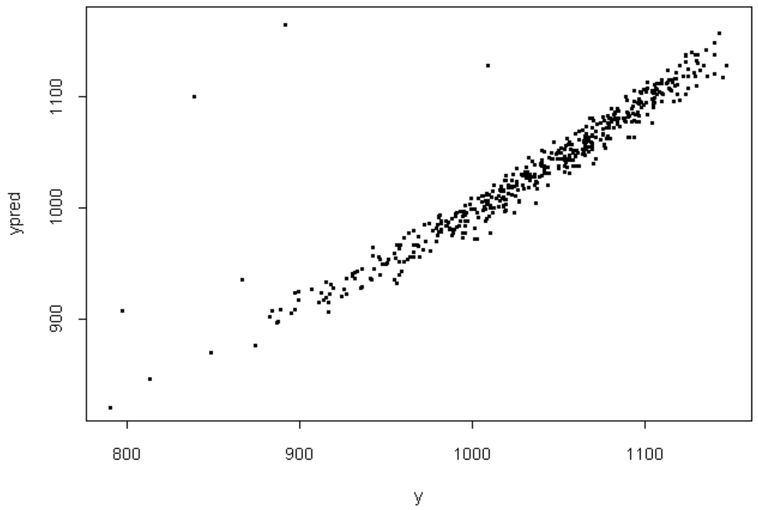


Figure 6-4-7: Scatter plot of y against y_{pred} of X3533

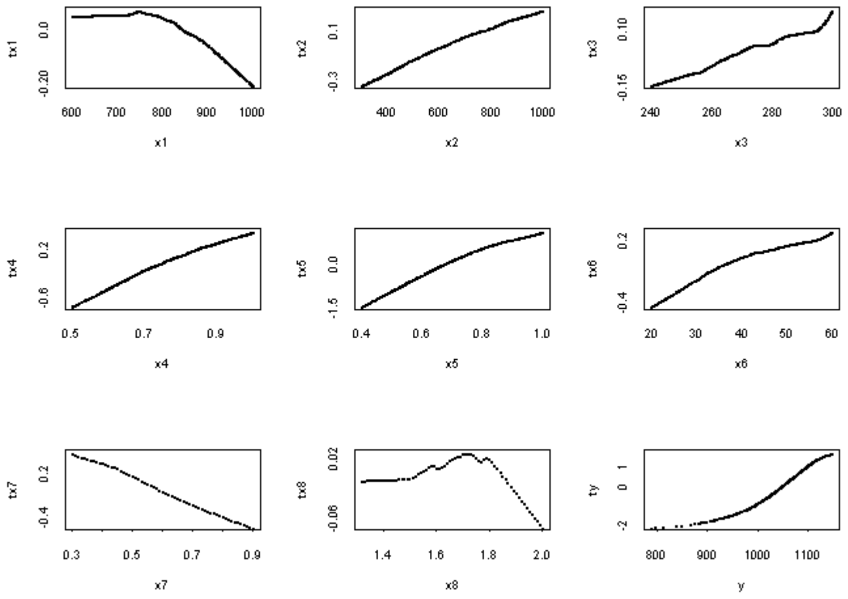


Figure 6-4-8: Plots of the transformed variables against the original data for X35333

Letting $$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}4\text{-}4)$$

it was found that the correlation between y and y_{pred} was 0.9864. A scatter plot of y against y_{pred} is shown in Figure 6-4-9.

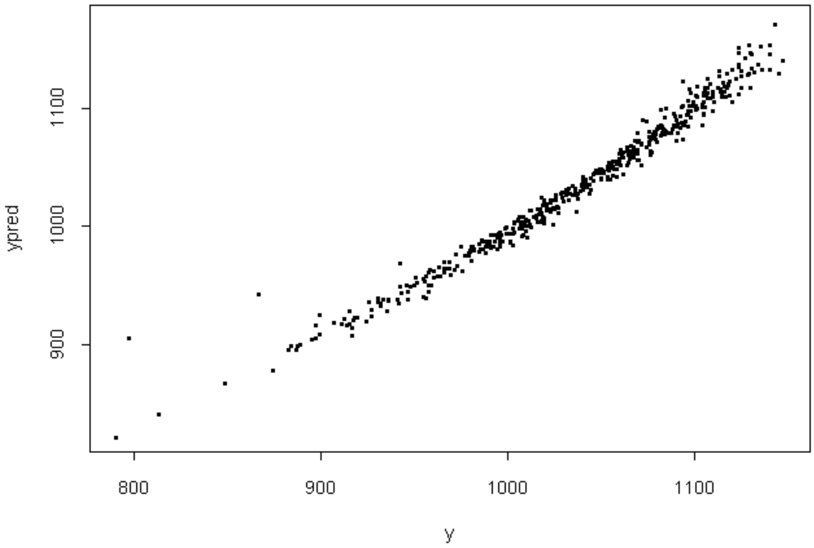


Figure 6-4-9: Scatter plot of y against y_{pred} of X35333

6.4.2.2 AVAS regression analysis for X3532

We apply the AVAS regression analysis to data X3532. It was found that there were 12 outliers. Deleting them (7, 157, 46, 30, 12, 82, 116, 249, 91, 27, 160, 77), we have X35321 which has 367 observations. And the transformed variables against original data are shown in Figure 6-4-10.

To the X35321 data set, we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon . \qquad (6\text{-}4\text{-}5)$$

The coefficient c was 74.15. Coefficients a_i and b_i were as in Table 6-4-2.

i	1	2	3	4	5	6	7	8
b_i	0.9711	0.09508	2.180	442.0	625.6	2.828	-68.73	-156.2
a_i	-0.00093	-0.00004	-0.00393	-197.7	-350.8	-0.02116	-14.93	45.49

Table 6-4-2: Values of quadratic regression coefficients for X35321

Setting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}4\text{-}6)$$

it was found that the correlation between y and y_{pred} is 0.9373. The scatter plot of y against y_{pred} is shown in Figure 6-4-11.

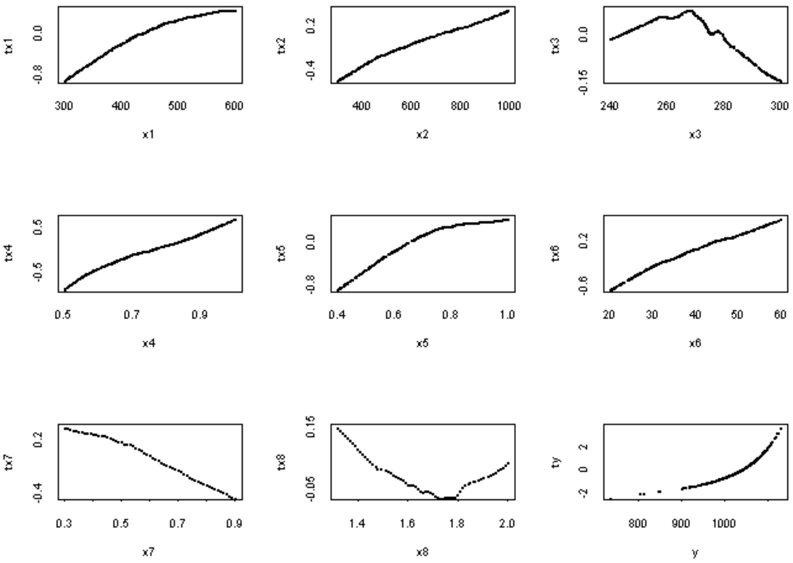


Figure 6-4-10: Plots of the transformed variables against the original data for X35321

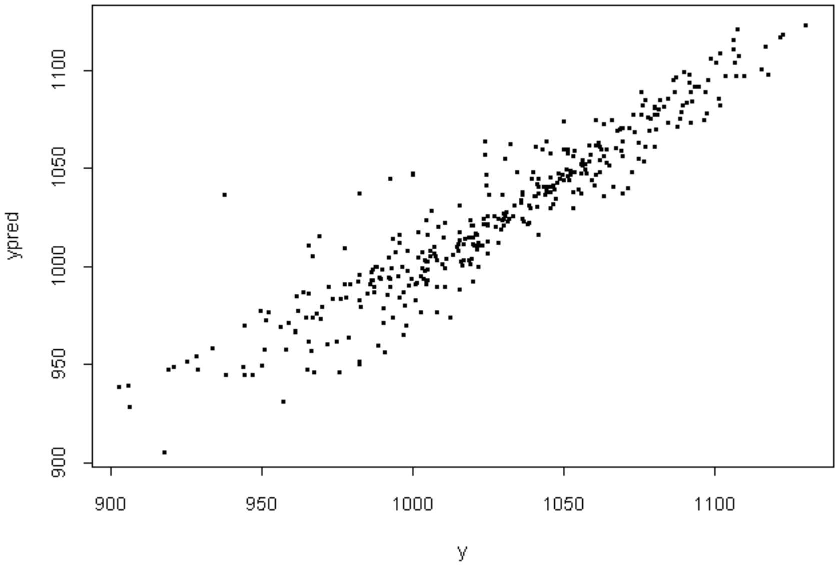


Figure 6-4-11: Scatter plot of y against y_{pred} X35321

6.4.3 AVAS regression analysis for X352 (0.96<Rf ≤ 1.32 m/sec)

The scatter plots of one variable against another for X352 are shown in Figure 6-4-12.

From analyzing Figure 6-4-12, sub-scatter plot of x_7 against x_8 has a special influence on the output, and further analysis shows that there is a change of behaviour of fire growth at equation:

$$x_8 = 0.8 + 0.603x_7$$

This equation $x_8 = 0.8 + 0.603x_7$ separates data X352 into two new data sets, referred to as X3521 and X3522 as follows

X3521: when $x_8 < 0.8 + 0.603x_7$ with 259 observations, and

X3522: when $x_8 > 0.8 + 0.603x_7$ with 208 observations.

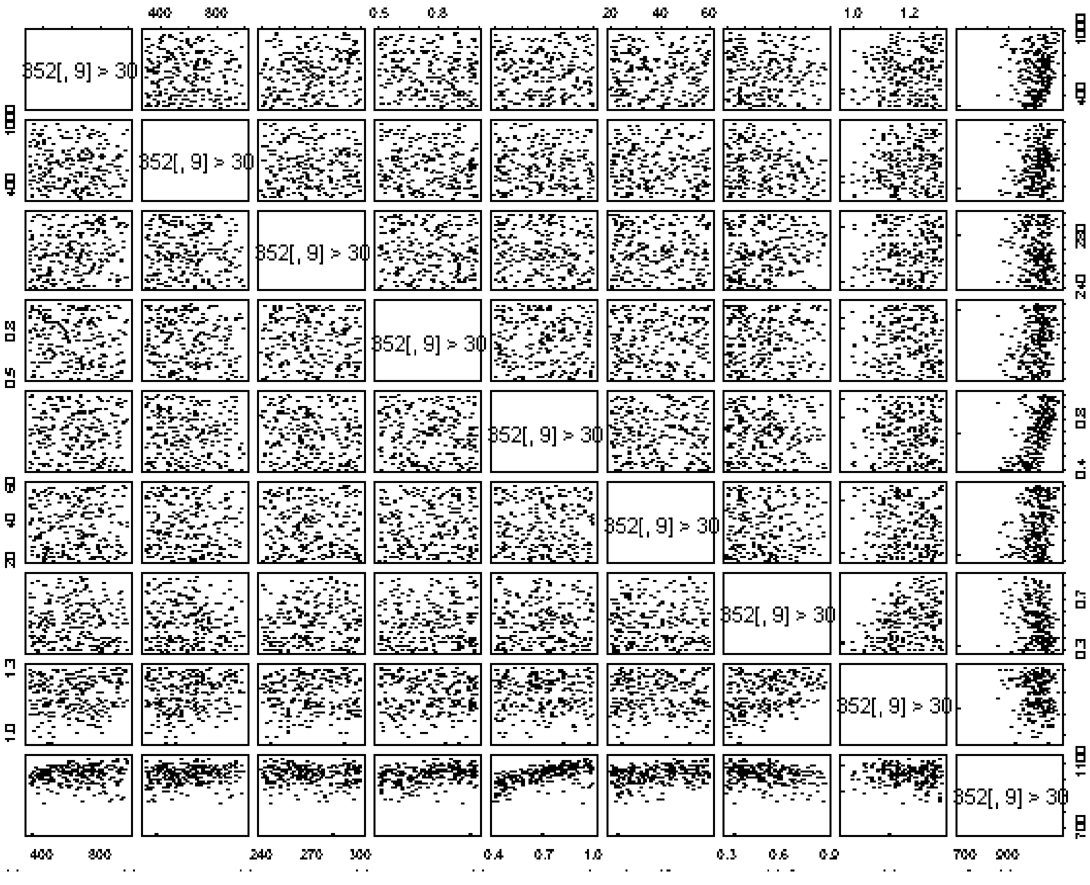


Figure 6-4-12: The scatter plots of one variable against another for X352.

6.4.3.1 AVAS regression analysis for X3522

By using the AVAS regression analysis on X3522, the transformed maximum temperature against the transformed predicted values of maximum temperature is shown in Figure 6-4-13. It is clear that there were some outliers.

Deleting outliers (6,7,12,14,21,30,40,44,63,69,83,87,134,146,167,186,208) in Figure 6-4-13, left us with new data set which we call it X3522f (191 obs). And its transformed variables against original data are shown in Figure 6-4-14.

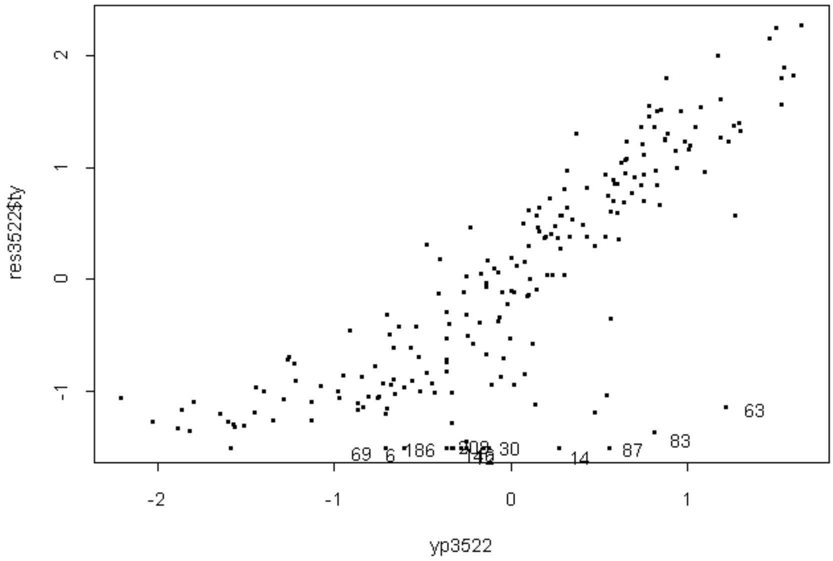


Figure 6-4-13: Transformed values against the predicted values of y in AVAS

To the X3522f data set we fitted a quadratic regression formula of the form

$$y = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c + \varepsilon . \tag{6-4-7}$$

The coefficient c was 823.0. Coefficients a_i and b_i were as in Table 6-4-3.

i	1	2	3	4	5	6	7	8
b_i	0.2535	0.1816	-0.5933	350.6	684.8	3.330	-130.8	-653.9
a_i	-0.00019	-0.00010	0.00150	-125.1	-336.7	-0.02716	34.38	264.5

Table 6-4-3: Values of quadratic regression coefficients for X3522f

Letting

$$y_{pred} = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c \qquad (6\text{-}4\text{-}8)$$

it is found that the correlation between y and y_{pred} (quadratic fit) is 0.937. The scatter plot is shown in Figure 6-4-15.

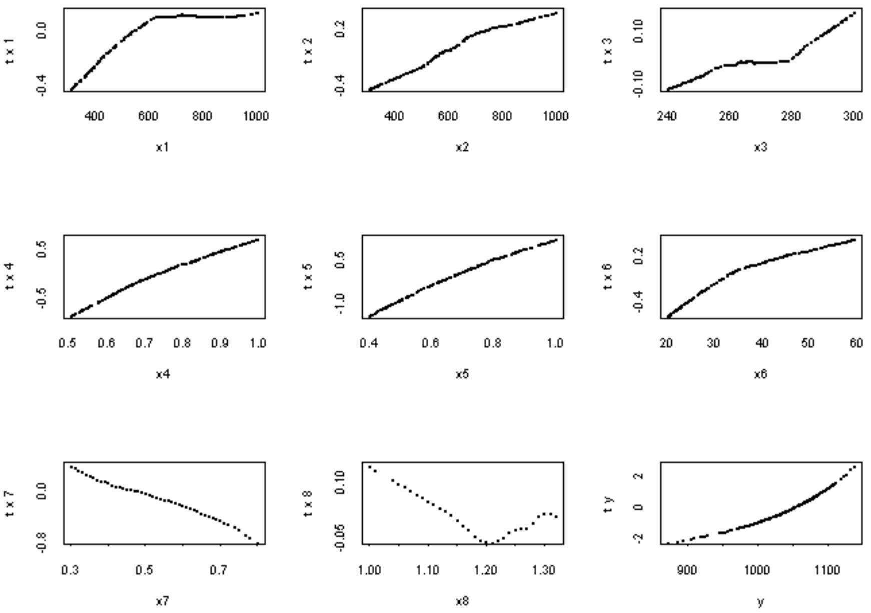


Figure 6-4-14: Transformed variables against original data in X3522f

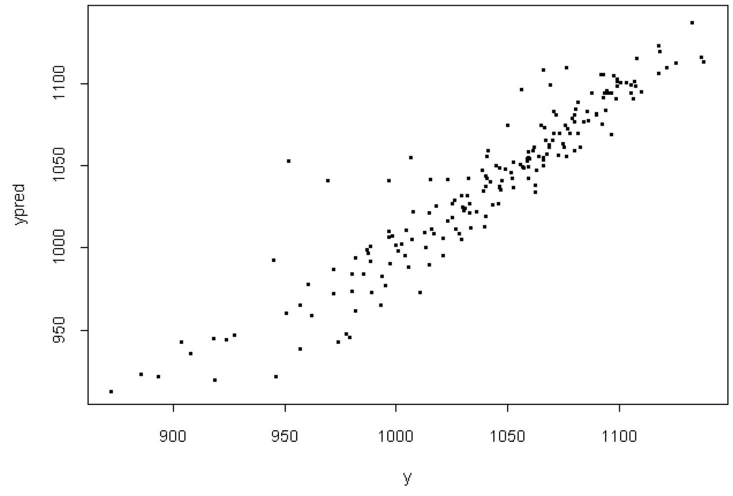


Figure 6-4-15: Scatter plot of y against y_{pred} (quadratic fitted to X3522f)

6.4.3.2 AVAS regression analysis for X3521

The scatter plots of one variable against another for X3521 are shown in Figure 6-4-16.

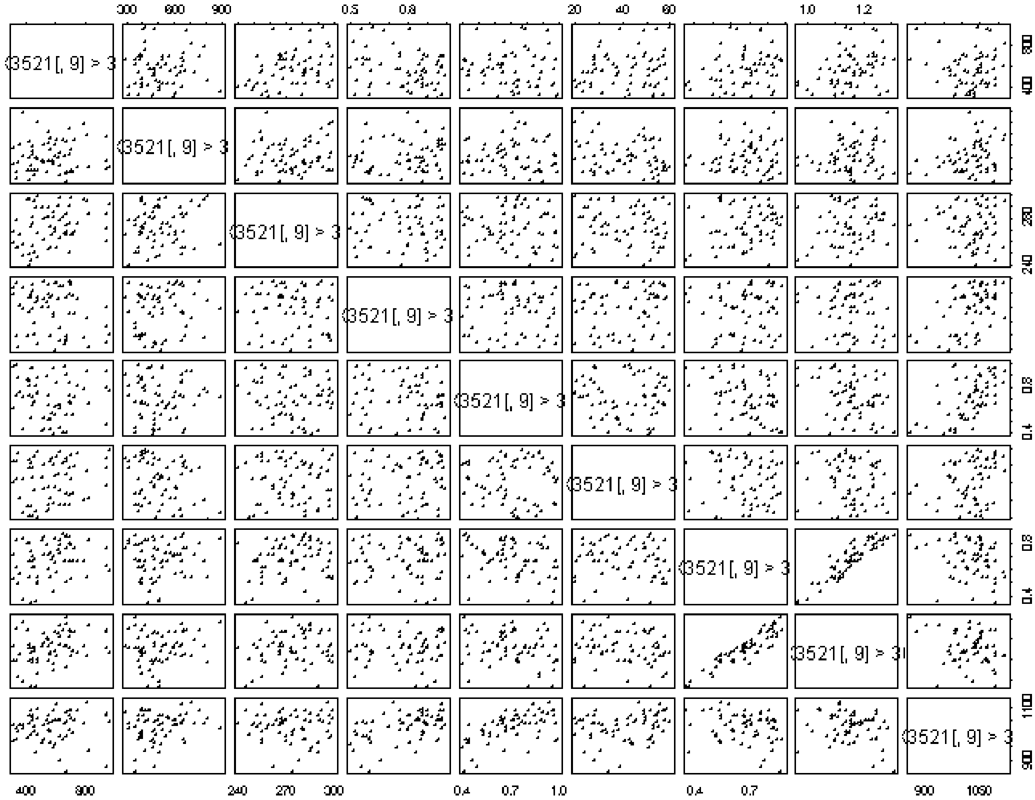


Figure 6-4-16: The scatter plots of one variable against another for X3521

From analyzing Figure 6-4-16, sub-scatter plot of x_7 against x_8 (which are fuel area factor and flame spread rate respectively) has a special influence on the output, and further analysis shows that there is a change of behaviour of fire growth at equation:

$$x_8 = 0.8503 + 0.3235x_7$$

This equation $x_8 = 0.8503 + 0.3235x_7$ separates data X3521 into two new data sets, referred to as X35211 and X35212 as follows

X35211with 113 observations: when $x_8 < 0.8503 + 0.3235x_7$;

X35212 with 146 observations: when $x_8 > 0.8503 + 0.3235x_7$.

6.4.3.2.1 AVAS regression analysis for X35211

By using AVAS regression analysis on X35211, the plots of transformed variables against the original data are shown in Figure 6-4-17.

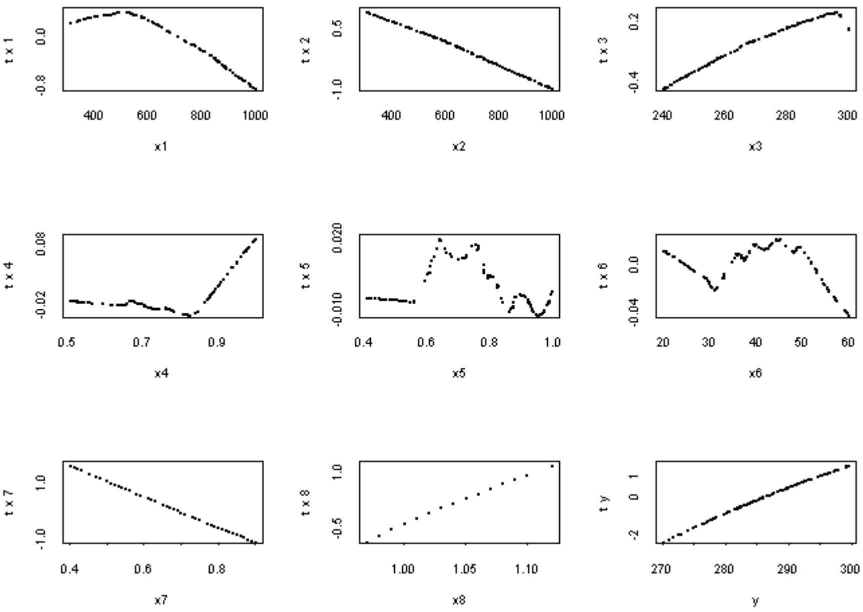


Figure 6-4-17: Transformed variables against original data of X35211

From analysing Figure 6-4- 17, it is found that inputs x_4, x_5 and x_6 can be ignored because their effect is very small. To the X35211data set we fitted a quadratic regression formula of the form

$$y = \sum_{i \in 1} a_i x_i^2 + \sum_{i \in 1} b_i x_i + c + \varepsilon \, , I = 1,2,3,7,8. \tag{6-4-9}$$

The coefficient c = -20.96. The a_i and b_i were as in Table 6-4-4.

i	1	2	3	7	8
b_i	0.03135	-0.01837	0.4483	-50.87	431.6
a_i	-0.0000345	-1.835e-006	-0.0006558	7.633	-159.3

Table 6-4-4: Values of quadratic regression coefficients for X35211

Letting

$$y_{pred} = \sum_{i \in 1} a_i x_i^2 + \sum_{i \in 1} b_i x_i + c \tag{6-4-10}$$

it is found that the correlation between y and y_{pred} (quadratic fit) is 0.9789, and the scatter plot is shown in Figure 6-4-18.

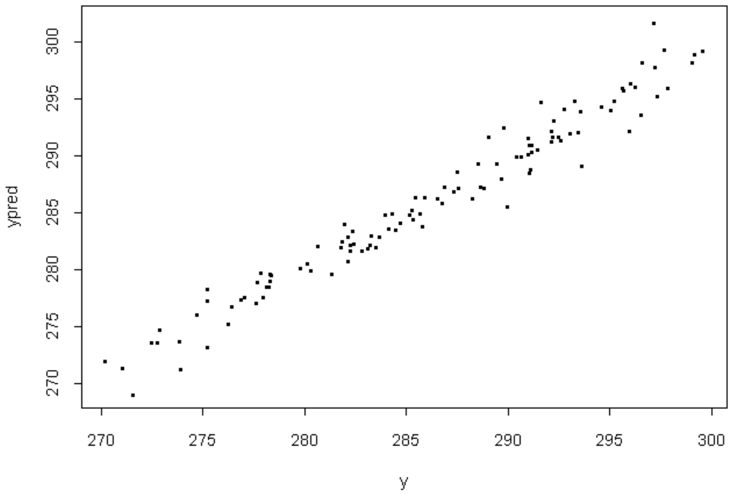


Figure 6-4-18: Scatter plot of y against y_{pred} (quadratic fitted) of X35211

6.4.3.2.2 AVAS regression analysis for X35212

By using the AVAS regression analysis on X35212, the scatter plot of transformed temperature against the transformed predicted values of the maximum temperature is shown in Figure 6-4-19. And plots of the transformed variables against the original data are shown in Figure 6-4-20.

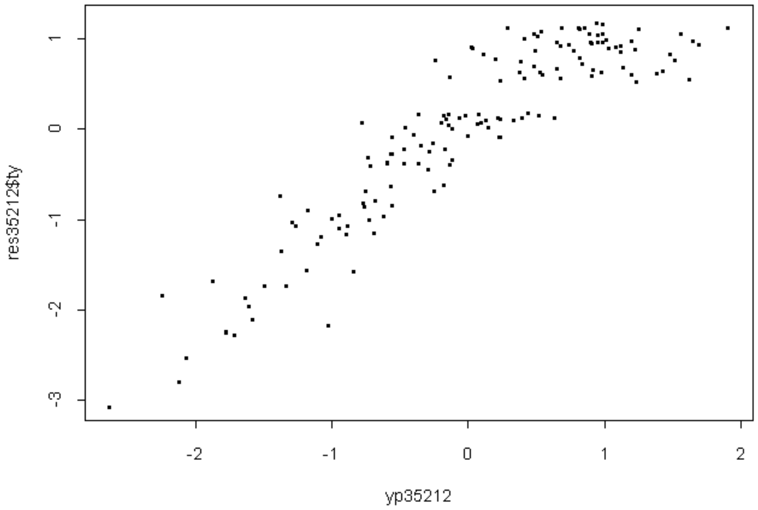


Figure 6-4-19: Scatter plot of transformed y against y_{pred} in AVAS of X35212

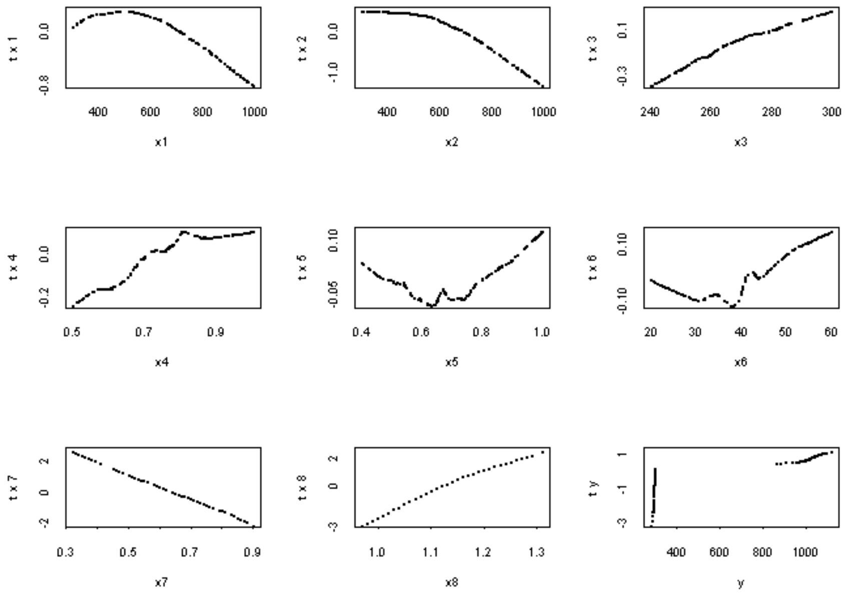


Figure 6-4-20: Transformed variables against original data AVAS of X35212

To X35212, we fitted a quadratic regression formula of the form

$$y = \sum_{i \in I} a_i x_i^2 + \sum_{i \in I} b_i x_i + c + \varepsilon \text{, } I = 1 \text{ to } 8. \qquad (6\text{-}4\text{-}11)$$

The coefficient c = 2635.2, b_i and a_i are as in Table 6-4-5.

i	1	2	3	4	5	6	7	8
b_i	1.114	0.1621	-15.62	305.0	437.0	-14.28	453.9	-4389.5
a_i	-0.00131	-0.00075	0.03580	-61.60	-134.7	0.2087	-1691.5	3814.9

Table 6-4-5: Values of quadratic regression coefficients for X35212

Letting

$$y_t = \sum_{i \in I} a_i x_i^2 + \sum_{i \in I} b_i x_i + c \qquad (6\text{-}4\text{-}12)$$

it was found that the correlation between y and y_t is 0.8061, and the scatter plot of y against y_t is shown in Figure 6-4-21.

To improve the correlation, we fit y_t with a cubic formula:

$$y_{predf} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 + \varepsilon . \qquad (6\text{-}4\text{-}13)$$

The coefficients are

C_0 = 337.3; C_1 = -1.799; C_2 = 0.005254; C_3 = -2.700e-006.

Setting

$$y_{predf} = C_0 + C_1 y_t + C_2 y_t^2 + C_3 y_t^3 \qquad (6\text{-}4\text{-}14)$$

the correlation between y and y_{predf} is now 0.8748, and the scatter plot of y against y_{predf} is shown in Figure 6-4-22.

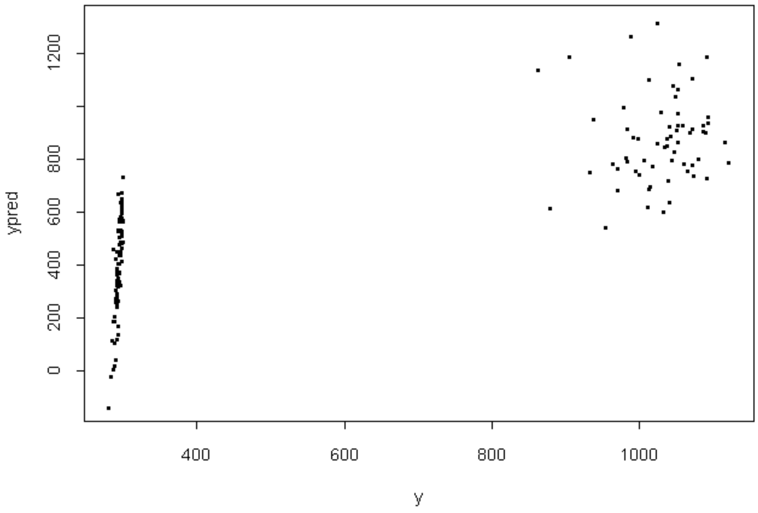


Figure 6-4-21: Scatter plot of y against y_{pred} (quadratic fitted)

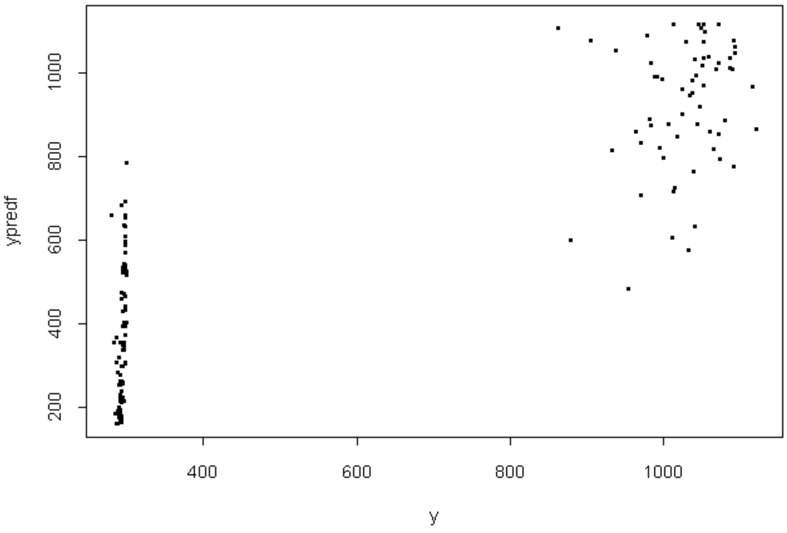


Figure 6-4-22: Scatter plot of y against y_{predf} (cubic fitted)

6.5 Statistics of outliers

Tables 6-5-1 to 6-5-4 show the number of outliers in each subset of the four scenarios. The total number of outliers was 259 out of 10,000 data sets, i.e. less than 2.6%. In addition all but one of the outliers lay below the regression line. This means that neglecting them and replacing them by a predicted value that was larger than the observed value in computing the reliability of a design would make the design safer (since the evaluated reliability would be lower than the one computed directly from the computer model).

The only outlier that lay above the regression line was in subset X122 (labelled 6 in Figure 6-3-18) and in any case lay in the lower range of maximum temperatures, so that rejecting it could not realistically affect the reliability of any reasonable design.

Name of sub-ranges	Total observations	Number of outliers
	X233: 1176	7
	X2322: 159	11
	X2323: 102	19
	X22322: 133	19
	X22321: 69	0
	X2233: 550	7
	X222: 311	29
Total	2500	92

Table 6-5-1 Outliers for DOWO scenario

Name of sub-ranges	Total observations	Number of outliers
	X133: 1181	5
	X132: 256	14
	X1232: 439	6
	X1233: 416	21
	X122: 208	10
Total	2500	56

Table 6-5-2 Outliers for DCWO scenario

Name of sub-ranges	Total observations	Number of outliers
	X433: 850	0
	X4322: 121	7
	X4323: 81	16
	X4222: 200	20
	X4223: 110	20
	X4233: 849	7
	X42322: 162	9
	X42323: 127	0
Total	2500	79

Table 6-5-3 Outliers for DOWC scenario

Name of sub-ranges	Total observations	Number of outliers
	X351: 1166	0
	X3533: 488	3
	X3532: 379	12
	X3522: 208	17
	X35211: 113	0
	X35212: 146	0
Total	2500	32

Table 6-5-4 Outliers for DCWC scenario

6.6 Summary of the regression analysis results

The regression results can be summarized as below:

1. The AVAS regression analysis result for DOWO scenario is summarized in Figure 6-6-1.

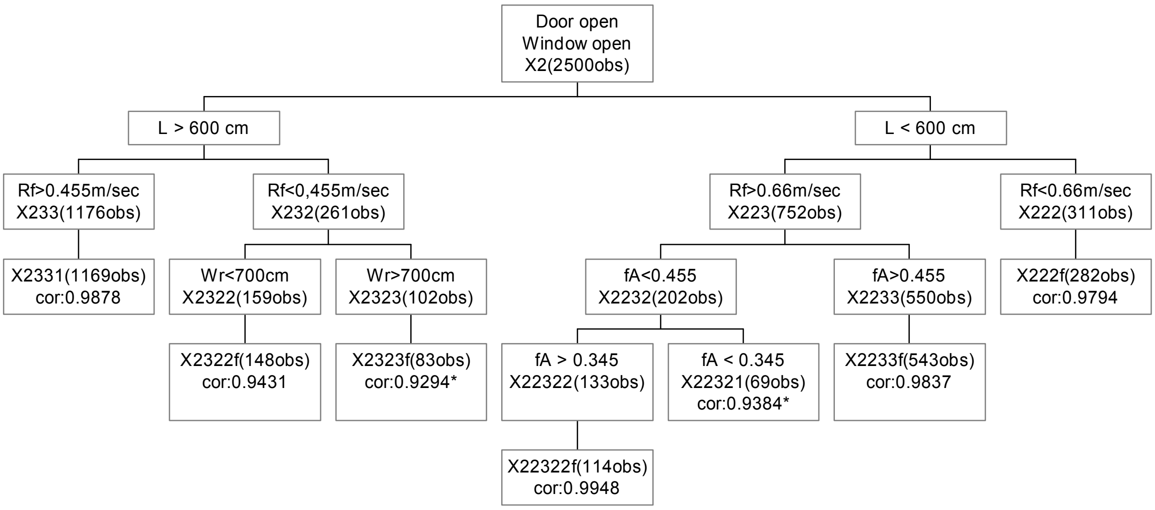


Figure 6-6-1: DOWO scenario

2. The AVAS regression analysis result for DCWO scenario is summarized in Figure 6-6-2.

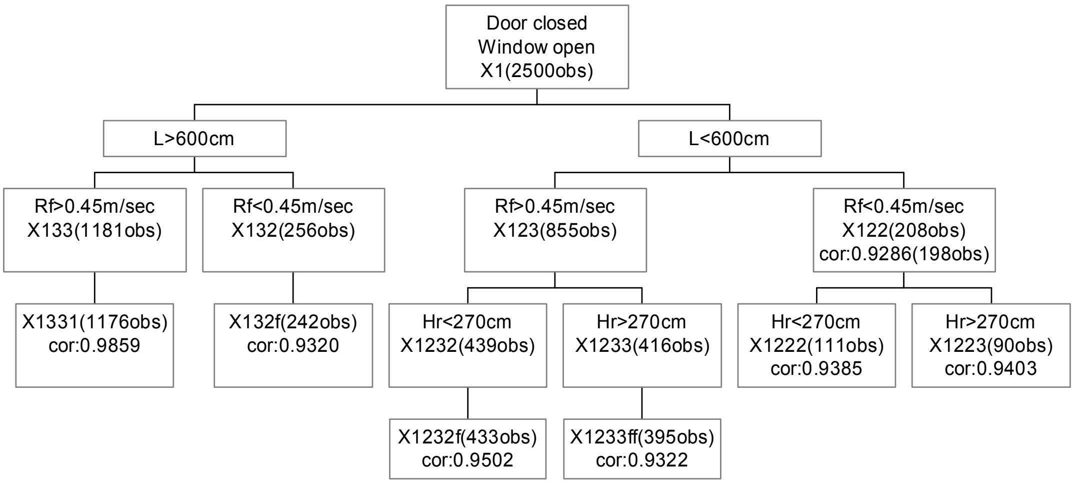


Figure 6-6-2: DCWO scenario

3. The AVAS regression analysis result for DOWC scenario is summarized in Figure 6-6-3.

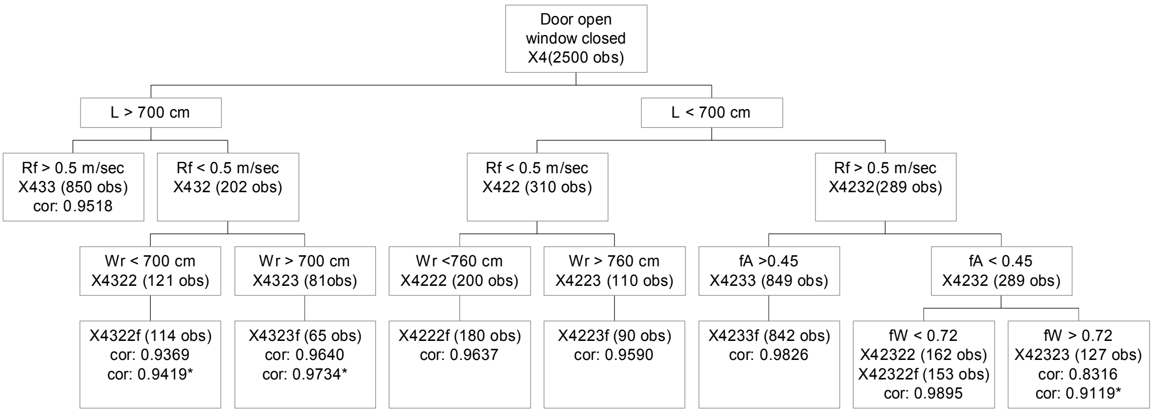


Figure 6-6-3: DOWC scenario

4. The AVAS regression analysis result for DCWC scenario is summarized in Figure 6-6-4.

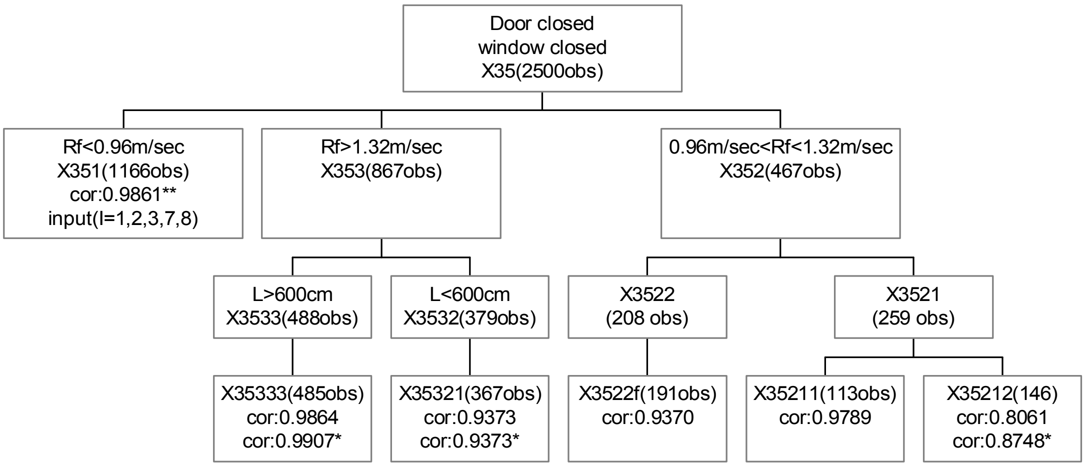


Figure 6-6-4: DCWC scenario

In Figures 6-6-1 to 6-6-4, rectangular boxes represent sub-ranges in the scenario. cor represent the correlation between original outputs (maximum temperature reached) and the predicted values by using the specified regression formulas. The values of correlation with an asterisk represent the correlation when a cubic fitted formula is used to improve the correlation.

