IMA Journal of Numerical Analysis (2005) **25**, 264–285
doi:10.1093/imanum/drh016
Advance Access publication 21 January 2005

Parameter estimation of ordinary differential equations

ZHENGFENG LI†
National Centre for Epidemiology and Population Health, Australian National University, Canberra, ACT 0200, Australia

MICHAEL R. OSBORNE‡
School of Mathematical Sciences, Australian National University, Canberra, ACT 0200, Australia

AND

TANIA PRVAN§
Department of Statistics, Macquarie University, Sydney, NSW 2109, Australia

[Received on 8 February 2002; revised on 5 January 2004]

This paper addresses the development of a new algorithm for parameter estimation of ordinary differential equations. Here, we show that (1) the simultaneous approach combined with orthogonal cyclic reduction can be used to reduce the estimation problem to an optimization problem subject to a fixed number of equality constraints without the need for structural information to devise a stable embedding in the case of non-trivial dichotomy and (2) the Newton approximation of the Hessian information of the Lagrangian function of the estimation problem should be used in cases where hypothesized models are incorrect or only a limited amount of sample data is available. A new algorithm is proposed which includes the use of the sequential quadratic programming (SQP) Gauss–Newton approximation but also encompasses the SQP Newton approximation along with tests of when to use this approximation. This composite approach relaxes the restrictions on the SQP Gauss–Newton approximation that the hypothesized model should be correct and the sample data set large enough. This new algorithm has been tested on two standard problems.

Keywords: ordinary differential equations; data fitting; parameter estimation; orthogonal cyclic reduction; constrained optimization; SQP methods; Gauss–Newton approximation.

1. Introduction

Assume that the ordinary differential equations (ODEs), after suitable normalization, have the form

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{\theta}), \tag{1.1}$$

where t denotes the independent variable, usually referred to as time, $\boldsymbol{\theta}$ is a p-dimensional vector of unknown parameters, $\boldsymbol{x} = \boldsymbol{x}(t, \boldsymbol{\theta})$ is an n-dimensional state variable vector depending on t and $\boldsymbol{\theta}$, and the function $\boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{\theta})$ maps $\Re \times \Re^n \times \Re^p$ into $\Re^n$. In addition, side constraints are often given to specify

†Email: zhengfeng.li@anu.edu.au
‡Corresponding Author. Email: Mike.Osborne@maths.anu.edu.au
§Email: tprvan@efs.mq.edu.au

further model properties such as boundary conditions, initial values or parameter restrictions

$$\boldsymbol{c}_p(\boldsymbol{x}(t_1), \boldsymbol{x}(t_f), \boldsymbol{\theta}) = 0 \in R^q, \tag{1.2}$$

where t_1 is the initial time and t_f is the final time.

In order to estimate the unknown parameters, a number of measurements, say N, are available for the process under consideration. These measurements often contain inherent errors and are characterized by

$$\hat{\boldsymbol{y}}_i = \boldsymbol{h}(\boldsymbol{x}(t_i, \boldsymbol{\theta})) + \boldsymbol{\epsilon}_i, \quad i = 1, \ldots, N, \tag{1.3}$$

where $\hat{\boldsymbol{y}}_i \in \Re^n$ is the measured value at t_i, and $\{\boldsymbol{\epsilon}_i\}_{i=1}^N$ are independently and identically normally distributed errors. A common special case is that in which the state variables are observed directly, i.e. $\hat{\boldsymbol{y}}_i = \boldsymbol{x}(t_i, \boldsymbol{\theta}) + \boldsymbol{\epsilon}_i$, $i = 1, \ldots, N$. If the dynamical structure requires a dense grid, but only a few experimental times are available, one could insert dummy values with zero weights. Note that of the set of parameters making up the vector $\boldsymbol{\theta}$, some may enter only in $\boldsymbol{f}$, others only in $\boldsymbol{c}_p$. The parameter estimation problem is to find reasonable values for $\boldsymbol{\theta}$ so that the solution of the system (1.1)–(1.2) with these values fits the given data $\{\hat{\boldsymbol{y}}_i\}_{i=1}^N$.

