ANZIAM J. 45(2003), 91–114

NUMERICAL ALGORITHMS FOR CONSTRAINED MAXIMUM LIKELIHOOD ESTIMATION

Z. F. LI[1], M. R. OSBORNE[2] and T. PRVAN[3]

(Received 27 February, 2001; revised 10 July, 2002)

Abstract

This paper describes a SQP-type algorithm for solving a constrained maximum likelihood estimation problem that incorporates a number of novel features. We call it MLESOL. MLESOL maintains the use of an estimate of the Fisher information matrix to the Hessian of the negative log-likelihood but also encompasses a secant approximation S to the second-order part of the augmented Lagrangian function along with tests for when to use this information. The local quadratic model used has a form something like that of Tapia's SQP augmented scale BFGS secant method but explores the additional structure of the objective function. The step choice algorithm is based on minimising a local quadratic model subject to the linearised constraints and an elliptical trust region centred at the current approximate minimiser. This is accomplished using the Byrd and Omojokun trust region approach, together with a special module for assessing the quality of the step thus computed. The numerical performance of MLESOL is studied by means of an example involving the estimation of a mixture density.

1. Introduction

There are good reasons for numerical analysts to study maximum likelihood estimation problems. In the first place, they are a computation of primary importance in statistical data analysis and hence in the social sciences, as well as in the more traditional areas within the physical sciences. Many special purpose algorithms is available in the unconstrained case, but only very few methods have been developed for the nonlinear constrained case. Osborne [15] shows that a simple application of a Powell-Hestenes

[1] National Centre for Epidemiology and Population Health, Australian National University, Canberra, ACT 0200, Australia; e-mail: zhengfeng.li@anu.edu.au.

[2] Centre for Mathematics and its Applications, School of Mathematical Sciences, Australian National University, Canberra, ACT 0200, Australia; e-mail: Mike.Osborne@maths.anu.edu.au.

[3] School of Mathematics and Statistics, The University of Canberra, Canberra, ACT 2617, Australia; e-mail: TaniaP@ise.canberra.edu.au.

multiplier method, including the use of an estimate of the Fisher information matrix while ignoring the second-order part of the constraints, retains typical features of special purpose methods for unconstrained problems. His method has been proven to be effective. However, there appear to us to be two important ways in which Osborne's method might still be improved. First, good performance of his method relies heavily on the availability of a large number of fitting data so that the theory of large samples can be applied. Second, the proposed fitting model is supposed to be correct. This paper primarily attempts to improve upon Osborne's method in these two regards and to be more efficient than general SQP secant methods, such as Schittkowski's NLPQL program, which are intended for general function minimisation. The proposed algorithm maintains the use of an estimate of the Fisher information matrix to the Hessian of the negative log-likelihood but also encompasses a secant approximation S to the second-order part of the augmented Lagrangian function along with tests for when to use this information. It also uses the ideas of Tapia's SQP augmented scale BFGS secant method for dealing with the local quadratic model. As far as we know, no global implementation of this kind of algorithm has yet been developed. In this paper we include a step size control strategy to guarantee a global convergence property. The step choice algorithm is based on minimising a local quadratic model subject to the linearised constraints and an elliptical trust region centred at the current approximate minimiser, together with a special module for assessing the quality of the step thus computed. Our primary purpose here is to report the details and to give some test results.

In Section 2 we set out the problem and review some algorithms for an easy reference. Section 3 briefly describes our new algorithm MLESOL. The globalisation version of the algorithm is described in Section 4, including sizing strategy for the augmentation, model switching strategy, step size control strategy and the computation of an approximate solution to the constrained quadratic subproblem. Section 5 concludes our discussion with test results.

2. Constrained maximum likelihood estimation

Computing a parameter estimate often reduces to minimising an objective function with some constraints which incorporate additional information on the parameter model. To be more specific, it will be useful to have a formal statement of the problem: Let $\hat{y}_1, \dots, \hat{y}_n$ be independent observations, each $\hat{y}_i$ is from a distribution with a density $p(y \mid x)$, where the parameter $x \in \Re^p$ is subject to the constraints $c(x) = 0$, here $c(x) = (c_1(x), c_2(x), \dots, c_m(x))^T$. We assume each $c_i(x)$ is an arbitrary twice differentiable function.

There are a variety of estimation methods which one might bring to bear on this problem, see Sen *et al.* [18]. We focus on the method of maximum likelihood. In this, one determines a maximum likelihood estimate $\hat{x}_n$ which maximises

$$\max_{x \in \Re^p} \prod_{i=1}^{n} p(\hat{y}_i \mid x) \qquad \text{s.t. } c(x) = 0. \tag{2.1}$$

For simplicity of notation, we set $f_i(x) = p(\hat{y}_i \mid x)$. Then problem equivalent to (2.1) is

$$\min_{x \in \Re^p} K(x) = \frac{1}{n} \sum_{i=1}^{n} -\log(f_i(x)) \qquad \text{s.t. } c(x) = 0. \tag{2.2}$$

This is a nonlinear optimisation problem that must be solved by some iterative technique. As is usual in many other minimisation problems we have to study the structure of the derivatives of the objective function $K(x)$ and the constraints $c(x)$.

Define $F(x) = (f_1(x), \ldots, f_n(x))^T$, $T(x) = (1/f_1(x), \ldots, 1/f_n(x))^T$, $D(x) = \operatorname{diag}(1/f_i(x))$ and let u be the n-dimensional vector whose components are unity, $J(x)$ be the Jacobian matrix of $F(x)$ at x and $M(x)$ be defined by

$$M(x) = \frac{1}{\sqrt{n}} D(x) J(x).$$

Then we obtain the following expressions for $\nabla K(x)$ and $\nabla^2 K(x)$:

$$\nabla K(x) = -\frac{1}{n} J(x)^T T(x) = -M(x)^T \left(\frac{u}{\sqrt{n}} \right)$$

and

$$\nabla^2 K(x) = M(x)^T M(x) - \frac{1}{n} \sum_{i=1}^{n} \frac{1}{f_i(x)} \nabla^2 f_i(x). \tag{2.3}$$

Most of the statistical literature only considers unconstrained problems, that is, problems where the constraints in (2.2) are removed, see, for example, Bunch [1], Osborne [14], Walker *et al.* [20], Gonglewski [8] and Gay *et al.* [7]. In this simple case, two standard methods are Newton's method and the method of scoring which are defined by the following iterations:

Newton's method:

$$x_{k+1} = x_k - [\nabla^2 K(x_k)]^{-1} \nabla K(x_k); \tag{2.4}$$

Method of scoring:

$$x_{k+1} = x_k - [I(x_k)]^{-1} \nabla K(x_k), \tag{2.5}$$

where $I(x)$ denotes the expectation $E[\nabla^2 K(x)] = E[M^T(x) M(x)]$.

The well-known advantage of (2.4) is that it converges q-quadratically to the solution if a good starting point is provided. However the methods defined by (2.4) and (2.5) are not guaranteed to be globally convergent. Some kind of step size control is often needed to expand the region of convergence. In optimisation algorithms, one often exercises step size control by doing an approximate line search. Due to the positive definiteness of $I(x)$, the method of scoring yields a natural globalisation via line search.

There are other disadvantages to (2.4) and (2.5). Newton's method requires calculation of the Hessian at each iteration, which can be quite expensive for complex models. The method of scoring often does not have a fast local convergence property.

Despite the difficulties outlined above, many efficient special purpose computer programs are available to solve an unconstrained maximum likelihood estimation, for example, the structured secant DFP method by Walker *et al.* [20] and the structured secant BFGS method by Martinéz *et al.* [11]. However, the situation changes if we want to solve a constrained maximum likelihood estimation. A combination of the Powell-Hestenes multiplier method with the method of scoring discussed above was proposed by Osborne [15] for solving constrained parameter estimation problems. The Powell-Hestenes multiplier method involves a sequence of minimisations of the augmented Lagrangian function with multiplier estimates being updated after each minimisation. The method of scoring is applied to minimise the augmented Lagrangian function in each of the sequence of steps.