CHAPTER 7
RELIABILITY ANALYSIS IN DESIGN FOR MAXIMUM TEMPERATURE REACHED

In this chapter, details of finding the design point and reliability index using Lagrange's method of undetermined multipliers will be given. Also, the reliability index for specific examples of maximum temperature reached for four scenarios will be calculated. The corresponding probability of failure for each of the scenarios is obtained by the use of FOSM Method and results validated by Monte-Carlo simulation.

7.1 Reliability index methodology in fire engineering

The particular shape of the regression equation derived from AVAS in Chapter 6 makes the task of finding the design point and the reliability index not difficult. For a fixed value of y_t the limit surface equation is

$$y_t = \sum_{i=1}^{n} a_i x_i^2 + \sum_{i=1}^{n} b_i x_i + c_i \tag{7-1-1}$$

Suppose the physical variables **X** are independent and normally distributed. For i = 1, …, n let X$_\text{i}$ have mean μ_i and standard deviation σ_i and let

$$u_i = \frac{x_i - \mu_i}{\sigma_i}. \tag{7-1-2}$$

Let the image of the limit surface in the U plane be

$$y = \sum_{i=1}^{n} A_i u_i^2 + \sum_{i=1}^{n} B_i u_i + C_i. \tag{7-1-3}$$

The design point D (which can be used to derive design values) is defined as the point on the limit surface nearest to the origin. In other words, we look for a vector u satisfying equation (7-1-3) that minimizes $\beta^2 = \sum_{i=1}^{n} u_i^2$.

It can be easily found by using Lagrange's method of undetermined multipliers.

Let

$$y^* = \lambda \sum_{i=1}^{n} u_i^2 + \sum_{i=1}^{n} A_i u_i^2 + \sum_{i=1}^{n} B_i u_i + C_i \tag{7-1-4}$$

Then we must have

$$\frac{\partial y^*}{\partial u_i} = 2\lambda u_i + 2A_i u_i + B_i = 0 \qquad (7\text{-}1\text{-}5)$$

from which we deduce

$$u_i = \frac{-B_i}{2(\lambda + A_i)}. \qquad (7\text{-}1\text{-}6)$$

Replacing in equation (7-1-3), we see that λ must satisfy the following equation (7-1-7)

$$\sum_{i=1}^{n} \frac{A_i B_i^2}{4(\lambda + A_i)^2} - \sum_{i=1}^{n} \frac{B_i^2}{2(\lambda + A_i)} + \sum_{i=1}^{n} C_i = y. \qquad (7\text{-}1\text{-}7))$$

This is a polynomial equation in λ of order $2n$.
Because of the construction of the problem, this equation always has at least one real root. We choose the real root that minimizes $\beta = \sqrt{\sum_{i=1}^{n} u_i^2}$. This minimum value of β is the required reliability index.

The corresponding probability of failure is given approximately by

$$p_F = \Phi(-\beta). \qquad (7\text{-}1\text{-}8)$$

The above methodology will be illustrated by applications in the following sections.

7.1.1 Choosing the correct subset for calculating the reliability index

When calculating the reliability index corresponding to a given limit surface, it is necessary to ensure that the limit surface in the neighbourhood of the design point is represented by the regression equation appropriate to that design point. In the first instance, the regression equation corresponding to the mean values of the inputs should be used and the design point found. If the design point does not lie in the same subset as the mean point, the subset in which it lies should be found and a new reliability calculation should be performed. If necessary, this procedure should be repeated until the design point lies in the subset corresponding to the regression equation used.

In all the examples that will be presented in the thesis the mean point and the design point lie in the same subset.

7.2 Reliability analysis of the DOWO scenario

7.2.1 Reliability Index for Engineering Design in DOWO scenario

Consider the DOWO scenario, when room length L > 600 cm and fire spread rate R_f > 0.455m/sec. For simplicity, we shall take just two input variables to be random: the fuel density ρ_f , denoted by x_6 and the fuel area factor f_A, denoted by x_7. The other input variables will be taken to be constant. This would usually be the case when dealing with a known building, since these other variables are just geometrical dimensions, apart from the flame spread rate R_f . However, we saw in Chapter 6 that the flame spread rate does not appear in the regression equation as long as it is larger than 0.455 m/sec in this scenario. Let $N(\mu, \sigma)$ denote a normal random variable with mean μ and standard deviation σ. The assumed values are as follows:

L = 800 cm, W = 500 cm, H = 250 cm, f_w = 0.7, f_H= 0.5, ρ_f = N(30,2), f_A = N(0.6,0.1), R_f = 1.5 m/sec.

Let: $x_6=\rho_f$, $x_7=f_A$.

The limiting state is taken to be $y_{\max} = 1050\,°\mathrm{C}$. This limiting state is chosen here to demonstrate the methodology for calculating the probability of failure defined as the probability of exceeding a given value. It is needed to determine the probability of danger to life and structure damage. Another probability of failure would be obtained with a different limiting condition. After introducing the given data into the formula developed in previous sections, Chapter 6, the regression equation is reduced to

$$y_t = 34.65x_7^2 - 0.01810x_6^2 + 2.419x_6 - 155.3x_7 + 1047.1 \qquad (7\text{-}2\text{-}1)$$

Equation (7-2-1) was standardised by using the following equations according to (7-1-2)

$$u_1 = \frac{x_6 - \mu_6}{\sigma_6} \qquad (7\text{-}2\text{-}2a)$$

$$u_2 = \frac{x_7 - \mu_7}{\sigma_7}. \qquad (7\text{-}2\text{-}2b)$$

This yields

$$y = \sum_{i=1}^{2} A_i u_i^2 + \sum_{i=1}^{2} B_i u_i + C_i \qquad (7\text{-}2\text{-}3)$$

where

$$A_1 = a_1\sigma_1^2$$
$$B_1 = 2a_1\sigma_1\mu_1 + b_1\sigma_1$$
$$C_1 = a_1\mu_1^2 + b_1\mu_1 + c_1$$

and

$$A_2 = a_2\sigma_2^2$$
$$B_2 = 2a_2\sigma_2\mu_2 + b_2\sigma_2$$
$$C_2 = a_2\mu_2^2 + b_2\mu_2 + c_2.$$

We find

A_1= -0.0732, B_1= 2.674, C_1= 1099.2, A_2= 0.3380, B_2= -11.36, C_2= -80.31, and C_i = C_1+ C_2= 1099.2 - 80.31 = 1018.9.

The standardized equation (limit state function) for the two standardized variables is as follows

$$y = -0.0732u_1^2 + 2.674u_1 + 0.3380u_2^2 - 11.36u_2 + 1018.9 \qquad (7\text{-}2\text{-}4)$$

From Lagrange's method of undetermined multipliers λ

$$\sum_{i=1}^{2}\frac{A_iB_i^2}{4(\lambda + A_i)^2} - \sum_{i=1}^{2}\frac{B_i^2}{2(\lambda + A_i)} + \sum_{i=1}^{2}C_i + \lambda\sum_{i=1}^{2}u_i^2 = y_{\max}. \qquad (7\text{-}2\text{-}5)$$

It is not difficult to derive the values of λ from the equation (7-2-5):

λ: 0.0573 0.0573 -0.1798 -2.978

From equation (7-1-6), we have (7-2-6a,b) as follows:

$$u_1 = \frac{-B_1}{2(\lambda + A_1)} \qquad (7\text{-}2\text{-}6a)$$

$$u_2 = \frac{-B_2}{2(\lambda + A_2)} \qquad (7\text{-}2\text{-}6b)$$

$$\beta^2 = \sum_{i=1}^{n}u_i^2 = u_1^2 + u_2^2 \qquad (7\text{-}2\text{-}7)$$

From equations (7-2-6) and (7-2-7), the corresponding values of β can be calculated

β: 89.51 89.51 34.51 2.204.

Obviously the smallest value β=2.204 is the reliability index. Illustration of the result is given in Figure 7-2-1 and Figure7-2-2.

From equation (7-1-8), the corresponding probability of failure is given approximately by $p_F = \Phi(-\beta) = 0.01376$. Therefore, the probability for the maximum temperature to be greater than 1050 °C is 0.0138 in the specified DOWO scenario.

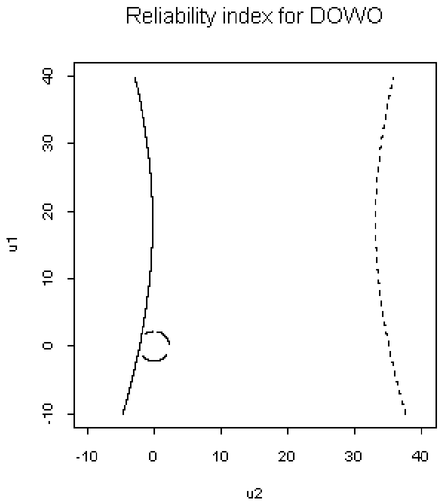


Figure 7-2-1: Illustration of the β index for the numerical example

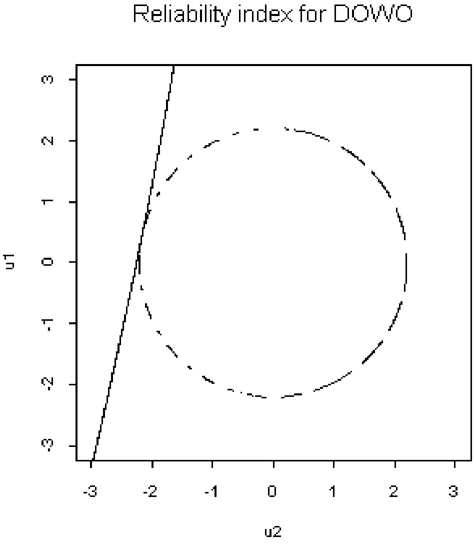


Figure 7-2-2: Illustration of the β index for the numerical example

7.2.2 Validation by Monte - Carlo simulation

The probability just obtained by using the reliability index in section **7.2.1** can be validated by Monte Carlo simulation: out of 200,000 simulations, 2,721 fell in the failure region y > 1050 °C, giving an estimated probability of failure of $p_{F(carlo)}$= 0.01361. The 95% confidence interval for the estimate is (0.0131, 0.0141), which contains the value (0.0138) obtained in using the reliability index method in the previous section **7.2.1**.

7.3 Reliability analysis of the DOWC scenario

7.3.1 Reliability Index for Engineering Design in DOWC scenario

Consider the DOWC scenario, when room length $L > 700$ cm and fire spread rate $R_f >$ 0.5m/sec (X433). For simplicity, we shall take as before just two input variables to be random: the fuel density ρ_f , denoted by x_6 and the fuel area factor f_A, denoted by x_7. The assumed values are as follows: L = 999.5 cm, W = 600 cm, H = 260 cm, f_w = 0.7, f_H = 0.6, ρ_f = N(40,2), f_A = N(0.8,0.1), R_f = 1.2 m/sec.

$x_6 = \rho_f$, $x_7 = f_A$

The regression equation is

$$y_t = -0.0198x_6^2 + 2.5803x_6 + 14.50x_7^2 - 134.3x_7 + 1035.5 \qquad (7\text{-}3\text{-}1)$$

The standardised equation is

$$y = \sum_{i=6}^{7} A_i u_i^2 + \sum_{i=6}^{7} B_i u_i + C_i \qquad (7\text{-}3\text{-}2)$$

The following values of coefficients are obtained:

A_6= -0.0792, A_7= 0.1450, B_6= 1.9926, B_7 = -11.11, C_6 = 1107.1, C_7 = -98.20,

$C_i = C_6 + C_7$ = 1008.9.

The standardized regression equation (limit state function) for the two-variable problem is

$$y = -0.0792u_6^2 + 1.9926u_6 + 0.1450u_7^2 - 11.11u_7 + 1008.9 \qquad (7\text{-}3\text{-}3)$$

From Lagrange's method of undetermined multipliers λ and for y_{max} = 1035 °C, it is not difficult to derive the values of λ as:

λ: 0.0743 -0.0738 0.0743 -2.6452.

By using equation (7-3-2a) and (7-3-2b) with

$$\beta^2 = \sum_{i=6}^{7} u_i^2 = u_6^2 + u_7^2 \qquad (7\text{-}3\text{-}4)$$

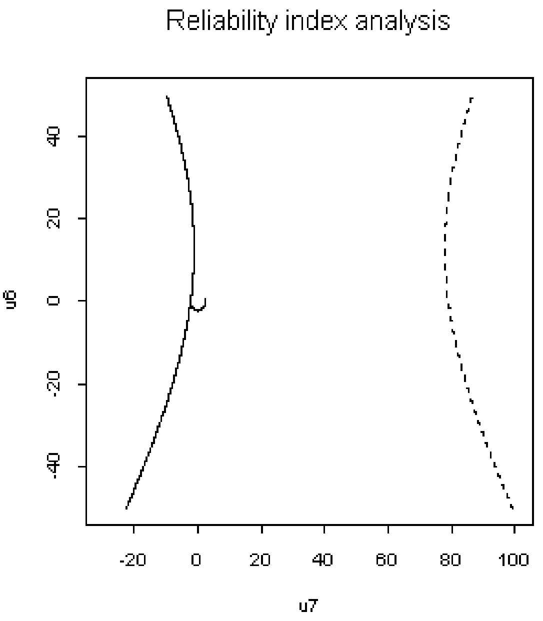
the reliability indices are:

β: 203.1 78.39 203.1 2.253.

Obviously the smallest value β = 2.253 is the reliability index. Illustration of the reliability index result is given in Figure 7-3-1 and Figure 7-3-2.

In this scenario, the approximate value of the probability for the maximum temperature greater than 1035°C is:

$p_{failure} = \Phi(-\beta) = 1 - \Phi(\beta) = 1 - \Phi(2.253) = 0.01214$.



-

Figure 7-3-1: Illustration of index for the numerical example (X433)

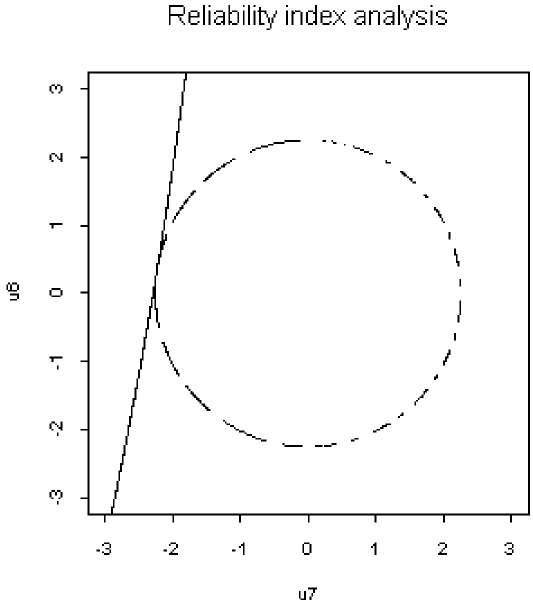


Figure 7-3-2: Illustration of index for the numerical example (X433)

7.3.2 Validation by Monte Carlo simulation

The probability just given can be validated by Monte Carlo simulation: Out of 200,000 simulations, 2464 fell in the failure region y_t > 1035°C, giving an estimated probability of failure of 0.01232. The 95% confidence interval for the estimate is (0.01184, 0.01280), which contains the value (0.01214) obtained in the previous section.

7.4 Reliability analysis of the DCWO scenario

7.4.1 Reliability Index for Engineering Design in DCWO scenario

In DCWO scenario, when room length L > 600 cm and fire spread rate R_f > 0.45m/sec (X133f), for simplicity, we shall take just two input variables to be random: the fuel density ρ_f, denoted by x_6 and the fuel area factor f_A, denoted by x_7.

The assumed values are as follows:

L= 999.5 cm, W = 600 cm, H =260 cm, fw = 0.7, f_H = 0.6, ρ_f = N(40,2), f_A = N(0.8,0.1),

R_f= 1.2m/sec.

Let: x_6=ρ_f, x_7 =f_A.

After bringing the given data into the formula developed in Chapter 6, we get the following reduced equation

$$y_t = -0.09107u_6^2 + 2.0209u_6 + 0.4414u_7^2 - 10.75u_7 + 967.2. \qquad (7\text{-}4\text{-}1)$$

The limiting state is taken to be y_{max} = 990°C. By using Lagrange's method of undetermined multipliers, it is not difficult to derive the values of λ as:

λ: 0.0792 0.0792 -0.2337 -3.254

and the corresponding values of β are as:

β: 85.52 85.52 26.07 1.935.

It is clear that the smallest value of β, 1.935, is the required reliability index.

From equation (7-1-8), the approximate probability of failure in this case is:

$p_F = \Phi(-1.935) = 0.02648.$

Illustration of the result is given in Figure 7-4-1 and Figure 7-4-2.

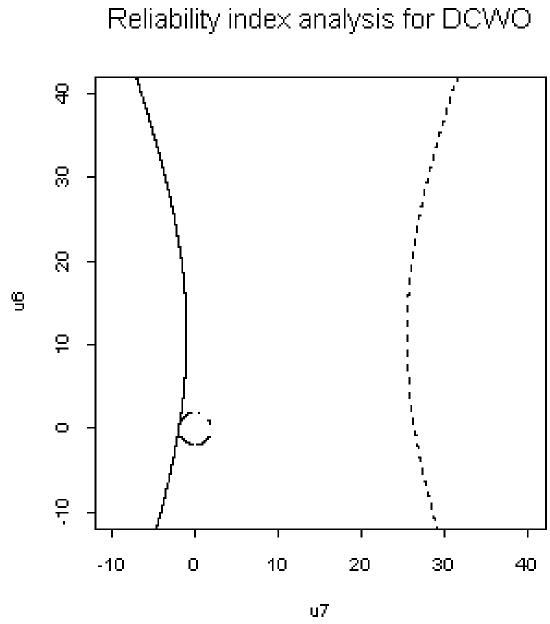


Figure 7-4-1: Illustration of the reliability index for the numerical example

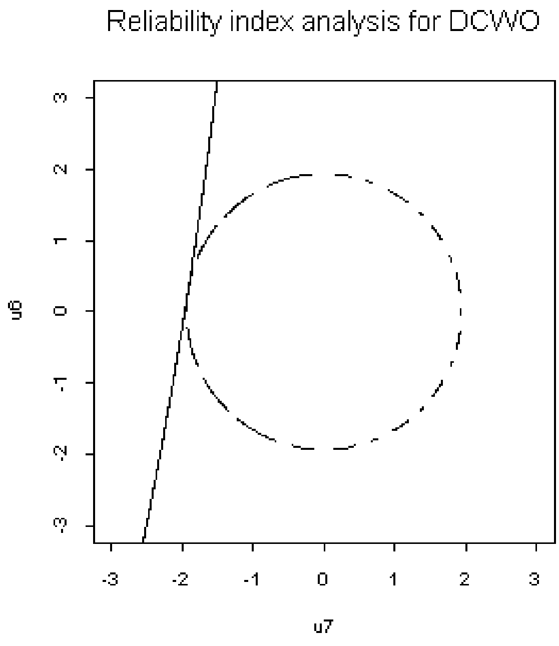


Figure 7-4-2: Illustration of the reliability index for the numerical example

7.4.2 Validation by Monte - Carlo simulation

The probability just given can be validated by Monte Carlo simulation: Out of 200,000 simulations, 5,229 fell in the failure region $y_t > 990$, giving an estimated probability of failure of 0.02615. The 95% confidence interval for the estimate is (0.02545, 0.02684), which contains the value 0.02648 obtained in the previous section.

7.5 Reliability analysis of the DCWC scenario

7.5.1 Reliability Index for Engineering Design in DCWC scenario (X351)

Consider the DCWC scenario, when the fire spread rate $R_f < 0.96$m/sec. Here, we shall take the two following input variables to be random: the fuel area factor f_A, denoted by x_7 and the fire spread rate R_f, denoted by x_8. The other input variables will be taken to be constant. This would usually be the case when dealing with a known building, since these other variables are geometrical dimensions. Moreover, we saw in

Chapter 6 that window width factor, window height factor and fuel density do not appear in the regression equation when $R_f < 0.96$m/sec in DCWC scenario.

The assumed values are as follows

$L = 900$ cm, $W_r = 500$ cm, $H_r = 260$ cm, $f_A = \mathrm{N}(0.6,0.1)$, $R_f = \mathrm{N}(0.8,0.1)$

We have the regression equation:

$$y_p = b_7 x_7 + b_8 x_8 + 143.5. \qquad (7\text{-}5\text{-}1)$$

By using $u_7 = (x_7 - 0.6)/0.1$, $u_8 = (x_8 - 0.8)/0.1$, we derive the standardised limiting state function as:

$$y_p = 17.79u_8 - 3.203u_7 + 266.6. \qquad (7\text{-}5\text{-}2)$$

For $y_{max} = 295$ °C, we have the limit equation:

$$17.79u_8 - 3.2034u_7 - 28.39 = 0. \qquad (7\text{-}5\text{-}3)$$

The reliability index is (the shortest distant from the origin to the limit curve line)

$$\beta = 1.571.$$

Thus the approximate value of the probability of failure from equation (7-1-8) is:

$$p_F = \Phi(-1.571) = 0.0581.$$

An illustration of the result is given in Figure 7-5-1.

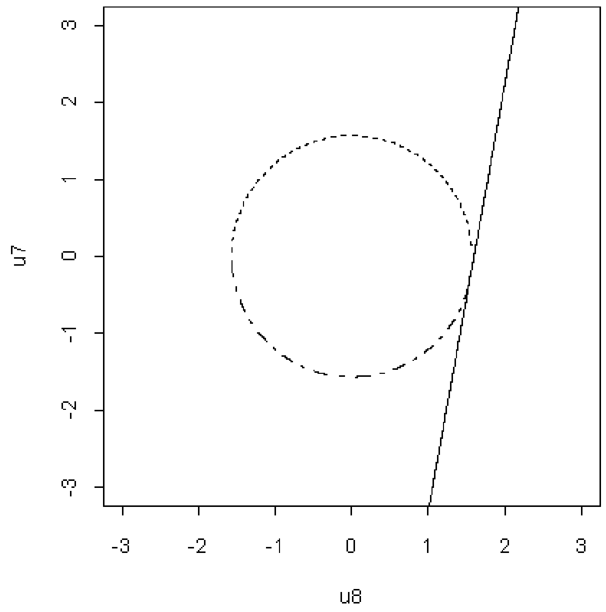


Figure 7-5-1: Illustration of the β index for the numerical example of DCWC (X351)

7.5.2 Validation by Monte - Carlo simulation for X351

The probability just given can be validated by Monte Carlo simulation: Out of 200,000 simulations 11493 fell in the failure region $y_p > 295$ °C, giving an estimated probability of failure of 0.0575. The 95% confidence interval for the estimate is (0.0565, 0.0585), which contains the value (0.0581) obtained in the previous section

7.5.3 Reliability index analysis for the DCWC scenario (when $R_f > 1.32$)

Consider the DCWC scenario, when the fire spread rate $R_f > 1.32$ m/sec (X3533). For simplicity, we shall take as before just two input variables to be random: the fuel density ρ_f, denoted by x_6 and the fuel area factor f_A, denoted by x_7.

The assumed values of input are as follows:
$L = 900$ cm, $W = 600$ cm, $H = 260$ cm, $f_w = 0.8$, $f_H = 0.7$, $\rho_f = \mathrm{N}(50,2)$, $f_A = \mathrm{N}(0.6,0.1)$, $R_f = 1.5$ m/sec.
Setting $x_6 = \rho_f$, $x_7 = f_A$.

The regression equation is:

$$y_t = 30.52x_7^2 - 162.3x_7 - 0.02195x_6^2 + 2.7812x_6 + 1060.5 \qquad (7\text{-}5\text{-}4)$$

and the reduced regression equation is

$$y_t = 0.3052u_7^2 - 12.57u_7 - 0.088u_6^2 + 1.1624u_6 + 1058.1. \qquad (7\text{-}5\text{-}5)$$

The limiting state is taken to be $y_{max} = 1100$°C.

Using Lagrange's method of undetermined multipliers, it is easy to calculate the values of λ as:

λ: 0.0859 -0.1628 0.0859 -2.347

and the corresponding values of β are:

β: 276.6 44.19 276.6 3.088.

The smallest value of $\beta = 3.088$, and it is the reliability index. Therefore, the approximate probability of failure is:

$p_F = \Phi(-3.088) = 0.0010.$

Illustration of the result is given in Figure 7-5-2 and Figure 7-5-3.

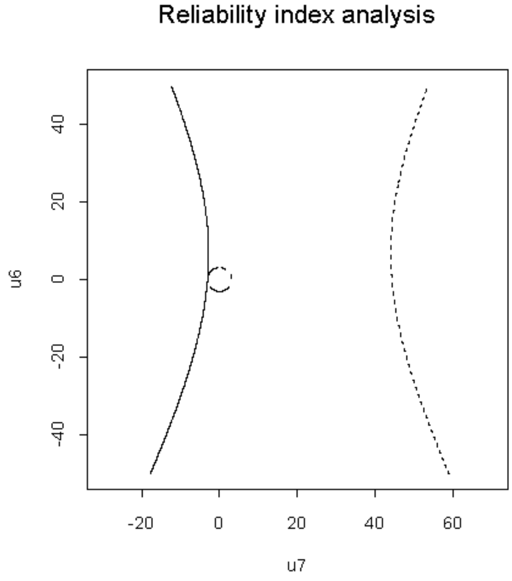


Figure 7-5-2: Illustration of the β index for the numerical example (X35333)

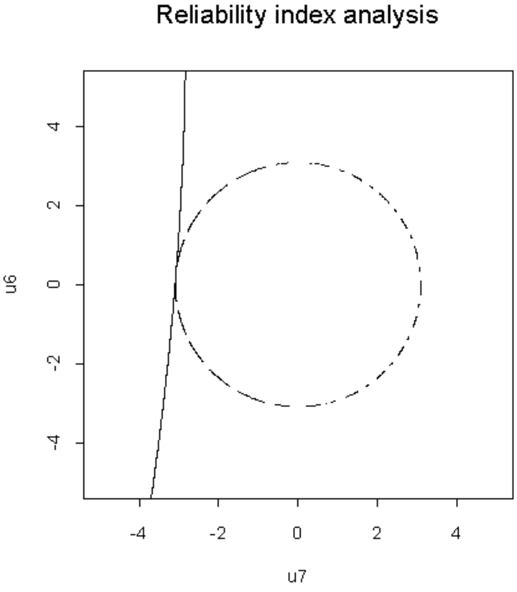


Figure 7-5-3: Illustration of the β index for the numerical example (X35333)

7.5.4 Validation by Monte-Carlo simulation:

The probability obtained by the β index can be validated by Monte Carlo simulation: Out of 200,000 simulations 220 fell in the failure region y_t > 1100 °C, giving an estimated probability of failure of 0.0011. The 95% confidence interval for the estimate is (0.00096, 0.00125), which contains the value (0.0010) obtained by using the β index methodology in the previous section.

7.6 Discussion of the regression results

The appropriate values of the highest temperature reached were chosen for each selected scenario, as shown in Table 7-1. The results of reliability analysis for the selected scenarios are also shown in Table 7-1. The analysis is based on data for specified constraints and specified input which were introduced in previous sections.

Scenarios	Number	Tem.	β	Φ (-β)	p_F	95% conf_interval
DOWO:X233	1169	1050°C	2.204	0.01376	0.01361	(0.0131, 0.0141)
DOWC:X433	850	1035°C	2.253	0.01214	0.01232	(0.01184, 0.01280)
DCWO:X133	1176	990°C	1.935	0.02648	0.02615	(0.02545, 0.02684)
DCWC:X351	1166	295°C	1.571	0.0581	0.0575	(0.0565, 0.0585)
X35333	485	1100°C	3.088	0.0010	0.0011	(0.00096, 0.00125)

Table 7-1: The results of reliability analysis for four scenarios

In table 7-1:

Number = number of observations; **Tem.** = highest temperature reached; β = reliability index; Φ = distribution function of normal standard distribution; p_F = probability of failure from Monte-Carlo simulation; **95% conf_interval** = 95% confidence interval of Monte-Carlo simulation.

CHAPTER 8
MODERN REGRESSION FOR TIME TO UNTENABLE CONDITIONS IN FIRES

8.1 Introduction

In this chapter, we will analyze another output variable, time to untenable conditions. Time to untenable conditions is defined as the time to a fatality occurring to occupants, that is, either occupant incapacitation will occur when the COHb (carboxy haemoglobin) dosage in blood exceeds a critical level or exposure to heat radiation reaches a critical high level.

8.1.1 Calculation of COHb Value

During evacuation under smoke conditions, occupants who are exposed to the smoke accumulate a COHb dosage in the blood, through inhaling CO and CO_2. Occupant incapacitation will occur when the contents of COHb in the blood exceed a critical level estimated to be 20% of total Hb. Fatality will occur when the critical level reaches 50%.

An equation (derived from experimental human exposures) for the prediction of COHb concentration is given by Stewart et al [60],

$$\%COHb = \int_0^t 3.317 \times 10^{-5} \times CO(t)^{1.036} \times RMV \times dt \qquad (8\text{-}1\text{-}1)$$

where *CO(t)* is the CO concentration in ppm as a function of time, *RMV* is the volume of air breathed (L/min) and *t* is the time of exposure (min.).

The toxic gases considered in CESARE-Risk are CO and CO_2 only, because for most practical situations the composition of the fire atmosphere is such that the toxic effects of CO are the most important. The effect of CO_2 is calculated by the Fire Growth Model. The CO_2 concentration is used to determine a factor by which the COHb from CO is multiplied to take into account the increase of the breathing rate caused by CO_2. This factor is calculated using

$$V_{CO_2} = \exp(0.2468 \times CO_2\% + 1.9086)/6.8 \qquad (8\text{-}1\text{-}2)$$

where V_{CO2} is the multiplication factor for CO_2 induced hyperventilation.

Thus, the total COHb with the effect of CO_2 is

$$\%COHb = \int_{t_1}^{t_2} 8.2925 \times 10^{-4} \{ppmCO(t)\}^{1.036} \times dt \times V_{CO_2} \qquad (8\text{-}1\text{-}3)$$

where 25 L/min is used for the rate of breathing which is the breathing rate for adults with light activity.

8.1.2 Fatality Caused by Heat

In this section another cause of incapacitation, exposure to heat radiation in a building cell, will be presented.

The most important sources of heat are radiative heat from the fire and convective heat from the hot gases. According to Babrauskas [61], the tenability limit for radiative heat flux Q_r is 2.5 kW/m^2 (0.25 W/cm^2). The radiative heat flux is closely related to the temperature. For simplicity, only temperature will be used for determining the occupant fatality condition.

For exposure to convective heat, the concept of fractional lethal dose (*FLD*) can be used to predict whether a fatality will occur. *FLD* is defined to be

FLD = dose received at time *t*/ dose to cause fatality. (8-1-4)

The fractional lethal dose due to convective heat (FLD_T) is calculated using:

$$FLD_T = \int_0^t dt/(199.3 \times \exp(\frac{60.89 - T(t)}{4.18}) + 52 \times \exp(\frac{60.89 - T(t)}{29.83})) \qquad (8\text{-}1\text{-}5)$$

where *T(t)* is the temperature of the hot gases as a function of time; *t* is the time at which the *FLD* is calculated.

When FLD_T is greater or equal to 1, the occupants are assumed to be fatalities. This equation is derived from the data presented in Purser [62] using curve fitting techniques.

Details of the above calculation in CESARE-RISK model can be found in Sanabria and Li [63].

8.2 Stochastic nature of time to untenable conditions

8.2.1 Stochastic nature of input variables

The stochastic nature of the inputs is the same as for the maximum temperature in the previous section.

8.2.2 Output variable

In the following section, we shall concentrate on the analysis of the time to untenable conditions = min (CO, Heat), denoted by *T*.

As before, we shall consider four scenarios:
Door closed, window open (DCWO);
Door open, window closed (DOWC);
Door closed, window closed (DCWC) and
Door open, window open (DOWO).

There are 10,000 simulation data sets available, 2,500 for each of the four scenarios.

8.3 Modern regression analysis for DCWO scenario

Applying the ACE regression algorithm to the DCWO data, referred to as U1, we obtain eight transformed inputs and a transformed output. Plots of the transformed data against the original data are shown in Figure 8-3-1.

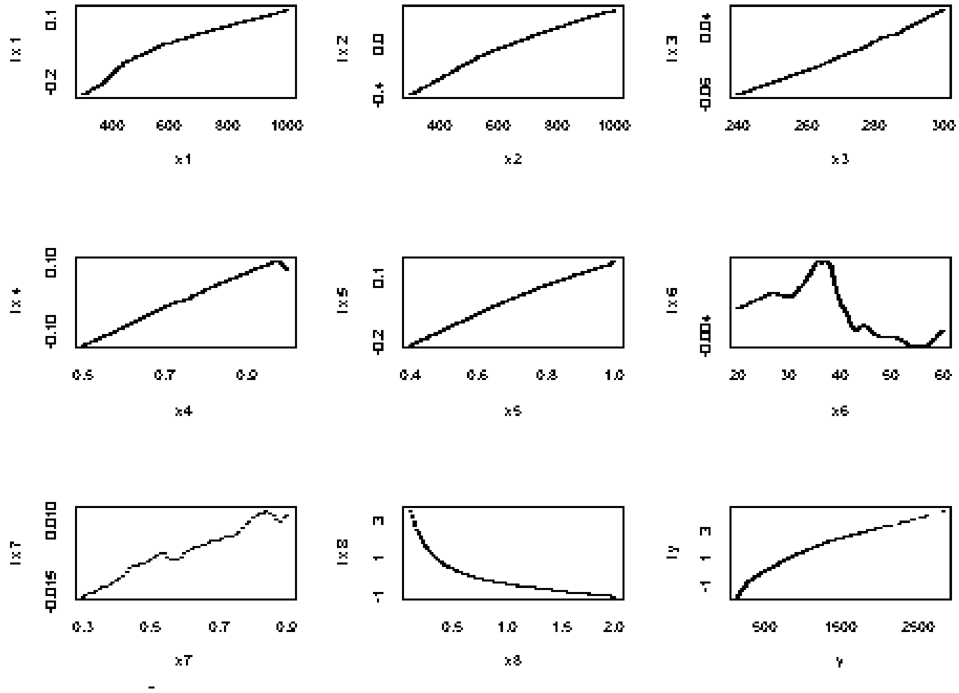


Figure 8-3-1: Plots of transformed variables against original data of U1

Figure 8-3-1 leads to the following conclusion: From the plot of variable x_8, which is flame spread rate R_f, it is clear that there is a change of behaviour at $x_8 = 0.5$ m/sec. There are significant mode changes in the curve of variable x_8, quick decrease and slow decrease. This corresponds to different modes of fire growth, and also the transformed value of variable x_6 is comparatively small thus can be neglected. This can be seen in the following section 8.3.1. It turns out that there are 512 data sets satisfying the constraint $R_f < 0.5$ m/sec, which we call U12, and 1988 data sets satisfying the constraint $R_f \geq 0.5$ m/sec, which we call U13.

8.3.1 ACE regression analysis of U12

Using the ACE algorithm on U12, the plots of transformed variables against original data are as shown in Figure 8-3-2. From analyzing Figure 8-3-2, it also turns out that x_6, the fuel density, can be ignored in the regression calculations for U12. So the set of indices used was just $I = 1, 2, 3, 4, 5, 7, 8$.

