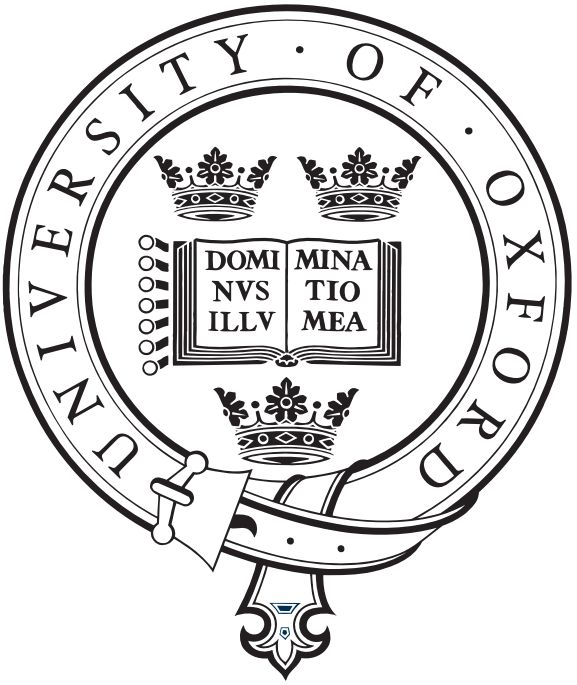
Linear Programming Algorithms for Detecting Separated Data in Binary Logistic Regression Models

Kjell Konis
Worcester College
University of Oxford



A thesis submitted for the degree of Doctor of Philosophy in Statistics

Hilary Term 2007

Above all I would like to thank my supervisor, Professor Brian D. Ripley, for his help and suggestions. Additionally, I am indebted to the Department of Statistics at the University of Oxford for providing a Teaching Assistant Bursary without which I would not have been able to carry out this research. Finally, I would like to thank the numerous others who helped me enjoy the research that lead to this thesis.

Linear Programming Algorithms for Detecting Separated Data in Binary Logistic Regression Models

Kjell Konis, Worcester College

DPhil Thesis, Department of Statistics

Trinity Term 2007

This thesis is a study of the detection of separation among the sample points in binary logistic regression models. We propose a new algorithm for detecting separation and demonstrate empirically that it can be computed fast enough to be used routinely as part of the fitting process for logistic regression models.

The parameter estimates of a binary logistic regression model fit using the method of maximum likelihood sometimes do not converge to finite values. This phenomenon (also known as *monotone likelihood* or *infinite parameters*) occurs because of a condition among the sample points known as *separation*. There are two classes of separation. When *complete separation* is present among the sample points, iterative procedures for maximizing the likelihood tend to break down, when it would be clear that there is a problem with the model. However, when *quasicomplete separation* is present among the sample points, the iterative procedures for maximizing the likelihood tend to satisfy their convergence criterion before revealing any indication of separation.

The new algorithm is based on a linear program with a nonnegative objective function that has a positive optimal value when separation is present among the sample points. We compare several approaches for solving this linear program and find that a method based on determining the feasibility of the dual to this linear program provides a numerically reliable test for separation among the sample points. A simulation study shows that this test can be computed in a similar amount of time as fitting the binary logistic regression model using the method of iteratively reweighted least squares: hence the test is fast enough to be used routinely as part of the fitting procedure.

An implementation of our algorithm (as well as the other methods described in this thesis) is available in the R package `safeBinaryRegression`.

Contents

Chapter 1

Binary Logistic Regression and Separated Data

1.1 Introduction

We consider the problem of fitting a binary logistic regression model to a set of n sample points each characterized by a binary response y_i and a vector x_i of $p \geq 1$ covariates. The model is assumed to contain an intercept term hence $x_{i1} = 1$ for $i = 1, \ldots, n$. *Separation* is said to exist among the sample points when there is a hyperplane H in the space of the covariates such that the sample points with $y_i = 0$ lie on one side of H and the sample points with $y_i = 1$ lie on the other. Sample points that lie in the separating hyperplane H may have either $y_i = 0$ or $y_i = 1$. The practical concern is that when separation is present among the sample points, estimation of the binary logistic regression model parameter vector using the method of maximum likelihood is not reliable.

Consider a binary logistic regression model fit to a single discrete covariate x with

levels l_1 through l_m. Suppose there is a level l_k, $1 \leq k \leq m$, such that $y_i = 1$ for every sample point i with $x_i = l_k$. Then $x = l_k$ is a perfect predictor of success. The fitted probability of any sample point with $x = l_k$ should therefore be 1 and this is achieved by taking the parameter associated with l_k to be infinite. This is a consequence of separation. The occurrence of separation is therefore related to the problem of determining the existence of the maximum likelihood estimate for the binary logistic regression model which has been considered in Haberman (1974, chap. 2), Wedderburn (1976), and Silvapulle (1981) among others.

This thesis is primarily concerned with the detection of separation among the sample points. When $p \leq 3$ separation can be easily detected using graphical methods. However, for models containing more than three covariates, the presence of separation among the sample points can go unnoticed since the iterative procedures commonly used to maximize the likelihood (e.g., iteratively reweighted least squares, Newton's method) may satisfy their convergence criterion before any trouble fitting the model is encountered.

The main result of this thesis is a numerically reliable test for separation among the sample points that can be computed fast enough to be used routinely as part of the fitting procedure for binary logistic regression models. Our test is based on a linear program with a nonnegative valued objective function that has a positive optimal value when separation is present among the sample points and is zero otherwise. We show by simulation that computing our test requires roughly the same amount of time as fitting the binary logistic regression model using the method of iteratively reweighted least squares; hence it is fast enough for routine use. An implementation of our test is available in the R (R Development Core Team, 2007) package `safeBinaryRegression` and an example of its use can be found in appendix A.

Linear programming techniques have been proposed in the literature (Silvapulle and Burridge, 1986; Clarkson and Jennrich, 1991) for the purpose of separation detection. These methods perform a preliminary transformation on the linear program to eliminate free decision variables. However, when these transformations are computed in double precision arithmetic, sufficient rounding error can accumulate that the boundedness of the feasible region is affected leading to an incorrect result. On the other hand, the dual to our proposed linear program is already in the standard form so no preliminary transformation is required. Hence we have found that it provides a more reliable test for separation among the sample points than existing linear programming based methods.

The remainder of this chapter is organized as follows. A statement of the binary logistic regression model considered is given in section 1.2. Section 1.3 defines the two classes of separation and gives arguments for the nonexistence of the maximum likelihood estimate of the binary logistic regression model parameter vector when separation is present. Section 1.4 briefly explains how to extend the notion of separation to multinomial logistic regression models and section 1.5 examines the similarity between the concept of linear separation in the pattern recognition literature and the two classes of separation described here.

In chapter 2 we provide examples of the behavior of both Newton's method and the method of iteratively reweighted least squares when used to maximize the likelihood for a binary logistic regression model fit to a set of separated sample points. In both cases the iterative methods satisfy their convergence criterion before any problems are encountered fitting the model.

In chapter 3 we consider the application of several existing techniques to the problem of detecting separation among the sample points. We begin with techniques specifi-

cally for the purpose of separation detection: the heuristic and algebraic approaches given in Albert and Anderson (1984) and the mixed integer linear program given in Santner and Duffy (1986). We then consider two pattern classification methods that can be implemented using linear programming. Finally, we describe the conditions given in Barndorff-Nielsen (1978) and Jacobsen (1989) for the existence of the maximum likelihood estimate of the binary logistic regression model parameter vector and propose algorithms for their verification.

Chapter 4 contains a description of our proposed linear program and the test for separation among the sample points based on its dual. Other methods for solving the linear program are described as well. Lastly, the performance of the proposed methods is investigated in a simulation study.

1.2 The Binary Logistic Model

Cox (1970) offers a good introduction to the analysis of binary data using logistic models and the bibliographic notes at the end of chapter 1 in the second edition (Cox and Snell, 1989) contain a discussion of work before 1970 on binary data and the logistic function. Nelder and Wedderburn (1972) showed that the binary logistic models developed in Cox (1970) could be viewed in the framework of generalized linear models and the parameters estimated using the method of iteratively reweighted least squares. We adopt this approach since it is used to fit binary logistic regression models in R (R Development Core Team, 2007) and S-PLUS® (Insightful, 2005), the environments for which we are developing a test for separation.

We use McCullagh and Nelder (1989) as a background reference for binary logistic regression models in the generalized linear models framework. The binary logistic

model supposes that, for each observation or sample point in the data, the response Y can take only one of two possible values which are denoted by 0 for failure and 1 for success. The relations

$$P(Y_i = 1) = \pi_i \quad \text{and} \quad P(Y_i = 0) = 1 - \pi_i \tag{1.1}$$

respectively give the probabilities of success and failure for the i^{th} sample point. Additionally, each sample point is associated with a vector of covariates or explanatory variables $x_i = (1, x_{i2}, \ldots, x_{ip})^T$. The structural component of the logistic model sets the logit of the probability of success equal to a linear combination of the covariates

$$\text{logit}(\pi_i) = \log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta^T x_i = \beta_1 + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} \tag{1.2}$$

where the model parameter vector $\beta = (\beta_1, \ldots, \beta_p)^T$ is a vector of p elements containing the model parameters which are to be estimated.

Often binary data is summarized in contingency tables or by covariate class. Since data in either of these formats can alternatively be represented as a set of binary sample points where each sample point describes the outcome of a single experimental trial (McCullagh and Nelder, 1989, chap. 4), we consider only this representation.

The response vector $y = (y_1, \ldots, y_n)^T$ contains the observed binary outcomes of n independent random variables $Y_1, \ldots, Y_n$ where $Y_i \sim \text{binomial}(1, \pi_i)$. The joint probability of the sample $y_1, \ldots, y_n$ is

$$P(Y_1 = y_1, \ldots, Y_n = y_n; \pi) = \prod_{i=1}^{n} \pi_i^{y_i}(1-\pi_i)^{(1-y_i)}.$$

where $\pi = (\pi_1, \ldots, \pi_n)^T$. The likelihood of π given the sample is

$$L(\pi; y_1, \ldots, y_n) = \prod_{i=1}^{n} \pi_i^{y_i} (1 - \pi_i)^{(1-y_i)}$$

and the log likelihood is given by

$$l(\pi; y) = \sum_{i=1}^{n} \big[y_i \log(\pi_i) + (1 - y_i) \log(1 - \pi_i) \big]. \tag{1.3}$$

The probability of success π_i for the i^{th} sample point is then modeled as a function of the covariate vector x_i. Let X be the $n \times p$ matrix with i^{th} row x_i^T for $i = 1, \ldots, n$. The matrix X is assumed to be of full rank, i.e., $\text{rank}(X) = p$. Also, since $x_{i1} = 1$ for $i = 1, \ldots, n$, the first column of X is identically one. The log likelihood function for the model parameter vector β can be obtained by substituting equation (1.2) into equation (1.3) giving

$$l(\beta; y, X) = \sum_{i=1}^{n} \sum_{j=1}^{p} y_i x_{ij} \beta_j - \sum_{i=1}^{n} \log\big(1 + \exp \sum_{j=1}^{p} x_{ij} \beta_j\big) \tag{1.4}$$

where x_{ij} is the element in the i^{th} row and j^{th} column of the design matrix X. Fitting the model by maximum likelihood then involves finding $\hat{\beta}$ such that when $\beta = \hat{\beta}$ equation (1.4) is maximized. There is no known closed-form approach for maximizing the log likelihood in the general case so iterative methods are used. McCullagh and Nelder (1989) describe how to compute $\hat{\beta}$ using iteratively reweighted least squares (IRLS). For data in contingency tables, iterative proportional fitting can be used (e.g., Bishop et al., 1975, chap. 3). Alternatively, $\hat{\beta}$ may be computed by direct maximization of the log likelihood function, for instance by using Newton's method.

Two common uses for binary logistic models are classification and regression. For classification, the quantity of interest is the mean-value parameter π_i. The simplest classification rule labels a sample point x_i a success if $\pi_i > 1/2$ and a failure if $\pi_i < 1/2$. Since π_i represents the probability of success for the i^{th} sample point, $\pi_i \in [0, 1]$. On the other hand, the logistic regression model assumes that inference on the vector of model parameters β is of interest. Since one does not generally allow infinite model parameters or covariates in regression, the linear combination $x_i^T\beta$ in equation (1.2) – the linear predictor in the terminology of generalized linear models – may be assumed finite. However, this assumption is at odds with the binary logistic classification model since it implies $\pi_i = \text{logit}^{-1}(x_i^T\beta) \in (0, 1)$.

In a regression context Haberman (1974, chap. 2) gives general necessary and sufficient conditions for the existence of the maximum likelihood estimate where, in his terminology, existence means finiteness of the parameter vector estimate (we use this definition as well). Silvapulle (1981) shows that a certain degree of overlap between the successes and the failures is a necessary and sufficient condition for the existence of the maximum likelihood estimate of the binary logistic regression model parameter vector β. Albert and Anderson (1984) build on Silvapulle's results by showing that the existence of the maximum likelihood estimate depends on the configuration of the sample points. We feel that the approach taken by Albert and Anderson is both the easiest to understand and easiest to implement and hence use it as the foundation for this thesis.

1.3 Separated Data

Albert and Anderson (1984) showed that a set of n sample points can be classified

into one of three mutually exclusive configurations: *completely separated*, *quasi-completely separated*, and *overlapped*. Additionally, we use the term *separated* to describe a set of sample points that belong to either the completely or quasicompletely separated configurations. When there is overlap among the sample points there is a unique maximum likelihood estimate of the model parameter vector β that lies in the interior of the parameter space. When there is complete separation among the sample points, estimation of β using the principle of maximum likelihood leads to a non-unique solution on the boundary of the parameter space. When there is quasicomplete separation among the sample points, estimation of β by maximum likelihood yields a possibly non-unique solution on the boundary of the parameter space. Since we are interested primarily in the binary logistic regression model, we say that the maximum likelihood estimate of the parameter vector β does not exist when the solution lies on the boundary of the parameter space. That is, when one or more of the elements of the maximum likelihood estimate of β is nonfinite.

These three configurations of the sample points are defined in this section. Additionally, for the cases of complete separation and quasicomplete separation, the arguments for the nonexistence of the maximum likelihood estimate given in Albert and Anderson (1984) and augmented by the corrections suggested in Santner and Duffy (1986) are presented.

1.3.1 Complete Separation

There is *complete separation* among a set of n sample points if there is a vector β such that

$$\beta^T x_i < 0 \quad \text{when} \quad y_i = 0 \quad \text{and} \quad \beta^T x_i > 0 \quad \text{when} \quad y_i = 1 \tag{1.5}$$

for $i = 1, \ldots, n$. That is, any vector β giving complete separation correctly predicts the response y_i given x_i for each of the n sample points. To simplify the statement of complete separation let $\tilde{y}$ be a vector of n elements such that $\tilde{y}_i = 1$ when $y_i = 1$ and $\tilde{y}_i = -1$ when $y_i = 0$ for $i = 1, \ldots, n$ and define $\bar{X} = \text{diag}(\tilde{y})\, X$. Then a vector β gives complete separation among the sample points if

$$\bar{X}\beta > 0$$

where the inequality is interpreted element-wise.

Let E_1 be the set of indices i such that $y_i = 0$ and E_2 be the set of indices i such that $y_i = 1$. If there is complete separation between the sample points then there exists a hyperplane of dimension $p - 1$ dividing $\mathbb{R}^p$ into two halfspaces such that x_i lies strictly in one halfspace when $i \in E_1$ and strictly in the other when $i \in E_2$. This configuration is illustrated for $p = 3$ (the intercept term is not shown) in figure 1.1.

Let A^c be the set of all vectors β satisfying (1.5), then A^c is a convex cone in $\mathbb{R}^p$. Additionally, if $\beta \in A^c$ then $\beta + \Delta \in A^c$ where $\Delta \neq k\beta$ for any real $k > 0$ is a perturbation chosen so that $\beta + \Delta$ satisfies (1.5). It follows that if there is complete separation among the sample points then A^c contains a continuum of vectors β satisfying (1.5).

The following theorem is the first of the two main results in Albert and Anderson (1984) adapted to the special case of a binary response. The theorem demonstrates that the maximum likelihood estimate does not exist (in our sense) when there is complete separation between the sample points.

Theorem 1. *If there is complete separation among the sample points, the maximum*

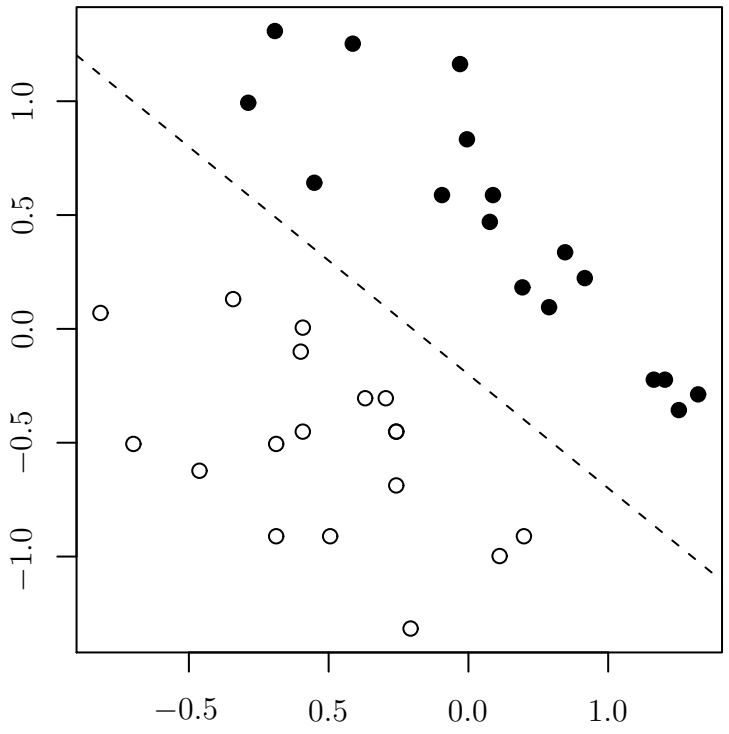


Figure 1.1: Complete Separation: the sample points in E_1 are represented by open circles and the sample points in E_2 by filled circles.

likelihood estimate $\hat{\beta}$ *does not exist , and*

$$\sup_{\beta \in \mathbb{R}^p} L(\beta; y, X) = 1.$$

Proof. Let $\beta(k) = k\beta$ for $\beta \in A^c$ and for real $k > 0$; consider the log likelihood as the function of $\beta(k)$

$$\log L(\beta(k); \bar{X}) = \sum_{i=1}^{n} \log \frac{1}{1 + e^{-k\beta^T \bar{x}_i}} \tag{1.6}$$

where $\bar{x}_i^T = \tilde{y}_i x_i^T$ is the i^{th} row of $\bar{X}$. Since $\beta \in A^c$ gives complete separation among

the sample points, $\beta^T \bar{x}_i > 0$ for $i = 1, \ldots, n$. The exponent in the denominator in equation (1.6) is therefore strictly negative for each term. Consider the behavior of equation (1.6) in the limit as $k \to \infty$. The exponential term in the denominator tends toward zero so that the right hand side of equation (1.6) tends to a sum of n logs of one. It follows that equation (1.6) tends to zero, its absolute maximum, as k approaches infinity. Thus the log likelihood is maximum and equal to zero on the boundary of the parameter space. Further, for any finite β, the log likelihood is strictly negative since the exponential terms in the denominator of (1.6) are strictly positive. Hence the maximum likelihood estimate $\hat{\beta}$ of the logistic regression model parameter vector β does not exist when there is complete separation among the sample points. □

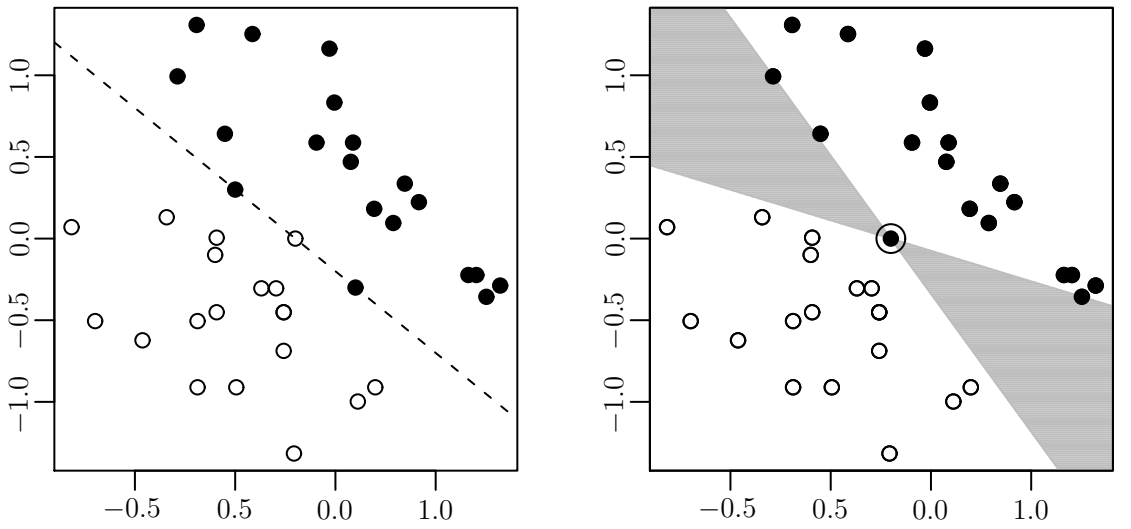
The proof of theorem 1 demonstrates that the maximum of the likelihood can be achieved by extending any vector $\beta \in A^c$ to infinity hence there is a continuum of points on the boundary of the parameter space where the likelihood is maximized and equal to one since there is a continuum of vectors $\beta \in A^c$.

1.3.2 Quasicomplete Separation

If there is no vector β that completely separates the n sample points then another configuration of separation may occur. A vector $\beta \neq 0$ is said to give *quasicomplete separation* among the sample points if

$$\beta^T x_i \leq 0 \quad \text{when} \quad y_i = 0 \quad \text{and} \quad \beta^T x_i \geq 0 \quad \text{when} \quad y_i = 1 \tag{1.7}$$

with equality holding for at least one $i \in \{1, \ldots, n\}$. This definition differs slightly from the one given by Albert and Anderson in that it specifically excludes the zero



(a) Quasicomplete separation – $\text{rank}(X^q) = p - 1$: the separating hyperplane is unique. There are 3 points lying in the separating hyperplane.

(b) Quasicomplete separation – $\text{rank}(X^q) < p-1$: there is a continuum of separating hyperplanes; any line lying in the shaded area gives quasicomplete separation among the sample points.

Figure 1.2: Two possible configurations of quasicomplete separation.

vector which trivially quasiseparates any set of sample points (Santner and Duffy, 1986).