The algorithm is simple to explain and motivate. At a current iterate x_k we choose two parameters $\theta_k \in R^m$ and $\rho_k \geq 0$, and attempt to generate a new iterate x_{k+1} by minimising

$$L^P(x, \theta_k, \rho_k) = K(x) + \frac{\rho_k}{2}[c(x) + \theta_k]^T[c(x) + \theta_k].$$

The parameter θ_k corresponds to shifts of origin and the scalar ρ_k controls the size of the penalty. In the following, for simplicity of notation we will drop the subscripts k and replace the subscripts $k+1$ by $+$. Sometimes we need to use the superscript (i) to denote the i-th component of a vector.

In fact it is more convenient to define $\lambda^{(i)} = \rho\theta^{(i)}$, $i = 1, 2, \ldots, m$, and ignore the term $\frac{\rho}{2}\sum_{i=1}^{m}(\theta^{(i)})^2$ (independent of x), giving the augmented Lagrangian function

$$L(x, \lambda, \rho) = K(x) + \lambda^T c(x) + \frac{\rho}{2}c(x)^T c(x).$$

Hence, given a multiplier estimate $\lambda \in R^m$, the next iterate x_+ can also be obtained by solving

$$\min_{x \in R^p} L(x, \lambda, \rho). \tag{2.6}$$

We now discuss how to solve the subproblem (2.6) in each Powell-Hestenes iteration. Notice that if we define $A(x)$ to be the Jacobian matrix of $c(x)$ by $A(x) = \big(\partial c_i(x)/\partial x_j\big)$ and using (2.3), we have the expressions for $\nabla_x L(x, \lambda, \rho)$ and $\nabla_x^2 L(x, \lambda, \rho)$

$$\nabla_x L(x, \lambda, \rho) = \nabla K(x) + A(x)^T[\lambda + \rho c(x)]$$

and

$$\begin{aligned} \nabla_x^2 L(x, \lambda, \rho) &= \nabla^2 K(x) + \rho A(x)^T A(x) + \sum_{i=1}^{m}[\lambda^{(i)} + \rho c_i(x)]\nabla^2 c_i(x) \\ &= C(x) + S(x, \lambda, \rho), \end{aligned} \tag{2.7}$$

where

$$C(x) = M(x)^T M(x) + \rho A(x)^T A(x) \tag{2.8}$$

and

$$S(x, \lambda, \rho) = -\frac{1}{n}\sum_{i=1}^{n}\frac{1}{f_i(x)}\nabla^2 f_i(x) + \sum_{i=1}^{m}[\lambda^{(i)} + \rho c_i(x)]\nabla^2 c_i(x).$$

Therefore, in addition to making a precise convergence test possible, having accurate Jacobian matrices $J(x)$ and $A(x)$ means that a good approximation to a portion of the Hessian is available as a by-product of the gradient computation. Note that the second-order term $S(x, \lambda, \rho)$ requires the calculation of $n + m$ (expensive) Hessian matrices. In fact there are some problems where it is often possible to ignore the second-order term $S(x, \lambda, \rho)$ of the Hessian.

Let $\hat{\lambda}_n$ be the corresponding multiplier of (2.2) at the solution $\hat{x}_n$. Notice that, at the solution $\hat{x}_n$, $\rho \sum_{i=1}^{m} c_i(\hat{x}_n)\nabla^2 c_i(\hat{x}_n) = 0$. In addition, when n is large and the fitting model is correct, one can show that (see Osborne [15]) the following two probability quantities hold: $P\big(\lim_{n\to\infty} \hat{\lambda}_n = 0\big) = 1$ and

$$P\left(\lim_{n\to\infty}\sum_{i=1}^{n}\frac{1}{f_i(\hat{x}_n)}\nabla^2 f_i(\hat{x}_n) = 0\right) = 1.$$

Hence, when x is close to the solution $\hat{x}_n$ and n is large enough, the second-order term $S(x, \lambda, \rho)$ of the Hessian can be neglected under appropriate conditions. Based on these arguments, Osborne [15] suggests minimising (2.6) by the iterative procedure

$$x_0 = x, \quad x_{j+1} = x_j + \alpha_j d_j, \quad j = 0, 1, 2, \dots,$$

where d_j solves

$$C(x_j)d_j = -\nabla_x L(x_j, \lambda, \rho) \tag{2.9}$$

and α_j is the step size by doing an approximate line search. He sets the last iterate x_{j_k} to x_+.

It is worth mentioning that d_j produced by (2.9) is always a descent direction

$$d_j^T \nabla_x L(x_j, \lambda, \rho) = -(\nabla_x L(x_j, \lambda, \rho))^T C(x)^{-1} \nabla_x L(x_j, \lambda, \rho) < 0$$

from the positive definiteness of $C(x)$. Hence this method yields a natural globalisation via line search.

Since (2.9) is the system of normal equations of the linear least squares problem

$$\min_{d \in R^p} \left\| \begin{bmatrix} \sqrt{\rho} A(x_j) \\ M(x_j) \end{bmatrix} d + \begin{bmatrix} \lambda / \sqrt{\rho} + \sqrt{\rho} c(x_j) \\ -u / \sqrt{n} \end{bmatrix} \right\|_2^2, \tag{2.10}$$

it is better to obtain d_j from a QR decomposition of the matrices $M(x_j)$ and $A(x_j)$. However the system (2.10) is often ill-conditioned when ρ is large. This is the typical case (the recommended value for ρ in Osborne [15] is $\sqrt{n}$).

However, there are some cases in which the second-order term $S(x, \lambda, \rho)$ can not be neglected. An alternative is to approximate the Hessian via a secant update, for example, the BFGS formula. Examination of (2.7) reveals that such an approach is tantamount to discarding the useful first-order term $C(x)$ (2.8) at each step. This may affect efficiency. Another approach is to augment the matrix $C(x)$ by adding an approximation to the second-order term $S(x, \lambda, \rho)$. This kind of structured secant methods was also used by Dennis *et al.* [3] for nonlinear unconstrained least squares problems. Since we also need the notation of structured secant methods in order to explain our ideas in the next section, we consider the following principle for using additive structure; also see Engels *et al.* [6].

Assume that, for a function $h(x)$,

$$\nabla^2 h(x) = H_1(x) + H_2(x), \tag{2.11}$$

where $H_1(x)$ is inexpensive to compute and symmetric. Given $B = H_1(x) + W$ as an approximation to $\nabla^2 h(x)$, where W is a secant approximate to $H_2(x)$, let $x_+ = x + s$. The question is how to update B to give B_+ such that B_+ is a good approximant to $\nabla^2 h(x_+)$. In doing this, we first update W by a secant approximant W_+ and ask it to satisfy the secant equation $W_+ s = y^p$, where W_+ is intended to approximate $H_2(x_+)$ and y^p is an approximation of $H_2(x_+)s$. Then we choose

$$B_+ = H_1(x_+) + W_+ \tag{2.12}$$

as an approximation of $\nabla^2 h(x_+)$. In order to ensure that B_+ has a property of least-change secant update, we update W as follows: let

$$y = H_1(x_+)s + y^p \tag{2.13}$$

be an approximation of $\nabla^2 h(x_+)s$ and

$$B^s = H_1(x_+) + W \tag{2.14}$$

be an approximation of $\nabla^2 h(x_+)$. Then we define

$$W_+ = W + U(s, y^p, W, v(s, y, B^s)), \tag{2.15}$$

where U is a secant update defined for $v^T s \neq 0$:

$$U(s, y, B, v) = \frac{(y - Bs)v^T + v(y - Bs)^T}{v^T s} - \frac{(y - Bs)^T s v v^T}{(v^T s)^2}. \tag{2.16}$$

The scale v usually depends on s, y and B. We will have occasion to use both the notation v and $v(s, y, B)$; the former will be used unless we feel it is important to emphasise the role of the parameter. We prefer to use the BFGS update which corresponds to the following choices:

$$v^{\text{BFGS}} = v^{\text{BFGS}}(s, y, B) = y + \sigma Bs, \quad \sigma = \sqrt{\frac{y^T s}{s^T Bs}}. \tag{2.17}$$

With the above procedure, one can show that B_+ defined by (2.12) has the following update formula: $B_+ = B^s + U(s, y, B^s, v(s, y, B^s))$.

