LINEAR RESPONSE SURFACE ANALYSIS AS A TECHNIQUE FOR VISUALIZING LINEAR MODELS AND DATA

Stephanus Esias Terblanche
B.Sc. Honours

Dissertation submitted in partial fulfilment of the requirements for the degree Magister Scientiae in the Department of Computer Science at the Potchefstroomse Universiteit vir Christelike Hoër Onderwys.

Supervisor: Prof. J.M. Hattingh
Assistant Supervisor: Prof. M.F. Kruger

Potchefstroom
2001

ABSTRACT

Linear Response Surface Analysis (LRSA) is a subset of the statistical field *Response Surface Methodology* (RSM). RSM is a research field dedicated to the optimization and forecasting of linear and non-linear models. These models are presented in terms of various "independent" variables that influence a dependent (or response) variable. The feature that distinguishes LRSA from RSM in general, is that LRSA can be applied to both planned and raw data, compared to RSM that is applied mainly to planned data. The terms "planned" and "raw" are used to differentiate between data collected from a planned experiment and data for which the cases are collected randomly (e.g. observational studies).

LRSA makes use of the mathematical programming technique *Linear Programming* to generate graphic representations of linear models and data. The objective of this study is to investigate the feasibility of these graphic results to reflect properties of linear models and data which will be useful for optimization and forecasting. Specific interest is shown in handling linear model building difficulties such as:

- Interpretation of models in case of interdependence among "independent" variables.
- Determining the importance of a variable to a model, relative to the other variables in the model.
- Deciding which variables to include or exclude from a model in case of multiple linear regression.
- Handling of state variables in case of optimization and forecasting.

In addition to the above objectives, software was developed for an experimental decision support system with new improved functionality, e.g. using a robust linear program solver, using parametric programming for more effective visualization, generating multi-variable response graphs, and an implementation of a parallel algorithm to speed up execution.

The outcome of the envisaged objectives was evaluated in the light of an empirical investigation using developed experimental software. In relation to each of the objectives stated it was shown that the graphic results generated with LRSA revealed important properties about the linear model and data that may aid the model building process. It is also shown that the new functionality was implemented successfully.

Key words: linear models, response surfaces, regression analysis, linear programming, parametric programming, optimization, forecasting, parallel processing.

OPSOMMING

LINEÊRE RESPONSEOPPERVLAKONTLEDING AS 'N TEGNIEK VIR DIE VISUALISERING VAN LINEÊRE MODELLE EN DATA

Lineêre Responsoppervlak Ontleding (LRO) is 'n onderafdeling van die statistiese vakgebied *Responsoppervlak Metodologie* (RM). RM fokus op die optimering van en voorspelling met lineêre en nie-lineêre modelle. Hierdie modelle word uitgedruk in terme van verskeie "onafhanklike" veranderlikes wat 'n afhanklike (of respons) veranderlike beinvloed. Die belangrikste onderskeid wat getref kan word tussen LRO en RM is dat LRO toegepas kan word op beide eksperimentele en rou data, waar RM grootliks toegespits is op beplande data. Die terme "beplande" en "rou" word gebruik om onderskeid te tref tussen data wat ingesamel word d.m.v. 'n eksperimentele ontwerp and data waarvan die waarnemings ewekansig verkry is.

LRO maak gebruik van die wiskundige programmerings tegniek Lineêre Programmering om grafiese voorstellings te genereer van linere modelle en data. Die doel van hierdie studie is om die uitvoerbaarheid te ondersoek van die grafiese voorstellings om eienskappe na vore te bring wat belangrik is vir optimering en voorspelling. Daar word ook spesifiek gekyk na aspekte wat die model bou proses kan bemoeilik soos byvoorbeeld:

- Die interpretasie van modelle waar daar interafhanklikheid voorkom tussen die "onafhanklike" veranderlikes.
- Die bepaling van die belangrikheid van 'n veranderlike in die model relatief tot die ander veranderlikes in die model.
- Die keuse om veranderlikes in te sluit of weg te laat uit 'n model in die geval van meervoudige lineêre regressie.
- Die hantering van staat veranderlikes in die geval van optimering en voorspelling.

Bykomend tot die bogenoemde doelstellings is programmatuur ontwikkel vir 'n eksperimentele besluitnemings stelsel. Hierdie stelsel beskik oor nuwe funksionaliteit soos byvoorbeeld die gebruik van 'n robuuste stelsel vir die oplos van lineêre programme, die gebruik van parametriese programmerings tegnieke om grafieke meer effektief voor te stel, die vermoë om meer-veranderlike grafieke te genereer, en 'n parallele algoritme vir die optimering van spoed.

Die uitkomste van die bogenoemde doelstellings is gemeet deur die resultate wat verkry is uit 'n emperiese studie te evalueer. Hierdie studie is uitgevoer deur gebruik te maak van die eksperimentele programmatuur.

Die empiriese resultate het getoon dat die grafiese voorstellings wel eienskappe bevat wat as hulpmiddel kan dien in die model bou proses. Die resultate het ook verder getoon dat die nuwe funksionaliteit suksesvol geimplementeer is in die eksperimentele programmatuur.

Sleutel woorde: Lineêre modelle, responsoppervlak, regressie-ontleding, lineêre programmering, parametriese programmering, optimering, voorspelling, parallelle verwerking.

Contents

Chapter 1

Introduction

1.1 Linear models and data

Linear models are used in almost every field of scientific research or situations where the analysis of data is considered. These models express relationships among variables which in turn can be applied in various ways, like prediction, description and optimization. As an example consider a situation where the human resources manager (HRM) of a software company wants to investigate the relationship between the skills of the programmers in the company and their work performance. A measure of their skills could be the level of education, e.g. secondary or tertiary education, the type of qualification, e.g. a diploma, degree, or post-graduate degree, or maybe the number of months experience in different aspects of software development. Irrespective of the composition of the measurement, it will probably be most convenient to use a scale indicating a low skill profile or a high skill profile for each programmer. A measure of each programmer's work performance would typically be the average number of software modules delivered per month, or possibly a scale indicating the frequency of delivering projects on time.

On deciding the skills and work performance measures, the investigation is commenced by randomly identifying a number of programmers and collecting the data. A linear model of the form $y = b_0 + b_1 x$ can now be estimated from the data where the variable y represents the work performance of the programmers and x the skills profile. Usually the variable y is denoted as the dependent variable and the variable x as the independent variable. For this concept one can think of y as depending on the value of x for an estimated b_0 and b_1. A descriptive use of this linear model to the HRM can now be one where the linear model indicates whether there is a positive or negative relationship among the two variables or no linear relationship. For a

positive relationship (the value of b_1 is positive) the deduction can be made that programmers who have a high count for skills profile, perform better in their job. This model can now also be used for predictive purposes: if a candidate applies for a position in the company, the HRM can predict the work performance of the candidate according to his/her skills profile. It is also possible for the HRM to use the model for optimizing the overall work performance of the company according to this model. That is, if a positive relationship exists between work performance and skills profile, and a cause-effect relationship can be inferred, then the HRM can determine what the skills of the employees should be to obtain optimal work performance from them.

For this example only one variable was considered and the use of such a model for optimization can be straight forward. For instance, to obtain a high value for y in the model $y = b_0 + b_1 x$ where b_1 is positive, x should be as high as possible. In real life situations several variables can influence the dependent variable. This makes the optimization of a linear model much more difficult. A research field dedicated to optimization and forecasting techniques for linear (and non-linear) models with multiple variables, is *response surface methodology* (RSM) (see Khuri and Cornell [KC87], Myers and Montgomery [MM95]).

1.2 Objectives of this study

Linear response surface analysis (LRSA) is a subset of the statistical field RSM. It originated from the research done by Bruwer and Hattingh [BH85]. The feature that distinguishes LRSA from RSM in general, is that LRSA can be applied to both planned and raw data, compared to RSM that is mainly concerned with optimization applied to planned data.[1]

The objective of this study is to explore the feasibility of LRSA as a technique for visualizing linear models and data by means of graphic results. The visual representations resulting from LRSA must reflect properties of linear models which will be useful for optimization and forecasting as well as for handling linear model building difficulties such as:

- Interpretation of models in case of interdependence among "independent" variables.

[1]For purposes of this study, planned data will refer to data collected from a designed experiment. Raw data are associated with social sciences where the cases for the variables are collected randomly. The two approaches of data collecting are also referred to as the fixed-X case and variable-X case (see Afifi and Clark [AC96]).

- Determining the importance of a variable to a model, relative to the other variables in the model.
- Deciding which variables to include or exclude from a model in case of multiple linear regression.
- The handling of state variables in case of optimization and forecasting.

1.3 Study approach

A literature study was conducted with regard to various subjects ranging from linear regression analysis, response surface methodology and parallel programming. The literature indicated the desirability of developing a decision support system. This forms an integral part of the study. The system is available as an experimental software package (see Appendix) and the experimental results used for evaluating the outcome of the study, was generated using the system.

1.4 Chapters and contents

In Chapter 2 the theory of linear regression analysis is discussed. The aim of the chapter is to give an overview of the statistical methods used by model builders in the linear regression analysis process. Some of the difficulties associated with the use of linear models are addressed.

Chapter 3 is concerned with the topic of RSM. The focus in this chapter is experimental designs and methodologies used for optimization of linear and non-linear models.

The core theory of LRSA is covered in Chapter 4. An overview of the mathematical programming approach linear programming is given, followed by a discussion on parametric programming.

Contributions made towards LRSA are presented in Chapter 5. Various extensions to LRSA are described in this chapter, which is a direct result of this study.

Chapter 6 presents the outcome of the study objectives. The results of an empirical investigation are presented and evaluated in the light of the objectives.

Finally a summary is given in Chapter 7, which also contains recommendations and a conclusion.

Chapter 2

Linear regression analysis

2.1 Simple linear regression analysis

The relationship expressed by a linear model of the form

$$y = \beta_0 + \beta_1 x \tag{2.1}$$

is one that suggests that the observed values of the variables y and x fall along a straight line with intercept β_0 and the slope as β_1 . For data collected randomly, like e.g. in our example where the programmers are selected randomly, the possibility of the observed values falling exactly along a straight line is very unlikely. For this reason the linear relationship is written as $y = \beta_0 + \beta_1 x + \varepsilon$ where ε can be thought of as a statistical error. The distribution of the error term is assumed to have mean zero and unknown variance σ^2.

If the variable x in our linear model $y = \beta_0 + \beta_1 x + \varepsilon$ is not viewed as a random variable, then for each given value of x, the variable y can be considered to be random with the following mean and unknown variance:

$$E(y|x) = \beta_0 + \beta_1 x \tag{2.2}$$

$$Var(y|x) = Var(\beta_0 + \beta_1 x + \varepsilon) = \sigma^2 \tag{2.3}$$

2.1.1 Estimation of the regression coefficients

The regression coefficients β_0 and β_1 are usually unknown. By using the method of *least squares* the values of β_0 and β_1 can be estimated. Now suppose the sample data collected is (y_1,x_1), (y_2,x_2),..., (y_n,x_n), then we can write

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \quad i = 1, 2, \ldots, n \tag{2.4}$$

The approach followed in the method of least squares is to select values for β_0 and β_1 in order for the sum of squares of the vertical distance between each observation (y_i,x_i) and the straight line $\beta_0 + \beta_1 x_i$ to be a minimum. That is, we need to estimate values for β_0 and β_1 in such a way that:

$$S(\beta_0, \beta_1) = \sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2 \tag{2.5}$$

is a minimum. By setting $\frac{\partial S}{\partial \beta_0} = 0$ and $\frac{\partial S}{\partial \beta_1} = 0$, we find the values of β_0 and β_1 for which 2.5 will be a minimum. This gives

$$\frac{\partial S}{\partial \beta_0} = -2\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i) = 0 \tag{2.6}$$

and

$$\frac{\partial S}{\partial \beta_1} = -2\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)x_i = 0 \tag{2.7}$$

or the equations

$$n\beta_0 + \beta_1 \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \tag{2.8}$$

and

$$\beta_0 \sum_{i=1}^{n} x_i + \beta_1 \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} y_i x_i \tag{2.9}$$

which are called the *least squares normal equations.* From these equations we can solve for the estimators of β_0 and β_1

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} y_i x_i - \frac{(\sum_{i=1}^{n} y_i)(\sum_{i=1}^{n} x_i)}{n}}{\sum_{i=1}^{n} x_i^2 - \frac{(\sum_{i=1}^{n} x_i)^2}{n}} \tag{2.10}$$

and

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x} \tag{2.11}$$

where $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ and $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ are the averages of y_i and x_i respectively.

For ease of use the right hand side of 2.10 can be written in terms of S_{xy} and S_{xx} where

$$\begin{aligned} S_{xy} &= \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) \\ &= \sum_{i=1}^{n}(x_i y_i - x_i \bar{y} - y_i \bar{x} + \bar{x}\bar{y}) \end{aligned}$$

$$
\begin{aligned}
&= \sum_{i=1}^{n} y_i(x_i - \bar{x}) - \bar{y}\sum_{i=1}^{n}(x_i - \bar{x}) \\
&= \sum_{i=1}^{n} y_i(x_i - \bar{x}) \qquad (2.12) \\
&= \sum_{i=1}^{n} y_i x_i - \frac{(\sum_{i=1}^{n} y_i)(\sum_{i=1}^{n} x_i)}{n}
\end{aligned}
$$

(2.13)

represents the corrected sum of products of x and y, and

$$
\begin{aligned}
S_{xx} &= \sum_{i=1}^{n}(x_i - \bar{x})^2 \\
&= \sum_{i=1}^{n}(x_i - 2x_i\bar{x} + \bar{x}^2) \\
&= \sum_{i=1}^{n} x_i^2 - 2\sum_{i=1}^{n} x_i\bar{x} + n\bar{x}^2 \\
&= \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i\bar{x} \\
&= \sum_{i=1}^{n} x_i^2 - \frac{(\sum_{i=1}^{n} x_i)^2}{n}
\end{aligned}
$$

(2.14)

which is the corrected sum of squares of x. Now equation 2.10 can be written as

$$\hat{\beta}_1 = \frac{S_{xy}}{S_{xx}} \qquad (2.15)$$

For future reference the corrected sum of squares of y is given as

$$S_{yy} = \sum_{i=1}^{n}(y_i - \bar{y})^2 \qquad (2.16)$$

which is a measure of the variability of the observations y_i.

2.1.2 The significance of regression

With a regression model at hand, it is helpful to know whether the estimated regression coefficients are of some significance. That is, how close does the estimated line $y = \hat{\beta}_0 + \hat{\beta}_1 x$ come to the true population model $y = \beta_0 + \beta_1 x$? This question can be dealt with if more information is available about the distribution of the estimators $\hat{\beta}_0$ and $\hat{\beta}_1$ around their targets β_0 and β_1 respectively.

Distribution properties

From equation 2.10 and 2.11 it is clear that $\hat{\beta}_1$ and $\hat{\beta}_0$ can be written as a linear combination of y_i. For example

$$\hat{\beta}_1 = \frac{S_{xy}}{S_{xx}} = \sum_{i=1}^{n} c_i y_i \tag{2.17}$$

where $c_i = (x_i - \bar{x})/S_{xx}$ for $i = 1, 2, \ldots, n$
and

$$\hat{\beta}_0 = \bar{y} - \bar{x}\hat{\beta}_1 \tag{2.18}$$

Before deriving the standard deviations for $\hat{\beta}_1$ and $\hat{\beta}_0$, the bias property of the two estimators must first be considered:

$$\begin{aligned} E(\hat{\beta}_1) &= E(\sum_{i=1}^{n} c_i y_i) \\ &= \sum_{i=1}^{n} c_i E(y_i) \\ &= \sum_{i=1}^{n} c_i (\beta_0 + \beta_1 x_i) \\ &= \beta_0 \sum_{i=1}^{n} c_i + \beta_1 \sum_{i=1}^{n} c_i x_i \end{aligned}$$

Since $E(\varepsilon_i) = 0$ by assumption and

$$\sum_{i=1}^{n} c_i = \frac{\sum_{i=1}^{n} (x_i - \bar{x})}{S_{xx}} = 0 \tag{2.19}$$

and

$$\begin{aligned} \sum_{i=1}^{n} c_i x_i &= \frac{\sum_{i=1}^{n} (x_i - \bar{x}) x_i}{S_{xx}} \\ &= \frac{\sum_{i=1}^{n} (x_i^2 - \bar{x} x_i)}{S_{xx}} \\ &= \frac{\sum_{i=1}^{n} x_i - \frac{(\sum_{i=1}^{n} x_i)^2}{n}}{S_{xx}} \\ &= \frac{S_{xx}}{S_{xx}} \\ &= 1 \end{aligned}$$

the result is

$$E(\hat{\beta}_1) = \beta_1 \tag{2.20}$$

which shows that $\hat{\beta}_1$ is an unbiased estimator of β_1. For β_0 we have:

$$\begin{aligned} E(\hat{\beta}_0) &= E(\bar{y} - \bar{x}\hat{\beta}_1) \\ &= E(\bar{y}) - E(\bar{x}\hat{\beta}_1) \\ &= \frac{1}{n}\sum_{i=1}^{n} E(y_i) - \bar{x}E(\hat{\beta}_1) \\ &= \frac{1}{n}\sum_{i=1}^{n}(\beta_0 + x_i\beta_1) - \bar{x}\beta_1 \\ &= \frac{1}{n}\sum_{i=1}^{n}\beta_0 + \bar{x}\beta_1 - \bar{x}\beta_1 \\ &= \beta_0 \end{aligned}$$

which shows that $\hat{\beta}_0$ is also an unbiased estimator of β_0.

For the standard deviations of $\hat{\beta}_1$ and $\hat{\beta}_0$ we have the following variances:

$$\begin{aligned} Var(\hat{\beta}_1) &= Var(\sum_{i=1}^{n} c_i y_i) \\ &= \sum_{i=1}^{n} c_i^2 Var(y_i) \\ &= \sigma^2 \sum_{i=1}^{n} c_i^2 \\ &= \frac{\sigma^2 \sum_{i=1}^{n}(x_i - \bar{x})^2}{S_{xx}^2} \\ &= \frac{\sigma^2}{S_{xx}} \end{aligned}$$

and

$$\begin{aligned} Var(\hat{\beta}_0) &= Var(\bar{y} - \hat{\beta}_1\bar{x}) \\ &= Var(\bar{y}) + \bar{x}^2 Var(\hat{\beta}_1) - 2\bar{x}Cov(\bar{y}, \hat{\beta}_1) \end{aligned}$$

where $Var(\bar{y}) = \sigma^2/n$ and

$$\begin{aligned} Cov(\bar{y}, \hat{\beta}_1) &= Cov(\bar{y}, \frac{\sum_{i=1}^{n}(x_i - \bar{x})y_i}{S_{xx}}) \\ &= \frac{1}{S_{xx}} Cov(\frac{y_1 + \ldots + y_n}{n}, x_1 y_1 \ldots + x_n y_n - \bar{x}(y_1 + \ldots + y_n)) \end{aligned}$$

$$= \frac{1}{S_{xx}}(\sigma^2(\frac{\sum_{i=1}^{n} x_i}{n}) - \bar{x}(\frac{n\sigma^2}{n}))$$
$$= 0$$

which result in

$$Var(\hat{\beta}_0) = \frac{\sigma^2}{n} + \bar{x}^2\frac{\sigma^2}{S_{xx}}$$
$$= \sigma^2(\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}}) \quad (2.21)$$

Estimation of σ^2

Before the distribution properties of $\hat{\beta}_1$ and $\hat{\beta}_0$ can be used in tests for significance, the parameter σ^2 needs to be estimated. Recall from previous assumptions that $Var(y|x) = Var(\varepsilon) = \sigma^2$ is unknown. The parameter can be estimated by using the *residual sum of squares* where a *residual* is defined as the difference between the observed value y_i, and the corresponding fitted value $\hat{y}_i$. That is, the ith residual is

$$e_i = y_i - \hat{y}_i = y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_1),\ i = 1, 2, \ldots, n \quad (2.22)$$

Now the residual sum of squares (also the *error sum of squares*) is given as

$$SS_E = \sum_{i=1}^{n} e_i^2$$
$$= \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

and if $\hat{\beta}_0 + \hat{\beta}_1 x_i$ is substituted for $\hat{y}_i$, we obtain

$$SS_E = \sum_{i=1}^{n} y_i^2 - n\bar{y}^2 - \hat{\beta}_1 S_{xy} \quad (2.23)$$

where $\sum_{i=1}^{n} y_i^2 - n\bar{y}^2 = \sum_{i=1}^{n}(y_i - \bar{y})^2 = S_{yy}$ is the corrected sum of squares for the observations as defined in 2.16. Finally we can write down a simplified expression for SS_E,

$$SS_E = S_{yy} - \hat{\beta}_1 S_{xy} \quad (2.24)$$

The estimation of σ^2 by SS_E involves two estimates namely $\hat{\beta}_0$ and $\hat{\beta}_1$. For this reason $n - 2$ degrees of freedom are associated with the estimator SS_E. Thus, the expected value of SS_E is $E(SS_E) = (n - 2)\sigma^2$ and so an unbiased estimator of σ^2 is

$$\hat{\sigma^2} = \frac{SS_E}{n - 2} = MS_E \quad (2.25)$$

where MS_E is called the *residual mean square.*

Hypothesis testing

Hypothesis testing involves using a test statistic to determine the significance of the estimated regression coefficients. The procedure requires an additional assumption about the regression model in order to deduce the distribution of the test statistic. This assumption is for the terms ε_i to be normally distributed. The complete set of assumptions for the error terms is that ε_i for $i = 1, 2, \ldots, n$ is normally and independently distributed with mean zero and variance σ^2. An abbreviation for these assumptions is ε_i for $i = 1, 2, \ldots, n$ is distributed NID(0,σ^2).

Now, consider an example where the significance of β_1 equals some constant, say β_{10}, needs to be tested, then the formal hypothesis is

$$\begin{aligned} H_0 &: \beta_1 = \beta_{10} \\ H_1 &: \beta_1 \neq \beta_{10} \end{aligned} \tag{2.26}$$

The distribution of the statistic associated with this hypothesis can now be deduced from the following: the observations y_i are NID($\beta_0 + \beta_1 x_i$,σ^2) distributed following from the assumption that the error terms are NID(0,σ^2) distributed. The esimator $\hat{\beta}_1$ can be written as a linear combination of y_i (2.17) and is, therefore, also normally distributed with mean β_1 (2.20) and variance σ^2/S_{xx} (2.1.2). The test statistic takes the form of a normalization function

$$Z_0 = \frac{\hat{\beta}_1 - \beta_{10}}{\sqrt{\sigma^2/S_{xx}}} \tag{2.27}$$

and consequently is distributed N(0,1). If the parameter σ^2 is unknown, the unbiased estimator MS_e can be used with $(n-2)MS_E/\sigma^2$ having a distribution of χ^n_{n-1}. The resulting test statistic where σ^2 is replaced with MS_E is

$$t_0 = \frac{\hat{\beta}_1 - \beta_{10}}{\sqrt{MS_E/S_{xx}}} \tag{2.28}$$

which is distributed as t with $n-2$ degrees of freedom. The $n-2$ degrees of freedom are the degrees of freedom associated with the estimator MS_E. Now, the result of this statistic is compared to the upper $\alpha/2$ percentage point of the t_{n-2} distribution ($t_{\alpha/2,n-2}$). If

$$|t_0| > t_{\alpha/2,n-2} \tag{2.29}$$

then the null hypothesis $H_0 : \beta_1 = \beta_{10}$ is rejected.

For the regression coefficient β_0, a similar hypothesis test can be used,

$$H_0 : \beta_0 = \beta_{00}$$
$$H_1 : \beta_0 \neq \beta_{00}$$

where $H_0 : \beta_0 = \beta_{00}$ is rejected if $|t_0| > t_{\alpha/2,n-2}$ with

$$t_0 = \frac{\hat{\beta}_0 - \beta_{00}}{\sqrt{MS_E(1/n + \bar{x}^2/S_{xx})}} \tag{2.30}$$

The hypothesis relating to the *significance of regression* is a special case of 2.26 and can be stated as follows:

$$H_0 : \beta_1 = 0$$
$$H_1 : \beta_1 \neq 0 \tag{2.31}$$

The semantics of the statement implies that if $H_0 : \beta_1 = 0$ is rejected, the estimated regression coefficient $\hat{\beta}_1$ differs significantly from zero, which in turn implies that there is a linear relationship between x and y. On the other hand, failing to reject the null hypothesis implies that no linear relationship between x and y can be inferred from the data.

Two approaches exist for testing the hypothesis 2.31. For the first approach recall that the measure of the variability in the observations y_i is S_{yy} (2.16). It can be shown that

$$S_{yy} = SS_R + SS_E \tag{2.32}$$

where $SS_R = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$ is called the *regression sum of squares.* By comparing 2.32 with 2.24 it is evident that the regression sum of squares can be computed as

$$SS_R = \hat{\beta}_1 S_{xy} \tag{2.33}$$

The test statistic is

$$F_0 = \frac{SS_R/10}{SS_E/(n-2)} = \frac{MS_R}{MS_E} \tag{2.34}$$

where MS_R is the *regression mean square.* The test statistic F_0 follows the $F_{1,n-2}$ distribution. The hypothesis $H_0 : \beta_1 = 0$ is rejected if $F_0 > F_{\alpha,1,n-2}$.

The second approach for testing the significance of regression uses the t-test equation (2.28) with $\beta_{10} = 0$.

$$t_0 = \frac{\hat{\beta}_1}{\sqrt{MS_E/S_{xx}}} \tag{2.35}$$

Note, however, that if both sides of 2.35 is squared we obtain

$$t_0^2 = \frac{\hat{\beta}_1^2}{S_{xx}} MS_E = \frac{\hat{\beta}_1 S_{xy}}{MS_E} = \frac{MS_R}{MS_E} \tag{2.36}$$

which is identical to F_0 in 2.34.

2.1.3 Coefficient of determination

An approach to quantify the worth of a linear regression model is to use either the error sum of squares, SS_E or the regression sum of squares, SS_R and divide the one to be used by S_{yy} in order to obtain a measure of the variability of y remaining after x has been considered. That is

$$R^2 = \frac{SS_R}{S_{yy}} = 1 - \frac{SS_E}{S_{yy}} \tag{2.37}$$

is the proportion of the variation explained by the independent variable x and is called the coefficient of determination. Because $0 \leq SS_E \leq S_{yy}$, it follows that $0 \leq R^2 \leq 1$ and values for R^2 close to 1 imply that a large portion of the variability in y is explained by the variability in the independent variable x.

2.2 Multiple linear regression analysis

The simple linear regression model (2.1) with only one independent variable can be extended to a multiple linear regression model of the form

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \ldots + \beta_k x_k + \varepsilon \tag{2.38}$$

where k number of independent variables are present. For this model the same set of assumptions concerning the error terms ε_i is applicable. That is, the errors are uncorrelated with mean zero and variance σ^2. Furthermore, the regression coefficients β_0, β_1, ..., β_k are unknown and need to be estimated.