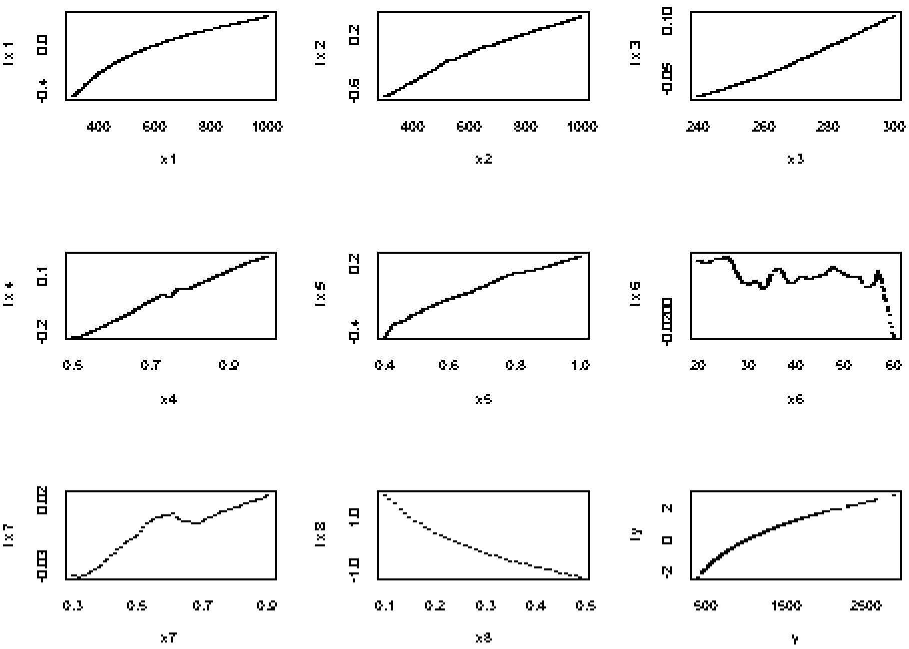


Figure 8-3-2: Plots of transformed variables against original data of U12

To the data set U12, we fitted a quadratic regression formula of the form

$$T_t = \sum_{i \in I} (a_i x_i^2 + b_i x_i) + c + \varepsilon . \qquad (8\text{-}3\text{-}1)$$

The coefficient $c =$ 3055.3, and the coefficients b_i and a_i were as in Table 8-3-1.

i	1	2	3	4	5	7	8
b_i	1.0401	0.9837	-15.53	390.5	745.0	-54.22	-10183
a_i	-0.000485	-0.000260	0.03158	-0.6815	-202.4	94.84	11840

Table 8-3-1: Values of quadratic regression coefficients of U12

Letting

$$T_t = \sum_{i \in I} (a_i x_i^2 + b_i x_i) + c \qquad (8\text{-}3\text{-}2)$$

it was found that the correlation between T and T_t was 0.9772.

A scatter plot of T against T_t is shown in Figure 8-3-3.

The second step in the fitting is to improve the fit of T_t by using a cubic regression formula of the form

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 + \varepsilon^* . \qquad (8\text{-}3\text{-}3)$$

The coefficients turned out to be:

C_0 = 175.3, C_1 = 0.9422, C_2 = -0.0003677, C_3 = 1.984e-007.

Letting

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 \qquad (8\text{-}3\text{-}4)$$

the correlation achieved between T and T_p is now 0.9926.

A scatter plot of T and T_p is shown in Figure 8-3- 4.

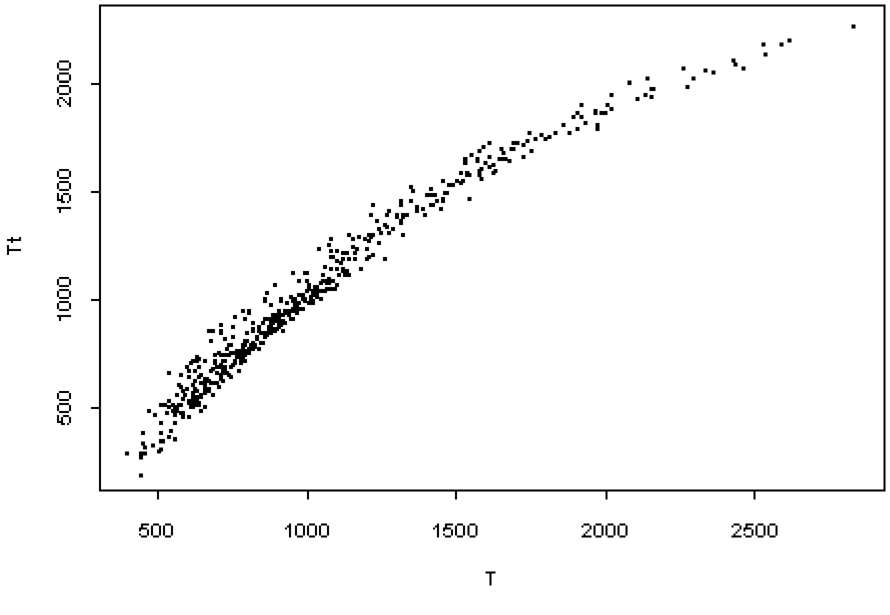


Figure 8-3-3: Scatter plot of T against T_t (quadratic) for U12 (x_6 is ignored)

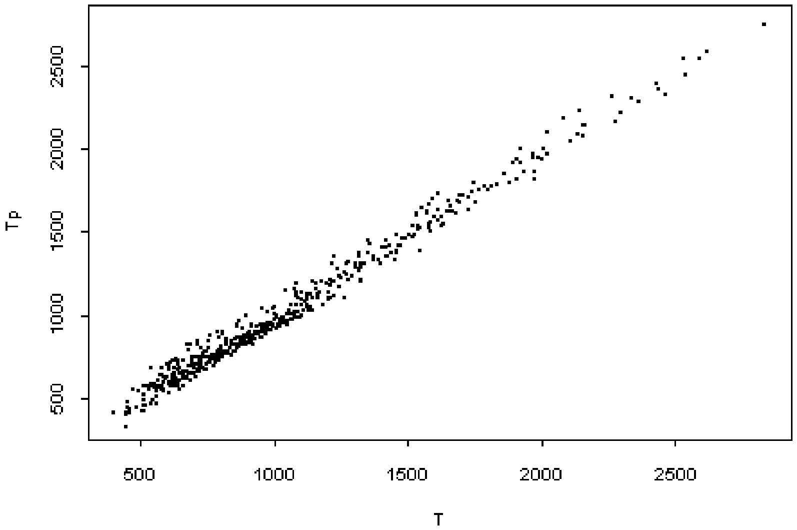


Figure 8-3-4: Scatter plot of *T* against T_p (cubic) for U12(x_6 is ignored)

8.3.2 ACE regression analysis of U13

Using the ACE algorithm on U13, the plots of transformed variables against original data are shown in Figure 8-3-5. It also turns out that the fuel density, x_6, can be ignored in the regression calculations in U13. So the set of indices used was just i = 1, 2, 3, 4, 5, 7, 8.

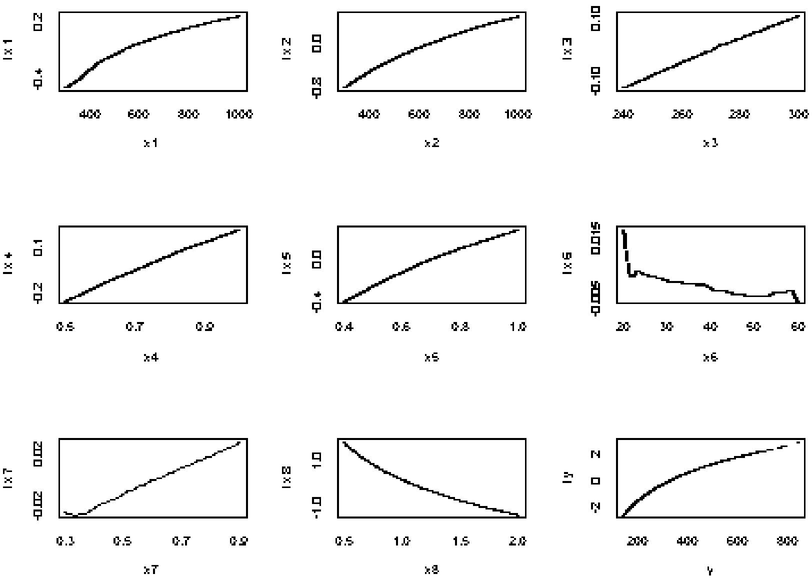


Figure 8-3-5: Transformed variables against original data for U13

To the data set U13, we fitted a quadratic regression formula of the form (8-3-1)

The coefficient $c =$ 30.81, and the coefficients b_i and a_i were as in Table 8-3-2.

i	1	2	3	4	5	7	8
b_i	0.2558	0.2725	2.050	197.1	198.3	-3.937	-678.6
a_i	-0.000105	-0.000060	-0.003031	-60.44	-49.73	14.25	184.0

Table 8-3-2:Values of quadratic regression fitted coefficients for U13

Letting

$$T_t = \sum_{i \in I} (a_i x_i^2 + b_i x_i) + c \qquad (8\text{-}3\text{-}5)$$

it was found that the correlation between T and T_t was 0.9811. The scatter plot is shown in Figure 8-3-6.

The second step in the fitting is to improve the fit of T_t by using a cubic regression formula of the form

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 + \varepsilon' . \qquad (8\text{-}3\text{-}6)$$

The coefficients turned out to be:

C_0 = 59.71, C_1 = 0.9165, C_2 = -0.001111, C_3 = 2.082e-006

Letting

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 . \qquad (8\text{-}3\text{-}7)$$

the correlation achieved between T and T_p is now 0.9940.

A scatter plot of T against T_p is shown in Figure 8-3-7.

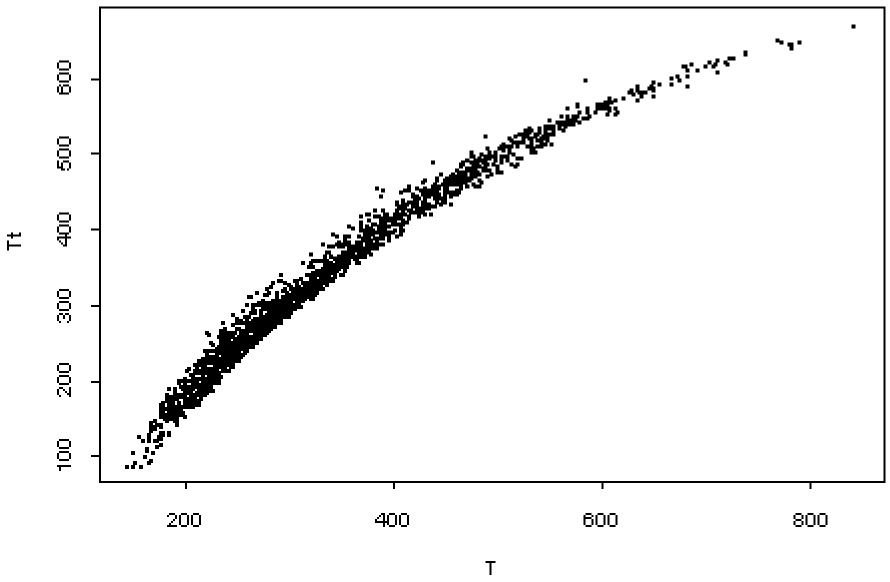


Figure 8-3-6: Scatter plot of T against T_t (quadratic fitted) for U13

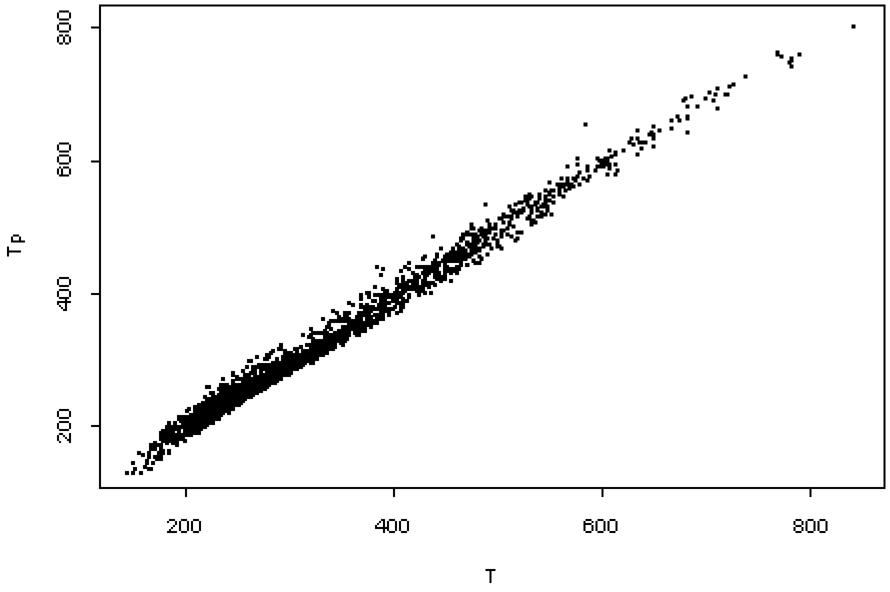


Figure 8-3-7: Scatter plot of T against T_p (cubic fitted) for U13

8.4 Modern regression analysis for DOWO scenario

Applying the ACE regression algorithm to the DOWO scenario, which is referred to as data U2, we obtain eight transformed inputs and a transformed output. Plots of the transformed data against the original data are shown in Figure 8-4-1.

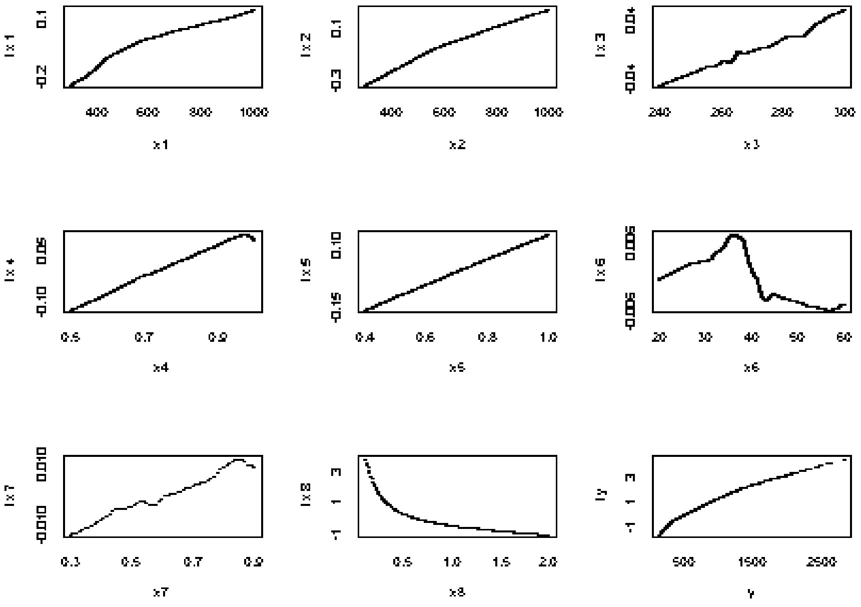


Figure 8-4-1: Plots transformed variables in ace against original data for U2

The plots lead to the following conclusion: From the plot of variable x_8, which is flame spread rate, it is clear that there is a change of behaviour at $x_8 = 0.5$ m/sec.

It turns out that there are 512 data sets satisfying the constraint $R_f < 0.5$ m/sec, which we call U22, and 1988 data sets satisfying the constraint $R_f \geq 0.5$ m/sec, which we call U23.

8.4.1 ACE regression analysis of U22

Using the ACE algorithm on U22, the plots of transformed variables against original data are shown in Figure 8-4-2. Analyzing Figure 8-4-2, it again turns out that the fuel density, x_6, can be ignored in the regression calculations in U22. So the set of indices used was just $i = 1, 2, 3, 4, 5, 7, 8$.

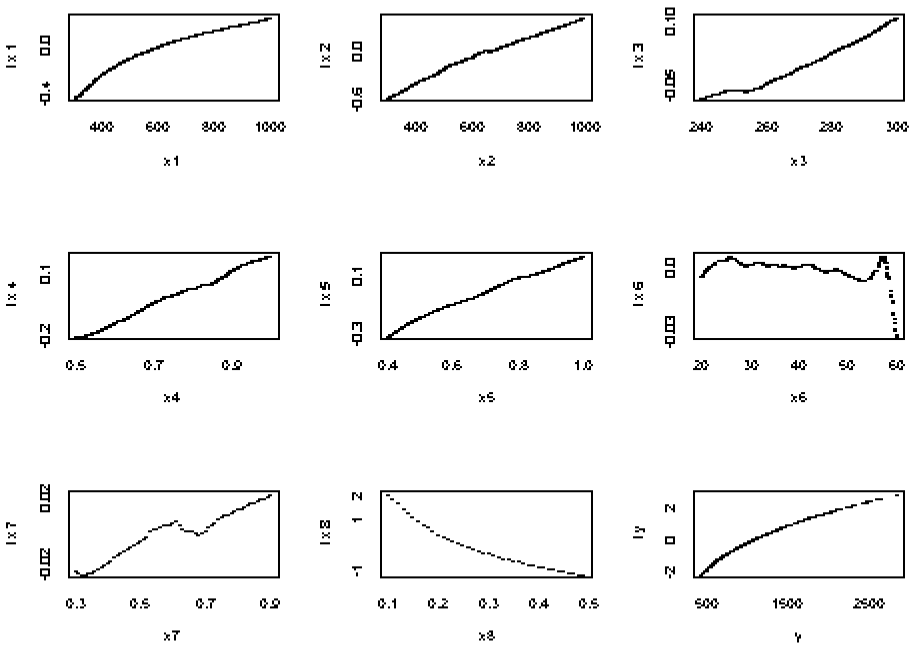


Figure 8-4-2: Plots of transformed variables against original data for U22

To the data set U22, we fitted a quadratic regression formula of the form (8-3-1)

The coefficient c = 3287.4, and the coefficients b_i and a_i were as in Table 8-4-1.

i	1	2	3	4	5	7	8
bi	1.094	0.8576	-14.38	320.8	469.2	-119.6	-10790
ai	-0.000525	-0.000190	0.02914	37.92	-50.83	145.4	12547

Table 8-4-1:Values of quadratic regression coefficients of U22 (x_6 is ignored)

Letting

$$T_t = \sum_{i \in I} (a_i x_i^2 + b_i x_i) + c \qquad (8\text{-}4\text{-}1)$$

it was found that the correlation between T and T_t was 0.9811. The scatter plot of T against T_t is shown in Figure 8-4-3.

The second step in the fitting is to improve the fit of T_t by using a cubic regression formula of the form

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 + \varepsilon'. \qquad (8\text{-}4\text{-}2)$$

The coefficients turned out to be:

C_0 = 158.2, C_1 = 0.9924, C_2 = -0.0003657, C_3 = 1.772e-007

Letting

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 \qquad (8\text{-}4\text{-}3)$$

the correlation achieved between T and T_p is now 0.9938.

A scatter plot of T against T_p is shown in Figure 8-4-4.

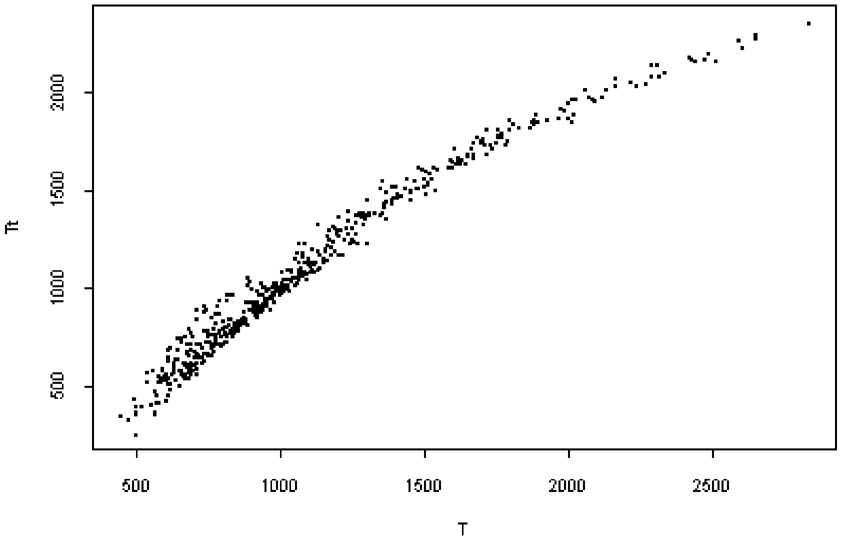


Figure 8-4-3: Scatter plot of T against T_t (quadratic) for U22

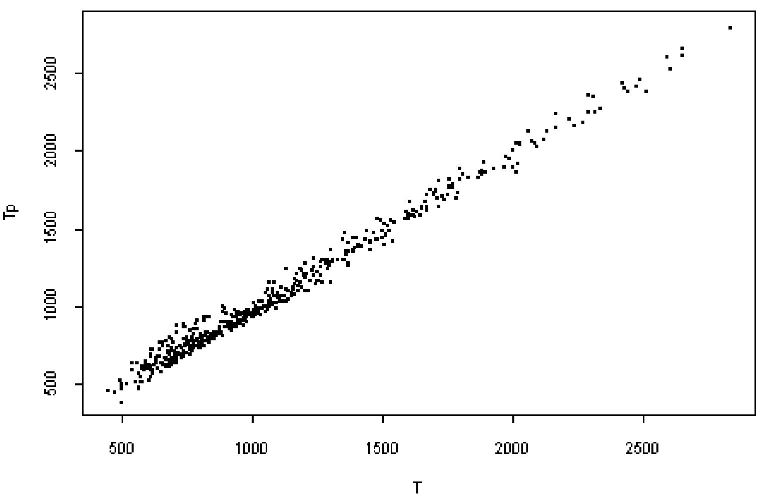


Figure 8-4-4: Scatter plot of T against T_p (cubic) for U22

8.4.2 ACE regression analysis of U23

Using the ACE algorithm on U23, the plots of transformed variables against original data are shown in Figure 8-4-5. From analyzing Figure 8-4-5, it again turns out that the fuel density, x_6, can be ignored in the regression calculations in U23. So the set of indices used was just $i = 1, 2, 3, 4, 5, 7, 8$.

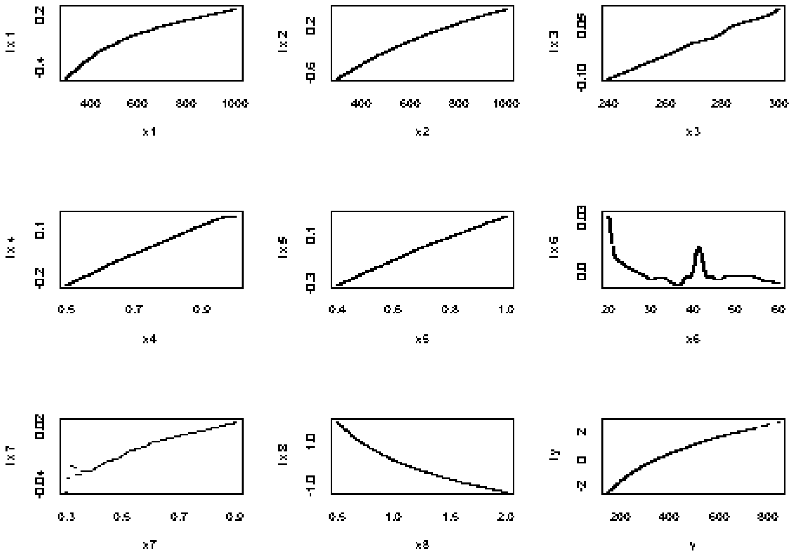


Figure 8-4-5: Plots of transformed variables in ACE against original data for U23

To the data set U23, we fitted a quadratic regression formula of the form (8-3-1).

The coefficient $c =$ 165.2, and the coefficients b_i and a_i were as in Table 8-4-2.

i	1	2	3	4	5	7	8
bi	0.2600	0.2482	1.766	192.4	123.9	-3.450	-719.9
ai	-0.000108	-0.000050	-0.002571	-59.48	-8.837	13.28	195.3

Table 8-4-2: Values of quadratic regression coefficients for U23

Letting

$$T_t = \sum_{i \in I} (a_i x_i^2 + b_i x_i) + c \qquad (8\text{-}4\text{-}4)$$

it was found that the correlation between T and T_t was 0.9843. A scatter plot of T against T_t is shown in Figure 8-4-6

The second step in the fitting is to improve the fit of T_t by using a cubic regression formula of the form

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 + \varepsilon' . \qquad (8\text{-}4\text{-}5)$$

The coefficients turned out to be:

C_0 = 45.49, C_1 = 1.043, C_2 = -0.001326, C_3 = 2.057e-006.

Letting

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 \qquad (8\text{-}4\text{-}6)$$

the correlation achieved between T and T_p is now 0.9950.

A scatter plot of T against T_p is shown in Figure 8-4-7.

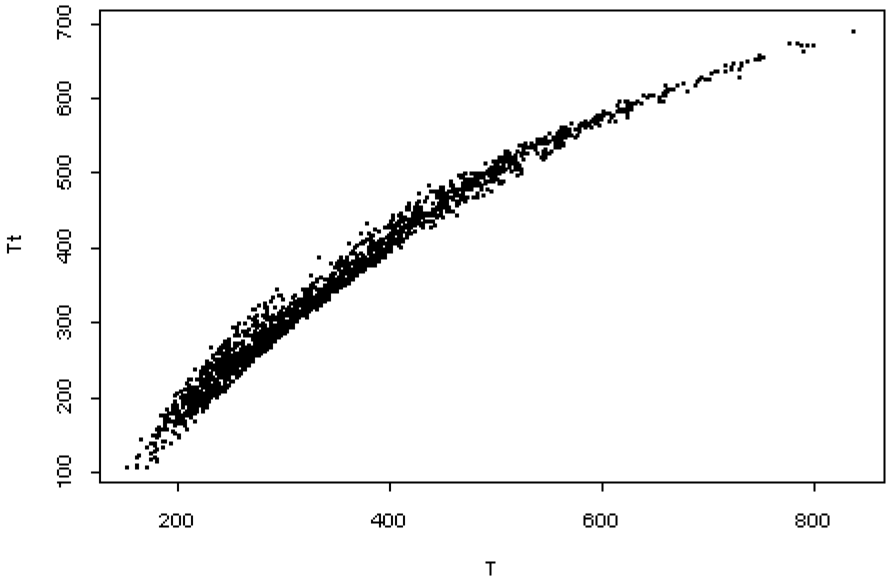


Figure 8-4-6: Scatter plot of T against T_t (quadratic fitted) for U23

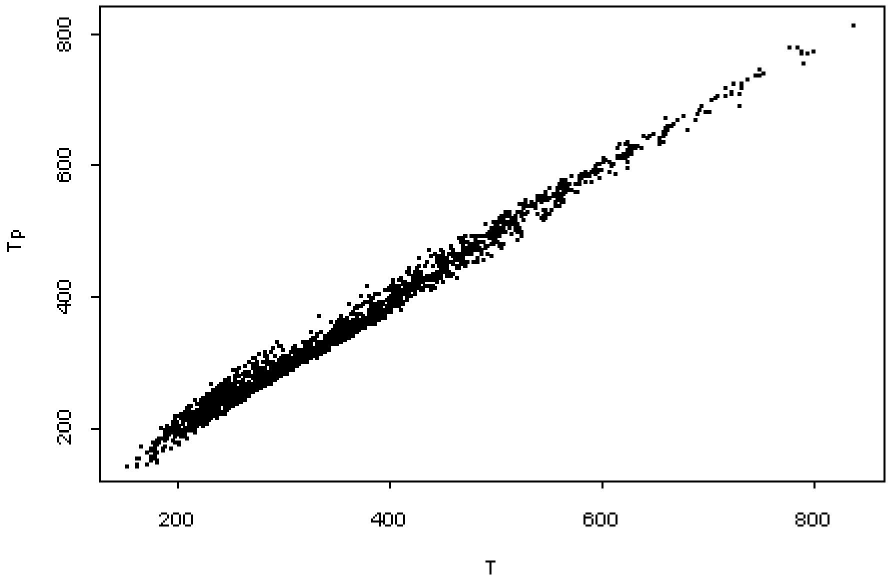


Figure 8-4-7: Scatter plot of T against of T_p (cubic fitted) for U23

8.5 Modern regression analysis for DCWC scenario

Applying the ACE regression algorithm to the DCWC, which is referred to as U3 data, we obtain eight transformed inputs and a transformed output. Plots of the transformed data against the original data are shown in Figure 8-5-1.

Figure 8-5-1 leads to the following conclusion:

From the plot of variable x_8, which is flame spread rate R_f, it is clear that there is a change of behaviour at a x_8 = 0.5 m/sec. It turns out that there are 512 data sets satisfying the constraint R_f < 0.5 m/sec, which we call U32, and 1988 data sets satisfying the constraint $R_f \geq$ 0.5 m/sec, which we call U33.

8.5.1 ACE regression analysis of U32

Using the ACE algorithm on U32, the plots of transformed variables against original data are shown in Figure 8-5-2. It turns out that x_4, x_5, x_6 (they are window width factor, window height factor and fuel density) can be ignored in the regression calculations in U32 (That x_4 and x_5 can be ignored should have been expected, since the window is closed). So the set of indices used was just i = 1, 2, 3, 7, 8.

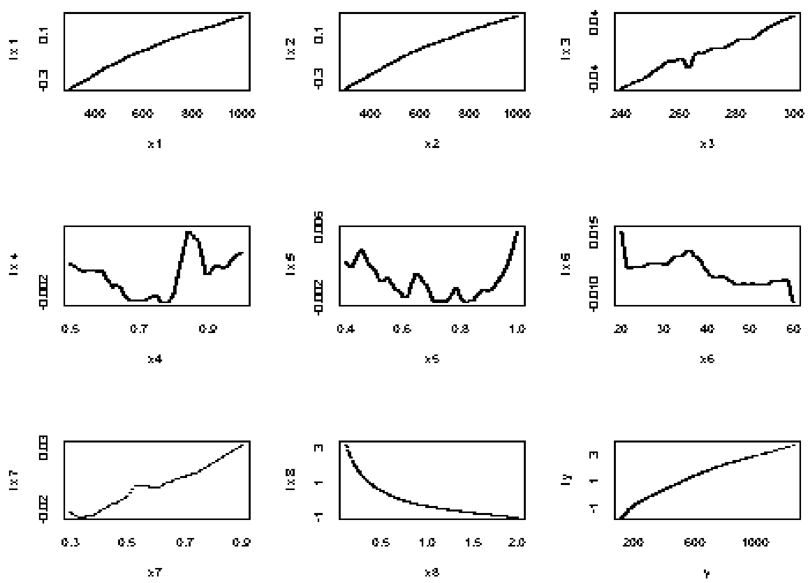


Figure 8-5-1: Plots of transformed variables against original data in ACE for U3

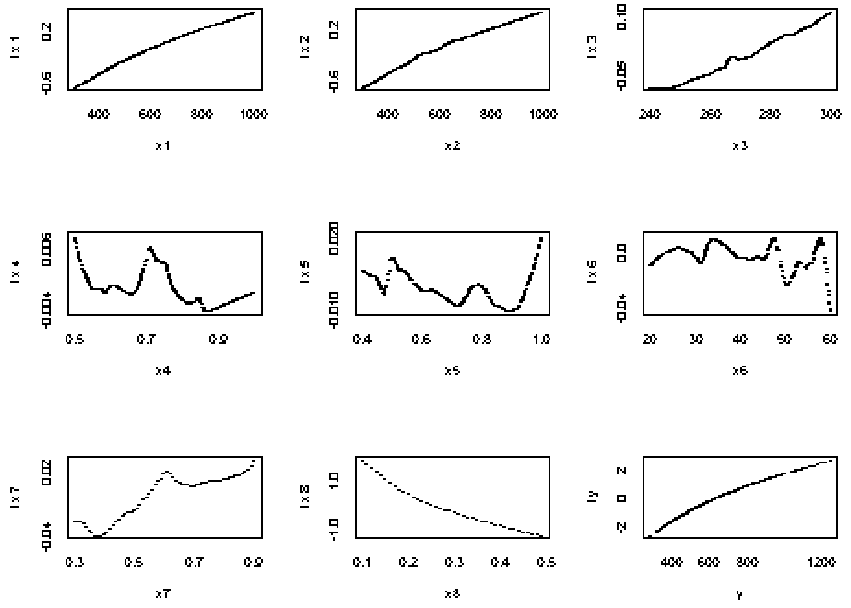


Figure 8-5-2: Plots of transformed variables against original data in ACE for U32

To the data set U32, we fitted a quadratic regression formula of the form

$$T_t = \sum_{i \in I} (a_i x_i^2 + b_i x_i) + c + \varepsilon . \qquad (8\text{-}5\text{-}1)$$

The coefficient c = 1163.5, and the coefficients b_i and a_i were as in Table 8-5-1.

i	1	2	3	7	8
b_i	0.5374	0.4909	-3.478	-1.801	-3647
c_i	-0.0001782	-0.0001420	0.007544	22.65	3889

Table 8-5-1: Values of quadratic regression coefficients for U32

Letting

$$T_t = \sum_{i \in I} (a_i x_i^2 + b_i x_i) + c \qquad (8\text{-}5\text{-}2)$$

it was found that the correlation between T and T_t was 0.9875. A scatter plot of T against T_t is shown in Figure 8-5-3.

The second step in the fitting is to improve the fit of T_t by using a cubic regression formula of the form

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 + \varepsilon' . \qquad (8\text{-}5\text{-}3)$$

The coefficients turned out to be:

C_0 = 22.17, C_1 = 1.326, C_2 = -0.001161, C_3 = 8.565e-007

Letting

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 \qquad (8\text{-}5\text{-}4)$$

the correlation achieved between T and T_p is now 0.9958.

A scatter plot of T against T_p is shown in Figure 8-5-4.

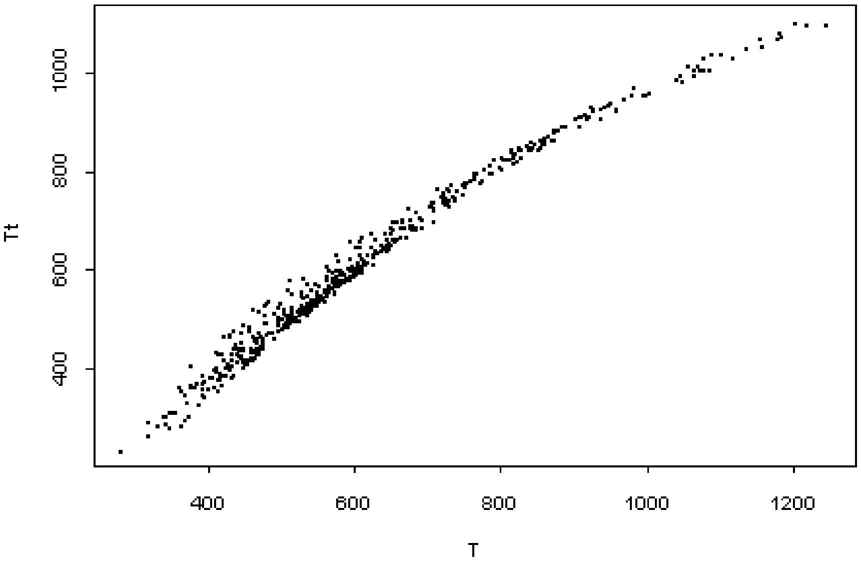


Figure 8-5-3: Scatter plot of T against T_t (quadratic fitted) for U32

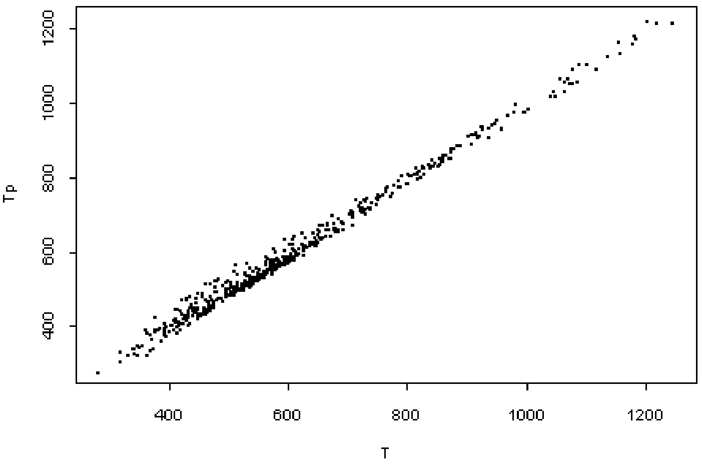


Figure 8-5-4: Scatter plot of T against T_p (cubic fitted) for U32

8.5.2 ACE regression analysis of U33 ($R_f \geq 0.5$ m/sec)

Using the ACE algorithm on U33, the plots of transformed variables against original data are shown in Figure 8-5-5. From Figure 8-5-5, it also turns out that x_4, x_5, x_6 (they are window width factor, window height factor and fuel density) can be ignored in the regression calculations in U33. So the set of indices used was just i = 1, 2, 3, 7, 8.

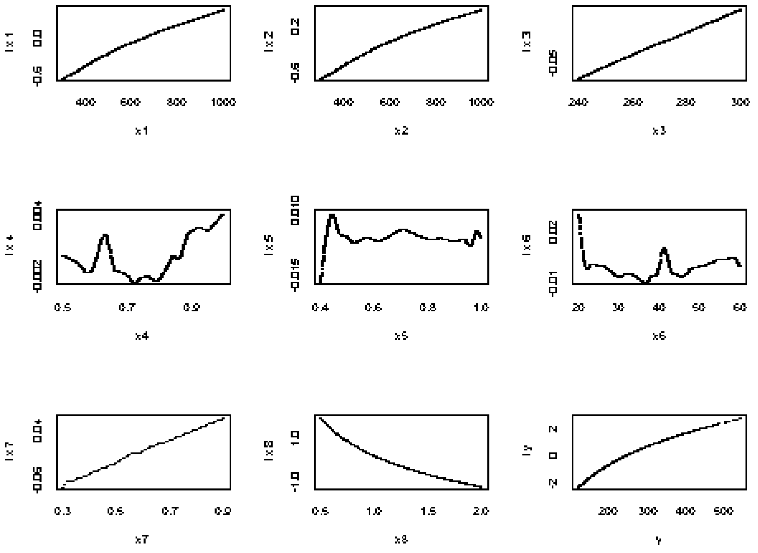


Figure 8-5-5: Plots of transformed variables against original data for U33

To the data set U33, we fitted a quadratic regression formula of the form (8-5-1)

The coefficient $c = 231.3$, and the coefficients b_i and a_i were as in Table 8-5-2.

i	1	2	3	7	8
b_i	0.1458	0.1812	1.005	3.184	-436.4
c_i	-0.00002680	-0.00004694	-0.001488	12.42	118.3

Table 8-5-2: Values of quadratic regression coefficients for U33

Letting

$$T_t = \sum_{i \in I} (a_i x_i^2 + b_i x_i) + c \tag{8-5-5}$$

it was found that the correlation between T and T_t was 0.9861. A scatter plot of T against T_t is shown in Figure 8-5-6.

The second step in the fitting is to improve the fit of T_t by using a cubic regression formula of the form

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 + \varepsilon^{'} . \tag{8-5-6}$$

The coefficients turned out to be:

$C_0 = 19.47$, $C_1 = 1.208$, $C_2 = -0.002617$, $C_3 = 5.326\text{e-}006$

Letting

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 \tag{8-5-7}$$

the correlation achieved between T and T_p is now 0.9959.

A scatter plot of T against T_p is shown in Figure 8-5-7.