One could apply the structure principle (2.11)–(2.17) to the Hessian of the augmented Lagrangian function (2.7) with $H_1(x) = C(x)$, $H_2(x) = S(x, \lambda, \rho)$ and augment $C(x)$ in (2.9) by adding an approximation to the second-order term $S(x, \lambda, \rho)$. In this paper we refer to this as the augmented version of Osborne's method.

Exploiting this structure within the framework of the Powell-Hestenes multiplier method may be worthwhile. However, compared with the performance of the proposed SQP-type algorithm in the next section, the improvement from this approach looks less attractive as explained below.

3. New algorithms based on SQP-type methods

Based on its good numerical performance, we prefer to use a SQP-type method to solve problem (2.2). Compared with SQP-type methods, the Powell-Hestenes multiplier method is less attractive, because at each step it may take several inner iterations to do unconstrained minimisation in order to find x_+ and hence needs more work. Another reason lies in its slow convergence rate. Before we present our new algorithm, we would like to introduce some known SQP methods.

Define the Lagrangian function associated with (2.2) by

$$l(x, \lambda) = K(x) + \lambda^T c(x). \tag{3.1}$$

At the current iterate x, the SQP Lagrangian secant method is characterised by the iterative procedure

$$\begin{aligned} x_+ &= x + s, & y^l &= \nabla_x l(x_+, \lambda_+) - \nabla_x l(x, \lambda_+), \\ \lambda_+ &= \lambda + \delta\lambda, & B^l_+ &= B^l + U\left(s, y^l, B^l, v^{\text{BFGS}}\right), \end{aligned} \tag{3.2}$$

where $U(s, y, B, v)$ is defined as in (2.16), s and $\delta\lambda$ are respectively the solution and the multiplier associated with the solution of the quadratic program

$$\min_{s \in R^p} \nabla_x l(x, \lambda)^T s + \frac{1}{2} s^T B^l s \quad \text{s.t. } c(x) + A(x)s = 0,$$

where B^l is intended to be an approximation to $\nabla_x^2 l(x, \lambda)$.

The deficiency of the SQP Lagrangian method is that the local convergence theory requires $\nabla_x^2 l(\hat{x}_n, \hat{\lambda}_n)$ to be positive definite and yet satisfaction of this condition is not guaranteed by standard assumptions, under which the update B may lose the positive definiteness property even if the starting matrix B_0 is the identity matrix. To circumvent the lack of positive definiteness of $\nabla_x^2 l(\hat{x}_n, \hat{\lambda}_n)$, one could replace the Lagrangian function (3.1) with the augmented Lagrangian function. Recall that the augmented Lagrangian function associated with (2.2) is

$$L(x, \lambda, \rho) = l(x, \lambda) + \frac{\rho}{2} c(x)^T c(x).$$

Observe that the Hessian of the augmented Lagrangian method is

$$\nabla_x^2 L(x, \lambda, \rho) = \nabla_x^2 l(x, \lambda) + \rho A(x)^T A(x) + \rho \sum_{i=1}^{m} c_i(x) \nabla^2 c_i(x). \tag{3.3}$$

At a solution $(\hat{x}_n, \hat{\lambda}_n)$, we have

$$\nabla_x^2 L(\hat{x}_n, \hat{\lambda}_n, \rho) = \nabla_x^2 l(\hat{x}_n, \hat{\lambda}_n) + \rho A(\hat{x}_n)^T A(\hat{x}_n) \tag{3.4}$$

from the fact that $c_i(\hat{x}_n) = 0$ for $i = 1, \dots, m$.

One can see from (3.4) that for any augmentation parameter ρ greater than a threshold value $\hat{\rho}$, $\nabla_x^2 L(x, \lambda, \rho)$ is positive definite at the solution $(\hat{x}_n, \hat{\lambda}_n)$. Using y as a generic term and denoting different choices of y by different superscripts, if y is defined to be $y^L = \nabla_x L(x_+, \lambda_+, \rho) - \nabla_x L(x, \lambda_+, \rho)$, we can guarantee that near the solution $s^T(y^L) > 0$ for ρ sufficiently large such that the BFGS secant update B shares the hereditary positive definiteness property.

We arrive at the SQP augmented Lagrangian secant method, given by the iterative procedure

$$\begin{aligned} x_+ &= x + s, & y^L &= \nabla_x L(x_+, \lambda_+, \rho) - \nabla_x L(x, \lambda_+, \rho), \\ \lambda_+ &= \lambda + \delta\lambda, & B^L_+ &= B^L + U\left(s, y^L, B^L, v^{\text{BFGS}}\right), \end{aligned}$$

where $U(s, y, B, v)$ is defined as in (2.16), s and $\delta\lambda$ are respectively the solution and the multiplier associated with the solution of the quadratic program

$$\min_{s \in R^p} \nabla_x L(x, \lambda, \rho)^T s + \frac{1}{2} s^T B^L s \quad \text{s.t.} \;\; c(x) + A(x)s = 0. \tag{3.5}$$

In (3.5), B^L is intended to be an approximation to $\nabla_x^2 L(x, \lambda, \rho)$.

Though theoretically attractive, this alternative has serious practical problems. First, $\hat{\rho}$ is not known *a priori*. Secondly, the attempt to use large ρ seems to present severe numerical problems; see Tapia [19] and Nocedal *et al.* [12].

It is worth noting that at the solution the Hessian of the augmented Lagrangian function (3.3) displays significant structure in that there is a clear separation between the first- and second-order terms. The term $\sum_{i=1}^m c_i(x)\nabla^2 c_i(x)$ is responsible for the bad fit in the SQP augmented Lagrangian secant methods because it is incompatible with the satisfaction of linearised constraints, see Tapia [19]. Therefore we use $\nabla_x^2 l(x, \lambda) + \rho A(x)^T A(x)$ to approximate the Hessian of the augmented Lagrangian $\nabla_x^2 L(x, \lambda, \rho)$ by ignoring the term $\sum_{i=1}^m c_i(x)\nabla^2 c_i(x)$. If we relate this observation to the structure principle (2.11)–(2.17) with $H_1(x) = \rho A(x)^T A(x)$, $H_2(x) = \nabla_x^2 l(x, \lambda)$ and use y^l defined as in (3.2) to approximate $\nabla_x^2 l(x, \lambda)s$, then we get the following baseline algorithm (we replace B and W by B^L and B^l, respectively) due to Tapia [19].

ALGORITHM 3.1 (Tapia [19]).

Step 1. Given x, λ, B^l and B^L $(= B^l + \rho A(x)^T A(x))$.

Step 2. Compute the solution $(s, \delta\lambda)$ of the QP-subproblem

$$\min_{s \in R^p} \nabla_x L(x, \lambda, \rho)^T s + \frac{1}{2} s^T B^L s \quad \text{s.t.} \;\; c(x) + A(x)s = 0.$$

Step 3. Set

$$\begin{aligned} x_+ &= x + s, & B^s &= \rho A(x_+)A(x_+) + B^l, \\ \lambda_+ &= \lambda + \delta\lambda, & v^L &= v^{\text{BFGS}}(s, y^{Ls}, B^s), \\ y^p &= y^l, & B_+^l &= B^l + U(s, y^p, B^l, v^L), \\ y^{Ls} &= y^p + \rho A(x_+)^T A(x_+)s, & B_+^L &= B_+^l + \rho A(x_+)A(x_+). \end{aligned}$$

The following proposition is useful in exploiting the structure of Algorithm 3.1. We represent it to fit our present need.

PROPOSITION 3.1. *Suppose that B^l is positive definite on $N(x) = \{s : A(x)s = 0\}$. Then the SQP augmented Lagrangian secant method which uses*

$$B^{Ls} = B^l + \rho A(x)^T A(x)$$

as an approximation to $\nabla_x^2 L(x, \lambda, \rho)$ generates the same iterate as the SQP Lagrangian secant method that uses B^l as an approximation to $\nabla_x^2 l(x, \lambda)$.

Applying Proposition 3.1, we know that under appropriate conditions, Algorithm 3.1 is equivalent to the following Algorithm 3.2 which was called the SQP augmented scale Lagrangian secant algorithm by Tapia [19]. However, from a practical point of view, their performances can be different. The lines differing from Algorithm 3.1 are in boldface.