Using the method of least squares, the parameter estimation problem (1.1)–(1.2) can be formulated as follows:

$$\min m(\boldsymbol{\theta}) = \frac{1}{2N} \sum_{i=1}^{N} [\hat{\boldsymbol{y}}_i - \boldsymbol{h}(\boldsymbol{x}(t_i, \boldsymbol{\theta}))]^{\mathrm{T}} [\hat{\boldsymbol{y}}_i - \boldsymbol{h}(\boldsymbol{x}(t_i, \boldsymbol{\theta}))] \tag{1.4}$$

$$\text{s.t.} \quad \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{\theta}) \tag{1.5}$$

$$\boldsymbol{c}_p(\boldsymbol{x}(t_1), \boldsymbol{x}(t_f), \boldsymbol{\theta}) = 0. \tag{1.6}$$

Traditionally, this kind of problem (1.4)–(1.6) is tackled by the initial-value problem approach, see Hemker (1971) and Bard (1974). However, this approach cannot deal with a case in which the fundamental matrix of the ODE (1.1) has exponentially increasing and decreasing modes or, more generally, has non-trivial dichotomy, see Ascher *et al.* (1995).

An alternative is the embedding approach, see Bock (1983), Nowak & Deuflhard (1985), Deuflhard & Nowak (1986), Bock & Schlöder (1987), Bock *et al.* (1988) and Childs & Osborne (1996). The embedding approach requires additional information on the solution structure such that the ODE model can be stably posed by adjoining suitable boundary conditions, see Osborne (1997). However, choosing an appropriate embedding for a general dynamical system can be difficult since a priori information about the solution structure of the ODE model may not be available: for example, problems occurring in chemical engineering are quoted by Tjoa & Biegler (1991) and Tanartkit & Biegler (1995). Moreover, if this embedding is carried out explicitly then it will increase the number of parameters that must be estimated from the observed data.

Another problem occurs in the solution of the optimization procedure involved. Typically, the SQP Gauss–Newton method is used for solving the resultant optimization problem to take advantage of the least-squares structure of the objective function (1.4), see Bock (1983), Bock & Schlöder (1987), Bock *et al.* (1988) and Childs & Osborne (1996). However, our numerical experiments have demonstrated that the SQP Gauss–Newton method works well only if the fitted model is exact and the sample data set is large enough. The use of the SQP Gauss–Newton method would result in poor performance when only

a limited amount of data is available or an inappropriate model is being fitted to the data as part of the comparison process, see Li (2000).

In this paper, we attempt to develop an efficient and stable algorithm for estimating unknown parameters in an ODE system. The method of least squares is used to define the objective function. In order to pose the ODE model in a stable fashion, we use the simultaneous approach of Tjoa & Biegler (1991). In doing so, the differential equations are transformed into difference equations. Thus the problem becomes a constrained nonlinear least-squares problem, in which both the parameters and the state variables are regarded as unknown variables. Other applications of the simultaneous approach include Irving & Dewson (1997), Baer *et al.* (1999) and Parlitz & Merkwirth (2000). This constrained minimization is large if N, the number of observations, is large. However, its effective degrees of freedom are determined by the ODE, rather than by the size of the data set. This motivates us to reduce it to a minimization problem with a fixed finite number of constraints by the use of orthogonal cyclic reduction. This reduction process is stable and does not need any explicit imposing of extra initial or boundary conditions, see Bock *et al.* (1988), Wright (1992), Gallitzendoerfer & Bock (1994) and Osborne (1997). In order to relax the restrictions on the Gauss–Newton-type optimization method, the treatment of the Hessian approximation in our SQP method combines the best of the Newton Hessian approximation and the Gauss–Newton approximation. A trust region global strategy is also added in our code to make our algorithm more robust.

The remainder of this paper is organized as follows: our new simultaneous approach algorithm is outlined in Section 2; then we present the orthogonal cyclic reduction procedure in Section 3; the details of our implementation are discussed in Section 4; and the final section describes the application of our new algorithm to two standard problems.