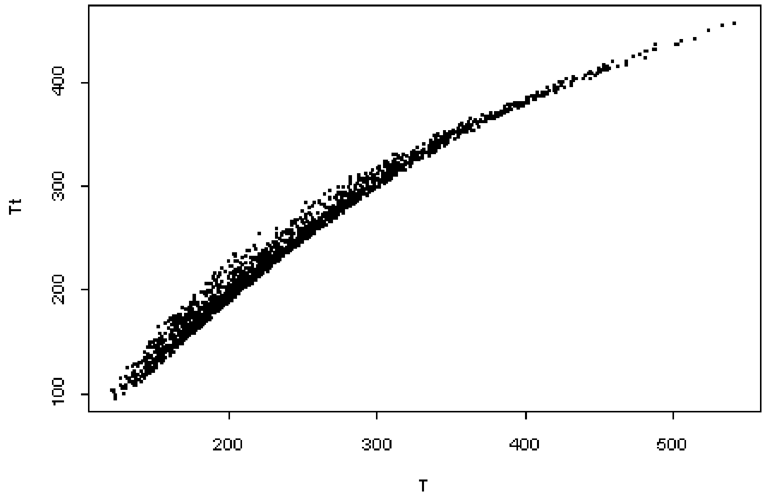


Figure 8-5-6: Scatter plot of T against T_t (quadratic fitted) for U33

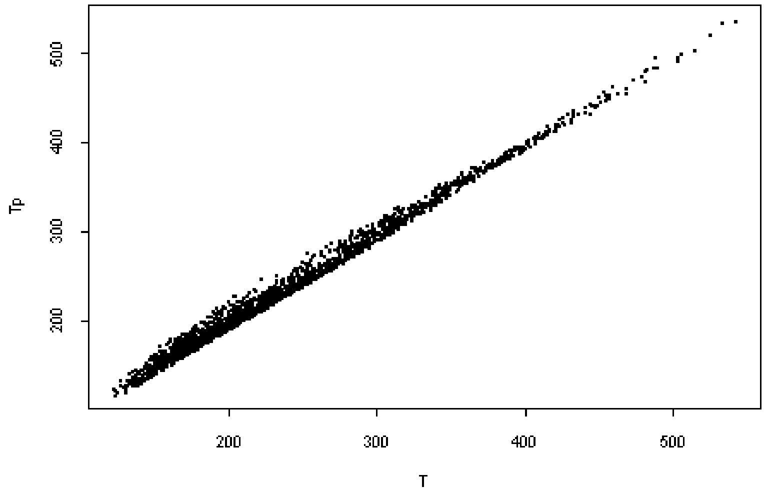


Figure 8-5-7: Scatter plot of T against T_p (cubic) for U33

8.6 Modern regression analysis for DOWC scenario

Applying the ACE regression algorithm to the DOWC scenario, which is referred to as data U4, we obtain eight transformed inputs and a transformed output. Plots of the transformed data against the original data are shown in Figure 8-6-1.

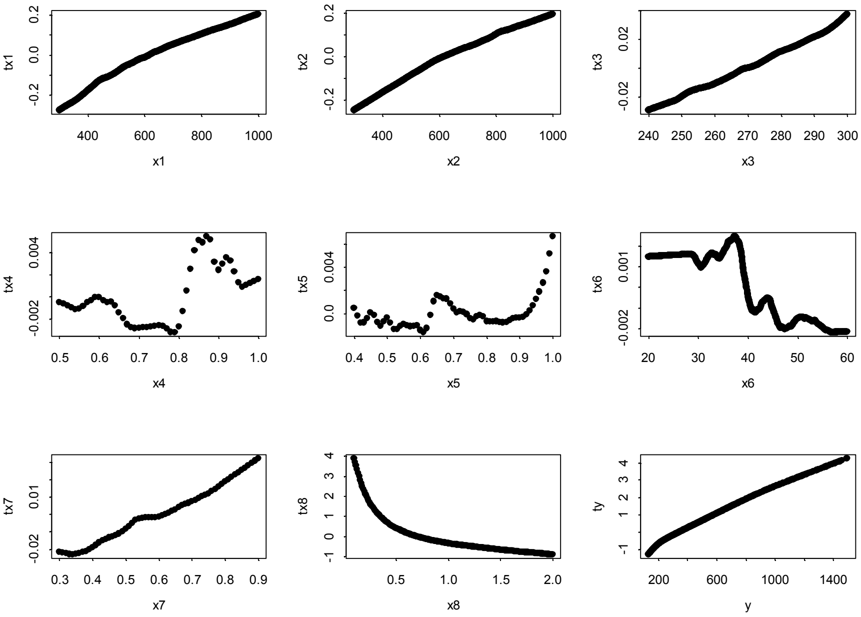


Figure 8-6-1: Plots of transformed variables against original data for U4

The plots lead to the following conclusion: From the plot of variable x_8, which is flame spread rate R_f, it is clear that there is a change of behaviour at $x_8 = 0.5$ m/sec. It turns out that there are 512 data sets satisfying the constraints $R_f < 0.5$ m/sec, which we call U42, and 1988 data sets satisfying the constraint $R_f \geq 0.5$ m/sec, which we call U43.

8.6.1 ACE regression analysis of U42

Using the ACE algorithm on U42, the plots of transformed variables against original data are shown in Figure 8-6-2. It also turns out that x_4, x_5, x_6 (they are window width factor, window height factor and fuel density) can be ignored in the regression calculations in U42. So the set of indices used was just $i = 1, 2, 3, 7, 8$.

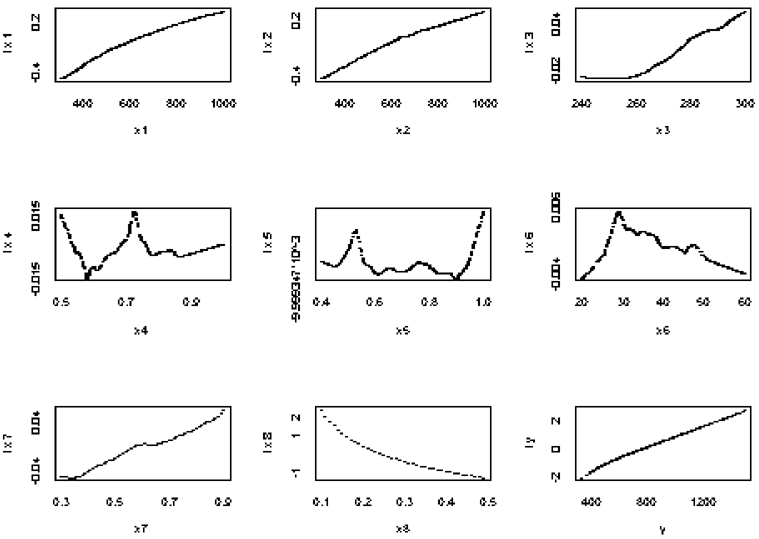


Figure 8-6-2: Plots of transformed variables against original data for U42

To the data set U42, we fitted a quadratic regression formula of the form (8-5-1) The coefficient c = 1875.2, and the coefficients b_i and a_i were as in Table 8-6-1.

i	1	2	3	7	8
b_i	0.6733	0.4759	-4.768	-33.10	-5826
a_i	-0.0002828	-0.0001694	0.009414	71.86	6403

Table 8-6-1: Values of quadratic regression coefficients for U42

Letting

$$T_t = \sum_{i \in I} (a_i x_i^2 + b_i x_i) + c \qquad (8\text{-}6\text{-}1)$$

it was found that the correlation between T and T_t was 0.9924. A scatter plot of T against T_t is shown in Figure 8-6-3.

The second step in the fitting is to improve the fit of T_t by using a cubic regression formula of the form

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 + \varepsilon' . \qquad (8\text{-}6\text{-}2)$$

The coefficients turned out to be:

C_0 = 19.84, C_1 = 1.166, C_2 = -0.0005202, C_3 = 3.135e-007.

Letting

$$T_p = C_0 + C_1T_t + C_2T_t^2 + C_3T_t^3 \qquad (8\text{-}6\text{-}3)$$

the correlation achieved between T and T_p is now 0.9954.

A scatter plot of T against T_p is shown in Figure 8-6-4.

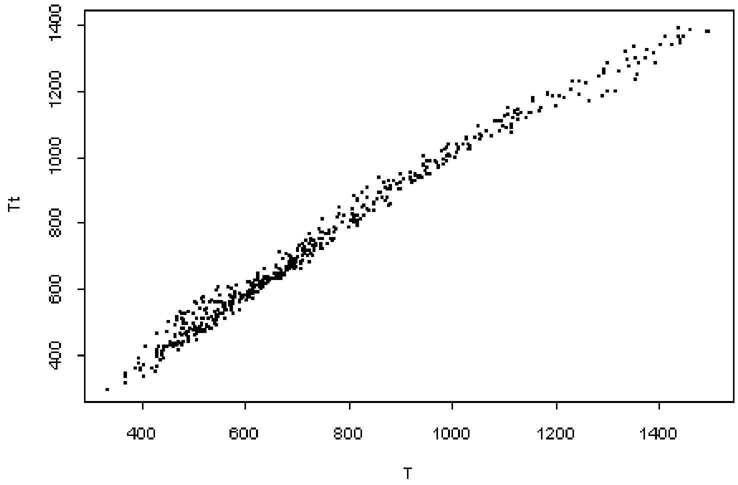


Figure 8-6-3: Scatter plot of T against T_t (quadratic fitted) for U42

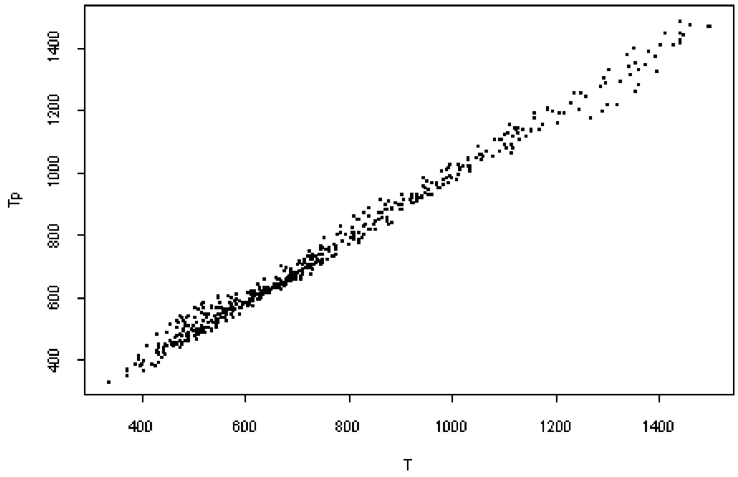


Figure 8-6-4: Scatter plot of T against T_p (cubic) for U42

8.6.2 ACE regression analysis of U43

Using the ACE algorithm on U43, the plots of transformed variables against original data are shown in Figure 8-6-5. Further analyzing Figure 8-6-5, it again turns out that x_4, x_5, x_6 (they are window width factor, window height factor and fuel density) can be ignored in the regression calculations in U43. So the set of indices used was just i = 1, 2, 3, 7, 8.

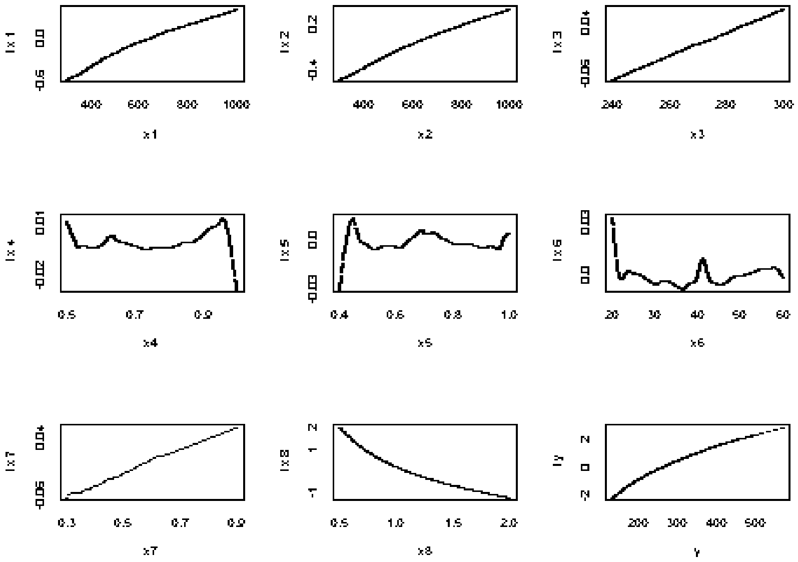


Figure 8-6-5: Plots of transformed variables against original data for U43

To the data set U43, we fitted a quadratic regression formula of the form (8-5-1)

The coefficient c = 301.3, and the coefficients b_i and a_i were as in Table 8-6-2.

i	1	2	3	7	8
b_i	0.1665	0.1454	1.038	5.446	-491.9
a_i	-0.00004267	-0.00002992	-0.001600	9.949	133.4

Table 8-6-2: Values of quadratic regression coefficients for U43

Letting

$$T_t = \sum_{i \in I} (a_i x_i^2 + b_i x_i) + c \qquad (8\text{-}6\text{-}4)$$

it was found that the correlation between T and T_t was 0.9886.

A scatter plot of T against T_t is shown in Figure 8-6-6.

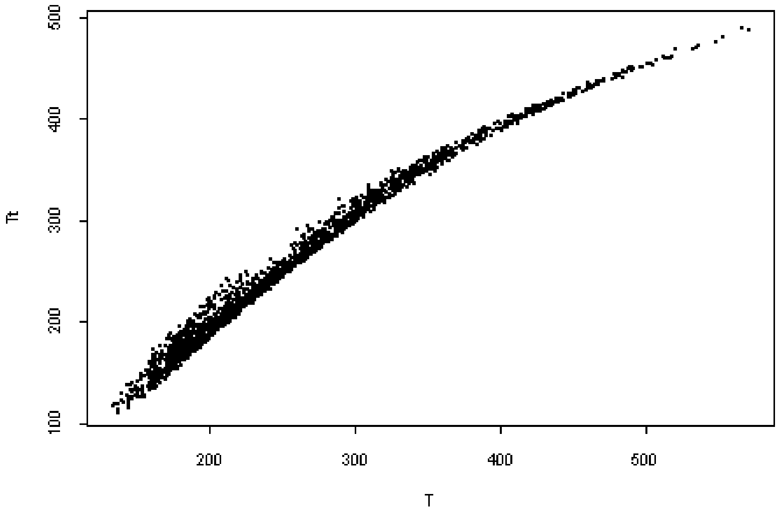


Figure 8-6-6: Scatter plot of $\bar{T}$ against T_t (quadratic fitted) for U43

The second step in the fitting is to improve the fit of T_t by using a cubic regression formula of the form

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 + \varepsilon' . \qquad (8\text{-}6\text{-}5)$$

The coefficients turned out to be:

C_0 = 1.485, C_1 = 1.393, C_2 = -0.002934, C_3 = 4.963e-006.

Letting

$$T_p = C_0 + C_1 T_t + C_2 T_t^2 + C_3 T_t^3 \qquad (8\text{-}6\text{-}6)$$

the correlation achieved between T and T_p is now 0.9963.

A scatter plot of T against T_p is shown in Figure 8-6-7.

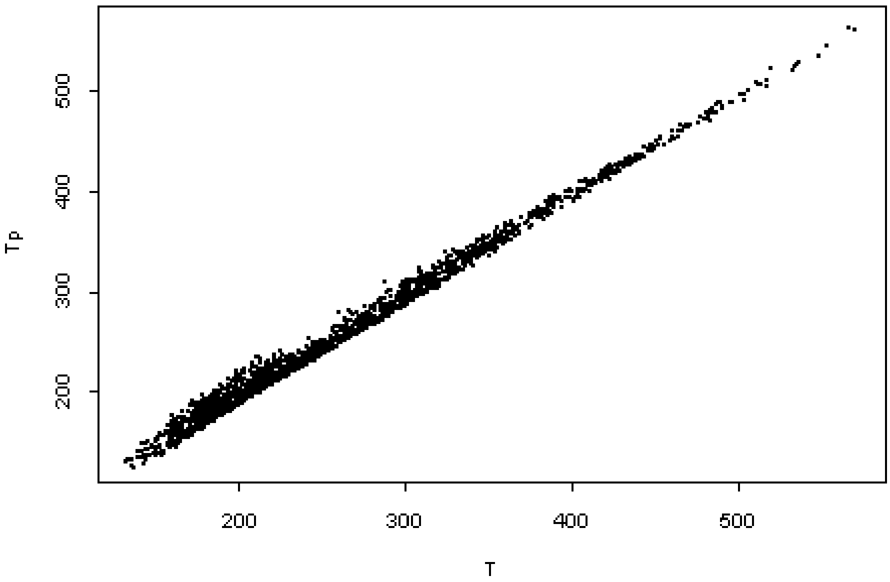


Figure 8-6-7: Scatter plot of T against T_p (cubic fitted) for U43

8.7 Summary of the events and result of correlation

The ACE regression analysis for time to untenable conditions can be summarized as follows in Figure 8-7-1.

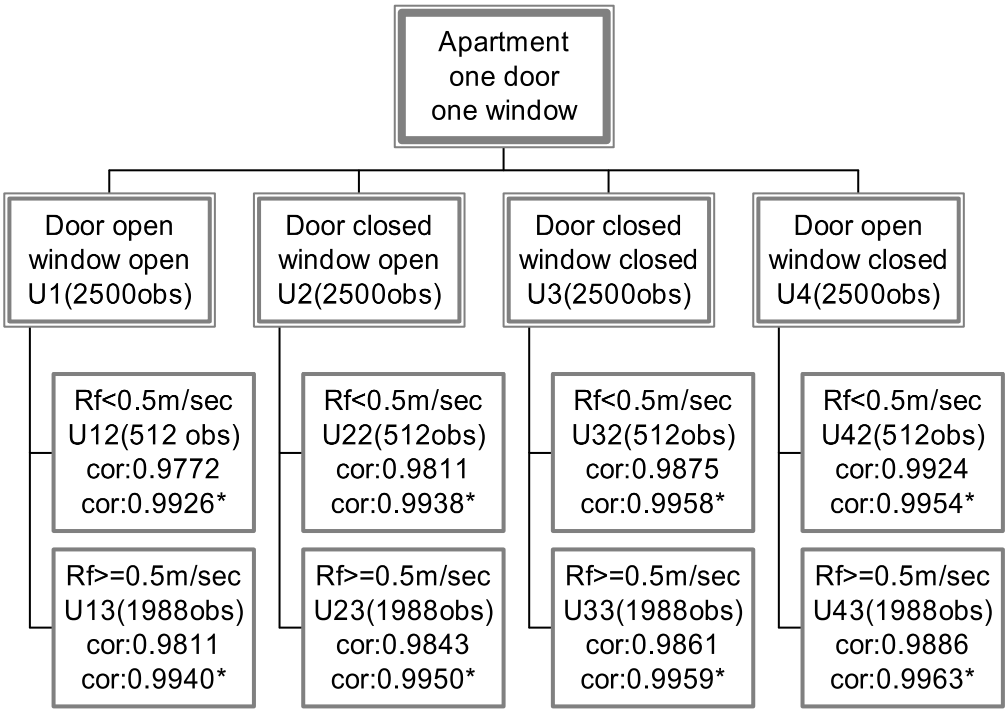

Apartment one door one window
Door open window open U1(2500obs)
Door closed window open U2(2500obs)
Door closed window closed U3(2500obs)
Door open window closed U4(2500obs)
Rf<0.5m/sec U12(512 obs) cor:0.9772 cor:0.9926*
Rf>=0.5m/sec U13(1988obs) cor:0.9811 cor:0.9940*
Rf<0.5m/sec U22(512obs) cor:0.9811 cor:0.9938*
Rf>=0.5m/sec U23(1988obs) cor:0.9843 cor:0.9950*
Rf<0.5m/sec U32(512obs) cor:0.9875 cor:0.9958*
Rf>=0.5m/sec U33(1988obs) cor:0.9861 cor:0.9959*
Rf<0.5m/sec U42(512obs) cor:0.9924 cor:0.9954*
Rf>=0.5m/sec U43(1988obs) cor:0.9886 cor:0.9963*

Figure 8-7-1: Events and result of correlation for time to untenable condition

In Figure 8-7-1:

1. Fuel density, x_6 is ignored in the regression analysis of data sets U12, U13, U22 and U23;
2. Window width factor, window height factor and fuel density are ignored (they are denoted by x_4, x_5 and x_6) in the regression analysis of data sets U32, U33, U42 and U43;
3. For correlation values marked with asterisk in Figure 8-7-1 a cubic was fitted to quadratic formula predicted values.

CHAPTER 9
RELIABILITY ANALYSIS IN DESIGN FOR TIME TO UNTENABLE CONDITIONS

In this chapter, the reliability index for a specific example for time to untenable conditions for four scenarios will be calculated. The corresponding probability of failure for each of the scenarios will be obtained by the use of First Order Second Moment Method and the results validated by Monte – Carlo simulation.

9.1 Reliability analysis of the DCWO scenario

9.1.1 Reliability index for engineering design in DCWO scenario

Consider the DCWO scenario, when the flame spread rate $R_f \geq 0.5$ m/sec (U13). For simplicity, we shall take just two input variables to be random: room width W_r, which is denoted by x_2 and flame spread rate R_f, which is denoted by x_8. The other input variables will be taken to be constant. However, we saw in Chapter 8 that x_6 does not appear in the regression equation as long as $R_f > 0.5$ m/sec in this scenario. Let N(μ, σ) denote a normal random variable with mean μ, and standard deviation σ.

The assumed values are as follows:

$L = 600$ cm, $W_r = $ N(450, 20), $H_r = 250$ cm, $f_W = 0.7$, $f_H = 0.6$, $\rho_f = 50$, $f_A = 0.6$,
$R_f = N(0.7, 0.1)$.

The limiting state is taken to be T_t = 311.1 (time to untenable conditions given T_p = 300 seconds). This limiting state is chosen to demonstrate the methodology to get the probability of failure defined as the probability of exceeding a given value. It is needed to determine the probability of reaching untenable conditions which could cause danger to occupants. Another probability of failure would be the result if with a different limiting value. After introducing the given data into the formula developed in Chapter 8, the standardized regression equation for the two variable problem is reduced to

$$T_t = -0.02357u_2^2 + 4.390u_2 + 1.8396u_8^2 - 42.10u_8 + 407.5\,.$$

Through Lagrange's method of undetermined multipliers λ, it is not difficult to derive the values of λ as:

λ: 0.008282 -0.02788 -0.7837 6.462

and the corresponding values of the reliability indices β are:

β: 144.0 44.22 20.12 2.559

The smallest value $\beta = 2.559$ is the reliability index.

In this scenario, from equation (7-1-8), the approximate value of the probability of failure is:

$p_F = \Phi(-2.559) = 0.00526.$

Illustration of the result is given in Figure 9-1-1, and Figure 9-1-2.

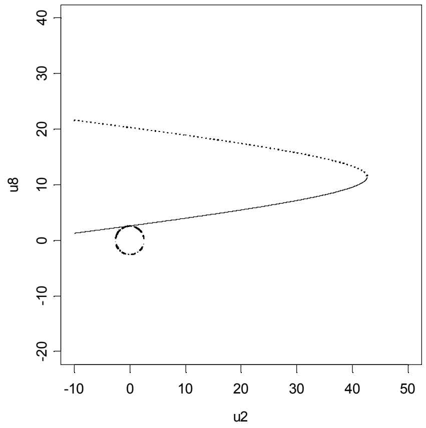


Figure 9-1-1: Illustration of reliability index for the numerical example U13

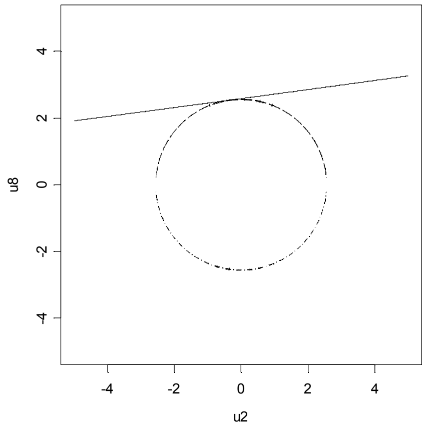


Figure 9-1-2: Illustration of reliability index for the numerical example U13

(Enlarged central part)

9.1.2 Validation by Monte – Carlo simulation

The probability just given can be validated by Monte – Carlo simulation: Out of 100000 simulations, 518 observations fell in the failure region $T_t < 311.1$ ($T_p < 300$ seconds), giving an estimated probability of failure 0.00518. The histogram of the distribution is also shown in Figure 9-1-3. The 95% confidence interval for the estimate is (0.00474, 0.00562), which contains the value (0.00526) obtained by using reliability index in the previous section.

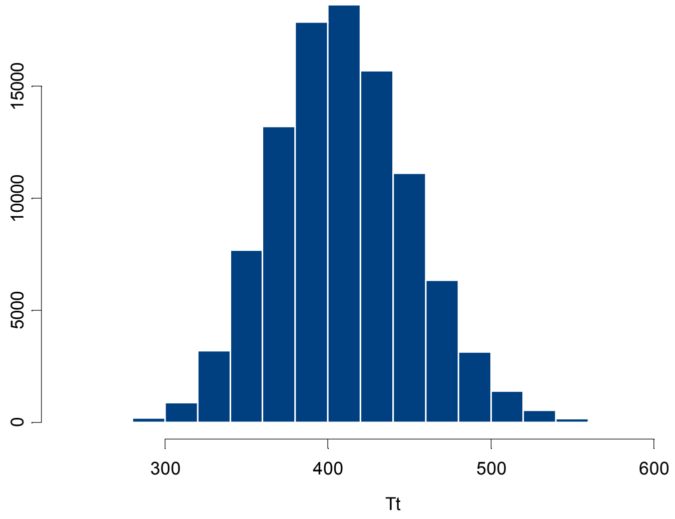


Figure 9-1-3: Histogram of U13 T_t in Monte –Carlo simulation

9.2 Reliability analysis of the DOWO scenario

9.2.1 Reliability index for engineering design in DOWO scenario

Consider the DOWO scenario, when flame spread rate $R_f \geq 0.5$ m/sec (U23). For simplicity, we shall take as before just two input variables to be random: room width W_r, which is denoted by x_2 and the flame spread rate R_f, which is denoted by x_8. The other input variables will be taken to be constant. However, we saw in Chapter 8 that x_6 does not appear in the regression equation as long as $R_f \geq 0.5$ m/sec in this scenario. Let N(μ, σ) denote a normal random variable with mean μ, and standard deviation σ.

The assumed values are as follows:

$L = 600$ cm, $W_\mathrm{r} = \mathrm{N}(450, 20)$, $H_\mathrm{r} = 250$ cm, $f_W = 0.7$, $f_\mathrm{H} = 0.6$, $\rho_f = 50$, $f_\mathrm{A} = 0.6$,
$R_f = N(0.7, 0.1)$

The limiting state is taken to be T_t = 330.9 (T_p = 320 seconds). After bringing the given data into the formula developed in Chapter 8, we get the following reduced standardized regression equation:

$$T_t = -0.01958u_2^2 + 4.083u_2 + 1.953u_8^2 - 44.64u_8 + 435.9\,.$$

Through using Lagrange's method of undetermined multipliers described in equation (7-1-4) to (7-1-7), it is not difficult to derive the values of λ as:

λ: 0.006893 -0.02331 -0.8319 6.549

and the corresponding values of β as:

β: 161.3 48.98 20.047 2.644

It is clear that the smallest value of β = 2.644, is the required reliability index.

From equation (7-1-8), the approximate value of the probability of failure in this case is

$$p_F = \Phi(-2.644) = 0.00410.$$

Illustration of the result is given in Figure 9-2-1 and Figure 9-2-2.

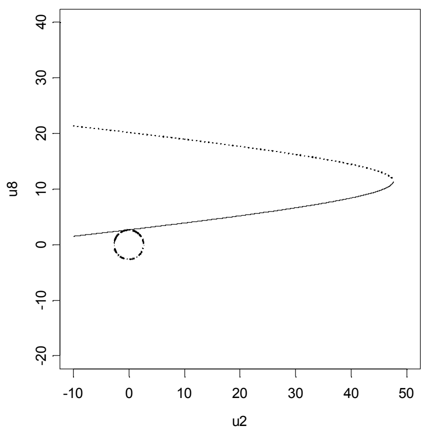


Figure 9-2-1: Illustration of index for the numerical example U23

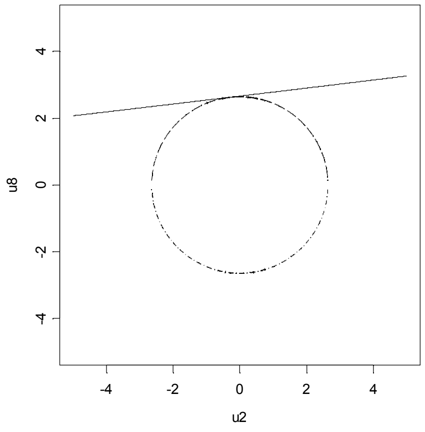


Figure 9-2-2: Illustration of index for the numerical example U23

(Enlarged central part)

9.2.2 Validation by Monte – Carlo simulation

The probability just given can be validated by Monte – Carlo simulation: Out of 100000 simulations, 419 observations fell in the failure region $T_t < 330.9$ ($T_p = 320$ seconds), giving an estimated probability of failure 0.00419. The histogram of the distribution is also shown in Figure 9-2-3. The 95% confidence interval for the estimate is (0.00379, 0.00459) , which contains the value (0.00410) obtained by using reliability index in the previous section.

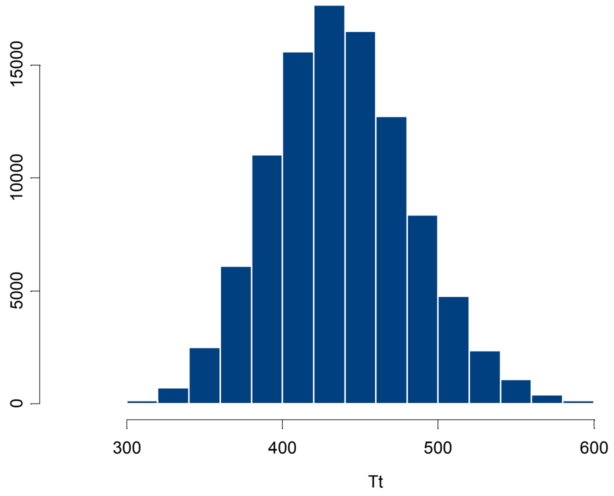


Figure 9-2-3: Histogram of U23 T_t in Monte – Carlo simulation

9.3 Reliability analysis of the DCWC scenario

9.3.1 Reliability index for engineering design in DCWC scenario

Consider the DCWC scenario, when flame spread rate $R_f \geq 0.5$ m/sec (U33). For simplicity, we shall take as before just two input variables to be random: room width W_r, denoted by x_2 and flame spread rate R_f, denoted by x_8. The other input variables will be taken to be constant. However, we saw in Chapter 8 that x_4 , x_5 and x_6 do not appear in the regression equation as long as $R_f \geq 0.5$ m/sec in this scenario. Let N(μ, σ) denote a normal random variable with mean μ, and standard deviation σ.

The assumed values are as follows:

$L = 600$, $W_\mathrm{r} = N(450, 20)$ cm, $H_\mathrm{r} = 250$ cm, $f_W = 0.7$, $f_\mathrm{H} = 0.6$, $\rho_f = 50$, $f_\mathrm{A} = 0.6$,
$R_f = N(0.7, 0.1)$

The limiting state is taken to be $T_t = 237.4$ ($T_p = 230$ seconds). After introducing the given data into the formula developed in Chapter 8, we get the following reduced standardized regression equation:

$$T_t = -0.01878u_2^2 + 2.780u_2 + 1.183u_8^2 - 27.07u_8 + 298.3\,.$$

Through using Lagrange's method of undetermined multipliers λ described in equation (7-1-4) to (7-1-7), it is not difficult to derive the values of λ as:

λ:	0.0043	-0.0078	-0.5064	4.263

and the corresponding values of β are:

β:	96.43	53.52	20.18	2.507.

It is clear that the smallest value of $\beta = 2.507$, is the required reliability index.

Thus, from equation (7-1-8), the approximate probability of failure in this case is:

$$p_F = \Phi(-2.507) = 0.00608.$$

Illustration of the result is given in Figure 9-3-1 and Figure 9-3-2.

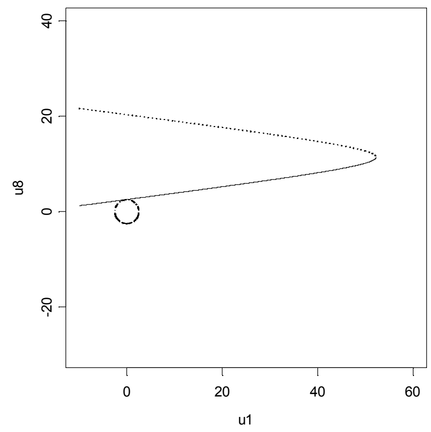


Figure 9-3-1: Illustration of the index for the numerical example of U33

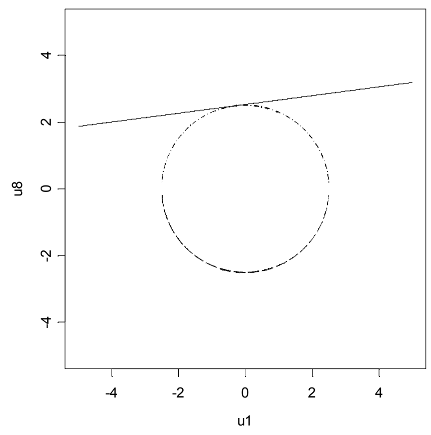


Figure 9-3-2: Illustration of the index for the numerical example of U33 (Enlarged central part)

9.3.2 Validation by Monte – Carlo simulation

The probability just given can be validated by Monte – Carlo simulation: Out of 100000 simulations, 591 observations fell in the failure region $T_t < 237.4$ ($T_p = 230$ seconds), giving an estimated probability of failure 0.00591. The histogram is illustrated in Figure 9-3-3. The 95% confidence interval for the estimate is (0.00543, 0.00639), which contains the value (0.00608) obtained by using reliability index in the previous section.

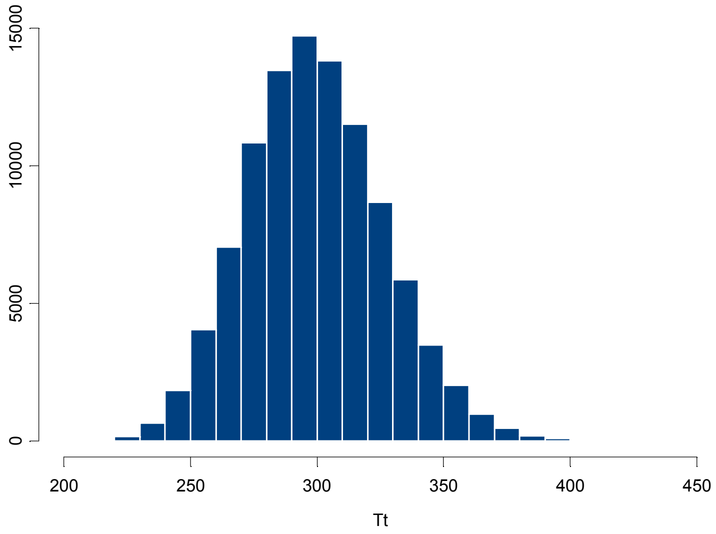


Figure 9-3-3: Histogram of U33 T_t in Monte – Carlo simulation

9.4 Reliability analysis of the DOWC scenario

9.4.1 Reliability index for engineering design in DOWC scenario

Consider the DOWC scenario, when flame spread rate $R_f \geq 0.5$ m/sec (U43). For simplicity, we shall take as before just two input variables to be random: room width W_r, denoted by x_2 and flame spread rate R_f, denoted by x_8. The other input variables will be taken to be constant. However, we saw in Chapter 8 that x_4, x_5 and x_6 do not appear in the regression equation as long as $R_f \geq 0.5$ m/sec in this scenario. Let $N(\mu, \sigma)$ denote a normal random variable with mean μ, and standard deviation σ.

The assumed values are as follows:

L = 600 cm, W_r = N(450, 20) cm, H_r = 250 cm, f_W = 0.7, f_H = 0.6, ρ_f = 50, f_A = 0.6, R_f = N(0.7, 0.1)

The limiting state is taken to be T_t = 257.2 (T_p = 250 seconds). After bringing the given data into the formula developed in Chapter 8, we get the reduced standardized regression equation as:

$$T_t = -0.01197u_2^2 + 2.369u_2 + 1.334u_8^2 - 30.52u_8 + 332.5\,.$$

Through using Lagrange's method of undetermined multipliers λ described in equation (7-1-4) to (7-1-7), it is not difficult to derive the values of λ as:

λ: 0.0034 -0.0077 -0.5652 4.146

and the corresponding values of β are:

β: 138.2 61.33 19.97 2.800.

It is clear that the smallest value of β=-2.800, is the required reliability index.

Therefore, from equation (7-1-8), the approximate probability of failure in this case is:

$$p_F = \Phi(-2.800) = 0.00255.$$

Illustration of the result is given in Figure 9-4-1 and Figure 9-4-2.

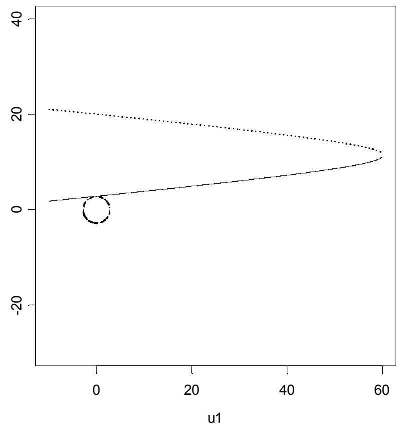


Figure 9-4-1: Illustration of β index for the numerical example of U43

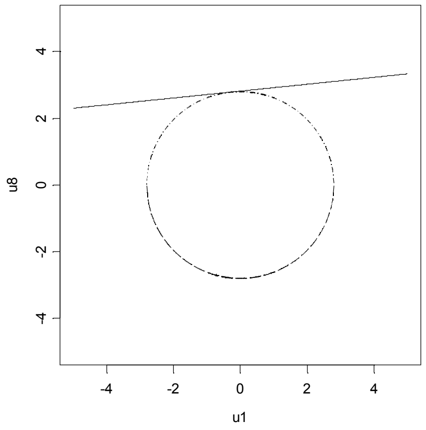


Figure 9-4-2: Illustration of β index for the numerical example of U43 (Enlarged central part)

9.4.2 Validation by Monte – Carlo simulation

The probability just given can be validated by Monte – Carlo simulation: Out of 100000 simulations, 275 observations fell in the failure region $T_t < 257.2$ ($T_p = 250$ seconds), given an estimated probability of failure 0.00275. The histogram of the distribution is illustrated in Figure 9-4-3. The 95% confidence interval for the estimate is (0.00243, 0.00307), which contains the value (0.00255) obtained by using reliability index in the previous section.

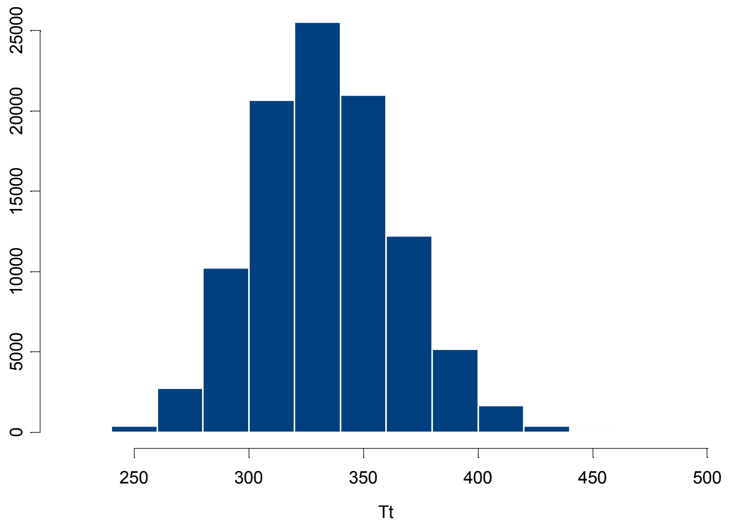


Figure 9-4-3: Histogram of U43 T_t in Monte – Carlo simulation

9.5 Comparison of the time to untenable conditions in four scenarios of the CESARE-Risk Model

To compare the four scenarios: DOWO, DOWC, DCWO and DCWC in the CESARE-Risk model, we set the input parameters with same values in these four scenarios. For simplicity, we take two input variables to be random: the width of room W_r, denoted by x_2 and the flame spread rate R_f, denoted by x_8. The other input variables are constant. x_2 has normal distribution with mean 450 and standard deviation 20. x_8 has normal distribution with mean 0.7 and standard deviation 0.1. The other input variables are constant. The values of the input parameters are shown in Table 9-5-1.