2.2.1 Estimation of β_0, β_1, ..., β_k

The method of least squares

The method of least squares can now also be applied to the multiple linear regression model to estimate the unknown parameters. The sample multiple

regression model is

$$\begin{aligned} y_i &= \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i}, \ldots + \beta_k x_{ki} + \varepsilon_i \\ &= \beta_0 + \sum_{j=1}^{k} \beta_j x_{ij} + \varepsilon_i, \ i = 1, 2, \ldots, n \end{aligned} \tag{2.39}$$

for the sample data $(y_i, x_{1i}, x_{2i}, \ldots, x_{ki})$, $i = 1, 2, \ldots, n$. Now, the least squares approach is to select values for β_0, β_1, ..., β_k in order for

$$S(\beta_0, \beta_1, \ldots, \beta_k) = \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{k} \beta_j x_{ij})^2 \tag{2.40}$$

to be a minimum. This is achieved by setting $\frac{\partial S}{\partial \beta_0} = 0$ and $\frac{\partial S}{\partial \beta_j} = 0$, $j = 1, 2, \ldots, k$. and solving for β_0 and β_j, $j = 1, 2, \ldots, k$. That is

$$\frac{\partial S}{\partial \beta_0} = -2 \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{k} \beta_j x_{ij}) = 0 \tag{2.41}$$

and

$$\frac{\partial S}{\partial \beta_1} = -2 \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{k} \beta_j x_{ij}) x_{ij} = 0, \ j = 1, 2, \ldots, k \tag{2.42}$$

which result in the least squares normal equations

$$\begin{array}{lllll} n\beta_0 & +\beta_1 \sum_{i=1}^{n} x_{i1} & +\beta_2 \sum_{i=1}^{n} x_{i2} & +\ldots & +\beta_k \sum_{i=1}^{n} x_{ik} = \sum_{i=1}^{n} y_i \\ \beta_0 \sum_{i=1}^{n} x_{i1} & +\beta_1 \sum_{i=1}^{n} x_{i1}^2 & +\beta_2 \sum_{i=1}^{n} x_{i1} x_{i2} & +\ldots & +\beta_k \sum_{i=1}^{n} x_{i1} x_{ik} = \sum_{i=1}^{n} x_{i1} y_i \\ \vdots & \vdots & \vdots & \vdots & = \quad \vdots \\ \beta_0 \sum_{i=1}^{n} x_{ik} & +\beta_1 \sum_{i=1}^{n} x_{ik} x_{i1} & +\beta_2 \sum_{i=1}^{n} x_{ik} x_{i2} & +\ldots & +\beta_k \sum_{i=1}^{n} x_{ik}^2 = \sum_{i=1}^{n} x_{i1} y_i \end{array} \tag{2.43}$$

The solution to these p=k+1 normal equations is the least squares estimators $\hat{\beta}_0$, $\hat{\beta}_1$, ..., $\hat{\beta}_k$. A more convenient way of representing multiple regression models is by using matrix notations. Therefore, the same procedure just discussed will now be repeated with matrix notation.

Using matrix notation for least squares

The multiple linear regression model in terms of the observations (2.39) can be expressed in matrix notation as

$$\mathbf{y} = \mathbf{X}\beta + \varepsilon \tag{2.44}$$

where

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1k} \\ 1 & x_{21} & x_{22} & \dots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{nk} \end{bmatrix}$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \varepsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

Applying the method of least squares requires from us to find the vector β that will minimize

$$\begin{aligned} S(\beta) &= \sum_{i=1}^{n} \varepsilon_i^2 = \varepsilon'\varepsilon \\ &= (\mathbf{y} - \mathbf{X}\beta)'(\mathbf{y} - \mathbf{X}\beta) \\ &= \mathbf{y}'\mathbf{y} - \beta'\mathbf{X}'\mathbf{y} - \mathbf{y}'\mathbf{X}\beta + \beta'\mathbf{X}'\mathbf{X}\beta \\ &= \mathbf{y}'\mathbf{y} - 2\beta'\mathbf{X}'\mathbf{y} + \beta'\mathbf{X}'\mathbf{X}\beta \end{aligned} \tag{2.45}$$

Therefore

$$\frac{\partial \mathbf{S}}{\partial \beta} = -2\mathbf{X}'\mathbf{y} + 2\mathbf{X}'\mathbf{X}\beta = \mathbf{0} \tag{2.46}$$

which simplifies to

$$\mathbf{X}'\mathbf{X}\beta = \mathbf{X}'\mathbf{y} \tag{2.47}$$

that yields the least squares normal equations. The solution to 2.47 is the least squares estimators $\hat{\beta}$ which can be obtained from multiplying both sides of 2.47 with the inverse of $\mathbf{X}'\mathbf{X}$. That is

$$\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} \tag{2.48}$$

provided that $(\mathbf{X}'\mathbf{X})^{-1}$ exists.

To evaluate the results of using matrix notation, 2.47 can be written as

$$\begin{bmatrix} n & +\sum_{i=1}^{n} x_{i1} & +\sum_{i=1}^{n} x_{i2} & +\ldots & +\sum_{i=1}^{n} x_{ik} \\ \sum_{i=1}^{n} x_{i1} & +\sum_{i=1}^{n} x_{i1}^2 & +\sum_{i=1}^{n} x_{i1}x_{i2} & +\ldots & +\sum_{i=1}^{n} x_{i1}x_{ik} \\ \vdots & \vdots & \vdots & & \vdots \\ \sum_{i=1}^{n} x_{ik} & +\sum_{i=1}^{n} x_{ik}x_{i1} & +\sum_{i=1}^{n} x_{ik}x_{i2} & +\ldots & +\sum_{i=1}^{n} x_{ik}^2 \end{bmatrix} \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \vdots \\ \hat{\beta}_k \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_{i1}y_i \\ \vdots \\ \sum_{i=1}^{n} x_{ik}y_i \end{bmatrix}$$

which is identical to the scalar version of the multiple least squares normal equations 2.43 .

2.2.2 The significance of multiple regression

Similar to simple linear regression, we need to obtain information about the distribution of $\hat{\beta}$ in order to do hypothesis testing concerning the significance of the regression.

Distribution properties

Following the matrix notation, it can be shown that $\hat{\beta}$ is an unbiased estimator for β.

$$\begin{aligned} E(\hat{\beta}) &= E\left[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}\right] \\ &= E\left[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{X}\beta + (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\varepsilon\right] \\ &= \beta \end{aligned}$$

since $E(\varepsilon) = \mathbf{0}$ and $(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{X} = \mathbf{I}$. The variance property of $\hat{\beta}$ is expressed by

$$\begin{aligned} Var(\hat{\beta}) &= Var\left[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}\right] \\ &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'Var(\mathbf{y})\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \\ &= (\mathbf{X}'\mathbf{X})^{-1}\sigma^2 \end{aligned}$$

since $Var(\mathbf{y}) = Var(\varepsilon) = \sigma^2\mathbf{I}$. Now, with $\mathbf{C} = (\mathbf{X}'\mathbf{X})^{-1}$ a $p \times p$ symmetric matrix, the variance of $\hat{\beta}_j$ is $C_{jj}\sigma^2$ and the co-variance between $\hat{\beta}_i$ and $\hat{\beta}_j$ is given by the off-diagonal element $C_{ij}\sigma^2$.

Estimation of σ^2

As in the simple linear regression model, the value of σ^2 needs to be estimated in order to use the distribution properties of $\hat{\beta}$ for hypothesis testing.

By defining the residuals in matrix notation as $\mathbf{e} = \mathbf{y} = \mathbf{X}\hat{\beta}$, the residual sum of squares becomes

$$\begin{aligned} SS_E &= \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \\ &= \sum_{i=1}^{n} e_i^2 \\ &= \mathbf{e}'\mathbf{e} \\ &= (\mathbf{y} - \mathbf{X}\hat{\beta})'(\mathbf{y} - \mathbf{X}\hat{\beta}) \\ &= \mathbf{y}'\mathbf{y} - \hat{\beta}\mathbf{X}'\mathbf{y} - \mathbf{y}'\mathbf{X}\hat{\beta} + \hat{\beta}'\mathbf{X}'\mathbf{X}\hat{\beta} \\ &= \mathbf{y}'\mathbf{y} - 2\hat{\beta}\mathbf{X}'\mathbf{y} + \hat{\beta}'\mathbf{X}'\mathbf{X}\hat{\beta} \end{aligned}$$

where $\mathbf{X}'\mathbf{X}\hat{\beta} = \mathbf{X}'\mathbf{y}$ and thus,

$$SS_E = \mathbf{y}'\mathbf{y} - \hat{\beta}'\mathbf{X}'\mathbf{y} \tag{2.49}$$

The degrees of freedom associated with the residual sum of squares is $n - p$ since p parameters are estimated in the regression model. Now, as in the linear regression model, it can be shown that an unbiased estimator for σ^2 is given by

$$MS_E = \frac{SS_E}{n - p} \tag{2.50}$$

that is, $E(MS_E) = \sigma^2$.

Hypothesis testing

The test for the significance of regression for multiple linear regression is similar to the hypothesis procedure used in simple linear regression. For the case where the significance of a specific regression coefficient, say $\hat{\beta}_j$, is investigated, the following statement applies:

$$\begin{aligned} H_0 &: \beta_1 = 0 \\ H_1 &: \beta_1 \neq 0 \end{aligned} \tag{2.51}$$

The test statistic for this hypothesis is

$$t_0 = \frac{\hat{\beta}_j}{\sqrt{\sigma^2 C_{jj}}} \tag{2.52}$$

and the null hypothesis is rejected if $|t_0| > t_{\alpha/2,n-k-1}$. That is, the null hypothesis is rejected when the regression coefficient $\hat{\beta}_j$ differs significantly from zero indicating a linear relationship between x_j and y.

If the significance of the overall model is tested, the hypothesis statement is extended to

$$H_0 : \beta_1 = \beta_2 = \ldots = \beta_k = 0$$
$$H_1 : \beta_j \neq 0 \text{ for at least one } j.$$

Rejecting $H_0 : \beta_j = 0$ implies that at least one of the variables x_1, x_2, ..., x_k contributes significantly to the model. The test statistic used for testing this hypothesis is a generalization of that used in simple linear regression. Consider the corrected sum of squares S_{yy} (2.16). From 2.32 we can write S_{yy} in terms of the sum of squares due to regression and the residual sum of squares

$$S_{yy} = SS_R + SS_E \tag{2.53}$$

If $H_0 : \beta_j = 0$ is true, then SS_R/σ^2 follows a χ^2_k distribution and SS_E/σ^2 follows a χ^2_{n-k-1} distribution. Furthermore, it can be shown that SS_E and SS_R are independent. The test statistic is

$$F_0 = \frac{SS_R/k}{SS_E/(n-k-1)} = \frac{MS_R}{MS_E} \tag{2.54}$$

and $H_0 : \beta_j = 0$ is rejected if $F_0 > F_{\alpha,k,n-k-1}$.

2.2.3 Coefficient of multiple determination

The coefficient of determination from simple linear regression analysis is extended for multiple linear regression. It measures the reduction in the variability of y obtained by including the variables x_1, x_2, ..., x_k in the model. The coefficient of multiple determination is defined as

$$R^2 = \frac{SS_R}{S_{yy}} = 1 - \frac{SS_E}{Syy} \tag{2.55}$$

The derivation of R^2 stays exactly the same as for the simple linear regression case. Values of R^2 close to 1 imply that most of the variation in y is explained by the independent variables x_1, x_2, ..., x_k.

Another measure of model adequacy frequently used in multiple linear regression is *adjusted* R^2. The ordinary R^2 measure will always increase when a new variable is added to the model. This suggests that an adequate model may sometimes be obtained by adding just enough variables to the

model, even though the variables might not be relevant. The $\bar{R}^2$ measure takes into account the number of parameters estimated in the model and penalizes the model for the inclusion of unnecessary variables in the model.

$$\bar{R}^2 = 1 - \frac{SS_E/(n-p)}{S_{yy}(n-1)} = 1 - \frac{n-1}{n-p}(1-R^2) \tag{2.56}$$

where p is the number of parameters estimated.

2.3 Variable selection

The use of the measure $\bar{R}^2$, which was discussed in the previous section, suggests that some variables should be removed from a multiple regression model in order to prevent *over-fitting* of the model. Many practitioners remove any independent variable from the model when there is not sufficient reason for rejecting $H_0 : \beta_j = 0$. By doing this, the values of the regression coefficients associated with the other independent variables may be altered when the model is re-fitted to the data, indicating that the remaining variables are now of more (or less) importance to the model. It is evident that two conflicting objectives are present: the model must be prevented from over-fitting, but "enough" variables are needed in the model to reveal the real relationship between y and the factors influencing y.

The goal of variable selection procedures is to find some compromise between the conflicting objectives. In other words, the "best" subset of variables needs to be selected for inclusion into the model. Unfortunately, there is no unique definition of "best" and we will see that each algorithm may produce a different subset which is considered to be the "best".

2.3.1 Forward selection

This algorithm starts with the current subset of variables empty. The approach is to add variables to the subset in order to maximize the multiple correlation between the subset of variables and y. That is, the variable with the highest correlation with y, say x_1, is inserted first. The variable x_1 will also be the one that will produce the largest value of the F-statistic for testing significance of regression. The next variable to enter the subset, say x_2, will be the one that has the largest correlation with y after adjusting for the effect of the first variable entered (x_1) on y. This means that the *partial* correlation between x_2 and y is calculated with the linear effect of x_1 removed. The variable x_2 will now be the one that will produce the largest *partial*

F-statistic. The partial F-statistic with the linear effect of x_1 removed is stated as

$$F = \frac{SS_R(x_2|x_1)}{MS_E(x_1, x_2)} \tag{2.57}$$

where

$$SS_R(x_2|x_1) = SS_R(\beta_2|\beta_1) = SS_R(\beta_0, \beta_1, \beta_2) - SS_R(\beta_0, \beta_1) \tag{2.58}$$

In practice a cut-off value for the significance test is specified, say F_{IN} (or F-to-enter), and variables are added to the subset as long as $F > F_{IN}$, where F is the partial test for significance calculated for the candidate variable. The procedure terminates either when the partial F-statistic for a candidate variable does not exceed F_{IN} or when the last candidate variable is added to the model.

2.3.2 Backward elimination

The starting point of this approach is one with all the variables included in the subset. The partial F-statistic is computed for each of the variables in the subset as if it were the last variable to enter the model. The variable to leave the subset will then be the one for which the partial F-statistic was the smallest. That is, the least useful variable will be removed from the subset.

The procedure of removing variables from the subset is continued as long as $F < F_{OUT}$ where F_{OUT} (or F-to-remove) is a preselected cut-off value for the significance test.

2.3.3 Stepwise regression

This algorithm is a combination of the forward and backward selection procedures. The algorithm commences with the forward selection where variables are added to the subset based on the largest partial F-statistic obtained among all candidates. After adding a variable to the subset all the variables in the subset are reassessed via their partial F-statistic. If the partial F-statistic for some variable is less than F_{OUT}, it is dropped from the subset. Variables are added to the subset only if their partial F-statistic is larger than F_{IN}. The procedure is terminated if no more variables can be added or removed from the subset.

In practice the preselected values F_{IN} and F_{OUT} are often chosen in such a way that $F_{IN} = F_{OUT}$. Montgomery and Peck [MP92] suggest that $F_{IN} > F_{OUT}$ so that it will be more difficult to add variables to the subset than to delete them.

2.3.4 All possible subsets

A common shortcoming of the methods forward selection, backward elimination and stepwise regression, is the inability to evaluate all possible subsets of variables.

The implicit or explicit enumeration of all possible combinations of variables (see Beale [EMD67]) gives the benefit of delivering the "best" subset, the second "best" subset, etc. of the chosen cardinality. There are various criteria which can be used in conjunction with this approach to identify the best subsets with. For instance, measures like R^2 and $\bar{R}^2$ can be computed for each subset. Other criteria like Mallows C_p are covered in detail by authors like Montgomery and Peck [MP92], Rayn [Ray97], and others.

2.4 Multicollinearity

2.4.1 Planned data vs. raw data

Data collected through controlled experimentation is labelled as planned data. The experiments are conducted according to an experimental design with the goal of selecting values for the independent variables x_1, x_2,..., x_k in such a way as to obtain *orthogonality*. The result is the absence of strong linear dependencies among the k independent variables.

Linear regression analysis is frequently applied to raw data gathered from observational studies. There is no guarantee that the columns of the independent variables from the raw data are going to be orthogonal. In such cases where some of the independent variables are highly intercorrelated, *multicollinearity* is said to be present.

2.4.2 The effects of multicollinearity

Consider the model

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon \tag{2.59}$$

where the variables x_1, x_2 and y are scaled to unit length. The least squares normal equations are

$$\begin{aligned} (\mathbf{X}'\mathbf{X})\hat{\beta} &= \mathbf{X}'\mathbf{y} \\ \begin{bmatrix} 1 & r_{12} \\ r_{12} & 1 \end{bmatrix} \begin{bmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \end{bmatrix} &= \begin{bmatrix} r_{1y} \\ r_{2y} \end{bmatrix} \end{aligned} \tag{2.60}$$

where r_{12} is the simple correlation between x_1 and x_2 and r_{jy} is the simple correlation between x_j and y, $j = 1, 2$. The inverse of $\mathbf{X}'\mathbf{X}$ is

$$\mathbf{C} = (\mathbf{X}'\mathbf{X})^{-1} = \begin{bmatrix} \frac{1}{(1-r_{12}^2)} & \frac{-r_{12}}{(1-r_{12}^2)} \\ \frac{-r_{12}}{(1-r_{12}^2)} & \frac{1}{(1-r_{12}^2)} \end{bmatrix} \tag{2.61}$$

The variances of $\hat{\beta}_1$ and $\hat{\beta}_2$ are $C_{11}\sigma^2$ and $C_{22}\sigma^2$ respectively. It is evident from 2.61 that if x_1 and x_2 are correlated, that is $|r_{12}| \to 1$, then $Var(\hat{\beta}_j) = C_{jj}\sigma^2 \to \infty$. This shows that multicollinearity between variables is responsible for large variances of the estimated regression coefficients.

For regression models consisting of more than two variables, the effect of multicollinearity can be demonstrated in a similar fashion. It can be shown (see Montgomery and Peck [MP92]) that the diagonal elements of $\mathbf{C} = (\mathbf{X}'\mathbf{X})^{-1}$ can be written as

$$C_{jj} = \frac{1}{1 - R_j^2}, \; j = 1, 2, \ldots, p \tag{2.62}$$

where R_j^2 is the coefficient of multiple determination from the regression of x_j on the remaining $p - 1$ independent variables. Now, if there is a strong correlation between x_j and any subset of the other $p - 1$ independent variables, then the value of R_j^2 will be close to one. The result will be $Var(\hat{\beta}_j) = C_{jj}\sigma^2 = \frac{1}{1-R_j^2}\sigma^2 \to \infty$. It is convenient to think about the quantity C_{jj} as the factor by which the variance of $\hat{\beta}_j$ is increased due to the presence of multicollinearity. This factor is commonly referred to as the *variance inflation factor* (VIF) and is used frequently for the detection of multicollinearity. In practice values of VIF exceeding 5 or 10 can be an indication of the presence of multicollinearity.

2.4.3 Remedial measures

Although there is no guarantee that raw data will be free of multicollinearity, some methods exist to deal with the problems induced by mulicollinearity.

Collecting additional data

According to Montgomery and Peck [MP92], the collection of additional data is considered to be the best approach to deal with the problem of multicollinearity. The additional data should be collected in such a way that the effect of multicollinearity is reduced. For example, if two variable x_i and x_j are positively correlated (high values of x_i are associated with high values of x_j), then additional data must be collected for cases where low values of x_i are associated with high values of x_j, or vice versa. Unfortunately, additional data with the desired properties (i.e. low values of x_i associated with

high values x_j) may not exist in the population or economic constraints may prevent the collection of additional data.

Model respecification

When multicollinearity is detected among some of the independent variables, consideration can be given to eliminating the variables which are responsible for the intercorrelation. This can be done in conjunction with the variable selection procedures where candidate subsets of variables can be generated to improve the criteria measure (e.g. $\bar{R}^2$) and to decrease the effect of multicollinearity.

Ridge regression

The effect of multicollinearity is the inflation of the variances of the estimated regression coefficients. The approach of ridge regression entails the use of a biased estimator with variance smaller than an unbiased estimator. By doing this we can obtain estimated regression coefficients where the effect of inflated variances are minimized. The method of ridge regression is considered to be very effective especially if used in conjunction with variable selection procedures. Ridge regression is discussed in detail by various authors, see Montgomery and Peck [MP92], Rayn [Ray97] etc.

2.5 Regression for control and optimization

Regression is applied as a mechanism for control when attempting to control the value of y in a system through the manipulation of x in the linear model

$$y = \beta_0 + \beta_1 x \tag{2.63}$$

For such an application a cause-effect relationship is assumed. This is only possible if the model is fitted to planned data (Rayn [Ray97]). This way we know that the variable y responded to a change in the value of x. Caution must be taken when attempting to use linear models for optimization and forecasting if multicollinearity is present. Consider the following example where data from some region are analyzed to investigate the feasibility of an irrigation scheme (Wonnacott [WW81]).

Year	Production(y)	Rainfall (x_1)	Temperature (x_2)
1963	60	8	56
1964	50	10	47
1965	70	11	53
1966	70	10	53
1967	80	9	56
1968	50	9	47
1969	60	12	44
1970	40	11	44

Table 2.1: Data on production for Example 2.1.

EXAMPLE 2.1 *For the data listed in Table 2.1 the variable Production (y) is taken as the dependent variable and the variables Rainfall (x_1) and Temperature (x_2) as the independent variables. By applying the method of least squares we obtain the estimated regression equation*

$$\hat{y} = -144.6 + 5.71x_1 + 2.95x_2 \tag{2.64}$$

with the coefficient of determination $R = 0.889$ and t-values for the coefficients b_0, b_1 and b_2 as -2.59, 2.13 and 4.26 respectively. These statistics indicate a reasonable fit of the model to the data points and both coefficients for the variables x_1 and x_2 are positive.

It would seem logical from Example 2.1 that in order to obtain high levels of production, we need to set the rainfall as high as possible, that is, we should implement an irrigation scheme. But on further investigation we notice that there is some intercorrelation among the two independent variables Rainfall and Temperature.

From the scatter diagram shown in Figure 2.1 it is evident that when high levels of rainfall occurred, low levels of temperature were measured and when low levels of rainfall occurred, high levels of temperature were measured. In other words, there is a negative correlation between Rainfall and Temperature, which indicates that multicollinearity exists. An implementation of an irrigation scheme in a region where high temperatures are present would, therefore, not necessarily increase production. This can be observed by looking at Figure 2.1. There is no area of experience where both Rainfall and Temperature take on high values (see north-eastern corner of Figure 2.1).

Our first logical proposition of high rainfall levels that would improve production may consequently not be the optimal solution to the problem being investigated.

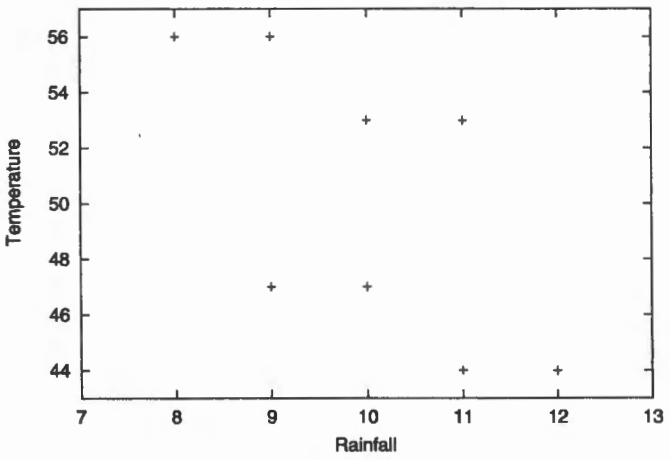

Figure 2.1: Scatter plot of Rainfall vs. Temperature.

2.6 Summary

Linear regression analysis is used in the linear model-building process where the first step is to estimate the unknown parameters in the linear model. Hypothesis testing is performed to determine the significance of the estimated parameters and measures like R^2 are used to verify the adequacy of the linear model in general. Various techniques exist for variable selection when the linear model needs to be reduced, or when an attempt is made to improve the fit. In the case where multicollinearity is present the model builder might consider ridge regression or even efforts to collect additional data. Once the model is found to be satisfactory, it can be applied for descriptive, forecasting and optimization purposes.

The use of linear models for optimization has become an important application. The previous section, however, suggests that linear models should be used for optimization only when the data that has been used for the model estimation, is free from interdependencies among the independent variables. Example 2.1 illustrated the difficulty in making meaningful suggestions for optimizing a linear model with the independent variables intercorrelated. Since the data set for the example only had two independent variables, it can be expected that meaningful suggestions for optimizing a linear model where more variables are present (with some of them highly intercorrelated), would even be more difficult. *Response surface methodology* is concerned with the investigation and optimization of linear (and non-linear) models, where the emphasis is on designing experiments in order to eliminate the possibility of multicollinearity. The topic of response surface methodology is covered in the next chapter.

Chapter 3

Response surface methodology

3.1 Introduction

Response surface methodology (RSM) was formally developed by G.E.P. Box and K.B. Wilson. Their research objective was to explore relationships between an ouput variable and a set of input variables. The output variable was, for instance, the yield of a chemical process and the input variables any other variables influencing the yield.

RSM is widely used throughout research areas like chemical engineering, mathematical modeling, business modeling, air-flow dynamics, product development and improvement, etc. Its popularity stems from the ability to explore the behavior of a function and to determine what the levels of the input variables should be to obtain good function values.

3.2 Terminology and approach

In linear regression analysis the estimated linear model is referred to as a regression function. In RSM terminology it is more often called a *response function* and the dependent variable is called the *response variable.* The factors influencing the response variable are referred to as *independent variables*, *input variables* or *regressors.* Any activity or process for which response values can be observed as a result of defining input levels for it, is called a *response system.*

The RSM approach has three stages (Khuri and Cornell [KC87]):

1. Designing the experiments for data collection. This involves deciding on values to substitute for the regressors in the response system. Usually appropriate combination of levels for the regressors is chosen in order to reduce or eliminate effects like multicollinearity.

2. Estimating a response function from the experimental data. The data is the preselected values for the regressors in the previous step, together with the response values obtained from the response system. Statistical hypothesis testing can be applied to determine the significance of the estimated response function.

3. Determining the levels of the regressors which will produce maximum (or minimum) response values in the response system. Various techniques like *search methods* and *gradient seeking methods* exist to obtain these maximum (or minimum) response values.

As an illustration consider a chemical process as a response system. In this response system chemical yield is dependent on regressors like, for instance, temperature and some catalyst. The first step in RSM would then be to design an experiment. This involves deciding on what levels of the temperature and the catalyst to substitute into the response system. By substituting these levels back into the response system and measuring the response values (i.e. the yield), the next step of fitting a response function to the experimental data can be performed. The last step in RSM is to use the estimated response function together with optimization techniques in order to find the levels of the temperature and the catalyst for which the chemical yield would be a maximum (or a minimum).