ALGORITHM 3.2 (Tapia [19]).

Step 1. **Given** $\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{B^l}$.

Step 2. Compute the solution $(s, \delta\lambda)$ of the QP-subproblem

$$\min_{\boldsymbol{s \in R^p}} \boldsymbol{\nabla_x l(x, \lambda)^T s + \frac{1}{2} s^T B^l s} \quad \text{s.t. } c(x) + A(x)s = 0.$$

Step 3. Set

$$\begin{aligned} x_+ &= x + s, \\ \lambda_+ &= \lambda + \delta\lambda, \\ y^p &= y^l, \end{aligned} \qquad \begin{aligned} y^{Ls} &= y^p + \rho A(x_+)^T A(x_+)s, \\ B^s &= \rho A(x_+)^T A(x_+) + B^l, \\ v^L &= v(s, y^{Ls}, B^s), \\ B_+^l &= B^l + U(s, y^p, B^l, v^L). \end{aligned}$$

It is worth noting that Algorithm 3.2 could be viewed as an SQP Lagrangian secant method, where only the part of the secant update corresponding to scale was changed. This change of scale seems to be important for updates which require a positive definite Hessian approximation, which has been confirmed by our numerical experiments.

If we have additional structure in $\nabla_x^2 l(x, \lambda)$ or $\nabla^2 f(x)$, as is the case in the equality constrained maximum likelihood estimation, we can apply the structure principle (2.11)–(2.17) to also incorporate this available part into the algorithm to obtain a better approximation. This idea has been exploited by Huschens [9] for solving nonlinear least squares problems with equality constraints. However, Huschens's method is restricted to a local framework.

Recall from (2.7) that the Hessian of the augmented Lagrangian function consists of two parts: one part is $C(x) = M(x)^T M(x) + \rho A(x)^T A(x)$ containing first-order terms and the other is the second-order term $S(x, \lambda, \rho)$. One can apply the structure principle (2.11)–(2.17) to this setting with $H_1(x) = C(x)$, $H_2(x) = S(x, \lambda, \rho)$. Before we give our main results, we have to discuss the choice of y^p that is used to approximate $S(x_+, \lambda_+, \rho_+)s$.

Recall that $F(x)$, $T(x)$, $J(x)$ and $M(x)$ have been defined in Section 2 and

$$S(x, \lambda, \rho) = -\frac{1}{n} \sum_{i=1}^{n} \frac{1}{f_i(x)} \nabla^2 f_i(x) + \sum_{i=1}^{m} \lambda^{(i)} \nabla^2 c_i(x) + \rho \sum_{i=1}^{m} c_i(x) \nabla^2 c_i(x). \tag{3.6}$$

First, note that when x and x_+ are close to the solution $\hat{x}_n$,

$$\begin{aligned}\sum_{i=1}^{m} \lambda_+^{(i)} \nabla^2 c_i(x_+)s &= \sum_{i=1}^{m} \lambda_+^{(i)} [\nabla c_i(x_+) - \nabla c_i(x)] + O(\|s\|_2^2) \\ &= [A(x_+) - A(x)]^T \lambda_+ + O(\|s\|_2^2) \\ &\approx [A(x_+) - A(x)]^T \lambda_+ \end{aligned} \tag{3.7}$$

and

$$\begin{aligned}\sum_{i=1}^{n} \frac{1}{f_i(x_+)} \nabla^2 f_i(x_+)s &= \sum_{i=1}^{n} \frac{1}{f_i(x_+)} [\nabla f_i(x_+) - \nabla f_i(x)] + O(\|s\|_2^2) \\ &= [J(x_+) - J(x)]^T T(x) + O(\|s\|_2^2) \\ &\approx [J(x_+) - J(x)]^T T(x). \end{aligned} \tag{3.8}$$

Secondly, we have at the solution $\hat{x}_n$

$$\rho \sum_{i=1}^{m} c_i(\hat{x}_n) \nabla^2 c_i(\hat{x}_n) = 0. \tag{3.9}$$

Substituting (3.7) and (3.8) into (3.6) and ignoring $\rho \sum_{i=1}^{m} c_i(x) \nabla^2 c_i(x)$ due to (3.9), we obtain

$$S(x_+, \lambda_+, \rho_+)s \approx -\frac{1}{n}[J(x_+) - J(x)]^T T(x_+) + [A(x_+) - A(x)]^T \lambda_+.$$

Therefore we choose

$$y_1^p = -\frac{1}{n}[J(x_+) - J(x)]^T T(x_+) + [A(x_+) - A(x)]^T \lambda_+$$

as our approximation to $S(x_+, \lambda_+, \rho_+)s$.

An alternative simple formulation for an approximation to $S(x_+, \lambda_+, \rho_+)s$ is derived by ignoring the term

$$\frac{1}{n} \sum_{i=1}^{n} \frac{1}{f_i(x_+)} \nabla^2 f_i(x_+),$$

as well as the term $\rho \sum_{i=1}^{m} c_i(x) \nabla^2 c_i(x)$, yielding

$$y_2^p = [A(x_+) - A(x)]^T \lambda_+.$$

Now we are ready to state our SQP augmented Lagrangian structured secant algorithm, which we call Algorithm 3.3 (we replace B and W by B^{Ls} and B^{ls}, respectively). This algorithm is analogous to Algorithm 3.1 but pays attention to the structure of the Hessian of the objective function. The lines differing from Algorithm 3.2 are in boldface.

ALGORITHM 3.3.

Step 1. **Given** $\boldsymbol{x, \ \lambda, \ B^{ls}}$ **and** $\boldsymbol{B^{Ls} = B^{ls} + M(x)^T M(x) + \rho A(x)^T A(x)}$.

Step 2. Compute the solution $(s, \delta\lambda)$ of the QP-subproblem

$$\boldsymbol{\min_{s \in R^p} \nabla_x L(x, \lambda, \rho)^T s + \frac{1}{2} s^T B^{Ls} s} \quad \text{s.t. } c(x) + A(x)s = 0.$$

Step 3. Set

$$\begin{aligned}
x_+ &= x + s, & \boldsymbol{y^{Ls}} &\boldsymbol{= y^p + M(x)^T M(x)s + \rho A(x_+)^T A(x_+)s,} \\
\lambda_+ &= \lambda + \delta\lambda, & \boldsymbol{B^s} &\boldsymbol{= M(x_+)^T M(x_+) + \rho A(x_+)^T A(x_+) + B^{ls},} \\
\boldsymbol{y^p} &\boldsymbol{= y_1^p,} & \boldsymbol{B_+^{ls}} &\boldsymbol{= B^{ls} + U(s, y^p, B^{ls}, v^{Ls}),} \\
\boldsymbol{v^{Ls}} &\boldsymbol{= v(s, y^{Ls}, B^s),} & \boldsymbol{B_+^{Ls}} &\boldsymbol{= B_+^{ls} + M(x_+)^T M(x_+) + \rho A(x_+)^T A(x_+).}
\end{aligned}$$

As with Algorithm 3.2, Proposition 3.1 motivates the following algorithm by dropping the term $\rho A(x)^T A(x)$ from the matrix B^{Ls}, which we call Algorithm 3.4 (we replace W by B^{ls}). This algorithm is analogous to the SQP augmented scale structured Lagrangian secant method by Huschens [9]. The lines differing from Algorithm 3.3 are in boldface.

ALGORITHM 3.4.

Step 1. **Given** $\boldsymbol{x, \ \lambda, \ B^{ls}}$ **and** $\boldsymbol{B = B^{ls} + M(x)^T M(x)}$.