2. Outline of the new algorithm

The simultaneous approach circumvents the drawbacks of both the initial-value problem approach and the embedding approach, see Tjoa & Biegler (1991) and Tanartkit & Biegler (1995). Furthermore, this approach does not need any explicit adjoining of boundary conditions to devise a stable embedding.

For the simultaneous approach, the ODE (1.1) is discretized by using finite differences or orthogonal collocation methods so that the discretized equations can be incorporated directly into the optimization formulation together with any other process constraints. The resulting constrained least squares problem is solved by regarding both the parameter $\boldsymbol{\theta}$ and the state variables as unknown variables.

For simplicity, in our implementation, we discretize the ODE (1.1) by using the box scheme. This gives

$$\boldsymbol{x}_{i+1} - \boldsymbol{x}_i = \Delta_t \boldsymbol{f}\left(t_{i+\frac{1}{2}}, \frac{\boldsymbol{x}_i + \boldsymbol{x}_{i+1}}{2}, \boldsymbol{\theta}\right), \quad i = 1, \ldots, N-1, \tag{2.1}$$

where $\boldsymbol{x}_i$ is an approximation to the solution $\boldsymbol{x}(t_i, \boldsymbol{\theta})$ of the problem (1.1)–(1.2) at t_i and Δ_t is the mesh spacing. Thus, problem (1.4)–(1.6) can be formulated as follows:

$$\min m(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N, \boldsymbol{\theta}) = \frac{1}{2N} \sum_{i=1}^{N} [\hat{\boldsymbol{y}}_i - \boldsymbol{h}(\boldsymbol{x}_i)]^{\mathrm{T}} [\hat{\boldsymbol{y}}_i - \boldsymbol{h}(\boldsymbol{x}_i)] \tag{2.2}$$

$$\text{s.t.} \quad \boldsymbol{c}_i(\boldsymbol{x}_i, \boldsymbol{x}_{i+1}, \boldsymbol{\theta}) = 0, \quad i = 1, 2, \ldots, N-1 \tag{2.3}$$

$$\boldsymbol{c}_p(\boldsymbol{x}_1, \boldsymbol{x}_N, \boldsymbol{\theta}) = 0, \tag{2.4}$$

where, for $i = 1, 2, \ldots, N-1$,

$$\boldsymbol{c}_i(\boldsymbol{x}_i, \boldsymbol{x}_{i+1}, \boldsymbol{\theta}) \stackrel{\Delta}{=} \boldsymbol{x}_{i+1} - \boldsymbol{x}_i - \Delta_t \boldsymbol{f}(t_{i+\frac{1}{2}}, \frac{\boldsymbol{x}_i + \boldsymbol{x}_{i+1}}{2}, \boldsymbol{\theta}). \tag{2.5}$$

This kind of discretization method will not contribute to overall accuracy loss as the stochastic errors are $O(N^{-1/2})$ compared with discretization error of $O(N^{-2})$. It has been noted that limitations in the data together with the approximate model involved do not justify the use of more complex quadrature formulae, see Foss (1971).

Problem (2.2)–(2.4) is a typical nonlinear optimization problem. However, general constrained optimization methods such as the Powell–Hestenes method or the SQP method have no special ability to deal with this kind of problem efficiently because they do not take the structure into account. It is desirable to use a method that can take advantage of the least squares structure of the objective function (2.2). For nonlinear least squares problems, Wedin & Lindström (1987) and Li *et al.* (2002) proposed a combination of Gauss–Newton method and Newton method, while Mahdavi-Amiri & Bartes (1989) proposed a reduced quasi-Newton method on the null space of the constraints based on a general exact-penalty-type method. In this context, the special structure of the state constraints (2.3) also require special treatment.

It is also interesting to note that the size of the problem (2.2)–(2.4) is generally large but with few effective degrees of freedom as these are determined by the ODE, rather than by the size of the data set. In order to make the variable elimination procedure efficient and robust, rather than using forward successive Gauss elimination, an elimination technique based on orthogonal cyclic reduction is used here. This reduction process has been used by Bock *et al.* (1988), Wright (1992), Gallitzendoerfer & Bock (1994) and Osborne (1997).