Let $Q(\beta)$ be the set of indices satisfying equality in (1.7). The sample points x_i for $i \in Q(\beta)$ are said to be *quasiseparated* by β. Assume that the set $Q(\beta)$ contains $r > 0$ elements and let X^q be the $r \times p$ matrix of quasiseparated observations. By definition $X^q\beta = 0$ so that the rows of X^q are linearly dependent; the r sample points thus belong to an affine space of dimension $\text{rank}(X^q) < p$. If $\text{rank}(X^q) = p-1$ then the separating hyperplane is unique. When $\text{rank}(X^q) < p - 1$ there is a continuum of hyperplanes containing the r points. This distinction is illustrated in figure 1.2.

Let A^q be the set of all vectors β giving quasicomplete separation of the sample points.

Lemma 1. *For any set of sample points for which $A^c = \emptyset$ and for which A^q is not*

empty

1. *A^q is a convex cone;*

2. *there is a minimal set Q^m such that $Q^m \subset Q(\beta)$ for all $\beta \in A^q$, and $A^{mq} = \{\beta : \beta \in A^q$ and $Q(\beta) = Q^m\}$ is not empty.*

Proof.

1. If β satisfies equation (1.7) with equality holding for at least one of the n sample points, so does $k\beta$ for any real number $k > 0$. Let α and β be elements of A^q and define $\gamma = \lambda\alpha + \mu\beta$ with $\lambda, \mu > 0$ and such that $\lambda + \mu = 1$. Then γ also satisfies (1.7). Further, γ must achieve equality in (1.7) for at least one $i \in \{1, \ldots, n\}$ otherwise γ gives complete separation which is excluded by assumption.

2. Let $i \in Q(\gamma)$, then x_i satisfies

$$\lambda\alpha^T x_i + \mu\beta^T x_i = 0. \tag{1.8}$$

Since $\lambda > 0$, $\mu > 0$, $\alpha \in A^q$ and $\beta \in A^q$, both terms on the left hand side of (1.8) have the same sign. Hence (1.8) is only satisfied if each term is zero which implies that $i \in Q(\alpha)$ and $i \in Q(\beta)$. In other words, $Q(\gamma) = Q(\alpha) \cap Q(\beta)$. Thus the class $E^q = \{Q(\beta) : \beta \in A^q\}$ is closed under intersection. Let

$$Q^m = \bigcap_{\beta \in A^q} Q(\beta).$$

By the closure property, Q^m is an element of E^q thus, by the definition of E^q, there is $\beta^* \in A^q$ such that $Q(\beta^*) = Q^m$. A^{mq} is not empty since $\beta^* \in A^{mq}$.

□

The second result from Albert and Anderson (1984) is presented, again for a binary response, in theorem 2. It shows that the maximum likelihood estimate does not exist when there is quasicomplete separation among the sample points and that the likelihood is bounded above by a value strictly less than one.

Theorem 2. *If there is quasicomplete separation among the sample points ($A^c = \emptyset$, $A^q \neq \emptyset$), then*

1. *the maximum likelihood estimate $\hat{\beta}$ does not exist;*
2. $$\sup_{\beta \in \mathbb{R}^p} L(X, \beta) = \sup_{\gamma \in A^q} L(X^{mq}, \gamma) < 1,$$
   where X^{mq} is the matrix of quasiseparated points corresponding to the minimal set Q^m.

Proof.

1. Choose a fixed $\beta \in A^q$. For any $\alpha \in \mathbb{R}^p$ let $\alpha^*(k) = \alpha + k\beta$ and consider the limit
   $$\lim_{k \to \infty} l(\alpha^*(k); y, X).$$
   where $l(\alpha^*(k); y, X) = \log L(\alpha^*(k); y, X)$. The derivative of $l(\alpha^*(k); y, X)$ is
   $$\begin{aligned} \frac{d}{dk} l(\alpha^*(k); y, X) &= \frac{d}{dk} \sum_{i=1}^{n} \log \frac{1}{1 + e^{-(\alpha + k\beta)^T \bar{x}_i}} \\ &= \sum_{i=1}^{n} \beta^T \bar{x}_i \frac{e^{\alpha + k\beta^T \bar{x}_i}}{1 + e^{\alpha + k\beta^T \bar{x}_i}} \end{aligned}$$
   which is greater than or equal to zero since $\beta^T \bar{x}_i \geq 0$ for $i = 1, \ldots, n$. Further,

because X is assumed to be of full rank there is at least one positive term; hence $l(\alpha^*(k); y, X)$ is strictly increasing in k. Let $\alpha^b = \lim_{k\to\infty} \alpha^*(k)$. Then $L(X, \alpha) = L(X, \alpha^*(0)) < L(X, \alpha^b)$ for any $\alpha \in \mathbb{R}^p$. Thus the maximum likelihood estimate does not exist.

2. By lemma 1 there is a minimal set of quasiseparated sample points Q^m, let β be any element of A^{mq}. Then, for any $\alpha \in \mathbb{R}^p$

$$\begin{aligned} l(\alpha; y, X) &\leq \sup_{k\in[0,\infty)} l(\alpha + k\beta; y, X) \\ &\leq \sup_{k\in[0,\infty)} \left[\sum_{i\in Q^m} \log \frac{1}{1 + e^{-\alpha^T \bar{x}_i} e^{-k\beta^T \bar{x}_i}} \right. \\ &\qquad \left. + \sum_{i\in E\setminus Q^m} \log \frac{1}{1 + e^{-\alpha^T \bar{x}_i} e^{-k\beta^T \bar{x}_i}} \right] \end{aligned} \tag{1.9}$$

where $E = \{1, \ldots, n\}$. Since $\beta^T \bar{x}_i > 0$ for $i \in E \setminus Q^m$, the summation over the sample points in $E \setminus Q^m$ vanishes in the supremum. Thus we have

$$l(\alpha; y, X) \leq \sum_{i\in Q^m} \log \frac{1}{1 + e^{\alpha^T \bar{x}_i}} = l(\alpha; y^{mq}, X^{mq})$$

where y^{mq} is the vector with elements y_i and X^{mq} is the matrix with rows x_i^T such that $i \in Q^m$. Taking the supremum

$$\sup_{\alpha\in\mathbb{R}^p} l(\alpha; y, X) = \sup_{\alpha\in\mathbb{R}^p} \left[\sup_{k\in[0,\infty)} l(\alpha + k\beta; y, X) \right] = \sup_{\alpha\in\mathbb{R}^p} l(\alpha; y^{mq}, X^{mq}). \tag{1.10}$$

The right-hand side of equation 1.10 is constant for all $\alpha \in A^q$. Lastly, the set $\{\alpha : \alpha^T \bar{x} > 0 \text{ for all } i \in Q^m\} = \emptyset$ so that equation 1.10 is strictly less than

zero. It follows then that

$$\sup_{\alpha \in A^q} L(\alpha; y^{mq}, X^{mq}) < 1.$$

□

1.3.3 Overlap

If the sample points are neither completely separated nor quasicompletely separated then they are said to *overlap*. That is for any vector β either

$$\beta^T x_i \geq 0 \text{ for at least one } i \in E_1 \quad \text{or} \quad \beta^T x_i \geq 0 \text{ for at least one } i \in E_2. \tag{1.11}$$

An illustration of overlap is shown in figure 1.3.

Silvapulle (1981) shows that, when there is overlap among the sample points, the maximum likelihood estimate of the parameter vector β exists and is unique. The theorem assumes that the n sample points have been ordered so that $y_i = 1$ for $i = 1, \ldots, r$ and $y_i = 0$ for $i = r+1, \ldots, n$. Let

$$S = \left\{ \sum_{i=1}^{r} k_i x_i \mid k_i > 0 \right\} \quad \text{and} \quad F = \left\{ \sum_{i=r+1}^{n} k_i x_i \mid k_i > 0 \right\}$$

be the relative interiors of the convex cones generated by $x_1, \ldots, x_r$ and $x_{r+1}, \ldots, x_n$ respectively and let G be a distribution function relating the probability of success to the linear predictor:

$$P(Y_i = 1) = G(x_i^T \beta).$$

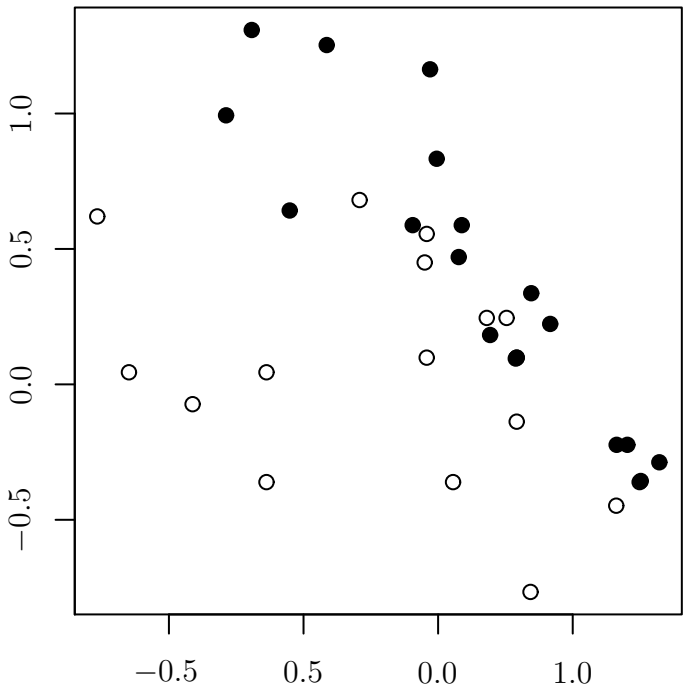


Figure 1.3: Overlapped Data: there is no hyperplane separating the failures from the successes.

Theorem 3. *Let the condition* Π *be defined by*

$$\Pi\text{: } S \cap F \neq \emptyset \text{ or one of } S,\ F \text{ is } \mathbb{R}^p\ (\emptyset = \text{empty set}).$$

1. *The mle* $\hat{\beta}$ *of* β *exists and the minimum set* $\{\hat{\beta}\}$ *is bounded only when* Π *is satisfied.*

2. *Suppose that* $l(\beta)$ *is a proper closed convex function on* $\mathbb{R}^p$*. Then the mle* $\hat{\beta}$ *exists and the minimum set* $\{\hat{\beta}\}$ *is bounded if and only if* Π *is satisfied.*

3. *Suppose that* $-\log G$ *and* $-\log\{1-G\}$ *are convex and* $x_{i1} = 1$ *for every* i*. Then* $\hat{\beta}$ *exists and the minimum set* $\{\hat{\beta}\}$ *is bounded if and only if* $S \cap F \neq \emptyset$*. Let us further assume that* G *is strictly increasing at every* t *satisfying* $0 < G(t) < 1$*. Then* $\hat{\beta}$ *is uniquely defined if and only if* $S \cap F \neq \emptyset$*.*

The proof is given in Silvapulle (1981).

In the binary logistic regression model, G is the inverse function of the logit link; that is $G(x) = e^x/(1 + e^x)$. Since

$$\frac{\partial^2}{\partial x^2}\left[-\log G(x)\right] = \frac{\partial^2}{\partial x^2}\left[-\log(1 - G(x))\right] = \frac{e^x}{(1 + e^x)^2} > 0$$

both $-\log G$ and $-\log\{1 - G\}$ are convex and since

$$\frac{\partial}{\partial x}G(x) = \frac{e^x}{(1 + e^x)^2} > 0,$$

G is strictly increasing for all x. The model is assumed to include an intercept term thus it follows from part 3 of theorem 3 that the maximum likelihood estimate $\hat{\beta}$ is uniquely defined if and only if $S \cap F \neq \emptyset$. Hence, for the logistic regression model, the maximum likelihood estimate exists and is unique if and only if there is overlap among the sample points.

1.3.4 Discussion

It is important to note that the presence of either complete or quasicomplete separation among the sample points depends on both the data and the model. Separation is therefore a statistical issue, not merely an artifact of the data. With continuous covariates separation can always be achieved by increasing the number of terms in the model (e.g., including higher-order and interaction terms). Hence the presence of separation may only indicate that the model is overfitting the data. Further, choosing a different model — discriminant analysis for example — can avoid the problems arising from separation altogether.

1.4 Extension to Multinomial Logistic Models

For brevity, the results presented in this thesis are for binary logistic models. However, they may be extended in a straight forward way to apply to multinomial logistic models as well. This approach is taken in Albert and Lesaffre (1986) and in Albert and Anderson (1984). Albert and Anderson, for example, take the conditional probabilities of group membership to have the extended logistic form (Anderson, 1972)

$$P(H = h|x) = \exp(\beta_h^T x) \cdot P(H = 1|x) \quad h = 2, \ldots, g,$$
$$P(H = 1|x) = \left[1 + \sum_{t=2}^{g} \exp(\beta_t^T x)\right]^{-1} \tag{1.12}$$

where g is the number of groups, H is the indicator of group membership taking values $1, \ldots, g$ and

$$\beta_h = (\beta_{h1}, \ldots, \beta_{hp})^T \quad \text{for} \quad h = 2, \ldots, p, \quad \text{and} \quad \beta_1 \equiv 0.$$

In the logistic classification model, the simplest classification rule becomes to allocate a sample point x to group $H = h$ if and only if

$$(\beta_h - \beta_t)^T x > 0$$

for $t = 1, \ldots, g$ except $t = h$. The model parameter vector is $\beta = (\beta_1^T, \beta_2^T, \ldots, \beta_g^T)^T$.

A vector β gives complete separation among a set of n multinomial sample points if for all $j, t = 1, \ldots, g$ $(j \neq t)$

$$(\beta_j - \beta_t)^T x_i > 0$$

holds for each $i \in E_j$. In other words, a vector β gives complete separation among the sample points if it allocates each of the n sample points to its correct group. When such a vector exists, the sample points are said to be *completely separable*.

If the sample points are not completely separable, then a vector β gives quasicomplete separation among a set of n multinomial sample points if for all $j, t = 1, \ldots, g$ $(j \neq t)$

$$(\beta_j - \beta_t)^T x_i \geq 0 \tag{1.13}$$

holds for each $i \in E_j$ and with equality for at least one (i, j, t) triplet. The sample points for which equality holds in (1.13) are said to be quasiseparated by β.

Let X_j be the matrix with rows x_i^T such that $i \in E_j$. Define the $(g-1) \times g$ block matrix $\bar{X}_j$ to have blocks X_j in each element of column j, blocks $-X_j$ in row t and column t for $t < j$ and in row $t-1$ and column t for $t > j$ and to be zero otherwise. For example, in the case of three groups $\bar{X}_2$ is given by

$$\bar{X}_2 = \begin{bmatrix} -X_2 & X_2 & 0 \\ 0 & X_2 & -X_2 \end{bmatrix}.$$

The matrix $\bar{X}_j$ is designed so that for a fixed j if $\bar{X}_j \beta \geq 0$ then (1.13) is satisfied for that particular value of j as well. If we let

$$\bar{X} = \begin{bmatrix} \bar{X}_1 \\ \vdots \\ \bar{X}_g \end{bmatrix}$$

then $\bar{X}\beta \geq 0$ implies that β satisfies (1.13). If all of the inequality relations are strictly satisfied then β completely separates the sample points. Since β_1 is by

definition zero, neither β_1 nor the corresponding columns of $\bar{X}$ need to be stored for computational purposes. Thus $\bar{X}$ can be stored in an $n\,(g-1) \times p\,(g-1)$ matrix.

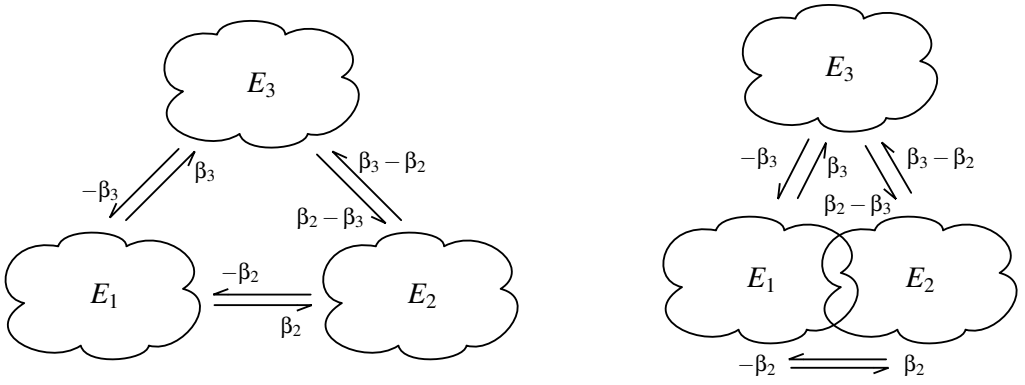
1.5 The Relationship Between Separated Data and Linear Separability

The concept of linear separability has been studied extensively in the context of pattern recognition and classification. Ripley (1996, chap. 3) for example, defines two groups of sample points to be linearly separable if there is a linear combination of the covariates $x^T\beta$ that is positive for the sample points in one group and negative for the sample points in the other. A function that computes a linear combination of the covariates and returns the sign is known as a *perceptron* (Rosenblatt, 1957). At first it may seem that the concept of linear separability is quite similar to the concept of separated data. However, the fact that the linear separability problem seeks to classify the two groups as either linearly separable or linearly inseparable while while the data separation problem has three possible outcomes turns out to be an important distinction.

For computational reasons the problem of finding linear separability is to find β such that

$$\bar{X}\beta \geq \delta \tag{1.14}$$

for some $\delta > 0$. The parameter δ specifies the width of the so-called dead-zone — a neighborhood of the separating hyperplane H in which no sample points are permitted to lie. The dead-zone is essentially the same as the geometrical margin in a support vector learning algorithm (Schölkopf and Smola, 2002, chap. 7).



(a) The vector $\beta = (0^T, \beta_2^T, \beta_3^T)^T$ with $\beta_2 \neq 0$ and $\beta_3 \neq 0$ gives linear separability between the three groups of sample points and complete separation among the sample points.

(b) The vector $\beta = (0^T, \beta_2^T, \beta_3^T)^T$ with $\beta_2 = 0$ and $\beta_3 \neq 0$ gives quasicomplete separation among the sample points. Linear separability is not possible.

Figure 1.4: Linear separability between $g \geq 3$ groups implies pairwise linear separability between the groups. However, quasicomplete separation among the sample points does not imply pairwise separation.

Clearly any vector β satisfying (1.14) also satisfies the definition of complete separation. On the other hand, if there is quasicomplete separation, then the sample points are not linearly separable since $\beta^T x_i = 0$ must hold for at least one $i \in \{1, \ldots, n\}$; hence the quasiseparated sample points will always lie in the dead-zone. Linear separability is therefore a sufficient condition for the nonexistence of the maximum likelihood estimate of the underlying logistic regression model. Conversely, linear inseparability is a necessary but not sufficient condition for the existence of the maximum likelihood estimate since quasicomplete separation is still a possibility.

When there are two groups, the difference between linear separability and separation among the sample points is small enough that methods for detecting linear separability may yield some insight into whether separation exists among the sample points. One such method is considered in chapter 3. However, when there are more than two groups, the distinction between the two definitions becomes more important. Smith (1969) extends the definition of linear separability to $g > 2$ groups

as the existence of a set of set of vectors w_j, $j = 1, \ldots, g$, satisfying the $g(g-1)$ inequalities

$$X_i w_i - X_i w_j \geq \delta \tag{1.15}$$

for $i = 1, \ldots, g$ and $j = 1, \ldots, g$ with $i \neq j$. It is clear from (1.15) the presence of linear separability between g groups implies complete separation among the sample points. An example is shown in figure 1.4(a). Alternatively, consider the example shown in figure 1.4(b). The three sets of sample points are not only not linearly separable but there is overlap between the sample points with indices in E_1 and E_2. However, there is a vector $\beta = (0^T, \beta_2^T, \beta_3^T)^T$ with $\beta_2 = 0$ and $\beta_3 \neq 0$ that gives quasicomplete separation. The sample points in E_3 satisfy (1.13) strictly while equality holds for the sample points in E_1 and E_2. This situation is known as *partial separation*. A thorough discussion of partial separation can be found in Lesaffre and Albert (1989).

Chapter 2

Effects of Separation

The motivation to develop methods for separation detection and, more generally, the study of sufficient conditions for the existence of the maximum likelihood estimate of the parameter vector in the logistic regression model, arose because of problems encountered when using iterative methods to maximize the likelihood (Silvapulle, 1981; Silvapulle and Burridge, 1986). The authors suggest that, in the event problems with the iterative procedure are encountered — for instance, failure to converge before the maximum number of iterations is reached or the occurrence of numerical breakdown — check for separation among the sample points. This seems a reasonable approach for completely separated configurations of the sample points. Anderson (1972) showed that if complete separation exists among the sample points (equivalently, when the successes and failures are linearly separable) then any convergent method for maximizing the likelihood will yield a solution revealing the complete separation. However, as we will demonstrate in this chapter, Anderson's result is far less useful for detecting quasicomplete separation. In fact, even for trivially small models, iterative methods (in particular we consider New-

ton's method and the method of iteratively reweighted least squares) for estimating the logistic regression model parameter vector can satisfy their convergence criteria before any indication of separation is given.

The occurrence of separation among the sample points is often attributed to small data sets or small to medium-size data sets containing several unbalanced and highly predictive risk factors. Heinze and Schemper (2002), for example, go so far as to summarize the probability of separation for models of various sizes. We believe however that insufficient attention has been paid to the case of quasicomplete separation, most likely because its presence rarely causes problems for common fitting techniques and hence goes unnoticed. In particular, the occurrence of quasicomplete separation may be relatively common in models that contain multiple factor variables, especially when interaction terms are included.

2.1 The Effect of Quasicomplete Separation on Model Fitting

To demonstrate the effect of quasicomplete separation on both Newton's method and the method of iteratively reweighted least squares (IRLS), we consider fitting a binary logistic regression model — using each of these algorithms — to the four quasicompletely separated sample points shown in (2.1). Note that, in the case of logistic regression, both Newton's method and IRLS compute the same sequence of steps through the parameter space given identical starting points. The algorithms differ in the choice of starting points and the stopping criteria. The natural approach when using Newton's method is to directly maximize the log-likelihood while IRLS is generally used to minimize the model deviance (SAS Institute Inc.,

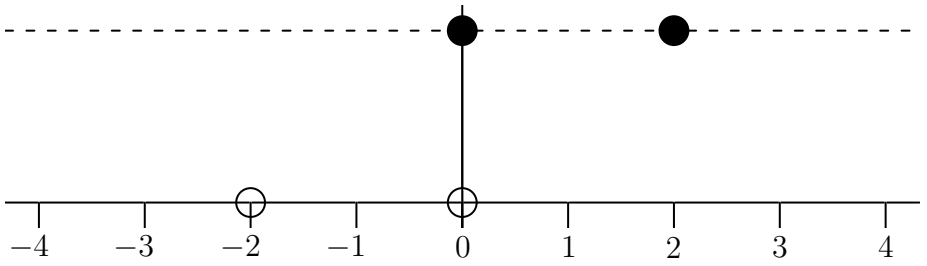


Figure 2.1: Quasicomplete Separation. Since the intercept term is identically 1 it is omitted from the plot. Points in E_1 ($y_i = 0$) are represented by open circles and points in E_2 ($y_i = 1$) by filled circles.