Step 2. Compute the solution $(s, \delta\lambda)$ of the QP-subproblem

$$\boldsymbol{\min_{s \in R^p} \nabla_x l(x, \lambda)^T s + \frac{1}{2} s^T B s} \quad \text{s.t. } c(x) + A(x)s = 0. \tag{3.10}$$

Step 3. Set

$$\begin{aligned}
x_+ &= x + s, & y^{Ls} &= y^p + M(x)^T M(x)s + \rho A(x_+)^T A(x_+)s, \\
\lambda_+ &= \lambda + \delta\lambda, & B^s &= M(x_+)^T M(x_+) + \rho A(x_+)^T A(x_+) + B^{ls}, \\
y^p &= y_1^p, & B_+^{ls} &= B^{ls} + U(s, y^p, B^{ls}, v^{Ls}, \\
v^{Ls} &= v^{\text{BFGS}}(s, y^{Ls}, B^s), & \boldsymbol{B_+^{Ls}} &\boldsymbol{= B_+^{ls} + M(x_+)^T M(x_+).}
\end{aligned}$$

Naturally, Algorithm 3.4 can be interpreted as a structured version of Tapia's SQP augmented scale Lagrangian secant method. However, the structure of the objective function is also exploited here. It shares the advantage that the augmentation parameter ρ only appears in the scale v but also incorporates all inexpensive first-order parts explicitly. This algorithm will be the point of interest for the rest of this work.

In Algorithm 3.4, if we set $B^{ls} \equiv 0$ at each iteration, then we get a SQP Gauss-Newton algorithm

$$\min_{x \in R^p} \nabla_x l(x, \lambda)^T s + \frac{1}{2} s^T [M(x)^T M(x)] s \quad \text{s.t. } c(x) + A(x)s = 0, \tag{3.11}$$

which is akin to using the Gauss-Newton method to solve

$$\begin{cases} \nabla_x l(x, \lambda) = 0, \\ \rho c(x) = 0, \end{cases}$$

but it pays more attention to the constraints $c(x)$ if a large penalty parameter ρ is used. This SQP Gauss-Newton algorithm also has a form analogous to (2.6) in Osborne's multiplier method, except that we now update the variables x and λ simultaneously, which we believe should work better.

Similar to the case of Algorithm 3.2, see Tapia [19]; one can show how the positive definiteness of B produced by Algorithm 3.4 is retained.

PROPOSITION 3.2. *Assume $B^{ls} + M(x_+)^T M(x_+)$ is symmetric and positive definite on $N(x_+)$ and $[y^p + M(x_+)^T M(x_+)s]^T s > 0$ for $s \in N(x_+)$. Let the augmentation parameter ρ be sufficiently large and v^{Ls} be such that $U(s, y^{Ls}, B^s, v^{Ls})$ can be viewed as an update of the Broyden convex class, that is,*

$$v^{Ls} = y^{Ls} + \tau B^s s, \quad \tau \in \left[0, \sqrt{s^T y^{Ls} / s^T B^s s}\right].$$

Then the matrix B_+ is positive definite on $N(x_+)$.

4. Implementation of Algorithm 3.4

Our numerical experiments shows that for good performance of Algorithm 3.4, a careful implementation is needed. We implement Algorithm 3.4 described above by including several additional interesting features.

- A trust-region strategy is used to make the algorithm globally convergent. Instead of subproblem (3.10), at each iteration the quadratic model $q(s, \lambda, B)$

$$\min_{s \in R^p} \nabla_x l(x, \lambda)^T s + \frac{1}{2} s^T B s \quad \text{s.t. } c(x) + A(x)s = 0, \ \|s\|_2 \le \Delta \tag{4.1}$$

is solved for the solution $(\tilde{s}, \delta\tilde{\lambda})$, where $B = M(x)^T M(x) + B^{ls}$ or $B = M(x)^T M(x)$, depending on which works better. A scaled trust region would clearly enhance the performance of the inner trust region iteration, but this has not yet been implemented. To avoid an empty feasible set because of the restricted step size, some modifications are needed. This will be discussed in Section 4.1.

- A model switching strategy is developed. Our software implementation includes two quadratic models. It sometimes uses the SQP Gauss-Newton algorithm (3.11) instead of the augmented algorithm (4.1). Because of the above mentioned difficulty in approximating $S(x, \lambda, \rho)$ by finite differences and because the SQP Gauss-Newton algorithm tends to do well initially, we use the zero matrix for the initial matrix B_0^{ls}, so that initially the two algorithms are equivalent. Regardless of which was used to make the step from x to x_+, the update of B^{ls} to B_+^{ls} is always made, but the step from x to x_+ is calculated using the algorithm whose predicted reduction best matches the actual reduction from x to x_+. The first trial step is calculated using the currently preferred algorithm whose predicted reduction best matches the actual reduction in the last iteration. When the first trial step fails, we test the alternate model to see if it would have predicted the observed failure at that point. If so, the alternate algorithm gets a chance to make a trial step with the same trust radius. If we do not decide to try changing models, or if the alternate algorithm fails to suggest a more successful step with the same trust radius, then we assume for the duration of the present iteration that our current algorithm preference is correct. We then decrease the trust radius until x_+ is determined or the algorithm fails.

Typically, this adaptive modelling causes us to use SQP Gauss-Newton steps until B^{ls} builds us useful second-order information, and then to switch to augmented steps defined by (4.1).

Finally, before each update B^{ls} in Algorithm 3.4, B^{ls} is multiplied by the *sizing* factor

$$\gamma = \min\left\{\frac{s^T y^p}{s^T B^{ls} s}, 1\right\},$$

we do this as in Dennis *et al.* [3].

4.1. Solving trust region subproblems We prefer the approach of Byrd and Omojokun for solving the trust region subproblem (4.1). The algorithm decomposes each constrained SQP-subproblem (4.1) into two smaller trust region subproblems which are easier to solve. This makes the Byrd-Omojokun method attractive. This method was also exploited for solving large scale problems by Lalee *et al.* [10]. However, as is well-known, restricting the size of the step by $\|s\|_2 \leq \Delta$ may preclude us from satisfying the linear constraints $c(x) + A(x)s = 0$. Therefore Byrd and Omojokun first compute a step that lies well within the trust region and that satisfies the linear constraints as much as possible. This is done by defining a relaxation parameter $\zeta \in (0, 1)$ and computing a step v that solves the vertical (or normal) subproblem

$$\min_{v \in R^p} \|A(x)v + c(x)\|_2 \quad \text{s.t.} \quad \|v\|_2 \leq \zeta\Delta. \tag{4.2}$$

This problem can have many solutions. Among these solutions the most interesting is that which lies in the range space of $A(x)^T$. This allows us to completely decouple this subproblem from the next one. In the following, we will use the notation A and c to denote $A(x)$ and $c(x)$, respectively, and the same rules carry over to other quantities. Also we denote $\nabla_x l(x, \lambda)$ and the matrix $B^{ls} + M(x)^T M(x)$ or $M(x)^T M(x)$ by g and B, respectively.

This algorithm is designed so that the full step s need not move any closer to the feasible manifold than v does, so we next reformulate (4.1) as

$$\min_{s \in R^p} s^T g + \frac{1}{2} s^T B s \quad \text{s.t.} \;\; As = Av, \;\; \|s\|_2 \leq \Delta. \tag{4.3}$$

This problem, unlike (4.1), has a nonempty feasible region because it already contains v. We solve for s by seeking a step complementary to v. To this end, we compute a $p \times (p - m)$ matrix Z whose columns form the orthogonal basis for the null space of A such that $AZ = 0$ and $Z^T Z = I$, and we define the total step of the algorithm as $s = v + Zu$, where the vector $u \in R^{p-m}$ is yet to be determined. Substituting for s in (4.3), noting that v and Zu are orthogonal, and ignoring constant terms, we obtain

$$\min_{u \in R^{p-m}} u^T [Z^T (g + Bv)] + \frac{1}{2} u^T Z^T B Z u \quad \text{s.t.} \;\; \|u\|_2 \leq \sqrt{\Delta^2 - \|v\|_2^2}.$$

If we define $\tilde{g} = Z^T (g + Bv)$, $\tilde{B} = Z^T B Z$ and $\tilde{\Delta} = \sqrt{\Delta^2 - \|v\|_2^2}$, then we get the following equivalent system

$$\min_{u \in R^{p-m}} h(u) = u^T \tilde{g} + \frac{1}{2} u^T \tilde{B} u \quad \text{s.t.} \;\; \|u\|_2 \leq \tilde{\Delta}. \tag{4.4}$$

Note that this has the same form as a trust region step in an unconstrained algorithm. We use the dogleg method for approximately solving the trust region problem (4.4) because the dogleg method requires the matrix $\tilde{B}$ to be positive definite. Therefore, when $\tilde{B}$ is not positive definite, we recommend changing $\tilde{B}$ to $\tilde{B} + \mu_c I$, where $\mu_c > 0$ is not much larger, ideally, than the smallest μ that will make $\tilde{B} + \mu I$ positive definite and reasonably well conditioned. Procedures for doing this can be found in Algorithm A5.5.1 of Dennis *et al.* [4]. For simplicity of notation, we still use $\tilde{B}$ to denote the resulting matrix if necessary.