For simplicity of notation, we first consider the case in which the point constraints (2.4) are removed. Thus, we have

$$\min m(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N, \boldsymbol{\theta}) = \frac{1}{2N} \sum_{i=1}^{N} [\hat{\boldsymbol{y}}_i - \boldsymbol{h}(\boldsymbol{x}_i)]^{\mathrm{T}} [\hat{\boldsymbol{y}}_i - \boldsymbol{h}(\boldsymbol{x}_i)] \tag{2.6}$$

$$\text{s.t.} \quad \boldsymbol{c}_i(\boldsymbol{x}_i, \boldsymbol{x}_{i+1}, \boldsymbol{\theta}) = 0, \quad i = 1, 2, \ldots, N-1. \tag{2.7}$$

Before we present our new algorithm, we need to introduce some notation. Define

$$\boldsymbol{z} = [\boldsymbol{x}_1^{\mathrm{T}}, \boldsymbol{x}_2^{\mathrm{T}}, \ldots, \boldsymbol{x}_N^{\mathrm{T}}, \boldsymbol{\theta}^{\mathrm{T}}]^{\mathrm{T}} \in \Re^{Nn+p}, \tag{2.8}$$

$$\boldsymbol{c}_d(\boldsymbol{z}) = [\boldsymbol{c}_1(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{\theta})^{\mathrm{T}}, \ldots, \boldsymbol{c}_{N-1}(\boldsymbol{x}_{N-1}, \boldsymbol{x}_N, \boldsymbol{\theta})^{\mathrm{T}}]^{\mathrm{T}} \in \Re^{(N-1)n} \tag{2.9}$$

and the Lagrangian function associated with (2.6)–(2.7) by

$$l(\boldsymbol{z}, \boldsymbol{\lambda}) = m(\boldsymbol{z}) + \boldsymbol{\lambda}^{\mathrm{T}} \boldsymbol{c}_d(\boldsymbol{z}), \tag{2.10}$$

where

$$\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1^{\mathrm{T}}, \ldots, \boldsymbol{\lambda}_{N-1}^{\mathrm{T}}]^{\mathrm{T}}, \quad \boldsymbol{\lambda}_i \in \Re^n, \ i = 1, \ldots, N-1. \tag{2.11}$$

Let

$$\boldsymbol{r}(\boldsymbol{x}(t)) \stackrel{\Delta}{=} \hat{\boldsymbol{y}} - \boldsymbol{h}(\boldsymbol{x}(t, \boldsymbol{\theta})) \tag{2.12}$$

and denote the residual $\boldsymbol{r}(\boldsymbol{x}(t))$ at the ith experimental measurement by $\boldsymbol{r}_i(\boldsymbol{x}_i)$; let $\boldsymbol{g}(z)$ be the gradient of $m(z)$ with respect to z, i.e. $\boldsymbol{g}(z) = \nabla_z m(z)$ and $A_d(z)$ be the Jacobian of $\boldsymbol{c}_d(z)$, i.e. $A_d(z) = \nabla_z \boldsymbol{c}_d(z)$; then, we have

$$\boldsymbol{g}(z) = \frac{1}{N}\sum_{i=1}^{N} \nabla_z \boldsymbol{r}_i(\boldsymbol{x}_i)^{\mathrm{T}} \boldsymbol{r}_i(\boldsymbol{x}_i), \tag{2.13}$$

$$A_d(z) = \begin{bmatrix} D_1 & C_1 & & & & E_1 \\ & D_2 & C_2 & & & E_2 \\ & & \ddots & \ddots & & \vdots \\ & & & D_{N-1} & C_{N-1} & E_{N-1} \end{bmatrix}, \tag{2.14}$$