2004; R Development Core Team, 2007). We demonstrate that, when quasicomplete separation is present among the sample points, numerical convergence occurs in both these criteria while the parameter estimates diverge.

The vector $\beta = (0, 1)^T$ gives quasicomplete separation among the four sample points

$$y = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \quad X = \begin{bmatrix} 1 & -2 \\ 1 & 0 \\ 1 & 0 \\ 1 & 2 \end{bmatrix} \tag{2.1}$$

which are displayed in figure 2.1.

2.1.1 Newton's Method

Newton's method has been widely used to maximize the likelihood in logistic regression models, see for example Haberman (1974) or McCullagh and Nelder (1989). We implemented Newton's method following the description given in Süli and Mayers (2003). A discussion of the convergence of Newton's method can be found in this source as well. The origin was used for the starting point. Table 2.1 contains

the iteration trace when Newton's method was used to maximize the log-likelihood arising from a binary logistic regression model fit to the sample points in (2.1).

iteration	log-likelihood	β	$\|\beta\|_2$
start	-2.77258872	$(0.0, 0.0)$	0.00000000
1	-1.64015038	$(0.0, 1.00000000)$	1.00000000
2	-1.47142683	$(0.0, 1.56766764)$	1.56766764
3	-1.41669525	$(0.0, 2.08941022)$	2.08941022
$\vdots$	$\vdots$	$\vdots$	$\vdots$
17	-1.38629439	$(0.0, 9.10144738)$	9.10144738
18	-1.38629437	$(0.0, 9.60144738)$	9.60144738
19	-1.38629436	$(0.0, 10.10144739)$	10.10144739
20	-1.38629436	$(0.0, 10.60144739)$	10.60144739

Table 2.1: An iteration trace of Newton's method for the quasicompletely separated sample points in (2.1).

Since the goal is to maximize the log-likelihood, convergence is declared when the relative change in the value of the log-likelihood is suitably small. That is, Newton's method is declared to have converged when

$$\frac{l(\beta^{(k+1)}; y, X) - l(\beta^{(k)}; y, X)}{a + l(\beta^{(k)}; y, X)} < \epsilon \tag{2.2}$$

for a given $\epsilon > 0$ and where a is a positive constant included to prevent division by zero and $\beta^{(k)}$ is the computed β after k iterations.

In table 2.1 we see that, after 18 iterations, the log-likelihood has converged to 8 significant digits while the 2-norm of the parameter vector β is still increasing in steps of size on the order of 0.5. The computed estimate $\hat{\beta}$ is therefore quite sensitive to the parameter ϵ. However, since the convergence criteria has been satisfied, the algorithm returns the final computed β with no indication that quasicomplete separation is present.

2.1.2 Iteratively Reweighted Least Squares

We implemented the IRLS algorithm following the description given chapter 4 of McCullagh and Nelder (1989) using the data-generated starting point (which can be seen in the first row of table 2.2). Since the logistic regression model is an instance of a generalized linear model, several software programs, in particular R (R Development Core Team, 2007) and S-PLUS® (Insightful, 2005), use IRLS for the model fitting. Both of these software packages use the relative change in the model deviance as an indicator of convergence. SAS® (SAS Institute Inc., 2003) also uses IRLS to fit logistic regression models and uses either relative change in the value of the log-likelihood function or relative change in the gradient of the log-likelihood function as the measure of convergence. Table 2.2 contains the iteration trace when IRLS was used to maximize the log-likelihood arising from a binary logistic regression model fit to the sample points in (2.1).

iteration	deviance	β	$\|\beta\|_2$
start	NA	$(0.0, 0.66666667)^T$	0.66666667
1	3.70843882	$(0.0, 1.29846523)^T$	1.29846523
2	3.06001759	$(0.0, 1.83571619)^T$	1.83571619
3	2.87307589	$(0.0, 2.34843619)^T$	2.34843619
⋮	⋮	⋮	⋮
17	2.77258880	$(0.0, 9.35562515)^T$	9.35562515
18	2.77258875	$(0.0, 9.85562516)^T$	9.85562516
19	2.77258873	$(0.0, 10.35562516)^T$	10.35562516

Table 2.2: IRLS iteration trace for the quasicompletely separated sample points in (2.1).

Table 2.2 shows that the IRLS method has essentially the same behavior as Newton's method: after 19 iterations, IRLS has achieved convergence to 8 significant digits in the model deviance. Hence the algorithm will return the final computed β with

no indication that quasicomplete separation is present among the sample points.

2.1.3 A Simple Heuristic Test for Separation Among the Sample Points

Perhaps the simplest possible test for quasicomplete separation among the sample points is to insist that the iterative method used to maximize the likelihood terminate only after both the value of the likelihood function and the parameter vector β have converged. That is (in the case of Newton's method), require that

$$\frac{\|\beta^{(k+1)} - \beta^{(k)}\|_2}{a + \|\beta^{(k)}\|_2} < \epsilon \tag{2.3}$$

hold in addition to the condition given in equation (2.2). The $\|\beta^{(k)}\|_2$ term in the denominator is necessary to make the stopping criteria independent of the scaling of the variables in the model. It has been our observation that $\|\beta^{(k)}\|_2$ increases roughly linearly in k while $\|\beta^{(k+1)} - \beta^{(k)}\|_2$ remains relatively constant so that, for large enough k, equation (2.3) will be satisfied. However, such values of k would be far beyond any sensible maximum number of iterations.

The convergence in the log-likelihood while at least one parameter diverges to $\pm\infty$ was noted in Heinze and Schemper (2002) as a symptom of separation but curiously the authors did not mention its use as a test for separation.

2.1.4 The Hauck-Donner Phenomenon

Hauck and Donner (1977) show that standard errors and t values computed using the Wald approximation to the log-likelihood can be misleading for estimated

parameters that are large in absolute terms. The condition is known as the Hauck-Donner phenomenon (Venables and Ripley, 1999). For i such that $|\hat{\beta}_i|$ is large, the curvature of the log-likelihood function can be much smaller at $\hat{\beta}_i$ than at $\beta_i = 0$ (under the null hypothesis). When this occurs, the Wald approximation underestimates the change in the log-likelihood upon setting $\beta_i = 0$. This happens in such a way that $t_i \to 0$ as $|\hat{\beta}_i| \to \infty$. Thus, highly significant parameters (via the likelihood ratio test), including parameters revealing separation among the sample points, may have nonsignificant t values. The t values returned by many software packages (R Development Core Team, 2007; Insightful, 2005; SAS Institute Inc., 2003) are computed using the Wald approximation.

A simple heuristic check for separation can be carried out by computing and comparing the p values using the Wald approximation and a likelihood ratio test for parameters large in absolute terms. However, such a check would be computationally demanding as it would require fitting the model $p + 1$ times where p is the number of parameters to be estimated.

2.2 Convergent Models for Separated Data

When either complete or quasicomplete separation exists among the sample points inference is still possible. Heinze and Schemper (2002), for example, show that the penalized likelihood employed by the bias reduction procedure in Firth (1993) yields convergent estimates when separation is present among the sample points. Other methods include exact logistic regression (Mehta and Patel, 1995) and extended maximum likelihood estimates (Clarkson and Jennrich, 1991). However, it is our opinion that the majority of users will only resort to these methods when they are

aware that separation is present in their data.

Chapter 3

Existing Methods for Detecting Separated Data and Linear Separation

This chapter considers several methods proposed in the literature which may be used to detect separated data. With the exception of the empirical method given in Albert and Anderson (1984) which is fully implemented in SAS® (SAS Institute Inc., 2003) and partially implemented in several other statistical software programs including R (R Development Core Team, 2007) and S-PLUS® (Insightful, 2005), to our knowledge none of these methods are implemented for use with binary logistic regression models.

In section 3.1 we describe the empirical and algebraic approaches suggested in Albert and Anderson (1984) as well as the mixed integer program suggested in Santner and Duffy (1986). These methods attempt specifically to classify a given set of sample points as either completely separated, quasicompletely separated or overlapped.

Section 3.2 describes two methods typical of those proposed in the machine learning literature for detecting linear separation and comments on their usefulness for identifying separated data. Finally, section 3.3 considers the conditions given in Barndorff-Nielsen (1978) and Jacobsen (1989) for the existence of the maximum likelihood estimate of the binary logistic regression model parameter vector. These conditions can be verified by computing convex hulls. Since the maximum likelihood estimate of the binary logistic regression model parameter vector exists only for overlapped data, these methods can also be used to classify a set of sample points as separated or overlapped.

3.1 Methods for the Detection and Classification of Separation among the Sample Points

3.1.1 The Empirical Approach Suggested in Albert and Anderson

Albert and Anderson (1984) suggested algorithms for detecting complete and quasi-complete separation among the sample points. The algorithm for detecting complete separation is based on the result from Anderson (1972) that any convergent method for maximizing the log likelihood

$$l(\beta; y, X) = \sum_{i=1}^{n} \sum_{j=0}^{p} y_i x_{ij} \beta_j - \sum_{i=1}^{n} \log\big(1 + \exp \sum_{j=0}^{p} x_{ij} \beta_j\big) \tag{3.1}$$

yields a solution giving complete separation if such a solution exists. The method involves inserting a stopping rule into the iterative algorithm used to maximize

equation (3.1). After the k^{th} iteration let $\hat{\beta}^{(k)}$ be the current iterative approximation to $\hat{\beta}$. If $x_i^T\hat{\beta}^{(k)} < 0$ for $i \in E_1$ and $x_i^T\hat{\beta}^{(k)} > 0$ for $i \in E_2$ then the current approximation $\hat{\beta}^{(k)}$ gives complete separation among the sample points. Albert and Anderson (1984) implemented this rule and claim that it is very effective.

The empirical algorithm for the detection of quasicomplete separation depends on the observation that, as the iterative procedure diverges, there is at least one sample point for which the probability of correct allocation tends rapidly towards one. During each iteration $k \geq K$, compute the probability of correct allocation for each sample point: $\text{pr}_i^m(\hat{\beta}^{(k)}) = \text{logit}^{-1}(x_i^T\hat{\beta}^{(k)})$ for $i \in E_2$ and $\text{pr}_i^m(\hat{\beta}^{(k)}) = 1 - \text{logit}^{-1}(x_i^T\hat{\beta}^{(k)})$ for $i \in E_1$. Let $\text{pr}^m(\hat{\beta}^{(k)}) = \max_i \text{pr}_i^m(\hat{\beta}^{(k)})$. If, for some suitable choice of ϵ,

$$\text{pr}^m(\hat{\beta}^{(k)}) > \min\{1-\epsilon, \text{pr}^m(\hat{\beta}^{(k-1)})\} \tag{3.2}$$

issue a warning that there may be quasicomplete separation among the sample points and continue with the iterations. Albert and Anderson (1984) recommend choosing K to be around 7 or 8 to avoid early false warnings.

In the event that this algorithm does issue a warning, there is (at least) one sample point x_i for which the probability of correct allocation is very close to one. There are two possible reasons for such an occurrence.

Case 1: There is overlap among the sample points and the sample point x_i is an atypical observation in its own group and far from the mean. In this case, the warning is unnecessary and the iterative algorithm should be allowed to continue until the maximum of the log likelihood is reached.

Case 2: There is quasicomplete separation among the sample points and the sample point x_i belongs to the set of completely separated points ($x_i \in \bar{Q}$). In

this situation, the asymptotic dispersion matrix is unbounded. Albert and Anderson therefore recommend that the iterative algorithm be rerun after scaling the columns of X (except for the intercept column) to have mean zero and variance one. During the iterations monitor the diagonal elements of the dispersion matrix and stop if any of these values exceeds 10^3.

Unfortunately, there is no straight forward way to distinguish between these two cases. The binary logistic regression model must therefore be refit as described in case 2 whenever the condition given in equation (3.2) is satisfied during the iterative maximization. Finally, note that although this method is likely to detect quasicomplete separation, the detection is not guaranteed.

Software Implementations

The LOGISTIC procedure in SAS® v9.1 (SAS Institute Inc., 2003) implements the empirical method of Albert and Anderson (1984) to identify complete and quasi-complete separation among the sample points. The values of the parameters used are $K = 9$ and $\epsilon = 0.05$ and the IRLS iterations terminate when at least one of the diagonal elements of the dispersion matrix for the standardized observations exceeds $5,000$ (SAS Institute Inc., 2004, chap. 42).

Both S-PLUS® (Insightful, 2005) and R (R Development Core Team, 2007) monitor the predicted probabilities during the IRLS iterations and issue a warning when the probability of correct classification for at least one sample point becomes sufficiently high. R issues a warning when a probability numerically equal to one or zero is encountered. The warning issued by S-PLUS® occurs in a closed-source portion of code hence the specific threshold cannot be obtained.

3.1.2 The Algebraic Approach Suggested in Albert and Anderson

The algebraic approach suggested in Albert and Anderson (1984) uses the ideas of linear programming and is related to unpublished work in the thesis of Burridge (1981). Suppose there is complete or quasicomplete separation among the sample points, then there is a nonzero vector β satisfying

$$\bar{X}\beta \geq 0. \tag{3.3}$$

For quasicomplete separation equality must hold for at least one sample point. If (3.3) cannot be satisfied with $\beta \neq 0$ then there is overlap among the sample points. Albert and Anderson correctly note that the set of all vectors satisfying equation (3.3) is equivalent to the set of feasible solutions to a linear programming problem with constraints (3.3) and suggest that an adaptation of a linear programming technique could be used to determine whether this set contains only the zero vector.

The method of feasible directions provides one approach for testing Albert and Anderson's condition. A feasible direction method is a technique generally used in nonlinear programming (e.g., Luenberger, 1984, chap. 11). The underlying idea is to take steps through the feasible region of the form

$$\beta_{(m+1)} = \beta_{(m)} + k_{(m)}d_{(m)} \tag{3.4}$$

where $d_{(k)}$ is a direction vector (i.e., the feasible direction) and $k_{(m)}$ is a nonnegative scalar. When the feasible direction is used for nonlinear optimization, the parameter $k_{(m)}$ is chosen to maximize the value of the objective function in the direction $d_{(k)}$

from $\beta_{(k)}$. However, since we are only trying to decide whether the feasible region contains a nonzero point, it is sufficient to compute only the initial direction $d_{(0)}$ or determine that no such direction exists. If we can find a nonzero $d_{(0)}$ then there is a nonzero feasible point $\beta_{(1)}$ satisfying equation (3.3). Otherwise $\beta \equiv 0$ is the only feasible point. Zoutendijk (1960) introduces the following method to compute a feasible direction d_0. Consider the minimization problem

$$\begin{aligned} \text{minimize:} \quad & -e^T \bar{X} d \\ \text{subject to:} \quad & -\bar{x}_i^T d \leq 0, \quad i \in I \\ & \textstyle\sum_{j=1}^{p} |d_j| = 1 \end{aligned} \tag{3.5}$$

where $\bar{x}_i^T$ is the i^{th} row of $\bar{X}$ and I is the set of active constraints (the constraints for which equality holds) in (3.3) given $\beta = 0$. Since each element on the right-hand side of (3.3) is zero, the set $I = \{1, 2, \ldots, n\}$. If we split each element of d into positive and negative components $d_j = (d_j^+ - d_j^-)$ with $d_j^+ \geq 0$ and $d_j^- \geq 0$ then (3.5) can be stated as the linear program

$$\begin{aligned} \text{minimize:} \quad & c^T d^* \\ \text{subject to:} \quad & Ad^* \leq 0 \\ & e_{2p}^T d^* = 1 \\ & d^* \geq 0 \end{aligned} \tag{3.6}$$

where

$$c = \begin{bmatrix} -e^T \bar{X} \\ e^T \bar{X} \end{bmatrix}, \qquad A = \begin{bmatrix} -\bar{X} & 0 \\ 0 & \bar{X} \end{bmatrix}, \qquad d^* = \begin{bmatrix} d^+ \\ d^- \end{bmatrix}$$

and e_{2p} is a vector of $2p$ ones. The constraint $e_{2p}^T d^* = 1$ eliminates $d \equiv 0$ as a

feasible point; hence, if (3.6) is feasible, there is a direction d such that the feasible set defined by equation (3.3) contains the point kd for some $k > 0$. Further, $\beta = kd$ gives either complete or quasicomplete separation among the sample points. If (3.6) is infeasible then $\beta \equiv 0$ is the only point satisfying (3.3) indicating overlap among the sample points.

3.1.3 A Mixed Integer Linear Program

In their note on Albert and Anderson's article, Santner and Duffy (1986) provide a mixed integer linear program capable of classifying a set of n sample points as completely separated, quasicompletely separated or overlapped. In the case that there is quasicomplete separation among the sample points, the mixed integer linear program additionally identifies the minimal set of quasiseparated points Q^m. The mixed integer linear program described in Santner and Duffy is for a multinomial response. It can be stated concisely for binary data as

$$\begin{aligned} \text{maximize:} \quad & e_n^T z \\ \text{subject to:} \quad & z \leq 2\bar{X}\beta \;/\; M \\ & \beta \ \text{ free} \end{aligned} \tag{3.7}$$

where $z_i \in \{0, 1\}$ for $i = 1, \ldots, n$, $z = (z_1, \ldots, z_n)^T$, e_n^T is a vector of n ones and $M > 0$ ($M \approx 10^{-09}$) is a tolerance parameter.

The point $(z \equiv 0, \beta \equiv 0)$ is always feasible, hence the mixed integer linear program (3.7) is always feasible. The mixed integer linear program works because if any sample point satisfies $x_i^T \beta \geq M$ then the corresponding z_i may take the value one. Thus $z_i = 1$ for the sample points that are completely separated by β and $z_i = 0$ for the sample points quasiseparated by β. If there is overlap among the sample

points then (3.7) terminates with an optimal value of zero. If there is complete separation among the sample points then (3.7) terminates with an optimal value of n. Finally, when there is quasicomplete separation among the sample points, (3.7) terminates with a positive optimal value less than n. Further, $Q^m = \{i : z_i = 0\}$ and the computed optimal β satisfies $Q(\beta) = Q^m$.

Mixed integer linear programs can be solved using the branch-and-bound method, details of which can be found in Papadimitriou and Steiglitz (1982). Essentially, branch-and-bound involves constructing a sequence of linear programs that proves an integer solution is optimal by successively partitioning the solution space.

We have implemented the mixed integer linear program described in Santner and Duffy (1986) using the lp_solve library (Berkelaar et al., 2007) (see section 4.1.1) and have found that, in practice, the time required to test for separation using the mixed integer linear program is greater than that for fitting the model using the method of iteratively reweighted least squares by two to three orders of magnitude. Hence this implementation of the mixed integer linear program is not suitable for routine use as part of the fitting procedure for binary logistic regression models.

3.2 Methods for Finding Linear Separability

Following from section 1.5, the problem of detecting separation among the sample points is closely related to the problem of finding linear separability between the two groups of sample points. In particular, in the case of a binary response, the existence of linear separability is equivalent to the case of complete separation.

3.2.1 Perceptron Learning Rule

Rosenblatt (1957)'s perceptron learning rule provides a simple iterative procedure for computing a weight vector γ that linearly separates the n sample points (Schölkopf and Smola, 2002, chap. 7). Recall that a perceptron is a function that returns the sign of a linear combination of the covariates associated with each sample point. The updating rule for the weights vector γ takes into account the output of the perceptron; hence γ is only updated if the current pattern is misclassified. The perceptron learning rule has the form

$$\gamma := \gamma + 2\eta x_i^T I(x_i^T \gamma \leq 0)$$

where $I(\cdot)$ is the indicator function. Further, we may take $\eta = 1/2$ since this only requires that γ be rescaled. Rosenblatt (1958) showed that by following this rule, γ will converge to a value γ^* that linearly separates the groups of sample points in a finite number of iterations when such a solution exists. In the case of linear inseparability, the perceptron learning rule will eventually cycle (Ripley, 1996, chap. 3). However, the cycling is difficult to detect and will almost certainly take a long time to develop. Detecting cycling in the perceptron learning rule is therefore not a practical method to test for linear inseparability.

3.2.2 Linear Programming Methods for Pattern Separation

In the context of machine learning, linear programming has been used to design discriminant functions for pattern classification. Let X_1 be the $n_1 \times p$ matrix with rows x_i^T for $i \in E_1$ and let X_2 be the $n_2 \times p$ matrix with rows x_i^T for $i \in E_2$. An alternative definition for the linear separability of the two groups of sample points

E_1 and E_2 is the existence of a hyperplane H with equation

$$x^T\gamma = 0 \tag{3.8}$$

where $x_1 \equiv 1$ and γ is a vector of p coefficients satisfying

$$X_1\gamma < 0 \text{ and } X_2\gamma > 0. \tag{3.9}$$

If there is no such γ then the sample points in the groups E_1 and E_2 are said to be *linearly inseparable*. Determining whether there is a weights vector γ giving linear separability can be accomplished using the techniques of linear programming. For example, Mangasarian (1965) proposes the linear program

$$\begin{aligned} \text{maximize:} \quad & d_1 + d_2 \\ \text{subject to:} \quad & d_1 + X_1\gamma \leq 0 \\ & d_2 - X_2\gamma \leq 0 \\ & -f \leq \gamma_j \leq f \text{ for } j = 1, \ldots, p \\ & d_1 \geq 0, \quad d_2 \geq 0 \end{aligned} \tag{3.10}$$

where d_1 and d_2 are scalars and $f > 0$ provides a bound on the elements of γ. A graphical representation of the problem is shown in figure 3.1. If the solution of (3.10) is greater than zero then there the two groups of sample points are linearly separable. On the other hand, if the maximum value of (3.10) is equal to zero then the two groups of sample points are linearly inseparable.

Mangasarian's linear program is typical of the approaches considered early in the machine learning literature in that it seeks to maximize the margin between the two sets of sample points (Minnick, 1961; Ripley, 1996). Methods taking this approach

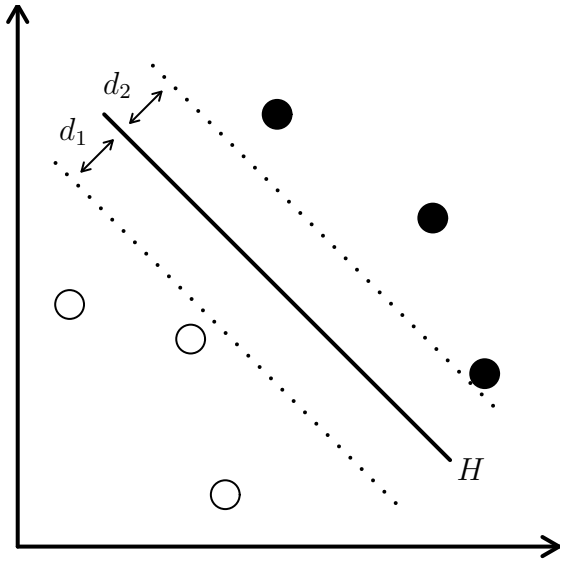


Figure 3.1: Six sample points linearly separated by the hyperplane H with equation $x^T\gamma = 0$. The sample points in the set E_1 are represented by open circles and the sample points in the set E_2 by filled circles. Following from Mangasarian's linear program, the size of the margin is independent for each set.

compute a nonzero γ only if the two groups of sample points are linearly separable. In particular, if there is quasicomplete separation among the sample points then the optimal weights vector computed by Mangasarian's linear program is the zero vector. Hence, when the two groups of sample points are linearly inseparable, the configuration of the sample points cannot be classified. Both overlap and quasicomplete separation are possible.