Variables	Name of variables	Symbol	Unit	Values
x_1	Length of Room	L	cm	600
x_2	Width of Room	W_r	cm	N(450, 20)
x_3	Height of Room	H_r	cm	250
x_4	Window Width Factor	f_W		0.7
x_5	Window Height Factor	f_H		0.6
x_6	Fuel Density	ρ_f	kg/ m^2	50
x_7	Fuel Area Factor	f_A		0.6
x_8	Flame Spread Rate	R_f	m/sec	N(0.7, 0.1)

Table 9-5-1: The values of input parameters for comparing analysis

Appropriate values of the time to untenable conditions were chosen for each scenario, as shown in Table 9-5-2. The results of reliability analysis for the four scenarios are shown in the Table 9-5-2. The analysis is based on data for which $R_f \geq 0.5$ m/sec.

Scenario	N1	N2	N12	Tun	β	$\Phi(-\beta)$	p_F	95% conf_interval
DCWO	656	1332	1988	300	2.5585	0.00526	0.00518	(0.00474, 0.00562)
DOWO	489	1499	1988	320	2.6438	0.00410	0.00419	(0.00379, 0.00459)
DCWC	1769	219	1988	230	2.5073	0.00608	0.00591	(0.00543, 0.00639)
DOWC	1519	469	1988	250	2.8001	0.00255	0.00275	(0.00243, 0.00307)

Table 9-5-2: The results of reliability analysis for four scenarios

Let:

TUN1 = time to untenable conditions due to heat reaching fatality level;

TUN2 = time to untenable conditions due to CO reaching fatality level;

In Table 9-5-2:

N1 = number of observations for which TUN1 < TUN2 (CO reached fatality level before Heat reached fatality level);

N2 = number of observations for which TUN1 > TUN2 (Heat reached fatality level before CO reached fatality level);

N12 = total number of observations

β = reliability index;

Φ = distribution function of normal standard distribution;

Tun = minimum required time to untenable conditions, in seconds, for safety;

p_F = probability of failure from Monte-Carlo simulation;

95% conf_interval = 95% confidence interval of Monte-Carlo simulation.

The results, in Table 9-5-2, lead to following conclusions:

a) In DCWO scenario, there are 1988 observations. In 67% of the observations untenable conditions were caused by CO reaching fatality level before heat reached fatality level. In 33% of the observations untenable conditions were caused by heat.

b) In DOWO scenario, there are 1988 observations. In 75% of the observations untenable conditions were caused by CO reaching fatality level before heat reached fatality level. In 25% of the observations untenable conditions were caused by heat.

c) In DCWC scenario, there are 1988 observations. In 11% of the observations untenable conditions were caused by CO reaching fatality level before heat reached fatality level. In 89% of the observations untenable conditions were caused by heat.

d) In DOWC scenario, there are 1988 observations. In 24% of the observations untenable conditions were caused by CO reaching fatality level before heat reached fatality level. In 76% of the observations untenable conditions were caused by heat.

From Figure 9-5-1, which is the transformed time to untenable conditions against reliability index β, it is clear that:

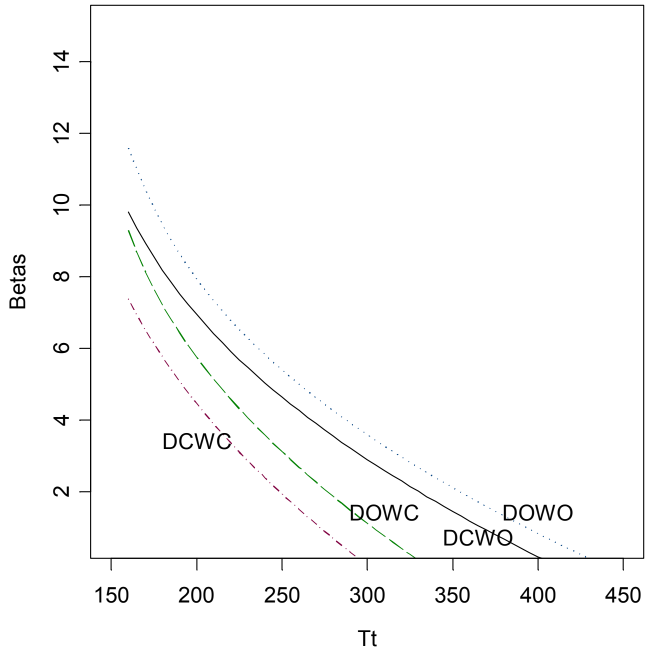


Figure 9-5-1: The transformed time to untenable conditions T_t against reliability index β for four scenarios

Scenario DCWC has the smallest value of reliability index β (which implies largest probability of failure) for each value of T_t , time to untenable conditions, so that DCWC is the most dangerous scenario to the occupants for the assumed input data in Table 9-5-1.

Scenario DOWO has the largest value of reliability index β (which implies smallest probability of failure) for each value of T_t , time to untenable conditions, so that DOWO is the safest scenario for the occupants for the assumed input data in Table 9-5-1.

CHAPTER 10
REGRESSION ANALYSIS FOR TIME TO UNTENABLE CONDITIONS BY USING LOGARITHMIC FIT TO THE OUTPUT

In this chapter, we shall use a logarithmic fit to the output, time to untenable conditions. Then, the reliability index for some examples for the four scenarios will be calculated. Also the corresponding probability of failure for each of scenarios derived from FOSM method will be compared with Monte Carlo simulation and with the direct fit results.

10.1 Regression analysis for U13 in DCWO scenario

10.1.1 Derivation of regression equations for DCWO scenario

To the data set U13, we fitted a regression formula of the form (I = 1, 2, 3, 4, 5, 7, 8.)

$$\log(T_{tL}) = \sum_{i\in I}(a_i x_i^2 + b_i x_i) + c + \varepsilon\,. \qquad (10\text{-}1\text{-}1)$$

The coefficient c = 4.636, and coefficients b_i and a_i were as in Table 10-1-1.

i	1	2	3	4	5	7	8
b_i	0.0008904	0.001136	0.003880	0.5549	0.7209	0.01602	-1.402
a_i	-4.161e-007	-4.202e-007	-4.863e-006	-0.1652	-0.2498	0.01485	0.3062

Table 10-1-1: Values of quadratic regression coefficients for U13

Setting

$$\log(T_{tL}) = \sum_{i\in I}(a_i x_i^2 + b_i x_i) + c \qquad (10\text{-}1\text{-}2)$$

$$T_{tL} = \exp(\sum_{i\in I}(a_i x_i^2 + b_i x_i) + c) \qquad (10\text{-}1\text{-}3)$$

it was found that the correlation between T_{tL} and the original value T was 0.9978.

The second step in the fitting is to improve the fit of T_{tL} by using a cubic regression formula of the form

$$T_{PL} = C_{L0} + C_{L1}T_{tL} + C_{L2}T_{tL}^2 + C_{L3}T_{tL}^3 + \varepsilon_L^*\,. \qquad (10\text{-}1\text{-}4)$$

The coefficients turned out to be:

C_{L0} = -4.318, C_{L1} = 1.099, C_{L2} = -0.0004658, C_{L3} = 5.677e-007.

Letting

$$T_{PL} = C_{L0} + C_{L1}T_{tL} + C_{L2}T_{tL}^2 + C_{L3}T_{tL}^3 \qquad (10\text{-}1\text{-}5)$$

the correlation between T_{pL} and the original values T is 0.9985.

A scatter plot of T against T_{tL} is shown in Figure 10-1-1.

A scatter plot of T against T_{pL} is shown in Figure 10-1-2.

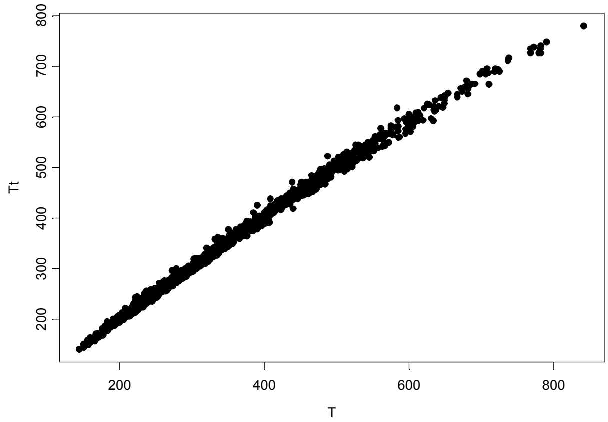


Figure 10-1-1: Scatter plot of original output T against T_{tL} for U13

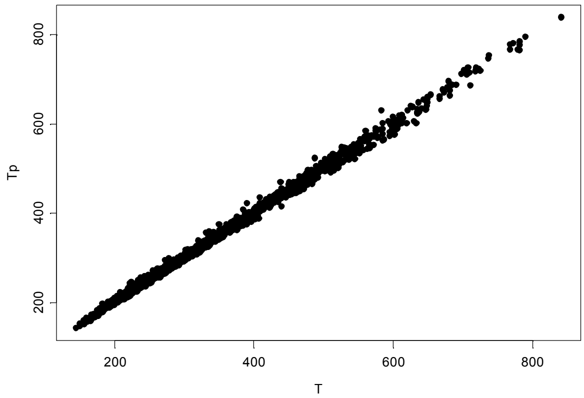


Figure 10-1-2: Scatter plot of original output T against T_{pL} for U13

10.1.2 Calculation of reliability index for DCWO scenario

In DCWO scenario, we just take two input variables to be random: room width W_r, which is denoted by x_2 and flame spread rate R_f , which is denoted by x_8. The other input variables will be taken to be constant. Let $N(\mu, \sigma)$ denote a normal random variable with mean μ and standard deviation σ.

The assumed values are as follows:

$L = 600$, $W_r = N(450,20)$, $H_r = 250$, $f_W = 0.7$, $f_H = 0.6$, $\rho_f = 50$, $f_A = 0.6$, $R_f = N(0.7, 0.1)$.

The limiting state is taken to be T_{pLmin} = 300 seconds. The corresponding T_{tLmin} = 301.2, and the standardized limit equation is

$$\log(301.2) = -0.0001681u_2^2 + 0.01515u_2 + 0.003062u_8^2 - 0.09735u_8 + 5.946\,.$$

Using the methodology described in equations (7-1-4) to (7-1-7) in Chapter 7:

the values of λ are:

λ: 0.00006086 0.00006086 -0.001335 0.01575

and the corresponding values of β are:

β: 72.36 72.36 28.64 2.633.

So, the smallest value β = 2.633 is the reliability index.

From equation (7-1-8), the approximate value of the probability of failure is:

$p_F = \Phi(-2.633) = 0.00423$.

Illustration of the result is given in Figure 10-1-3 and 10-1-4.

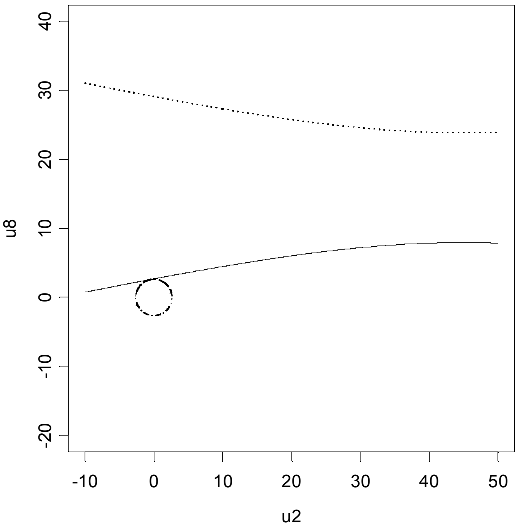


Figure 10-1-3: Illustration of reliability index for U13 (logarithmic fit)

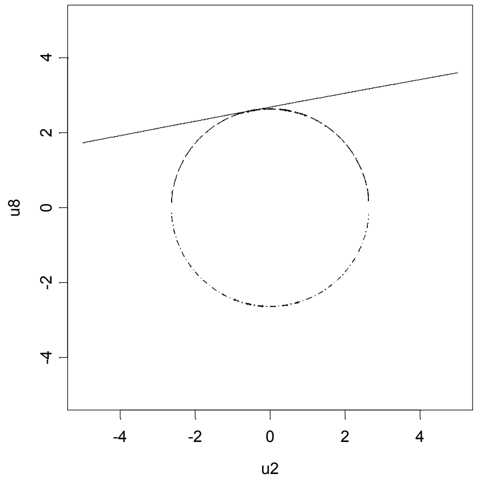


Figure 10-1-4: Illustration of reliability index for U13 (logarithmic fit)
(Enlarged central part)

10.1.3 Validation by Monte - Carlo simulation

The probability just given by the reliability index can be validated by Monte - Carlo simulation: Out of 100000 simulations, 447 observations fell in the failure region $T_p <$ 300 seconds ($T_{tL} <$ 301.2 seconds), giving an estimated probability of failure of 0.00477. The histogram is shown in Figure 10-1-5. The 95% confidence interval for the estimate is (0.00406, 0.00488), which contains the value (0.00423) obtained by using the reliability index in the previous section **10.1.2**.

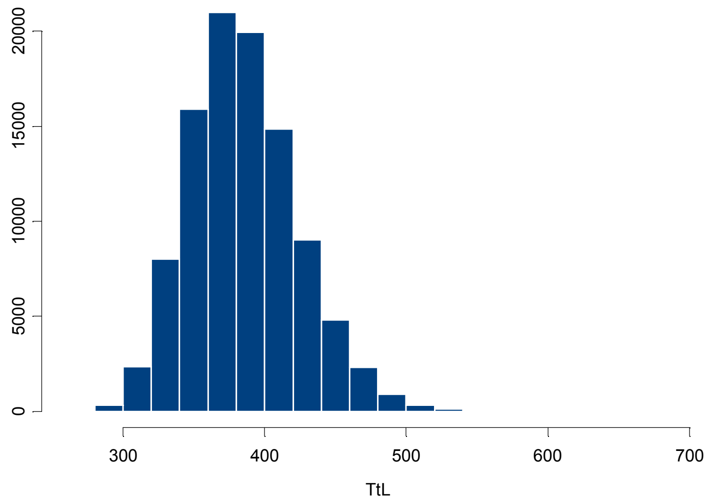


Figure 10-1-5: Histogram of Monte Carlo simulation for U13

10.2 Regression analysis of U23 in DOWO scenario

10.2.1 Derivation of regression equations for DOWO scenario

To the data set U23, we fitted a regression formula of the form

$$\log(T_{tL}) = \sum_{i \in I}(a_i x_i^2 + b_i x_i) + c + \varepsilon_L^*, \; I = 1,2,3,4,5,7,8. \qquad (10\text{-}2\text{-}1)$$

The coefficient c = 5.102, and coefficients b_i and a_i were as in Table 10-2-1.

i	1	2	3	4	5	7	8
b_i	0.0008828	0.0009746	0.002679	0.4939	0.4470	0.01757	-1.419
a_i	-4.1906e-07	-3.4301e-07	-2.9298e-06	-0.1402	-0.09876	0.01027	0.3102

Table 10-2-1: Values of quadratic regression coefficients for U23

Setting

$$\log(T_{tL}) = \sum_{i \in I}(a_i x_i^2 + b_i x_i) + c \qquad (10\text{-}2\text{-}2)$$

$$T_{tL} = \exp(\sum_{i \in I}(a_i x_i^2 + b_i x_i) + c) \qquad (10\text{-}2\text{-}3)$$

it was found that the correlation between T_{tL} and the original value T is 0.9984.

The second step in the fitting is to improve the fit of T_{tL} by using a cubic regression formula of the form

$$T_{PL} = C_{L0} + C_{L1}T_{tL} + C_{L2}T_{tL}^2 + C_{L3}T_{tL}^3 + \varepsilon_L^*. \qquad (10\text{-}2\text{-}4)$$

The coefficients turned out to be:

C_{L0} = -7.254, C_{L1} = 1.119, C_{L2} = -0.0004849, C_{L3} = 5.394e-007.

Letting

$$T_{PL} = C_{L0} + C_{L1}T_{tL} + C_{L2}T_{tL}^2 + C_{L3}T_{tL}^3 \qquad (10\text{-}2\text{-}5)$$

the correlation between T_{pL} and the original values T is 0.9990.

A scatter plot of T against T_{tL} is shown in Figure 10-2-1.

A scatter plot of T against T_{pL} is shown in Figure 10-2-2.

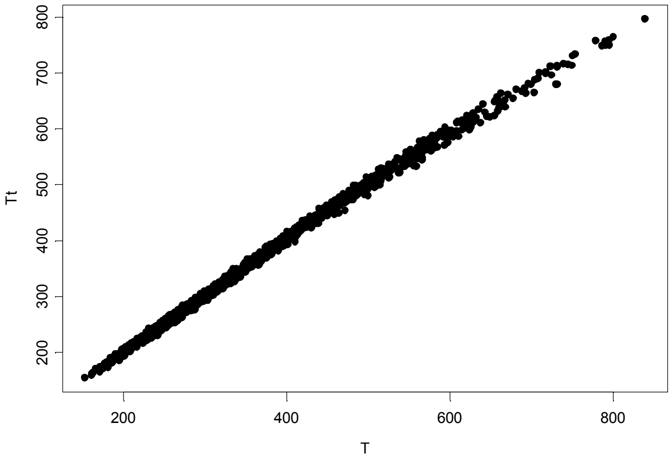


Figure 10-2-1: A scatter plot of *T* against T_{tL} for U23

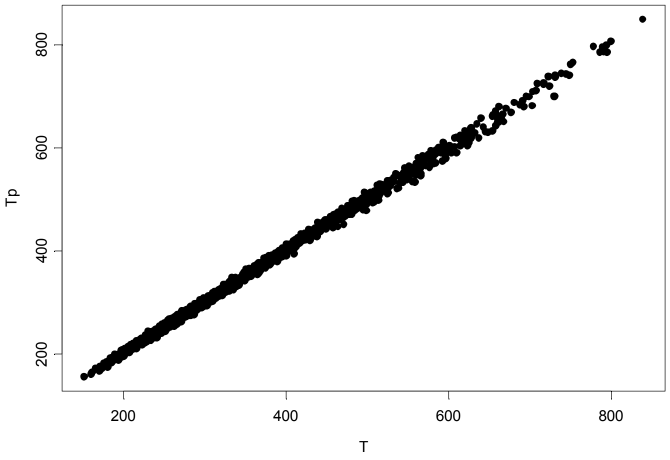


Figure 10-2-4: A scatter plot of *T* against T_{pL} for U23

10.2.2 Calculation of reliability index for DOWO scenario

In DOWO scenario, we just take two input variables to be random: fuel area factor W_r, which is denoted by x_2 and flame spread rate R_f , which is denoted by x_8. The other input variables will be taken to be constant. Let $N(\mu, \sigma)$ denote a normal random variable with mean μ and standard deviation σ.

The assumed values are as follows:

$L = 600$, $W_\mathrm{r} = N(450, 20)$, $H_r = 250$, $f_W = 0.7$, $f_H = 0.6$, $\rho_f = 50$, $f_A = 0.6$, $R_f = N(0.7, 0.1)$.

The limiting state is taken to be $T_{pLmin} = 320$ seconds, and the corresponding value of $T_{tLmin} = 321.1$.

Thus the regression equation reduces in **U** plane to:

$$\log(321.1) = -0.0001372u_2^2 + 0.01332u_2 + 0.003102u_8^2 - 0.0985u_8 + 6.019 . \tag{10-2-6}$$

Using the methodology described in equations (7-1-4) to (7-1-7) in Chapter 7:

the values of λ are

λ: 0.00005426 0.00005426 -0.001361 0.01528

and the corresponding values of β are

β: 81.78 81.78 28.63 2.715.

So the smallest value $\beta = 2.715$ is the reliability index.

From equation (7-1-8), the approximate value of the probability of failure is:

$p_F = \Phi(-2.715) = 0.00332$.

Illustration of the result is given in Figures 10-2-3 and 10-2-4.

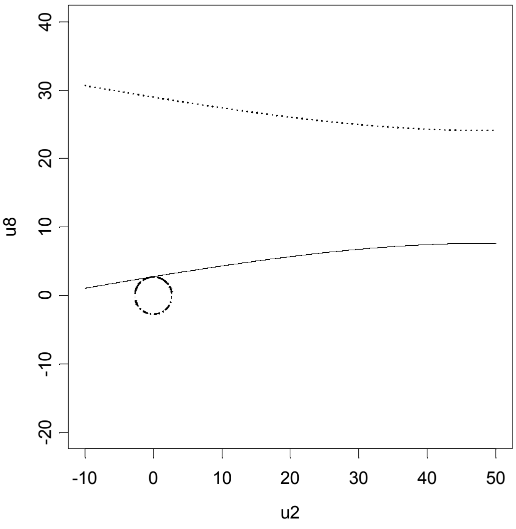


Figure 10-2-3: Illustration of reliability index for U23 (logarithmic fit)

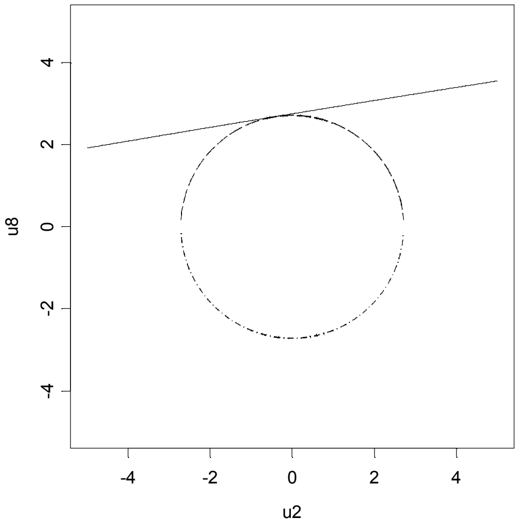


Figure 10-2-3: Illustration of reliability index for U23 (logarithmic fit)
(Enlarged central part)

10.2.3 Validation by Monte - Carlo simulation

The probability just given by the reliability index can be validated by Monte - Carlo simulation: Out of 100000 simulations, 334 observations fell in the failure region $T_p <$ 320 seconds ($T_{tL} <$ 321.1 seconds), giving an estimated probability of failure of 0.00334. The histogram is shown in Figure 10-2-5. The 95% confidence interval for the estimate is (0.00298, 0.00370), which contains the value (0.00332) obtained by using the reliability index in the previous section **10.2.2**.

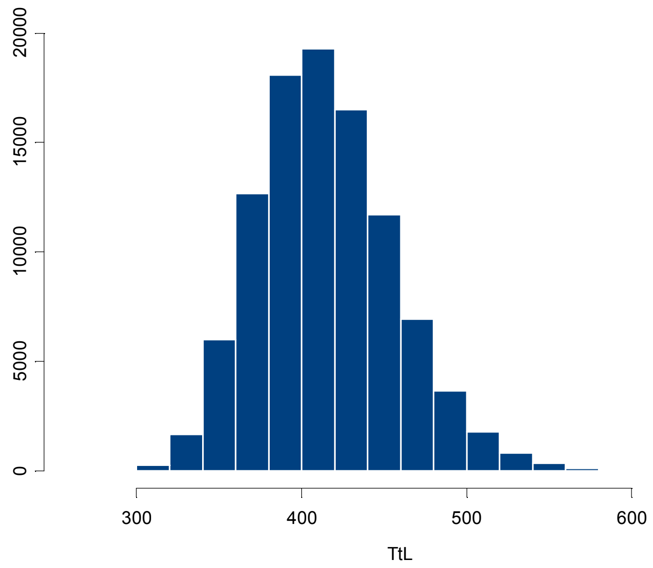


Figure 10-2-5: Histogram of Monte Carlo simulation for U23

10.3 Regression analysis for U33 in DCWC scenario

10.3.1 Derivation of regression equations for DCWC scenario

To the data set U33, we fitted a regression formula of the form

$$\log(T_{tL}) = \sum_{i\in I}(a_i x_i^2 + b_i x_i) + c + \varepsilon_L^* \, . I = 1,2,3,7,8. \qquad (10\text{-}3\text{-}1)$$

The coefficient c = 5.240, and coefficients b_i and a_i were as in Table 10-3-1.

i	1	2	3	7	8
b_i	0.0008228	0.0009558	0.002149	0.04475	-1.293
a_i	-2.830e-007	-3.502e-007	-2.336e-006	0.01545	0.2913

Table 10-3-1: Values of quadratic regression coefficients for U33

Setting

$$\log(T_{tL}) = \sum_{i\in I}(a_i x_i^2 + b_i x_i) + c \qquad (10\text{-}3\text{-}2)$$

$$T_{tL} = \exp(\sum_{i\in I}(a_i x_i^2 + b_i x_i) + c) \qquad (10\text{-}3\text{-}3)$$

it was found that the correlation between T_{tL} and original value T is 0.9984.

The second step in the fitting is to improve the fit of T_{tL} by using a cubic regression formula of the form

$$T_{PL} = C_{L0} + C_{L1}T_{tL} + C_{L2}T_{tL}^2 + C_{L3}T_{tL}^3 + \varepsilon_L^* \, . \qquad (10\text{-}3\text{-}4)$$

The coefficients turned out to be:

C_{L0} = -8.501, C_{L1} = 1.171, C_{L2} = -0.000939, C_{L3} = 1.480e-006.

Letting

$$T_{PL} = C_{L0} + C_{L1}T_{tL} + C_{L2}T_{tL}^2 + C_{L3}T_{tL}^3 \qquad (10\text{-}3\text{-}5)$$

the correlation between T_{pL} and the original values T is 0.9990.

A scatter plot of T against T_{tL} is shown in Figure 10-3-1.

A scatter plot of T against T_{pL} is shown in Figure 10-3-2.

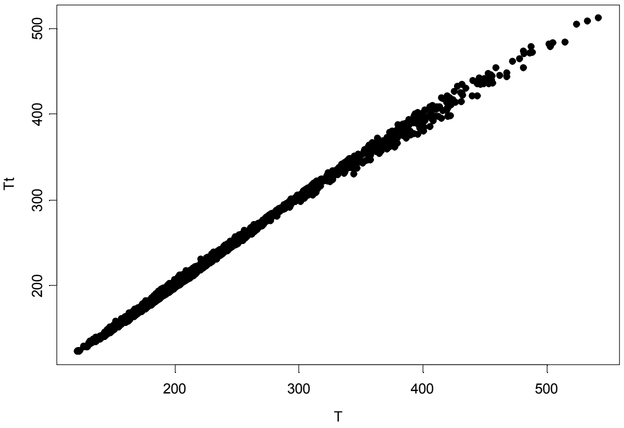


Figure 10-3-1: A scatter plot of *T* against T_{tL} for U33

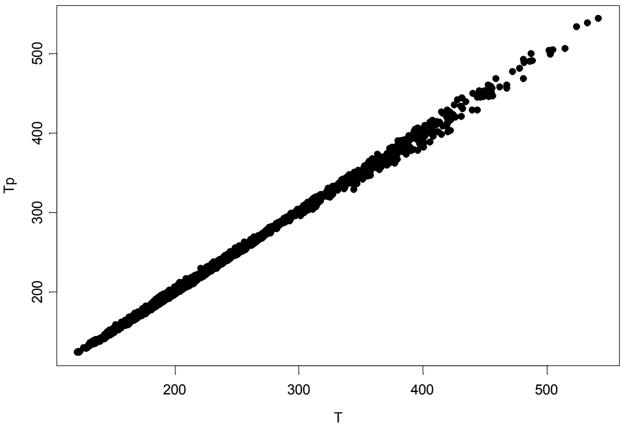


Figure 10-3-2: A scatter plot of *T* against T_{pL} for U33

10.3.2 Calculation of reliability index for DCWC scenario

In DCWC scenario, we just take two input variables to be random: fuel area factor W_r, which is denoted by x_2 and flame spread rate R_f, which is denoted by x_8. The other input variables will be taken to be constant. Let $N(\mu, \sigma)$ denote a normal random variable with mean μ and standard deviation σ.

The assumed values are as follows:

$L = 600$, $W_r = N(450,20)$, $H_r = 250$, $f_W = 0.7$, $f_H = 0.6$, $\rho_f = 50$, $f_A = 0.6$, $R_f = N(0.7, 0.1)$.

The limiting state is taken to be $T_{pLmin} = 230$ seconds. The corresponding value of $T_{tLmin} = 230.9$, and the regression equation reduces in **U** plane to:

$$\log(230..9) = -0.0001401u_2^2 + 0.01281u_2 + 0.002913u_8^2 - 0.08847u_8 + 5.653 . \tag{10-3-3}$$

Using the methodology described in equation (7-1-4) to (7-1-7) in Chapter 7:

the values of λ are

λ: 0.0001 0.0001 -0.0013 0.0146

and the corresponding values of β are

β: 73.57 73.57 27.37 2.565.

The smallest value $\beta = 2.565$ is the reliability index.

From equation (7-1-8), the approximate value of the probability of failure is:

$p_F = \Phi(-2.565) = 0.00517$.

Illustration of the result is given in Figure 10-3-3 and 10-3-4.

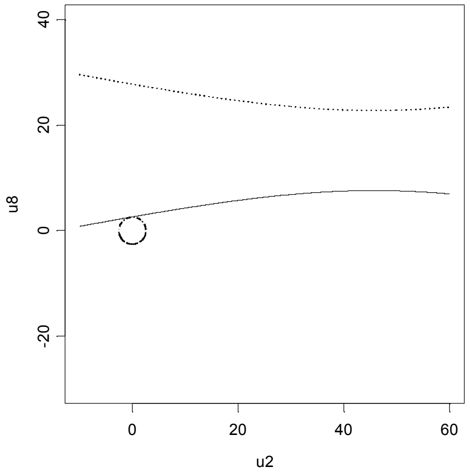


Figure 10-3-3: Illustration of reliability index for U33 (logarithmic fit)

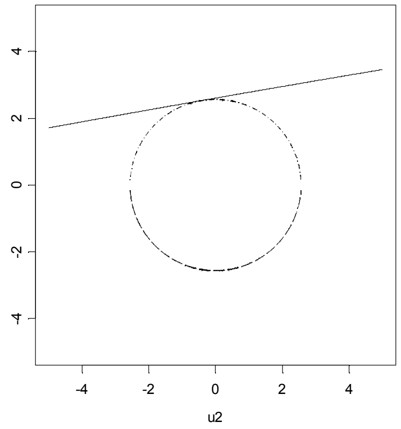


Figure 10-3-4: Illustration of reliability index for U33 (logarithmic fit)
(Enlarged central part)

10.3.3 Validation by Monte - Carlo simulation

The probability just given by the reliability index can be validated by Monte - Carlo simulation: Out of 100000 simulations, 529 observations fell in the failure region $T_p <$ 230 seconds ($T_{tL} <$ 230.9 seconds), giving an estimated probability of failure of 0.00529. The histogram is shown in Figure 10-3-5. The 95% confidence interval for the estimate is (0.00484, 0.00574), which contains the value (0.00517) obtained by using the reliability index in the previous section **10.3.2**.

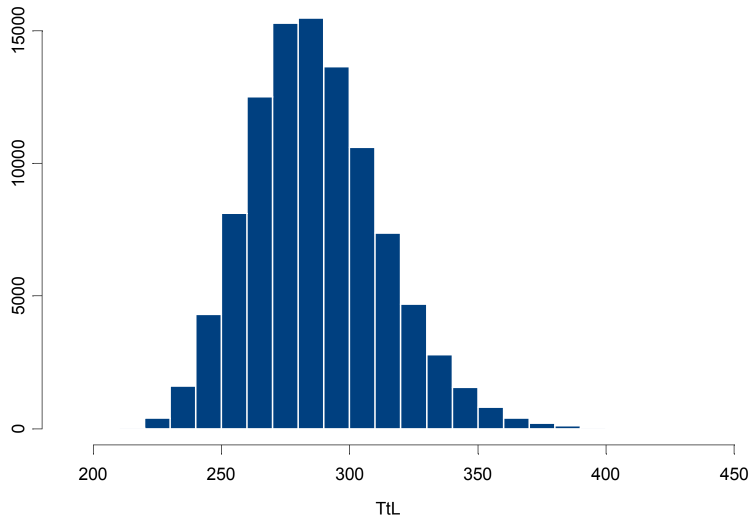


Figure 10-3-5: Histogram of Monte Carlo simulation for U33

10.4 Regression analysis for U43 in DOWC scenario

10.4.1 Derivation of regression equations for DOWC scenario

The set of indices used was just $I = 1, 2, 3, 7, 8$.

To the data set U43, we fitted a regression formula of the form

$$\log(T_{tL}) = \sum_{i \in I}(a_i x_i^2 + b_i x_i) + c + \varepsilon_L^*. \qquad (10\text{-}4\text{-}1)$$

The coefficient $c = 5.553$, and coefficients b_i and a_i were as in Table 10-4-1.

i	1	2	3	7	8
b_i	0.0008355	0.0007051	0.001737	0.04742	-1.338
a_i	-3.182e-007	-2.257e-007	-1.902e-006	0.007970	0.2988

Table 10-4-1: Values of quadratic regression coefficients for U43

Setting

$$\log(T_{tL}) = \sum_{i \in I}(a_i x_i^2 + b_i x_i) + c \qquad (10\text{-}4\text{-}2)$$

$$T_{tL} = \exp(\sum_{i \in I}(a_i x_i^2 + b_i x_i) + c) \qquad (10\text{-}4\text{-}3)$$

it was found that the correlation between T_{tL} and the original value T is 0.9987.

The second step in the fitting is to improve the fit of T_{tL} by using a cubic regression formula of the form

$$T_{PL} = C_{L0} + C_{L1}T_{tL} + C_{L2}T_{tL}^2 + C_{L3}T_{tL}^3 + \varepsilon_L^*. \qquad (10\text{-}4\text{-}4)$$

The coefficients turned out to be:

C_{L0} = -19.88, C_{L1} = 1.284, C_{L2} = -0.001238, C_{L3} = 1.639e-006.

Letting

$$T_{PL} = C_{L0} + C_{L1}T_{tL} + C_{L2}T_{tL}^2 + C_{L3}T_{tL}^3 \qquad (10\text{-}4\text{-}5)$$

the correlation between T_{pL} and the original values T is 0.9992.

A scatter plot of T against T_{tL} is shown in Figure 10-4-1.

A scatter plot of T against T_{pL} is shown in Figure 10-4-2.

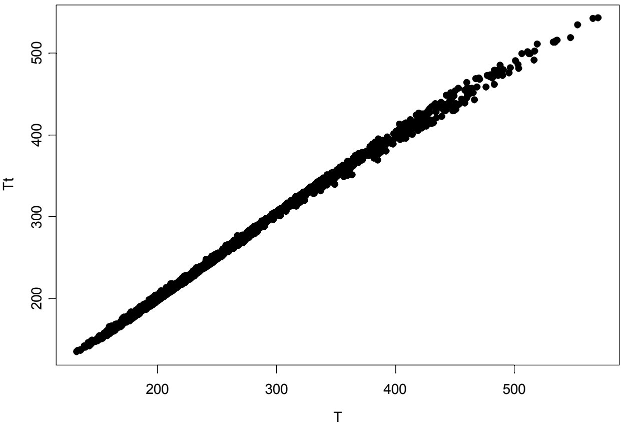


Figure 10-4-1: A scatter plot of T against T_{tL} in U43L

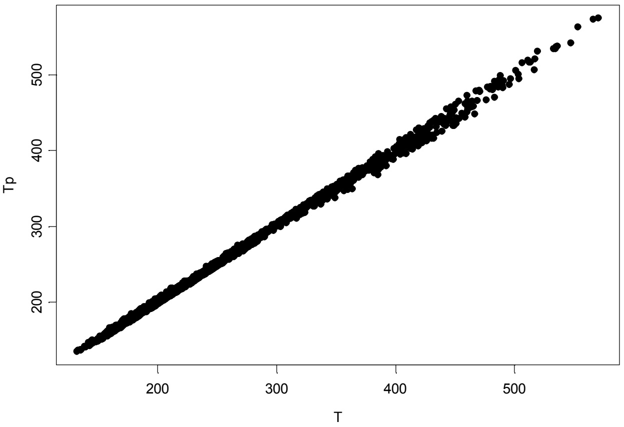


Figure 10-4-2: A scatter plot of T against T_{pL} in U43L

10.4.2 Calculation of reliability index for DCWC scenario

In DCWC scenario, we just take two input variables to be random: fuel area factor W_{r}, which denoted by x_2 and flame spread rate R_f, which denoted by x_8. The other input variables will be taken to be constant. Let $N(\mu, \sigma)$ denote a normal random variable with mean μ and standard deviation σ.

The assumed values are as follows:

$L = 600$, $W_{\mathrm{r}} = N(450{,}20)$, $H_r = 250$, $f_W = 0.7$, $f_H = 0.6$, $\rho_f = 50$, $f_A = 0.6$, $R_f = N(0.7{,}0.1)$.

The limiting state is taken to be $T_{pLmin} = 250$ seconds, and the corresponding value of $T_{tLmin} = 250.7$. The regression equation reduces in **U** plane to:

$$\log(250.7) = -0.00009029u_2^2 + 0.01004u_2 + 0.002988u_8^2 - 0.09201u_8 + 5.767\,. \tag{10-4-3}$$

Using the methodology described in equations (7-1-4) to (7-1-7) in Chapter 7:

The values of λ are

λ: 0.0000 0.0000 -0.0013 0.0131

and the corresponding values of β are

β: 93.95 93.95 27.63 2.891

The smallest value $\beta = 2.891$ is the reliability index.

From equation (7-1-8), the approximate value of the probability of failure is:

$p_F = \Phi(-2.891) = 0.00192$.

Illustration of the result is given in Figure 10-4-3 and 10-4-4.

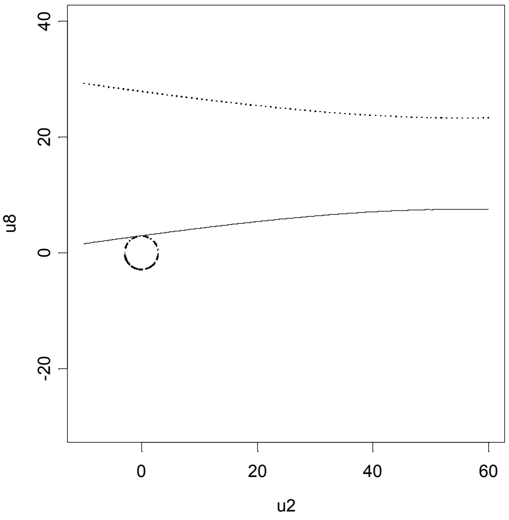


Figure 10-4-3: Illustration of reliability index for U43 (logarithmic fit)

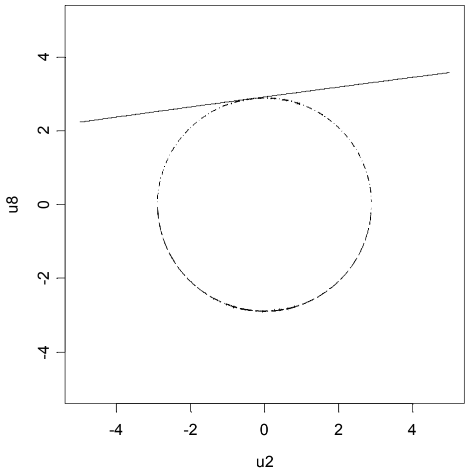


Figure 10-4-4: Illustration of reliability index for U43 (logarithmic fit)
(Enlarged central part)

10.4.3 Validation by Monte - Carlo simulation

The probability just given by reliability index can be validated by Monte - Carlo simulation: Out of 100000 simulations, 198 observations fell in the failure region $T_p <$ 250 seconds ($T_{tL} <$ 250.7 seconds), giving an estimated probability of failure of 0.00198. The histogram is shown in Figure 10-4-5. The 95% confidence interval for the estimate is (0.00170, 0.00226), which contains the value (0.00192) obtained by using the reliability index in the previous section **10.4.2**.