where for $i = 1, \ldots, N-1$,

$$D_i = -I - \frac{\Delta_t}{2} \nabla_{\boldsymbol{x}} \boldsymbol{f}(t_{i+\frac{1}{2}}, \boldsymbol{x}, \boldsymbol{\theta}) \mid_{\frac{x_i + x_{i+1}}{2}}, \tag{2.15}$$

$$C_i = I - \frac{\Delta_t}{2} \nabla_{\boldsymbol{x}} \boldsymbol{f}(t_{i+\frac{1}{2}}, \boldsymbol{x}, \boldsymbol{\theta}) \mid_{\frac{x_i + x_{i+1}}{2}}, \tag{2.16}$$

$$E_i = -\Delta_t \nabla_{\boldsymbol{\theta}} \boldsymbol{f}(t_{i+\frac{1}{2}}, \boldsymbol{x}, \boldsymbol{\theta}) \mid_{\frac{x_i + x_{i+1}}{2}} \tag{2.17}$$

and the Hessian of $l(z, \boldsymbol{\lambda})$

$$\begin{aligned} \nabla_z^2 l(z, \boldsymbol{\lambda}) = \frac{1}{N} \Bigg[& \sum_{i=1}^{N} \nabla_z \boldsymbol{r}_i(\boldsymbol{x}_i)^{\mathrm{T}} \nabla_z \boldsymbol{r}_i(\boldsymbol{x}_i) \\ & + \sum_{i=1}^{N} \left(\sum_{j=1}^{n} [\boldsymbol{r}_i]_j(\boldsymbol{x}_i) \nabla_z^2 [\boldsymbol{r}_i]_j(\boldsymbol{x}_i) \right) \\ & + \sum_{i=1}^{N} \nabla_z^2 (\boldsymbol{\lambda}_i^{\mathrm{T}} \boldsymbol{c}_i(z)) \Bigg] \\ \stackrel{\Delta}{=} & \frac{1}{N} [H_1(z) + H_2(z) + H_3(z, \boldsymbol{\lambda})], \end{aligned} \tag{2.18}$$

where

$$\begin{aligned} H_1(z) &= \sum_{i=1}^{N} [\nabla_z \boldsymbol{r}_i(\boldsymbol{x}_i)^{\mathrm{T}} \nabla_z \boldsymbol{r}_i(\boldsymbol{x}_i)] \\ &= \begin{bmatrix} \nabla_{\boldsymbol{x}} \boldsymbol{r}_1(\boldsymbol{x}_1)^{\mathrm{T}} \nabla_{\boldsymbol{x}} \boldsymbol{r}_1(\boldsymbol{x}_1) & & & \\ & \ddots & & \\ & & \nabla_{\boldsymbol{x}} \boldsymbol{r}_N(\boldsymbol{x}_N)^{\mathrm{T}} \nabla_{\boldsymbol{x}} \boldsymbol{r}_N(\boldsymbol{x}_N) & \\ & & & 0_p \end{bmatrix}, \end{aligned} \tag{2.19}$$

$$H_2(z) = \sum_{i=1}^{N}\sum_{j=1}^{n}[\boldsymbol{r}_i]_j(\boldsymbol{x}_i)\nabla_z^2[\boldsymbol{r}_i]_j(\boldsymbol{x}_i)$$
$$= \begin{bmatrix} \sum_{j=1}^{n}[\boldsymbol{r}_1]_j(\boldsymbol{x}_1)\nabla_{\boldsymbol{x}}^2[\boldsymbol{r}_1]_j(\boldsymbol{x}_1) & & \\ & \ddots & \\ & & \sum_{j=1}^{n}[\boldsymbol{r}_N]_j(\boldsymbol{x}_N)\nabla_{\boldsymbol{x}}^2[\boldsymbol{r}_N]_j(\boldsymbol{x}_N) \\ & & & 0_p \end{bmatrix} \tag{2.20}$$

and

$$H_3(z,\boldsymbol{\lambda}) = \sum_{i=1}^{N}\nabla_z^2(\boldsymbol{\lambda}_i^{\mathrm{T}}\boldsymbol{c}_i(z)), \tag{2.21}$$

where 0_p denotes the p by p zero matrix.