The subsequent sections of this chapter are devoted to step number one and three in the list outlined above. This entails that some of the experimental designs for RSM are discussed coupled with a survey of techniques for finding maximum (or minimum) response values for the response system. The second step of estimating a response function is similar to the estimation techniques discussed in the chapter on linear regression analysis.

3.3 Experimental designs

During the discussion of linear regression analysis in the previous chapter, only regression functions with degree 0 and degree 1 were mentioned. Such functions are called first-order models and the general form of a first-order model with k input variables x_1,x_2,...,x_k is

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \varepsilon \tag{3.1}$$

In the RSM process the researcher will usually at first try to fit a first-order model to the planned data. If a lack of fit is determined by means of

hypothesis testing, a second-order model is considered. Second-order models are of a higher degree with a general form of

$$y = \beta_0 + \sum_{i=1}^{k} \beta_i x_i + \sum_{i=1}^{k} \beta_i x_i^2 + \sum_{i=1,i<j}^{k-1} \sum_{j=2}^{k} \beta_{ij} x_i x_j \tag{3.2}$$

Although second-order models are not linear in the variables x_1,x_2,...,x_k, it is still linear in the coefficients β_0,β_1,...,β_k and it is, therefore, still possible to estimate the unknown parameters β_i with methods like least squares. Different experimental designs for first-order and second-order models exist (see Khuri and Cornell [KC87], and Myers and Montgomery [MM95]).

3.3.1 Designs for first-order models

The purpose of experimental designs is to determine the appropriate combination of levels for the regressors in order to obtain more reliable estimates for the response function coefficients. Specifically, the aim is to estimate regression coefficients that are uncorrelated with minimum variance, which will reduce the effect of multicollinearity. This can be achieved by orthogonal designs. Consider the design matrix

$$D = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1k} \\ x_{21} & x_{22} & \dots & x_{2k} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nk} \end{bmatrix} \tag{3.3}$$

which represents the levels of the variables x_1,x_2,...,x_k selected to substitute back into the response system. An orthogonal design will be one where the columns of the design matrix D are orthogonal, that is

$$\sum_{u=1}^{n} x_{ui} x_{uj} = 0 \text{ for } i, j = 1, 2, \dots, k \text{ with } i \neq j \tag{3.4}$$

Various orthogonal designs exist for achieving orthogonality among the columns of the design matrix. A few of these designs will be discussed briefly.

Factorial designs

As far as factorial designs are concerned, the approach is to choose all possible combinations of levels for the regressors. For example assume that a model has two variables and the range of levels of each variable has been selected as:

$$\begin{aligned} x_1 &: 150, 200, 250 \\ x_2 &: 8, 10, 12 \end{aligned} \tag{3.5}$$

Then the design matrix is constructed by using all possible combinations of x_1 and x_2

$$\mathbf{D} = \begin{bmatrix} 150 & 8 \\ 150 & 10 \\ 150 & 12 \\ 200 & 8 \\ 200 & 10 \\ 200 & 12 \\ 250 & 8 \\ 250 & 10 \\ 250 & 12 \end{bmatrix} \tag{3.6}$$

Some of the subclasses of factorial designs are like 2^k-factorial designs and fractional replication of 2^k-factorial designs. The approach to 2^k-factorial designs is to select only a high and a low level for each regressor and to encode these two levels to either a +1 or a -1. This can be achieved by applying the transformation

$$x_{iu} = \frac{2(X_{ui} - \bar{X}_i)}{R_i} \tag{3.7}$$

where X_{ui} is the uth observation from variable i, and $\bar{X}$ is the average for the low and high settings. R_i is the range between the two settings. Now the design matrix is constructed by using all possible combinations of the levels +1 and -1. The motivation for fractional replication of the 2^k-factorial design is to construct only a subset (or fraction) of the design points of a 2^k-factorial design. This is useful when the number of possible combinations of regressor levels is very large and a cost-conscious practitioner does not want to use all 2^k factorial combinations for model fitting.

Simplex design

The simplex design is an orthogonal design with $N = k + 1$ design points where k is the number of variables in the model. The basic approach is to construct the design matrix in such a way that the design points are located at the vertices of a k-dimensional regular-sided figure (a simplex). The characteristics of this design are that the angle, θ, which any two points make with the origin is such that $\cos\theta = -1/k$.

For $k = 2$, the simplex design points are the vertices of an equilateral triangle, and for $k = 3$, the design points are the vertices of a tetrahedron (see Figure 3.1).

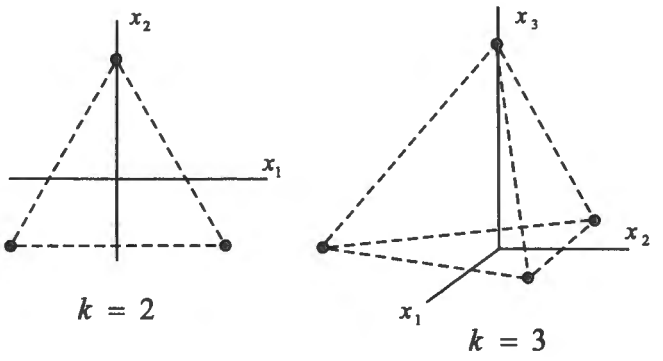

Figure 3.1: Simplex design for $k = 2$ and $k = 3$.

The design matrix for the simplex design with k variables can be written as:

$$\mathbf{D} = \begin{bmatrix} -s_1 & -s_2 & -s_3 & \dots & -s_k \\ s_1 & -s_2 & -s_3 & \dots & -s_k \\ 0 & 2s_2 & -s_3 & \dots & -s_k \\ 0 & 0 & 3s_3 & \dots & -s_k \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \dots & ks_k \end{bmatrix} \tag{3.8}$$

with $s_i = c(k+1)/[i(i+1)]^{1/2}$ and c some scaling factor.

Plackett-Burman design

The design matrix of the Plackett-Burman design is constructed in an attempt to obtain $N = k + 1$ design points, where N is a multiple of 4. These design points are in fact a subset of the design points constructed by a 2^k-factorial design. The first row of the design matrix is constructed by selecting elements equal to +1 or -1 in such a way that the number of positive elements is $(k+1)/2$ and the number of negative elements is $(k-1)/2$. The next $k-1$ rows are generated from the first row by shifting it cyclically one place $k-1$ times. This means that the second row is generated from the first row by shifting the first row one place, the third row is generated by shifting the second row one place, and so forth.

As an example consider a model with seven variables such that $N = 7+1 = 8$, which is a multiple of 4. Now the elements of the first row consists of four +1 elements and three -1 elements, for example

$$+1 \quad +1 \quad +1 \quad -1 \quad +1 \quad -1 \quad -1$$

The next six rows generated by means of shifting cyclically are then

$$\begin{array}{ccccccc} -1 & +1 & +1 & +1 & -1 & +1 & -1 \\ -1 & -1 & +1 & +1 & +1 & -1 & +1 \\ +1 & -1 & -1 & +1 & +1 & +1 & -1 \\ -1 & +1 & -1 & -1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 & -1 & +1 & +1 \\ +1 & +1 & -1 & +1 & -1 & -1 & +1 \end{array}$$

and the last row is selected as

$$+1 \quad +1 \quad +1 \quad -1 \quad +1 \quad -1 \quad -1$$

3.3.2 Designs for second-order models

3^k-factorial designs

With the exception of coding the values for the regressors to three levels, 3^k-factorial designs are similar to 2^k-factorial designs in the case of first-order models. The levels of the variables can, for instance, be coded to -1, 0 and +1, and once again all combinations for the variables at the three levels are generated.

$$\mathbf{D} = \begin{bmatrix} -1 & -1 \\ -1 & 0 \\ -1 & 1 \\ 0 & -1 \\ 0 & 0 \\ 0 & 1 \\ 1 & -1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \tag{3.9}$$

The design points of the design matrix $\mathbf{D}$ are represented in Figure 3.2.

Box-Behnken designs

This design are constructed by combining two-level factorial designs with *balanced incomplete block designs*. An example of a balanced incomplete block design occurs where four inputs and six blocks, with each block containing

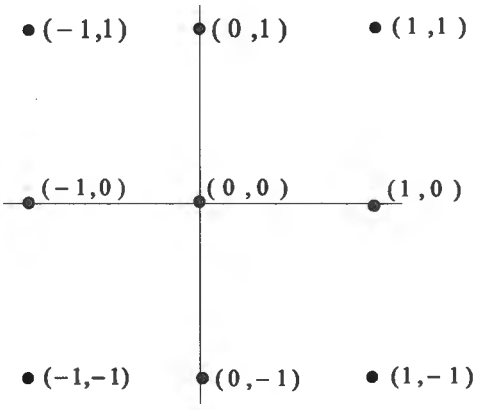

Figure 3.2: A 3^k-factorial design for $k = 2$.

two treatments, are invlolved.

$$\begin{array}{cccccc} & & x_1 & x_2 & x_3 & x_4 \\ & 1 & * & * & & \\ & 2 & & & * & * \\ \textit{Blocks} & 3 & * & & & * \\ & 4 & & * & * & \\ & 5 & & * & & * \\ & 6 & * & & * & \end{array} \tag{3.10}$$

The treatments are indicated by the asterisks and can, for instance, be composed out of an experimental design such as the 2^k-factorial design. This means that each pair of asterisks in a block can be replaced by the two columns of the 2^k-factorial design

$$\begin{array}{cc} x_i & x_j \\ -1 & -1 \\ 1 & -1 \\ -1 & 1 \\ 1 & 1 \end{array} \tag{3.11}$$

and that where no asterisk is present a column of zeros is included.

The resulting Box-Behnken design in four input variables consists of the following 27 points:

$$
\begin{array}{lcccc}
 & x_1 & x_2 & x_3 & x_4 \\
 & -1 & -1 & 0 & 0 \\
 & 1 & -1 & 0 & 0 \\
\textit{block } 1 & -1 & 1 & 0 & 0 \\
 & 1 & 1 & 0 & 0 \\
 & 0 & 0 & -1 & -1 \\
 & 0 & 0 & 1 & -1 \\
\textit{block } 2 & 0 & 0 & -1 & 1 \\
 & 0 & 0 & 1 & 1 \\
 & 0 & 0 & 0 & 0 \\
 & . & . & . & . \\
 & -1 & 0 & 0 & -1 \\
\textit{block } 3 & 1 & 0 & 0 & -1 \\
 & -1 & 0 & 0 & 1 \\
 & 1 & 0 & 0 & 1 \\
 & 0 & -1 & -1 & 0 \\
 & 0 & 1 & -1 & 0 \\
\textit{block } 4 & 0 & -1 & 1 & 0 \\
 & 0 & 1 & 1 & 0 \\
 & 0 & 0 & 0 & 0 \\
 & . & . & . & . \\
 & 0 & -1 & 0 & -1 \\
 & 0 & 1 & 0 & -1 \\
\textit{block } 5 & 0 & -1 & 0 & 1 \\
 & 0 & 1 & 0 & 1 \\
 & -1 & 0 & -1 & 0 \\
 & 1 & 0 & -1 & 0 \\
\textit{block } 6 & -1 & 0 & 1 & 0 \\
 & 1 & 0 & 1 & 0 \\
 & 0 & 0 & 0 & 0
\end{array}
\tag{3.12}
$$

Note that a row of zero elements is augmented at blocks 2, 4 and 6 in order for the 3 composite blocks, indicated by dots (block 1+2, block 3+4 and block 5+6), to be orthogonal.

Central Composite design

The *central composite design* consists of

- A complete (or fraction of a) 2^k-factorial design, where the levels of the input variables are coded to either a +1 or a -1.
- n_0 center points ($n_0 \geq 1$)
- Two axial points on the axis of each design variable at a distance of α from the design center. This portion is called the axial portion of the design.

The values of n_0 and α are chosen according to some criteria (Khuri and Cornell [KC87, pages 117–122]). For a rotatable design the value of n_0 can be any positive integer and $\alpha = M^1/4$ where M is the number of design points of the 2^k-factorial design, that is : $M = 2^K$. As an example consider a central composite design in $k = 2$ variables with $n_0 = 1$ and $\alpha = \sqrt{2}$. Then the design matrix will be of the form

$$\mathbf{D} = \begin{bmatrix} -1 & -1 \\ 1 & -1 \\ -1 & 1 \\ 1 & 1 \\ \sqrt{2} & 0 \\ -\sqrt{2} & 0 \\ 0 & \sqrt{2} \\ 0 & -\sqrt{2} \\ 0 & 0 \end{bmatrix} \tag{3.13}$$

with a total number of design points $N = M + 2k + n_0$. Figure 3.3 represents the design points graphically.

3.4 Experiments with mixtures

The experimental designs discussed previously ensure that the levels chosen for the input variables are independent of each other. In some situations such a choice of levels for the input variables may not be possible. Consider a chemical experiment where the etching of semiconductor wafers due to the application of 3 types of acid is investigated (example taken from Myers and Montgomery [MM95, page 535]). For this experiment a fixed size etching chamber is used which means that the proportion of volume taken up by

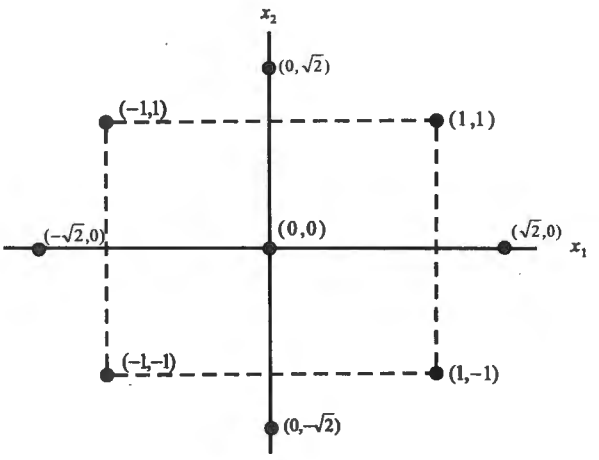

Figure 3.3: Central composite design for $k = 2$ and $\alpha = \sqrt{2}$.

the 3 types of acid will always add up to 100%. Let x_1,x_2 and x_3 be the proportions of volume occupied by each of the 3 types of acid, then the following are examples of possible blends for the experiment:

Blend 1:	$x_1 = 0.3$	$x_2 = 0.3$	$x_3 = 0.4$
Blend 2:	$x_1 = 0.2$	$x_2 = 0.5$	$x_3 = 0.3$
Blend 3:	$x_1 = 0.5$	$x_2 = 0.5$	$x_3 = 0.0$
Blend 4:	$x_1 = 1.0$	$x_2 = 0.0$	$x_3 = 0.0$

In general a requirement for a mixture experiment with n input variables will be

$$x_1 + x_2 + \ldots + x_n = 1 \tag{3.14}$$

which shows that the levels of the input variables are no longer independent. Figure 3.4 illustrates graphically the mixture constraints for $n = 3$. For three variables the feasible space for the mixture experiment is a triangle which corresponds to a simplex design. In general, the experimental region for a mixture problem with n variables will be a simplex with n vertices in $(n - 1)$ dimensions. Figure 3.5 shows the simplex coordinate system used for determining the combinations of proportions for a 3-variable mixture experiment. The constraint 3.4 imposed on mixture experiments requires a different type of experimental design, as well as a different type of polynomial for describing the model.

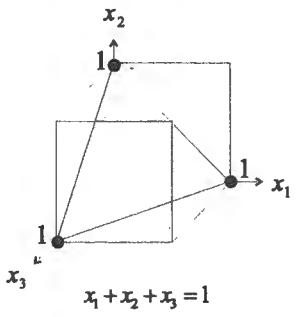

Figure 3.4: Constrained variable space for $n = 3$.

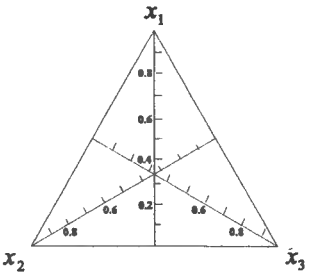

Figure 3.5: Simplex coordinate system for three components.

3.4.1 Experimental designs and models

An example of a mixture experimental design is the *simplex lattice design* which is simply a uniformly spaced set of points on a simplex. For a {n,m} simplex design the proportions taken on by each of the n number of variables, will be

$$x_i = 0, \frac{1}{m}, \frac{2}{m}, \ldots, 1 \quad i = 1, 2, \ldots, n \tag{3.15}$$

where m specifies the number of intervals on the coordinate system. Figure 3.6 shows the design for {3,2} simplex lattice design where $x_i = 0, \frac{1}{2}, 1$ for $i = 1, 2, 3$. The six points generated for the design are

$$(x_1, x_2, x_3) = (1,0,0), (0,1,0), (0,0,1), (\frac{1}{2}, \frac{1}{2}, 0), (\frac{1}{2}, 0, \frac{1}{2}), (0, \frac{1}{2}, \frac{1}{2})$$

According to Myers and Montgomery ([MM95]), the parameters $\beta_0, \beta_1, \beta_2, \ldots, \beta_n$ for the first-order model

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k n_n \tag{3.16}$$

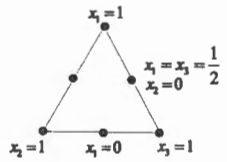

Figure 3.6: A {3,2} simplex lattice design.

are not unique when the model is fitted to a design with the constraint $x_1 + x_2 + \ldots + x_n = 1$. For this reason special models need to be estimated for mixture experiments. An approach suggested by Myers and Montgomery ([MM95]) is to multiply some of the terms in the original response surface model with the identity $x_1 + x_2 + \ldots + x_n = 1$. For example, consider the first-order model 3.16 and multiply the term β_0 with $x_1 + x_2 + \ldots + x_n = 1$. Now the first-order model becomes

$$\begin{aligned} y &= \beta_0(x_1 + x_2 + \ldots + x_n) + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n \\ &= \beta_1^* x_1 + \beta_2^* x_2 + \ldots + \beta_n^* x_n \end{aligned} \tag{3.17}$$

where $\beta_i^* = \beta_0 + \beta_i$. This formulation of the first-order model is called the *canonical form.*

Geometrically the parameter β_i^* represents the expected response to the blend $x_i = 1, x_j = 0, j \neq i$ and represents the height of the mixture surface at the vertex $x_i = 1$. Figure 3.7 shows a mixture model in the case where $n = 3$ and $\beta_1 > \beta_2 > \beta_3$.

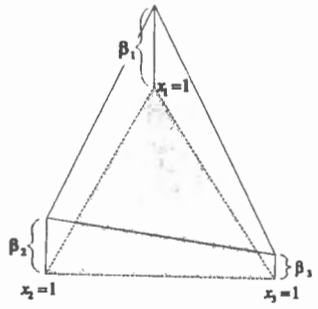

Figure 3.7: Linear mixture model with $\beta_1 > \beta_2 > \beta_3$.

3.4.2 Constraints on input variables

In many cases bounds are imposed on the input variables for mixture experiments. The bounds are of the form

$$L_i \leq x_i \leq U_i \quad i = 1, 2, \ldots, n \tag{3.18}$$

where L_i denotes the lower bound on the variable, and U_i the upper bound. The effect of bounds on input variables is that the feasible space defined by the simplex of a mixture design is reduced. To illustrate the effect of lower bounds on the input variables, consider an example with

$$0.3 \leq x_1, \quad 0.4 \leq x_2, \quad 0.1 \leq x_3$$

Figure 3.8 shows the feasible region when the lower bounds are imposed on the design. Notice that the shape of the feasible region remains simplex and consequently designs like the simplex-lattice are still applicable. In the case of upper bounds, the feasible region is not guaranteed to be a simplex.

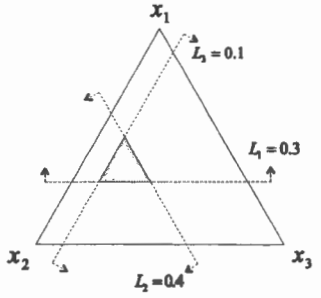

Figure 3.8: A feasible mixture space with lower bounds.

Consider the following upper bounds

$$x_1 \geq 0.7, \quad x_2 \geq 0.5, \quad x_3 \geq 0.8$$

Figure 3.9 shows the feasible region for these upper bounds and it clearly indicates that the problem is not a simplex. In cases like these computer-generated designs (see Myers and Montgomery [MM95]) are alternatives.

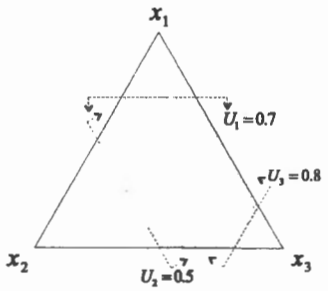

Figure 3.9: A feasible mixture space with upper bounds.

3.5 Determining optimum conditions

In Section 3.2 the stages of RSM were identified as designing an experiment, estimating a response function and determining the optimum conditions. In most cases these three stages of RSM are applied in an iterative manner in an attempt to determine the true behavior of the response function. The following is an iterative strategy proposed by Khuri and Cornell [KC87, page 150] where the three stages of RSM are incorporated.

- The simplest polynomial model (a first-order model) is fitted to experimental data and tested for adequacy of fit by means of hypothesis testing.
- If the model is found to be adequate, then the conditions for which the response will be more desirable are determined. These conditions are values for the regressors which may be located inside or outside the experimental design.
- In this new region where the optimum conditions are identified, the process of fitting a first-order model and testing it for lack of fit is repeated. If the model is found to be inadequate, for instance, nonplanarity is detected in the shape of the surface, the model is upgraded by adding cross-product terms and/or pure quadratic terms.
- If the second-order fit is found to be more adequate, then an investigation into the behaviour of the model in the experimental region can be undertaken by using contour plots. It is possible to obtain information from these plots about the region where the optimum might be.
- Finally, more experiments are designed in the possible optimum region and the optimal values are determined for the regressors which will produce the optimum response.

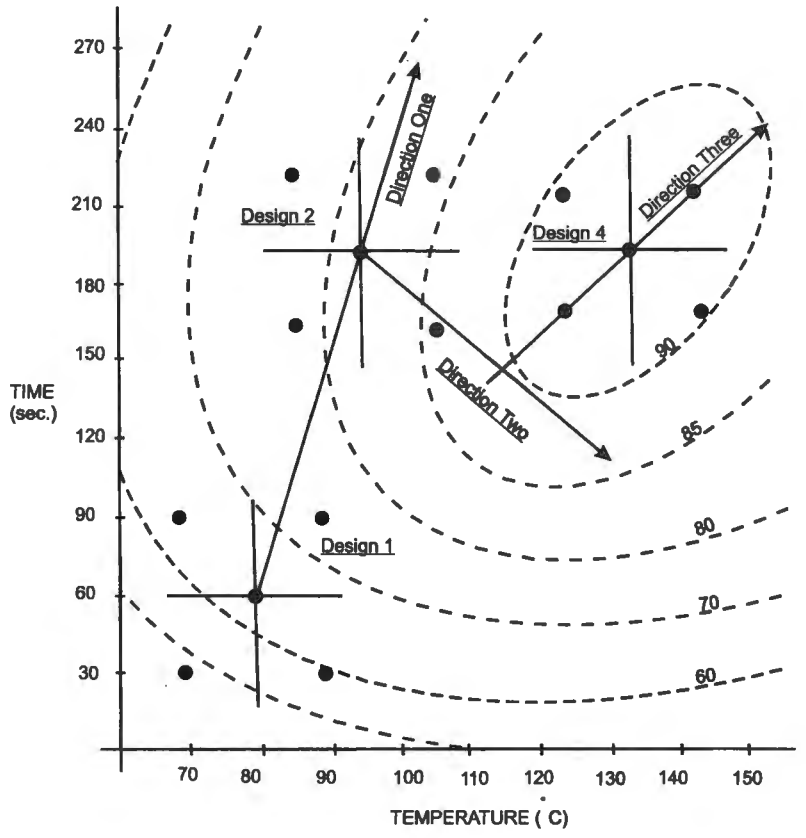

Figure 3.10: Finding the optimum path using the method of steepest ascent.

Figure 3.10 illustrates the iterative approach where the model

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon \tag{3.19}$$

was considered. The variable y represents the percentage of yield obtained in a chemical reaction with x_1 (the temperature) and x_2 (the duration of the experiment). The first step in the iterative approach is to design and conduct experiments in order to measure the percentage yield for varying values of temperature and duration time. This was done for the experimental region indicated by the first design point in the figure. A first-order model was found to be adequate and the next order of business was to determine what the values of the regressors should be to obtain the optimal percentage yield. The method of *steepest ascent* suggests that this is done by moving away from the center of the design in a direction that will result in the highest increase of percentage yield. The first direction indicated in the figure is therefore the path of steepest ascent. Values for the independent variables, found at

regular intervals along the path of steepest ascent, are used in experimental runs and the response measured. If no increase in response value is observed while moving along the path, then the next round of designing experiments, fitting of a model and finding the optimum path has been reached. Therefore, the process was repeated at the second design point and the second direction gave way to the next path of steepest ascent. At the fourth experimental design point, a second-order model was found to be more adequate (note that the third experimental design point is not shown in Figure 3.10 to improve readability). Additional experiments were performed in this region and the optimum conditions for temperature and duration time were determined by doing local exploration of the fitted second-order surface.

3.5.1 The method of steepest ascent

During the iterative approach the method of steepest ascent is applied to a first-order model

$$y = \beta_0 + \sum_{i=1}^{k} \beta_i x_i \tag{3.20}$$

which is fitted to measurements from a first-order designed experiment. The objective of the method is to find values for x_1,x_2,...,x_k which will maximize

$$\hat{y} = \hat{\beta}_0 + \sum_{i=1}^{k} \hat{\beta}_i x_i \tag{3.21}$$

subject to the constraint

$$\sum_{i=1}^{k} x_i^2 = r^2 \tag{3.22}$$

In other words, find for all fixed points that are r units away from the center of the experiment design, the point that will maximize $\hat{y}$.

The maximization is performed by using the *Lagrange multiplier* μ and the response function becomes

$$Q(x_1, x_2, \ldots, x_k) = \hat{\beta}_0 + \sum_{i=1}^{k} \hat{\beta}_i x_i - \mu(\sum_{i=1}^{k} x_i^2 - r^2) \tag{3.23}$$

To obtain the maximizing values for the variables x_1,x_2,...,x_k in 3.23, the partial derivatives are set to zero.

$$\frac{\partial Q(x)}{\partial x_i} = \hat{\beta}_i - 2\mu x_i = 0 \text{ for } i = 1, 2, \ldots, k \tag{3.24}$$

and

$$\frac{\partial Q(x)}{\partial \mu} = -\sum_{i=1}^{k} x_i^2 + r^2 = 0 \tag{3.25}$$

The solutions to 3.24 and 3.25 are the values of x_i, where

$$x_i = \frac{\hat{\beta}_i}{2\mu} \text{for } i = 1, 2, \ldots, k \tag{3.26}$$

and the value of μ is yet to be determined.