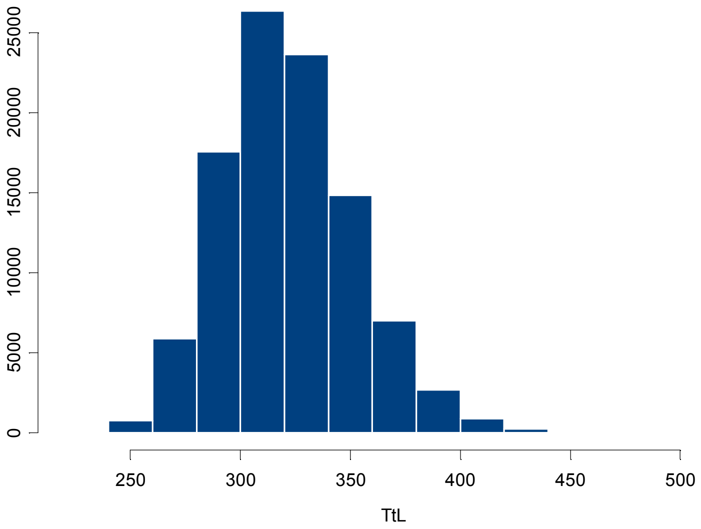


Figure 10-4-5: Histogram of Monte Carlo simulation for U43

10.5 Results for time to untenable conditions with logarithmic fit to output

To compare the four scenarios: DOWO, DOWC, DCWO and DCWC in the CESARE-Risk model with logarithmic fit to the output, time to untenable conditions, we set the input parameters with same values in these four scenarios. For simplicity, we take two input variables to be random: the width of room W_r, denoted by x_2 and the flame spread rate R_f, denoted by x_8. The other input variables are constant. x_2 has normal distribution with mean 450 and standard deviation 20. x_8 has normal distribution with mean 0.7 and standard deviation 0.1. The values of the input parameters are shown in Table 10-5-1.

Variables	Name of variables	Symbol	Unit	Values
x_1	Length of Room	L	cm	600
x_2	Width of Room	W_r	cm	N(450, 20)
x_3	Height of Room	H_r	cm	250
x_4	Window Width Factor	f_W		0.7
x_5	Window Height Factor	f_H		0.6
x_6	Fuel Density	ρ_f	kg/ m^2	50
x_7	Fuel Area Factor	f_A		0.6
x_8	Flame Spread Rate	R_f	m/sec	N(0.7, 0.1)

Table 10-5-1: The values of input parameters for comparison analysis

Appropriate values of the time to untenable conditions were chosen for each scenario, as shown in Table 10-5-2. The results of reliability analysis for the four scenarios, are shown in Table 10-5-2 (when $R_f \geq 0.5$ m/sec).

Scenario	N1	N2	N12	Tun	β_L	$\Phi(-\beta_L)$	p_F	95% conf_interval
DCWO	656	1332	1988	300	2.6334	0.00423	0.00477	(0.00406, 0.00488)
DOWO	489	1499	1988	320	2.7146	0.00332	0.00334	(0.00298, 0.00370)
DCWC	1769	219	1988	230	2.5645	0.00517	0.00529	(0.00484, 0.00574)
DOWC	1519	469	1988	250	2.8908	0.00192	0.00198	(0.00170, 0.00226)

Table 10-5-2: The results of reliability analysis for four scenarios

Let:

TUN1 = time to untenable conditions due to heat reaching fatality level;

TUN2 = time to untenable conditions due to CO reaching fatality level;

In Table 10-5-2:

N1 = observations TUN1 < TUN2 (CO reached fatality level before Heat reached fatality level , 300°C);

N2 = observations TUN1 > TUN2 (Heat reached fatality level, 300°C before CO reached fatality level);

N12 = total number of observations (TUN12=min (TUN1,TUN2));

β = reliability index;

β_L = reliability index with fit to logarithmic output;

Φ = distribution function of normal standard distribution;

Tun = time to untenable conditions, seconds;

p_F = probability of failure from Monte-Carlo simulation;

95% conf_interval = 95% confidence interval of Monte-Carlo simulation.

Scenario ($R_f \geq 0.5$)	**N12**	Tun	β	**Φ(-β)**	β_L	**Φ(-β_L)**
DCWO	1988	300	2.5585	0.00526	2.6334	0.00423
DOWO	1988	320	2.6438	0.00410	2.7146	0.00332
DCWC	1988	230	2.5073	0.00608	2.5645	0.00517
DOWC	1988	250	2.8001	0.00255	2.8908	0.00192

Table 10-5-3: Comparation of β and β_L

From table 10-5-3, the value of reliability index β_L is slightly larger than the value β within the same scenario, for same values of input parameters and same value of the time to untenable conditions. This is because the logarithmic fit resulted in more curvature to the limit state surface (line). Of course, more curvature here leads to more accurate fit, as can be seen from comparing correlations between $T \sim T_p$ to $T \sim T_{pL}$ in the same data set of same scenario, which will give a slightly larger value of reliability index β. This can be seen clearly through the comparison of limit state lines and related β and β_L in Figure 10-5-1 to Figure 10-5-8.

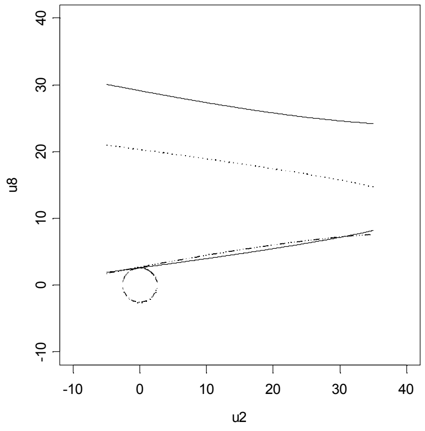


Figure 10-5-1: Comparation of limit state lines and related β and β_L of U13

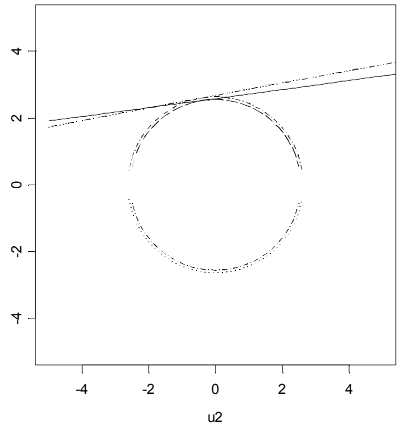


Figure 10-5-2: Comparation of limit state lines and related β and β_L of U13 (Enlarged central part, dot line is logarithmic fit)

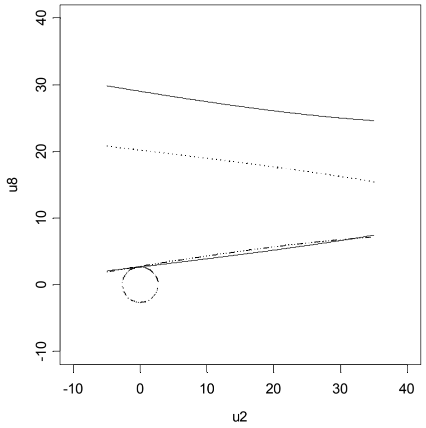


Figure 10-5-3: Comparation of limit state lines and related β and β_L of U23

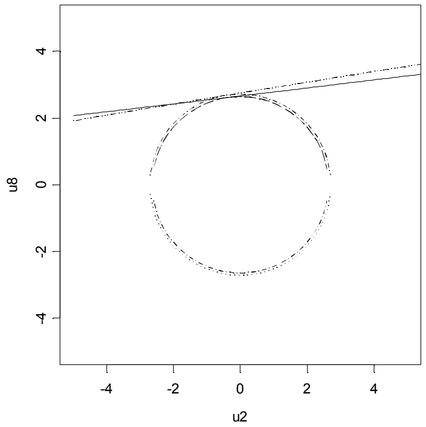


Figure 10-5-4: Comparation of limit state lines and related β and β_L of U23 (Enlarged central part, dot line is logarithmic fit)

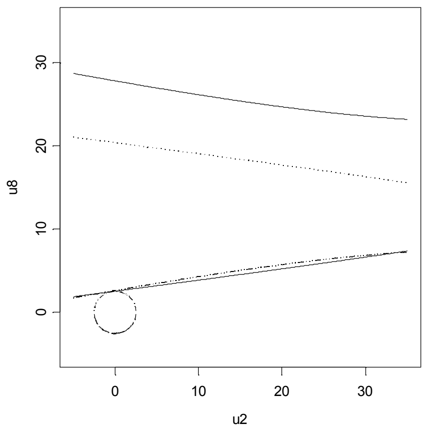


Figure 10-5-5: Comparation of limit state lines and related β and β_L of U33

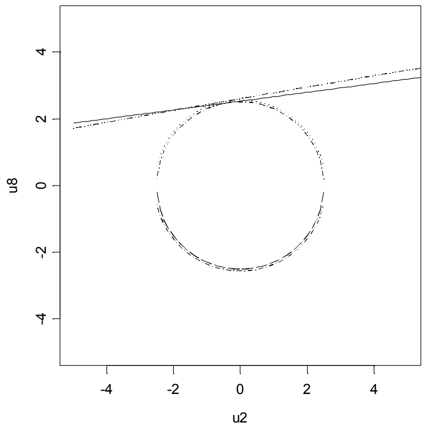


Figure 10-5-6: Comparation of limit state lines and related β and β_L of U33

(Enlarged central part, dot line is logarithmic fit)

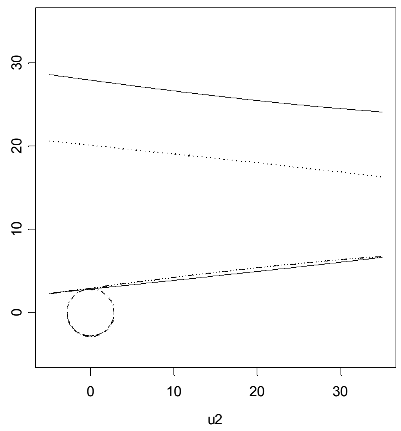


Figure 10-5-7: Comparation of limit state lines and related β and β_L of U43

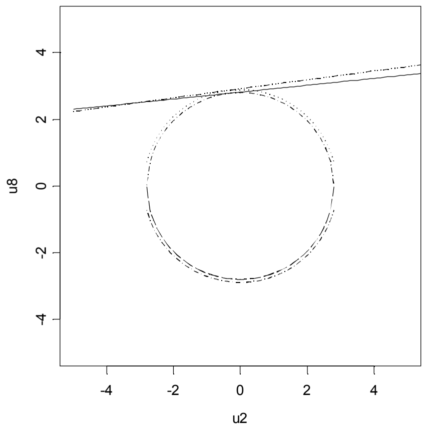


Figure 10-5-8: Comparation of limit state lines and related β and β_L of U43

(Enlarged central part, dot line is logarithmic fit)

CHAPTER 11

CONCLUSION AND RECOMMENDATIONS

The aim of the research presented in this thesis was

(1) To identify some aspects of probability-based indices of safety for use by practising engineers in comparing competing building designs through using statistical analysis in fire engineering.

(2) To develop a methodology which can be used with minimal effort to obtain an accurate value for the reliability of a considered design, once the probability of the input is decided.

The main contribution of this research is the use of the modern regression methods, AVAS and/or ACE, followed by polynomial approximations to the non - linear transformations of the parameters, which provides us with simple response surfaces that represent the output of a computer fire model, the CESARE-Risk Model. Once the probability distribution of the input is decided, the response surfaces can be used with minimal effort to obtain an accurate value for the reliability index, which is the probability of failure of the building design based on fire safety. The probability of failure can be obtained by First Order Second Moment (FOSM) calculation or by Monte-Carlo simulation. The procedure is shown in Figure 11-1.

This research confines itself to four scenarios: DOWC, DCWO, DCWC, and DOWO, eight input parameters and two output parameters: maximum temperature reached and time to untenable conditions, to illustrate the methodology. A comparison of the results of time to untenable conditions for the four scenarios is show in table 11-1.

Scenario	**N12**	**Tun**	$\boldsymbol{\beta}$	$\boldsymbol{\Phi(-\beta)}$	p_F	**95% conf_interval**
DCWO	1988	300	2.5585	0.00526	0.00518	(0.00474, 0.00562)
DOWO	1988	320	2.6438	0.00410	0.00419	(0.00379, 0.00459)
DCWC	1988	230	2.5073	0.00608	0.00591	(0.00543, 0.00639)
DOWC	1988	250	2.8001	0.00255	0.00275	(0.00243, 0.00307)

Table 11-1: The results of reliability analysis for four scenarios

The methodology has been tested on other output parameters, with equal success.

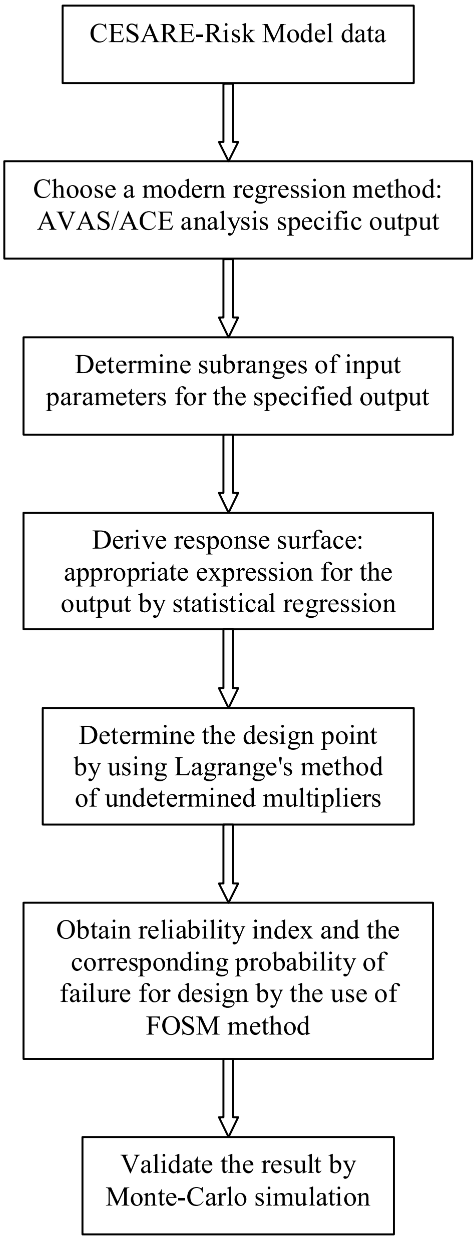

CESARE-Risk Model data
Choose a modern regression method: AVAS/ACE analysis specific output
Determine subranges of input parameters for the specified output
Derive response surface: appropriate expression for the output by statistical regression
Determine the design point by using Lagrange's method of undetermined multipliers
Obtain reliability index and the corresponding probability of failure for design by the use of FOSM method
Validate the result by Monte-Carlo simulation

Figure 11-1: Procedure for calculation of the reliability of a considered design

The value of the developed methodology that is described in this thesis depends on the accuracy of the results of the computer model used. But the author is confident that the developed method will remain equally successful when applied to better computer fire models that will no doubt be developed in coming years. Even with the best computer models, the problem of uncertainty in the input parameters will still need to be addressed, and the methodology presented in this thesis will make the task much easier.

REFERENCES

1. HADJISOPHOCLEOUS, G. V., BENICHOU, N. and TAMIM, A. S., 1998, Literature Review of Performance – Based Fire Codes and Design Environment, Journal of Fire Protection Engineering, 9 (1), pp. 12-40.
2. RAMACHANDRAN, G., 1988, The Economics of Fire Protection, Preface, E & FN Spon.
3. RAMACHANDRAN, G., 1979/1980, Statistical Method in Risk Evaluation, Fire Safety Journal, 2, pp.125-145.
4. FRANTZICH H., 1998, Uncertainty and Risk Analysis in Fire Safety Engineering, Ph.D. Dissertation.
5. BECK, V. R., 1987, A Cost-Effective Decision-Making Model for Building Fire Safety and Protection. Fire Safety Journal, 12, pp. 121-138.
6. TAKEDA, H. and YUNG, D., 1992, Simplified Fire Growth Models for Risk-Cost Assessment in Apartment Buildings, Journal of Fire Protection Engineering, 4 (2) pp. 53-66.
7. YUNG, D. and BECK, V. R., 1995, Building Fire Safety Risk Analysis, Society of Fire Protection Engineers Handbook of Fire Protection Engineering, 2nd Edition, National Fire Protection Association, Quincy, MA, pp. 5-95 to 5-101.
8. COOPER, J. and YUNG, D., 1997, NRCC - CNRC Fire Growth Model for Apartment Buildings. This is an internal report of the Institute for Research in Construction, Internal Report No. 734.
9. FRIEDMAN, R., 1990, Survey of Computer Models for Fire and Smoke, Factory Mutual Research Corp., Norwood, MA, 02062.
10. HADJISOPHOCLEOUS, G. V. and YUNG, D.,1992, A Model for Calculating the Probabilities of Smoke Hazard from Fire in Multi - Storey Buildings, Journal of Fire Protection Engineering, pp. 67-80.
11. RAMACHANDRAN G., 1991, Non- Deterministic Modelling of Fire Spread, Journal of Fire Protection Engineering, 3 (2), pp. 37-48.

12. LING, T. W. and WILLIAMSON, R. B.,1986, The Use of Probabilistic Network for Analysis of Smoke Spread and the Egress of People in Buildings, Fire Safety Science – Proceedings of the First International Symposium, pp. 953 – 962.

13. LING, T. W. and WILLIAMSON, R. B., 1985, Modelling of Fire Spread through Probabilistic Networks, Fire Safety Journal, 9, pp. 287 - 300.

14. BECK, V. R., YUNG, D., HE, Y., and SUMATHIPALA, K., 1996, Experimental Validation of a Fire Growth Model. Proc. INTERFLAM '96 Conference. Cambridge, England.

15. HASOFER, A. M. and BECK, V. R., 1997, A Stochastic Model for Fire Spread in a Compartment. Fire Safety Journal, 28, pp. 207 - 225.

16. BEARD ALAN N., 1997, Fire Models and Design, Fire Safety Journal 28, pp. 117-138.

17. RAMACHANDRAN, G., 1995, Stochastic Models of Fire growth. SFPE handbook of Fire Protection Engineering, Second Edition, National Fire Protection Association, USA, Section 3, Chapter 15, pp. 296-311.

18. HE, Y.,1998, CESARE – RISK: Fire Growth and Smoke Spread Models, Volume 1, Fire Code Reform Centre Ltd.

19. Fire Engineering Guidelines, Fire Code Reform Centre Limited March 1996 ISBN 0 7337 04549.

20. PEEBLES, Jr. P. Z., 1993, Probability, Random variables, and Random Signals Principles, 3rd Ed., McGraw-Hill International Ed.

21. TAT M. A., 1999, Stochastic Modelling and Optimal Control of Compartment Fires, Ph.D. Dissertation.

22. HASOFER A. M., BECK V. R. & ODIGIE D., 1998, Stochastic Fire Spread Over a Network, Integrated Risk Assessment, Proceedings of the Third Conference on Integrated Risk Assessment Newcastle/ N.S.W./AUSTRALIA, pp. 101-106.

23. HASOFER A. M. and ODIGIE D. O., 2000, Stochastic Modelling for Occupant Safety in a Building Fire, Fire Safety Journal, 36 (2001), pp. 269 - 289.

24. MAGNUSSON, S. E., FRANTZICH, H. & HARADA, K., 1996, Fire Safety Design Based on Calculations: Uncertainty Analysis and Safety Verification, Fire Safety Journal 27 (1996), pp. 305-334.

25. WALTON, W. D. and THOMAS, P. H., 1990, Estimating Temperatures in Compartment Fire, SFPE Handbook of Fire Protection Engineering, First Edition, pp. 2.16-32.

26. ZHAO, L. and BECK, V. R., 1997, The Definition of Scenarios for the CESARE-RISK Model. Fire Safety Science-Proc. of the 5th Int. Symp., pp. 655-666.
27. LAPIN, LAWRENCE L., 1983, Probability and Statistics for Modern Engineering, Wadsworth, Inc. California, USA.
28. BEARD ALAN N., 1983, A Stochastic Model for the Number of Deaths Resulting from a Fire in a Bay in a Hospital Ward, Fire Safety Journal, 6 (1983), pp. 121-128.
29. BEARD ALAN N., 1984, A Stochastic Model for the Number of Deaths in a Fire: Further Considerations, Fire Safety Journal, 8 (1984/85) pp. 201-226.
30. BEARD ALAN, 1992, Evaluation of Deterministic Fire Models: Part I—Introduction, Fire Safety Journal 19 (1992), pp. 295-306.
31. BEARD ALLAN, DRYSDALE DOUGAL, HOLBOM PAUL, and BISHOP STEVEN, 1994, A Model of Instability and Flashover, Journal of Applied Fire Science, Volume 4, Number 1, 1994-95.
32. TAT MEHMET A., HASOFER ABRAHAM M., 1995, Modeling the Spread of Fire Using Non-Stationary Stochastic Processes, Integrated Risk Assessment, pp. 215-222.
33. YUNG, D. and BECK, V. R., 1995, Building Fire Safety Risk Analysis, Society of Fire Protection Engineers Handbook of Fire Protection Engineering, 2nd Edition, National Fire Protection Association, Quincy, MA, pp. 5-95 to 5-101.
34. BREIMAN, L. and FRIEDMAN, J. H., 1985, Estimating Optimal Transformations for Multiple Regression and Correlation, Journal of the American Statistical Association, September 1985, Vol. 80, No. 391, Theory and Methods, pp. 580 - 619.
35. TIBSHIRANI, R., 1988, Estimating Transformations for Regression Via Additivity and Variance Stabilization, , Journal of the American Statistical Association, June 1988, Vol. 83,No. 402, Theory and Methods, pp. 394-405.
36. HASOFER, A. M. and LIND, N. C., 1974, Exact and Invariant Second-moment Code Format, J. Eng. Mech. Div., ASCE, 100, EM1, pp. 111-121.
37. MELCHERS, R. E., 1987, Structural Reliability: Analysis and Prediction, Ellis Horwood Limited ISBN 0-85312-930-4, pp. 104-141.
38. FRANTZITCH,H., MAGNUSSON, S. E., HOLMQUIST, B., and RYDEN, J., 1997, Derivation of Partial Safety Factors for the Fire Safety Evaluation Using the Reliability Index Method, International Association for Fire Safety Science Yuji

Hasemi, editor. Fifth International Symposium on Fire Safety Science, pp. 667-678.

39. DITLEVSEN, O. and MADSEN, H. O., 1996, Structural Reliability Methods. Wiley.
40. BREITUNG, K. and FARAVELLI, L., 1996, Response Surface Methods and Asymptotic Approximations. Chapter 5 of Mathematical Models for Structural Reliability Analysis (F. Cascial and J. B. Roberts eds.) CRC Mathematical Modelling Series, pp. 227-285.
41. FRIEDMAN, R., 1992, An International Survey of Computer Models for Fire and Smoke, Journal of Fire Protection Engineering, 4 (3), 1992, pp. 81-92.
42. HASOFER, A. M. and BECK, V. R., 2000, The Probability of Death in the Room of Fire Origin: An Engineering Formula, Journal of Fire Protection Engineering, Vol. 10 (4), 2000, pp. 19-28.
43. MYERS, R. H., 1971, Response Surface Methodology, Allyn and Bacon, Boston.
44. BOX, G. E. P. and DRAPER, N. R.,1987, Empirical Model Building and Response Surface, Wiley, New York.
45. FARAVELLI, L., 1989, A Response Surface Approach for Reliability Analysis. Journal of the Engineering Mechanics Division ASCE, 115 (12), pp. 2763-2781.
46. PAPOULIS, A. 1984, Probability, Random Variables and Stochastic Processes, Published by McGraw-Hill, U.S.A., pp. 535-553.
47. ENGELUND, S. and RZCKWITZ, R., 1992, Experiences with Experimental Design Schemes for Failure Surface Estimation and Reliability, Proceedings of the Sixth Specialty Conference on Probabilistic Mechanics and Structural and Geotechnical Reliability, ASCE, pp. 252-255.
48. BECKER, N. G., 1989, Analysis of Infectious Diseases Data, Chapman and Hall, London, New York.
49. STECKLER, K. D., QUINTIERE, J. G. and RINKINEN, W., 1983, Flow Induced by Fire in a Compartment, NBSIR 82-2520, Natl. Standards and Tech., Gaithersberg, MD, 1983.
50. MYERS, R. H., 1971, Response Surface Methodology, Allyn and Bacon, Boston.
51. FARAVELLI, L., 1989, Response Surface Approach for Reliability Analysis, J. of Engng. Mech., 115 (12) (December 1989).

52. KHURI, A. I. and CORNELL, J. A., 1987, Response Surfaces, Designs and Analyses, Statistics Textbooks and Monographs series, Volume 81, Marcel Dekker, New York.

53. PETERSEN, R. G., 1985, Design and Analysis of Experiments, Marcel Dekker, New York.

54. BUCHER, C. G. and BOURGUND, U., 1990, A Fast and Efficient Response Surface Approach for Structural Reliability Problems, Structural Safety, 7(1990) pp. 57-66.

55. RAJASHEKHAR, M. R. and ELLINGWOOD, B. R., 1993, A New Look at the Response Surface Approach for Reliability Analysis, Structural Safety, 12(1993) pp. 205-220.

56. S-PLUS for Windows User's Manual, Version 3.1, Seattle: Statistical Sciences, Inc., 1993.

57. BOX, G. E. P. and COX, D. R., 1964, An Analysis of Transformations (with discussion). Journal of the Royal Statistical Society, B, 26, pp. 211-246.

58. BOX, G. E. P. and TIDWELL, P. W., 1962, Transformations of Independent Variables. Technometrics 4, pp. 531-550.

59. WEISBERG, S., 1985, Applied Linear Regression, 2nd edition, New York, John Wiley.

60. STEWART, R. D., PETERSON, J. E., FISHER, T. N., HOSKO, M. J., BARETTA, E. D., DODD, H. C. and HERMANN, A. A., 1973, Archives of Environmental Health, 26, pp. 1-7.

61. BABRAUSKAS, V., 1979, Technical Note 1103, National Bureau of Standards, Washington.

62. PURSER, D. A., 1995, Toxicity Assessment of Combustion Products, SFPE Handbook of Fire Protection Engineering. Society of Fire Protection Engineers. 2nd Edition.

63. SANABRIA, L. A. and LI, S., 1998, CESARE-RISK: Evacuation Model-Computer Program and Results, Report of FCRC Project 4, Victoria University of Technology & Fire Reform Centre Ltd.

64. PIPES, L. A. and HARVILL, L. R., 1971, Applied Mathematics for Engineers and Physicists, Third Edition, McGraw-Hill International Book Company, ISBN 0-07-085577-3.

65. HASOFER, A. M. and QU, J., 2002, Response Surface Modelling of Monte Carlo Fire Data, Fire Safety Journal, 37, pp. 772 – 784.

66. SEARLE, S.R. , 1987, Linear Models for Unbalanced Data, Wiley.

67. STAUDTE, R.G. and SHEATHER, S.J., 1990, Robust estimation and testing, Wiley.

68. AKSIT, M., MOSS, J.B. and RUBINI, P.A., 2001, Field Modelling of Surface Flame Spread over Charring Materials, Proceedings of the Ninth International Conference – INTERFLAM 2001. Interscience Communications Ltd., pp. 1459 - 1464. ISBN 0-9532312-7-5.

69. AKSIT, M., MACKIE, P. and RUBINI, P.A., 2000, Coupled Radiative Heat Transfer and Flame Spread Simulation in a Compartment, 3rd International Seminar on Flame and Explosion Hazards, Windermere, UK.

70. KARLSSON, B. and QUINTIERE, J.G., 1999, Enclosure Fire Dynamics, Boca Raton, FL: CRC Press.

71. RAMACHANDRAN, G., 1988, Probabilistic Approach to Fire Risk Evaluation, Fire Technology, 24, 3, pp. 204 - 226.

72. RAMACHANDRAN, G., 1995, Probability – Based Building Design for Fire Safety, Fire Technology, Part 1, 31, 3, pp. 265 – 275; Part 2, 31, 4, pp. 355-368.

73. RAMACHANDRAN, G., 1990, Probability – Based Fire Safety Code, Journal of Fire Protection Engineering, 2, 3, pp. 75 – 91.

74. RAMACHANDRAN, G., 1993, Probabilistic Evaluation of Design Evacuation Time, Proceedings of the CIB W 14 International Symposium on Fire Safety Engineering, University of Ulster, Part 1, pp. 189 – 207.

75. RAMACHANDRAN, G., 1998, Probabilistic Evaluation of Structural Fire Protection – A Simplified Guide Fire Note 8. Fire Research Station, Building Research Establishment, UK.

APPENDIX

FUNCTIONS

A.1 Functions for finding values of regression coefficients

```
reslm<-function(x, y)
{
   #reslm:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 + x41 + x42 + x51 + x52 +
   x71 + x72 + x81 + x82)
   res
}
```

```
resUlm<-function(x,y)
{
   #reslmU:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   resU<-lm(y~x11+x12+x21+x22+x31+x32+x71+x72+x81+x82)
   resU
}
```

A.2 Functions for calculating predicted values, correlations and draw scatterplots

```
YtG1andLA<-function(x, y)
{
   #YtG1andL:calculate predicted values:with or without log to U13:
   #x6 is ignored:
   #reslm:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 +
      x31 + x32 + x41 + x42 + x51 +
      x52 + x71 + x72 + x81 + x82)
   reslm(x, y)$coef
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(
      x, y)$coef[4], reslm(x, y)$
      coef[6], reslm(x, y)$coef[
      8])
   bb <- c(bb, reslm(x, y)$coef[10], reslm(
      x, y)$coef[12], reslm(x, y)$
      coef[14])
   cc <- c(reslm(x, y)$coef[3], reslm(
      x, y)$coef[5], reslm(x, y)$
      coef[7], reslm(x, y)$coef[
      9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(
      x, y)$coef[13], reslm(x, y)$
      coef[15])
   yt<-
   aa+bb[1]*x11+bb[2]*x21+bb[3]*x31+bb[4]*x41+bb[5]*x51+bb[6]*x71+bb[7]*x81
   ytG1<-
   yt+cc[1]*x12+cc[2]*x22+cc[3]*x32+cc[4]*x42+cc[5]*x52+cc[6]*x72+cc[7]*x82

   dd <- c(lsfit(cbind(ytG1, ytG1^2, ytG1^
      3), y)$coef)
   ytG1f <- dd[1] + dd[2] * ytG1 + dd[
      3] * ytG1^2 + dd[4] * ytG1^
      3
   correlation1<-cor(ytG1,y)
   correlation2<-cor(ytG1f,y)

   x <- U13[, 1:8]
   y <- log(U13[, 9])
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 +
      x31 + x32 + x41 + x42 + x51 +
      x52 + x71 + x72 + x81 + x82)
   reslm(x, y)$coef
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(
      x, y)$coef[4], reslm(x, y)$
      coef[6], reslm(x, y)$coef[
      8])
   bb <- c(bb, reslm(x, y)$coef[10], reslm(
      x, y)$coef[12], reslm(x, y)$
      coef[14])
   cc <- c(reslm(x, y)$coef[3], reslm(
      x, y)$coef[5], reslm(x, y)$
      coef[7], reslm(x, y)$coef[
      9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(
      x, y)$coef[13], reslm(x, y)$
      coef[15])
   ytL<-
   aa+bb[1]*x11+bb[2]*x21+bb[3]*x31+bb[4]*x41+bb[5]*x51+bb[6]*x71+bb[7]*x81
   yt1L<-
   ytL+cc[1]*x12+cc[2]*x22+cc[3]*x32+cc[4]*x42+cc[5]*x52+cc[6]*x72+cc[7]*x82
   yt1Le <- exp(yt1L)
   ddL <- c(lsfit(cbind(yt1Le, yt1Le^
      2, yt1Le^3), U13[, 9])$coef)
   yt1Lef <- ddL[1] + ddL[2] * yt1Le +
      ddL[3] * yt1Le^2 + ddL[4] *
      yt1Le^3
   correlation3<-cor(yt1Le,U13[,9])
   correlation4<-cor(yt1Lef,U13[,9])
   graphics.off()
   win.graph()
   plot(U13[,9],yt1Le,xlab="T",ylab="Tt")
   win.graph()
   plot(U13[,9],yt1Lef,xlab="T",ylab="Tp")
   graphics.off()
   win.graph()
   plot(ytG1f, yt1Lef, xlim = c(100, 800),
      ylim = c(100, 800), type = "l")
}
```

```
YtG2andLA<-function(x, y)

{
   #YtG2andL:calculate predicted values:with or without log to U23:
   #x6 is ignored:
   #reslm:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
```

```
x51 <- x[, 5]
x52 <- x51^2
x61 <- x[, 6]
x62 <- x61^2
x71 <- x[, 7]
x72 <- x71^2
x81 <- x[, 8]
x82 <- x81^2
res <- lm(y ~ x11 + x12 + x21 + x22 +
   x31 + x32 + x41 + x42 + x51 +
   x52 + x71 + x72 + x81 + x82)
reslm(x, y)$coef
aa <- reslm(x, y)$coef[1]
bb <- c(reslm(x, y)$coef[2], reslm(
   x, y)$coef[4], reslm(x, y)$
   coef[6], reslm(x, y)$coef[
   8])
bb <- c(bb, reslm(x, y)$coef[10], reslm(
   x, y)$coef[12], reslm(x, y)$
   coef[14])
cc <- c(reslm(x, y)$coef[3], reslm(
   x, y)$coef[5], reslm(x, y)$
   coef[7], reslm(x, y)$coef[
   9])
cc <- c(cc, reslm(x, y)$coef[11], reslm(
   x, y)$coef[13], reslm(x, y)$
   coef[15])
yt<-
aa+bb[1]*x11+bb[2]*x21+bb[3]*x31+bb[4]*x41+bb[5]*x51+bb[6]*x71+bb[7]*x81
ytG2<-
yt+cc[1]*x12+cc[2]*x22+cc[3]*x32+cc[4]*x42+cc[5]*x52+cc[6]*x72+cc[7]*x82

dd <- c(lsfit(cbind(ytG2, ytG2^2, ytG2^
   3), y)$coef)
ytG2f <- dd[1] + dd[2] * ytG2 + dd[
   3] * ytG2^2 + dd[4] * ytG2^
   3
x <- U23[, 1:8]
y <- log(U23[, 9])
x11 <- x[, 1]
x12 <- x11^2
x21 <- x[, 2]
x22 <- x21^2
x31 <- x[, 3]
x32 <- x31^2
x41 <- x[, 4]
x42 <- x41^2
x51 <- x[, 5]
x52 <- x51^2
x61 <- x[, 6]
x62 <- x61^2
x71 <- x[, 7]
x72 <- x71^2
x81 <- x[, 8]
x82 <- x81^2
res <- lm(y ~ x11 + x12 + x21 + x22 +
   x31 + x32 + x41 + x42 + x51 +
   x52 + x71 + x72 + x81 + x82)
reslm(x, y)$coef
aa <- reslm(x, y)$coef[1]
bb <- c(reslm(x, y)$coef[2], reslm(
   x, y)$coef[4], reslm(x, y)$
   coef[6], reslm(x, y)$coef[
   8])
bb <- c(bb, reslm(x, y)$coef[10], reslm(
   x, y)$coef[12], reslm(x, y)$
   coef[14])
cc <- c(reslm(x, y)$coef[3], reslm(
      x, y)$coef[5], reslm(x, y)$
      coef[7], reslm(x, y)$coef[
      9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(
      x, y)$coef[13], reslm(x, y)$
      coef[15])
   ytL<-
   aa+bb[1]*x11+bb[2]*x21+bb[3]*x31+bb[4]*x41+bb[5]*x51+bb[6]*x71+bb[7]*x81
   yt2L<-
   ytL+cc[1]*x12+cc[2]*x22+cc[3]*x32+cc[4]*x42+cc[5]*x52+cc[6]*x72+cc[7]*x82
   yt2Le <- exp(yt2L)
   ddL <- c(lsfit(cbind(yt2Le, yt2Le^
      2, yt2Le^3), U23[, 9])$coef)
   yt2Lef <- ddL[1] + ddL[2] * yt2Le +
      ddL[3] * yt2Le^2 + ddL[4] *
      yt2Le^3
   correlation3<-cor(yt2Le,U23[,9])
   correlation4<-cor(yt2Lef,U23[,9])
   graphics.off()
   win.graph()
   plot(U23[,9],yt2Le,xlab="T",ylab="Tt")
   win.graph()
   plot(U23[,9],yt2Lef,xlab="T",ylab="Tp")

   win.graph()
   plot(ytG2f, yt2Lef, xlim = c(100, 800),
      ylim = c(100, 800), type = "l")
}
```

```
YtG3andLA<-function(x, y)
{
   #YtG3andL:calculate predicted values:with or without log to U23:
   #x4,x5,x6 are ignored:
   #reslmU:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   resU<-lm(y ~ x11 + x12 + x21 + x22 +
      x31 + x32 + x71 + x72 + x81 +
      x82)
   resUlm(x, y)$coef
   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(
      x, y)$coef[4], resUlm(x, y)$
      coef[6], resUlm(x, y)$coef[
      8], resUlm(x, y)$coef[10])

   cc <- c(resUlm(x, y)$coef[3], resUlm(
      x, y)$coef[5], resUlm(x, y)$
      coef[7], resUlm(x, y)$coef[
      9],resUlm(x, y)$coef[11])
```

```
    yt<-aa+bb[1]*x11+bb[2]*x21+bb[3]*x31+bb[4]*x71+bb[5]*x81
    ytG2<-yt+cc[1]*x12+cc[2]*x22+cc[3]*x32+cc[4]*x72+cc[5]*x82

    dd <- c(lsfit(cbind(ytG2, ytG2^2, ytG2^
        3), y)$coef)
    ytG2f <- dd[1] + dd[2] * ytG2 + dd[
        3] * ytG2^2 + dd[4] * ytG2^
        3
    correlation1<-cor(ytG2,y)
    correlation2<-cor(ytG2f,y)

    x <- U33[, 1:8]
    y <- log(U33[, 9])
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x41 <- x[, 4]
    x42 <- x41^2
    x51 <- x[, 5]
    x52 <- x51^2
    x61 <- x[, 6]
    x62 <- x61^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2

    aa <- resUlm(x, y)$coef[1]
    bb <- c(resUlm(x, y)$coef[2], resUlm(
        x, y)$coef[4], resUlm(x, y)$
        coef[6], resUlm(x, y)$coef[
        8], resUlm(x, y)$coef[10])

    cc <- c(resUlm(x, y)$coef[3], resUlm(
        x, y)$coef[5], resUlm(x, y)$
        coef[7], resUlm(x, y)$coef[
        9], resUlm(x, y)$coef[11])

    ytL<-aa+bb[1]*x11+bb[2]*x21+bb[3]*x31+bb[4]*x71+bb[5]*x81
    yt2L<-ytL+cc[1]*x12+cc[2]*x22+cc[3]*x32+cc[4]*x72+cc[5]*x82
    yt2Le <- exp(yt2L)
    ddL <- c(lsfit(cbind(yt2Le, yt2Le^
        2, yt2Le^3), U33[, 9])$coef)
    yt2Lef <- ddL[1] + ddL[2] * yt2Le +
        ddL[3] * yt2Le^2 + ddL[4] *
        yt2Le^3
    correlation3<-cor(yt2Le,U33[,9])
    correlation4<-cor(yt2Lef,U33[,9])
    graphics.off()
    win.graph()
    plot(U33[,9],yt2Le,xlab="T",ylab="Tt")
    win.graph()
    plot(U33[,9],yt2Lef,xlab="T",ylab="Tp")

    win.graph()
    plot(ytG2f, yt2Lef, xlim = c(100, 800),
        ylim = c(100, 800), type = "l")
}
```

```
YtG4andLA<-function(x, y)
{
   #YtG4andL:calculate predicted values:with or without log to U43:
   #x4,x5,x6 are ignored:
   #reslmU:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   resU<-lm(y ~ x11 + x12 + x21 + x22 +
      x31 + x32 + x71 + x72 + x81 +
      x82)
   resUlm(x, y)$coef
   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(
      x, y)$coef[4], resUlm(x, y)$
      coef[6], resUlm(x, y)$coef[
      8], resUlm(x, y)$coef[10])

   cc <- c(resUlm(x, y)$coef[3], resUlm(
      x, y)$coef[5], resUlm(x, y)$
      coef[7], resUlm(x, y)$coef[
      9],resUlm(x, y)$coef[11])

   yt<-aa+bb[1]*x11+bb[2]*x21+bb[3]*x31+bb[4]*x71+bb[5]*x81
   ytG2<-yt+cc[1]*x12+cc[2]*x22+cc[3]*x32+cc[4]*x72+cc[5]*x82

   dd <- c(lsfit(cbind(ytG2, ytG2^2, ytG2^
      3), y)$coef)
   ytG2f <- dd[1] + dd[2] * ytG2 + dd[
      3] * ytG2^2 + dd[4] * ytG2^
      3
   correlation1<-cor(ytG2,y)
   correlation2<-cor(ytG2f,y)

   x <- U43[, 1:8]
   y <- log(U43[, 9])
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2

   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(
      x, y)$coef[4], resUlm(x, y)$
      coef[6], resUlm(x, y)$coef[
      8], resUlm(x, y)$coef[10])

   cc <- c(resUlm(x, y)$coef[3], resUlm(
      x, y)$coef[5], resUlm(x, y)$
      coef[7], resUlm(x, y)$coef[
      9], resUlm(x, y)$coef[11])

   ytL<-aa+bb[1]*x11+bb[2]*x21+bb[3]*x31+bb[4]*x71+bb[5]*x81
   yt2L<-ytL+cc[1]*x12+cc[2]*x22+cc[3]*x32+cc[4]*x72+cc[5]*x82
   yt2Le <- exp(yt2L)
   ddL <- c(lsfit(cbind(yt2Le, yt2Le^
      2, yt2Le^3), U43[, 9])$coef)
   yt2Lef <- ddL[1] + ddL[2] * yt2Le +
      ddL[3] * yt2Le^2 + ddL[4] *
      yt2Le^3
   correlation3<-cor(yt2Le,U43[,9])
   correlation4<-cor(yt2Lef,U43[,9])

   graphics.off()
   win.graph()
   plot(U43[,9],yt2Le,xlab="T",ylab="Tt")
   win.graph()
   plot(U43[,9],yt2Lef,xlab="T",ylab="Tp")

   win.graph()
   plot(ytG2f, yt2Lef, xlim = c(100, 800),
      ylim = c(100, 800), type = "l")
}
```