Smith (1968) proposes a linear program that minimizes the same objective used in the fixed-increment adaptive design method (Mays, 1964) and returns a useful weights vector γ regardless of whether the two groups of sample points are linearly separable. When the two groups of sample points are linearly separable, the linear program finds a weights vector γ linearly separating the two groups. Smith's linear program is based on the concept of the pattern error function (Koford, 1964; Koford

and Groner, 1966). The pattern error h_i for the i^{th} sample point is given by

$$h_i = \begin{cases} -(\bar{x}_i^T\gamma - d) & \text{when } \bar{x}_i^T\gamma < d, \\ 0 & \text{when } \bar{x}_i^T\gamma \geq d, \end{cases} \tag{3.11}$$

where $\bar{x}_i$ is the structure vector associated with the i^{th} sample point and $d > 0$ is the desired size of the margin. The choice of d is arbitrary since γ may be rescaled hence $d = 1$ is assumed. The objective is to minimize the mean error function

$$\bar{h} = \sum_{i=i}^{n} \pi_i h_i \tag{3.12}$$

where π_i is a weighting coefficient which, for our purposes, takes the value $1/n$ for $i = 1, \ldots, n$.

Several adaptive methods have been proposed to minimize equation (3.12). For example, Koford (1964) uses the steepest-descent design method which starts from some initial γ (e.g., $\gamma = 0$) and changes γ in increments

$$\Delta\gamma = -\frac{\psi}{n}\frac{\partial h(\gamma)}{\partial\gamma} = \frac{\psi}{n}\sum_i \pi_i x_i$$

where ψ/n is a constant that controls the size of $\Delta\gamma$ and the summation is made over all $i \in \{1, \ldots, n\}$ such that $h_i \neq 0$.

An alternative is the so-called one-at-a-time design method considered in Mays (1964), Nilsson (1965) and Koford (1964). The one-at-a-time design method is an approximation to the steepest-descent design method achieved by considering each pattern individually rather than all the patterns at once. The weights vector γ is

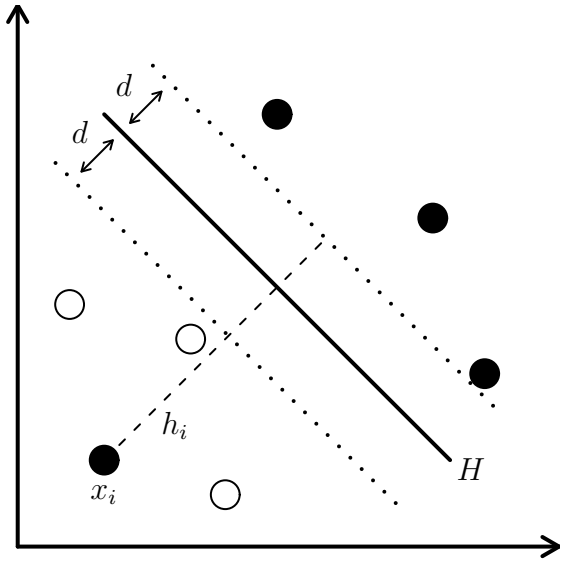


Figure 3.2: Seven sample points and a candidate separating hyperplane H with equation $x^T\gamma = 0$. The sample point x_i has nonzero pattern error $h_i = -(x_i^T\gamma - d)$. The pattern error for each of the remaining six sample points is zero.

updated according to the rule

$$\Delta\gamma = \begin{cases} \frac{\psi}{n}\pi_i x_i & \text{if } h_i \neq 0 \\ 0 & \text{if } h_i = 0 \end{cases}$$

after each $x_i^T\gamma$ is evaluated.

Since the mean error function (3.12) is a linear segment function of γ, its minimum can also be computed using linear programming. Smith (1968) shows that the linear program

$$\begin{aligned} \text{maximize:} \quad & \pi^T\bar{X}\gamma - \pi^T s \\ \text{subject to:} \quad & \bar{X}\gamma + h - s = e_n \\ & h \geq 0, \quad s \geq 0 \\ & \gamma \text{ free} \end{aligned} \tag{3.13}$$

where $h = (h_1, \ldots, h_n)^T$, $\pi = (\pi_1, \ldots, \pi_n)^T$ and s is a vector of n slack variables

attains an optimal value of $\pi^T e_n$ if there exists a vector γ^* giving linear separability between the two groups of sample points. If the two sets of patterns are linearly inseparable then the linear program terminates with an objective value strictly less than $\pi^T e_n$.

When there is quasicomplete separation among the sample points, Smith's linear program terminates with an optimal value of $\pi^T e_n - k/n$ where k is the number of sample points in the set $C = \{i : h_i > 0\}$. Further, $h_i = d$ for $i \in C$. Thus it is possible to classify the sample points as either quasicompletely separated or overlapped when the optimal value of the linear program is less than $\pi^T e_e$.

Smith's linear program also provides a bound on the *regression depth* as defined in Christmann and Rousseeuw (2001). The regression depth of a binary logistic regression model is the minimum number of sample points that must be removed from the data to achieve separation. Suppose Smith's linear program terminates with a computed weights vector γ^* and an optimal value indicating overlap. Let C be the set $\{i : h_i > d\}$. If the sample points in C are removed from the data then γ^*, suitably rescaled, separates the remaining sample points. It follows that $|C|$, the number of sample points in the set C, is an upper bound on the regression depth. Further, in practice the solution to the linear program can be computed in far less time than the resampling method proposed in Christmann and Rousseeuw (2001).

3.3 Methods for Determining the Existence of the Maximum Likelihood Estimate

Lastly, we consider two tests for the existence of the maximum likelihood estimate of the binary logistic regression model parameter vector that do not rely on determining the configuration of the sample points. The first, proposed by Barndorff-Nielsen (1978), involves computing the convex hull of the support for the sufficient statistic then deciding whether the observed value of the sufficient statistic lies in its interior. The second method, proposed by Jacobsen (1989), involves determining whether the point zero lies in the interior of the convex hull of the structure vectors (the rows of $\bar{X}$). We consider computing both of these tests using the Quickhull Algorithm (Barber et al., 1996) and additionally note that Jacobsen's condition is the same as the algebraic approach given in Albert and Anderson (1984).

3.3.1 Barndorff-Nielsen's Test

The test suggested in Barndorff-Nielsen (1978) for the existence of the maximum likelihood estimate of the binary logistic regression model parameter vector is based on the result of the following theorem. Since the maximum likelihood estimate exists only when there is overlap among the sample points, this test can additionally be used to classify a set of sample points as separated or overlapped.

Denote by C the support for the sufficient statistic. The binary logistic regression model is discrete, hence the support C is the set of all possible values of the sufficient statistic

$$t = T(y) = \left(\sum_{i=1}^{n} y_i, \sum_{i=1}^{n} y_i x_{i2}, \ldots, \sum_{i=1}^{n} y_i x_{ip} \right)^T . \tag{3.14}$$

Since the model is assumed to include an intercept term, $x_{i1} = 1$ for $i = 1, \ldots, n$ and is omitted from the formula for the sufficient statistic t_1. Let conv C be the convex hull of the set C and let P be the polytope with vertices conv C (see Papadimitriou and Steiglitz, 1982, chap. 2).

Theorem 4 (Barndorff-Nielsen: Theorem 9.13)**.** *The log-likelihood function l has a maximum if and only if $t \in int\, P$, and then the maximum is unique.*

If the observed value of the sufficient statistic t^{obs} is extreme, then it lies in at least one of the facets of P. In this case the parameter vector β cannot be estimated by maximum likelihood. If no facet F of P contains the observed value of the sufficient statistic then t^{obs} lies in the interior of P and the maximum likelihood estimate exists.

Barndorff-Nielsen (1978) also shows that, although the maximum likelihood estimate of β does not exist when t^{obs} lies on the boundary of P, it is still possible that a certain subset of the parameters can be estimated by maximum likelihood. This result is similar to but more general than the reduced problems considered in Santner and Duffy (1986). Suppose that t^{obs} belongs to the relative interior of a face F of P of dimension d. Then a d dimensional affine transformation of β is estimable by maximum likelihood.

Computing Barndorff-Nielsen's Test

The first task in computing Barndorff-Nielsen's test is to compute the convex hull of the sufficient statistics. We can use the `perms` function in the Matlab® (The MathWorks, 2006) Statistics Toolbox to generate the $2^n \times p$ matrix Y whose rows contain all possible outcomes for the response in a binary logistic regression model.

This matrix is used in conjunction with equation (3.14) to compute a $2^n \times p$ matrix T such that the i^{th} row of T contains the sufficient statistic corresponding to the outcome in the i^{th} row of Y. We can then use Matlab®'s `convhulln` function to compute conv C, the convex hull of the sufficient statistics, using the Quickhull Algorithm (Barber et al., 1996). The output of the `convhulln` function is a $K \times p$ matrix Z where K is the number of facets comprising the convex hull. The i^{th} row z_i^T of Z contains the indices of the rows of T which are the vertices determining the i^{th} facet.

The second task involved in computing Barndorff-Nielsen's test is to determine whether the observed value of the sufficient statistic t^{obs} lies in one or more facets of P. For each facet, we determine whether t^{obs} lies in the hyperplane spanned by the p vertices defining the facet. If t^{obs} does not belong to any of the facets of P then it lies in the interior of P and the maximum likelihood estimate of the parameter vector β exists and is unique. Otherwise, let m be the number of facets containing t^{obs}. Then t^{obs} belongs to a face of P of dimension $d = p - m$.

Example

Consider a simple logistic dose-response model consisting of six sample points. We assume that the explanatory variables are in increasing order, i.e.,

$$x = (-1, -1, 0, 0, 1, 1)^T.$$

Recall that we consider the data as a set of sample points rather than covariate classes hence the repeated measurements. For the logistic does-response model

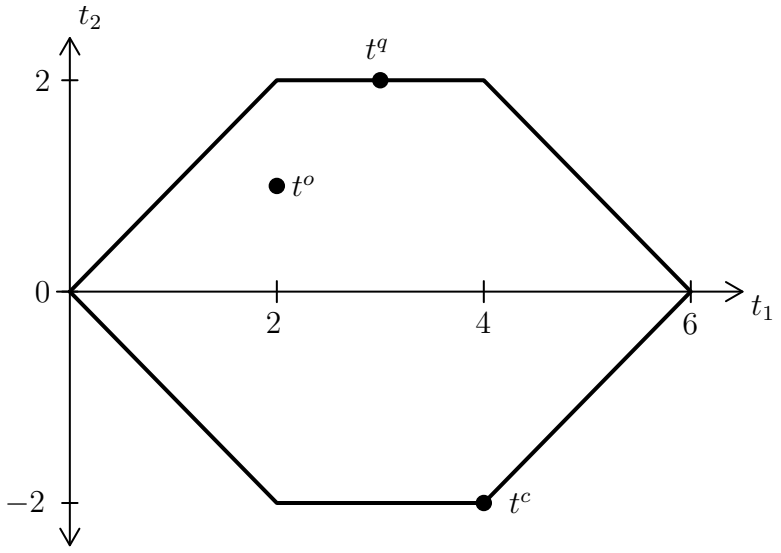


Figure 3.3: The polytope P with vertices conv C. When the observed value of the sufficient statistic lies in the interior (e.g., t^o) then β can be estimated by maximum likelihood. When the observed value of the sufficient statistic lies in a face of dimension one (e.g., t^q) subparameters of β can be estimated. When the observed value of the sufficient statistic lies in a face of dimension zero (e.g., t^c) then β can not be estimated.

$p = 2$ hence the support C can be written as

$$C = \left\{ (t_1, t_2) : t_1 = \sum_{i=1}^{n} y_i,\ t_2 = \sum_{i=1}^{n} y_i x_i,\ \ y_i = 0, 1,\ \ i = 1, \ldots, n \right\}. \tag{3.15}$$

The polytope generated by the convex hull of the set C is shown in figure 3.3. Although the convex hull may be computed using the algorithm presented in the previous section, a short cut is possible when $p = 2$. If we condition on the number of successes t_1 when $t_1 \geq 1$ then it is easy to see that the maximum and minimum values of t_2 are given by

$$\max(t_2|t_1) = \sum_{i=n+1-t_1}^{n} x_i \qquad \text{and} \qquad \min(t_2|t_1) = \sum_{i=1}^{t_1} x_i$$

and clearly $(t_2|t_1 = 0) = 0$. The points obtained by calculating the maximum and minimum values of t_2 for $t_1 = 0, 1, \ldots, 6$ determine the facets of P.

A simple graphical method can now be used to complete Barndorff-Nielsen's test. Draw the polytope P as in figure 3.3 and plot the observed value of the sufficient statistic. There are three possibilities.

1. The observed value of the sufficient statistic lies in the interior of C, for example T^o; β may be estimated by maximum likelihood. This case corresponds to overlap among the sample points.

2. The observed value of the sufficient statistic lies in a face of dimension one, for example T^q; β_1 may be estimate by maximum likelihood but the maximum likelihood estimate of β_2 does not exist. This case corresponds to quasicomplete separation among the sample points.

3. The observed value of the sufficient statistic lies in a face of dimension zero (i.e., $t^{obs} \in \text{conv } C$), for example T^c; the maximum likelihood estimate of β does not exist. This case corresponds to complete separation among the sample points.

Interestingly, Barndorff-Nielsen also shows that if the observed value of the sufficient statistic lies on a face of dimension one that is not parallel to one of the coordinate axis then a one-dimensional affine transformation of β may be estimated by maximum likelihood. For example, if $t^{obs} = (1, 1)$ then the quantity $\beta_1 + \beta_2$ can be estimated by maximum likelihood so that $P(Y = 1|x = 1)$ can be recovered while $P(Y = 1|x = 0)$ and $P(Y = 1|x = -1)$ cannot. In contrast, the reduced problem given in Santner and Duffy (1986) would take neither β_1 nor β_2 to be estimable in this case.

3.3.2 Jacobsen's Method

While Barndorff-Nielsen provides an elegant test to determine the existence and uniqueness of the maximum likelihood estimate of β, the fact that it requires computing the convex hull of all possible values of the sufficient statistic makes it computationally impractical as the value of n grows beyond 20. Jacobsen (1989) proposes a similar test that does not rely on the sufficient reduction of the observations. The test is based on the following theorem which is stated using our notation.

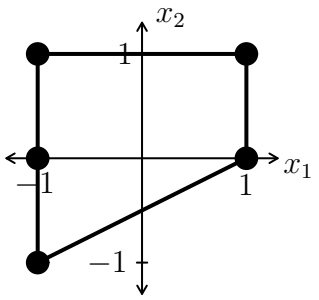
Theorem 5 (Jacobsen: Theorem (b))**.** *The likelihood attains its maximal value at a unique point $\hat{\beta} \in \mathbb{R}^p$ if and only if either of the following two conditions is satisfied:*

b(i) $0 \in$ *int conv* $\{\bar{x}_i : i = 1, \ldots, n\}$,

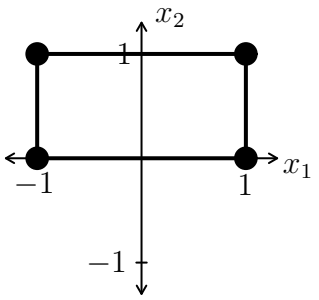
b(ii) *there does not exist* $\beta \in \mathbb{R}^p$ *with* $\beta \neq 0$ *such that* $\langle \beta, \bar{x}_i \rangle \geq 0$ *for* $i = 1, \ldots, n$.

Let $S = \{\bar{x}_1, \ldots, \bar{x}_n\}$ and $S^* = \{S\} \bigcup \{0\}$. Denote by conv S^* the convex hull of the set S^* and let P be the convex polytope with vertices conv S^*.

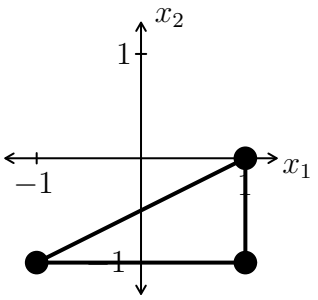
We can use the result $b(i)$ of Jacobsen's theorem to classify a set of n sample points as completely separated, quasicompletely separated or overlapped as follows. If $0 \in$ conv S^* then the convex hull of the structure vectors does not contain the point 0 hence there is complete separation among the sample points. If $0 \notin$ conv S^* but $0 \in F$ for some facet F of P then there is quasicomplete separation among the sample points. Further, let H be the supporting hyperplane of F and let β be a normal vector to H. Then β gives quasicomplete separation among the sample points and minimal set $\{x_i : i \in Q^m\}$ of quasiseparated sample points lies in the hyperplane H. Finally, if $0 \notin$ conv S^* and if $0 \notin F$ for all facets F of P then 0 lies in the interior of S^*. In this case, there is overlap among the sample points and



(a) Overlap: the convex hull strictly contains the origin: β is estimable by maximum likelihood.



(b) Quasicomplete separation: a facet of the convex hull contains the origin: β_1 is estimable by maximum likelihood but β_2 is not.



(c) Complete separation: the convex hull does not contain the origin: the maximum likelihood estimate of β does not exist.

Figure 3.4: Jacobsen's test for the existence of the maximum likelihood estimate. The examples correspond to the observed sufficient statistics t^o, t^q and t^c in figure 3.3.

the maximum likelihood estimate of the binary logistic regression model parameter vector β exists and it unique.

Computing Jacobsen's Test

Computing Jacobsen's test is relatively straight forward. Append a row of zeros to the matrix $\bar{X}$ whose rows contain the structure vectors $\bar{x}_i$ and compute the convex hull using the Matlab® function `convhulln`. The output includes a $K \times p$ matrix Z whose rows contain the indices of the p sample points determining each of the K facets of the convex hull. If at least one of the elements of Z is equal to $n+1$ then 0 is an extreme point and is not contained in the convex hull of the structure vectors. If 0 is not an extreme point then it is still possible that $0 \in F_i$ for some $i \in \{1, \ldots, K\}$ where $\{F_i\}_{i=1}^K$ are the K facets of P. For each facet F_i determine the supporting hyperplane H_i with equation

$$h_{i1}x_1 + \cdots + h_{ip}x_p = 0.$$

If $h_{i1} = 0$ for some i then $0 \in F_i$. To allow for small computational inaccuracies, we say that $h_{i1} = 0$ if $|h_{i1}| < \epsilon$ for some suitably small choice of $\epsilon > 0$. If 0 is neither extreme nor contained in any facet of P then $0 \in \text{int } S$.

Verifying Jacobsen's Condition Using Linear Programming

The second condition $(b(ii))$ in Jacobsen's theorem can be stated in matrix form as there is no $\beta \neq 0$ such that

$$\bar{X}\beta \geq 0$$

which is precisely the condition in the algebraic approach considered in Albert and Anderson (1984). The linear program described in the next chapter requires far less time than computing the convex hull of the structure vectors. For set of 1000 sample points in 16 dimensions computing the convex hull takes roughly 100 times longer than solving the linear program. Hence the linear programming approach is the preferred method for testing Jacobsen's condition.

Chapter 4

A Linear Program for Separation Detection

This chapter shows how separation detection can be posed as an optimization problem and discusses the practical solution of this problem using the techniques of linear programming. The use of linear programming to test for the presence of separation among the sample points was mentioned in Albert and Anderson (1984) and further developed in Silvapulle and Burridge (1986) and in Clarkson and Jennrich (1991). The approaches described in both of these papers rely on transforming the linear program to eliminate the free decision variables. We have found that when these transformations are carried out using double precision arithmetic, sufficient rounding error accumulates that the solution is no longer reliable. Our approach differs in that we work with the dual of the linear program which can be stated using nonnegative decision variables; hence no initial transformation is required.

The two main results stemming from our approach are an algorithm providing a test for separation among the sample points and an algorithm for finding a vector β

revealing separation among the sample points when it is present. Both algorithms are based on the same linear program; the test for separation stops as soon as a positive value of the objective function is found (i.e., when any type of separation is present among the sample points) while the second algorithm continues until the optimal value of the objective function is reached. The vector β obtained from the second algorithm is can also be used to distinguish between the cases of complete separation and quasicomplete separation. However, this classification is not guaranteed: if $Q(\beta)$ is the set of sample points quasiseparated by the computed β, then $Q^m \subseteq Q(\beta)$ where Q^m is the minimal set described in Albert and Anderson (1984). We have so far been unable to construct an example demonstrating this behavior. Both algorithms distinguish between the cases of overlap and separation and thus provide a test for the existence of the maximum likelihood estimate of the binary logistic regression model parameter vector β.

The chapter is organized as follows. In section 4.1 we give a brief description of linear programming and discuss the solution of linear programs using the revised simplex method implemented in the open source library lp_solve (Berkelaar et al., 2007) and the interior point method provided in the MATLAB® (The MathWorks, 2006) Optimization Toolbox. In section 4.2 we describe the linear program underlying our approach and in section 4.3 we compare it to the methods suggested in Albert and Anderson (1984) and Silvapulle and Burridge (1986). In section 4.4 we describe the techniques we chose to use to solve the linear program and show how the linear program can be used to test for separation and to find a vector revealing separation. Section 4.5 discusses some numerical concerns and section 4.6 contains a simulation study comparing the performance of the methods for solving the linear program.

4.1 Linear Programming

Linear programming is the branch of optimization theory concerned with minimizing a linear combination of a set of so-called decision variables subject to a set of linear equality and inequality constraints. Linear programming was introduced independently by Leonid Kantorovich in 1939 (Orden, 1993) and George Dantzig in 1947 (Dantzig and Thapa, 1997). Since then, linear programming has been used to solve a wide variety of optimization problems. In fact, linear programming initially proved so useful that early funding for the development of computers was granted to make it possible to solve linear programs (Dantzig and Thapa, 1997). Consequently, there are several robust, efficient software packages available today for solving linear programs. The linear programs presented in this chapter can be solved by the revised simplex method using the freely available lp_solve (Berkelaar et al., 2007) library. We have developed an interface to lp_solve so that these routines may be executed from within R (R Development Core Team, 2007). We also make use of the `linprog` function in the MATLAB® Optimization Toolbox (The MathWorks, 2006) to solve linear programs using an interior point method.

For a thorough description of linear programming see Luenberger (1984). We adopt his terminology, theorems and the descriptions of both the simplex method and the revised simplex method which are given in chapter 3. Alternative resources for linear programming can be found in Dantzig and Thapa (1997), Chvátal (1983), and Papadimitriou and Steiglitz (1982) among others.