Consider some variable X_j with it's regression coefficient positive. If the value of X_j changes by an amount, say $\triangle_j$, the response value will also increase. The amount of change for the coded variable x_j is $\triangle_j / s_i$ where s_i is the scale factor considered in the transformation $x_j = (X_j - \bar{X}_j)/s_j)$. This transformation corresponds to the coding convention of the first-order design (see 3.7). Substituting the value of $\triangle_j / s_j$ for x_j in 3.26, the value of μ becomes $\mu = \hat{\beta}_j / 2x_j = \hat{\beta}_j s_j / 2\triangle_j$. The values of the remaining variables can be calculated by substituting the value for μ in 3.26 and the first point on the path of steepest ascent, x_1^1,x_2^1,...,x_k^1 is determined. The remaining points depicting the path of steepest ascent are determined by calculating μ for multiples of $\triangle_j$.

3.5.2 Steepest ascent subject to a linear constraint

The method of steepest ascent involves conducting experiments along the optimal path. In some situations the optimal path may require for some of the variables to take on values that are impractical. For instance, it is possible that an ingredient concentration may exceed its limit. As a result of this it becomes necessary to impose a constraint on the design variables. This entails that the optimal path explored by the steepest ascent method becomes bounded.

Figure 3.11 shows how the direction of the optimal path is altered when the steepest ascent method reaches the constraint

$$c_0 + c_1 x_1 + c_2 x_2 = 0 \tag{3.27}$$

at the point **O**. Experiments are conducted further along the modified optimal path.

Consider the first point on the path of steepest ascent (3.26)

$$x_i = \rho \hat{\beta}_i \text{ for } i = 1, 2, \ldots, k \tag{3.28}$$

where $\rho = 1/2\mu$. Let the modified path be the direction vector

$$\hat{\beta}_i - dc_i \text{ for } i = 1, 2, \ldots, k \tag{3.29}$$

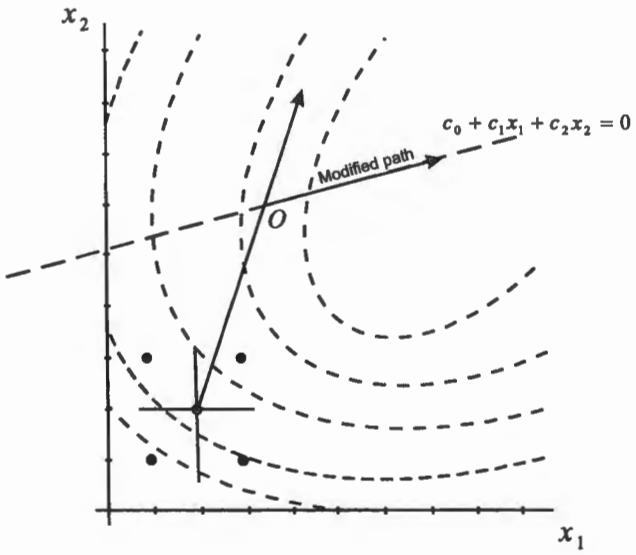

Figure 3.11: Steepest ascent with a linear constraint.

where the c_i are the coefficients of the constraint imposed. The modified path will start at the point **O** and, therefore, proceed along the direction $\hat{\beta}_1 - dc_1, \hat{\beta}_2 - dc_2, \ldots, \hat{\beta}_k - dc_k$.

The value of d is chosen in such a manner that

$$\sum_{i=1}^{k}(\hat{\beta}_i - dc_i)^2 \tag{3.30}$$

is minimized. This will ensure that the modified path is taken so as to be "closest" to the original path. The minimization of 3.30 is a simple regression situation in which $\hat{\beta}_i$ is being regressed against the c_i. The value d plays the role of the slope of the regression and, consequently, can be obtained by the minimization of the "residual" sum of squares

$$d = \frac{\sum_{i=1}^{k}\hat{\beta}_i c_i}{\sum_{i=1}^{k} c_i^2} \tag{3.31}$$

To determine the point **O**, consider it to lie on both the original optimal path and the constraint plane. Therefore, the first point on the modified path (3.26) needs to satisfy the constraint

$$c_0 + c_1x_1 + c_2x_2 + \ldots + c_kx_k = 0 \tag{3.32}$$

By substituting 3.26 into 3.32 we obtain

$$c_0 + (c_1\hat{\beta}_1 + c_2\hat{\beta}_2 +, \ldots, c_k\hat{\beta}_k)\rho_0 = 0 \tag{3.33}$$

As a result, the modified path starts at $x_j, 0 = \rho_0 \hat{\beta}_j$ for $j = 1, 2, \ldots, k$ where

$$\rho_0 = \frac{-c_0}{\sum_{i=1}^{k} c_i \hat{\beta}_i} \tag{3.34}$$

3.5.3 Exploration of a fitted second-order surface

It is suggested that a second-order model is fitted during the iterative approach in the case where the first-order model is found to be inadequate and when curvature is observed in the region of experimentation. With the second-order model fitted, graphic methods like contour plots can be used to explore the surface and to determine the optimum conditions for the regressors. The optimum conditions will be revealed by a stationary point on the surface. This is a point at which the slope of the surface is zero when taken in all directions. In the case where a second-order model has more than two independent variables, contour plots will become insufficient (graphic representations are restricted to three dimensions) and the stationary point needs to be determined mathematically.

Consider the second-order model 3.2 in matrix notation:

$$y = \beta_0 + \mathbf{x}^T\mathbf{b} + \mathbf{x}^T\mathbf{B}\mathbf{x}^T \tag{3.35}$$

where

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_k \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} b_{11} & \frac{b_{12}}{2} & \cdots & \frac{b_{1k}}{2} \\ & b_{22} & \cdots & \frac{b_{2k}}{2} \\ & & \ddots & \vdots \\ & & & \frac{b_{k-1,k}}{2} \\ \text{symmetric} & & & b_{kk} \end{bmatrix}$$

The partial derivatives of y with respect to $x_1, x_2, \ldots, x_k$ are

$$\begin{aligned} \frac{\partial y}{\partial x_1} &= b_1 + 2b_{11}x_1 + \sum_{j=2}^{k} b_{1j}x_j \\ \frac{\partial y}{\partial x_2} &= b_2 + 2b_{22}x_2 + \sum_{j\neq 2}^{k} b_{2j}x_j \\ &\vdots \\ \frac{\partial y}{\partial x_k} &= b_k + 2b_{kk}x_k + \sum_{j=1}^{k-1} b_{kj}x_j \end{aligned} \qquad = \mathbf{b} + 2\mathbf{B}\mathbf{x}$$

In order to obtain the stationary point on the fitted second-order surface, the partial derivatives are set equal to zero and solved for the values of x_i. The coordinates of the stationary point are then given by the $k \times 1$ vector

$$\mathbf{x}_0 = -\frac{\mathbf{B}^{-1}\mathbf{b}}{2} \tag{3.36}$$

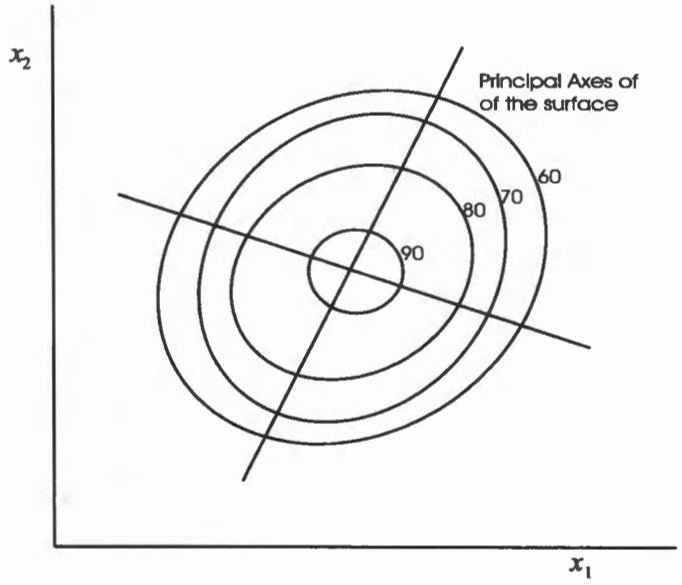

Figure 3.12: Second-order system with point of maximum response.

This point may represent the location at which the surface attains a maximum or minimum as shown in Figure 3.12. A stationary point may also represent a minimax or saddle point as illustrated in Figure 3.13. Information obtained from solving conical equations of the fitted second-order model can be used to differentiate between the different types of stationary points (see Khuri and Cornell [KC87]).

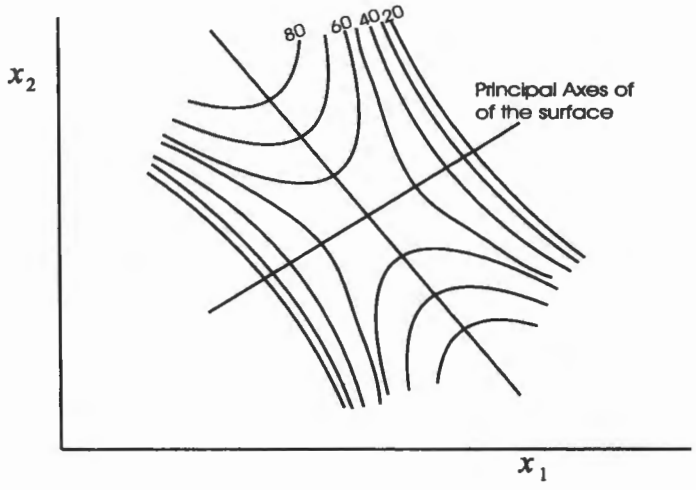

Figure 3.13: Second-order system with saddle point.

3.6 Response surface methodology and raw data

In many research fields the application of RSM may not be feasible. This is usually the case where raw data is collected by observation (e.g. questionnaires for social studies) and not from a designed experiment where the values for the dependent variable y are measured by specifying values for the independent variables x_1,x_2,. . .,x_k.

The inability to preselect values for the independent variables in the case of raw data implies that the absence of multicollinearity can not be guaranteed. Furthermore, the process of collecting raw data is usually expensive. An iterative optimum seeking method where experiments are performed along the optimal path may, therefore, be economically infeasible. Consider Example 2.1 used in Section 2.5, where the linear model

$$\hat{y} = -144.6 + 5.71x_1 + 2.95x_2 \tag{3.37}$$

was investigated for the possibility of an irrigation scheme. The variables identified to influence the production (y) of the region are rainfall (x_1) and temperature (x_2). There is no possibility of a designed experiment for such an investigation (after all, temperature cannot be controlled in an farming situation). Consequently an iterative optimum seeking approach cannot be followed (the approach requires experiments conducted along the path of steepest ascent/descent). By creating a contour plot of the model it may be possible to observe the behaviour of the response value y for values of x_1

and x_2 substituted into 3.37. The nature of a first-order linear model will obviously always be monotonically increasing/decreasing as shown by the contour plot in Figure 3.14

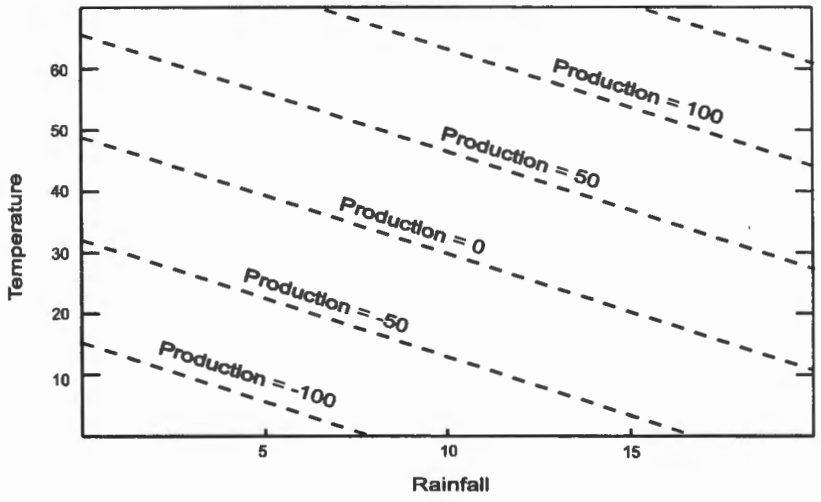

Figure 3.14: Two-dimensional contours for Example 2.1.

The contour plot also reveals that high production values may be expected where high levels of rainfall and high levels of temperature occur (since both coefficients for x_1 and x_2 are positive). If it was found that a second-order model is more appropriate for the data, a contour plot might have displayed more curvature or even a local maximum or minimum location. In this respect RSM can be applied in some way to raw data by using contour plots.

It was, however, shown in Section 2.5 that the variables x_1 and x_2 for this example are intercorrelated. Therefore the argument presented in Section 2.5 that high levels of rainfall may not coincide with high levels of temperature, still applies. In this respect the contour plot in Figure 3.14 fails in revealing the estimated response by taking into account the area of experience.

3.7 Summary

RSM provides techniques for investigating the behaviour of a response function and finding the optimum conditions for the regressors. The process involves designing experiments to ensure that the input values for the regressor are chosen to be orthogonal. This is necessary to eliminate the effect of multicollinearity. Finding the optimum conditions for the regressors requires an iterative approach where experiments are conducted along the path of

steepest ascent. In the vicinity of the optimum second-order models are used to describe the response function and to find stationary points.

In many research situations where raw data are collected by observation, RSM may not be feasible. The data are not guaranteed to be free of inter-correlations and an iterative approach of conducting experiments may not be possible (e.g economic constraints). In the next chapter *linear response surface analysis* (LRSA) is presented. LRSA is concerned with the investigation and optimization of the response function as obtained by fitting the function to raw data.

Chapter 4

A mathematical programming approach to response surface analysis

4.1 Introduction

Example 2.1 (investigating the feasibility of an irrigation scheme) was used in both sections 2.5 (linear regression analysis) and 3.6 (response surface methodology) to illustrate the difficulty of optimization when multicollinearity is present. The main argument was that, although both the regression coefficients for the variables Rainfall and Temperature (the variables influencing the dependent variable Production) are positive, there are no cases in the data set for which there are high levels of rainfall as well as high levels of temperature (see Figure 2.1). In the article "Constrained regression models for optimization and forecasting", Bruwer and Hattingh ([BH85]) suggested that only the area of experience should be considered for optimization of linear models. In Example 2.1 such an area of experience for the variables Rainfall and Temperature is the convex hull of data points given in Figure 4.1.

According to Bruwer and Hattingh the linear model can be optimized with the convex hull as a constraint. The field of *mathematical programming* includes several areas of linear optimization where the solution must conform to certain constraints. Bruwer and Hattingh applied the technique of *linear programming* which is a subset of the field of mathematical programming, to optimize a linear model constrained to the convex hull of the data points.

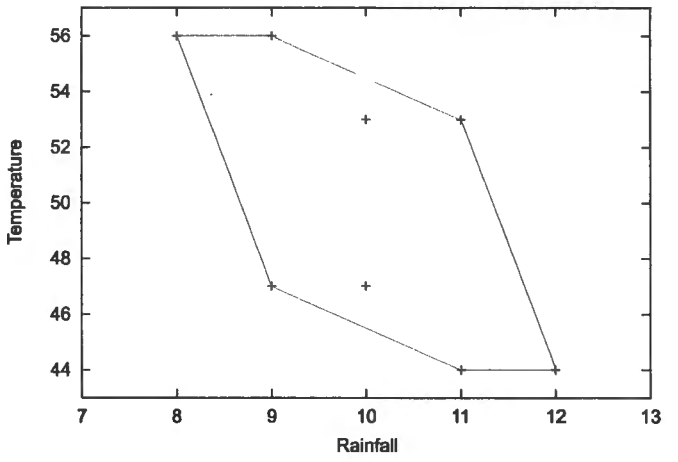

Figure 4.1: Convex hull for data points in Example 2.1.

4.2 Linear programming

Linear programming is concerned with finding the optimum conditions for a linear function while satisfying a set of linear equality and/or inequality constraints. The formulation of a *linear program* is

$$\begin{array}{lrrrrrr} \text{Minimize} & c_1x_1+ & c_2x_2+ & \ldots+ & c_nx_n & & \\ \text{Subject to} & a_{11}x_1+ & a_{12}x_2+ & \ldots+ & a_{1n}x_n & \leq & b_1 \\ & a_{21}x_1+ & a_{22}x_2+ & \ldots+ & a_{2n}x_n & \leq & b_2 \\ & \vdots & \vdots & & \vdots & \vdots & \\ & a_{m1}x_1+ & a_{m2}x_2+ & \ldots+ & a_{mn}x_n & \leq & b_m \\ & x_1, & x_2, & \ldots & x_n & \geq & 0 \end{array} \tag{4.1}$$

where $c_1x_1 + c_2x_2 + \ldots + c_nx_n$ is the objective function to be minimized. The coefficients $c_1, \ldots, c_2, c_n$ are the (known) cost coefficients and $x_1, x_2, \ldots, x_n$ are the (unknown) decision variables. The inequality $\sum_{j=1}^{n} a_{ij}x_j \leq b_i$ denotes the ith constraint and column vector $b_1, b_2, \ldots, b_n$ is known as the right-hand-side vector. The constraint $x_1, x_2, \ldots, x_n \geq 0$ is known as the non-negativity constraint which prevents the decision variables from taking on negative values. A set of values for the variables $x_1, x_2, \ldots, x_n$ satisfying all the constraints is called a *feasible solution.*

For ease of illustration in subsequent sections the linear program (4.1) will now be formulated in matrix notation. Denote the row vector $(c_1, c_2, \ldots, c_n)$ by $\mathbf{c}$, and consider the following column vectors $\mathbf{x}$ and $\mathbf{b}$ and the $m \times n$ matrix $\mathbf{A}$.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}, \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

The problem can now be written as

$$\begin{array}{rlcl} \text{Min} & \mathbf{cx} & & \\ \text{Subject to} & \mathbf{Ax} & \leq & \mathbf{b} \\ & \mathbf{x} & \geq & 0 \end{array} \tag{4.2}$$

The goal of linear programming is to find a vector $\mathbf{x}$ among all feasible vectors, which minimizes the objective function.

EXAMPLE 4.1

$$\begin{array}{rlcl} \text{Min} & -x_1 - 3x_2 & & \\ \text{Subject to} & 2x_1 + 3x_2 & \leq & 6 \\ & -x_1 + x_2 & \leq & 1 \\ & x_1, x_2 & \geq & 0 \end{array}$$

For this example there are two decision variables, x_1 and x_2, and an objective function $Z = x_1 + 3x_2$ that needs to be maximized (note that minimize Z = maximize -Z). The constraints for the problem define a feasible region which is the shaded area in Figure 4.2.

To solve the linear program, values for the decision variables need to be selected which fall in the feasible region and also give the maximum function value when substituted into the objective function. This corresponds to moving the objective function parallel to itself (dotted contour lines) from the origin until it reaches the maximum point within the feasible region. In Figure 4.2 this point is $[\frac{3}{5}, \frac{8}{5}]^T$. If the objective function is to be moved further beyond the optimum point then the decision variables take on values which will violate the constraints. Note that in general the optimum point is one of the corner points of the feasible region that are called *extreme points*.

Depending on the feasible region defined by the constraints different solution scenarios exist. One unique solution was found for Example 4.1. In some other cases it is possible that alternative solutions can be found. This will be the case when the objective function is moved away from the origin up to a constraint where the constraint is parallel to the objective function. Any point that falls along the line segment defined by the constraint is then

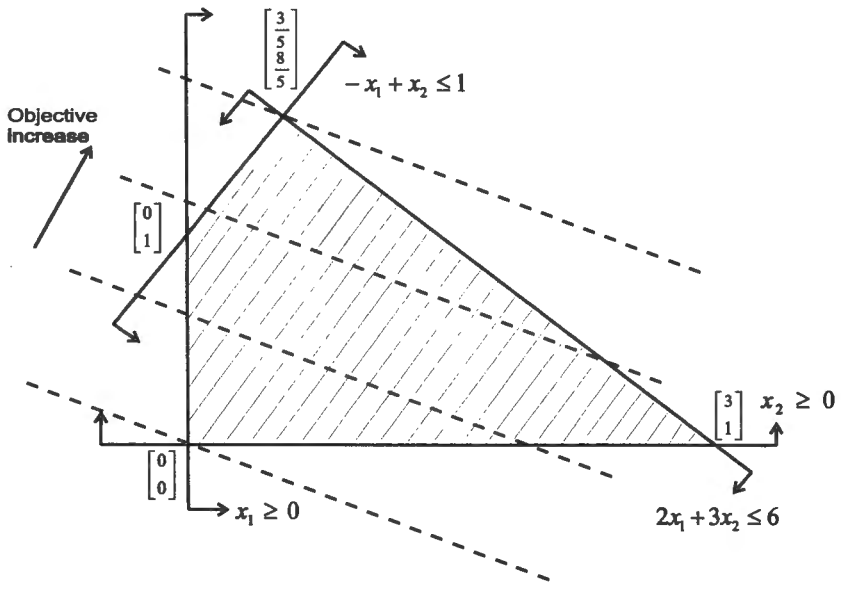

Figure 4.2: Feasible region for Example 4.1.

considered to be optimal. It is also possible that there is no extreme point or line segment limiting the objective function when moving away from the origin. This means that the objective function can be increased indefinitely. This solution scenario is known as *unbounded solutions.* In all of the above solution scenarios a non-empty feasible region was assumed. If this is not the case and the constraints are defined in such a way that no feasible solution exists, the problem is said to be *infeasible.*

4.2.1 Geometric properties of the feasible region

The graphical method explained in the previous example to obtain the optimal solutions vector for a linear program can only be applied to problems with two or maybe three dimension. For higher dimensional linear programs an algebraic method called the *simplex* method is used. Before discussing the simplex method, some definitions and theorems will be presented to reveal the geometric properties of the feasible region of a linear program. Consider the following definitions.

- A set of points S is a *convex set* if and only if $\alpha\mathbf{x} + (1-\alpha)\mathbf{y} \in S$ for all $\mathbf{x} \in S$,$\mathbf{y} \in S$ and for all α with $0 \leq \alpha \leq 1$.
- A convex set C is a *convex hull* of a set of points S if C is the smallest convex set containing S. A more explicit characterization of convex

hulls is given by Chvatal ([Chv83]): Let S be an arbitrary set of points $z_1, z_2, \ldots, z_k$ and C the convex hull of S. Then all the points $\mathbf{z} \in C$ can be written as $\mathbf{z} = \sum_{i=1}^{k} \alpha_i \mathbf{z}_i$ with $\alpha_i \geq 0$ for $i = 1, 2, \ldots, k$ and $\sum_{i=1}^{k} \alpha_i = 1$.

- A point $\mathbf{x}$ in the convex set S is said to be an *extreme point* of S if it cannot be expressed as a convex combination of two other distinct points in S. That is, $\mathbf{x}$ is an extreme point of S if there do not exist $\mathbf{x}' \in S$, and $\mathbf{x}'' \in S$ $(\mathbf{x}' \neq \mathbf{x}'')$ such that $\mathbf{x} = \alpha\mathbf{x}' + (1 - \alpha)\mathbf{x}''$ for some α with $0 \leq \alpha \leq 1$.

We state some well known linear programming theorems:

THEOREM 4.1 *The feasible region of a linear programming problem is a convex set.*

THEOREM 4.2 *If the feasible region of a linear programming problem is bounded, then at least one extreme point is optimal.*

THEOREM 4.3 *The feasible region of a linear programming problem has a finite number of extreme points.*

4.2.2 The simplex method

In the previous section a graphical approach to solve a linear program was proposed. This corresponds to finding the extreme point for which the objective function is a minimum (or maximum). An algebraic approach could be to enumerate all the extreme points and to find the one that minimizes (or maximizes) the objective function. This approach is acceptable for smaller linear programs. However, as the number of variables and constraints increase, the number of extreme points to be evaluated can become enormous. The key to the simplex method is to evaluate only a subset of extreme points. These extreme points are adjacent to one another and the simplex algorithm moves from one extreme point to the other until it reaches the optimal extreme point.

In order to apply the simplex algorithm information regarding the extreme points needs to be extracted from the algebraic formulation of a linear program (4.2). This is achieved on the basis of the following definitions and theorems.

Fundamental definitions and theorems

- Assume the rank $(\mathbf{A})$=m. Then a nonsingular submatrix $\mathbf{B}$ from $\mathbf{A}$, formed by any m linearly independent columns from $\mathbf{A}$, is a *basis.* After rearranging the columns of $\mathbf{A}$, let $\mathbf{A} = [\mathbf{B}, \mathbf{N}]$ and $\mathbf{x} = (\mathbf{x_B}, \mathbf{X_N})$, then $\mathbf{Ax} = \mathbf{b}$ becomes $\mathbf{Bx_B} + \mathbf{Nx_N} = \mathbf{b}$. $\mathbf{x_B}$ are the m basic variables corresponding to the columns of $\mathbf{B}$ and $\mathbf{x_N}$ are the $n - m$ nonbasic variables corresponding the to remaining $n - m$ columns in $\mathbf{N}$. The solution found by setting the variables $\mathbf{x_N} = \mathbf{0}$ and $\mathbf{x_B} = \mathbf{B^{-1}b}$ is a *basic feasible solution.*

- Corresponding to the system $\mathbf{Ax} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}$, a *homogeneous solution* is a vector $\mathbf{y}$ satisfying $\mathbf{Ay} = \mathbf{0}$ and $\mathbf{y} \geq \mathbf{0}$.

- A homogeneous solution is called an *extreme homogeneous solution* if and only if it is a basic feasible solution to the system

$$\mathbf{Ay} = \mathbf{0}, \mathbf{y} \geq \mathbf{0}$$

and

$$\sum_{j=1}^{n} y_j = 1$$

- Let S be the set of feasible solutions to the linear program 4.2. Suppose $\bar{\mathbf{x}}$ is a basic feasible solution and $\bar{\mathbf{y}}$ is an extreme homogeneous solution of the constraint matrix in 4.2. Then every point on the halfline $\{\mathbf{x} : \bar{\mathbf{x}} + \theta\bar{\mathbf{y}}, \theta \geq 0\}$ is a feasible solution. This halfline is an *extreme halfline* of S if every point $\hat{\mathbf{x}}$ on it can be represented as $\hat{\mathbf{x}} = \alpha\mathbf{x}' + (1 - \alpha)\mathbf{x}''$, for some $0 < \alpha < 1$ and $\mathbf{x}' \in S$, $\mathbf{x}'' \in S$, where $\mathbf{x}'$ and $\mathbf{x}''$ both lie on the half line.