A.3 Functions for finding reliability index

```
BetaG1<-function(x, y)
{
   #BetaG1:calculate lambda, beta to U13:
   #x6 is ignored:
   #reslm:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 +
      x31 + x32 + x41 + x42 + x51 +
      x52 + x71 + x72 + x81 + x82)
   reslm(x, y)$coef
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(
      x, y)$coef[4], reslm(x, y)$
      coef[6], reslm(x, y)$coef[
      8])
   bb <- c(bb, reslm(x, y)$coef[10], reslm(
      x, y)$coef[12], reslm(x, y)$
      coef[14])
   cc <- c(reslm(x, y)$coef[3], reslm(
      x, y)$coef[5], reslm(x, y)$
      coef[7], reslm(x, y)$coef[
      9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(
      x, y)$coef[13], reslm(x, y)$
      coef[15])
   #calculate reliability index beta T=240sec:
   ymin <- 311.0956
   input <- c(600, 450, 250, 0.7, 0.6,
      50, 0.6, 0.7)
   constant <- aa + bb[1] * input[1] +
      bb[3] * input[3] + bb[4] *
      input[4]
   constant <- constant + bb[5] * input[
      5] + bb[6] * input[7]
   constant <- constant + cc[1] * input[
      1]^2 + cc[3] * input[3]^2 +
      cc[4] * input[4]^2
   constant <- constant + cc[5] * input[
      5]^2 + cc[6] * input[7]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[
      2] * sigma1
   A8 <- cc[7] * sigma2^2
   B8 <- 2 * cc[7] * sigma2 * mu2 + bb[
      7] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[
      7] * mu2^2 + bb[7] * mu2 +
      constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
   #calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 -
      2 * B7^2 * A7 * A8^2 - 2 * B8^
      2 * A8 * A7^2 + A7 * A8^2 *
      B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^
      2 + 2 * A7^2 * A8) - 2 * B7^
      2 * (A8^2 + 2 * A7 * A8) - 2 *
      B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 *
      B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^
      2 + 4 * A7 * A8) - 2 * B7^
      2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^
      2 + A8 * B8^2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 *
      A8) - (2 * B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2,
      D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(lambda, beta), digits = 8)
}
```

```
BetaG2<-function(x, y)
{
   #BetaG2:calculate lambda, beta to U23:
   #x6 is ignored:
   #reslm:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 +
      x31 + x32 + x41 + x42 + x51 +
      x52 + x71 + x72 + x81 + x82)
   reslm(x, y)$coef
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(
    x, y)$coef[4], reslm(x, y)$
    coef[6], reslm(x, y)$coef[
    8])
bb <- c(bb, reslm(x, y)$coef[10], reslm(
    x, y)$coef[12], reslm(x, y)$
    coef[14])
cc <- c(reslm(x, y)$coef[3], reslm(
    x, y)$coef[5], reslm(x, y)$
    coef[7], reslm(x, y)$coef[
    9])
cc <- c(cc, reslm(x, y)$coef[11], reslm(
    x, y)$coef[13], reslm(x, y)$
    coef[15])
#calculate reliability index beta T=320sec:
ymin <- 330.9244
input <- c(600, 450, 250, 0.7, 0.6,
    50, 0.6, 0.7)
constant <- aa + bb[1] * input[1] +
    bb[3] * input[3] + bb[4] *
    input[4]
constant <- constant + bb[5] * input[
    5] + bb[6] * input[7]
constant <- constant + cc[1] * input[
    1]^2 + cc[3] * input[3]^2 +
    cc[4] * input[4]^2
constant <- constant + cc[5] * input[
    5]^2 + cc[6] * input[7]^2
constant
#cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
sigma1 <- 20
mu1 <- 450
sigma2 <- 0.1
mu2 <- 0.7
A7 <- cc[2] * sigma1^2
B7 <- 2 * cc[2] * sigma1 * mu1 + bb[
    2] * sigma1
A8 <- cc[7] * sigma2^2
B8 <- 2 * cc[7] * sigma2 * mu2 + bb[
    7] * sigma2
C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[
    7] * mu2^2 + bb[7] * mu2 +
    constant
#ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
D0 <- 4 * (C - ymin) * A7^2 * A8^2 -
    2 * B7^2 * A7 * A8^2 - 2 * B8^
    2 * A8 * A7^2 + A7 * A8^2 *
    B7^2 + A7^2 * A8 * B8^2
D1 <- 4 * (C - ymin) * (2 * A7 * A8^
    2 + 2 * A7^2 * A8) - 2 * B7^
    2 * (A8^2 + 2 * A7 * A8) - 2 *
    B8^2 * (A7^2 + 2 * A7 * A8) +
    2 * (A7 * A8 * B7^2 + A7 * A8 *
    B8^2)
D2 <- 4 * (C - ymin) * (A8^2 + A7^
    2 + 4 * A7 * A8) - 2 * B7^
    2 * (2 * A8 + A7) - 2 * B8^
    2 * (2 * A7 + A8) + A7 * B7^
    2 + A8 * B8^2
D3 <- 4 * (C - ymin) * (2 * A7 + 2 *
    A8) - (2 * B7^2 + 2 * B8^2)
D4 <- 4 * (C - ymin)
lambda <- Re(polyroot(c(D0, D1, D2,
    D3, D4)))
u7 <-  - B7/(2 * (lambda + A7))
u8 <-  - B8/(2 * (lambda + A8))
beta <- (u8^2 + u7^2)^(1/2)
round(rbind(lambda, beta), digits = 8)
```

```
}







BetaG3<-function(x, y)
{
#BetaG3:calculate lambda,beta for G3:x4 to x6 are ignored:
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
      4], resUlm(x, y)$coef[6], resUlm(x, y)$
      coef[8], resUlm(x, y)$coef[10])
   cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
      5], resUlm(x, y)$coef[7], resUlm(x, y)$
      coef[9], resUlm(x, y)$coef[11])
   #calculate reliability index beta for Tp=230sec:
   ymin <-  237.3830
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
      0.7)
   constant <- aa + bb[1] * input[1] + bb[4] *
      input[7] + bb[3] * input[3]
   constant <- constant + cc[1] * input[1]^2 + cc[
      4] * input[7]^2 + cc[3] * input[3]^2
   constant
   #cc[2]*x[,7]^2+bb[2]*x[,7]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[5] * sigma2^2
   B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
      2 + bb[5] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
      A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
      A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
      2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
      A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
      A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
      2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
      B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
```

```
    beta <- (u8^2 + u7^2)^(1/2)
    round(rbind(lambda, beta), digits = 4)
    #choose smallest beta
}
```

```
BetaG4<-function(x, y)
{
#BetaG4:calculate lambda,beta for G3:x4 to x6 are ignored:
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    aa <- resUlm(x, y)$coef[1]
    bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
        4], resUlm(x, y)$coef[6], resUlm(x, y)$
        coef[8], resUlm(x, y)$coef[10])
    cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
        5], resUlm(x, y)$coef[7], resUlm(x, y)$
        coef[9], resUlm(x, y)$coef[11])
    #calculate reliability index beta Tp=250sec:
    ymin <-  257.1608
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
        0.7)
    constant <- aa + bb[1] * input[1] + bb[4] *
        input[7] + bb[3] * input[3]
    constant <- constant + cc[1] * input[1]^2 + cc[
        4] * input[7]^2 + cc[3] * input[3]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
    A8 <- cc[5] * sigma2^2
    B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
        2 + bb[5] * mu2 + constant
    #ymin=A7*u2^2+B7*u2+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
    D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
        A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
        A8^2 * B7^2 + A7^2 * A8 * B8^2
    D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
        2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
        A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
        2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
    D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
        A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
        2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
        2
    D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
        B7^2 + 2 * B8^2)
    D4 <- 4 * (C - ymin)
    lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
    u7 <-  - B7/(2 * (lambda + A7))
```

```
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(lambda, beta), digits = 4)
   #choose smallest beta
}



BetaG1L<-function(x, y)
{
   #BetaG1L:calculate lambda, beta to U13:
   #x6 is ignored:
   #reslm:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 +
      x31 + x32 + x41 + x42 + x51 +
      x52 + x71 + x72 + x81 + x82)
   reslm(x, y)$coef
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(
      x, y)$coef[4], reslm(x, y)$
      coef[6], reslm(x, y)$coef[
      8])
   bb <- c(bb, reslm(x, y)$coef[10], reslm(
      x, y)$coef[12], reslm(x, y)$
      coef[14])
   cc <- c(reslm(x, y)$coef[3], reslm(
      x, y)$coef[5], reslm(x, y)$
      coef[7], reslm(x, y)$coef[
      9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(
      x, y)$coef[13], reslm(x, y)$
      coef[15])
   #calculate reliability index beta for Tp=217.3738:
   ymin <-  log(301.1548)
   input <- c(600, 450, 250, 0.7, 0.6,
      50, 0.6, 0.7)
   constant <- aa + bb[1] * input[1] +
      bb[3] * input[3] + bb[4] *
      input[4]
   constant <- constant + bb[5] * input[
      5] + bb[6] * input[7]
   constant <- constant + cc[1] * input[
      1]^2 + cc[3] * input[3]^2 +
      cc[4] * input[4]^2
   constant <- constant + cc[5] * input[
      5]^2 + cc[6] * input[7]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[
      2] * sigma1
   A8 <- cc[7] * sigma2^2
   B8 <- 2 * cc[7] * sigma2 * mu2 + bb[
      7] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[
      7] * mu2^2 + bb[7] * mu2 +
      constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
   #calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 -
      2 * B7^2 * A7 * A8^2 - 2 * B8^
      2 * A8 * A7^2 + A7 * A8^2 *
      B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^
      2 + 2 * A7^2 * A8) - 2 * B7^
      2 * (A8^2 + 2 * A7 * A8) - 2 *
      B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 *
      B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^
      2 + 4 * A7 * A8) - 2 * B7^
      2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^
      2 + A8 * B8^2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 *
      A8) - (2 * B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2,
      D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(lambda, beta), digits = 8)
}


BetaG2L<-function(x, y)
{
   #BetaG2L:calculate lambda, beta to U23:
   #x6 is ignored:
   #reslm:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 +
      x31 + x32 + x41 + x42 + x51 +
      x52 + x71 + x72 + x81 + x82)
   reslm(x, y)$coef
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(
      x, y)$coef[4], reslm(x, y)$
      coef[6], reslm(x, y)$coef[
      8])
```

```
    bb <- c(bb, reslm(x, y)$coef[10], reslm(
       x, y)$coef[12], reslm(x, y)$
       coef[14])
    cc <- c(reslm(x, y)$coef[3], reslm(
       x, y)$coef[5], reslm(x, y)$
       coef[7], reslm(x, y)$coef[
       9])
    cc <- c(cc, reslm(x, y)$coef[11], reslm(
       x, y)$coef[13], reslm(x, y)$
       coef[15])
    #calculate reliability index beta for Tp=320:
    ymin <-  log(321.1342)
    input <- c(600, 450, 250, 0.7, 0.6,
       50, 0.6, 0.7)
    constant <- aa + bb[1] * input[1] +
       bb[3] * input[3] + bb[4] *
       input[4]
    constant <- constant + bb[5] * input[
       5] + bb[6] * input[7]
    constant <- constant + cc[1] * input[
       1]^2 + cc[3] * input[3]^2 +
       cc[4] * input[4]^2
    constant <- constant + cc[5] * input[
       5]^2 + cc[6] * input[7]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[
       2] * sigma1
    A8 <- cc[7] * sigma2^2
    B8 <- 2 * cc[7] * sigma2 * mu2 + bb[
       7] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[
       7] * mu2^2 + bb[7] * mu2 +
       constant
    #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
    #calculate lambda,beta
    D0 <- 4 * (C - ymin) * A7^2 * A8^2 -
       2 * B7^2 * A7 * A8^2 - 2 * B8^
       2 * A8 * A7^2 + A7 * A8^2 *
       B7^2 + A7^2 * A8 * B8^2
    D1 <- 4 * (C - ymin) * (2 * A7 * A8^
       2 + 2 * A7^2 * A8) - 2 * B7^
       2 * (A8^2 + 2 * A7 * A8) - 2 *
       B8^2 * (A7^2 + 2 * A7 * A8) +
       2 * (A7 * A8 * B7^2 + A7 * A8 *
       B8^2)
    D2 <- 4 * (C - ymin) * (A8^2 + A7^
       2 + 4 * A7 * A8) - 2 * B7^
       2 * (2 * A8 + A7) - 2 * B8^
       2 * (2 * A7 + A8) + A7 * B7^
       2 + A8 * B8^2
    D3 <- 4 * (C - ymin) * (2 * A7 + 2 *
       A8) - (2 * B7^2 + 2 * B8^2)
    D4 <- 4 * (C - ymin)
    lambda <- Re(polyroot(c(D0, D1, D2,
       D3, D4)))
    u7 <-  - B7/(2 * (lambda + A7))
    u8 <-  - B8/(2 * (lambda + A8))
    beta <- (u8^2 + u7^2)^(1/2)
    round(rbind(lambda, beta), digits = 8)
}
```

```
BetaG3L<-function(x, y)
{
#BetaG3L:calculate lambda,beta for G3:x4 to x6 are ignored:
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    aa <- resUlm(x, y)$coef[1]
    bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
        4], resUlm(x, y)$coef[6], resUlm(x, y)$
        coef[8], resUlm(x, y)$coef[10])
    cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
        5], resUlm(x, y)$coef[7], resUlm(x, y)$
        coef[9], resUlm(x, y)$coef[11])
    #calculate reliability index beta Tp=230:
    ymin <-  log(230.9052)
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
        0.7)
    constant <- aa + bb[1] * input[1] + bb[4] *
        input[7] + bb[3] * input[3]
    constant <- constant + cc[1] * input[1]^2 + cc[
        4] * input[7]^2 + cc[3] * input[3]^2
    constant
    #cc[4]*x[,7]^2+bb[4]*x[,7]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
    A8 <- cc[5] * sigma2^2
    B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
        2 + bb[5] * mu2 + constant
    #ymin=A7*u2^2+B7*u2+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
    D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
        A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
        A8^2 * B7^2 + A7^2 * A8 * B8^2
    D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
        2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
        A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
        2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
    D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
        A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
        2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
        2
    D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
        B7^2 + 2 * B8^2)
    D4 <- 4 * (C - ymin)
    lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
    u7 <-  - B7/(2 * (lambda + A7))
    u8 <-  - B8/(2 * (lambda + A8))
    beta <- (u8^2 + u7^2)^(1/2)
    round(rbind(lambda, beta), digits = 4)
    #choose smallest beta
}


BetaG4L<-function(x, y)
```

```
{
#BetaG4L:calculate lambda,beta for G4:x4 to x6 are ignored in log:
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
      4], resUlm(x, y)$coef[6], resUlm(x, y)$
      coef[8], resUlm(x, y)$coef[10])
   cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
      5], resUlm(x, y)$coef[7], resUlm(x, y)$
      coef[9], resUlm(x, y)$coef[11])
   #calculate reliability index beta Tp=250sec:
   ymin <-  log(250.6776)
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
      0.7)
   constant <- aa + bb[1] * input[1] + bb[4] *
      input[7] + bb[3] * input[3]
   constant <- constant + cc[1] * input[1]^2 + cc[
      4] * input[7]^2 + cc[3] * input[3]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[5] * sigma2^2
   B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
      2 + bb[5] * mu2 + constant
   #ymin=A7*u2^2+B7*u2+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
      A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
      A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
      2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
      A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
      A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
      2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
      B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(lambda, beta), digits = 4)
   #choose smallest beta
}
```

A.4 Functions for drawing figures to illustrate reliability index and response surface for specific examples

```
StateU1<-function(x, y)
{
#StateU1:lambda, beta,constraint,design point D;
#x6 is ignored in U13:
#reslm:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
      x41 + x42 + x51 + x52 + x71 + x72 + x81 +
      x82)
   reslm(x, y)$coef #calculate correlations:
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
      ], reslm(x, y)$coef[6], reslm(x, y)$
      coef[8])
   bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
      coef[12], reslm(x, y)$coef[14])
   cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
      ], reslm(x, y)$coef[7], reslm(x, y)$
      coef[9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
      coef[13], reslm(x, y)$coef[15])
   ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
      x31 + bb[4] * x41 + bb[5] * x51 + bb[6] *
      x71 + bb[7] * x81
   ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
      3] * x32 + cc[4] * x42 + cc[5] * x52 +
      cc[6] * x72 + cc[7] * x82  #ypred
#graphics.off()
#win.graph()
#plot(y, ypred)
#win.graph()
   dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
      )$coef)
   ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
      2 + dd[4] * ypred^3 #plot(y, ypredf)
   correlation1 <- cor(y, ypred)
   correlation2 <- cor(y, ypredf)
   round(rbind(correlation1, correlation2), digits
       = 6)
   #calculate reliability index beta:
   ymin <- 311.0956
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
      0.7)
   constant <- aa + bb[1] * input[1] + bb[6] *
      input[7] + bb[3] * input[3]
   constant <- constant + bb[4] * input[4] + bb[5] *
      input[5]
   constant <- constant + cc[1] * input[1]^2 + cc[
      6] * input[7]^2 + cc[3] * input[3]^2
   constant <- constant + cc[4] * input[4]^2 + cc[
       5] * input[5]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[7] * sigma2^2
   B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
       2 + bb[7] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
       A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
       A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
       2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
       A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
       2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
       A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
       2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
       2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
       B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(beta, lambda), digits = 4)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   U7 <- seq(-10, 50, 0.1)
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
       C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U92, U91)
   U8f <- cbind(U8, U9)
   graphics.off()
   win.graph()
   par(pty = "s")
   matplot(U7, U8f, col = 1, xlim = c(-10, 50),
       ylim = c(-20, 40), type = "l", xlab = "u2", ylab = "u8")
   win.graph()
   par(pty = "s")
   matplot(U7, U8f, col = 1, xlim = c(-5, 5), ylim
        = c(-5, 5), type =    "l", xlab = "u2", ylab = "u8")
}
```

```
StateU2<-function(x, y)
{
#StateU2:lambda, beta,constraint,design point D;
#x6 is ignored in U23:
```

```
#reslm:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
      x41 + x42 + x51 + x52 + x71 + x72 + x81 +
      x82)
   reslm(x, y)$coef #calculate correlations:
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
      ], reslm(x, y)$coef[6], reslm(x, y)$
      coef[8])
   bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
      coef[12], reslm(x, y)$coef[14])
   cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
      ], reslm(x, y)$coef[7], reslm(x, y)$
      coef[9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
      coef[13], reslm(x, y)$coef[15])
   ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
      x31 + bb[4] * x41 + bb[5] * x51 + bb[6] *
      x71 + bb[7] * x81
   ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
      3] * x32 + cc[4] * x42 + cc[5] * x52 +
      cc[6] * x72 + cc[7] * x82  #ypred
#graphics.off()
#win.graph()
#plot(y, ypred)
#win.graph()
   dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
      )$coef)
   ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
      2 + dd[4] * ypred^3 #plot(y, ypredf)
   correlation1 <- cor(y, ypred)
   correlation2 <- cor(y, ypredf)
   round(rbind(correlation1, correlation2), digits
       = 6)
   #calculate reliability index beta:
   ymin <- 330.9244
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
      0.7)
   constant <- aa + bb[1] * input[1] + bb[6] *
      input[7] + bb[3] * input[3]
   constant <- constant + bb[4] * input[4] + bb[5] *
      input[5]
   constant <- constant + cc[1] * input[1]^2 + cc[
      6] * input[7]^2 + cc[3] * input[3]^2
   constant <- constant + cc[4] * input[4]^2 + cc[
      5] * input[5]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[7] * sigma2^2
   B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
      2 + bb[7] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
      A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
      A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
      2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
      A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
      A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
      2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
      B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(beta, lambda), digits = 4)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   U7 <- seq(-10, 50, 0.1)
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
      C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U92, U91)
   U8f <- cbind(U8, U9)
   graphics.off()
   win.graph()
   par(pty = "s")
   matplot(U7, U8f, col = 1, xlim = c(-10, 50),
      ylim = c(-20, 40), type = "l", xlab = "u2", ylab = "u8")
   win.graph()
   par(pty = "s")
   matplot(U7, U8f, col = 1, xlim = c(-5, 5), ylim
       = c(-5, 5), type =    "l", xlab = "u2", ylab = "u8")
}
```

```
StateU3<-function(x, y, n)
{
#StateU3:calculate U33:lambda,beta constraint,design point D x4~6 are
   ignored:
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
      4], resUlm(x, y)$coef[6], resUlm(x, y)$
      coef[8], resUlm(x, y)$coef[10])
   cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
      5], resUlm(x, y)$coef[7], resUlm(x, y)$
      coef[9], resUlm(x, y)$coef[11])
   ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
      x31 + bb[4] * x71 + bb[5] * x81
   ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
      3] * x32 + cc[4] * x72 + cc[5] * x82
   #ypred
   graphics.off()
   win.graph()
   plot(y, ypred)
   win.graph()
   dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
      )$coef)
   ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
      2 + dd[4] * ypred^3
   plot(y, ypredf)
   correlation1 <- cor(y, ypred)
   correlation2 <- cor(y, ypredf)
   round(rbind(correlation1, correlation2), digits
       = 6)
   #calculate reliability index beta:
   ymin <- 237.3830
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
      0.7)
   constant <- aa + bb[1] * input[1] + bb[3] *
      input[3] + bb[4] * input[7]
   constant <- constant + cc[1] * input[1]^2 + cc[
      3] * input[3]^2 + cc[4] * input[7]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[2]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[5] * sigma2^2
   B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
      2 + bb[5] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
      A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
      A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
      2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
      A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
      A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
      2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
      B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(lambda, beta), digits = 8)
```

```
    #choose smallest beta
    betas <- min(beta)
    betas  #Find design point:
    U7 <- seq(-10, 60, 0.1)
    delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
       C - ymin))^(1/2)
    U81 <- ( - B8 + delta)/(2 * A8)
    U82 <- ( - B8 - delta)/(2 * A8)
    U8 <- cbind(U82, U81)
    U91 <- (betas^2 - U7^2)^(1/2)
    U92 <-  - (betas^2 - U7^2)^(1/2)
    U9 <- cbind(U91, U92)
    U8f <- cbind(U8, U9)   #win.graph()
#matplot(U7, U9,xlim = c(-15, 15), ylim = c(-15, 15))
#win.graph()
#matplot(U7, U8,xlim = c(-15, 15), ylim = c(-15, 15))
    win.graph()
    par(pty = "s")
    matplot(U7, U8f, col = 1, xlim = c(-10, 60),
       ylim = c(-30, 40), type = "l", xlab = "u2", ylab = "u8")
    win.graph()
    par(pty = "s")
    matplot(U7, U8f, col = 1, xlim = c(-5, 5), ylim
        = c(-5, 5), type =    "l", xlab = "u2", ylab = "u8")
}
```

```
StateU4<-function(x, y, n)
{
#StateU4:calculate U43:lambda,beta constraint,design point D x4~6 are
    ignored:
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    aa <- resUlm(x, y)$coef[1]
    bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
       4], resUlm(x, y)$coef[6], resUlm(x, y)$
       coef[8], resUlm(x, y)$coef[10])
    cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
       5], resUlm(x, y)$coef[7], resUlm(x, y)$
       coef[9], resUlm(x, y)$coef[11])
    ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
       x31 + bb[4] * x71 + bb[5] * x81
    ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
       3] * x32 + cc[4] * x72 + cc[5] * x82
    #ypred
    graphics.off()
    win.graph()
    plot(y, ypred)
    win.graph()
    dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
       )$coef)
    ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
       2 + dd[4] * ypred^3
    plot(y, ypredf)
    correlation1 <- cor(y, ypred)
    correlation2 <- cor(y, ypredf)
    round(rbind(correlation1, correlation2), digits
         = 6)
    #calculate reliability index beta:
    ymin <- 257.1608
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
       0.7)
    constant <- aa + bb[1] * input[1] + bb[3] *
       input[3] + bb[4] * input[7]
    constant <- constant + cc[1] * input[1]^2 + cc[
       3] * input[3]^2 + cc[4] * input[7]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[2]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
    A8 <- cc[5] * sigma2^2
    B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
       2 + bb[5] * mu2 + constant
    #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
    D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
       A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
       A8^2 * B7^2 + A7^2 * A8 * B8^2
    D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
       2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
       A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
       2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
    D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
       A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
       2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
       2
    D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
       B7^2 + 2 * B8^2)
    D4 <- 4 * (C - ymin)
    lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
    u7 <-  - B7/(2 * (lambda + A7))
    u8 <-  - B8/(2 * (lambda + A8))
    beta <- (u8^2 + u7^2)^(1/2)
    round(rbind(lambda, beta), digits = 8)
    #choose smallest beta
    betas <- min(beta)
    betas  #Find design point:
    U7 <- seq(-10, 60, 0.1)
    delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
       C - ymin))^(1/2)
    U81 <- ( - B8 + delta)/(2 * A8)
    U82 <- ( - B8 - delta)/(2 * A8)
    U8 <- cbind(U82, U81)
    U91 <- (betas^2 - U7^2)^(1/2)
    U92 <-  - (betas^2 - U7^2)^(1/2)
    U9 <- cbind(U91, U92)
    U8f <- cbind(U8, U9)   #win.graph()
#matplot(U7, U9,xlim = c(-15, 15), ylim = c(-15, 15))
#win.graph()
#matplot(U7, U8,xlim = c(-15, 15), ylim = c(-15, 15))
    win.graph()
    par(pty = "s")
    matplot(U7, U8f, col = 1, xlim = c(-10, 60),
       ylim = c(-30, 40), type = "l", xlab = "u2", ylab = "u8")
    win.graph()
    par(pty = "s")
    matplot(U7, U8f, col = 1, xlim = c(-5, 5), ylim
         = c(-5, 5), type =    "l", xlab = "u2", ylab = "u8")
}
```

```
StateU1L<-function(x, y)
{
#StateU1L:lambda, beta,constraint,design point D;
#x6 is ignored in U13 with logorithmic fit to Tt:
#reslm:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
      x41 + x42 + x51 + x52 + x71 + x72 + x81 +
      x82)
   reslm(x, y)$coef #calculate correlations:
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
      ], reslm(x, y)$coef[6], reslm(x, y)$
      coef[8])
   bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
      coef[12], reslm(x, y)$coef[14])
   cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
      ], reslm(x, y)$coef[7], reslm(x, y)$
      coef[9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
      coef[13], reslm(x, y)$coef[15])
   ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
      x31 + bb[4] * x41 + bb[5] * x51 + bb[6] *
      x71 + bb[7] * x81
   ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
      3] * x32 + cc[4] * x42 + cc[5] * x52 +
      cc[6] * x72 + cc[7] * x82  #ypred
#graphics.off()
#win.graph()
#plot(y, ypred)
#win.graph()
   dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
      )$coef)
   ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
      2 + dd[4] * ypred^3 #plot(y, ypredf)
   correlation1 <- cor(y, ypred)
   correlation2 <- cor(y, ypredf)
   round(rbind(correlation1, correlation2), digits
       = 6)
   #calculate reliability index beta Tp=300 sec:
   Ttmin<-301.1548
   ymin <- log(Ttmin)
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
      0.7)
   constant <- aa + bb[1] * input[1] + bb[6] *
      input[7] + bb[3] * input[3]
   constant <- constant + bb[4] * input[4] + bb[5] *
      input[5]
   constant <- constant + cc[1] * input[1]^2 + cc[
      6] * input[7]^2 + cc[3] * input[3]^2
   constant <- constant + cc[4] * input[4]^2 + cc[
      5] * input[5]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[7] * sigma2^2
   B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
      2 + bb[7] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
      A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
      A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
      2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
      A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
      A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
      2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
      B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(beta, lambda), digits = 4)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   U7 <- seq(-10, 50, 0.1)
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
      C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U92, U91)
   U8f <- cbind(U8, U9)
   graphics.off()
   win.graph()
   par(pty = "s")
   matplot(U7, U8f, col = 1, xlim = c(-10, 50),
      ylim = c(-20, 40), type = "l", xlab = "u2", ylab = "u8")

   U7 <- seq(-5, 5, 0.1)
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
      C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U92, U91)
   U8f <- cbind(U8, U9)

   win.graph()
```

```
    par(pty = "s")
    matplot(U7, U8f, col = 1, xlim = c(-5, 5), ylim
         = c(-5, 5), type =    "l", xlab = "u2", ylab = "u8")
}
```

```
StateU2L<-function(x, y)
{
#StateU2L:lambda, beta,constraint,design point D;
#x6 is ignored in U23 with logorithmic fit to Tt:
#reslm:calculate the coefficients of X
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x41 <- x[, 4]
    x42 <- x41^2
    x51 <- x[, 5]
    x52 <- x51^2
    x61 <- x[, 6]
    x62 <- x61^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
        x41 + x42 + x51 + x52 + x71 + x72 + x81 +
        x82)
    reslm(x, y)$coef #calculate correlations:
    aa <- reslm(x, y)$coef[1]
    bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
        ], reslm(x, y)$coef[6], reslm(x, y)$
        coef[8])
    bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
        coef[12], reslm(x, y)$coef[14])
    cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
        ], reslm(x, y)$coef[7], reslm(x, y)$
        coef[9])
    cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
        coef[13], reslm(x, y)$coef[15])
    ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
        x31 + bb[4] * x41 + bb[5] * x51 + bb[6] *
        x71 + bb[7] * x81
    ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
        3] * x32 + cc[4] * x42 + cc[5] * x52 +
        cc[6] * x72 + cc[7] * x82  #ypred
#graphics.off()
#win.graph()
#plot(y, ypred)
#win.graph()
    dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
        )$coef)
    ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
        2 + dd[4] * ypred^3 #plot(y, ypredf)
    correlation1 <- cor(y, ypred)
    correlation2 <- cor(y, ypredf)
    round(rbind(correlation1, correlation2), digits
         = 6)
    #calculate reliability index beta Tp=320 sec:
    Ttmin<-321.1342
    ymin <- log(Ttmin)
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
        0.7)
    constant <- aa + bb[1] * input[1] + bb[6] *
        input[7] + bb[3] * input[3]
    constant <- constant + bb[4] * input[4] + bb[5] *
        input[5]
    constant <- constant + cc[1] * input[1]^2 + cc[
        6] * input[7]^2 + cc[3] * input[3]^2
    constant <- constant + cc[4] * input[4]^2 + cc[
        5] * input[5]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
    A8 <- cc[7] * sigma2^2
    B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
        2 + bb[7] * mu2 + constant
    #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
    D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
        A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
        A8^2 * B7^2 + A7^2 * A8 * B8^2
    D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
        2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
        A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
        2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
    D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
        A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
        2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
        2
    D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
        B7^2 + 2 * B8^2)
    D4 <- 4 * (C - ymin)
    lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
    u7 <-  - B7/(2 * (lambda + A7))
    u8 <-  - B8/(2 * (lambda + A8))
    beta <- (u8^2 + u7^2)^(1/2)
    round(rbind(beta, lambda), digits = 4)
    #choose smallest beta
    betas <- min(beta)
    betas  #Find design point:
    U7 <- seq(-10, 50, 0.1)
    delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
        C - ymin))^(1/2)
    U81 <- ( - B8 + delta)/(2 * A8)
    U82 <- ( - B8 - delta)/(2 * A8)
    U8 <- cbind(U82, U81)
    U91 <- (betas^2 - U7^2)^(1/2)
    U92 <-  - (betas^2 - U7^2)^(1/2)
    U9 <- cbind(U92, U91)
    U8f <- cbind(U8, U9)
    graphics.off()
    win.graph()
    par(pty = "s")
    matplot(U7, U8f, col = 1, xlim = c(-10, 50),
        ylim = c(-20, 40), type = "l", xlab = "u2", ylab = "u8")

    U7 <- seq(-5, 5, 0.1)
    delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
        C - ymin))^(1/2)
    U81 <- ( - B8 + delta)/(2 * A8)
    U82 <- ( - B8 - delta)/(2 * A8)
    U8 <- cbind(U82, U81)
    U91 <- (betas^2 - U7^2)^(1/2)
    U92 <-  - (betas^2 - U7^2)^(1/2)
    U9 <- cbind(U92, U91)
    U8f <- cbind(U8, U9)

    win.graph()
    par(pty = "s")
    matplot(U7, U8f, col = 1, xlim = c(-5, 5), ylim
        = c(-5, 5), type =    "l", xlab = "u2", ylab = "u8")
 }




 StateU3L<-function(x, y, n)
 {
 #StateU3:calculate U33:lambda,beta constraint,design point D x4~6 are
    ignored:
 #with logorithmic fit to Tt:
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    aa <- resUlm(x, y)$coef[1]
    bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
       4], resUlm(x, y)$coef[6], resUlm(x, y)$
       coef[8], resUlm(x, y)$coef[10])
    cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
       5], resUlm(x, y)$coef[7], resUlm(x, y)$
       coef[9], resUlm(x, y)$coef[11])
    ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
       x31 + bb[4] * x71 + bb[5] * x81
    ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
       3] * x32 + cc[4] * x72 + cc[5] * x82
    #ypred
    graphics.off()
    win.graph()
    plot(y, ypred)
    win.graph()
    dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
       )$coef)
    ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
       2 + dd[4] * ypred^3
    plot(y, ypredf)
    correlation1 <- cor(y, ypred)
    correlation2 <- cor(y, ypredf)
    round(rbind(correlation1, correlation2), digits
        = 6)
    #calculate reliability index beta Tp=230:
    Ttmin<-230.9052
    ymin <- log(Ttmin)
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
       0.7)
    constant <- aa + bb[1] * input[1] + bb[3] *
       input[3] + bb[4] * input[7]
    constant <- constant + cc[1] * input[1]^2 + cc[
       3] * input[3]^2 + cc[4] * input[7]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[2]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
    A8 <- cc[5] * sigma2^2
    B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
       2 + bb[5] * mu2 + constant
    #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
 #calculate lambda,beta
    D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
       A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
       A8^2 * B7^2 + A7^2 * A8 * B8^2
    D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
       2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
       A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
       2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
    D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
       A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
       2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
       2
    D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
       B7^2 + 2 * B8^2)
    D4 <- 4 * (C - ymin)
    lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
    u7 <-  - B7/(2 * (lambda + A7))
    u8 <-  - B8/(2 * (lambda + A8))
    beta <- (u8^2 + u7^2)^(1/2)
    round(rbind(lambda, beta), digits = 8)
    #choose smallest beta
    betas <- min(beta)
    betas  #Find design point:
    U7 <- seq(-10, 60, 0.1)
    delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
       C - ymin))^(1/2)
    U81 <- ( - B8 + delta)/(2 * A8)
    U82 <- ( - B8 - delta)/(2 * A8)
    U8 <- cbind(U82, U81)
    U91 <- (betas^2 - U7^2)^(1/2)
    U92 <-  - (betas^2 - U7^2)^(1/2)
    U9 <- cbind(U91, U92)
    U8f <- cbind(U8, U9)   #win.graph()
 #matplot(U7, U9,xlim = c(-15, 15), ylim = c(-15, 15))
 #win.graph()
 #matplot(U7, U8,xlim = c(-15, 15), ylim = c(-15, 15))
    win.graph()
    par(pty = "s")
    matplot(U7, U8f, col = 1, xlim = c(-10, 60),
       ylim = c(-30, 40), type = "l", xlab = "u2", ylab = "u8")

    U7 <- seq(-5, 5, 0.1)
    delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
       C - ymin))^(1/2)
    U81 <- ( - B8 + delta)/(2 * A8)
    U82 <- ( - B8 - delta)/(2 * A8)
    U8 <- cbind(U82, U81)
    U91 <- (betas^2 - U7^2)^(1/2)
    U92 <-  - (betas^2 - U7^2)^(1/2)
    U9 <- cbind(U91, U92)
    U8f <- cbind(U8, U9)

    win.graph()
    par(pty = "s")
    matplot(U7, U8f, col = 1, xlim = c(-5, 5), ylim
        = c(-5, 5), type =    "l", xlab = "u2", ylab = "u8")
 }
```

```
StateU4L<-function(x, y, n)
{
#StateU3:calculate U43:lambda,beta constraint,design point D x4~6 are
   ignored:
#with logorithmic fit to Tt:
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
      4], resUlm(x, y)$coef[6], resUlm(x, y)$
      coef[8], resUlm(x, y)$coef[10])
   cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
      5], resUlm(x, y)$coef[7], resUlm(x, y)$
      coef[9], resUlm(x, y)$coef[11])
   ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
      x31 + bb[4] * x71 + bb[5] * x81
   ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
      3] * x32 + cc[4] * x72 + cc[5] * x82
   #ypred
   graphics.off()
   win.graph()
   plot(y, ypred)
   win.graph()
   dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
      )$coef)
   ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
      2 + dd[4] * ypred^3
   plot(y, ypredf)
   correlation1 <- cor(y, ypred)
   correlation2 <- cor(y, ypredf)
   round(rbind(correlation1, correlation2), digits
       = 6)
   #calculate reliability index beta Tp=250 sec:
   Ttmin<-250.6776
   ymin <- log(Ttmin)
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
      0.7)
   constant <- aa + bb[1] * input[1] + bb[3] *
      input[3] + bb[4] * input[7]
   constant <- constant + cc[1] * input[1]^2 + cc[
      3] * input[3]^2 + cc[4] * input[7]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[2]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[5] * sigma2^2
   B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
      2 + bb[5] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
      A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
```

```
       A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
       2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
       A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
       2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
       A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
       2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
       2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
       B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(lambda, beta), digits = 8)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   U7 <- seq(-10, 60, 0.1)
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
       C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U91, U92)
   U8f <- cbind(U8, U9)   #win.graph()
#matplot(U7, U9,xlim = c(-15, 15), ylim = c(-15, 15))
#win.graph()
#matplot(U7, U8,xlim = c(-15, 15), ylim = c(-15, 15))
   win.graph()
   par(pty = "s")
   matplot(U7, U8f, col = 1, xlim = c(-10, 60),
       ylim = c(-30, 40), type = "l", xlab = "u2", ylab = "u8")

   U7 <- seq(-5, 5, 0.1)
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
       C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U91, U92)
   U8f <- cbind(U8, U9)

   win.graph()
   par(pty = "s")
   matplot(U7, U8f, col = 1, xlim = c(-5, 5), ylim
        = c(-5, 5), type =    "l", xlab = "u2", ylab = "u8")
}
```