Cauchy point. The Cauchy point u^{cp} of (4.4) is defined as the minimiser of $h(u)$ in the direction of steepest descent at $u = 0$, subject to the trust constraint. Therefore we have

$$u^{\text{cp}} = -\tilde{\alpha} \tilde{g}, \tag{4.5}$$

where

$$\tilde{\alpha} = \begin{cases} (\tilde{g}^T \tilde{g}) / (\tilde{g}^T \tilde{B} \tilde{g}) & \text{if} \;\; (\tilde{g}^T \tilde{g})^{3/2} / (\tilde{g}^T \tilde{B} \tilde{g}) \leq \tilde{\Delta}, \\ \tilde{\Delta} / \|\tilde{g}\|_2 & \text{otherwise} \end{cases}$$

and the Newton step $u^n = -\tilde{B}^{-1}\tilde{g}$. The dogleg path consists of the two segments from $u = 0$ to $u = u^{cp}$ and from $u = u^{cp}$ to $u = u^n$. The dogleg method finds the minimiser along this path subject to $\|u\|_2 \le \tilde{\Delta}$. Since h decreases monotonically along the path, we simply find the intersection point with the trust region boundary, or we use the Newton step if the path lies entirely inside the trust region. We denote the solution of this horizontal (or tangential) subproblem by $\hat{u}$, and define the total step as

$$\hat{s} = v + Z\hat{u}, \quad \widehat{\delta\lambda} = -(AA^T)^{-1}A(g + B\hat{s}). \tag{4.6}$$

We then set $s = \hat{s}$, $\delta\lambda = \widehat{\delta\lambda}$, $x_+ = x + s$, $\lambda_+ = \lambda + \delta\lambda$, provided x_+ gives a satisfying reduction in the merit function; otherwise, we change models or retain our current model and reduce the trust region to compute a new trial step.

Let us reiterate our strategy to solve the trust region subproblem (4.2). It is easy to see that it is equivalent to the following problem:

$$\min_{v \in R^p} v^T A^T c + \frac{1}{2} v^T A^T A v \quad \text{s.t. } \|v\|_2 \le \zeta \Delta. \tag{4.7}$$

Comparing (4.7) with the formulation (4.4), we see that $u = v$, $\tilde{B} = A^T A$ is positive semidefinite, $\tilde{\Delta} = \zeta\Delta$ and $\tilde{g} = A^T c$. Substituting into (4.5), we obtain the Cauchy point $v^{\mathrm{cp}} = -\alpha A^T c$, where

$$\alpha = \begin{cases} \|A^T c\|_2^2 / \big(c^T A(A^T A) A^T c\big) & \text{if } \|A^T c\|_2^3 / \big(c^T A(A^T A) A^T c\big) \le \zeta\Delta, \\ \zeta\Delta / \|A^T c\|_2 & \text{otherwise.} \end{cases}$$

To formulate a dogleg method for (4.7) we need to define the Newton step. Since we want the solution v to lie in the range space of A^T, we choose the Newton step as the shortest step in the whole manifold of minimisers $\min_{v \in R^p} v^T A^T c + \frac{1}{2} v^T A^T A v$, which is uniquely given by $v^n = -A^T(AA^T)^{-1}c$. Together, v^{cp} and v^n define a dogleg method for the problem (4.7).

4.2. Merit function and choice of penalty parameter ρ. Let $(\hat{s}, \widehat{\delta\lambda})$ be the step computed by the above procedure using the currently preferred model; see (4.6). The merit function is used to decide whether the step $\hat{s}$ makes sufficient progress toward the solution of the problem (2.1). We follow the idea in El-Alem [5] and use as a merit function the augmented Lagrangian

$$\phi(x, \lambda, \rho) = K(x) + \lambda^T c(x) + \frac{\rho}{2} c(x)^T c(x). \tag{4.8}$$

The actual reduction in the merit function as we go from (x, λ) to $(x + \hat{s}, \lambda + \widehat{\delta\lambda})$ is given by ared $= \phi(x, \lambda, \rho) - \phi(x + \hat{s}, \lambda + \widehat{\delta\lambda}, \rho)$.

We can write

$$\text{ared} = l(x, \lambda) - l(x + \hat{s}, \lambda + \widehat{\delta\lambda}) + \frac{\rho}{2}\left[\|c(x)\|_2^2 - \|c(x + \hat{s})\|_2^2\right].$$

Let us replace $K(x)$ by the model objective (4.3) with an augmentation term K which is the estimate of $K(x)$ at x and linearise the constraints in (4.8) and the multiplier λ to give the model merit function ψ at (x, λ)

$$\psi(s, \delta\lambda, \rho) = K + s^T g + \frac{1}{2} s^T B s + (\lambda + \delta\lambda)^T [c + As] + \frac{\rho}{2}\|c + As\|_2^2.$$

We now define the predicted reduction pred in the model merit function ψ by

$$\begin{aligned} \text{pred} &= \psi(0, 0, \rho) - \psi(\hat{s}, \widehat{\delta\lambda}, \rho) \\ &= -\hat{s}^T g - \frac{1}{2}\hat{s}^T B \hat{s} - \widehat{\delta\lambda}^T (c + A\hat{s}) + \frac{\rho}{2}\left[\|c\|_2^2 - \|c + A\hat{s}\|_2^2\right]. \end{aligned}$$

For convenient presentation of our model switching strategy, we use $q(s, \lambda, B)$ and $\psi(s, \delta\lambda, \rho)$ to denote the currently preferred model and its model merit function, respectively. We also use $q^q(s, \lambda, B^a)$ for the alternate model and $\psi^a(s, \delta\lambda, \rho)$ for the corresponding model merit function.

We accept the step and set $x_+ = x + \hat{s}$ and $\lambda_+ = \lambda + \widehat{\delta\lambda}$, if

$$\text{ared}/\text{pred} \geq \eta_1, \tag{4.9}$$

where $\eta_1 \in (0, 1)$ is a small fixed constant.

If the step is rejected, we first test whether it might be useful to try changing models, but only if this is the first time through (4.9) in the current iteration. If

$$\frac{|\psi(\hat{s}, \widehat{\delta\lambda}, \rho) - \phi(\hat{s}, \widehat{\delta\lambda}, \rho)|}{|\psi^a(\hat{s}, \widehat{\delta\lambda}, \rho) - \phi(\hat{s}, \widehat{\delta\lambda}, \rho)|} > 1.5 \tag{4.10}$$

then we change our model preference with the same trust radius and penalty parameter ρ and return to test (4.9); otherwise we retain our current model preference and decrease the radius of the trust region by picking

$$\Delta_+ \in \left[\alpha_1 \|\hat{s}\|_2, \alpha_2 \|\hat{s}\|_2\right],$$

where $0 < \alpha_1 < \alpha_2 < 1$.

If the step is accepted, then the trust region radius is updated by comparing the value of ared with pred. Namely, if $\eta_1 \leq \text{ared}/\text{pred} < \eta_2$ where $\eta_2 \in (\eta_1, 1)$, then the radius of the trust region is updated by $\Delta_+ = \min[\Delta, \alpha_3 \|\hat{s}\|_2]$, where $\alpha_3 > 1$. However, if $\text{ared}/\text{pred} \geq \eta_2$, then we increase the radius of the trust region by setting $\Delta_+ = \min[\Delta_*, \max(\Delta, \alpha_3 \|\hat{s}\|_2)]$, where Δ_* is a positive constant. This can be summarised in Scheme 4.1.