THEOREM 4.4 *The set S of feasible solutions to the constraint matrix of 4.2 is a convex set. A point $\bar{\mathbf{x}} \in S$ is an extreme point of S if and only if it is a basic feasible solution to the system of equations.*

THEOREM 4.5 *If the system in 4.2 has a feasible solution, then it has a basic feasible solution.*

THEOREM 4.6 *If the linear program 4.2 has an optimal solution, then it has an optimal basic feasible solution.*

THEOREM 4.7 *Let S be the set of feasible solutions to the constraint matrix in 4.2. Then any feasible solution* $\mathbf{x}$ *can be expressed as either*

1. *a convex combination of basic feasible solutions (if S is bounded), or*
2. *the sum of a convex combination of basic feasible solutions and a non-negative linear combination of extreme homogeneous solutions (if S is not bounded).*

Geometric motivation

Consider example 4.1. A linear programming problem is often written in the standard form

$$\begin{array}{rrcl} \text{Min} & \mathbf{cx} & & \\ \text{Subject to} & \mathbf{Ax} & = & \mathbf{b} \\ & \mathbf{x} & \geq & 0 \end{array} \tag{4.3}$$

The inequality constraints in Example 4.1 are changed to equality constraints by adding slack variables (x_3 and x_4) to the constraints. The cost coefficients for the slack variables are set to zero and the formulation for Example 4.1 becomes

$$\begin{array}{rrrrrrrrrrr} \text{Min} & -x_1 & - & 3x_2 & & & & & & & \\ \text{Subject to} & 2x_1 & + & 3x_2 & + & x_3 & & & = & 6 \\ & -x_1 & + & x_2 & & & + & x_4 & = & 1 \\ & x_1, & & x_2, & & x_3, & & x_4 & \geq & 0 \end{array}$$

Figure 4.3 represents the feasible region of the linear programming problem in the (x_1,x_2) space.

A possible solution to this linear program is to set the variables x_1 and x_2 to zero. In Figure 4.3 this solution corresponds in the (x_1,x_2) space to the point $[0,0]^T$, which in turn is an extreme point for the feasible region. From the fundamental theorems of linear programming we know that an extreme point corresponds to a basic feasible solution. Therefore, the solution $x_1 = x_2 = 0$, $x_3 = 6$ and $x_4 = 1$ is a basic feasible solution for which the objective value is $z = 0$. We have x_1 and x_2 as non-basic variables, and the variables x_3 and x_4 as the basic variables.

Another possible (and even better) solution could be one of the other extreme points of the feasible region. Consequently we need to move from the current basic feasible solution to another basic feasible solution that will result in a better objective value. This is achieved by raising the level of one of the non-basic variables, say x_2, and lowering the level of one of the basic variables. The basic variable which is decreased should not become

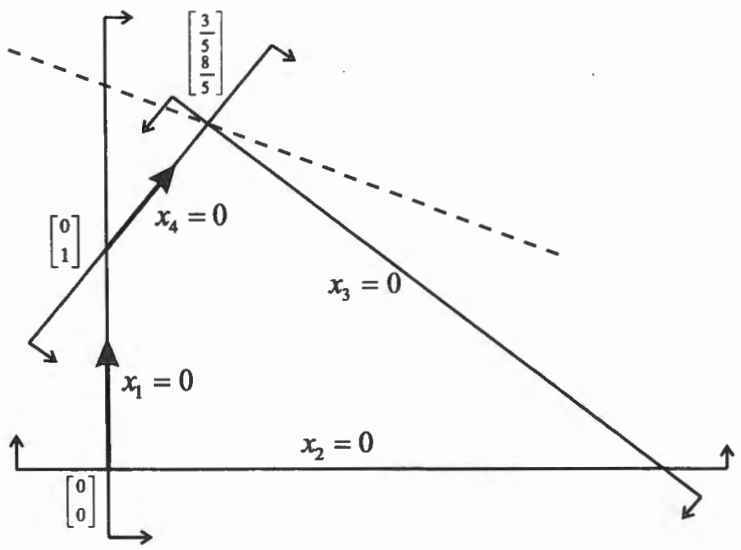

Figure 4.3: Simplex path for finding the optimal solution.

negative. In geometric terms this corresponds to moving from the point $[0,0]^T$ in Figure 4.3, along the arrow (effectively increasing x_2) up to the point $[0,1]^T$. The reason for not moving beyond this point is to obey non-negative constraints. If $[0,1]^T$ is substituted into the constraint, $-x_1+x_2+x_4 = 1$, x_4 takes on the value of zero. Clearly, if x_2 were to be increased further, x_4 will violate the non-negative constraint. Therefore, a new basic feasible solution is found: $x_1 = x_4 = 0$, $x_2 = 1$ and $x_3 = 3$, with a new (and better) objective value of $z = 3$. Now the new non-basic variables are x_1 and x_4, and the new basic variables are x_2 and x_3. In simplex terminology, x_2 is the *entering variable* (entering the basis), and x_4 is the *leaving variable*. The constraint $-x_1 + x_2 + x_4 = 1$ is called the *blocking hyperplane* and it is labelled $x_4 = 0$ in Figure 4.3.

The above procedure is repeated to move from the extreme point $[0,1]^T$ to $[\frac{3}{5}, \frac{8}{5}]^T$ where the optimal value of $z = \frac{27}{5}$ is found. Note that the stop criterium for the simplex algorithm is achieved when no increase in objective value is possible when moving to another adjacent extreme point.

Algebra of the Simplex Algorithm

Consider the linear programming problem in standard form (4.3). The standard form can be written in terms of basic and non-basic variables

$$\begin{array}{rlllll} \text{Min} & \mathbf{c}_B\mathbf{x}_B & + & \mathbf{c}_N\mathbf{x}_N & & \\ \text{Subject to} & \mathbf{B}\mathbf{x}_B & + & \mathbf{N}\mathbf{x}_N & = & \mathbf{b} \\ & \mathbf{x}_B & , & \mathbf{x}_N & \geq & 0 \end{array} \tag{4.4}$$

where $\mathbf{A} = [\mathbf{B}, \mathbf{N}]$ is an $m \times n$ matrix with rank m. The basic feasible solution to this problem is

$$\begin{aligned} \mathbf{x}_B &= \mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_N & (4.5) \\ &= \mathbf{B}^{-1}\mathbf{b} - \sum_{j \in R} \mathbf{B}^{-1}\mathbf{a}_j x_j \\ &= \bar{\mathbf{b}} - \sum_{j \in R} \mathbf{y}_j x_j & (4.6) \end{aligned}$$

where R is the set of the indices of the current non-basic variables. The objective function for the linear programming problem 4.4 is given by $z = \mathbf{c}_B\mathbf{x}_B + \mathbf{c}_N\mathbf{x}_N$. Substituting the result 4.6 into z, we obtain

$$\begin{aligned} z &= \mathbf{c}_B\left(\mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_N\right) + \mathbf{c}_N\mathbf{x}_N & (4.7) \\ &= \mathbf{c}_B\left(\mathbf{B}^{-1}\mathbf{b} - \sum_{j \in R} \mathbf{B}^{-1}\mathbf{a}_j x_j\right) + \sum_{j \in R} c_j x_j \\ &= z_0 - \sum_{j \in R} (z_j - c_j) x_j \end{aligned}$$

where $z_0 = \mathbf{c}_B\mathbf{B}^{-1}\mathbf{b}$ and $z_j = \mathbf{c}_B\mathbf{B}^{-1}\mathbf{a}_j$. The linear programming problem can now be rewritten as

$$\begin{array}{ll} \text{Min} & z = z_0 - \sum_{j \in R} (z_j - c_j) x_j \\ \text{Subject to} & \sum_{j \in R} \mathbf{y}_j x_j + \mathbf{x}_B = \bar{\mathbf{b}} \\ & x_j \geq 0, j \in R, \text{ and } \mathbf{x}_B \geq 0 \end{array} \tag{4.8}$$

The value $z_0 = \mathbf{c}_B\mathbf{B}^{-1}\mathbf{b}$ denotes the objective value of the current basic feasible solution. The values $(z_j - c_j)$, called the *reduced cost coefficient*, are an indication of optimality:

> If $(z_j - c_j) \leq 0$ for all $j \in R$, then the current basic feasible solution is optimal

If $(z_k - c_k) > 0$, the solution can be improved by increasing x_k. Suppose the

variable x_k is selected to enter the basis. Then analogue to the geometric illustration of the simplex method where one moves along the axis of x_k to the adjacent extreme point, the value x_k is increased while maintaining $x_j = 0$ for $j \in R - \{k\}$. From 4.8 we obtain

$$z = z_0 - (z_k - c_k)x_k \tag{4.9}$$

and

$$\begin{bmatrix} x_{B_1} \\ x_{B_2} \\ \vdots \\ x_{B_r} \\ \vdots \\ x_{B_m} \end{bmatrix} = \begin{bmatrix} \bar{b}_1 \\ \bar{b}_2 \\ \vdots \\ \bar{b}_r \\ \vdots \\ \bar{b}_m \end{bmatrix} - \begin{bmatrix} y_{1k} \\ y_{2k} \\ \vdots \\ y_{rk} \\ \vdots \\ y_{mk} \end{bmatrix} x_k \tag{4.10}$$

If the value of x_k increases, then x_{B_i} will increase as long as $y_{ik} \leq 0$. If $y_{ik} > 0$, x_{B_i} will decrease. In order to obey the nonnegativity constraints, x_k is increased until a basic variable x_{B_r} drops to zero for the first time. This corresponds to the geometric illustration where movement along the axis of x_k is blocked by a hyperplane . From equation 4.10 the first basic variable to drop to zero corresponds to the minimum of $\bar{b}_i/y_{ik}$ for y_{ik} positive. That is, x_k can be increased until

$$x_k = \frac{\bar{b}_r}{y_{rk}} \equiv \min_{1 \leq i \leq m} \left(\frac{\bar{b}_i}{y_{ik}} : y_{ik} > 0 \right) \tag{4.11}$$

A new feasible solution is obtained as x_k increases from a level zero to $\bar{b}_r/y_{rk}$. Substituting $x_k = \bar{b}_r/y_{rk}$ in equation 4.10 gives the new basic feasible solution

$$\begin{aligned} x_{B_i} &= \bar{b}_i - \frac{y_{ik}}{y_{rk}}\bar{b}_r \; i = 1, 2, \ldots, m \; i \neq r \\ x_k &= \frac{\bar{b}_r}{y_{rk}} \end{aligned} \tag{4.12}$$

and all the other x_j's zero.

With this new basic feasible solution, the reduced cost coefficients are recalculated and the procedure of checking for optimality, selecting the new entering and leaving variable, and updating the basis, is repeated for the next iteration.

Summary of the simplex algorithm (minimization problem)

1. Choose a starting basic feasible solution with basis $\mathbf{B}$. If such a basis is not readily available, use techniques like the *big M* method, or the *simplex two phase* method (see Bazaraa, Jarvis, and Sherali [BJS90], Chvatal [Chv83], and Dantzig [Dan63]).

2. Calculate current basic solution $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b} = \bar{\mathbf{b}}$. Let $\mathbf{x}_B = \bar{\mathbf{b}}$, $\mathbf{x}_N = \mathbf{0}$ and $z = \mathbf{c}_B\mathbf{x}_B$.

3. Calculate reduced cost coefficients $z_j - c_j = \mathbf{c}_B\mathbf{B}^{-1}\mathbf{a}_j - c_j$. Choose a k for which
$$z_k - c_k = \max_{j \in R}(z_j - c_j)$$
where R is the set of indices associated with the current non-basic variables. If $z_k - c_k \leq 0$, then the current basic feasible solution is optimal and the process is terminated. Otherwise, proceed to step 4 with x_k as the entering variable.

4. Calculate the value $\mathbf{y}_k = \mathbf{B}^{-1}\mathbf{a}_k$. If $\mathbf{y}_k \leq \mathbf{0}$, then the optimal solution is unbounded and the process is terminated. Otherwise, go to step 5.

5. Let x_k be the entering variable. The index r of the leaving variable x_{B_r} is determined by
$$\frac{\bar{b}_r}{y_{rk}} = \min_{1 \leq i \leq m}\left(\frac{\bar{b}_i}{y_{ik}} : y_{ik} > 0\right)$$
Update R, the basis inverse $\mathbf{B}_{-1}$, and the new basic feasible solution
$$\begin{aligned} x_{B_i} &= \bar{b}_i - \frac{y_{ik}}{y_{rk}}\bar{b}_r, \; i = 1, 2, \ldots, m \; ; i \neq r \\ x_k &= \frac{\bar{b}_r}{y_{rk}} \\ x_j &= 0, \; j \in R \end{aligned}$$
and repeat step 3.

4.2.3 Parametric programming

Sensitivity analysis, of which *parametric analysis/programming* is a subtopic, is concerned with investigating the influence on the optimal solution when changes are made to the linear program. These changes may constitute alterations in the cost vector $\mathbf{c}$, right-hand side vector $\mathbf{b}$ or the constraint

matrix **A**. Parametric programming aims at finding the range of values for which the cost vector or right-hand side vector can change, that will bring about changes in the basis and consequently also changes in the optimal solution. This is achieved by applying a perturbation technique to the cost vector or right-hand side vector respectively. In this study only perturbation of the right-hand side is considered.

Consider the linear program formulation 4.8 where the objective function and the basic feasible solution is given as

$$\begin{aligned} z &= \mathbf{c}_B\mathbf{B}^{-1}\mathbf{b} - (\mathbf{c}_B\mathbf{B}^{-1}\mathbf{N} - \mathbf{c}_N)\,\mathbf{x}_N \\ \mathbf{x}_B &= \mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_N \end{aligned} \tag{4.13}$$

Suppose $\mathbf{b}$ is replaced by $\mathbf{b} + \lambda\mathbf{b}'$ where $\lambda \geq 0$, this means that the right-hand-side $\mathbf{b}$ is perturbed along the vector $\mathbf{b}'$. Then 4.13 becomes

$$\begin{aligned} z &= \mathbf{c}_B\mathbf{B}^{-1}(\mathbf{b} + \lambda\mathbf{b}') - (\mathbf{c}_B\mathbf{B}^{-1}\mathbf{N} - \mathbf{c}_N)\,\mathbf{x}_N \\ \mathbf{x}_B &= \mathbf{B}^{-1}(\mathbf{b} + \lambda\mathbf{b}') - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_N \end{aligned} \tag{4.14}$$

The current basis will remain optimal with the newly perturbed right-hand side as long as $\mathbf{B}^{-1}(\mathbf{b} + \lambda\mathbf{b}')$ remains nonnegative. To determine the value of λ at which another basis becomes optimal, let $S = \{i : \bar{b}'_i < 0\}$ where $\bar{\mathbf{b}}' = \mathbf{B}^{-1}\mathbf{b}'$. If $S = \Phi$, then the current basis is optimal for all values of $\lambda \geq 0$. If this is not the case, let

$$\hat{\lambda} = \frac{\bar{b}_r}{-\bar{b}'_r} = \min_{i \in S}\left(\frac{\bar{b}_i}{-\bar{b}'_i}\right)$$

Let $\lambda_1 = \hat{\lambda}$. For $\lambda \in [0, \lambda_1]$ the current basis is optimal with $\mathbf{x}_B = \mathbf{B}^{-1}(\mathbf{b} + \lambda\mathbf{b}')$. If the value of λ is increased up to the point where $\lambda = \lambda_1$, the right-hand side is replaced by $\mathbf{B}^{-1}(\mathbf{b} + \lambda_1\mathbf{b}')$, x_{B_r} is removed from the basis, and an appropriate non-basic variable is selected to enter the basis. Note, however, that the current basic solution is optimal and that all reduced cost coefficients are negative. Consequently an alternative pricing criterion for selecting the entering variable, the *dual simplex* method criterion (see Bazaraa, Jarvis, and Sherali [BJS90], Chvatal [Chv83], and Dantzig [Dan63]), is used. The basis is updated to reflect the changes introduced by the entering variable.

The process of finding the next range $[\lambda_1, \lambda_2]$ over which the new basis is optimal and where $\lambda_2 = \hat{\lambda}$, is repeated. If S is empty , or all the values in

row r (the row corresponding to the leaving variable) are nonnegative, the process is terminated. In the latter case it means that no feasible solutions exist for all values of λ greater than the current value.

The results from applying parametric programming to a problem, is a range of optimal objective values corresponding to a range of right-hand side values. Figure 4.4 illustrates how the objective values may behave as a function of λ.

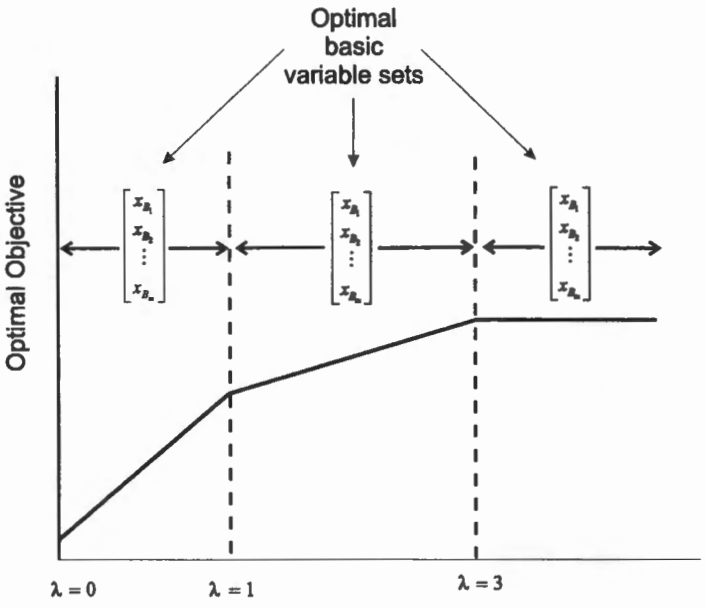

Figure 4.4: Optimal solutions found with parametric programming.

4.3 Applying linear programming to constrained linear models

In the article "Constrained regression models for optimization and forecasting" Bruwer and Hattingh ([BH85]) suggested that the area of experience for a linear model should be taken into account for optimization and forecasting purposes. This was achieved by using the convex hull of the observed values for which the linear model was fitted.

4.3.1 Representing the area of experience as a convex hull

Consider the estimated regression function

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \ldots + \hat{\beta}_k x_k \tag{4.15}$$

fitted to the data points

$$\mathbf{w}_y = \begin{bmatrix} w_{y_1} \\ w_{y_2} \\ \vdots \\ w_{y_m} \end{bmatrix} \text{ and } \mathbf{W}_x = \begin{bmatrix} w_{x_{11}} & w_{x_{12}} & \ldots & w_{x_{1k}} \\ w_{x_{21}} & w_{x_{22}} & \ldots & w_{x_{2k}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{x_{m1}} & w_{x_{m2}} & \ldots & w_{x_{mk}} \end{bmatrix}$$

where w_{y_i} denotes the i^{th} value observed for the response variable y and $w_{x_{ij}}$ the i^{th} observation measured for variable x_j. Let $\mathbf{w}_{\mathbf{x_1}}, \mathbf{w}_{\mathbf{x_2}}, \ldots, \mathbf{w}_{\mathbf{x_m}}$ denote the row vectors of $\mathbf{W}_x$, then the convex hull for the observed values is

$$C = \{\mathbf{X} : \mathbf{X} \in E^k \text{ and } \mathbf{X} = \sum_{i=1}^{m} \alpha_i \mathbf{w}_{\mathbf{x_i}} \tag{4.16}$$
$$\text{with } \alpha_i \geq 0 \text{ and } \sum_{i=1}^{m} \alpha_i = 1\}$$

4.3.2 Optimization of the linear model over the convex hull

The definition of the convex hull for the observations, is followed by the formulation of the linear program. Bruwer and Hattingh suggested taking the estimated regression function as the objective function of the linear program, and the convex hull as the constraints for the linear program.

$$
\begin{array}{lccccccccccc}
\text{Max} & \hat{\beta}_1 x_1 & + & \hat{\beta}_2 x_2 & + & \ldots & + & \hat{\beta}_k x_k & & & & \\
\text{Subj. to} & x_1 & & & & & & & - & \sum_{i=1}^{m} \alpha_i w_{x_{i1}} & = & 0 \\
 & & & x_2 & & & & & - & \sum_{i=1}^{m} \alpha_i w_{x_{i2}} & = & 0 \\
 & & & & & \ddots & & & & \vdots & & \vdots \\
 & & & & & & & x_k & - & \sum_{i=1}^{m} \alpha_i w_{x_{ik}} & = & 0 \\
 & & & & & & & & & \sum_{i=1}^{m} \alpha_i & = & 1 \\
 & & & & & & & \alpha_1, & \alpha_2, & \ldots, \alpha_m & \geq & 0
\end{array}
\tag{4.17}
$$

The solution to the linear program 4.17 gives the optimal solutions for the independent variables $x_1, x_2, \ldots, x_k$, within the convex hull of data points. Note that the constant β_0 is not included in the formulation, but needs to be added to the optimal value of the objective function. The linear program 4.17 can also be solved for the minimization case.

Bruwer and Hattingh suggested that an additional constraint $\boxed{x_j = p_{jl}}$ be added to the constraints in 4.17. By solving the linear program 4.17 iteratively for a range of values $p_{j1}, p_{j2}, \ldots, p_{js}$ with

$$p_{jl} \in \left[\min_{1 \leq i \leq m} (w_{x_{ij}}), \max_{1 \leq i \leq m} (w_{x_{ij}}) \right]$$

a range of optimal values $z_{j1}, z_{j2}, \ldots, z_{js}$ is generated. Displaying the range of values for p_{jl} against the optimal values z_{jl} represents the behavior of the linear model within the convex hull with respect to x_j.

4.3.3 Generating graphical results

Consider Example 2.1 (investigating the feasibility of an irrigation scheme) where the linear model

$$\hat{y} = -144.6 + 5.71x_1 + 2.95x_2 \tag{4.18}$$

was fitted to the data listed in Table 2.1. In this example, the factors influencing the production $\hat{y}$ are rainfall (x_1) and temperature (x_2). Let

$$\hat{\beta} = [5.71, 2.95], \ \mathbf{x} = [x_1, x_2]$$

and the matrix $\mathbf{W} = [\mathbf{w}_y, \mathbf{W}_x]$ denote the data listed in Table 2.1 where

$$\mathbf{w}_y = \begin{bmatrix} 60 \\ 50 \\ 70 \\ 70 \\ 80 \\ 50 \\ 60 \\ 40 \end{bmatrix}$$

are the observed values for the response variable y and

$$\mathbf{W}_x = \begin{bmatrix} 8 & 56 \\ 10 & 47 \\ 11 & 53 \\ 10 & 53 \\ 9 & 56 \\ 9 & 47 \\ 12 & 44 \\ 11 & 44 \end{bmatrix}$$

the the observed values for the independent variables x_1 and x_2. The problem can be formulated according to 4.17 in matrix notation as

$$\begin{array}{rrcrcl} \text{Max} & \hat{\beta}\mathbf{x} & & & & \\ \text{Subject to} & \mathbf{I}\mathbf{x} & - & \mathbf{W}_x^T\alpha & = & 0 \\ & & & \mathbf{1}\alpha & = & 1 \\ & & & \alpha & \geq & 0 \end{array} \tag{4.19}$$

where $\mathbf{I}$ is an 2×2 identity matrix and $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_8]^T$. Note that we do not in general require the x-vector to be nonnegative. In the present instance we could require it without loss of generality. In general a transformation of the model could be done to have one with nonnegativity (see Dantzig [Dan63]) The graphical results are generated by constraining one of the independent variables x_1 or x_2 to the values $p_{j1}, p_{j2}, \ldots, p_{js}$, as explained in the previous section, and to solve each linear program 4.19. Suppose the variable x_2 is under investigation, then $p_{2l} \in [44, 56]$ takes on the values, say, $44, 45.5, 47, 48.5, 50, 51.5, 53, 54.5, 56$ respectively (if, for example, 8 equal intervals are considered).

For $l = 1$, the solution to the linear program 4.19 with additional constraint $x_2 = p_{21} = 44$ is $z_{21} = 198.32 - \beta_0 = 53.72$. For $l = 2$, the additional

constraint becomes $x_2 = p_{22} = 45.5$, and the solution is $z_{22} = 201.791 - \beta_0 = 57.19$. By solving linear programs for the 9 values of p_{2l} a set of results z_{2l} is recorded for $l = 1, 2, \ldots, 9$. Plotting the values z_{2l} (which is the optimal values of $\hat{y}$ with variables constrained to the convex hull) against p_{2l} (the values to which x_2 was constrained) as a line graph produces the estimated maximum response line in Figure 4.5.

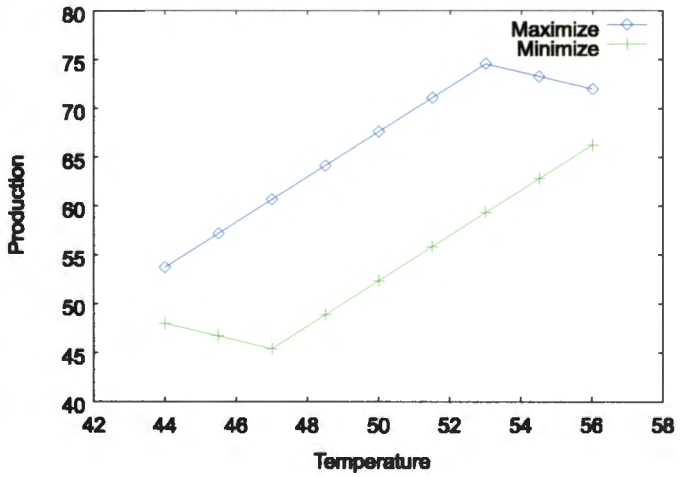

Figure 4.5: Response graph for Temperature in Example 2.1.

The foregoing procedure of generating the maximum response line is repeated to obtain the minimum response line in Figure 4.5. The only change to be made is for the linear program 4.19 to be a minimization problem.

4.3.4 Properties of the graphical results

An important property to notice from the graphical results generated, is that for each solution z_i found, the optimal conditions $\mathbf{x}_i^* = [x_1^*, x_2^*, \ldots, x_k^*]$ can be obtained from the solution to the linear program. For example, the solution found for solving 4.19 with the constraint $x_2 = p_{21} = 44$ was $z_{21} = 53.72$ with the optimal conditions for the independent variables as $\mathbf{x}_1^* = [12, 44]$.

The vertical distances between the maximum and minimum piecewise linear graphs are an indication of the relative importance of other independent variables not fixed in the linear program. It is easy to see that if a variable x_j is constrained at some level, and there is a large difference between the minimum and maximum solution, then the other non-constrained variables are contributing towards these differences. Figure 4.6 shows the results when the variable x_1 is investigated. The vertical distances between the minimum

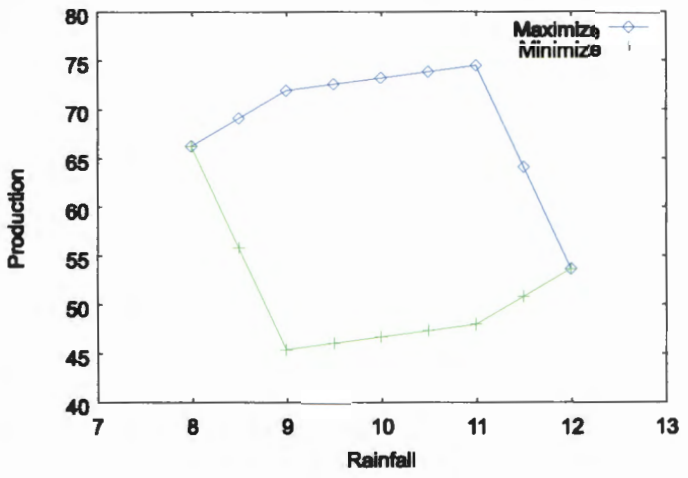

Figure 4.6: Response graph for Rainfall in Example 2.1.

and maximum graphs for x_1 are considerably larger than the vertical distances displayed between the minimum and maximum graphs for x_2. This indicates that the variable x_2 has a greater effect on the response variable.