The standard form of a linear program is to find values of the decision variables $\gamma_1 \geq 0$, $\gamma_2 \geq 0$, $\ldots$, $\gamma_m \geq 0$ and the minimum value z satisfying

$$\begin{array}{ccccccccccc} a_{11}\gamma_1 & + & a_{12}\gamma_2 & + & \ldots & + & a_{1m}\gamma_m & = & b_1 & & \\ a_{21}\gamma_1 & + & a_{22}\gamma_2 & + & \ldots & + & a_{2m}\gamma_m & = & b_2 & & \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots & & \\ a_{n1}\gamma_1 & + & a_{n2}\gamma_2 & + & \ldots & + & a_{nm}\gamma_m & = & b_n & & \\ c_1\gamma_1 & + & c_2\gamma_2 & + & \ldots & + & c_m\gamma_m & = & z & \text{(min)} \end{array} \tag{4.1}$$

where c_j is the cost associated with the j^{th} decision variable γ_j, the j^{th} coefficient in the i^{th} constraint equation is a_{ij} and b_i is the right hand side of the i^{th} constraint. The standard form can be stated concisely using matrix notation as

$$\begin{aligned} \text{minimize:} \quad & c^T\gamma \\ \text{subject to:} \quad & A\gamma = b \\ & \gamma \geq 0 \end{aligned} \tag{4.2}$$

where $b = (b_1, \ldots, b_n)^T$, $c = (c_1, \ldots, c_m)^T$, $\gamma = (\gamma_1, \ldots, \gamma_m)^T$, A is an $n \times m$ matrix with elements a_{ij} and the notation $\gamma \geq 0$ means that each element of the vector γ is greater than or equal to zero. Any vector γ that satisfies (4.2) is called a *feasible solution* and a vector γ representing feasible solution for which the value of $z = c^T\gamma$ is minimum is called an *optimal feasible solution.*

Notation

The $n \times p$ design matrix from the binary logistic regression model is denoted by X and the binary response vector by y. The elements of y are 0 for the sample points in E_1 and 1 for the sample points in E_2. The response vector is also denoted as $\tilde{y}$ where $\tilde{y}_i = -1$ for the sample points in E_1 and $\tilde{y}_i = 1$ for the sample points in E_2. Following Jacobsen (1989) we call $\bar{x}_i = \tilde{y}_i x_i$ for $i = 1, \ldots, n$ the *structure*

vectors; the matrix whose rows are the structure vectors is denoted by $\bar{X}$. The $n \times m$ matrix A contains the coefficients of the constraint equations for the linear program when written in the standard form. We also use A to denote the simplex tableau in which case the $(n+1)^{th}$ row contains the coefficients of the objective function c^T and the $(m+1)^{th}$ column is the vector of constraints b. We use γ for generic decision variables and β for the decision variables corresponding to the parameters in the binary logistic regression model.

4.1.1 Solving Linear Programs with the Revised Simplex Method

The lp_solve library provides routines for solving pure linear, mixed integer/binary, semi-continuous and special ordered sets (SOS) programs using the revised simplex method. Full documentation is available in the reference API (Berkelaar et al., 2007). We provide an R interface to the lp_solve library via the function

```
lpSolve(obj, A, b, Aeq = NULL, beq = NULL, lb = 0, ub = Inf,
        intvec = integer(0), control = list())
```

which solves the linear program

$$\begin{aligned} \text{minimize:} \quad & \texttt{obj}^T \gamma \\ \text{subject to:} \quad & \texttt{A}\ \gamma \leq \texttt{b} \\ & \texttt{Aeq}\ \gamma = \texttt{beq} \\ & \texttt{lb[j]} \leq \gamma_j \leq \texttt{ub[j]} \ \text{ for } \ j = 1, \ldots, p \end{aligned} \tag{4.3}$$

where the `obj` argument is the vector of objective function coefficients, `A` and `Aeq` are matrices of constraint coefficients, `b` and `beq` are vectors containing the constraint

values and `ub` and `lb` are vectors containing the upper and lower bounds on the decision variables. The bounds on the decision variables are permitted to take infinite values. The `intvec` argument can be used to specify decision variables that may take only integer values, hence `lpSolve` can be used to solve the mixed integer linear program suggested in Santner and Duffy (1986) as well. The `control` argument is used to pass optional control parameters to the lp_solve solver; these will be described as needed.

Through a sequence of simple algebraic manipulations, any linear program may be represented in the form of (4.3) (Luenberger, 1984).

The output consists of a status message indicating whether an optimal solution was found, the optimal value of the objective function and a vector γ corresponding to the optimal solution. The value of the objective function and the computed γ are only meaningful when the status message indicates that an optimal solution was achieved.

4.1.2 Solving Linear Programs with an Interior Point Method

The MATLAB® Optimization Toolbox (The MathWorks, 2006) provides the function `linprog` which solves the linear program in (4.3). The interface for the function is given by

```
linprog(obj, A, b, Aeq, beq, lb, ub, x0, options)
```

where the arguments are the same as those in (4.3) except for `x0` and `options`. To solve the linear program using an interior point method we set the following option

```
options = optimset('LargeScale', 'on')
```

to specify that the linear program should be solved using the large scale algorithm, a variant of the predictor-corrector algorithm proposed by Mehrotra (1992). The argument `x0` is not used by the large scale algorithm.

The output consists of an exit flag indicating whether an optimal solution was found, the optimal value of the objective function and a vector γ corresponding to the optimal solution. Again, the value of the objective and the computed γ are only meaningful when the exit flag indicates that an optimal solution was achieved.

4.2 Separation Detection as an Optimization Problem

We propose a nonnegative linear objective function that may take positive values when there is separation among the sample points and is identically zero when there is overlap.

Suppose there is complete separation among a set of n sample points. Then there is a hyperplane H such that all of the sample points in E_1 lie on one side of H and all of the sample points in E_2 lie on the other. This is illustrated for the case $p = 3$ (the intercept term is not shown) in figure 4.1.

Let $\tilde{\beta}$ be a unit vector normal to H. Then the distance of the i^{th} sample point from H is given by $\tilde{s}_i = x_i^T \tilde{\beta}$. Since we have assumed that there is complete separation among the sample points, there is a vector $\tilde{\beta}$ such that $\tilde{s}_i > 0$ for $i \in E_2$ and $\tilde{s}_i < 0$ for $i \in E_1$. The configuration shown in figure 4.1 takes $\tilde{s}_i > 0$ for $i \in \{1, 2, 3\}$ and $\tilde{s}_i < 0$ for $i \in \{4, 5, 6\}$. The complete separation assumption can be relaxed to allow $\tilde{\beta}$ to give quasicomplete separation among the sample points as well. In this

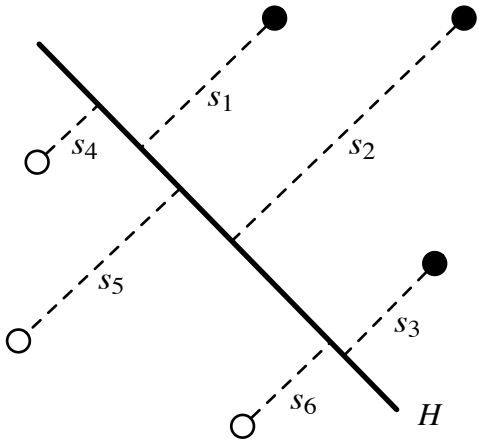


Figure 4.1: Six sample points completely separated by the hyperplane H. The distance of the i^{th} sample point from the hyperplane H is denoted by s_i for $i = 1, \ldots, 6$.

case $\tilde{s}_i \geq 0$ for $i \in E_2$ and $\tilde{s}_i \leq 0$ for $i \in E_1$ with $\tilde{s}_i = 0$ for at least one value of $i \in \{1, \ldots, n\}$. Also, $\tilde{s}_i \neq 0$ holds for at least one value of $i \in \{1, \ldots, n\}$ since the design matrix X is assumed to be of full rank.

Let $S(\tilde{\beta}) = \sum_{i=1}^{n} \tilde{s}_i$. Define an optimal separating hyperplane H^* to be a hyperplane with unit normal vector $\tilde{\beta}^*$ such that $S(\tilde{\beta}^*)$ is maximum. Finding $\tilde{\beta}^*$ can then be posed as the optimization problem

$$\begin{aligned} \text{maximize:} \quad & S(\tilde{\beta}) = \textstyle\sum_{1=1}^{n} \tilde{s}_i \\ \text{subject to:} \quad & \tilde{s}_i = x_i^T \tilde{\beta} \geq 0 \ \text{ for } \ i \in E_2 \\ & \tilde{s}_i = x_i^T \tilde{\beta} \leq 0 \ \text{ for } \ i \in E_1 \\ & \tilde{\beta}^T \tilde{\beta} = 1. \end{aligned} \tag{4.4}$$

The constraint inequalities require that each sample point lie in or on the appropriate side of H^* and the quadratic constraint $\tilde{\beta}^T \tilde{\beta} = 1$ forces $\tilde{\beta}$ to have unit length. If there is overlap among the sample points then there is no vector $\tilde{\beta}$ satisfying

the constraints of (4.4) hence the problem is infeasible. If (4.4) is feasible then its solution provides a vector $\tilde{\beta}^*$ maximizing $S(\tilde{\beta})$ and revealing either complete or quasicomplete separation among the sample points. Also, note that deciding whether there is separation, and hence demonstrating the nonexistence of the binary logistic regression model maximum likelihood estimate, is equivalent to determining the feasibility of (4.4). Solving (4.4) additionally permits the classification of sample points as completely separated or quasiseparated by the computed $\tilde{\beta}^*$.

The optimization task in (4.4) belongs to a class of problems called nonlinearly constrained optimization. Methods for solving nonlinearly constrained optimization problems generally follow a sequence of infeasible points to approximate the optimal value of the objective function (Gill et al., 1981, chap. 3); hence nonlinearly constrained optimization is not appropriate for our purposes.

The quadratic constraint $\tilde{\beta}^T\tilde{\beta} = 1$ in (4.4) asserts that $\tilde{\beta}$ is a unit vector. Each $\tilde{s}_i$ may therefore be interpreted as the distance from the i^{th} sample point to the hyperplane H. However, since we are only interest in whether each sample point is completely separated (i.e., $\tilde{s}_i > 0$) or quasiseparated (i.e., $\tilde{s}_i = 0$) by $\tilde{\beta}$, the length of $\tilde{\beta}$ is irrelevant. Removing the quadratic constraint yields the following optimization problem expressed in the unscaled distances $s_i = \tilde{s}_i \, \|\beta\|_2$ for $i = 1, \ldots, n$ and where $\|\cdot\|_2$ denotes the 2-norm.

$$\begin{aligned} \text{maximize:} \quad & \textstyle\sum_{i=1}^{n} s_i \\ \text{subject to:} \quad & s_i = x_i^T\beta \geq 0 \;\text{ for }\; i \in E_2 \\ & s_i = x_i^T\beta \leq 0 \;\text{ for }\; i \in E_1 \\ & -\infty < \beta_j < \infty \;\text{ for }\; j = 1, \ldots, p \end{aligned} \tag{4.5}$$

This problem has a linear objective function and a set of linear constraints and can

therefore be solved using the techniques of linear programming. We can simplify the statement of (4.5) by writing $s_i = \bar{x}_i^T \beta$. The constraints may then be written as

$$s_i = \bar{x}_i^T \beta \geq 0$$

for $i = 1, \ldots, n$. The unscaled distances s_i can now be eliminated from the linear program by noting that $s = \bar{X}\beta$ where $s = (s_1, \ldots, s_n)^T$. The objective function can similarly be expressed in terms of β alone as

$$\sum_{i=1}^{n} s_i = e_n^T s = e_n^T \bar{X} \beta = (e_n^T \bar{X})\beta$$

where e_n is a vector of n ones. Making these substitutions in (4.5) yields the linear program

$$\begin{aligned} \text{maximize:} \quad & e_n^T \bar{X} \beta \\ \text{subject to:} \quad & \bar{X}\beta \geq b \\ & \beta \ \text{ free} \end{aligned} \tag{4.6}$$

where $b = (0, \ldots, 0)^T$ and the notation β free means $-\infty < \beta_j < \infty$ for $j = 1, \ldots, p$. The remainder of this chapter focuses on the practical solution of this linear program which we call the *primal linear program*.

4.3 Literature Review

The idea of using linear programming for separation detection was first suggested in in the literature by Albert and Anderson (1984) who in turn refer to unpublished work in the thesis of J. Burridge. They argue that a vector $\beta \neq 0$ giving either

complete or quasicomplete separation among a set of sample points satisfies

$$\bar{X}\beta \geq 0 \tag{4.7}$$

and that these inequalities can be viewed as the constraint equations defining the set of feasible solutions of a linear program. However, they do not provide an objective function.

Silvapulle and Burridge (1986) go one step further by providing the following linear program arising from the constraint equations in (4.7)

$$\begin{aligned} \text{minimize:} \quad & e_p^T s \\ \text{subject to:} \quad & Bs \geq b \\ & s \geq 0 \end{aligned} \tag{4.8}$$

where $b = (-1, \ldots, -1)^T$ which they claim has a finite optimal solution when there is overlap among the sample points and is unbounded when there is either complete or quasicomplete separation. The $(n-p)\times p$ matrix B is formed as follows. Since $\bar{X}$ is assumed to be of full rank it can be partitioned as

$$\bar{X} = \Pi \begin{bmatrix} \bar{X}_1 \\ \bar{X}_2 \end{bmatrix} \tag{4.9}$$

where the $p\times p$ matrix $\bar{X}_1$ is nonsingular and Π is a suitable permutation matrix. The matrix B is then defined to be $B = \bar{X}_2\bar{X}_1^{-1}$.

Clarifications for the Linear Program Suggested by Silvapulle and Burridge

The linear program (4.8) suggested in Silvapulle and Burridge (1986) requires two clarifications. The first is that the objective function should seek to maximize $e^T s$, otherwise the optimal value of the objective function is trivially zero. Equation (4.8) can be written in feasible canonical form as

$$\begin{bmatrix} 0^T & e_p^T \\ I & -B \end{bmatrix} \begin{bmatrix} s^* \\ s \end{bmatrix} = \begin{bmatrix} z \\ 1 \end{bmatrix} \tag{4.10}$$

where s^* is a vector of slack variables. Equation (4.10) shows that the initial basic feasible solution ($s = 0, s^* = 1$) is optimal; hence (4.8) has an optimal solution of zero for all possible configurations of the sample points. If (4.8) is instead stated as a maximization problem then the linear program will be unbounded when the feasible region is unbounded. This outcome, presumably, is what the authors intended.

The second clarification arises from the suggestion in Silvapulle and Burridge (1986) that the feasibility of the dual linear program be used as a test for whether (4.8) is bounded. The dual linear program to (4.8, when posed as a maximization problem) is

$$\begin{aligned} \text{minimize:} \quad & -b^T \lambda \\ \text{subject to:} \quad & -B^T \lambda \geq e_p \\ & \lambda \geq 0 \end{aligned} \tag{4.11}$$

where λ is a vector of n dual decision variables. The feasibility of the dual is determined by the constraints in (4.11); in particular it does not depend on the primal right-hand side b. Thus any finite choice of b is permissible. Choosing $b = (-1, \ldots, -1)^T$ requires both phases of the two-phase algorithm to be computed

in order to find the optimal value of (4.11). Phase one to determine an initial feasible point and phase-two to find the optimal value. On the other hand if we choose $b \equiv 0$ then only phase one needs to be computed since the optimal value — when the dual is feasible — is clearly zero. Thus choosing $b \equiv 0$ yields a test requiring roughly half the computational effort as that proposed in Silvapulle and Burridge (1986).

Also, note that the linear program proposed in Silvapulle and Burridge (1986) can be used to test for the presence of separation but, when separation is present, does not provide a direction vector β revealing it.

Numerical Difficulties

The transformation $B = \bar{X}_2\bar{X}_1^{-1}$ provides an elegant method to obtain a linear program expressed in nonnegative decision variables. However, in our tests, the numerical error accumulated in computing the product $\bar{X}_2\bar{X}_1^{-1}$ was from time to time sufficient to change the boundedness of the feasible region.

4.4 Solving the Linear Program

The API given in (4.3) is capable of solving the primal linear program (4.6). However, it is worth considering precisely how the software solves the linear program since there are certain pitfalls that could cause the solver to require excessive computation time or in fact not terminate at all.

Algorithms for solving linear programs generally begin by converting the input linear program into the standard form. Both lp_solve and the MATLAB® `linprog`

function take this approach. The two principal tasks involved in representing (4.6) in the standard form are (1) expressing the free decision variables as nonnegative decision variables and (2) transforming the inequality constraints into an equivalent system of equality constraints through the introduction of slack variables. In this section we describe how the software conducts these transformations and suggest alternative methods that may lead to a faster algorithm for our particular purpose.

4.4.1 Free Decision Variables

Both lp_solve and the MATLAB® `linprog` function begin by converting the input linear program into the standard form, e.g., (4.1). We first consider how the software deals with the free decision variables in the input linear program.

Expressing Free Decision Variables as the Difference Between Two Nonnegative Decision Variables

Both lp_solve and the MATLAB® `linprog` function represent free decision variables as the difference between two nonnegative decision variables. That is, if γ_j is free, then the substitution $\gamma_j = (\gamma_j^+ - \gamma_j^-)$ is made where $\gamma_j^+ \geq 0$ and $\gamma_j^- \geq 0$ (Berkelaar et al., 2007; The MathWorks, 2006). The revised simplex method allows at most one of $\{\gamma_j^+, \gamma_j^-\}$ to be nonzero (Luenberger, 1984) so that γ_j^+ and γ_j^- do not grow unbounded as a pair while γ_j remains constant.

Substituting $\beta_j = (\beta_j^+ - \beta_j^-)$ into (4.6) yields

$$\begin{aligned} \text{maximize:} \quad & (e^T\bar{X})^T\beta^+ - (e^T\bar{X})^T\beta^- \\ \text{subject to:} \quad & \bar{X}\beta^+ \geq 0 \\ & -\bar{X}\beta^- \geq 0 \\ & \beta^+ \geq 0, \;\; \beta^- \geq 0 \end{aligned} \tag{4.12}$$

which can be stated concisely as

$$\begin{aligned} \text{maximize:} \quad & c^T\beta^* \\ \text{subject to:} \quad & -\tilde{X}\beta^* \leq 0 \\ & \beta^* \geq 0 \end{aligned} \tag{4.13}$$

where

$$c = \begin{bmatrix} \bar{X}^T e \\ -\bar{X}^T e \end{bmatrix}, \;\; \beta^* = \begin{bmatrix} \beta^+ \\ \beta^- \end{bmatrix}, \;\; \text{and} \;\; \tilde{X} = \begin{bmatrix} \bar{X} & 0 \\ 0 & -\bar{X} \end{bmatrix}.$$

This approach yields a linear program stated in $2p$ nonnegative decision variables. Expressing the inequality constraints as a system of equality constraints requires the addition of a further $2n$ slack variables so that the standard form representation of (4.6) requires $2n + 2p$ decision variables and $2n$ equality constraints.

Eliminating the Free Decision Variables

An alternative approach is to express the free decision variables as linear combinations of the slack variables then substitute them out of the linear program (Luenberger, 1984). This approach is used in Clarkson and Jennrich (1991). The

idea is to construct the *reduced linear program*

$$\begin{aligned} \text{maximize:} \quad & \tilde{c}^T s^* \\ \text{subject to:} \quad & B s^* \geq \tilde{b} \\ & s^* \geq 0 \end{aligned} \tag{4.14}$$

where $\tilde{c}$ and s^* are vectors of p elements and B is an $(n-p) \times p$ matrix. Note that the decision variables in the reduced linear program are nonnegative. Additionally, an optimal solution to the primal linear program (4.6) can be recovered from an optimal solution to the reduced linear program by backward substitution. The reduced linear program is similar to the linear program suggested in Silvapulle and Burridge (1986) but does not require matrix inversion nor finding a permutation matrix such that the matrix $\bar{X}_1$ is nonsingular.

The idea is to eliminate each free decision variable from the linear program by expressing it as a linear combination of the nonnegative decision variables then substituting it out of the linear program. Consider a linear program written in canonical form (e.g., 4.1 but where one or more of the decision variable may be unrestricted in sign). Suppose that the decision variable γ_1 is free. We may take any one of the constraint equations in (4.1) that has a nonzero coefficient for γ_1, say

$$a_{i1}\gamma_1 + a_{i2}\gamma_2 + \cdots + a_{i,m}\gamma_m = b_i \tag{4.15}$$

with $a_{i1} \neq 0$ and solve for γ_1 expressing it as a linear combination of the decision variables $\gamma_2, \ldots, \gamma_m$ plus the constant b_i. If we substitute this expression for γ_1 everywhere in (4.1) we obtain a a new linear program describing the same optimization problem but expressible in the decision variables $\gamma_2, \ldots, \gamma_{n+p}$. The substitution is justified because the linear combination of the free and nonnegative decision vari-

ables $\gamma_2, \ldots, \gamma_{n+p}$ in (4.15) provides a feasible value of γ_1. After the substitution, the i^{th} constraint equation is redundant and may be removed from the linear program as well. Thus we obtain a linear program expressible in canonical form having $m-1$ decision variables and $n-1$ constraint equations. The reduced linear program is obtained by applying this substitution, in turn, to each free decision variable in (4.6).

The following example illustrates this approach. Consider a set of five sample points arising from a binary logistic regression model fit to a single two-level factor variable with one success and one failure at the base level and two successes and one failure at the treatment level. The response vector and design matrix (including an intercept term) computed using treatment contrasts are given by

$$y = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{and} \quad X = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}.$$

Substituting these values of X and y into (4.6) gives rise to the linear program described by the simplex tableau in table (4.1a). The first step is to eliminate β_1. This is accomplished by pivoting on the top-left element of the simplex tableau; doing so yields the simplex tableau in table (4.1b). The first row and first column in table (4.1b) are grayed out to indicate that no further changes should be made to these elements of the simplex tableau. The remaining elements in the simplex tableau form a sub-tableau representing a linear program with six decision variables and four constraint equations. The remaining free decision variable β_2 can be eliminated by pivoting on the top-left element of the sub-tableau. Doing so yields the simplex tableau in table (4.1c).

a)

β_1	β_2	s_1	s_2	s_3	s_4	s_5	b
1	0	1	0	0	0	0	0
1	1	0	1	0	0	0	0
−1	0	0	0	1	0	0	0
−1	−1	0	0	0	1	0	0
−1	−1	0	0	0	0	1	0
1	1	0	0	0	0	0	0

b)

β_1	β_2	s_1	s_2	s_3	s_4	s_5	b
1	0	1	0	0	0	0	0
0	1	−1	1	0	0	0	0
0	0	1	0	1	0	0	0
0	−1	1	0	0	1	0	0
0	−1	1	0	0	0	1	0
0	1	−1	0	0	0	0	0

c)

β_1	β_2	s_1	s_2	s_3	s_4	s_5	b
1	0	1	0	0	0	0	0
0	1	−1	1	0	0	0	0
0	0	1	0	1	0	0	0
0	0	0	1	0	1	0	0
0	0	0	1	0	0	1	0
0	0	0	−1	0	0	0	0

Table 4.1: Eliminating the free decision variables through substitution: a) the initial simplex tableau, β_1 and β_2 are free; b) the simplex tableau after pivoting on a_{11}; c) the simplex tableau after pivoting on a_{22} ignoring the first row and first column of the full tableau.