Note that

$$\begin{aligned}\frac{\partial^2(\boldsymbol{\lambda}_i^{\mathrm{T}}\boldsymbol{c}_i(z))}{\partial\boldsymbol{x}_i\partial\boldsymbol{x}_i{}^{\mathrm{T}}} &= -\frac{\Delta_t}{4}\frac{\partial^2(\boldsymbol{\lambda}_i^{\mathrm{T}}\boldsymbol{f}(t_{i+\frac{1}{2}},\boldsymbol{x},\boldsymbol{\theta}))}{\partial\boldsymbol{x}\partial\boldsymbol{x}^{\mathrm{T}}}\Big|_{\boldsymbol{x}=\frac{\boldsymbol{x}_i+\boldsymbol{x}_{i+1}}{2}} \\ &= \frac{\partial^2(\boldsymbol{\lambda}_i^{\mathrm{T}}\boldsymbol{c}_i(z))}{\partial\boldsymbol{x}_i\partial\boldsymbol{x}_{i+1}{}^{\mathrm{T}}} \\ &= \frac{\partial^2(\boldsymbol{\lambda}_i^{\mathrm{T}}\boldsymbol{c}_i(z))}{\partial\boldsymbol{x}_{i+1}\partial\boldsymbol{x}_{i+1}{}^{\mathrm{T}}} \\ &\triangleq L_i^{xx},\end{aligned} \tag{2.22}$$

$$\begin{aligned}\frac{\partial^2(\boldsymbol{\lambda}_i^{\mathrm{T}}\boldsymbol{c}_i(z))}{\partial\boldsymbol{x}_i\partial\boldsymbol{\theta}^{\mathrm{T}}} &= -\frac{\Delta_t}{2}\frac{\partial^2(\boldsymbol{\lambda}_i^{\mathrm{T}}\boldsymbol{f}(t_{i+\frac{1}{2}},\boldsymbol{x},\boldsymbol{\theta}))}{\partial\boldsymbol{x}\partial\boldsymbol{\theta}^{\mathrm{T}}}\Big|_{\boldsymbol{x}=\frac{\boldsymbol{x}_i+\boldsymbol{x}_{i+1}}{2}} \\ &= \frac{\partial^2(\boldsymbol{\lambda}_i^{\mathrm{T}}\boldsymbol{c}_i(z))}{\partial\boldsymbol{x}_{i+1}\partial\boldsymbol{\theta}^{\mathrm{T}}} \\ &\triangleq L_i^{x\theta}\end{aligned} \tag{2.23}$$

and

$$\frac{\partial^2(\boldsymbol{\lambda}_i^{\mathrm{T}}\boldsymbol{c}_i(z))}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^{\mathrm{T}}} = -\Delta_t\frac{\partial^2(\boldsymbol{\lambda}_i^{\mathrm{T}}\boldsymbol{f}(t_{i+\frac{1}{2}},\boldsymbol{x},\boldsymbol{\theta}))}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^{\mathrm{T}}}\Big|_{\boldsymbol{x}=\frac{\boldsymbol{x}_i+\boldsymbol{x}_{i+1}}{2}} \triangleq L_i^{\theta\theta}. \tag{2.24}$$

Hence $H_3(z, \lambda)$ has the general almost tridiagonal form

$$\begin{bmatrix} L_1^{xx} & L_1^{xx} & & & & L_1^{x\theta} \\ L_1^{xx} & L_1^{xx}+L_2^{xx} & L_2^{xx} & & & L_1^{x\theta}+L_2^{x\theta} \\ & \ddots & \ddots & \ddots & & \vdots \\ & & L_{N-2}^{xx} & L_{N-2}^{xx}+L_{N-1}^{xx} & L_{N-1}^{xx} & L_{N-2}^{x\theta}+L_{N-1}^{x\theta} \\ & & & L_{N-1}^{xx} & L_{N-1}^{xx} & L_{N-1}^{x\theta} \\ L_1^{\theta x} & L_1^{\theta x}+L_2^{\theta x} & \dots & L_{N-2}^{\theta x}+L_{N-1}^{\theta x} & L_{N-1}^{\theta x} & \sum_{i=1}^{N-1} L_i^{\theta\theta} \end{bmatrix}, \tag{2.25}$$