SCHEME 4.1.

```
Step 1.  Set chanmodel = false;
Step 2.  If ared/pred $< \eta_1$, then
             If (4.10) is true and chanmodel == false, then
                 chanmodel = true
                 set $q(s, \lambda, B) = q^a(s, \lambda, B^a)$
                 recompute $(\hat{s}, \widehat{\delta\lambda})$ by solving $q(s, \lambda, B)$
                 return to Step 2
             Else
                 set $\Delta \in [\alpha_1 \|\hat{s}\|_2, \alpha_2 \|\hat{s}\|_2]$
                 If $\Delta > 10^{-8}$, then
                     chanmodel = true
                     return to Step 2
                 Else exit
         Else if $\eta_1 \le$ ared/pred $< \eta_2$, set
             $x_+ = x + \hat{s}$
             $\lambda_+ = \lambda + \delta\lambda$
             $\Delta_+ = \min[\Delta, \alpha_3 \|\hat{s}\|_2]$
         Else set
             $x_+ = x + \hat{s}$
             $\lambda_+ = \lambda + \delta\lambda$
             $\Delta_+ = \min[\Delta_*, \max(\Delta, \alpha_3 \|\hat{s}\|_2)]$
```

Our numerical testing suggests that the following values give good performances:

$$\begin{aligned} &\eta_1 = 0.1, \qquad &&\alpha_1 = \alpha_2 = 0.5, \qquad &&\Delta_0 = 1, \\ &\eta_2 = 0.75, \qquad &&\alpha_3 = 1.5, \qquad &&\Delta_* = 1000. \end{aligned} \tag{4.11}$$

Now we describe our strategy for updating the penalty parameter ρ. Numerical experiments have suggested that efficient performance of the algorithm requires us to keep the penalty parameter as small as possible. However, global convergence theory requires that the sequence ρ be nondecreasing, and that the predicted reduction in the merit function at each iteration be at least as much as a fraction of the Cauchy decrease of the residual of the linearised constraints. The idea now is to keep the penalty parameter as small as possible, subject to satisfying these two conditions needed for the convergence. Hence our strategy will be to start with $\rho = 1$ and increase it only when necessary for satisfying these two conditions. The aim can be achieved by the following strategy.

SCHEME 4.2.

If pred $\geq \frac{\rho}{4}\left[\|c\|_2^2 - \|c + A\hat{s}\|_2^2\right]$ then set $\rho_+ = \rho$.

Else set

$$\rho_+ = 4\,\frac{g^T\hat{s} + \frac{1}{2}\hat{s}^T B\hat{s} + \widehat{\delta\lambda}^T(c + A\hat{s})}{\|c\|_2^2 - \|c + A\hat{s}\|_2^2} + 0.01.$$

As a consequence of the updating of ρ, one can show that

$$\text{pred} \geq \frac{\rho_+}{4}\left[\|c\|_2^2 - \|c + A\hat{s}\|_2^2\right].$$

The following represents the outline of our algorithm MLESOL for constrained maximum likelihood estimation.

Algorithm MLESOL.

```
Step 0.  Set $x_0 \in R^p$, $B_0^{ls} \in R^{p\times p}$, $\lambda_0 \in R^m$, $\rho_0 \geq 0$, $\Delta_0 > 0$, $0 < \alpha_1 \leq \alpha_2 \leq 1$, $\alpha_3 > 1$, $0 < \eta_1 \leq \eta_2 \leq 1$, $\epsilon$, $k = 0$;
Step 1.  If $\|\nabla l(x_k, \lambda_k)\|_2 = \left\|[\nabla_x l(x_k, \lambda_k)^T, c(x)^T]^T\right\|_2 \leq \epsilon$, then stop;
Step 2.  Compute $(\hat{s}_k, \widehat{\delta\lambda_k})$ according to Section 4.1 above;
Step 3.  Update the penalty parameter according to Scheme 4.2;
Step 4.  Test the step, the model choice and update $\Delta_k$ according to Scheme 4.1;
Step 5.  Update $B_k^{ls}$ and $B_k$ as in Section 3.
Step 6.  Set $k := k + 1$ and go to Step 1.
```

5. Numerical results

We tested our new algorithm on the estimation of equality constrained mixture density problem, which is defined by $x^T = [\beta_1, \mu_1, \sigma_1, \beta_2, \mu_2, \sigma_2]$ and

$$f(y \mid x) = \frac{\beta_1}{\sqrt{2\pi}\sigma_1}\exp\left[-\frac{(y-\mu_1)^2}{2\sigma_1^2}\right] + \frac{\beta_2}{\sqrt{2\pi}\sigma_2}\exp\left[-\frac{(y-\mu_2)^2}{2\sigma_2^2}\right]$$

subject to the constraints

$$c_1(x) = \beta_1 - \frac{\mu_1}{\mu_1+\mu_2} = 0, \qquad c_2(x) = \beta_2 - \frac{\mu_2}{\mu_1+\mu_2} = 0.$$

Numerical results are presented for computations carried out using $\mu_1 = 1.0$ and $\mu_2 = 2.0$ for two cases:

- $\sigma_1 = \sigma_2 = 0.5$;
- $\sigma_1 = \sigma_2 = 0.7$.

Random numbers are generated according to a realisation of the assumed true model incorporating the use of the acceptance/rejection strategy in Ripley [16]. For testing purposes, the stopping tolerance was $\epsilon = 10^{-4}$, the values of the other parameters are set in (4.11).

The algorithms described above have been programmed and tested on a Sun Ultra Sparc 5 Workstation in double precision C with compiler Sun C version 4.2. Its operating system is Solaris 2.6. We also solved this problem with Schittkowski's NLPQL method [17] and Osborne's method [15], using the same tolerance of 10^{-4}. In Osborne's method, the inner scoring step was terminated when the norm of the gradient of the augmented Lagrangian was less than 10^{-6} or the number of iterations exceeded 40. In each case, both the assumed true parameters and an arbitrary point were taken as the starting point.

For case 1, we have

(a) starting from the true parameters $\beta_1 = 0.33$, $\mu_1 = 1.0$, $\sigma_1 = 0.5$, $\beta_2 = 0.67$, $\mu_2 = 2.0$, $\sigma_2 = 0.5$;

(b) starting from the hypothesised values $\beta_1 = 0.30$, $\mu_1 = 0.6$, $\sigma_1 = 0.4$, $\beta_2 = 0.7$, $\mu_2 = 2.4$, $\sigma_2 = 0.6$.

For case 2, we have

(a) starting from the true parameters $\beta_1 = 0.33$, $\mu_1 = 1.0$, $\sigma_1 = 0.7$, $\beta_2 = 0.67$, $\mu_2 = 2.0$, $\sigma_2 = 0.7$;

(b) starting from the hypothesised values $\beta_1 = 0.30$, $\mu_1 = 0.6$, $\sigma_1 = 1.0$, $\beta_2 = 0.7$, $\mu_2 = 2.4$, $\sigma_2 = 0.5$.

The tests were run for $n = 10$, 10^2, 10^3 and 10^4. Typical results are reported in the appended tables. In summary:

(1) All methods converge to the same solutions and there are no failures.

(2) The performance of MLESOL and Osborne's method becomes better and better as n increases. At the solution the Hessian estimates of the two methods become closer and closer to the true Hessian. No such result exists for NLQPL. Compared with that of Osborne's method, the performance of MLESOL does not deteriorate when n decreases.

(3) MLESOL clearly outperforms Osborne's method and NLPQL in each case.

(4) The model switching in MLESOL is not necessary when n is sufficiently large, so an algorithm using $B_k = M(x_k)^T M(x_k)$ would suffice. The same phenomenon was observed for the augmented version of Osborne's method.

Appendix: Tables

The values of n_{scor} and n_{f} listed in the tables are the numbers of scoring iteration steps and function evaluations in each Powell-Hestenes step, respectively. In the other tables, n_{iter} and n_{f} denote the numbers of iteration steps and function evaluations to satisfy the stopping tolerance.