4.4 Summary

The technique outlined above is referred to as *linear response surface analysis* (LRSA). Since the publication of the article "Constrained regression models for optimization and forecasting" by Bruwer and Hattingh [BH85] the technique has been used in a number of applications. One of these applications is using LRSA for interpreting economic models (see Roux [Rou94]). Another application by Bruwer [Bru83] is evaluating computer based systems using LRSA. In an application modelling the success of database technology in companies, Jordaan [DJT93] extended the method somewhat by using multiple goal approaches. Van Deventer [Dev95] used the technique as a basis for a decision support system on a personal computer. This system can take as input a data set and then display graphs where one variable at a time is restricted to various levels. Queries concerning the levels of response estimated at certain levels of variables can be handled by the system. Although the results obtained from using LRSA are regarded as useful, some extensions and refinements to LRSA will be considered in this study.

Chapter 5

Extensions of linear response surface analysis

5.1 Introduction

The objective of this study is to investigate the feasibility of LRSA as a decision support system and to enhance current functionality. Some of the needs identified are

- **Robustness**: Current experimental code is restricted to small sets of data and a more robust simplex solver is required.
- **Automated interval selection**: The current system requires user input to specify the range of values for which the variable under investigation is restricted. Automation is required to achieve this.
- **Multi-variable investigation**: The results of LRSA display response graphs for a single variable at a time. Multi-variable response graphs are required since models often contain more than one state variable.
- **Real-time interactivity**: The analysis and investigation into linear models using LRSA, requires an interactive system. For large problem sizes the execution time may defeat the idea of an interactive decision support system that can produce graphical representations on demand. The solving process thus has to be optimized for speed.

5.2 Implementation of a third party simplex solver

The decision for using a third party simplex solver is to address the problem of robustness. The *Optimization Subroutine Library* (OSL), a product of IBM, was selected for this purpose.[1] To meet the platform requirements of OSL, the code for LRSA is developed with the programming language C++. Other advantages of developing the code with C++ is platform independence and the use of *Object Orientation Methodology.*

5.3 A Parametric approach to interval selection

The primary process outlined for the LRSA algorithm (Section 4.3), suggests iterative solving of the linear program 4.17. With each iteration the right-hand side of the constraint $x_j = p_{jl}$ is updated with the values $p_{j1}, p_{j2}, \ldots, p_{js}$. The number of equally spaced intervals k that determines the number of right-hand side values s, i.e. $s = k + 1$ (see Section 4.3.3), is specified by the user of the LRSA system. Consequently, the number of result sets (z_{jl}, p_{jl}) that will appear in the response graph is user dependent.

Figures 4.5 and 4.6 are the results obtained from applying LRSA to the data of Example 2.1 (investigating the feasibility of an irrigation scheme). The number of equally spaced intervals that were specified for the response graphs is 8. Now suppose we repeat the exercise of generating a response graph for variable x_1 with only 4 equally spaced intervals, then from Figure 5.1, we notice that the results differ (the original graph generated from 8 intervals is indicated with dotted lines). Clearly, some information is lost if the right-hand-side values for the additional constraint are selected too far apart. This problem can be solved much more elegantly with parametric programming.

Consider the linear program formulation of LRSA (4.17). For illustration purposes, suppose the variable x_1 is under investigation and

$$p_{j,min} = \min_{1 \leq i \leq m} (w_{x_{i1}})$$

with $w_{x_{i1}}$ the i^{th} observation for x_1. When the constraint $x_1 = p_{1,\text{min}}$ is

[1]The product is available for research purposes. The implementation of OSL as part of LRSA, therefore, falls within legal copy right laws. This is the case as long as LRSA is developed for experimental use.

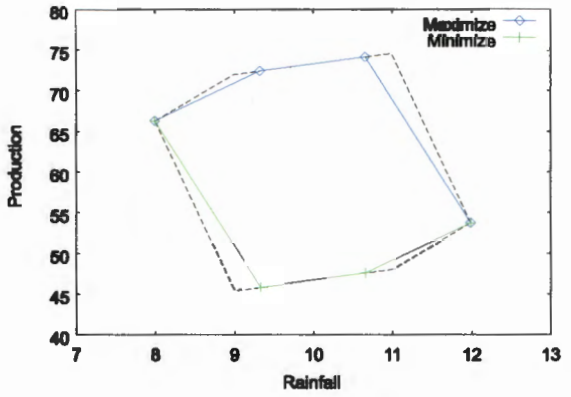

Figure 5.1: Differences in response due to fewer intervals specified.

included in 4.17, the linear program becomes

$$\begin{array}{lccccccccccc}
\text{Max} & \hat{\beta}_1 x_1 & + & \hat{\beta}_2 x_2 & + & \ldots & + & \hat{\beta}_k x_k & & & & \\
\text{Subj. to} & x_1 & & & & & & & - & \sum_{i=1}^{m} \alpha_i w_{x_{i1}} & = & 0 \\
& & & x_2 & & & & & - & \sum_{i=1}^{m} \alpha_i w_{x_{i2}} & = & 0 \\
& & & & & \ddots & & & & \vdots & & \vdots \\
& & & & & & & x_k & - & \sum_{i=1}^{m} \alpha_i w_{x_{ik}} & = & 0 \\
& & & & & & & & & \sum_{i=1}^{m} \alpha_i & = & 1 \\
& x_1 & & & & & & & & & = & p_{1,\min} \\
& & & & & & & \alpha_1, & \alpha_2, & \ldots, \alpha_m & \geq & 0
\end{array} \tag{5.1}$$

With the optimal basis at hand after solving the linear program 5.1, the parametric programming procedure suggests that the right-hand side $\mathbf{b}$ is perturbed along the vector $\mathbf{b}'$. Since the influence of x_1 is being investigated, we would like to perturb along a vector with respect to the constraint $x_1 = p_{1,\min}$. Therefore, the right-hand side of 5.1 becomes

$$\mathbf{b} + \lambda \mathbf{b}' = \begin{bmatrix} \mathbf{0} \\ 1 \\ p_{1,\min} \end{bmatrix} + \lambda \begin{bmatrix} \mathbf{0} \\ 0 \\ 1 \end{bmatrix}$$

where $\mathbf{0}$ is a $k \times 1$ vector of zeros. Following the parametric programming procedure outlined in Section 4.2.3, a range of values for λ corresponding to optimal basis changes in the linear program, $\lambda = [\lambda_{11}, \lambda_{12}, \ldots, \lambda_{1u}]$, is obtained. We also find objective values z_{1l} corresponding to each λ_{1l}. Note that the number of λ-values depends on the convex space and, therefore, u cannot be predetermined.

Graphing the values $p_{1l} = p_{1,\min} + \lambda_{1l}$ against the objective values z_{1l} for $l = 1, 2, \ldots, u$ gives a maximum response graph for variable x_1. To generate the minimum response graph for x_1, 5.1 is changed to a minimization problem.

5.4 Multi-variable response graphs

The practical implication of generating response graphs using LRSA is that for each optimal response obtained, the optimal values for the other non-constraint variables in the model are given. This is very useful in situations where state variables (variables not under the control of the experimenter) are investigated. For instance, the variable x_2 (temperature) from Example 2.1 (investigating the feasibility of an irrigation scheme) was considered as a state variable. From the response graphs in Figure 4.6, it is now possible to determine what the values for the other regressors should be in order to obtain the best production, given the temperature scenario for a specific region.

In applications where more than one state variable is present, the need may exist to view the estimated response for two state variables. Consequently the results will be a response surface in three dimensions.

5.4.1 Three-dimensional response surfaces

Consider the formulation of LRSA (4.17) for the linear model

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \ldots + \hat{\beta}_k x_k \tag{5.2}$$

with $k > 2$. From the discussion of LRSA we know that a set of results is generated by constraining some variable, say x_j, at levels $p_{j1}, p_{j2}, \ldots, p_{js}$ respectively. This gives rise to minimum (maximum) estimated response graphs. Suppose that an extra constraint is added to the formulation requiring that some other (state) variable, say x_t, is fixed at a level p_t. That is, the constraint $x_t = p_t$ needs to be part of the constraint set of 4.17. Solving the problem iteratively, while setting x_j equal to the values p_{jl} at each iteration, will yield solutions z_{jl} for $l = 1, 2, \ldots, s$ where x_t is fixed at a level p_t. Figure 5.2 shows the results for this problem when drawing the response graphs in the (x_j, x_t, z) space. Clearly, a range of response graphs for the variable x_j can now be generated for a range of values $p_{t1}, p_{t2}, \ldots, p_{tr}$ for the variable x_t which will display a response surface. To illustrate the concept, let the function $LP_{Minimize}(index\_j, value\_p, index\_t, value\_q)$ denote

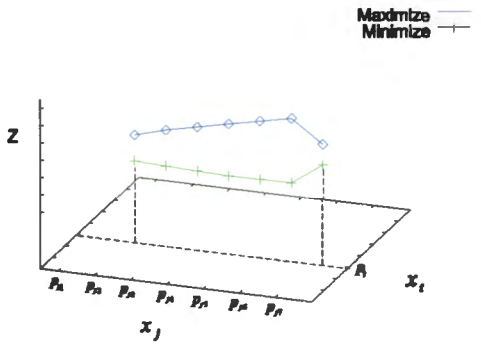

Figure 5.2: Response graph for x_j with $x_t = p_t$.

the LRSA formulation (4.17), with additional constraints $\boxed{x_{index\_j} = value\_p}$ and $\boxed{x_{index\_t} = value\_q}$ in the following pseudo code

for m=1 to s do
    begin
        for n=1 to r do
            begin
                $z[m][n] = LP_{Minimize}(j, p[m], t, q[n])$
            end
    end

The matrix $\mathbf{z}$ holds the minimized values for the LRSA problem for all combinations of $p_{j1}, p_{j2}, \ldots, p_{js}$ and $p_{t1}, p_{t2}, \ldots, p_{tr}$ used in the constraints $x_j = p_{jm}$ and $x_t = p_{tn}$. Note, however, that not all combinations of constraints will have feasible solutions and that the matrix $\mathbf{z}$ may have null entries. Graphing the points $(p_{j1}, p_{th}, z_{1h}), (p_{j2}, p_{th}, z_{2h}), \ldots, (p_{js}, p_{th}, z_{sh})$ for $h = 1, 2, \ldots, r$ with $z_{mn} \in \mathbf{z}$, we obtain Figure 5.3 which is a range of response graphs for x_j corresponding to each p_{th}. Graphing the points $(p_{jl}, p_{t1}, z_{j1}), (p_{jl}, p_{t2}, z_{l2}), \ldots, (p_{jl}, p_{tr}, z_{lr})$ for $l = 1, 2, \ldots, s$ with $z_{mn} \in \mathbf{z}$, we obtain Figure 5.4 which is a range of response graphs for x_t corresponding to each p_{jl}. Combining the graphs from Figure 5.3 and 5.4, a three-dimensional surface plot is created (Figure 5.5). Although the foregoing illustration only considered the minimization formulation of LRSA, a maximum response surface can be generated by changing the LRSA formulation to maximization.

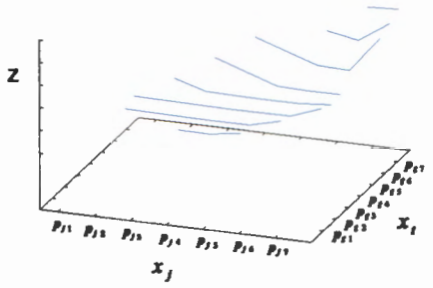

Figure 5.3: Response graphs for x_j with $x_t = p_{th}$.

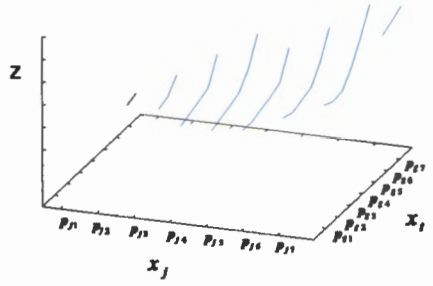

Figure 5.4: Response graphs for x_t with $x_j = p_{jl}$.

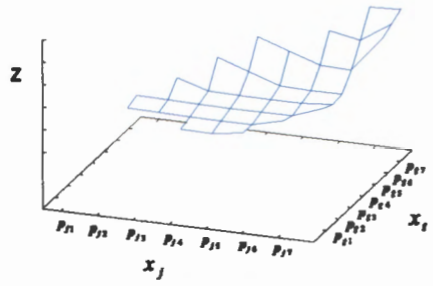

Figure 5.5: Combined response graphs for x_j and x_t.

5.4.2 Two-dimensional contour graphs

Another effective way to display the behaviour of a response function given, two variables at the axis, is the use of contour plots. A contour plot will reveal the optimal response surface in terms of response contours with each contour representing a different response level. An approach to generate contour graphs for the LRSA problem (4.17) is to reformulate the problem in such a way that the objective function is constrained at some level. A range of values

for the two variables under consideration, representing a contour if displayed in two dimensions, can be found that will satisfy the said constraint.

Consider the following parametric linear program:

$$
\begin{array}{lccccccccccc}
\text{Min/Max} & p_t & = & x_t & & & & & & & & \\
\text{Subj. to} & \hat{\beta}_1 x_1 & + & \hat{\beta}_2 x_2 & + & \ldots & + & \hat{\beta}_k x_k & & & = & z \\
& x_1 & & & & & & & - & \sum_{i=1}^{m} \alpha_i w_{x_{i1}} & = & 0 \\
& & & x_2 & & & & & - & \sum_{i=1}^{m} \alpha_i w_{x_{i2}} & = & 0 \\
& & & & & \ddots & & & & \vdots & & \vdots \\
& & & & & & & x_k & - & \sum_{i=1}^{m} \alpha_i w_{x_{ik}} & = & 0 \\
& & & & & & & & & \sum_{i=1}^{m} \alpha_i & = & 1 \\
& & & & & & & & & x_j & = & p_j \\
& & & x_1, & x_2, & \ldots, & x_k, & \alpha_1, & \alpha_2, & \ldots, \alpha_m & \geq & 0
\end{array}
\tag{5.3}
$$

The parametric linear program 5.3 will not have solutions for all values of z and p_j. In fact it will definitely not have solutions for

$$z \notin [\hat{y}_{\min}, \hat{y}_{\max}]$$

where

$$\hat{y}_{\min} = \min_{1 \leq i \leq m} (\hat{\beta}_0 + \hat{\beta}_1 w_{x_{ij}} + \ldots + \hat{\beta}_k w_{x_{ik}})$$

and

$$\hat{y}_{\max} = \max_{1 \leq i \leq m} (\hat{\beta}_0 + \hat{\beta}_1 w_{x_{ij}} + \ldots + \hat{\beta}_k w_{x_{ik}})$$

If maximized for a feasible z and p_j, the parametric linear program 5.3 will produce an optimal objective value p_t. For this optimal solution the constraint denoting the original LRSA objective function is fixed at a level z and the variable x_j is fixed at a level p_j. The point (p_j, p_t) in the (x_j, x_t) space is a single point of a contour representing a response level of z.

The parametric programming procedure of perturbing along some vector $\mathbf{b}'$ can now be applied with respect to the constraint $x_j = p_{j,\min}$ in 5.3 in order to determine a range of points that will represent a contour. For a

specified z the value $p_{j,\min}$ can be determined by solving the linear program

$$\begin{array}{lllllllllllll}
\text{Min} & p_{j,\min} & = & x_j & & & & & & & & & \\
\text{Subj. to} & \hat{\beta}_1 x_1 & + & \hat{\beta}_2 x_2 & + & \ldots & + & \hat{\beta}_k x_k & & & = & z \\
& x_1 & & & & & & & - & \sum_{i=1}^{m} \alpha_i w_{x_{i1}} & = & 0 \\
& & & x_2 & & & & & - & \sum_{i=1}^{m} \alpha_i w_{x_{i2}} & = & 0 \\
& & & & & \ddots & & & & \vdots & & \vdots \\
& & & & & & & x_k & - & \sum_{i=1}^{m} \alpha_i w_{x_{ik}} & = & 0 \\
& & & & & & & & & \sum_{i=1}^{m} \alpha_i & = & 1 \\
& & & x_1, & x_2, & \ldots, & x_k, & \alpha_1, & \alpha_2, & \ldots, \alpha_m & \geq & 0
\end{array} \tag{5.4}$$

If 5.4 has a feasible solution then the right-hand side of 5.3 becomes

$$\mathbf{b} + \lambda \mathbf{b}' = \begin{bmatrix} z \\ \mathbf{0} \\ 1 \\ p_{j,\min} \end{bmatrix} + \lambda \begin{bmatrix} 0 \\ \mathbf{0} \\ 0 \\ 1 \end{bmatrix}$$

where $\mathbf{0}$ is a $(k+1)\times 1$ vector of zeros. Following the parametric programming procedure outlined in Section 4.2.3, a range of optimal values $p_{jl} = p_{j,\min} + \lambda_{jl}$ and corresponding p_{tl} for $l = 1, 2, \ldots, s$ is obtained. Note, however, that these optimal values are obtained through maximizing the parametric linear program 5.3 and that this, therefore, represents only a portion of the contour. The remaining portion of the contour is obtained by minimizing the parametric linear program 5.3. Let us denote the optimal values obtained through maximizing and minimizing the parametric linear program by the points $(p_{jl}, p_{tl})_{max}$ and $(p_{jh}, p_{th})_{min}$ respectively. Figure 5.6 illustrates

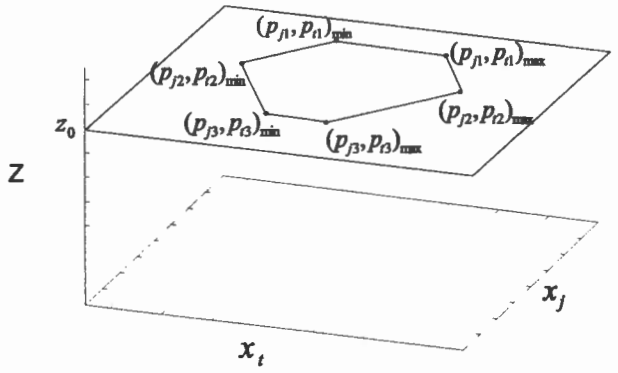

Figure 5.6: A contour for the objective function at z_0.

a typical contour for some response level z_0. The contour is generated by connecting the points $(p_{jl}, p_{tl})_{max}$ and $(p_{jh}, p_{th})_{min}$ sequentially.

Contours corresponding to a range of response levels can be created by repeating the above process for a range of response levels $z_1, z_2, \ldots, z_q$. Figure 5.7 shows an illustration of a typical contour plot in the (x_j, x_t) space

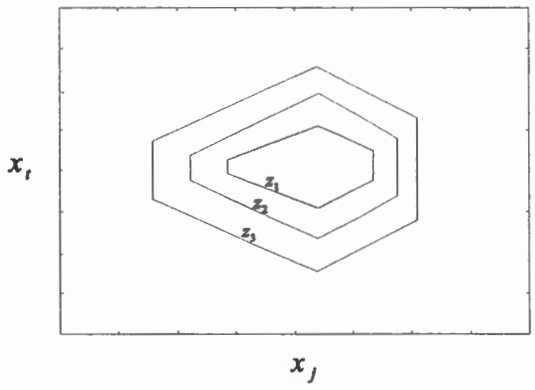

Figure 5.7: Typical contours in the (x_j, x_t) space.

where the parametric process was applied to the response levels z_1, z_2, z_3. The contour levels z_1, z_2, z_3 in the contour plot were chosen to satisfy the inequality $z_3 < z_2 < z_1$. This entails that the inner contour associated with z_1 is the maximum estimated response found for the different combinations of levels for the two variables at the axis. For purposes of illustration only a top view of the response surface was considered in the contour plot. This means that the contours representing the lower response levels have been omitted from the graph. Furthermore, the contours were chosen to be concentric. Figure 5.8 shows a more realistic representation of a typical contour plot. For this illustration the inequality $z_5 < z_4 < z_3 < z_2 < z_1$ holds and the contours span the entire range of the response levels. This entails that the contour labelled z_5 represents the minimum estimated response and the contour labelled z_1 represents the maximum estimated response. The overlapping of contours (which is a result of the convex property of the model in three dimensions) makes interpretation of the contour plot difficult. An approach to make interpretation easier is to colour each contour polygon with a different colour as shown in Figure 5.9. The order in which the contour polygons are coloured determine whether a "minimum response" or a "maximum response" graph is displayed. The contour plot in Figure 5.9 is considered to be a "maximum response" graph. The reason for this is that the contour representing the maximum response level z_1 was the last one to be coloured.

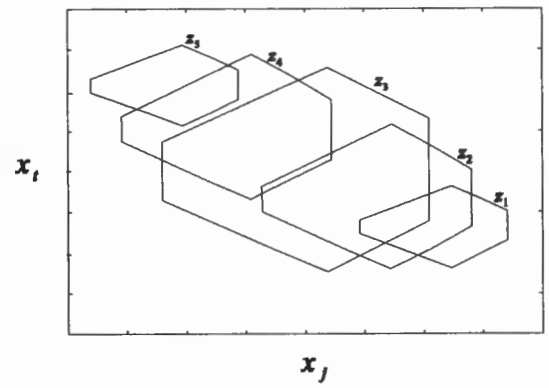

Figure 5.8: Overlapping contours.

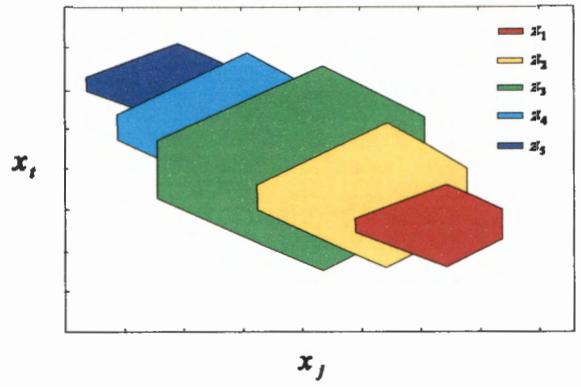

Figure 5.9: Maximum response graph with coloured contours.

The contour, therefore, is not being overlapped by any other contour and is entirely visible. A "minimum response" graph will, consequently, make the minimum response level z_5 entirely visible. Figure 5.10 shows the "minimum response" graph generated by colouring the contour polygons in the order $z_1, z_2, \ldots, z_5$.

In Chapter 6 the various extensions discussed so far will be applied to an example.

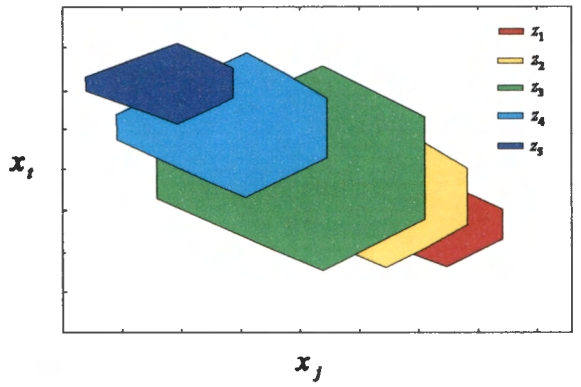

Figure 5.10: Minimum response graph with coloured contours.

5.5 Enhancements for optimizing performance

5.5.1 Parallel implementation of LRSA

For the implementation of Linear Response Surface Analysis, a parallel processing architecture is considered. The main reason for this is the need for processing power in order to apply the method to huge amounts of data and to deliver instant graphical results. The structure of LRSA is also feasible for decomposition, especially in the extensions of multi-variable investigation.

5.5.2 Classification models for parallel computers

Classification models for parallel computers are desirable for the evaluation and comparison of computers according to their architecture. One of the first classification models used for computers is the Von Neumann model based on the architecture of the first computer developed by Von Neumann. The Von Neuman model consists of a single processing unit (CPU) and a single storage unit (memory) where a single sequence of instructions operate on a single sequence of data.

Currently parallel computers consist of more than one processing unit. Some computer architectures include multiple sequences of instructions which can operate on multiple sequences of data. Flynn [Fly72] proposed a new classification where four major models are distinguished: SISD, SIMD, MISD and MIMD.

The SISD Model

The *Single Instruction, Single Data* model refers to the traditional Von Neumann architecture where a single processing unit is used which executes a single stream of instructions on a single stream of data.

The SIMD Model

The *Single Instruction, Multiple Data* model is a more specialized class of parallel computer where a single instruction stream is executed on different streams of data. SIMD computers are appropriate for problems where the same operations are performed on different data sets, for example some smoothing techniques used in image processing.

The MISD Model

There is not really any computer architecture that falls in the category of *Multiple Instruction, Single Data.* Such a computer would require that different instruction streams operate on the same set of data simultaneously. Although this is not impossible, its effectiveness will be in question considering the effort in controlling read and write access to the data. The architecture closest to this concept is pipelined computers. In this case a number of processors pass a stream of data from one processor to the next and each processor performs a different operation on the data.

The MIMD Model

The MIMD model refers to a *Multiple Instruction, Multiple Data* architecture. That entails that each processor in a MIMD computer executes its own instruction stream and operates on its own set of data. There are also two variations of this model: distributed-memory systems and shared-memory systems. In distributed-memory systems each processor has its own local memory.

Distributed-memory systems are sometimes referred to as multicomputers. The distinction is usually based on the coupling scheme of the processors. This means that if the processors are tightly coupled (e.g. in the same space) with typically high speed communication links then it is a multicomputer, otherwise it is a distributed system.

In shared-memory systems the processors of the computer share a common memory space. Each processor also has access to a local cache which enhances performance if the same data items are fetched frequently. Shared-memory systems are also referred to as multiprocessors.

5.5.3 Programming Paradigms

The Von Neumann machine instruction set consists of only a few primitive instructions with which it is possible to perform only the basic operations like addition, subtraction, etc. By adding consecutively higher level languages like micro code, assembler and finally something like C or Fortran, it is possible to do more complex operations like iterations (loops), function calls, etc. By adding abstract data types (e.g. object-oriented classes) another level of abstraction is achieved. In order to transform one level of abstraction to another, the concept of modularity is used. This entails that one program instruction at a higher level of abstraction represents a number of lower level instructions.

Abstraction and modularity are also important in the development of parallel programs. It would be tedious to try and write machine instructions for each task on each processor and keep track of all the interprocess communications on machine level. The abstraction is also necessary to describe the mechanisms needed to obtain concurrency and scalability.

Tasks and channels

A task encapsulates a sequential program and local memory. It and can be executed concurrently with other tasks. Each task can perform four basic actions in addition to reading and writing its local memory (if local memory is presént): sending and receiving messages to/from other tasks, creating new tasks, and terminating. Tasks are finally mapped to physical processors and the mapping employed does not affect the semantics of the program. Channels are the message queues which connect the tasks with each other. More than one channel can be used to connect two tasks with each other and a naming convention is used to uniquely identify each channel.