The sub-tableau in table (4.1c) describes the linear program

$$\begin{aligned} \text{minimize:} \quad & 0s_1^* - 1s_2^* \\ \text{subject to:} \quad & \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} s_1^* \\ s_2^* \end{bmatrix} \leq \tilde{b} \\ & s_1^* \geq 0, \quad s_2^* \geq 0 \end{aligned} \tag{4.16}$$

where $\tilde{b} = (0, \ldots, 0)^T$. The values of the original decision variables β_1 and β_2 can

be expressed as the linear combinations of s_1^* and s_2^*

$$\beta_1 = -s_1^* \quad \text{and} \quad \beta_2 = s_1^* - s_2^*$$

which are apparent from the top two rows of table (4.1c). Hence, an optimal solution to the primal linear program can be obtained from the optimal solution of the reduced linear program.

Implementation Details

We now describe precisely the algorithm to convert the primal linear program (4.6) to its reduced form (4.14). The structure of the initial simplex tableau describing the primal linear program is shown in figure 4.2.

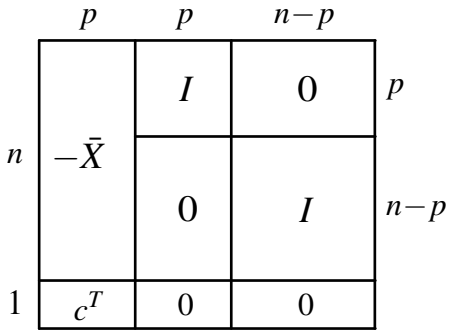


Figure 4.2: The structure of the initial simplex tableau for the primal linear program.

The aim is to pivot on each of the first p diagonal elements of this simplex tableau in column order.

First, observe that since the top-right block of the simplex tableau is composed entirely of zeros, no pivot carried out in the first p rows will have any affect on the identity matrix in the middle-right block or the objective coefficients in the bottom-right block. Therefore it is sufficient to consider only the first $2p$ columns

of the simplex tableau; the memory requirement is thus $2p(n+1)$ double precision values. We call this $(n+1) \times 2p$ block of the simplex tableau the *skinny tableau*.

Since the design matrix X is assumed to include an intercept term there will be either a 1 or a -1 in the top-left element of the skinny tableau. We can therefore always pivot on this element. However, it is possible that there may be a zero in the diagonal element of the j^{th} column when a pivot is attempted. In this case, find a row k with $j < k \leq n$ such that $a_{kj} \neq 0$ then swap rows k and j. If there is no such k then the model matrix X is not of full rank which violates our assumption. Hence, constructing the reduced linear program also provides a check for the linear independence of the columns of the model matrix.

When swapping rows to introduce a nonzero value in j^{th} diagonal element of the skinny tableau, it is only necessary to swap the elements in columns j to $p+j-1$ of rows j and k. The first $j-1$ elements in each of these rows are already zero because of the first $j-1$ pivots. The last $p-j$ elements in rows j and k are similarly unaffected by the first $j-1$ pivots since the upper-right $j \times (p-j)$ block of the skinny tableau contains only zeros. Note that swapping rows implies that s_j becomes the slack variable for what was originally the k^{th} sample point. This change in labeling has no effect on the outcome of the linear program.

Similarly, each pivot only needs to be carried out in a subset of the columns. The elements below the diagonal in the block of the skinny tableau initially holding $-\bar{X}$ are not referenced after the pivoting is complete, so it is not necessary to compute their values. Further, the j^{th} row of the skinny tableau, after $j-1$ pivots, has a one in position $p+j$ and zeros in positions $p+j+1$ through $2p$. Thus, it is only necessary for the j^{th} pivot to write into columns $j+1$ through $p+j+1$. This optimization cuts the number of operations necessary to obtain the reduced linear

program roughly in half.

The algorithm for computing the reduced linear program can be summarized as follows.

1. Set $j := 1$.
2. If $a_{jj} = 0$ then find k $(j < k \leq n)$ such that $a_{kj} \neq 0$; swap the elements of rows j and k in columns from j to $(p-j-1)$. If no such k exists then report that the design matrix is not of full rank.
3. Pivot on a_{jj} updating the values only in rows j through $n+1$.
4. If $j = p$ stop, the sub-tableau contains the reduced linear program. Otherwise, set $j := j + 1$ and go to step 2.

When the reduction algorithm is complete the tableau has the structure shown in figure 4.3.

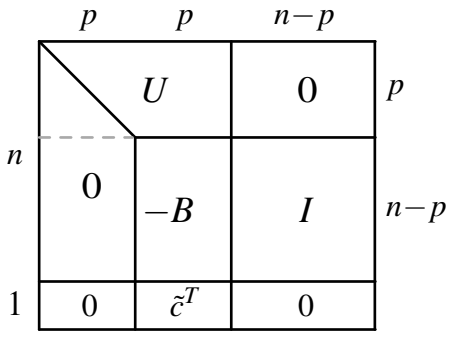


Figure 4.3: The structure of the reduced simplex tableau.

A vector β optimal for the primal linear program can be obtained from the reduced linear program when the reduced linear program has a finite optimal solution. Because the left $p \times p$ block of the matrix U is upper triangular, an optimal β can be computed from an optimal s_1 using backwards substitution. The general expression

for β_j is given by

$$\beta_j = -u_{j,j+1}\beta_{j+1} - \cdots - u_{j,p}\beta_p - u_{p,p+1}s_1 - \cdots - u_{p,2p}s_p$$

and the original vector β can be obtained by first solving for β_p, then β_{p-1}, and so on.

4.4.2 Pivoting Rules for the Revised Simplex Method

The description of the revised simplex method given in (Luenberger, 1984, chapter 3, section 7) describes how, for a given basic feasible solution, to identify the eligible nonbasic variables which may become basic and, once a nonbasic variable has been selected to enter the basis, how to identify the basic variables that are eligible to leave the basis. The revised simplex method itself does not uniquely determine the pivot but rather a family of eligible pivots. The method of choosing a specific pivot is called a pivoting rule and can greatly affect the performance of the revised simplex algorithm (Maros, 2001). In particular, a poor choice of pivoting rule can lead to a phenomenon known as cycling in which the revised simplex method fails to terminate.

In general, the revised simplex method permits any nonbasic variable with a negative reduced cost coefficient to enter the basis (Dantzig and Thapa, 1997). This criterion is more general than that given in Luenberger (1984) which selects the nonbasic variable with the most negative reduced cost coefficient. Having selected a nonbasic variable γ_q which is to enter the basis, the revised simplex method then permits the

selection of any basic variable γ_i such that

$$\frac{a_{i0}}{a_{iq}} = \min\left\{\frac{a_{i0}}{a_{iq}} : a_{iq} > 0\right\} \tag{4.17}$$

to leave the basis. If $a_{iq} \leq 0$ for $i = 1, \ldots, n$ then the linear program is unbounded. Note that the right-hand sides of both the primal linear program (4.6) and the reduced linear program (4.14) are identically zero. Given q, the set of indices I satisfying (4.17) is the set such that $a_{iq} > 0$ when $i \in I$. Hence the set of eligible pivots allowed by the revised simplex method can be quite large.

A pivoting rule that is consistent with the revised simplex method and further restricts the choice of either the pivot column or the pivot row is called a refinement of the simplex method (Bland, 1977). We adopt Bland's terminology and say that a refinement determines *a simplex method*, as opposed to *the simplex method* which refers to the family of methods determined by all possible refinements. Clearly, any useful refinement must give a unique choice of pivot column and pivot row for any given non-optimal basic feasible solution.

Degeneracy and Cycling

A basic feasible solution is said to be degenerate if one or more of the basic variables has the value zero. We additionally use the terminology fully degenerate to describe a basic feasible solution in which all the basic variables are zero. Since the right-hand side of the primal linear program (4.6) and the right-hand side of the reduced linear program (4.14) are both identically zero, it follows that the basic variables in both linear programs are zero as well. Hence their initial basic feasible solutions are fully degenerate.

Given a non-degenerate basic feasible solution, Dantzig and Thapa (1997) show that any pivot allowed by the simplex method causes a decrease in the value of the objective function. However, when the basic feasible solution is degenerate, it is possible (for a fully degenerate basic feasible solution it is guaranteed) that the minimum ratio in (4.17) is zero in which case the pivot has no effect on the value of the objective function. When a pivot causes no change in the value of the objective function, the new basic feasible solution obtained is degenerate as well. It is conceivable then that a sequence of pivots could return the simplex tableau to a previously achieved basic feasible solution. This phenomenon is called a cycle. When cycling occurs the revised simplex method fails to terminate.

Cycling is a real problem and can occur in even small linear programs. For example, Papadimitriou and Steiglitz (1982) show that the linear program

$$\begin{array}{rrcrcrcrcl} \text{minimize:} & -\frac{3}{4}x_1 & + & 20x_2 & - & \frac{1}{2}x_3 & + & 6x_4 & & \\ \text{subject to:} & \frac{1}{4}x_1 & - & 8x_2 & - & 1x_3 & + & 9x_4 & \leq & 1 \\ & \frac{1}{2}x_1 & - & 12x_2 & - & \frac{1}{2}x_3 & + & 3x_4 & \leq & 0 \\ & 0x_1 & + & 0x_2 & + & 1x_3 & + & 0x_4 & \leq & 0 \end{array} \tag{4.18}$$

with $x_j \geq 0$ for $j = 1, 2, 3, 4$ cycles when the following refinement to the simplex method is used as the pivoting rule.

- Choose the pivot column j such that r_j is the most negative.
- In the case of a tie when choosing the pivot row, choose the row with the smallest index.

Since the initial basic feasible solutions for both the original linear (4.6) program and the reduced linear program (4.14) are fully degenerate the possibility that a cycle will develop is quite high if an inappropriate refinement is used. A refinement that guarantees the revised simplex method will terminate after a finite number of

pivots and hence precludes the possibility of cycling is given in the next section.

Bland's Rule

It is well known that the simplex method can fail to terminate due to the possibility of cycling. Several rules have been suggested, for instance the lexicographic rule (Dantzig, 1963), that restrict the selection of the pivot row in such a way that cycling cannot occur. Bland (1977) describes a rule which restricts both the choice of the pivot column and the pivot row by imposing the following two restrictions on the revised simplex method.

- Select the column to enter the basis by $j = \min\{j \ : \ r_j < 0\}$; i.e., select the decision variable with the smallest index from among those having negative reduced cost coefficients.
- In the case that ties occur in the criterion for determining the pivot row, select the row with the smallest index.

Bland (1977) shows that the simplex method, and hence the revised simplex method, terminates in a finite number of iterations when this refinement is used as a pivoting rule.

4.4.3 Testing for Separation

In section 4.2 we showed that if there is overlap among the sample points then the optimal value of the primal linear program (4.6) is zero. If the optimal value of (4.6) is greater than zero then there is a nonzero vector β feasible for (4.6) and giving either complete or quasicomplete separation among the sample points. Thus, to test for separation, it is sufficient to check whether the optimal value of (4.6) is

greater than zero. This often requires less work than finding a vector revealing the separation. Our test is based on the following remark.

Remark 1. *The linear program in (4.6) either has an optimal value of zero or is unbounded.*

Proof. Consider the following two cases.

Case 1: There is a nonzero point β' feasible for (4.6), then $\bar{X}\beta' \geq 0$ (with at least one element strictly greater than zero) and $e_n^T \bar{X}\beta' > 0$. For $k > 0$ the point $k\beta'$ is also feasible since $k\bar{X}\beta' \geq 0$, thus $k\beta'$ is feasible for all $k > 0$ and $ke_n^T \bar{X}\beta'$ grows unbounded as $k \to \infty$.

Case 2: $\beta \equiv 0$ is the only point feasible for (4.6); the optimal value of the objective function is $e_n^T \bar{X}\beta \equiv e_n^T \bar{X}0 = 0$.

□

If there is either complete or quasicomplete separation among the sample points then there is a nonzero β feasible for the primal linear program (4.6) hence it is unbounded. If there is overlap among the sample points then $\beta \equiv 0$ is the only point feasible for the primal linear program hence its optimal value is zero.

Testing the Boundedness of the Primal Linear Program with the Revised Simplex Method

We first consider solving the primal linear program (4.6) directly using the revised simplex method. Assume that the free decision variables have been represented as the difference between two nonnegative decision variables as in (4.13) so that the standard form is expressed in $2n+2p$ decision variables and $2n$ constraint equations.

The revised simplex method allows a pivot in any column corresponding to a decision variable with a negative reduced cost coefficient (Dantzig and Thapa, 1997). Let $J = \{j : r_j < 0\}$ be the set of indices for which the reduced cost coefficient is negative. The linear program is unbounded if there is $j \in J$ such that $B^{-1}D_{ij} \leq 0$ for $i = 1, \ldots, 2n$ and where B is the current basis. Normally, when using the revised simplex method, we would choose a pivot column then declare the linear program unbounded only if all the elements in that specific column are less than or equal to zero. However, a more thorough test for an unbounded solution is also possible. During each iteration of the revised simplex method if

$$\max(B^{-1}D_{\bullet j}) \leq 0$$

holds for any $j \in J$ then the linear program is unbounded. If there is no pivot giving an unbounded solution and the current basic feasible solution is not optimal then pivot according to Bland's rule and repeat this test on the new basic feasible solution. This algorithm must yield either an unbounded solution or an optimal solution of zero in a finite number of pivots. When an unbounded solution exists this method is likely to require fewer pivots to detect it than the standard revised simplex method.

Testing for boundedness in the primal linear program (4.6) requires using this method on a linear program expressible in $2p + 2n$ decision variables and $2n$ constraint equations when written in the standard form. Since all of the constraints in the primal linear program are inequality constraints, the $2n$ slack variables provide an initial basic feasible solution with the identity matrix providing the initial basis. Further, since the right-hand side is identically zero this initial basic feasible solution is fully degenerate so we use Bland's rule as the pivoting rule.

Example

The following example demonstrates how the revised simplex method detects an unbounded solution. Let

$$y = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \quad X = \begin{bmatrix} 1 & -1 \\ 1 & a \\ 1 & -a \\ 1 & 1 \end{bmatrix}$$

represent a set of 4 sample points. There is overlap among the sample points if $a > 0$, quasicomplete separation if $a = 0$ and complete separation if $a < 0$; also we assume $|a| \leq 1/2$ to avoid an ambiguity. The relative cost coefficients for the initial basic feasible solution are

$$r^T = [\, 0 \quad -2+2a \quad 0 \quad 2-2a \,].$$

Because $-2 + 2a$ is the only negative reduced cost coefficient, the second column of $-\tilde{X}$ (corresponding to β_2^+) is chosen to enter the basis. Since

$$(B^{-1}D_{\bullet,2})^T = [\; -1 \quad a \quad a \quad -1 \quad 0 \quad 0 \quad 0 \quad 0 \;]$$

the solution is bounded if and only if $a > 0$, i.e., when there is overlap among the sample points.

Testing for boundedness in the reduced linear program (4.14) requires applying this method to a linear program expressible in n decision variables and $n - p$ constraint equations when written in the standard form. Again, since all of the constraints in the reduced linear program are inequality constraints, the slack variables provide an initial basic feasible solution with the identity matrix providing the initial

basis. The reduced linear program is fully degenerate since the right-hand side is identically zero, thus Bland's rule should again be used as the pivoting rule. Note that converting the primal linear program into the reduced linear program requires $O(np^2)$ floating point operations.

Testing for Boundedness: Dual Feasibility

An alternative approach for checking the boundedness of the primal linear program (4.6) is provided by the duality theorem of linear programming.

Theorem 6 (Duality Theorem of Linear Programming)**.** *If either the primal linear program or the dual linear program has a finite optimal solution, so does the other, and the corresponding values of the objective functions are equal. If either problem has an unbounded objective, the other problem has no feasible solution.*

Thus we can determine whether the primal linear program (4.6) is bounded by checking whether the dual linear program is feasible. Using the rules for the duals of mixed systems given in Dantzig and Thapa (1997) we can write the dual linear program to (4.6) as

$$\begin{aligned} \text{minimize:} \quad & -b^T\lambda \\ \text{subject to:} \quad & \bar{X}^T\lambda = -\bar{X}^T e_n \\ & \lambda \geq 0 \end{aligned} \tag{4.19}$$

where λ is a vector of n dual decision variables and $b = (0, \ldots, 0)^T$. It is clearly preferable to work with the dual linear program since it is already in the standard form; that is, all of the constraints are equality constraints and the decision variables are nonnegative.

To test the feasibility of (4.19) it is only necessary to carry out phase one of the

two-phase algorithm. Let $v = \bar{X}^T e_p$ and let M be a $p \times p$ diagonal matrix with $M_{ii} = -1$ if $v_i < 0$ and $M_{ii} = 1$ otherwise. The phase one linear program is given by

$$\begin{aligned} \text{minimize:} \quad & e_p^T u + 0^T \lambda \\ \text{subject to:} \quad & \begin{bmatrix} I & -M\bar{X}^T \end{bmatrix} \begin{bmatrix} u \\ \lambda \end{bmatrix} = \tilde{v} \\ & \lambda \geq 0 \\ & u \geq 0 \end{aligned} \tag{4.20}$$

where $\tilde{v} = Mv$ and u is a vector of p artificial variables. The point $(u = \tilde{v}, \lambda = 0)$ provides an initial basic feasible solution with the identity matrix as the initial basis. When (4.19) is feasible, (4.20) has an optimal value of zero and when (4.19) is infeasible the optimal value of (4.20) is strictly greater than zero.

Since the objective function coefficients in the dual linear program (4.19) are identically zero, any basic feasible solution is an optimal basic feasible solution. Thus, if we use the two-phase algorithm to solve (4.19), the phase two problem is trivial. Hence we can determine whether (4.19) is feasible simply by solving it with lp_solve. Since there is no apparent initial basic feasible solution lp_solve will transparently solve the phase one problem. If the optimal value of (4.20) is greater than zero, lp_solve will report that (4.19) is infeasible. Otherwise, lp_solve will return the solution to the phase two problem which is trivially zero.

Some care is needed since lp_solve by default uses the dual simplex method to solve the phase one problem. Doing so essentially converts the dual linear program (4.19) to its primal form (4.6) before attempting its solution. Clearly this is not what we want. Thus we must provide the control argument

```
control = list(simplex.type = c("primal", "primal"))
```

to lp_solve to specify that the primal simplex method be used for both phases of the two-phase algorithm.

Testing for boundedness using the method of dual feasibility requires solving a linear program whose standard form representation requires $n + p$ decision variables and p constraint equations. Since the right-hand side vector $\tilde{v}$ is equal to the coefficient vector from the primal linear program up to sign, it is quite likely that the initial basic feasible solution is nondegenerate.

A Modification to the Primal Linear Program for Nondegeneracy

Another approach to the problem of determining whether the primal linear program (4.6) is bounded is suggested by examination of the dual linear program. The set of feasible points for the dual linear program (4.19) depends only on the matrix of constraint coefficients $\bar{X}$. In particular, it does not depend on the vector b containing the right-hand side in the primal linear program. The Duality Theorem of Linear Programming then implies that if (4.6) is bounded when $b = 0$, it is also bounded for any finite choice of the right-hand side b. It follows that the linear program

$$\begin{aligned} \text{maximize:} \quad & e_n^T \bar{X} \gamma \\ \text{subject to:} \quad & \bar{X}\gamma \geq -1 \\ & \gamma \ \text{ free} \end{aligned} \tag{4.21}$$

where $b = (-1, \ldots, -1)^T$ rather than $b = (0, \ldots, 0)^T$ is bounded when the primal linear program (4.6) is bounded and unbounded when the primal linear program (4.6) is unbounded. The notation used for the decision variables is changed from β to γ because the solution vector can not be interpreted in the parameter space of the underlying logistic regression model. We call (4.21) the *modified linear program.*

A similar modification can be made to the reduced linear program by setting $\tilde{b} = (-1, \ldots, -1)^T$. This yields

$$\begin{aligned} \text{maximize:} \quad & \tilde{c}^T s_1 \\ \text{subject to:} \quad & Bs_1 \geq -1 \\ & s_1 \geq 0 \end{aligned} \tag{4.22}$$

which we call the *modified reduced linear program.*

The modified linear program is expressible in the standard form using $2n + 2p$ decision variables and $2n$ constraint equations. The initial basic feasible solution $\gamma^* = 0$ with the identity matrix providing the initial basis is nondegenerate.

The modified reduced linear program is expressible in the standard form using n decision variables and $n - p$ constraint equations. The initial basic feasible solution $s_1 = 0$ with the identity matrix providing the initial basis is nondegenerate.

By default lp_solve uses the Devex pricing strategy (Harris, 1973) for pivot selection which reduces the number of simplex pivots quite considerably (Maros, 2001). Solving (4.21) using the Devex pivoting rule in lp_solve is likely to require fewer simplex pivots than solving (4.6) using Bland's rule.

Alternatively, the modified linear programs can be solved using the interior point method provided by the MATLAB® `linprog` function. The points $\gamma = 0$ and $s_1 = 0$ respectively lie strictly in the interiors of the modified linear program and modified reduced linear program and hence provide initial feasible points. A simulation based timing comparison between the simplex method and this interior point method is given in section 4.6.

4.4.4 Finding the Direction of Separation

Finding a vector β revealing either complete or quasicomplete separation among the sample points requires only a trivial change to the primal linear program. Finite upper and lower bounds are placed on the decision variables so that the linear program remains bounded when separation is present. This results in the linear program

$$\begin{aligned} \text{maximize:} \quad & e_n^T \bar{X}\ \beta \\ \text{subject to:} \quad & \bar{X}\beta \geq 0 \\ & -1 \leq \beta \leq 1 \end{aligned} \tag{4.23}$$

where the notation $-1 \leq \beta \leq 1$ means that each element of β is restricted to the interval $[-1, 1]$. The linear program in (4.23) has p decision variables and n constraint inequalities. According to the lp_solve documentation, when there is a finite lower bound on a decision variable β_j, lp_solve internally makes the substitution $\beta_j = w_j + l_j$ so that the linear program is represented internally in the nonnegative decision variables w_j. Making this substitution in (4.23) yields the linear program

$$\begin{aligned} \text{maximize:} \quad & e_n^T \bar{X}\ w \\ \text{subject to:} \quad & \bar{X}w \geq \bar{X}e_p \\ & 0 \leq w \leq 2 \end{aligned} \tag{4.24}$$

where the constant term in the objective function has been omitted. This linear program can easily be solved using the revised simplex method for bounded decision variables (Luenberger, 1984).