TABLE 1. Osborne's algorithm starting from the true parameters for case 1.

n	Iteration	1	2	3	4	5	6	7	8	9	10	11	12	13	14
10	n_{score}	38	40	40	12	11	9	8	7	4	4	4	1	1	1
	n_{f}	41	42	44	21	19	15	13	11	6	6	6	1	1	1
10^2	n_{score}	13	3	1	1										
	n_{f}	15	3	1	1										
10^3	n_{score}	10	2	1											
	n_{f}	11	2	1											
10^4	n_{score}	11	2												
	n_{f}	11	2												

TABLE 2. MOA starting from the true parameters for case 1.

n	Iteration	1	2	3	4	5	6	7	8	9	10	11	12
10	n_{score}	24	21	20	19	18	16	14	12	10	7	3	1
	n_{f}	27	21	20	19	18	16	14	12	10	7	3	1
10^2	n_{score}	11	3	2	2								
	n_{f}	12	3	2	2								
10^3	n_{score}	6	2	1	1								
	n_{f}	7	2	1	1								
10^4	n_{score}	11	2										
	n_{f}	11	2										

TABLE 3. Osborne's algorithm starting from the hypothesised parameters for case 1.

n	Iteration	1	2	3	4	5	6	7	8	9	10	11	12	13	14
10	n_{score}	40	40	40	12	11	9	8	7	4	4	4	1	1	1
	n_{f}	41	42	44	21	19	15	13	11	6	6	6	1	1	1
10^2	n_{score}	12	3	1	1										
	n_{f}	14	3	1	1										
10^3	n_{score}	4	2	1											
	n_{f}	4	2	1											
10^4	n_{score}	2	2												
	n_{f}	2	2												

TABLE 4. MOA starting from the hypothesised parameters for case 1.

n	Iteration	1	2	3	4	5	6	7	8	9	10	11	12	13
10	n_{score}	29	21	20	20	19	18	16	14	12	10	7	3	1
	n_{f}	32	21	20	20	19	18	16	14	12	10	7	3	1
10^2	n_{score}	9	3	2	2									
	n_{f}	10	3	2	2									
10^3	n_{score}	4	2	1										
	n_{f}	4	2	1										
10^4	n_{score}	2	2											
	n_{f}	2	2											

TABLE 5. Osborne's algorithm starting from the true parameters for case 2.

n	Iteration	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
10	n_{score}	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	30	21
	n_{f}	102	115	118	125	133	136	136	136	136	135	136	137	136	135	136	102	71
10^2	n_{score}	7	3	2	1													
	n_{f}	7	3	2	1													
10^3	n_{score}	4	2	1														
	n_{f}	4	2	1														
10^4	n_{score}	3	2															
	n_{f}	3	2															

TABLE 6. MOA starting from the true parameters for case 2.

n	Iteration	1	2	3	4	5	6	7	8	9	10	11	12
10	n_{score}	15	16	12	13	7	6	6	6	2	2	3	1
	n_{f}	73	83	57	72	33	26	30	29	7	6	3	1
10^2	n_{score}	7	3	2	2								
	n_{f}	7	3	2	2								
10^3	n_{score}	4	3	1									
	n_{f}	4	3	1									
10^4	n_{score}	3	2										
	n_{f}	3	2										

TABLE 7. Osborne's algorithm starting from the hypothesised parameters for case 2.

n	Iteration	1	2	3	4	5	6	7	8	9	10
10	n_{score}	20	11	11	10	9	8	6	5	3	2
	n_{f}	32	19	19	17	15	13	10	8	5	3
10^2	n_{score}	9	3	2	1						
	n_{f}	9	3	2	1						
10^3	n_{score}	10	2	1							
	n_{f}	10	2	1							
10^4	n_{score}	10	2								
	n_{f}	10	2								

TABLE 8. MOA starting from the hypothesised parameters for case 2.

n	Iteration	1	2	3	4	5	6	7	8	9	10
10	n_{score}	40	39	27	26	19	14	17	10	10	4
	n_{f}	142	146	75	73	64	38	47	29	28	7
10^2	n_{score}	9	3	2	1						
	n_{f}	9	3	2	1						
10^3	n_{score}	13	6	2							
	n_{f}	14	7	2							
10^4	n_{score}	13	4								
	n_{f}	13	4								

TABLE 9. Comparison results of MLESOL and NLQPL: (a) starting from true parameters, (b) starting from hypothesised values.

		(a)				(b)			
		MLESOL		NLQPL		MLESOL		NLQPL	
case 1	n	n_{iter}	n_{f}	n_{iter}	n_{f}	n_{iter}	n_{f}	n_{iter}	n_{f}
	10	9	10	15	23	9	10	17	21
	10^2	11	12	9	12	16	17	11	25
	10^3	4	5	13	22	8	9	13	16
	10^4	3	4	9	18	6	7	15	19
		(a)				(b)			
		MLESOL		NLQPL		MLESOL		NLQPL	
case 2	n	n_{iter}	n_{f}	n_{iter}	n_{f}	n_{iter}	n_{f}	n_{iter}	n_{f}
	10	16	17	15	17	11	12	19	20
	10^2	6	7	21	37	8	9	29	35
	10^3	6	7	9	21	4	5	14	26
	10^4	4	5	16	23	6	7	25	36

References

[1] D. Bunch, “Maximum likelihood estimation of probabilistic choice models”, *SIAM J. Sci. Stat. Comput.* **8** (1987) 56–70.

[2] R. H. Byrd, “Robust trust region methods for constrained optimization”, in *Third SIAM Conference on Optimization, Houston, TX, May, 1987*, (SIAM, 1987).

[3] J. E. Dennis, D. M. Gay and R. E. Welsch, “An adaptive nonlinear least-squares algorithm”, *Trans. Math. Software* **7** (1981) 348–368.

[4] J. E. Dennis and R. B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations* (Prentice-Hall, Englewood Cliffs, NJ, 1983).

[5] M. El-Alem, “A global convergence theory for the Celis-Dennis and Tapia trust region algorithm for constrained optimization”, *SIAM J. Numer. Anal.* **28** (1991) 266–290.

[6] J. R. Engels and H. J. Martinez, “Local and superlinear convergence for partially known quasi-Newton methods”, *SIAM J. Optim.* **1** (1991) 42–56.

[7] D. M. Gay and R. E. Welsch, “Maximum likelihood and quasi-likelihood for nonlinear exponential family regression models”, *J. Amer. Assoc.* **83** (1988) 990–998.

[8] J. D. Gonglewski, “Quasi-Newton methods for maximum-likelihood estimation”, Ph. D. Thesis, University of Houston, TX, 1986.

[9] J. Huschens, “Exploiting additional structure in equality constrained optimization by structured SQP secant algorithms”, *J. Optim. Theory Appl.* **77** (1993) 382–359.

[10] M. Lalee, J. Nocedal and T. Plantenga, “On the implementation of an algorithm for large-scale equality constrained optimization”, *SIAM J. Optim.* **8** (1998) 682–706.

[11] H. J. Martinez, Y. R. Pérez and El Método, “Bfgs estructurado para el problema de maxima verosimilitud”, Reporte Tecnico, Depto. de Matmáticas, Universidad del Valle, Cali, Colombia, 1989.

[12] J. Nocedal and M. Overton, “Projected Hessian updating algorithms for nonlinear constrained optimization calculation”, *SIAM J. Numer. Anal.* **22** (1985) 821–850.

[13] E. O. Omojokun, “Trust region algorithms for optimization with nonlinear equality and inequality constraints”, Ph. D. Thesis, Department of Computer Science, University of Colorado, Boulder, 1989.

[14] M. R. Osborne, “Estimating nonlinear models by maximum likelihood for the exponential family”, *SIAM J. Sci. Stat. Comput.* **8** (1987) 446–456.

[15] M. R. Osborne, “Scoring with constraints”, *ANZIAM J.* **42** (2000) 9–25.

[16] B. D. Ripley, “Computer generation of random variables: A tutorial”, *Internat. Statist. Review* **53** (1983) 301–319.

[17] K. Schittkowski, “NLPQL: A FORTRAN subroutine solving constrained nonlinear programming problems”, *Ann. Oper. Res.* **5** (1985/6) 485–500.

[18] P. K. Sen and J. M. Singer, *Large sample methods in statistics* (Chapman, 1993).

[19] R. A. Tapia, “On secant updates for use in general constrained optimization”, *Math. Comput.* **51** (1988) 181–203.

[20] H. Walker and J. D. Gonglewski, “Quasi-Newton methods for maximum-likelihood estimation”, Technical report, University of Houston, TX, 1986.