Message passing

A message passing model is considered to be a minor variation of the task/channel model. Instead of sending a message along a specific channel the message is sent to a specific task. Other differences are the inability of message passing models to create or terminate tasks dynamically, the execution of multiple tasks per processor, and the execution of different programs by different tasks. In practice most message-passing systems create a fixed number of identical tasks at program start-up and do not allow new tasks to be created in addition to the current ones. The termination of current tasks is also not allowed during program execution. This kind of parallel program implementation is known as *Single Program, Multiple Data* (SPMD) models.

Data parallelism

This model is applied in situations where the same operation is performed on multiple data elements. The implementation of this model usually requires that data are to be distributed over processors. A data parallel compiler is used to translate the data parallel program into a SPMD formulation.

Shared memory

In this model a common address space is used and the tasks can read or write to this common address space in an asynchronous way. This means that more than one task can read or write to the same memory location simultaneously. To avoid reading and writing of the same memory location by two different tasks, mechanisms like locks and semaphores (commonly used in operating systems) can be used.

5.5.4 Designing parallel algorithms

The first step in designing parallel algorithms is obviously the classification of the computer according to its architecture. With this classification at hand it is possible to decide on a programming paradigm that suits the underlying architecture best. The computer model considered to be most flexible and most promising to achieve scalability is MIMD (Aki [Aki89], Foster [Fos95], Chalmers and Tidmus [CT96]). Scalability implies that if processors are added to the system, the work performance of the algorithm improves. Another advantage of MIMD architecture is the ability of porting alorithms from SIMD to MIMD. In this case the same program is executed on each of the processors of the MIMD computer. For these reasons the remainder of this discussion will only be based on the MIMD model.

Now, in order to do problem solving in parallel according to the MIMD model, the problem is decomposed into subproblems which are then assigned to different processors to be solve simultaneously. There are two approaches to decomposing a problem: *functional decomposition* and *domain decomposition* (Foster [Fos95], Chalmers and Tidmus [CT96]). Consider an example from Chalmers and Tidmus where the problem is to mark a large number of exam scripts. One way to go about this is for the lecturer to mark all the scripts him-/herself by comparing each script to a memo and assigning a mark. The structure of this problem lends itself to parallelism in cases where more than one lecturer can assist in marking the scripts. The first approach would be for each of the lecturers to mark different questions. If, for example, the paper has four questions and there are four lecturers, then lecturer

number one marks question number one of all the scripts, lecturer number two marks question number two of all the scripts, etc. This is functional decomposition. Implementing this approach means that each processor executes a different part of an algorithm but on the same data. The second approach is where all the lecturers mark the same questions but for different scripts. Say for instance there are 200 scripts and five lecturers, then each lecturer has to mark all the questions for 40 of the scripts. This is an example of domain decomposition. The domain of the problem refers to the set of data required to solve the problem. Implementing this approach means that all the processors execute the same algorithm on a subset of the data. This approach of using a single algorithm on multiple streams of data is also known as SAMD (*Single Algorithm, Multiple Data*).

The SAMD approach to parallelism can be implemented on a SIMD computer, in which case the instructions for the algorithm are executed on the processors synchronously. It can also be implemented on a MIMD computer on which the processors may execute their own instruction stream asynchronously and communicate when co-operation is necessary. The SAMD approach can be applied to a wide range of problems and in this study of parallel design only SAMD is considered.

Once the problem has been explored for possible decomposition to fit the SAMD approach, the next step is designing the sequential part of the algorithm which will be executed by each processor. For our example of marking the examination scripts (5 lecturers marking 40 scripts each with 4 questions per script), the sequential part of the algorithm for each of the five processors ("lecturers") is the following:

```
for i=start_script to end_script do
    begin
        for j=1 to 4 do
            begin
                Compare(Memo[j],Script[i],Question[j])
                Assign_Mark(Script[i],Question[j])
            end
    end
```

In the *for* loop the two variables start_script and end_script indicate from which script to which script the current processor needs to process. The values to be assigned to this variables can be determined locally i.e.

```
if processor1 then
  start_script=1
  end_script=40
```

which implies that processor number one is assigned to mark the scripts 1 to 40. Now that we know the task of each processor, the remainder of the designing process is concerned with the following issues.

- **Load balancing:** The previous paragraph showed that processor number one is assigned to process scripts 1 to 40, processor number 2 assigned to process scripts 41 to 80, etc. It can be assumed that the processing time for each processor to process 40 scripts is the same. An exception occurs if questions are incompletely answered. Then it is possible for some processors to complete their tasks before others, and to sit and idle. If this is the case consideration must be given to dynamic load balancing. For our example it would mean that scripts assigned to other processors can be re-assigned to processors currently being idle. Various load balancing techniques are discussed in Foster [Fos95].

- **Granularity:** Granularity describes the amount of computational work allocated to a processor in between communications. A processor is said to be *coarse grained* if it performs a large amount of computational work in between communications. As far as our example is concerned, a coarse grained processor will complete all of its scripts before sending back the results. For the processors to be *fine grained* the amount of computational work in between communications is smaller. For instance, in our example a fine grained processor will send back the results after completion of each script.

- **Data dependency:** As far as our example is concerned each processor must be provided with a subset of the domain. This can be achieved by some processor (let us call it the master) sending the data to all the other processors (say the slaves). Therefore, each slave processor is dependent of input data from the master processor. The presence of data dependency among slave processors is more common when a problem is decomposed according to its functional structure. As for our example of marking exam scripts is concerned, there is no data dependency among the slave processors. Therefore no processor needs to sit and idle while waiting for some other processor to sent the data.

All of the above issues involve a measure of communication. For example, if dynamic load balancing is used, more communication is needed for synchronization. If it is decided to adjust the grain of the processors to be more fine grained, then more communication is needed to provide the processor with data, or to gather results from the processors more frequently. And if

the problem involves data dependency among processors, more communication is needed for data sharing. It is important to note that communication between processors consumes useful computational time. Therefore, load balancing, granularity and data dependency together with their associated level of communication are considered to be the main factors influencing the performance of a parallel algorithm (see IBM [IBM96c]). A parallel algorithm design must include the appropriate choices for load balancing, granularity and data dependency to reduce overhead communication. Unfortunately there is no standard recipe for this. Various methodologies have, however, been proposed by authors like Foster [Fos95], Chamlers and Tidmus [CT96] and others to assist designers of parallel algorithms in this matter. In most of these methodologies the factors mentioned above which influence parallel performance, are addressed in one way or the other. For purposes of this study the different methodologies will not be discussed and evaluated, but rather used as guidelines to design a parallel algorithm for LRSA.

5.5.5 Performance measures

Measuring the performance of a parallel algorithm is a way of evaluating the design and implementation of the algorithm. It also serves to analyze the algorithm for optimization. Different performance metrics are defined by authors like Aki [Aki89], Chalmers and Tidmus [CT96] and, Censor and Zenios [CZ97]. Some of the most commonly used performance metrics are discussed next.

Speed-up

Speed-up is the ratio of the time taken to execute the sequential algorithm on a single processor to the time taken to execute the parallel algorithm on multiple processors. The sequential algorithm is executed on one of the processors of the parallel system.

$$\text{speed–up} = \frac{\text{elapsed time on one processor}}{\text{elapsed time on multiprocessor}} \tag{5.5}$$

From the equation it is evident that if the elapsed time on the multi processors decreases, the speed-up will increase. For the ideal parallel algorithm it is expected that if n processors are added to the system, then the algorithm will be n times faster, the elapsed time will decrease with a factor of n and consequently the speed-up will increase with a factor of n. The increase of speed-up with factor n with the addition of n processors is called *linear speed-up*. Unfortunately, cases of linear speed-up are unlikely to be

achieved. According to Chalmers and Tidmus [CT96] the failure to achieve linear speed-up is ascribed to the parallel algorithm suffering from *realisation penalty.* A realisation penalty can arise from two sources:

- **an algorithmic penalty:** Parallel algorithms inherently consist of some parts which need to be executed serially. For example the allocation of data to the processors and the collecting of the results from the processors need to be done serially. For parallel algorithms consisting of large portions of sequential execution, the possibility for linear speed-up is less likely, because of the restriction that a problem cannot be decomposed into smaller pieces indefinitely. Furthermore, subsequential execution in parallel algorithm requires synchronization. Consequently, processors might sit idle while waiting for other processors to synchronize.

- **an implementation penalty:** To solve a problem twice as fast on two processors implies that the processors spend 100% of their execution time on useful computation. This implies further that no load balancing is necessary and no data dependencies exist. This in turn implies that no communication is present. This scenario is, however, very unlikely for any parallel algorithm and implementation penalty is thus mainly caused by overhead communication.

Using Amdahl's law (Aki [Aki89], Chalmers and Tidmus [CT96], Censor and Zenios [CZ97], and Foster [Fos95]), it is possible to determine an upper bound for the speed-up of a parallel algorithm given a number of processors. Say the number of processors is n then:

$$\text{Maximum speed–up} = \frac{1}{s + \frac{p}{n}} \tag{5.6}$$

where s is the fraction of time the parallel algorithm spends on purely sequential execution (i.e. the allocation of data to each processor) and p the fraction of time the parallel algorithm spends on parallel execution ($s + p = 1$). If the sequential part did not exist in this formula then $p = 1$ and Amdahl's law reduces to maximum speed-up $= n$ for the n processor model.

Efficiency

The efficiency of a parallel algorithm is a measure which can be used to determine what percentage of a processor's time is being spent on useful computation.

$$\text{efficiency} = \frac{\text{speed–up} \times 100}{\text{number of processors}} \tag{5.7}$$

For linear speed-up, the efficiency of each processor will equal 100%, indicating that each processor spends all of its time on computation.

Other metrices

- **MIPS:** This performance metrics measure how many Million Instructions Per Second a computer is capable of performing.
- **FLOPS:** The number of Floating-point Operations Per Second executed by a computer is called FLOPS. MFLOPS donate how many Million (Mega) FLOPS are executed by a computer.
- **Dhrystone:** This is a benchmark for measuring the integer performance of a computer.
- **Whetstone:** This is a synthetic benchmark for measuring the floating point performance of a computer.

5.5.6 Decomposition of the LRSA problem

A parallel implementation is most likely to be used for the generation of multi-variable response graphs. The experimental parallel algorithm is based on the LRSA formulation for generating two dimensional contour graphs (see Section 5.4.2).

In order to generate a two dimensional contour graph we need to solve the parametric linear program 5.3 iteratively for a range of response levels. This corresponds to data decomposition of a problem where the same code is applied (simplex solver) but for different data (a range of response levels).

The resources we had available for a parallel implementation of LRSA is an IBM SP/2 multiprocessor. The architecture is based on a MIMD model with a message passing programming paradigm (see [IBM96b] and [IBM96a]). A very attractive feature of the system is the presence of a shared network file system. Static data can be allocated to the processors by means of the shared network file system.

Figure 5.11 is a conceptual representation of a data decomposition algorithm for LRSA. The shared file system is utilized by each slave processor to load the LRSA formulation (5.3) and the configuration data needed to solve the problem (e.g. the range of response levels $z_1, z_2, \ldots, z_q$). This entails that each slave processor has its own copy of the parametric linear

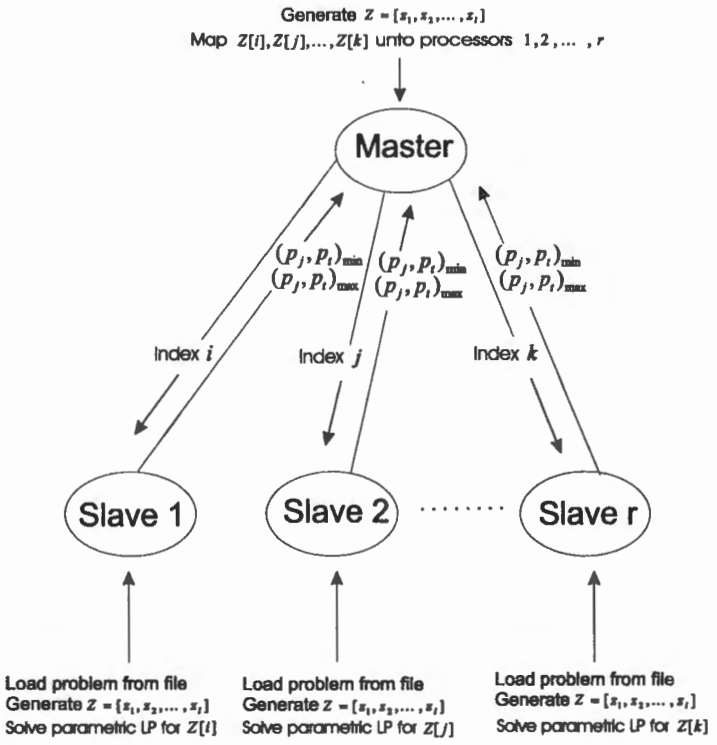

Figure 5.11: Parallel decomposition of the LRSA problem.

program. A master-slave topology is used to manage the tasks. The master processor, which is only running for administrative purposes, broadcasts different indices to the different slave processors. The indices indicate to the slave processors what response level to use from the vector $z_1, z_2, \ldots, z_q$ for substituting into the right-hand side of the constraint restricting the response level. After solving the parametric linear program the optimal points $(p_{j1}, p_{t1})_{min}, (p_{j2}, p_{t2})_{min}, \ldots, (p_{jr}, p_{tr})_{min}$ and $(p_{j1}, p_{t1})_{max}, (p_{j2}, p_{t2})_{max}, \ldots, (p_{js}, p_{ts})_{max}$ are sent back to the master processor by each slave processor. The master processor must keep track of which contour points the slave processors have generated for which response level.

Some implementation considerations were taken into account to address the following issues:

- **Load balancing:** Consider the parametric linear program 5.3 that produces a set of optimal points $(p_{jh}, p_{th})_{min}$ for $h = 1, 2, \ldots, r$ and $(p_{jl}, p_{tl})_{max}$ for $l = 1, 2, \ldots, s$. The number of optimal points corresponding to r and s, for the minimization and maximization case re-

spectively, cannot be predetermined (see Section 5.3). For this reason we cannot assume that the processing time for all the slave processors to solve a parametric linear program will be the same. Therefore, the implementation relies on a pool system where a slave processor can request from the master processor the next response level to be substituted in the parametric linear program after completion of its previous task. The master process is only running for these administrative duties and, therefore, the time delay in replying to all the requests will be minimal.

- **Granularity:** Two alternatives concerning granularity were considered. The first alternative is for each slave processor to batch solve a range of parametric linear programs for a specified range of response levels. This entails that a slave processor will request more than one response level at a time from the master processor. It will also provide the master processor with a range of results relating to the range of response levels. The disadvantage of this approach is the continued amount of time spent by the master processor for collecting the results from a single slave processor. Another slave processor may complete its task but needs to wait (and sit idle) for the master processor to complete its task of collecting data from another slave processor. The second alternative, which was implemented in the parallel algorithm, is for each processor to solve only a single parametric linear program at a time. This approach is considered to be *fine grained* since the solving of a single parametric linear program is the smallest unit of a task that can be performed by a slave processor. Consequently, the amount of time spent by the master processor collecting data from a slave processor is minimized.

- **Data dependency:** For the master-slave topology, the only data dependency is between the master process and each slave process. It is only necessary to present the slave processors with a subset of data through message passing, since all the other data are captured from the shared file system. The master process is dependent on the results generated by the slave processors.

5.6 Summary

The benefit of the parametric approach is the ability to display the estimated response more accurately without user intervention. The parametric approach also contributes to the optimization of speed considering that the

linear program is not resolved from scratch every time to obtain a range of optimal solutions.

Displaying a contour plot of the optimal estimates with two variables at the axis equips the model builder with a useful tool when working with state variables. Optimal estimates for a grid of values for the variables at the axis are obtained through the solving of parametric linear programs for a range of response levels. This entails that a solution is provided for all the other variables in the model for a given grid point. The model builder can, therefore, determine what the levels of the other variables should be to obtain an optimal response for the state variables fixed at some level.

The motivation for implementing a parallel algorithm for LRSA is driven by the demand for an interactive decision support system. The algorithm is applied to the problem of multi-variable response graphs.

An empirical investigation was conducted to evaluate the extensions discussed in this chapter. The investigation is presented in the next chapter.

Chapter 6

Empirical evaluation

6.1 Introduction

The properties of LRSA can best be observed by looking at the graphical results obtained when the method is applied to a data set. For this purpose an example from Draper and Smith [DS67] is considered. The variables in the data set are factors influencing water usage at a factory, which has been identified as the most important expense of the factory. By knowing more about the factors influencing water usage, suggestions can be made to decrease the use of water. This may consequently result in a decrease in operating cost.

The graphical results presented in this chapter were generated with the experimental software package *LRSA v1.0.* [1]

6.2 The data set

Table 6.1 lists the data collected over a period of 17 months. The following gives a short description of each variable in the table.

- **WATER** The average water usage for each month (the dependent variable).
- **TEMP** The average temperature for each month.
- **PROD** The production for each month.
- **DAYS** The number of working days in each month.

[1] The software is provided on CD. Please read the appendix for installation instructions.

WATER	TEMP	PROD	DAYS	EMPL
3067	58.8	7107	21	129
2828	65.2	6373	22	141
2891	70.9	6796	22	153
2994	77.4	9208	20	166
3082	79.3	14792	25	193
3898	81.0	14564	23	189
3502	71.9	11964	20	175
3060	63.9	13526	23	186
3211	54.5	12656	20	190
3286	39.5	14119	20	187
3542	44.5	16691	22	195
3125	43.6	14571	19	206
3022	56	13619	22	198
2922	64.7	14575	22	192
3950	73	14556	21	191
4488	78.9	18573	21	200
3295	79.4	15618	22	200

Table 6.1: Data on water usage from the example in [DS67].

- **EMPL** The number of employees for each month.

Table 6.2 gives a summary of the multiple regression results and Table 6.3 lists the variance inflation factor (*VIF*) for each of the variables.

6.3 Interpreting the graphical results

The response graphs for each of the independent variables are displayed in Figures 6.1, 6.2, 6.3 and 6.4.

An important feature to observe is that the response graph for the variable *PROD* (Figure 6.2) has the smallest average vertical distance between the minimum and maximum graph lines, indicating that the variable has the most significant influence on the response. This observation corresponds to the regression summary (Table 6.2) where *PROD* has the largest absolute t-value.

The issue of using linear models for optimization when multicollinearity is present, is addressed by the response graphs. From the regression summary it seems logical that the minimum water usage will be achieved by letting

Variable	R-Square=0.76 Adj.R-Square=0.68		
	Coefficient	t(32)	p-level
Intercpt	6360.33	4.83	0.0004
TEMP	13.86	2.68	0.0197
PROD	0.21	4.64	0.0005
DAYS	-126.69	-2.63	0.0216
EMPL	-21.81	-2.99	0.0111

Table 6.2: Regression summary.

Variable	R-Square	VIF
TEMP	0.20	1.25
PROD	0.84	6.65
DAYS	0.21	1.27
EMPL	0.84	6.62

Table 6.3: Multicollinearity summary.

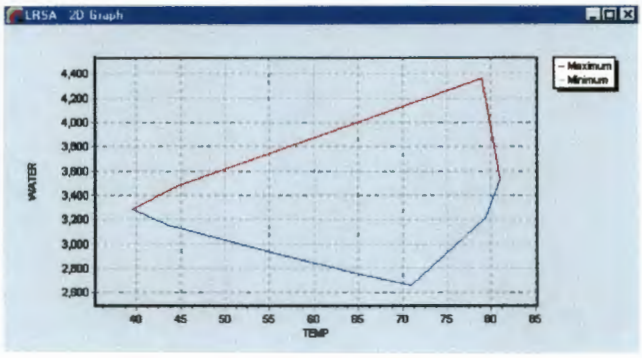

Figure 6.1: Response graph for variable *TEMP*.

the variables with negative coefficients take on high values, and the variables with positive coefficients take on low values. In particular, the variable *PROD* should be as low as possible (it has a positive coefficient) and *EMPL* should be as high as possible (it has a negative coefficient) in order to obtain the minimum water usage. From the multicollinearity information given in Table 6.3 one should, however, notice that the variables *PROD* and *EMPL* have a high *VIF*. The correlation matrix (Table 6.4) clarifies the interdepen-

	TEMP	PROD	DAYS	EMPL	WATER
TEMP	1.00	-.02	.43	-.08	.28
PROD	-.02	1.00	.10	.91	.63
DAYS	.43	.10	1.00	.03	-.08
EMPL	-.08	.91	.03	1.00	.41
WATER	.28	.63	-.08	.41	1.00

Table 6.4: Correlation matrix.

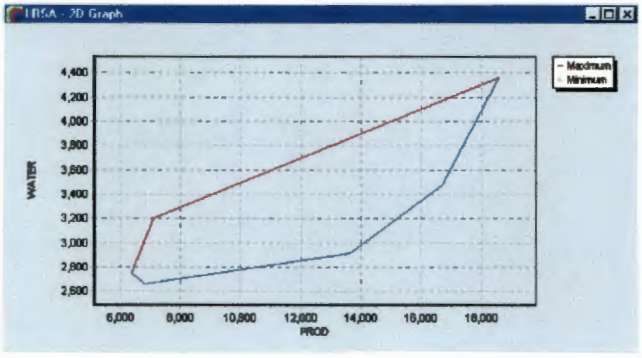

Figure 6.2: Response graph for variable *PROD*.

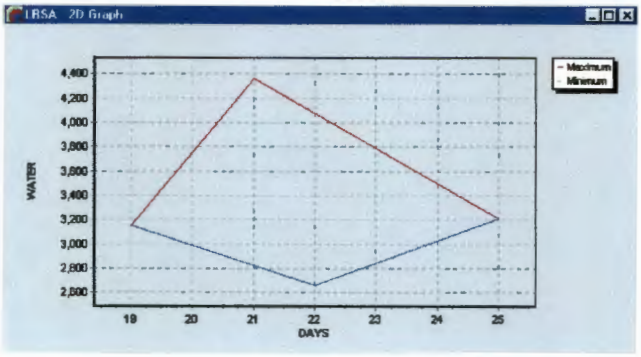

Figure 6.3: Response graph for variable *DAYS*.

dencies, showing that *PROD* and *EMPL* are positively correlated. Figure 6.5 shows a scatter plot of *PROD* vs. *EMPL* and indicates that there is no area of experience where *PROD* takes on low values and *EMPL* takes on high values (see the northwest corner of the scatter plot). Taking into account the experience with variable combinations in the past, the suggestion of setting

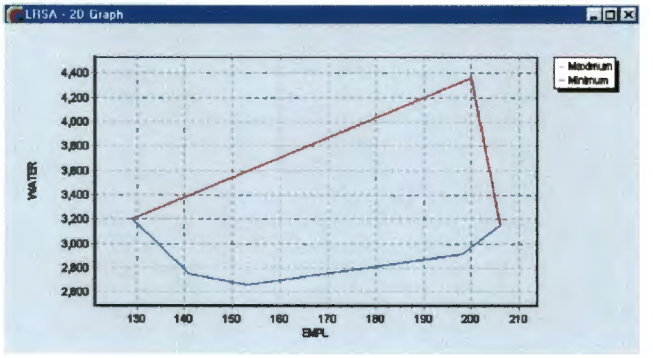

Figure 6.4: Response graph for variable *EMPL*.

PROD as low as possible and *EMPL* as high as possible will, therefore, not necessarily result in a feasible operating policy.

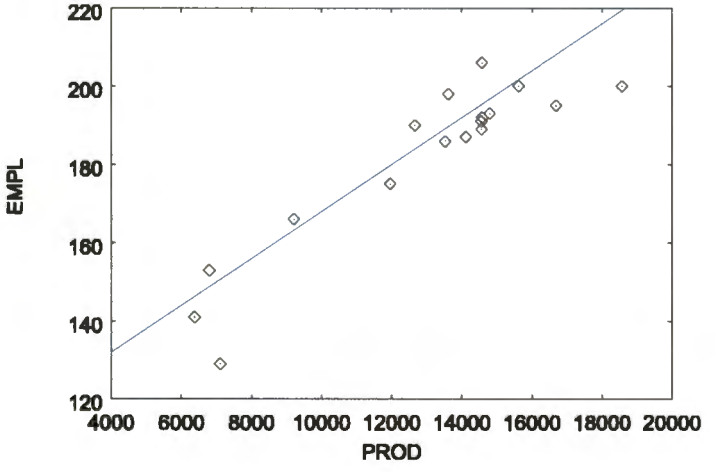

Figure 6.5: Scatter plot of *PROD* vs. *EMPL*.

The response graph for the variable *EMPL* (Figure 6.4) indeed shows that the minimum values for *WATER* are not found at high levels of *EMPL*. The minimum estimated response can be observed where *EMPL* is approximately equal to 153.

If a linear program is solved successfully (i.e the problem is not infeasible or unbounded) a solution for all the decision variables in the linear program is obtained. The *graph point tool* implemented as part of the LRSA software makes use of this property. The tool is activated by simply selecting a desired level for the variable under investigation from the response graph. The

tool then solves the LRSA problem (minimizing or maximizing the response depending on which response line is referenced) with the variable under investigation fixed at the preselected level. The results displayed by the tool is the solution for all the variables in the model for a minimum or maximum response.

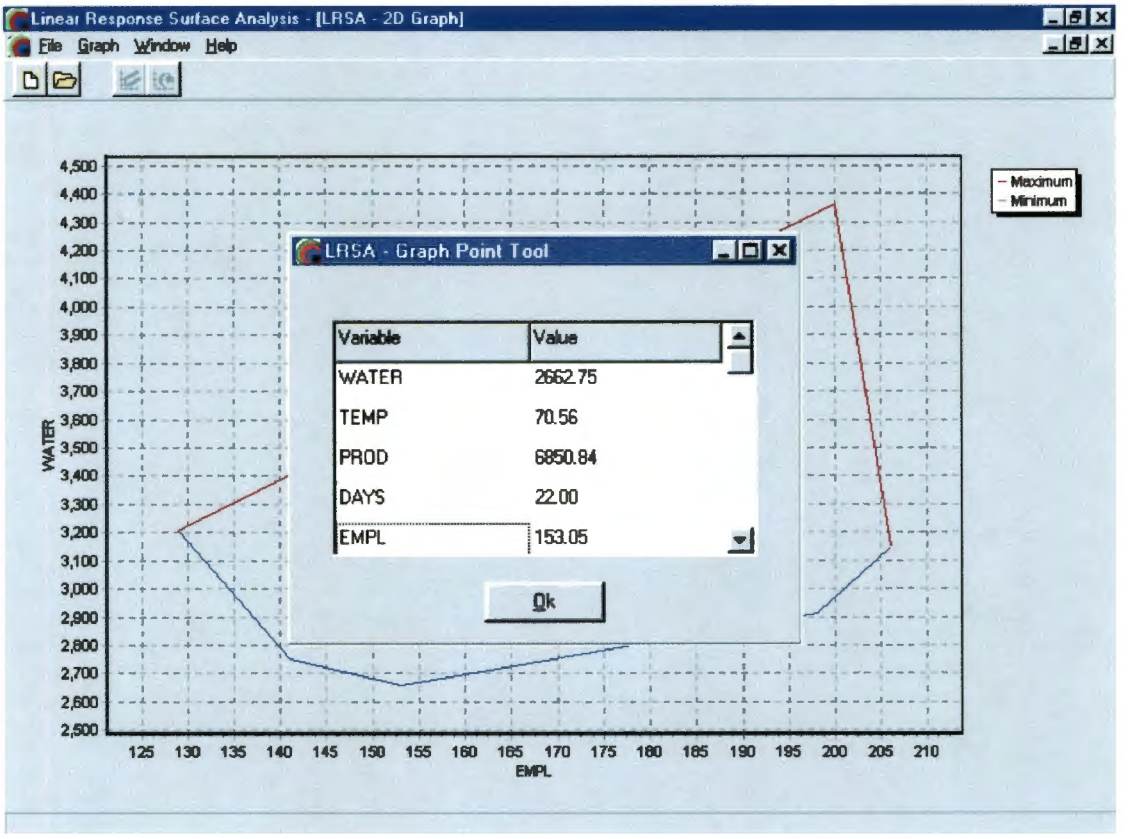

Figure 6.6: Results displayed by the *graph point tool.*

Figure 6.6 shows the results the graph point tool displays when the user wishes to obtain the optimum solution for all the variables in the model when the variable *EMPL* is fixed at a level of 153.05 (the level at which the minimum water usage is found).