While the point $w = e$ is clearly feasible for (4.24), there is no immediately apparent

initial basis. Therefore, to solve (4.24), lp_solve must solve both phases of the two-phase algorithm. Again, it is quite likely that degenerate basic feasible solutions will be encountered during one or both phases so Bland's rule is used to avoid cycling.

4.5 Numerical Considerations

So far we have assumed that the linear programs can be solved exactly. However, both the revised simplex method provided in lp_solve (Berkelaar et al., 2007) and the interior point method provided in the Matlab® Optimization Toolbox (The MathWorks, 2006) are implemented using the double precision system; representation error and roundoff error may therefore adversely affect the computed solution.

4.5.1 Representation Error

Representation error arises from the fact that numbers which have a finite representation in decimal notation do not necessarily have a finite representation (Higham, 1995, chap. 2) in the double precision system (IEC 60559:1989, 1989). In particular, only fractional numbers that can be written in the form a/b where b is a power of two can be represented exactly in the double precision system. For example, even the seemingly trivial value of 0.3 can not be represented exactly in the double precision system.

It turns out that, for the purpose of separation detection, representation error is not a concern. Let $\tilde{X}$ and $\tilde{y}$ be the input design matrix and binary response vector and let X and y be respectively their representations in the double precision system. First of all, $\tilde{y} \equiv y$ since $\tilde{y} \in \{0, 1\}$ and both zero and one have exact representations

in double precision. Second, the software used to fit the binary logistic regression model is going to represent the design matrix and response vector using the double precision system. Hence, the real concern is whether there exists separation among the sample points described by y and X since these are the values to which the model will be fit. This is precisely the separation that our method tests for.

4.5.2 Rounding Error

Rounding error is the difference between the calculated result of a mathematical operation represented in the double precision system and true result of the mathematical operation. A brief description of rounding error is given in Björck (1996) and a thorough discussion of floating point arithmetic can be found in chapter 2 of Higham (1995). Rounding error arises because a computer is only capable of representing a finite subset of the real numbers. For our purposes, this subset is the set of numbers that may represented exactly in the double precision system. When a computer performs a mathematical operation on two double precision numbers x and y the result must also be stored in the double precision system. That is

$$dps(x \circ y) = (x \circ y)(1 + \delta), \qquad |\delta| \leq u,$$

where $dps(v)$ represents the double precision approximation of the real number v, $\circ$ is one of $\{+, -, \times, \div\}$, and u is the unit roundoff – the maximum relative error in storing a number in the double precision system. The difference $\delta(x \circ y)$ between the true result and the computed result is the rounding error.

In the revised simplex method, the eligible pivot columns are those with negative reduced cost coefficients and, given the pivot column j, the eligible pivot rows are

those such that $B^{-1}D_{\bullet j} < 0$ where B is the current basis. If the design matrix contains a large number of zero elements – this would be the case for instance if the design matrix contains several coded factor variables – then even a small amount of rounding error could cause the revised simplex method to choose an illegal pivot. The lp_solve library (Berkelaar et al., 2007) therefore uses a set tolerance parameters to specify when a double precision value should be considered different than zero. The relevant tolerance parameters are

ϵ_b the tolerance for the right-hand side, if $|b_i| < \epsilon_b$ then b_i is considered to be zero;

ϵ_d the tolerance for the reduced cost coefficients, if $|r_j| < \epsilon_d$ then r_j is considered to be zero;

ϵ_{el} the default tolerance, this value is used when no other tolerance specifically applies;

ϵ_{pivot} the tolerance for the pivot element, if the absolute value of the candidate pivot variable is less than ϵ_{pivot} then it is rejected as a pivot element.

The lp_solve API includes a routine called `epslevel` which sets all of the tolerance parameters to one of four internally consistent levels. On the recommendation of the authors (Berkelaar et al., 2007), we set `epslevel` to "baggy" which is the least tight level for the tolerances. The loss in accuracy caused by this setting is not important since we are generally only interested in deciding whether a linear program is either bounded or feasible.

In general we have found that lp_solve solves the primal linear program quite reliably. Since the primal linear program is fully degenerate, the basic decision variables and the slack variables have value zero. Further, since the right-hand side is zero as well, the eligible pivot rows are identified only by the signs of the respective elements of

the pivot column. The tolerances in lp_solve allow this to be done reliably. The modified linear program (i.e., the right-hand side $b = 1$) does not share this property and from time to time the modified linear program returns an incorrect result or terminates prematurely due to numerical instability. The modified linear program fails often enough that we did not include it in the simulation study given in the next section and do not recommend its use for separation detection.

Testing for separation using the dual linear program can be done reliably using lp_solve. The test for separation using the dual linear program only involves determining whether the dual is feasible. Solving phase one of the dual linear program either provides a feasible point or shows that no feasible point exists. In the case that phase one returns a feasible point, it can be checked by evaluating the constraints of the dual linear program. If the point is in fact feasible then we can say with high reliability that there is overlap among the sample points and that the maximum likelihood estimate of the binary logistic regression model parameter vector β exists.

The reduced linear program can be solved as reliably as the primal linear program. However, it is possible that rounding error accumulated while computing the reduced form of the primal linear program can alter the feasible region so that the solution of the reduced linear program no longer corresponds to the optimal value of the primal linear program. Hence, in the general case, we find the reduced linear program unreliable for separation detection. However, the design matrix is composed entirely of ones and zeros when it is generated by coding only factor variables. In this case it is sometimes possible to compute the reduced linear program without accumulating any rounding error by performing each pivot on a one or minus one. When this can be done, the numerical reliability of the reduced linear program is

similar to that of the primal linear program.

4.6 Simulation Study

The revised simplex method has worst-case exponential complexity. For example, Klee and Minty (1972) show that for every integer $d > 1$ there is a linear program with $2d$ constraint equations, $3d$ decision variables and integer constraints whose absolute value is bounded by 4, such that the simplex method may take $2^d - 1$ pivots before reaching the optimal value of the objective function. However, in practice the actual number of pivots required to solve a linear program using the simplex method tends to be many times smaller. Hence the theoretical upper bound on the number of pivots is often a misleading benchmark for the actual performance of the simplex method (Dantzig and Thapa, 1997).

Simulation methods were therefore used to assess the performance of the tests for separation. The simulations involving the revised simplex method were carried out in the R environment (R Development Core Team, 2007) and the simulations involving the interior point method were done in Matlab®. All simulations were run on the same machine, a 2.00 GHz Intel® XEON™ with 3.5 GB of RAM (this amount far exceeded the maximum required to compute any of the simulations) running Fedora-Core 5 Linux.

Each simulation consisted of randomly generating a set of n sample points with p variables represented by an $n \times p$ design matrix X and a binary response vector y containing n elements. The binary response vector y was simulated using the `sample` function with `prob = 0.5` and the design matrix X was simulated using the following steps.

1. Simulate four 4-level factor vectors each of length n using the `sample` function.
2. Generate an $n \times 256$ design matrix including an intercept term, the main effects and all possible interaction terms using the `model.matrix` function.
3. Set X to be the first p columns of this saturated design matrix.

The time taken to test for separation or compute a vector giving separation among the simulated sample points was then measured. Also, since the overall goal is to produce a test fast enough to be used routinely as part of the fitting process of a binary logistic regression model, the time taken to estimate the model parameter vector using the method of iteratively reweighted least squares (IRLS) was measured as well to serve as a benchmark.

Note that this method of simulation takes $P(Y_i = 1 \mid x_i) = P(Y_i = 1) = 0.5$ for $i = 1, \ldots, n$; that is, the distribution of Y given X is the same as the distribution of Y. Since the linear programming algorithms compared in this simulation study are not influenced by the underlying logistic regression model nor the presence of separation, the simplest possible model was used to generate the data. On the other hand, the IRLS algorithm used to compute the maximum likelihood estimates takes considerably longer when separation is present among the sample points. This property manifests itself as the kink seen in the IRLS curve in several of the figures in this section.

4.6.1 Testing for Separation with the Revised Simplex Method

The simulations testing for separation using the revised simplex method used the implementation provided in the lp_solve library (Berkelaar et al., 2007) through the

interface presented in section 4.1.1. Two simulation studies were conducted. The first to assess the computational complexity in the number of sample points and the second to assess the computational complexity as the number of columns in the design matrix is increased.

Computational Complexity versus the Number of Sample Points

Two simulations were conducted to investigate the complexity of the tests for separation in terms of the number of sample points. For each value of n from 1000 to 10000 in lots of 1000, twenty-five simulations were conducted with $p = 50$ and a further twenty-five with $p = 250$. The tests for separation based on the primal, the dual and the reduced linear programs were applied to each simulated X and y and the user time reported by the function `unix.time` was recorded. Since we are interested in the worst-case complexity, the longest of the twenty-five times for each test was taken. We call this the *empirical worst-case time* based on twenty-five simulations. The results from these simulations are shown in figure 4.4 and figure 4.5.

Figure 4.4 shows that, when $p = 50$, the empirical worst-case time grows roughly linearly in the number of sample points for all three methods. The empirical worst-case time for the test based on the dual linear program is approximately equal to the empirical worst-case time taken to fit the underlying model using the IRLS method. The tests based on the primal linear program and the reduced linear program take anywhere from two to three times longer over the range of sizes considered in the simulation.

Figure 4.5 shows that, when $p = 250$, the empirical worst-case times for both the test based on the dual linear program and the test based on the reduced linear

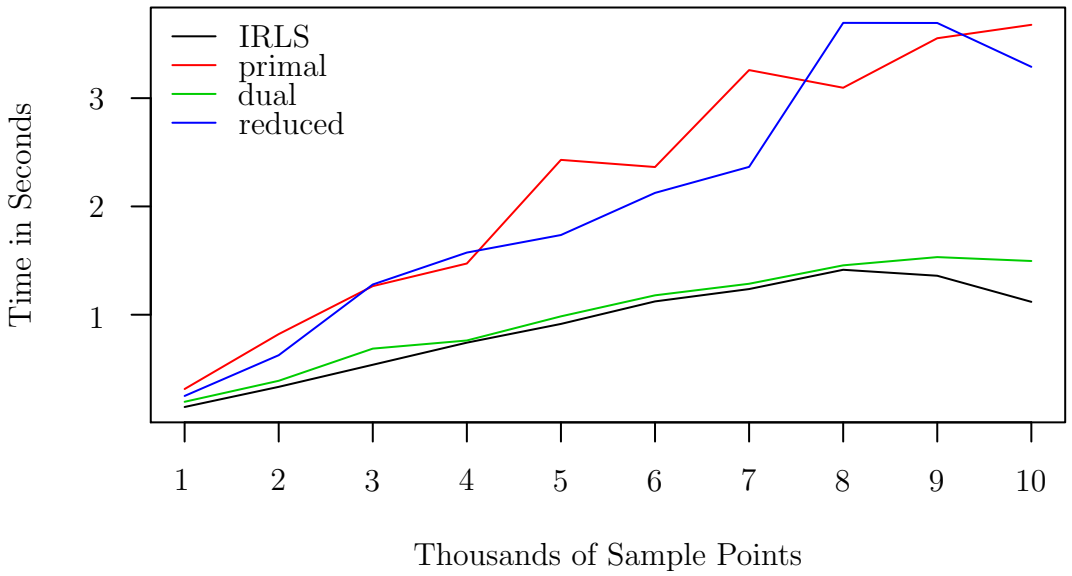


Figure 4.4: Empirical worst-case computing time versus number of sample points for the 50 column design matrix.

program are approximately equal to the empirical worst-case time taken by the IRLS method. The test based on the primal linear program is again slower, this time by a factor of between three and four. The kink in the IRLS curve was caused by the presence of separation in at least one of the simulated sets of sample points. When separation was present among the sample points, significantly more IRLS iterations were required to fit the binary logistic regression model. Separation was not present in any of the simulated sets of sample points for $n \geq 5000$.

Computational Complexity versus the Number of Columns in the Design Matrix

Two additional simulation studies were conducted to investigate the complexity of the tests for separation in terms of the number of columns in the design matrix. For each value of p from 16 to 256 in lots of 16, twenty-five simulations were conducted with $n = 2000$ and a further twenty-five with $n = 4000$. The tests for separation

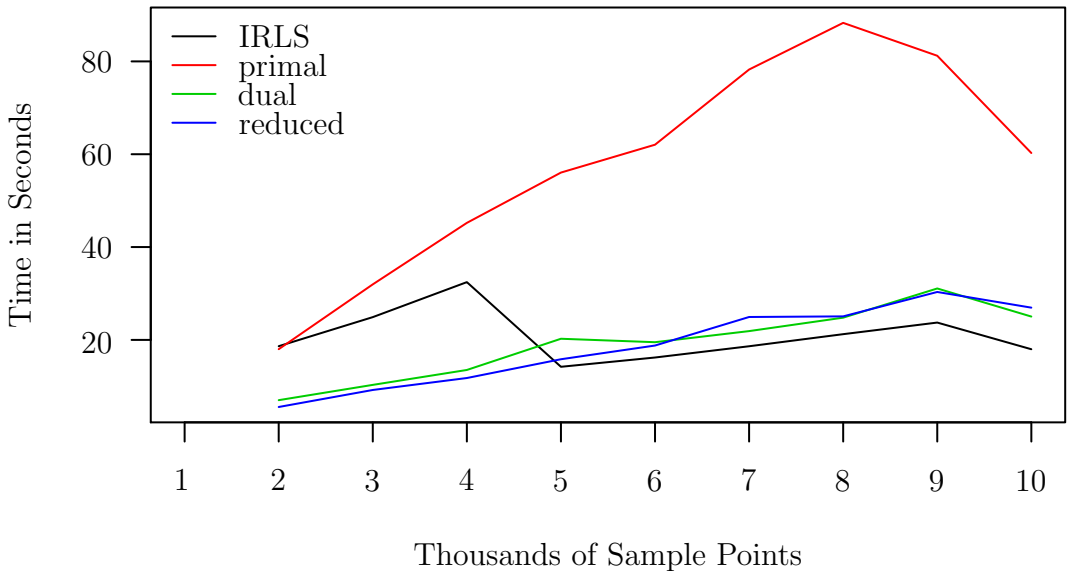


Figure 4.5: Empirical worst-case computing time versus number of sample points for the 250 column design matrix.

based on the primal, the dual and the reduced linear programs were applied to each simulated X and y and the user time reported by the function `unix.time` was recorded. The results are shown for $n = 2000$ in figure 4.6 and for $n = 4000$ in figure 4.7.

Figure 4.6 shows that for the case $n = 2000$, the empirical worst-case time taken to test for separation among the sample points scales roughly linearly in the number of columns in the design matrix for all three tests. For $p \leq 160$ separation was not present in any of the simulations. In these cases, the empirical worst-case times for the tests based on the dual and reduced linear programs are approximately equal to the worst-case time necessary to fit the model using IRLS. Separation was present among the sample points in at least one of each of the simulations for each value of $p \geq 176$. In these cases the empirical worst-case times to test for separation using the tests based on the dual and reduced linear programs continue their trends while the empirical worst-case time to fit the model using IRLS increases significantly.

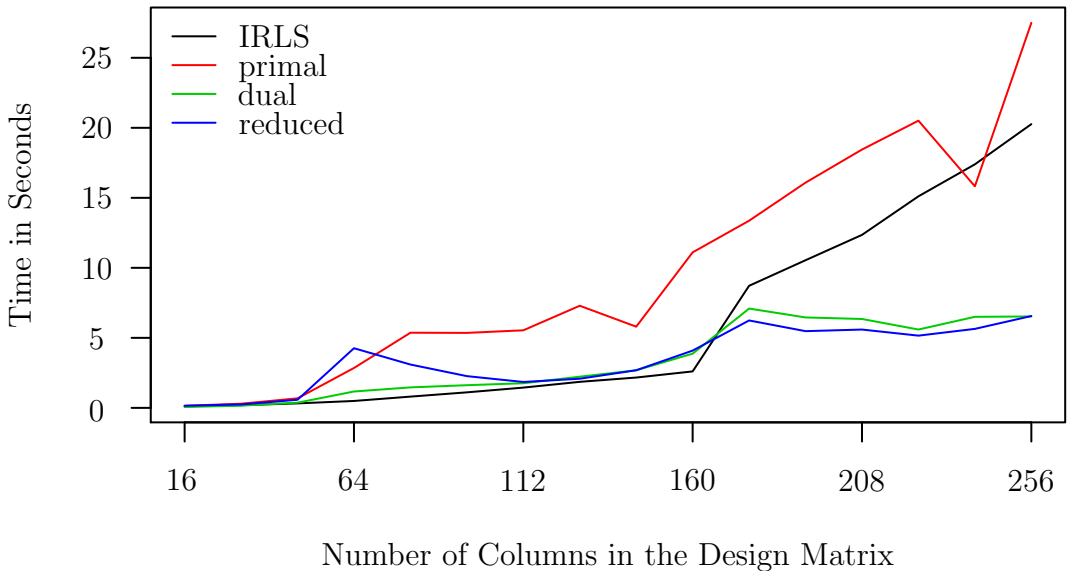


Figure 4.6: Empirical worst-case computing time versus the number of columns in the design matrix; 2000 sample points were used.

The test based on the primal linear program underperforms the others by a factor of between three and five.

The simulations summarized in figure 4.7 again show that, for the case $n = 4000$, all three tests scale roughly linearly in the number of columns in the design matrix. The kink in the IRLS curve corresponds to the point where separation begins to occur in the simulated sample points. The tests for separation based on the dual and reduced linear programs are slightly slower than IRLS when separation is not present and faster when separation is present. The test based on the primal linear program again underperforms the others by a factor of between three and five.

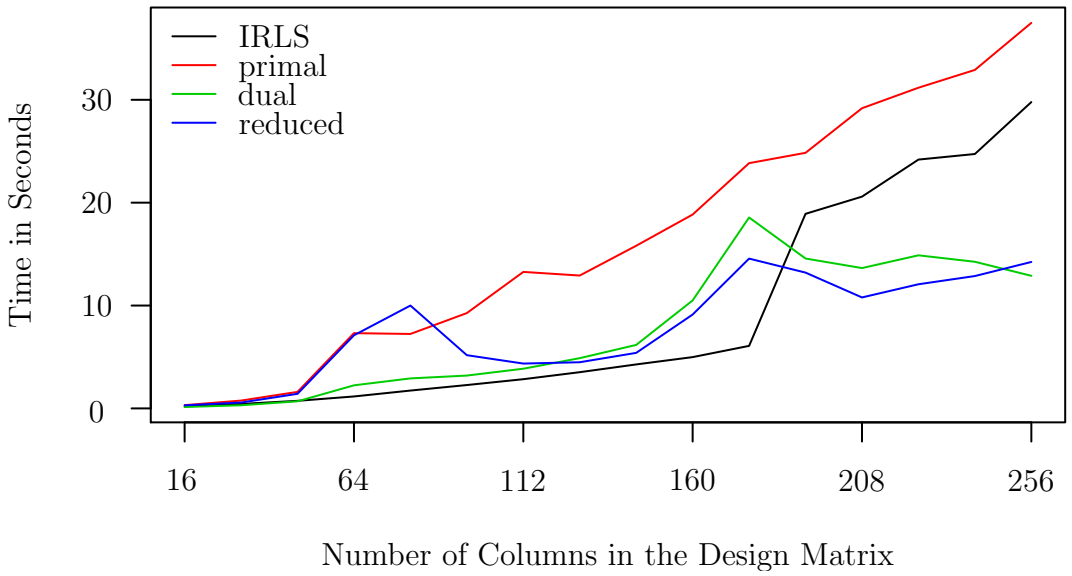


Figure 4.7: Empirical worst-case computing time versus the number of columns in the design matrix; 4000 sample points were used.

4.6.2 Computing the Direction of Separation Using the Revised Simplex Method

A further set of simulations was conducted to assess the computational complexity of the linear programs described in section 4.4.4 for finding a vector revealing separation. The first study was conducted to assess the complexity as the number of sample points is increased. Twenty-five 250 column design matrices were simulated for each value of n ranging from 1000 to 10000 in lots of 1000 and the primal and reduced linear programs were used to compute a vector giving separation. The time reported by the function `unix.time` was recorded for each simulation. The second study was conducted to assess the complexity as the number of columns of the design matrix increased. Twenty-five design matrices were simulated for each value of p ranging from 16 to 256 in lots of 16; 4000 sample points were used. Again, the primal and reduced linear programs were used to compute a vector revealing separation and the time reported by the function `unix.time` recorded.

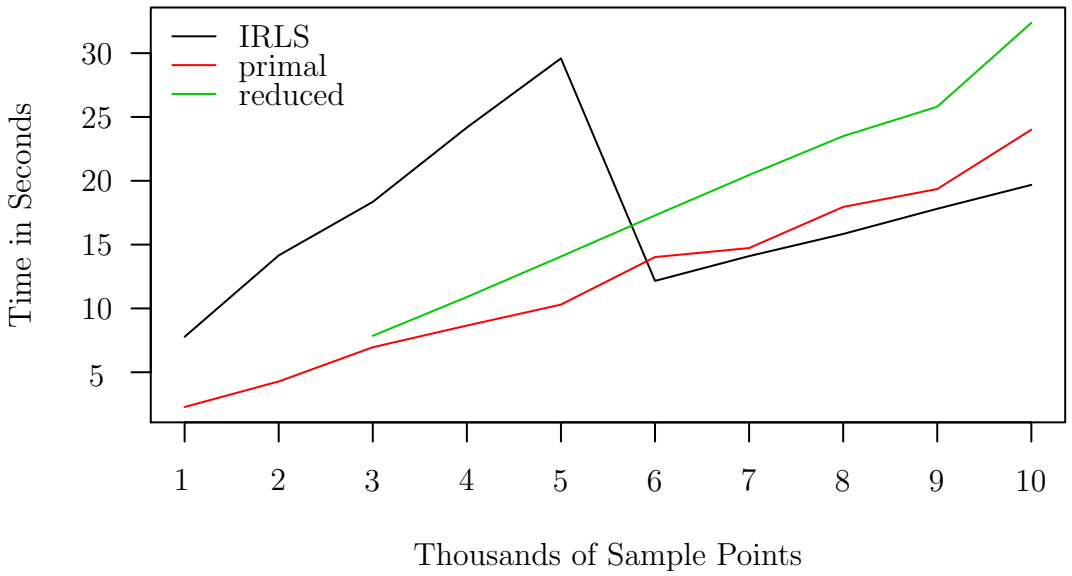


Figure 4.8: A comparison of the time taken to compute a direction revealing separation among the sample points as the number of sample points increases. Each curve shows the empirical worst-case time based on 25 simulations for the respective method.

The empirical worst-case complexity in the number of sample points is shown in figure 4.8 and the empirical worst-case complexity in the number of columns in the design matrix is shown in figure 4.9. The primal linear program and the reduced linear program again appear to scale linearly in both the number of sample points and the number of columns in the design matrix. Also, note that since the basis for the primal linear program using bounded variables is an $n \times n$ matrix (rather than the $2n \times 2n$ matrix necessary for the free variable version used in the tests for separation), the performance is greatly improved.