A.5 Functions for Monte-Carlo simulation by using response surface

```
CarloG1<-function(x, y, n)
{
#CarloG1:#x6 is ignored,variables:x2,x8:
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
      x41 + x42 + x51 + x52 + x71 + x72 + x81 +
      x82)
   reslm(x, y)$coef #calculate correlations:
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
      ], reslm(x, y)$coef[6], reslm(x, y)$
      coef[8])
   bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
      coef[12], reslm(x, y)$coef[14])
   cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
      ], reslm(x, y)$coef[7], reslm(x, y)$
      coef[9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
      coef[13], reslm(x, y)$coef[15])
   Tmin <- 311.0956
   ymax <-  Tmin
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6, 0.7)
   constant <- aa + bb[1] * input[1] + bb[6] *
      input[7] + bb[3] * input[3]
   constant <- constant + bb[4] * input[4] + bb[5] *input[5]
   constant <- constant + cc[1] * input[1]^2 + cc[
      6] * input[7]^2 + cc[3] * input[3]^2
   constant <- constant + cc[4] * input[4]^2 + cc[
      5] * input[5]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[7] * sigma2^2
   B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
      2 + bb[7] * mu2 + constant
   x7 <- rnorm(n, mu1, sigma1)
   x8 <- rnorm(n, mu2, sigma2)
   u7 <- (x7 - mu1)/sigma1
   u8 <- (x8 - mu2)/sigma2
   Tt <- A7 * u7^2 + B7 * u7 + A8 * u8^2 + B8 * u8 +
      C
   graphics.off()
   win.graph()
```

```
    hist(Tt)
    pnorm <- length(Tt[Tt < ymax])/n
    round(rbind(Tmin, ymax, pnorm), digits = 6)
}
```

```
CarloG2<-function(x, y, n)
{
#CarloG2:#x6 is ignored,variables:x2,x8:
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x41 <- x[, 4]
    x42 <- x41^2
    x51 <- x[, 5]
    x52 <- x51^2
    x61 <- x[, 6]
    x62 <- x61^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
        x41 + x42 + x51 + x52 + x71 + x72 + x81 +
        x82)
    reslm(x, y)$coef #calculate correlations:
    aa <- reslm(x, y)$coef[1]
    bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
        ], reslm(x, y)$coef[6], reslm(x, y)$
        coef[8])
    bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
        coef[12], reslm(x, y)$coef[14])
    cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
        ], reslm(x, y)$coef[7], reslm(x, y)$
        coef[9])
    cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
        coef[13], reslm(x, y)$coef[15])
    Tmin <- 330.9244
    ymax <-  Tmin
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6, 0.7)
    constant <- aa + bb[1] * input[1] + bb[6] *
        input[7] + bb[3] * input[3]
    constant <- constant + bb[4] * input[4] + bb[5] *input[5]
    constant <- constant + cc[1] * input[1]^2 + cc[
        6] * input[7]^2 + cc[3] * input[3]^2
    constant <- constant + cc[4] * input[4]^2 + cc[
        5] * input[5]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
    A8 <- cc[7] * sigma2^2
    B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
        2 + bb[7] * mu2 + constant
    x7 <- rnorm(n, mu1, sigma1)
    x8 <- rnorm(n, mu2, sigma2)
    u7 <- (x7 - mu1)/sigma1
    u8 <- (x8 - mu2)/sigma2
    Tt <- A7 * u7^2 + B7 * u7 + A8 * u8^2 + B8 * u8 +
        C
    graphics.off()
    win.graph()
    hist(Tt)
    pnorm <- length(Tt[Tt < ymax])/n
    round(rbind(Tmin, ymax, pnorm), digits = 6)
}
```

```
CarloG3<-function(x, y, n)
{
#CarloG3:#x4~6 are ignored,variables:x2,x8:
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x41 <- x[, 4]
    x42 <- x41^2
    x51 <- x[, 5]
    x52 <- x51^2
    x61 <- x[, 6]
    x62 <- x61^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    resU <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
         x71 + x72 + x81 + x82)
    resUlm(x, y)$coef
    aa <- resUlm(x, y)$coef[1]
    bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[4
        ], resUlm(x, y)$coef[6], resUlm(x, y)$
        coef[8], resUlm(x, y)$coef[10])

    cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[5
        ], resUlm(x, y)$coef[7], resUlm(x, y)$
        coef[9], resUlm(x, y)$coef[11])

    Tmin <- 237.3830
    ymax <-  Tmin
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6, 0.7)
    constant <- aa + bb[1] * input[1] + bb[3] *
        input[3] + bb[4] * input[7]
    constant <- constant + cc[1] * input[1]^2 + cc[
        3] * input[3]^2 + cc[4] * input[7]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-ymax=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
    A8 <- cc[5] * sigma2^2
    B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
        2 + bb[5] * mu2 + constant
    x7 <- rnorm(n, mu1, sigma1)
    x8 <- rnorm(n, mu2, sigma2)
    u7 <- (x7 - mu1)/sigma1
```

```
   u8 <- (x8 - mu2)/sigma2
   Tt <- A7 * u7^2 + B7 * u7 + A8 * u8^2 + B8 * u8 + C
   graphics.off()
   win.graph()
   hist(Tt)
   pnorm <- length(Tt[Tt < ymax])/n
   round(rbind(Tmin, ymax, pnorm), digits = 6)
}
```

```
CarloG4<-function(x, y, n)
{
#CarloG4:#x4~6 are ignored,variables:x2,x8:
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   resU <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
       x71 + x72 + x81 + x82)
   resUlm(x, y)$coef
   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[4
      ], resUlm(x, y)$coef[6], resUlm(x, y)$
      coef[8], resUlm(x, y)$coef[10])

   cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[5
      ], resUlm(x, y)$coef[7], resUlm(x, y)$
      coef[9], resUlm(x, y)$coef[11])

   Tmin <- 257.1608
   ymax <-  Tmin
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6, 0.7)
   constant <- aa + bb[1] * input[1] + bb[3] *
      input[3] + bb[4] * input[7]
   constant <- constant + cc[1] * input[1]^2 + cc[
      3] * input[3]^2 + cc[4] * input[7]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymax=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[5] * sigma2^2
   B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
      2 + bb[5] * mu2 + constant
   x7 <- rnorm(n, mu1, sigma1)
   x8 <- rnorm(n, mu2, sigma2)
   u7 <- (x7 - mu1)/sigma1
   u8 <- (x8 - mu2)/sigma2
   Tt <- A7 * u7^2 + B7 * u7 + A8 * u8^2 + B8 * u8 + C
```

```
    graphics.off()
    win.graph()
    hist(Tt)
    pnorm <- length(Tt[Tt < ymax])/n
    round(rbind(Tmin, ymax, pnorm), digits = 6)
}
```

```
CarloG1L<-function(x, y, n)
{
#CarloG1L:#x6 is ignored,variables:x2,x8:
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x41 <- x[, 4]
    x42 <- x41^2
    x51 <- x[, 5]
    x52 <- x51^2
    x61 <- x[, 6]
    x62 <- x61^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
        x41 + x42 + x51 + x52 + x71 + x72 + x81 +
        x82)
    reslm(x, y)$coef #calculate correlations:
    aa <- reslm(x, y)$coef[1]
    bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
        ], reslm(x, y)$coef[6], reslm(x, y)$
        coef[8])
    bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
        coef[12], reslm(x, y)$coef[14])
    cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
        ], reslm(x, y)$coef[7], reslm(x, y)$
        coef[9])
    cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
        coef[13], reslm(x, y)$coef[15])
    Tmin <- 301.1548
    ymax <-  log(Tmin)
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6, 0.7)
    constant <- aa + bb[1] * input[1] + bb[6] *
        input[7] + bb[3] * input[3]
    constant <- constant + bb[4] * input[4] + bb[5] *input[5]
    constant <- constant + cc[1] * input[1]^2 + cc[
        6] * input[7]^2 + cc[3] * input[3]^2
    constant <- constant + cc[4] * input[4]^2 + cc[
        5] * input[5]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
    A8 <- cc[7] * sigma2^2
    B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
        2 + bb[7] * mu2 + constant
    x7 <- rnorm(n, mu1, sigma1)
    x8 <- rnorm(n, mu2, sigma2)
    u7 <- (x7 - mu1)/sigma1
    u8 <- (x8 - mu2)/sigma2
    TtL <- A7 * u7^2 + B7 * u7 + A8 * u8^2 + B8 * u8 + C
    TtL<-exp(TtL)
    graphics.off()
    win.graph()
    hist(TtL)
    pnorm <- length(TtL[TtL < Tmin])/n
    round(rbind(Tmin, pnorm), digits = 6)
}
```

```
CarloG2L<-function(x, y, n)
{
#CarloG2L:#x6 is ignored,variables:x2,x8:
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x41 <- x[, 4]
    x42 <- x41^2
    x51 <- x[, 5]
    x52 <- x51^2
    x61 <- x[, 6]
    x62 <- x61^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
        x41 + x42 + x51 + x52 + x71 + x72 + x81 +
        x82)
    reslm(x, y)$coef #calculate correlations:
    aa <- reslm(x, y)$coef[1]
    bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
        ], reslm(x, y)$coef[6], reslm(x, y)$
        coef[8])
    bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
        coef[12], reslm(x, y)$coef[14])
    cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
        ], reslm(x, y)$coef[7], reslm(x, y)$
        coef[9])
    cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
        coef[13], reslm(x, y)$coef[15])
    Tmin <- 321.1342
    ymax <-  log(Tmin)
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6, 0.7)
    constant <- aa + bb[1] * input[1] + bb[6] *
        input[7] + bb[3] * input[3]
    constant <- constant + bb[4] * input[4] + bb[5] *input[5]
    constant <- constant + cc[1] * input[1]^2 + cc[
        6] * input[7]^2 + cc[3] * input[3]^2
    constant <- constant + cc[4] * input[4]^2 + cc[
        5] * input[5]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
    A8 <- cc[7] * sigma2^2
    B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
       2 + bb[7] * mu2 + constant
    x7 <- rnorm(n, mu1, sigma1)
    x8 <- rnorm(n, mu2, sigma2)
    u7 <- (x7 - mu1)/sigma1
    u8 <- (x8 - mu2)/sigma2
    TtL <- A7 * u7^2 + B7 * u7 + A8 * u8^2 + B8 * u8 + C
    TtL<-exp(TtL)
    graphics.off()
    win.graph()
    hist(TtL)
    pnorm <- length(TtL[TtL < Tmin])/n
    round(rbind(Tmin, pnorm), digits = 6)
}
```

```
CarloG3L<-function(x, y, n)
{
#CarloG3L:#x4~6 are ignored,variables:x2,x8:
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x41 <- x[, 4]
    x42 <- x41^2
    x51 <- x[, 5]
    x52 <- x51^2
    x61 <- x[, 6]
    x62 <- x61^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    resU <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
        x71 + x72 + x81 + x82)
    resUlm(x, y)$coef
    aa <- resUlm(x, y)$coef[1]
    bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[4
       ], resUlm(x, y)$coef[6], resUlm(x, y)$
       coef[8], resUlm(x, y)$coef[10])

    cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[5
       ], resUlm(x, y)$coef[7], resUlm(x, y)$
       coef[9], resUlm(x, y)$coef[11])

    Tmin <- 230.9052
    ymax <-  log(Tmin)
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6, 0.7)
    constant <- aa + bb[1] * input[1] + bb[3] *
       input[3] + bb[4] * input[7]
    constant <- constant + cc[1] * input[1]^2 + cc[
       3] * input[3]^2 + cc[4] * input[7]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-ymax=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
```

```
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[5] * sigma2^2
   B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
      2 + bb[5] * mu2 + constant
   x7 <- rnorm(n, mu1, sigma1)
   x8 <- rnorm(n, mu2, sigma2)
   u7 <- (x7 - mu1)/sigma1
   u8 <- (x8 - mu2)/sigma2
   TtL <- A7 * u7^2 + B7 * u7 + A8 * u8^2 + B8 * u8 + C
   TtL<-exp(TtL)
   graphics.off()
   win.graph()
   hist(TtL)
   pnorm <- length(TtL[TtL < Tmin])/n
   round(rbind(Tmin, pnorm), digits = 6)
}
```

```
CarloG4L<-function(x, y, n)
{
#CarloG4L:#x4~6 are ignored,variables:x2,x8:
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   resU <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
       x71 + x72 + x81 + x82)
   resUlm(x, y)$coef
   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[4
      ], resUlm(x, y)$coef[6], resUlm(x, y)$
      coef[8], resUlm(x, y)$coef[10])

   cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[5
      ], resUlm(x, y)$coef[7], resUlm(x, y)$
      coef[9], resUlm(x, y)$coef[11])

   Tmin <- 250.6776
   ymax <-  log(Tmin)
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6, 0.7)
   constant <- aa + bb[1] * input[1] + bb[3] *
      input[3] + bb[4] * input[7]
   constant <- constant + cc[1] * input[1]^2 + cc[
      3] * input[3]^2 + cc[4] * input[7]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-ymax=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
```

```
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
    A8 <- cc[5] * sigma2^2
    B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
       2 + bb[5] * mu2 + constant
    x7 <- rnorm(n, mu1, sigma1)
    x8 <- rnorm(n, mu2, sigma2)
    u7 <- (x7 - mu1)/sigma1
    u8 <- (x8 - mu2)/sigma2
    TtL <- A7 * u7^2 + B7 * u7 + A8 * u8^2 + B8 * u8 + C
    TtL<-exp(TtL)
    graphics.off()
    win.graph()
    hist(TtL)
    pnorm <- length(TtL[TtL < Tmin])/n
    round(rbind(Tmin, pnorm), digits = 6)
}
```

A.6 Functions for comparing reliability index both standard and logarithmic fit and drawing figures to illustrate their difference

```
StateU1S<-function(x, y)


{
#StateU1S:lambda, beta,constraint,design point D;
#x6 is ignored in U13:
#reslm:calculate the coefficients of X
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
      x41 + x42 + x51 + x52 + x71 + x72 + x81 +
      x82)
   reslm(x, y)$coef #calculate correlations:
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
      ], reslm(x, y)$coef[6], reslm(x, y)$
      coef[8])
   bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
      coef[12], reslm(x, y)$coef[14])
   cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
      ], reslm(x, y)$coef[7], reslm(x, y)$
      coef[9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
      coef[13], reslm(x, y)$coef[15])
   ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
      x31 + bb[4] * x41 + bb[5] * x51 + bb[6] *
      x71 + bb[7] * x81
   ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
      3] * x32 + cc[4] * x42 + cc[5] * x52 +
      cc[6] * x72 + cc[7] * x82  #ypred
#graphics.off()
#win.graph()
#plot(y, ypred)
#win.graph()
   dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
      )$coef)
   ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
      2 + dd[4] * ypred^3 #plot(y, ypredf)
   correlation1 <- cor(y, ypred)
   correlation2 <- cor(y, ypredf)
   round(rbind(correlation1, correlation2), digits
       = 6)
   #calculate reliability index beta:
   ymin <- 311.0956
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
      0.7)
   constant <- aa + bb[1] * input[1] + bb[6] *
      input[7] + bb[3] * input[3]
   constant <- constant + bb[4] * input[4] + bb[5] *
      input[5]
   constant <- constant + cc[1] * input[1]^2 + cc[
      6] * input[7]^2 + cc[3] * input[3]^2
   constant <- constant + cc[4] * input[4]^2 + cc[
      5] * input[5]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[7] * sigma2^2
   B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
      2 + bb[7] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
      A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
      A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
      2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
      A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
      A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
      2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
      B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(beta, lambda), digits = 4)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   U7 <- seq(-5, 35, 0.1)
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
      C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U92, U91)
   U8f <- cbind(U8, U9)
   graphics.off()
   win.graph()
   par(pty = "s")
   matplot(U7, U8f, col = 1, xlim = c(-10, 50),
      ylim = c(-20, 40), type = "l", xlab = "u2", ylab = "u8")


#StateU1L:lambda, beta,constraint,design point D;
#x6 is ignored in U13 with logorithmic fit to Tt:
#reslm:calculate the coefficients of X
   x<-U13[,1:8]
   y<-log(U13[,9])
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
      x41 + x42 + x51 + x52 + x71 + x72 + x81 +
      x82)
   reslm(x, y)$coef #calculate correlations:
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
      ], reslm(x, y)$coef[6], reslm(x, y)$
      coef[8])
   bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
      coef[12], reslm(x, y)$coef[14])
   cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
      ], reslm(x, y)$coef[7], reslm(x, y)$
      coef[9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
      coef[13], reslm(x, y)$coef[15])
   ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
      x31 + bb[4] * x41 + bb[5] * x51 + bb[6] *
      x71 + bb[7] * x81
   ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
      3] * x32 + cc[4] * x42 + cc[5] * x52 +
      cc[6] * x72 + cc[7] * x82  #ypred
#graphics.off()
#win.graph()
#plot(y, ypred)
#win.graph()
   dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
      )$coef)
   ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
      2 + dd[4] * ypred^3 #plot(y, ypredf)
   correlation1 <- cor(y, ypred)
   correlation2 <- cor(y, ypredf)
   round(rbind(correlation1, correlation2), digits
       = 6)
   #calculate reliability index beta Tp=300 sec:
   Ttmin<-301.1548
   ymin <- log(Ttmin)
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
      0.7)
   constant <- aa + bb[1] * input[1] + bb[6] *
      input[7] + bb[3] * input[3]
   constant <- constant + bb[4] * input[4] + bb[5] *
      input[5]
   constant <- constant + cc[1] * input[1]^2 + cc[
      6] * input[7]^2 + cc[3] * input[3]^2
   constant <- constant + cc[4] * input[4]^2 + cc[
      5] * input[5]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[7] * sigma2^2
   B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
      2 + bb[7] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
      A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
      A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
      2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
      A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
      A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
      2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
      B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(beta, lambda), digits = 4)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
      C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8L <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9L <- cbind(U92, U91)
   U8fL <- cbind(U8L, U9L)
   #graphics.off()
   win.graph()
   par(pty = "s")
   matplot(U7, U8fL, col = 1, xlim = c(-10, 50),
      ylim = c(-20, 40), type = "l", xlab = "u2", ylab = "u8")

   U8U<-cbind(U8f,U8fL)
   win.graph()
   par(pty = "s")
   matplot(U7, U8U, col = 1, xlim = c(-10, 40), ylim
       = c(-10, 40), type ="l", xlab = "u2", ylab = "u8")
   U8U<-cbind(U8f,U8fL)
   win.graph()
   par(pty = "s")
   matplot(U7, U8U, col = 1, xlim = c(-5, 5), ylim
       = c(-5, 5), type ="l", xlab = "u2", ylab = "u8")

}




StateU2S<-function(x, y)


{
#StateU2S:lambda, beta,constraint,design point D;
#x6 is ignored in U23:
```

```
#reslm:calculate the coefficients of X
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x41 <- x[, 4]
    x42 <- x41^2
    x51 <- x[, 5]
    x52 <- x51^2
    x61 <- x[, 6]
    x62 <- x61^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
        x41 + x42 + x51 + x52 + x71 + x72 + x81 +
        x82)
    reslm(x, y)$coef #calculate correlations:
    aa <- reslm(x, y)$coef[1]
    bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
        ], reslm(x, y)$coef[6], reslm(x, y)$
        coef[8])
    bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
        coef[12], reslm(x, y)$coef[14])
    cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
        ], reslm(x, y)$coef[7], reslm(x, y)$
        coef[9])
    cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
        coef[13], reslm(x, y)$coef[15])
    ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
        x31 + bb[4] * x41 + bb[5] * x51 + bb[6] *
        x71 + bb[7] * x81
    ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
        3] * x32 + cc[4] * x42 + cc[5] * x52 +
        cc[6] * x72 + cc[7] * x82  #ypred
#graphics.off()
#win.graph()
#plot(y, ypred)
#win.graph()
    dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
        )$coef)
    ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
        2 + dd[4] * ypred^3 #plot(y, ypredf)
    correlation1 <- cor(y, ypred)
    correlation2 <- cor(y, ypredf)
    round(rbind(correlation1, correlation2), digits
         = 6)
    #calculate reliability index beta:
    ymin <- 330.9244
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
        0.7)
    constant <- aa + bb[1] * input[1] + bb[6] *
        input[7] + bb[3] * input[3]
    constant <- constant + bb[4] * input[4] + bb[5] *
        input[5]
    constant <- constant + cc[1] * input[1]^2 + cc[
        6] * input[7]^2 + cc[3] * input[3]^2
    constant <- constant + cc[4] * input[4]^2 + cc[
        5] * input[5]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
```

```
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[7] * sigma2^2
   B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
      2 + bb[7] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
      A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
      A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
      2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
      A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
      A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
      2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
      B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(beta, lambda), digits = 4)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   U7 <- seq(-5, 35, 0.1)
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
      C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U92, U91)
   U8f <- cbind(U8, U9)
   graphics.off()
   win.graph()
   par(pty = "s")
   matplot(U7, U8f, col = 1, xlim = c(-10, 50),
      ylim = c(-20, 40), type = "l", xlab = "u2", ylab = "u8")


#StateU2L:lambda, beta,constraint,design point D;
#x6 is ignored in U13 with logorithmic fit to Tt:
#reslm:calculate the coefficients of X
   x<-U23[,1:8]
   y<-log(U23[,9])
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x41 <- x[, 4]
   x42 <- x41^2
   x51 <- x[, 5]
   x52 <- x51^2
   x61 <- x[, 6]
   x62 <- x61^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   res <- lm(y ~ x11 + x12 + x21 + x22 + x31 + x32 +
      x41 + x42 + x51 + x52 + x71 + x72 + x81 +
      x82)
   reslm(x, y)$coef #calculate correlations:
   aa <- reslm(x, y)$coef[1]
   bb <- c(reslm(x, y)$coef[2], reslm(x, y)$coef[4
      ], reslm(x, y)$coef[6], reslm(x, y)$
      coef[8])
   bb <- c(bb, reslm(x, y)$coef[10], reslm(x, y)$
      coef[12], reslm(x, y)$coef[14])
   cc <- c(reslm(x, y)$coef[3], reslm(x, y)$coef[5
      ], reslm(x, y)$coef[7], reslm(x, y)$
      coef[9])
   cc <- c(cc, reslm(x, y)$coef[11], reslm(x, y)$
      coef[13], reslm(x, y)$coef[15])
   ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
      x31 + bb[4] * x41 + bb[5] * x51 + bb[6] *
      x71 + bb[7] * x81
   ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
      3] * x32 + cc[4] * x42 + cc[5] * x52 +
      cc[6] * x72 + cc[7] * x82  #ypred
#graphics.off()
#win.graph()
#plot(y, ypred)
#win.graph()
   dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
      )$coef)
   ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
      2 + dd[4] * ypred^3 #plot(y, ypredf)
   correlation1 <- cor(y, ypred)
   correlation2 <- cor(y, ypredf)
   round(rbind(correlation1, correlation2), digits
       = 6)
   #calculate reliability index beta Tp=300 sec:
   Ttmin<-321.1342
   ymin <- log(Ttmin)
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
      0.7)
   constant <- aa + bb[1] * input[1] + bb[6] *
      input[7] + bb[3] * input[3]
   constant <- constant + bb[4] * input[4] + bb[5] *
      input[5]
   constant <- constant + cc[1] * input[1]^2 + cc[
      6] * input[7]^2 + cc[3] * input[3]^2
   constant <- constant + cc[4] * input[4]^2 + cc[
      5] * input[5]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[,2]+cc[7]*x[,8]^2+bb[7]*x[,8]+constant-yt=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[7] * sigma2^2
   B8 <- 2 * cc[7] * sigma2 * mu2 + bb[7] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[7] * mu2^
      2 + bb[7] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
      A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
      A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
      2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
      A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
       A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
       2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
       2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
       B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(beta, lambda), digits = 4)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
       C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8L <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9L <- cbind(U92, U91)
   U8fL <- cbind(U8L, U9L)
   #graphics.off()
   win.graph()
   par(pty = "s")
   matplot(U7, U8fL, col = 1, xlim = c(-10, 50),
       ylim = c(-10, 50), type = "l", xlab = "u2", ylab = "u8")
   U8U<-cbind(U8f,U8fL)
   win.graph()
   par(pty = "s")
   matplot(U7, U8U, col = 1, xlim = c(-10, 40), ylim
        = c(-10, 40), type ="l", xlab = "u2", ylab = "u8")
   U8U<-cbind(U8f,U8fL)
   win.graph()
   par(pty = "s")
   matplot(U7, U8U, col = 1, xlim = c(-5, 5), ylim
        = c(-5, 5), type ="l", xlab = "u2", ylab = "u8")

}





StateU3S<-function(x, y)


{
#StateU3S:calculate U33:lambda,beta constraint,design point D x4~6 are
   ignored:
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
       4], resUlm(x, y)$coef[6], resUlm(x, y)$
       coef[8], resUlm(x, y)$coef[10])
   cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
       5], resUlm(x, y)$coef[7], resUlm(x, y)$
       coef[9], resUlm(x, y)$coef[11])
   ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
       x31 + bb[4] * x71 + bb[5] * x81
   ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
       3] * x32 + cc[4] * x72 + cc[5] * x82
   #ypred
   graphics.off()
   #win.graph()
   #plot(y, ypred)
   #win.graph()
   dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
       )$coef)
   ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
       2 + dd[4] * ypred^3
   #plot(y, ypredf)
   correlation1 <- cor(y, ypred)
   correlation2 <- cor(y, ypredf)
   round(rbind(correlation1, correlation2), digits
        = 6)
   #calculate reliability index beta:
   ymin <- 237.3830
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
       0.7)
   constant <- aa + bb[1] * input[1] + bb[3] *
       input[3] + bb[4] * input[7]
   constant <- constant + cc[1] * input[1]^2 + cc[
       3] * input[3]^2 + cc[4] * input[7]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[2]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[5] * sigma2^2
   B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
       2 + bb[5] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
       A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
       A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
       2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
       A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
       2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
       A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
       2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
       2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
       B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(lambda, beta), digits = 8)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   U7 <- seq(-5, 35, 0.1)
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
       C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U91, U92)
   U8f <- cbind(U8, U9)   #win.graph()
#matplot(U7, U9,xlim = c(-15, 15), ylim = c(-15, 15))
#win.graph()
#matplot(U7, U8,xlim = c(-15, 15), ylim = c(-15, 15))
   win.graph()
   par(pty = "s")
   matplot(U7, U8f, col = 1, xlim = c(-10, 50),
      ylim = c(-10, 50), type = "l", xlab = "u2", ylab = "u8")

#StateU3:calculate U33:lambda,beta constraint,design point D x4~6 are
   ignored:
#with logorithmic fit to Tt:
   x<-U33[,1:8]
   y<-log(U33[,9])
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
      4], resUlm(x, y)$coef[6], resUlm(x, y)$
      coef[8], resUlm(x, y)$coef[10])
   cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
      5], resUlm(x, y)$coef[7], resUlm(x, y)$
      coef[9], resUlm(x, y)$coef[11])
   ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
      x31 + bb[4] * x71 + bb[5] * x81
   ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
      3] * x32 + cc[4] * x72 + cc[5] * x82
   #ypred
   #graphics.off()
   #win.graph()
   #plot(y, ypred)
   #win.graph()
   dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
      )$coef)
   ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
      2 + dd[4] * ypred^3
   #plot(y, ypredf)
   correlation1 <- cor(y, ypred)
   correlation2 <- cor(y, ypredf)
   round(rbind(correlation1, correlation2), digits
       = 6)
   #calculate reliability index beta Tp=230:
   Ttmin<-230.9052
   ymin <- log(Ttmin)
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
      0.7)
   constant <- aa + bb[1] * input[1] + bb[3] *
      input[3] + bb[4] * input[7]
   constant <- constant + cc[1] * input[1]^2 + cc[
      3] * input[3]^2 + cc[4] * input[7]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[2]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[5] * sigma2^2
   B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
      2 + bb[5] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
      A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
      A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
      2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
      A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
      A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
      2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
      B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(lambda, beta), digits = 8)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
      C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U91, U92)
   U8fL <- cbind(U8, U9)  #win.graph()
#matplot(U7, U9,xlim = c(-15, 15), ylim = c(-15, 15))
#win.graph()
#matplot(U7, U8,xlim = c(-15, 15), ylim = c(-15, 15))
   win.graph()
   par(pty = "s")
   matplot(U7, U8fL, col = 1, xlim = c(-5, 35),
      ylim = c(-5, 35), type = "l", xlab = "u2", ylab = "u8")

   U8U <- cbind(U8f, U8fL)
   win.graph()
   par(pty = "s")
   matplot(U7, U8U, col = 1, xlim = c(-5, 35), ylim
       = c(-5, 35), type =   "l", xlab = "u2", ylab = "u8")

   win.graph()
   par(pty = "s")
   matplot(U7, U8U, col = 1, xlim = c(-5, 5), ylim
       = c(-5, 5), type =    "l", xlab = "u2", ylab = "u8")
}
```

```
StateU4S<-function(x, y)


{
#StateU4S:calculate U33:lambda,beta constraint,design point D x4~6 are
    ignored:
    x11 <- x[, 1]
    x12 <- x11^2
    x21 <- x[, 2]
    x22 <- x21^2
    x31 <- x[, 3]
    x32 <- x31^2
    x71 <- x[, 7]
    x72 <- x71^2
    x81 <- x[, 8]
    x82 <- x81^2
    aa <- resUlm(x, y)$coef[1]
    bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
        4], resUlm(x, y)$coef[6], resUlm(x, y)$
        coef[8], resUlm(x, y)$coef[10])
    cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
        5], resUlm(x, y)$coef[7], resUlm(x, y)$
        coef[9], resUlm(x, y)$coef[11])
    ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
        x31 + bb[4] * x71 + bb[5] * x81
    ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
        3] * x32 + cc[4] * x72 + cc[5] * x82
    #ypred
    graphics.off()
    #win.graph()
    #plot(y, ypred)
    #win.graph()
    dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
        )$coef)
    ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
        2 + dd[4] * ypred^3
    #plot(y, ypredf)
    correlation1 <- cor(y, ypred)
    correlation2 <- cor(y, ypredf)
    round(rbind(correlation1, correlation2), digits
         = 6)
    #calculate reliability index beta:
    ymin <- 257.1608
    input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
        0.7)
    constant <- aa + bb[1] * input[1] + bb[3] *
        input[3] + bb[4] * input[7]
    constant <- constant + cc[1] * input[1]^2 + cc[
        3] * input[3]^2 + cc[4] * input[7]^2
    constant
    #cc[2]*x[,2]^2+bb[2]*x[2]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
    sigma1 <- 20
    mu1 <- 450
    sigma2 <- 0.1
    mu2 <- 0.7
    A7 <- cc[2] * sigma1^2
    B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
    A8 <- cc[5] * sigma2^2
    B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
    C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
        2 + bb[5] * mu2 + constant
    #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
    D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
        A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
        A8^2 * B7^2 + A7^2 * A8 * B8^2
    D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
      2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
      A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
      2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
      A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
      2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
      2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
      B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(lambda, beta), digits = 8)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   U7 <- seq(-5, 35, 0.1)
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
      C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U91, U92)
   U8f <- cbind(U8, U9)   #win.graph()
#matplot(U7, U9,xlim = c(-15, 15), ylim = c(-15, 15))
#win.graph()
#matplot(U7, U8,xlim = c(-15, 15), ylim = c(-15, 15))
   win.graph()
   par(pty = "s")
   matplot(U7, U8f, col = 1, xlim = c(-10, 50),
      ylim = c(-10, 50), type = "l", xlab = "u2", ylab = "u8")

#StateU4:calculate U33:lambda,beta constraint,design point D x4~6 are
   ignored:
#with logorithmic fit to Tt:
   x<-U43[,1:8]
   y<-log(U43[,9])
   x11 <- x[, 1]
   x12 <- x11^2
   x21 <- x[, 2]
   x22 <- x21^2
   x31 <- x[, 3]
   x32 <- x31^2
   x71 <- x[, 7]
   x72 <- x71^2
   x81 <- x[, 8]
   x82 <- x81^2
   aa <- resUlm(x, y)$coef[1]
   bb <- c(resUlm(x, y)$coef[2], resUlm(x, y)$coef[
      4], resUlm(x, y)$coef[6], resUlm(x, y)$
      coef[8], resUlm(x, y)$coef[10])
   cc <- c(resUlm(x, y)$coef[3], resUlm(x, y)$coef[
      5], resUlm(x, y)$coef[7], resUlm(x, y)$
      coef[9], resUlm(x, y)$coef[11])
   ypred <- aa + bb[1] * x11 + bb[2] * x21 + bb[3] *
      x31 + bb[4] * x71 + bb[5] * x81
   ypred <- ypred + cc[1] * x12 + cc[2] * x22 + cc[
      3] * x32 + cc[4] * x72 + cc[5] * x82
   #ypred
   #graphics.off()
   #win.graph()
   #plot(y, ypred)
   #win.graph()
   dd <- c(lsfit(cbind(ypred, ypred^2, ypred^3), y
       )$coef)
   ypredf <- dd[1] + dd[2] * ypred + dd[3] * ypred^
       2 + dd[4] * ypred^3
   #plot(y, ypredf)
   correlation1 <- cor(y, ypred)
   correlation2 <- cor(y, ypredf)
   round(rbind(correlation1, correlation2), digits
        = 6)
   #calculate reliability index beta Tp=230:
   Ttmin<-250.6776
   ymin <- log(Ttmin)
   input <- c(600, 450, 250, 0.7, 0.6, 50, 0.6,
       0.7)
   constant <- aa + bb[1] * input[1] + bb[3] *
       input[3] + bb[4] * input[7]
   constant <- constant + cc[1] * input[1]^2 + cc[
       3] * input[3]^2 + cc[4] * input[7]^2
   constant
   #cc[2]*x[,2]^2+bb[2]*x[2]+cc[5]*x[,8]^2+bb[5]*x[,8]+constant-ymin=0
   sigma1 <- 20
   mu1 <- 450
   sigma2 <- 0.1
   mu2 <- 0.7
   A7 <- cc[2] * sigma1^2
   B7 <- 2 * cc[2] * sigma1 * mu1 + bb[2] * sigma1
   A8 <- cc[5] * sigma2^2
   B8 <- 2 * cc[5] * sigma2 * mu2 + bb[5] * sigma2
   C <- cc[2] * mu1^2 + bb[2] * mu1 + cc[5] * mu2^
       2 + bb[5] * mu2 + constant
   #ymin=A7*u7^2+B7*u7+A8*u8^2+B8*u8+C standardized constraint function:
#calculate lambda,beta
   D0 <- 4 * (C - ymin) * A7^2 * A8^2 - 2 * B7^2 *
       A7 * A8^2 - 2 * B8^2 * A8 * A7^2 + A7 *
       A8^2 * B7^2 + A7^2 * A8 * B8^2
   D1 <- 4 * (C - ymin) * (2 * A7 * A8^2 + 2 * A7^
       2 * A8) - 2 * B7^2 * (A8^2 + 2 * A7 *
       A8) - 2 * B8^2 * (A7^2 + 2 * A7 * A8) +
       2 * (A7 * A8 * B7^2 + A7 * A8 * B8^2)
   D2 <- 4 * (C - ymin) * (A8^2 + A7^2 + 4 * A7 *
       A8) - 2 * B7^2 * (2 * A8 + A7) - 2 * B8^
       2 * (2 * A7 + A8) + A7 * B7^2 + A8 * B8^
       2
   D3 <- 4 * (C - ymin) * (2 * A7 + 2 * A8) - (2 *
       B7^2 + 2 * B8^2)
   D4 <- 4 * (C - ymin)
   lambda <- Re(polyroot(c(D0, D1, D2, D3, D4)))
   u7 <-  - B7/(2 * (lambda + A7))
   u8 <-  - B8/(2 * (lambda + A8))
   beta <- (u8^2 + u7^2)^(1/2)
   round(rbind(lambda, beta), digits = 8)
   #choose smallest beta
   betas <- min(beta)
   betas  #Find design point:
   delta <- (B8^2 - 4 * A8 * (A7 * U7^2 + B7 * U7 +
       C - ymin))^(1/2)
   U81 <- ( - B8 + delta)/(2 * A8)
   U82 <- ( - B8 - delta)/(2 * A8)
   U8 <- cbind(U82, U81)
   U91 <- (betas^2 - U7^2)^(1/2)
   U92 <-  - (betas^2 - U7^2)^(1/2)
   U9 <- cbind(U91, U92)
   U8fL <- cbind(U8, U9)  #win.graph()
#matplot(U7, U9,xlim = c(-15, 15), ylim = c(-15, 15))
#win.graph()
#matplot(U7, U8,xlim = c(-15, 15), ylim = c(-15, 15))
   win.graph()
   par(pty = "s")
   matplot(U7, U8fL, col = 1, xlim = c(-5, 35),
       ylim = c(-5, 35), type = "l", xlab = "u2", ylab = "u8")

   U8U <- cbind(U8f, U8fL)
   win.graph()
   par(pty = "s")
   matplot(U7, U8U, col = 1, xlim = c(-5, 35), ylim
        = c(-5, 35), type =   "l", xlab = "u2", ylab = "u8")

   win.graph()
   par(pty = "s")
   matplot(U7, U8U, col = 1, xlim = c(-5, 5), ylim
        = c(-5, 5), type =    "l", xlab = "u2", ylab = "u8")
}
```