6.4 Handling state variables

The LRSA software proposes levels for the independent variables that may result in the response attaining a minimum (or maximum). The proposed

levels will not always be practical. Consider the proposed optimal levels displayed in Figure 6.6 as an example. If the variable *TEMP* is considered to be a state variable then the proposed level of 70 may not be practical (e.g. climate conditions cannot be changed). This also applies to the proposed level of 22 for the variable *DAYS*. The number of working days per month cannot be adjusted on demand.

The multi-variable extension of LRSA is a useful tool in this regard. A response graph with two state variables at the axis will reveal the optimal response level for different combinations of the two variables. The graph point tool can be used on the response graph to determine what the levels of the remaining variables should be to obtain the optimal response level. Figure 6.7 shows the "minimum" response graph with the two variables *TEMP* and *DAYS* at the axis.

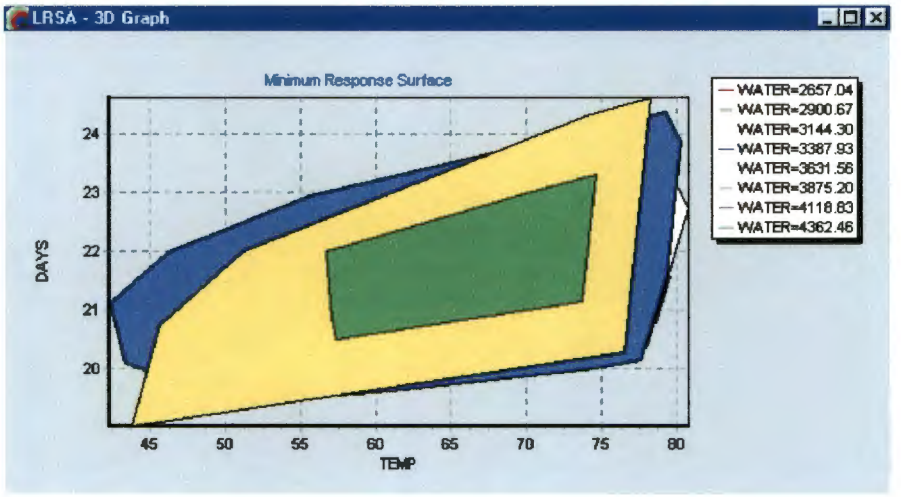

Figure 6.7: Response graph for variables *TEMP* and *DAYS*.

Consider a scenario where the two state variables *TEMP* and *DAYS* take on the values 50 and 20 respectively. From the response graph in Figure 6.7 it is clear that the point (50,20) maps onto the yellow coded response plane. This response plane represents a minimum water usage of 3144 (see legend on graph). This indicates that with the current scenario the minimum water usage that can be expected is in the interval [2900,3144], if the remaining variables in the model can be adjusted to their optimal levels. The optimal levels for the response and the remaining variables *PROD* and *EMPL* can be obtained through the graph point tool. The tool is activated by selecting the point (50,20) on the response graph. The optimal levels displayed for *WATER*, *PROD* and *EMPL* by the graph point tool are 3028, 11568 and

176 respectively. In conclusion, with the variables *TEMP* and *DAYS* at levels 50 and 20, a minimum water usage of 3028 can be expected if the levels of the variables *PROD* and *EMPL* can be adjusted to 11568 and 176 respectively.

6.5 Advantages of parametric programming

The figures presented in this chapter have all been generated using the parametric programming approach for selecting the intervals at the x-axis. Figure 6.8 shows the difference in graph shapes for the parametric approach compared with the approach where equally spaced intervals are considered.

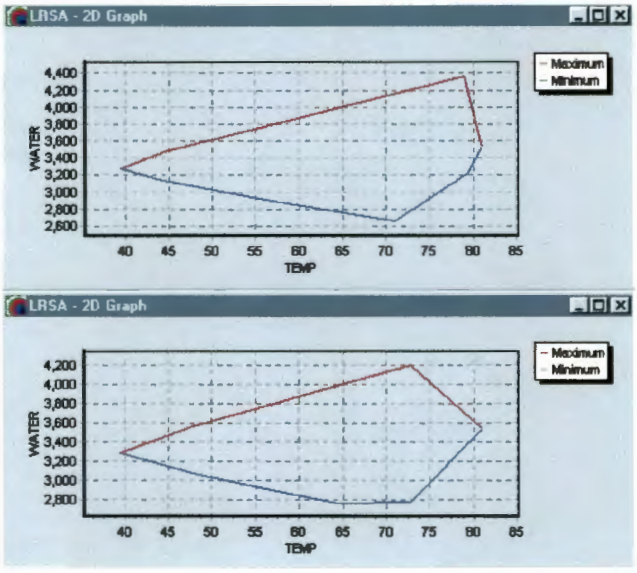

Figure 6.8: Differences in response when using parametric programming.

The graph displayed at the bottom of Figure 6.8 is generated with the number of equally spaced intervals as 6 (manual interval selection).

Clearly when too few intervals are used with manual interval selection, some information is lost. For instance, the graph at the top of Figure 6.8, generated with parametric interval selection, displays a minimum water usage of approximately 2600 where *TEMP* takes on the value of 71. For the manual interval selection case at the bottom of the figure, the minimum water usage level is shown to be higher (approximately 2800) without a prominent optimal point.

Figure 6.9 shows the results where the same experiment was performed considering multi-variable response graphs. The response graph at the top

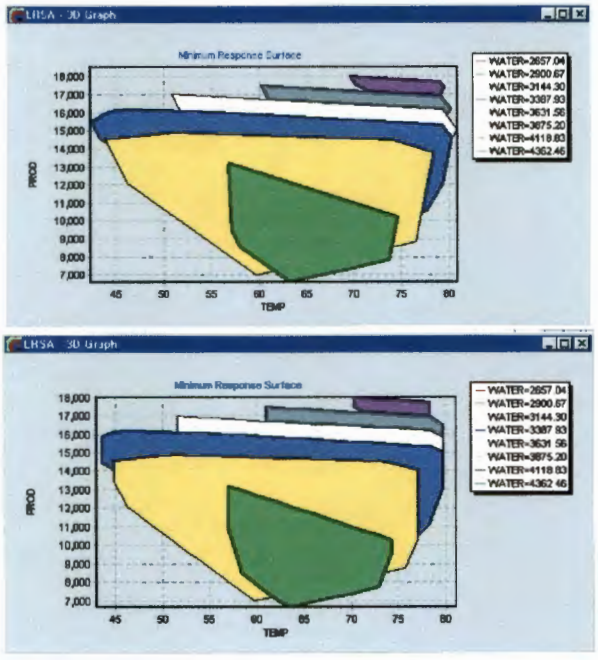

Figure 6.9: Differences in contours when using parametric programming.

of the figure is obtained by using the parametric approach to determine the intervals at the axis. The response graph generated by selecting the number of equally spaced intervals manually, which is the graph at the bottom of the figure, differs slightly. Some of the differences can be observed by looking at the edges of the response planes where the variable *TEMP* takes on higher values.

6.6 Evaluating the parallel LRSA algorithm

Experiments were conducted using the parallel version of the LRSA software, *Parallel LRSA v1.0*, in a UNIX operating environment.[2] This version of the software is command line driven and includes basic functionality like, for instance, the display of response graphs. The main purpose of the software is to report on the execution time for solving the LRSA problem (specifically

[2]The source code for *Parallel LRSA v1.0* is provided as part of the *LRSA v1.0* software installation.

Data set	# Variables
A	5
B	10
C	20

Table 6.5: Data sets used for parallel evaluation.

# Processors	Execution time (sec.)		
	Data set A	Data set B	Data set C
1	37	61	113
2	21	35	68
3	14	27	49
4	13	25	37
5	10	20	35

Table 6.6: Parallel execution times.

to generate graph points for a two dimensional contour graph) for a specified number of processors. Refer to Section 5.5.6 for an outline of the parallel algorithm implemented.

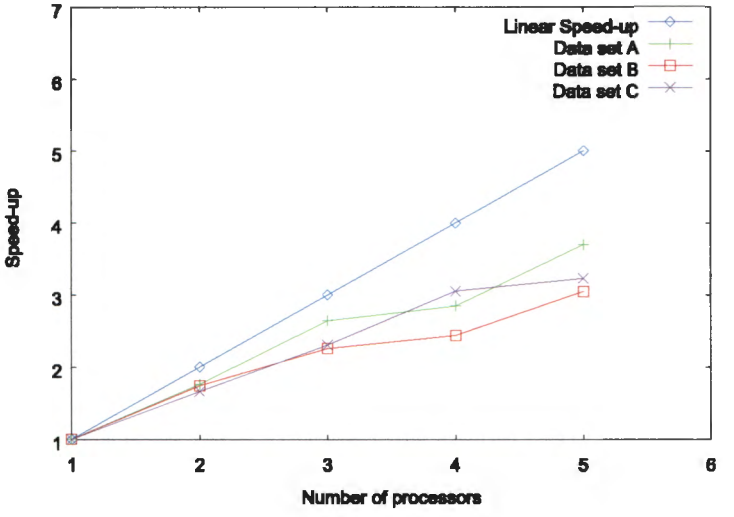

Figure 6.10: Speed-up for the parallel LRSA algorithm.

Table 6.5 lists the dimensions for each of the data sets used in the parallel experiments. Each data set had a total of 1500 cases.

Each experiment involved specifying the number of processors to be used and recording the execution time. A total of 6 processors were available for conducting the experiments. Note, however, that one of the processors was involved in a master process in the parallel algorithm and, therefore, only five processors were available to perform parallel slave tasks. The experiments where conducted by randomly identifying two variables from each data set that would appear on the axis of the two dimensional contour graphs. Furthermore, the number of response levels that had to be displayed by each contour graph was specified to be eight. The contour graphs are omitted from the discussion since we are only interested in the execution time.

Table 6.6 shows the results of the experiments conducted using the three data sets. On inspection it is evident that an increase in the number of processors reduces the execution time of the algorithm. A more appropriate performance measure is *speed-up* (see Section 5.5.5). Figure 6.10 shows the speed-up for the parallel algorithm for each of the three cases. The performance of the parallel algorithm in terms of speed-up does not differ significantly for the three data sets. This means that the algorithm is successful in maintaining a measure of scalability despite an increase in the dimension of the input data.

6.7 Variable selection

The absolute t-values for the variables in the Draper and Smith example (Table 6.2) are relatively high and the chances of a practitioner excluding some of the variables are very slim. In fact, a variable selection procedure applied to the data set will show that all the variables are needed in the model to obtain the best fit. In some situations practitioners would like to remove variables from a model on the grounds of poor t-values or as a result of a variable selection procedure. The response graphs for these variables may provide valuable information that may prevent the variables from being excluded from the model. Figure 6.11 shows a response graph (an example from Terblanche and Hattingh [TH99]) where the variable at the x-axis, *ILLIT*, denotes the illiteracy count of a country and where the response at the y-axis, *GNP*, denotes the gross national product per capita of the country. The t-value for the variable *ILLIT* is -1.314. For some practitioners such a low t-value is enough reason for excluding the variable from the model. From inspection we notice, however, that other variables in the model play a more important role in influencing the response for lower levels of *ILLIT* (see Terblanche and Hattingh [TH99]). This means that if *ILLIT* is low then the *GNP* is either high or low depending on the levels

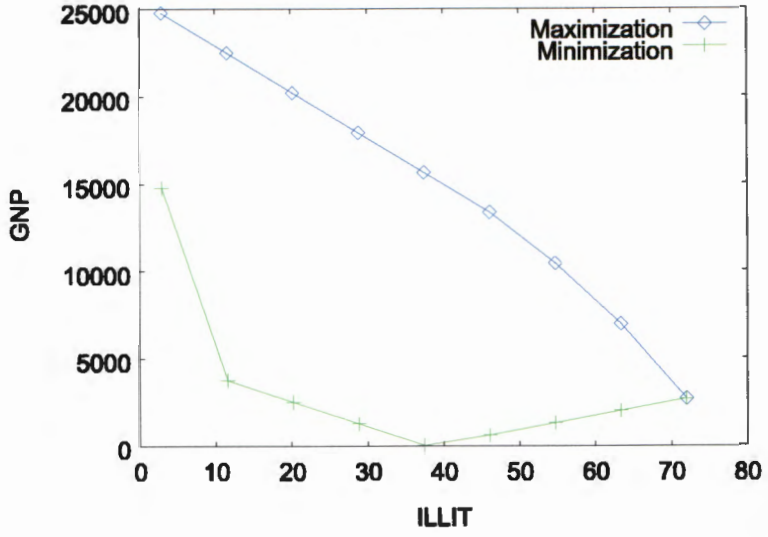

Figure 6.11: Response graph for variable *ILLIT*.

of the other factors. On the other hand if the level of *ILLIT* is high, then the other variables do not play an important role in influencing the response and there is not much difference in the maximum or the minimum response. Thus the data indicate a rather poor, almost fixed, gross national product for cases of high illiteracy.

6.8 Summary

In this chapter the response graphs for the Draper and Smith example were used to illustrate the properties of LRSA with regard to the following:

- Determining the importance of a variable to the response relative to the other variables in the model. The response graph for the variable *PROD* has the smallest average vertical distance between the minimum and maximum graph lines, indicating that *PROD* has a greater impact on the response than the other variables in the model.

- Predicting the optimal response levels, considering that interdependence exists among the independent variables. The coefficient for the variables *PROD* is positive and for the variable *EMPL* it is negative. This suggests that a decrease in water usage can be obtained by increasing the level of *EMPL* and decreasing the level of *PROD*. It is, however, shown that the two variables are correlated and the suggested

levels may be impractical. Since the response graphs only take the area of experience into account, the *graph point tool* proposes optimum levels for the two variables when applied to a minimum point residing on a minimum graph line).

- Handling of state variables. The variables *TEMP* and *DAYS* are considered to be state variables since they cannot be adjusted on demand. By placing these variables at the axis of a response graph the maximum or minimum response, with *TEMP* and *DAYS* fixed at some level, can be read from the graph. The graph point tool, applied to this fixed point on the graph, gives the solution values of the unconstrained variables in the model.

- Including or excluding variables from the model. The LRSA software is not a variable selection procedure. It rather provides valuable information that may assist the model builder in deciding what variables to include or exclude from the model. The t-values for the variables in the Draper and Smith example are relatively high and it is very unlikely that a practitioner will exclude any of them from the model. However, the example presented in Section 6.7 showed that a variable with a poor t-value may still remain in a model since the response graph for the variable reveals the importance of the other variables in the model with the said variable at different levels.

Empirical results were also presented for the LRSA extensions:

- Multi-variable extension. For purposes of illustrating the handling of state variables the multi-variable extension of LRSA was used. The two variables, *TEMP* and *DAYS*, were placed on the x-axis and y-axis respectively and a contour graph of the response *WATER* was generated.

- Parametric extension. A comparative example revealed that response graphs generated by using a parametric programming approach is more reliable since data are not omitted from the response graph as in the case of specifying intervals manually.

- Parallel extension. The results obtained from the experiments conducted on the parallel LRSA algorithm showed improved execution time when generating response graphs in the case of the multi-variable extension.

Chapter 7

Summary and Conclusions

7.1 Introduction

The use of linear models for optimization has become an important application. Response surface methodology provides the model builder with methods for determining optimal variable levels that will optimize the response. Unfortunately response surface methodology is only suitable for planned data since the process involves conducting experiments along the path of steepest ascent/descent and the experiments are conducted according to a specific design. In the case of raw data model builders alternatively use the sign and magnitude of the regression function coefficients for suggesting optimal levels for the independent variables. Due to complex interaction between the variables the estimated signs of the regression coefficients can be deceiving. The proposed combination of levels for the variables may, therefore, not be in the area of experience. LRSA produces response graphs that give optimal estimates of the response within the area of experience. The solutions presented by the response graphs save the model builder the effort of analyzing the complex interaction between the variables in order to find the expected optimal response levels.

The resulting response graphs are generated by solving linear programs iteratively. The formulation of a linear program entails constraining a specified variable at a level. The solution to the linear program gives the optimal estimates for the unconstrained variables in the model as well as the optimal estimated response. The benefit to the model builder is suggestions to improve the response with a state variable constrained at various levels.

The model builder is frequently faced with the challenge of reducing the dimension of the linear model. LRSA is not a variable selection procedure. It rather provides the model builder with valuable information to assist in

deciding which variables to include or exclude from the model. The vertical distances between the minimum and maximum response graphs for a model reveal such information as it indicates the importance of a variable relative to other variables in the model.

7.2 Contributions towards LRSA

The implementation of a parametric programming approach for managing interval selection has proven to be of much value, considering the correctness with which linear models and data are visualized when using this approach. It also reduces the amount of computational work since each linear program is not solved from scratch.

Contributing towards the robustness of the LRSA implementation is the use of a third-party simplex solver. The solver is implemented as part of the experimental software packages *LRSA v1.0* and *Parallel LRSA v1.0*.

Another exciting feature added to LRSA is the ability to investigate two variables and the contours of the expected response. This is very useful in situations where more than one state variable is present in the model. The resulting contour graph gives an optimal estimated response surface for a grid of values for the state variables. The graph point tool can be used to determine what the levels for the other variables in the model should be in order to obtain an optimal response for preselected values of the state variables.

A successful investigation into a parallel implementation of LRSA was conducted. The results showed that the parallel algorithm performs moderately well with data sets of up to 20 variables and 1500 cases. With multiprocessors readily available, LRSA can be implemented in parallel to ensure a highly interactive decision support system that can produce response graphs within reasonable time.

7.3 Recommendations and future work

The response graphs generated with the LRSA software reveal information about linear models and data that is complementary to regression analysis. For example the vertical distances between the maximum and minimum graphs correlates well with the t-statistic obtained through regression analysis. There is, however, no quantification of the information revealed by the vertical distances between the minimum and maximum graphs. It is only by comparing visually the vertical distances of various variables in the

model that we can determine the variable that has approximately the smallest average vertical distance between the minimum and maximum graphs. An approach to quantify the importance of a variable relative to the other variables in the model is to average vertical distances measured at regular intervals between the minimum and maximum graphs of a variable.

The ability of placing two independent variables at the axis of a response graph is of great benefit to the model builder. This is especially the case where the two variables are state variables. In many practical problems more than two state variables are present. It will be beneficial to a model builder if LRSA could produce response graphs showing the behaviour of the response for a larger number of state variables.

The formulation of LRSA is based on the assumption that the problem being investigated is linear in the objective function. Practical problems are more frequently considered to be non-linear. With some modification to the LRSA formulation response graphs of models with non-linear attributes can be displayed.

7.4 Conclusion

LRSA can successfully be used as a technique for visualizing linear models and data. The visual representations resulting from LRSA reflect properties of linear models and data that are useful to the model builder in performing the following activities:

- interpreting and optimizing linear models when interdependence among independent variables exists;
- determining the importance of a variable to a model relative to the other variables in the model;
- deciding on which variables to include or exclude from a linear model;
- handling of state variables in the case of optimization and forecasting.

The following poem (see Corlett [Cor64]) reflects some of the difficulties encountered in model building:

Ballade of multiple regression

If you want to deal best with your questions,
Use multi-regression techniques;
A computer can do in a minute
What, otherwise done, would take weeks.
For 'predictor selection' procedures
Will pick just the ones best for you
And provide the best-fitting equation
For the data you've fitted it to.

But did you collect the right data?
Were there 'glaring omissions' in yours?
Have the ones that score highly much meaning?
Can you tell the effect from the cause?
Are your 'cause' factors ones you can act on?
If not, you've got more work to do;
Your equation's as good - or as bad - as
The data you've fitted it to.

It is not suggested that LRSA replace linear regression analysis or any other statistical methods. It should rather be considered as an aid for resolving some of the model building difficulties.

Appendix A

Installing the software

A.1 *LRSA v1.0*

A.1.1 System Requirements

- Pentium 166MHz or higher.
- CD-ROM.
- Windows 95 Operating System.

A.1.2 Installation

1. Using Explorer open the root directory of the CD-ROM.
2. Install OSL V3 by executing the file /Installs/OSL V3/v3_osllib_win32.exe. This will copy setup files to a temporary directory.
3. From a DOS prompt in the temporary directory enter the command "setup trybuy". This will install a 60 day evaluation copy of OSL. For an academic license use "setup academic".
4. Copy the file /Installs/OSL V3/msvcrt.dll to the temporary directory if required by the system.
5. If experiencing any difficulty with the installation of OSL consult http://www6.software.IBM.com/sos/osl/optimization.htm.
6. Install *LRSA v1.0* by executing the file /Installs/LRSA V1.0/Setup.exe. The Setup program will install *LRSA v1.0* in the directory "c:/Program Files/Linear Response Surface Analysis" and create a menu item "LRSA" on the "Start-Programs" menu.

A.1.3 User Manuals

A user manual is supplied with the software and can be viewed by selecting the "Help" menu option after launching the program.

A.1.4 Source Code

LRSA has been developed using Borland C++ Builder 5. The source code and project file (lrsa.bpr) are included in /Source/LRSA V1.0 on the CD.

A.2 *Parallel LRSA v1.0*

A.2.1 System Requirements

- IBM SP/2 Multiprocessor.
- AIX 4.3 Operating System.
- Parallel Operating Environment (POE).
- Optimization Subroutine Library (OSL).
- GNUPLOT v3.6 (graphing software).

A.2.2 Installation

1. The software POE ,OSL and GNUPLOT need to be installed prior to installing *Parallel LRSA v1.0.*
2. Copy all the files from /Source/LRSA V1.0(parallel) on the CD-ROM to the SP/2 file system.
3. Build the executable with "make LRSA_Parallel".

A.2.3 User Manuals

A user manual in ascii format is supplied. See /Source/LRSA V1.0(parallel)/Manual.txt.

Bibliography

[AC96] A.A Afifi and V. Clark. *Computer-aided multivariate analysis.* Chapman & Hall, London, third edition edition, 1996.

[Aki89] S.G. Aki. *The design and analysis of parallel algorithms.* Prentice Hall, 1989.

[BH85] P.J.S. Bruwer and J.M. Hattingh. Constrained regression models for optimization and forecasting. *ORION*, 1(1):2–15, 1985.

[BJS90] M.S. Bazaraa, J.J. Jarvis, and H.D. Sherali. *Linear programming and network flows.* John Wiley & Sons, second edition, 1990.

[Bru83] P.J.S. Bruwer. Evaluating the performance of computer-based information systems using a restricted linear regression model. *Quaestiones Informaticae*, 2(3):1–6, 1983.

[Chv83] V. Chvatal. *Linear programming.* W.H. Freeman and Company, 1983.

[Cor64] T. Corlett. Ballade of multiple regression. *Applied statistics*, 12(3):145, 1964.

[CT96] A. Chalmers and J. Tidmus. *Practical parallel processing.* International Thomson Computer Press, 1996.

[CZ97] Y. Censor and S.A Zenios. *Parallel optimization.* Oxford University Press, 1997.

[Dan63] G.B. Dantzig. *Linear programming and extensions.* Princeton University Press, 1963.

[Dev95] M.H. Van Deventer. Lineêre respons oppervlak ontleding: 'n stelsel vir besluitnemingsondersteuning. M.Sc. dissertation, Potchefstroom University for CHE, 1995.

[DJT93] Jordaan D.B., Hattingh J.M., and Steyn T. Die versoening van konflikterende doelwitte in die databasis omgewing met behulp van regressie metodes. *S.A. Rekenaar Tydskrif*, (10), 1993.

[DS67] N.R. Draper and H. Smith. *Applied regression analysis.* John Wiley & Sons, Inc., New York, 1967.

[EMD67] Beale E.M.L., Kendall M.G., and Mann D.W. The discarding of variables in multivariate analysis. *Biometrika*, 54(3 and 4):357, 1967.

[Fly72] M.J. Flynn. Some computer organisations and their effectiveness. *IEEE Transactions on Computers*, 21(9):948–960, 1972.

[Fos95] I. Foster. *Designing and building parallel programs.* Addison Wesley, 1995.

[IBM96a] IBM. *IBM Parallel Environment for AIX: MPI Programming and Subroutine Reference*, 1996. Document number GC23-3894-01.

[IBM96b] IBM. *IBM Parallel Environment for AIX: Operation and Use, Volume 1*, 1996. Document number SC28-1979-00.

[IBM96c] IBM. *PVMe for AIX, User's Guide and Subroutine Reference*, 1996. Document number GC23-3884-01.

[KC87] A.I. Khuri and J.A. Cornell. *Response surfaces: Designs and analysis.* Marcel Dekker Inc., 1987.

[MM95] R.H. Myers and D.C. Montgomery. *Response surface methodology.* John Wiley & Sons, Inc., New York, 1995.

[MP92] D.C. Montgomery and E.A. Peck. *Introduction to linear regression analysis.* John Wiley & Sons, Inc., second edition, 1992.

[Ray97] T.P. Rayn. *Modern regression methods.* John Wiley & Sons, Inc., 1997.

[Rou94] T.P. Roux. 'n rekenaar gebaseerde stelsel om kwantifiseerbare aspekte van sosio-ekonomiese en sosio-politiese faktore van lande te ontleed. M.Comm. dissertation, Potchefstroom University for CHE, South Africa, 1994.

[TH99] S.E. Terblanche and J.M. Hattingh. Linear response surface analysis as a technique for visualizing linear models and data. In E. Kozan, editor, *Operations Research: From theory to real life*, volume 2, page 1257, Brisbane Australia, July 1999. Australian Society for Operations Research Incorporating, Queensland University of Technology Printery.

[WW81] T.H. Wonnacott and R.J. Wonnacott. *Regression: A second course in statistics.* John Wiley & Sons, Inc., 1981.