4.6.3 Testing for Separation Using an Interior Point Method

Finally, we conducted a simulation study to assess the relative performance between the simplex method and an interior point method. These simulations are not intended to be comparable with those given previously. The problems were chosen

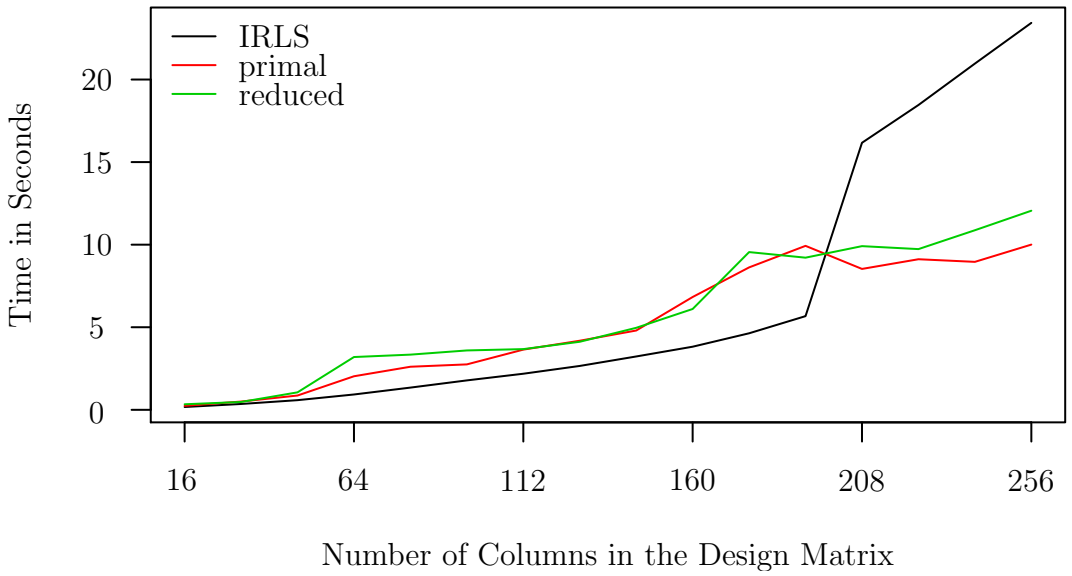


Figure 4.9: A comparison of the time taken to compute a direction revealing separation among the sample points as the number of columns in the design matrix increases. Each curve shows the empirical worst-case time based on 25 simulations for the respective method.

small enough that the modified primal linear program gave reliable results when used to test for separation among the sample points.

The large scale algorithm provided by the `linprog` function in the Matlab® (The MathWorks, 2006) Optimization Toolbox uses an interior point method with polynomial time worst-case complexity. The large scale algorithm is a variant of the predictor-corrector algorithm described in Mehrotra (1992). Although we have observed the approximately linear complexity in both the number of columns in the design matrix and in the number of sample points, we also conducted a simulation study to assess the performance of an interior point method.

For each simulation the binary response vector y and design matrix X were generated in R using the procedure described in the previous section then imported into Matlab®. The `linprog` function was then used to test for separation among the sample points using the test based on the modified linear program with $b = 1$. The

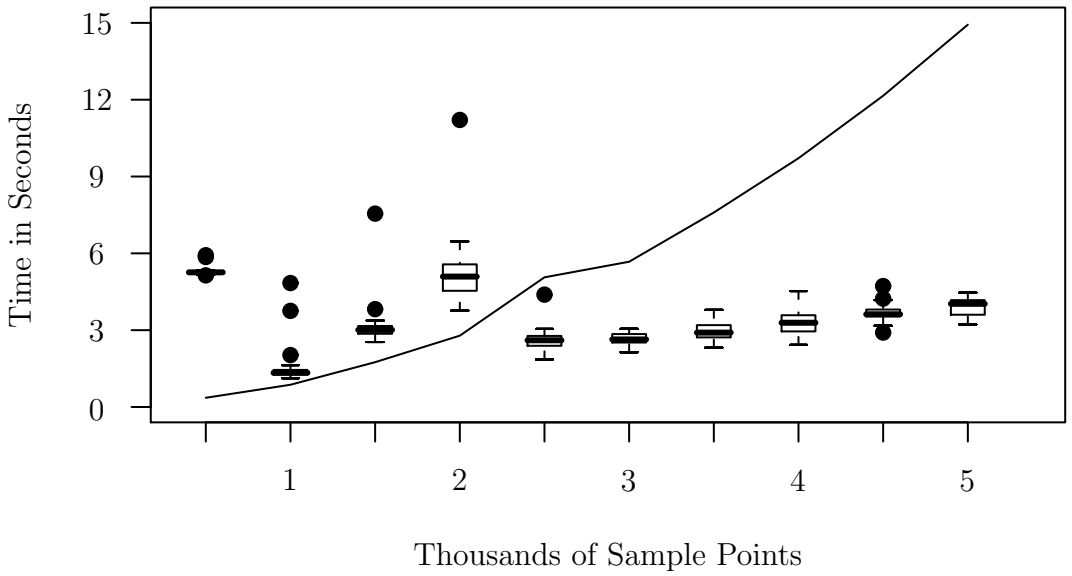


Figure 4.10: A comparison of the time required to compute the test for separation using the simplex method and using an interior point method versus the number of sample points. The line shows the empirical worst-case time based on 25 simulations for the simplex method and the box plots show the results of 25 simulations using the interior point method.

time taken to solve the linear program with

```
options = optimset('LargeScale', 'off', 'Simplex', 'on')
```

for the simplex method and with

```
options = optimset('LargeScale', 'on')
```

for the large scale algorithm was measured using the `tic` and `toc` functions. Also, note that Matlab® implements the simplex method and not the revised simplex method used previously.

To get an idea of the relative performance of the interior point method versus the simplex method, twenty-five simulations were carried out for each value of n ranging from 500 to 5000 in lots of 500; 16 column design matrices were used. Figure 4.10 shows the results of this simulation. There was considerably more variability in

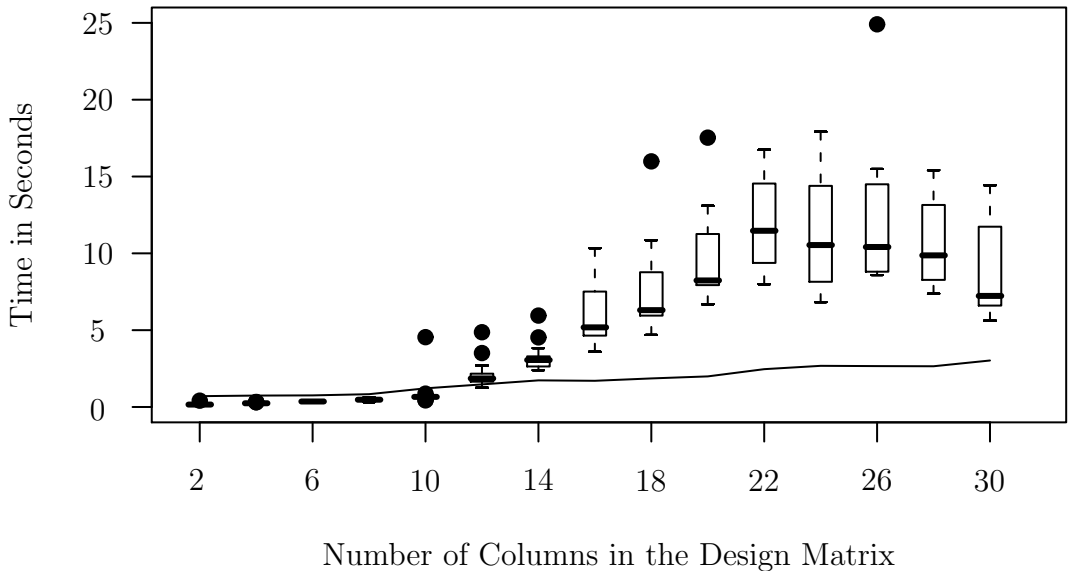


Figure 4.11: A comparison of the time required to compute the test for separation using the simplex method and using an interior point method versus the number of columns in the design matrix. The curve shows the empirical worst-case time based on 25 simulations for the simplex method and the box plots show the results of 25 simulations using the interior point method.

the time taken by the interior point method so the results of the 25 simulations for each value of n are displayed using box plots. The curve in figure 4.10 shows the worst-case computing time based on 25 simulations. We see that for smaller models, those with less than 2500 sample points, the simplex method is faster. However, as the number of sample points increases beyond this value, the interior point method then becomes the faster method.

To assess the relative performance of the simplex method versus the interior point method as the number of columns in the design matrix is increased, twenty-five simulations were carried out for each value of p ranging from 2 to 30 in lots of 2; 1000 sample points were used. Figure 4.11 shows the results of this simulation. The curve shows the empirical worst-case run time for the simplex method based on 25 simulations. The results of the interior point method are again displayed using

box plots. Also, as the number of columns in the design matrix was increased, the interior point method failed to converge for some simulations. Only the simulations where the method converged are shown in the plot. When the interior point method did not converge the time taken to reach the maximum number of iterations was on the order of 100 seconds.

4.6.4 Discussion

Our simulations show that, among the tests for separation based on the primal, dual and reduced linear programs, the test based on the dual linear program had the best empirical worst-case performance. Also, as the number of columns in the design matrix increased, the performance of the test based on the reduced linear program improved to approximately that of the test based on the dual. However, from time to time, the rounding error accumulated while computing the reduced form was sufficient to alter the boundedness of the feasible region leading to an incorrect result. In the simulations, the frequency of these incorrect results seemed to depend mostly on the number of decision variables; ranging from never for the simulations involving 16 column design matrices to 6 in 25 for the simulations using 2000×256 design matrices. Hence the test based on the reduced linear program is not reliable for routine use.

Since the test based on the dual linear program takes approximately the same amount of time as fitting the model using iteratively reweighted least squares, we feel that it is fast enough to be used routinely as part of the fitting process for the binary logistic regression model.

Interestingly, the primal linear program with bounded decision variables used to

compute a vector revealing separation had approximately the same empirical worst-case run time as the test for separation based on the dual linear program. This method returns the zero vector when there is overlap among the sample points and a nonzero vector when separation is present. Hence this method also provides a test for separation among the sample points. In the event that separation is present, the nonzero elements of the vector revealing separation can be used to identify the terms in the model causing separation. For an example see appendix A.

Chapter 5

Conclusion

This thesis studied the detection of separation among the sample points in binary logistic regression models using methods based on linear programming.

We proposed a linear program with a nonnegative objective function that has a positive optimal value when separation is present among the sample points and compared several methods for its solution. Our main result was demonstrating that a test for separation based on determining the feasibility of the dual to this linear program is both numerically reliable and can be computed fast enough to be used routinely as part of the fitting procedure for binary logistic regression models. The performance of the test was evaluated using simulation methods. The test is implemented in the accompanying R (R Development Core Team, 2007) package `safeBinaryRegression` so that, when the package is loaded, fitting binary logistic regression models in the standard way includes a test for separation (see appendix A).

We also found through direct experimentation that linear programming methods which use a preliminary transformation of the linear program to eliminate the free

decision variables are sometimes unreliable. Rounding error accumulated during this preliminary transformation is from time to time sufficient to alter the feasible region so that the optimal value of the transformed linear program no longer corresponds to the optimal value of the original linear program. Among our methods, we observed this behavior in the reduced linear program. The linear programs described in Silvapulle and Burridge (1986) and Clarkson and Jennrich (1991) are also affected. Our test based on the dual linear program does not require a preliminary transformation; hence it can be computed with a higher degree of reliability.

To compute our test for separation among the sample points we implemented an interface to the lp_solve library (Berkelaar et al., 2007). This interface provides a general, publicly available linear programming solver available for R. In addition to our test for separation, the lp_solve interface was also used to implement functions for

- verifying the algebraic condition for separation given in Albert and Anderson (1984),
- solving the mixed integer linear program described in Santner and Duffy (1986),
- finding linear separation based on the methods given in Mangasarian (1965) and Smith (1968),
- computing the test for separation described in Silvapulle and Burridge (1986), and
- verifying the condition given in Jacobsen (1989) for the existence of the maximum likelihood estimate of the binary logistic regression model parameter vector β.

These functions are included as well in the `safeBinaryRegression` package.

Appendix A

Implementation in R

To encourage routine use we have implemented our test for separation in the `safeBinaryRegression` package for R (R Development Core Team, 2007). This appendix is meant to be a brief overview of the `safeBinaryRegression` package. Complete documentation is provided in the package and is accessible using R's built-in help facility. A source version of the package can be downloaded from the url

`http://www.stats.ox.ac.uk/~konis/safeBinaryRegression.tgz`

and the package should be available on CRAN shortly.

The standard way to fit binary logistic regression models in R is to use the `glm` function (see for example, Venables and Ripley, 1999, chap. 7). For example, the following command fits a binary logistic regression model to a set of 50 quasiseparated sample points in the data frame `qs.dat`.

```
glm(y ~ x, family = binomial(), data = qs.dat)
```

This function produces the following output.

```
Call:  glm(formula = y ~ x, family = binomial())


Coefficients:
(Intercept)           xb           xc           xd
     0.1823      -0.5188       0.1054      15.3837

Degrees of Freedom: 49 Total (i.e. Null);  46 Residual
Null Deviance:      69.31
Residual Deviance: 66.88        AIC: 74.88
```

Note that there is no error or warning indicating the presence of separation.

Our package overrides R's `glm` function so that, from the user's point of view, no change to workflow is required. When the `safeBinaryRegression` package is loaded, the `glm` function performs a test for separation among the sample points when it is used to fit a binary logistic regression model. Otherwise the behavior of the `glm` function is unchanged.

```
library(safeBinaryRegression}
glm(y ~ x, family = binomial(), data = qs.dat)
Error in glm(y ~ x, family = binomial()) :
        Separation exists among the sample points.
        This model cannot be fit by maximum likelihood.
```

The version of the `glm` function provided by the `safeBinaryRegression` package contains one additional argument: `separation`. The default behavior is to test for separation. Alternatively, using `separation = "find"` lists the terms in the model causing the separation.

```
glm(y ~ x, family = binomial(), data = qs.dat,
      separation = "find")
Error in glm(y ~ x, family = binomial(), separation = "find") :
        The following terms are causing separation among the
        sample points: xd
```

If there is a sensible way to combine the level `x == "d"` (the level of the factor

variable `x` corresponding to term `xd` in the model) with one of the other levels of `x` then the resulting model could be fit using maximum likelihood.

Bibliography

Albert, A. and Anderson, J. A. (1984) On the existence of maximum likelihood estimates in logistic regression models. *Biometrika*, **71**, 1–10.

Albert, A. and Lesaffre, E. (1986) Multiple group logistic discrimination. *Computers & Mathematics with Applications-Part A*, **12**, 209–224.

Anderson, J. A. (1972) Separate sample logistic discrimination. *Biometrika*, **59**, 19–35.

Barber, C. B., Dobkin, D. P. and Huhdanpaa, H. (1996) The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, **22**, 469–483.

Barndorff-Nielsen, O. (1978) *Information and Exponential Families in Statistical Theory*. Chichester: John Wiley & Sons.

Berkelaar, M., Eikland, K. and Notebaert, P. (2007) *lp_solve*. URL `http://tech.groups.yahoo.com/group/lp_solve/`.

Bishop, Y. M. M., Fienberg, S. E. and Holland, P. W. (1975) *Discrete Multivariate Analysis: Theory and Practice*. Cambridge, Massachusetts and London, England: The MIT Press.

Björck, Å. (1996) *Numerical Methods for Least Squares Problems*. Philadelphia: SIAM.

Bland, R. G. (1977) New finite pivoting rules for the simplex method. *Mathematics of Operations Research*, **2**, 103–107.

Burridge, J. (1981) *Statistical Analysis of Grouped Lifetime Data in a Changing Environment*. Ph.D. thesis, Imperial College, London University.

Christmann, A. and Rousseeuw, P. J. (2001) Measuring overlap in binary regression. *Computational Statistics & Data Analysis*, **37**, 65–75.

Chvátal, V. (1983) *Linear Programming*. New York: W. H. Freeman and Company.

Clarkson, D. B. and Jennrich, R. I. (1991) Computing extended maximum likelihood estimates for linear parameter models. *Journal of the Royal Statistical Society, Series B*, **53**, 417–426.

Cox, D. R. (1970) *The Analysis of Binary Data*. 11 New Fetter Lane, London: Methuen & Co. Ltd.

Cox, D. R. and Snell, E. J. (1989) *Analysis of Binary Data*. London: Chapman and Hall, second edn.

Dantzig, G. (1963) *Linear Programming and Extensions*. Princeton, N.J.: Princeton University Press.

Dantzig, G. B. and Thapa, M. N. (1997) *Linear Programming 1: Introduction*. New York: Springer.

Firth, D. (1993) Bias reduction of maximum likelihood estimates. *Biometrika*, **80**, 27–38.

Gill, P. E., Murray, W. and Wright, M. H. (1981) *Practical Optimization*. London: Academic Press.

Haberman, S. J. (1974) *The Analysis of Frequency Data*. Chicago, Illinois: University of Chicago Press.

Harris, P. M. J. (1973) Pivot selection methods of the Devex LP code. *Mathematical Programming*, **5**, 1–28.

Hauck, W. W. and Donner, A. (1977) Wald's test as applied to hypotheses in logit analysis. *Journal of the American Statistical Association*, **72**, 851–853.

Heinze, G. and Schemper, M. (2002) A solution to the problem of separation in logistic regression. *Statistics in Medicine*, **21**, 2409–2419.

Higham, N. J. (1995) *Accuracy and Stability of Numerical Algorithms*. Philadelphia: SIAM.

IEC 60559:1989 (1989) *Binary floating-point arithmetic for microprocessor systems*. International Electrotechnical Commission, 3, Rue de Varembé, Geneva, Switzerland. URL `http://www.iec.ch`.

Insightful (2005) *S-PLUS®*. URL `http://www.insightful.com`.

Jacobsen, M. (1989) Existence and unicity of MLEs in discrete exponential family distributions. *Scandinavian Journal of Statistics*, **16**, 335–349.

Kantorovich, L. (1939) *Mathematical Methods of Organizing and Planning Production*. Leningrad, Russia: Leningrad State University Press.

Klee, V. L. and Minty, G. J. (1972) How good is the simplex algorithm. In *Inequalities III* (ed. O. Shisha), 159–175. London and New York: Academic Press.

Koford, J. S. (1964) Adaptive pattern dichotomization. *Tech. Rep. 6201-1*, Stanford Electronics Labs, Stanford, CA.

Koford, J. S. and Groner, G. F. (1966) The use of an adaptive threshold element to design a linear optimal pattern classifier. *IEEE Transactions on Information Theory*, **12**, 42–50.

Lesaffre, E. and Albert, A. (1989) Partial separation in logistic discrimination. *Journal of the Royal Statistical Society, Series B*, **51**, 109–116.

Luenberger, D. G. (1984) *Linear and Nonlinear Programming*. Reading, Massachusetts: Addison-Wesley.

Mangasarian, O. L. (1965) Linear and nonlinear separation of patterns by linear programming. *Operations Research*, **13**, 444–452.

Maros, I. (2001) A general pricing scheme for the simplex method. *Tech. Rep. ISSN 1469-4174*, Imperial College, London.

Mays, C. M. (1964) Effects of adaptation parameters on convergence time and tolerance for adaptive threshold elements. *IEEE Transactions on Electronic Computers*, **EC-13**, 465–468.

McCullagh, P. and Nelder, J. A. (1989) *Generalized Linear Models*. Boca Raton, FL: Chapman & Hall, second edn.

Mehrotra, S. (1992) On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, **2**, 575–601.

Mehta, C. R. and Patel, N. R. (1995) Exact logistic regression: Theory and examples. *Statistics in Medicine*, **14**, 2143–2160.

Minnick, R. C. (1961) Linear-input logic. *IRE Transactions on Electronic Computers*, **10**, 6–16.

Nelder, J. A. and Wedderburn, R. W. M. (1972) Generalized linear models. *Journal of the Royal Statistical Society, Series A*, **135**, 370–384.

Nilsson, N. J. (1965) *Learning Machines: Foundations of Trainable Pattern-Classifying Systems*. New York: McGraw-Hill.

Orden, A. (1993) LP from the '40s to the '90s. *INTERFACES*, **23**, 2–12.

Papadimitriou, C. H. and Steiglitz, K. (1982) *Combinatorial Optimization Algorithms and Complexity*. Englewood Cliffs, New Jersey: Prentice-Hall, Inc.

R Development Core Team (2007) *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL `http://www.R-project.org`. ISBN 3-900051-07-0.

Ripley, B. D. (1996) *Pattern Recognition and Neural Networks*. Cambridge, UK: Cambridge University Press.

Rosenblatt, F. (1957) The perceptron—a perceiving and recognizing automaton. *Tech. Rep. 85-460-1*, Cornell Aeronautical Laboratory.

— (1958) The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, **65**, 386–408.

Santner, T. J. and Duffy, D. E. (1986) A note on A. Albert and J. A. Anderson's conditions for the existence of maximum likelihood estimates in logistic regression models. *Biometrika*, **73**, 755–758.

SAS Institute Inc. (2003) *SAS® v9.1*. Cary, NC, USA. URL `http://www.sas.com`.

— (2004) *SAS STAT® 9.1 User's Guide*. Cary, NC, USA. URL `http://support.sas.com/documentation/onlinedoc/91pdf/sasdoc_91/stat_ug_7313.pdf`.

Schölkopf, B. and Smola, A. J. (2002) *Learning with Kernels*. Cambridge, Massachusetts and London, England: The MIT Press.

Silvapulle, M. J. (1981) On the existence of maximum likelihood estimators for the binomial response models. *Journal of the Royal Statistical Society, Series B*, **43**, 310–313.

Silvapulle, M. J. and Burridge, J. (1986) Existence of maximum likelihood estimates in regression models for grouped and ungrouped data. *Journal of the Royal Statistical Society, Series B*, **48**, 100–106.

Smith, F. W. (1968) Pattern classifier design by linear programming. *IEEE Transactions on Computers*, **C-17**, 367–372.

— (1969) Design of multicategory pattern classifiers with two-category classifier design procedures. *IEEE Transactions on Computers*, **C-18**, 548–551.

Süli, E. and Mayers, D. (2003) *An Introduction to Numerical Analysis*. Cambridge, UK: Cambridge University Press.

The MathWorks (2006) *MATLAB® v7.2*. Natick, MA.

Venables, W. N. and Ripley, B. D. (1999) *Modern Applied Statistics with S-PLUS*. New York: Springer, third edn.

Wedderburn, R. W. M. (1976) On the existence and uniqueness of the maximum likelihood estimates for certain generalized linear models. *Biometrika*, **63**, 27–32.

Zoutendijk, G. (1960) *Methods of feasible directions: a study in linear and nonlinear programming*. Amsterdam; London: Elsevier.