where

$$L_i^{\theta x} = (L_i^{x\theta})^{\mathrm{T}}. \tag{2.26}$$

By using the KKT conditions, we have

$$\nabla m(z) + A_d(z)^{\mathrm{T}}\lambda = 0. \tag{2.27}$$

Substituting (2.14) and (2.15)–(2.17) into (2.27) and using the notation

$$\nabla_x f_{i+\frac{1}{2}} \stackrel{\Delta}{=} \nabla_x f(t_{i+\frac{1}{2}}, x, \theta)\,|_{\frac{x_i+x_{i+1}}{2}}, \tag{2.28}$$

we obtain

$$\left[-I - \frac{\Delta_t}{2}\nabla_x f_{1+\frac{1}{2}}^{\mathrm{T}}\right]\lambda_1 - \nabla_x h(x_1)^{\mathrm{T}} r_1(x_1) = 0, \tag{2.29}$$

$$\left[I - \frac{\Delta_t}{2}\nabla_x f_{N-\frac{1}{2}}^{\mathrm{T}}\right]\lambda_{N-1} - \nabla_x h(x_N)^{\mathrm{T}} r_N(x_N) = 0, \tag{2.30}$$

$$\sum_{i=1}^{N-1}\left[-\Delta_t \nabla_\theta f_{i+\frac{1}{2}}^{\mathrm{T}}\right]\lambda_i = 0, \tag{2.31}$$

and for $i = 2, \dots, N-1$,

$$\left[-I - \frac{\Delta_t}{2}\nabla_x f_{i-\frac{1}{2}}^{\mathrm{T}}\right]\lambda_{i-1} + \left[I - \frac{\Delta_t}{2}\nabla_x f_{i+\frac{1}{2}}^{\mathrm{T}}\right]\lambda_i - \nabla_x h(x_i)^{\mathrm{T}} r_i(x_i) = 0. \tag{2.32}$$

Hence, we have for $i = 2, \dots, N-1$,

$$\lambda_i - \lambda_{i-1} = -\frac{\Delta_t}{2}\left[\nabla_x f_{i-\frac{1}{2}}^{\mathrm{T}}\lambda_{i-1} + \nabla_x f_{i+\frac{1}{2}}^{\mathrm{T}}\right]\lambda_i - \nabla_x h(x_i)^{\mathrm{T}} r_i(x_i) = 0. \tag{2.33}$$

Therefore, for cases where the hypothesized model is correct and there is no measurement error in our data, we obtain the limit form

$$\frac{\mathrm{d}\lambda_*(t)}{\mathrm{d}t} = -\nabla_x f(t, x_*, \theta_*)^{\mathrm{T}}\lambda_*(t), \tag{2.34}$$

$$\int_0^1 \nabla_\theta f(t, x, \theta_*)^{\mathrm{T}}\lambda_*(t)\mathrm{d}t = 0, \tag{2.35}$$

where $\boldsymbol{x}_*(t)$ is determined by

$$\frac{\mathrm{d}\boldsymbol{x}_*}{\mathrm{d}t} = \boldsymbol{f}(t, \boldsymbol{x}_*, \boldsymbol{\theta}_*). \tag{2.36}$$

It is interesting to note that the trivial solution of problem (2.34)–(2.35) is

$$\boldsymbol{\lambda}_*(t) = 0. \tag{2.37}$$

Our numerical results confirm this theory: we observed that, as an approximation to $\boldsymbol{\lambda}_*(t)$ in the case where the number of observations N is large, $\boldsymbol{\lambda}_N(t)$ is small. Therefore, for simplicity of testing, initially we choose $\boldsymbol{\lambda}_i = 0$ for $i = 1, \ldots, N-1$.

In cases where measurement errors are present $\boldsymbol{r}_i(\boldsymbol{x}_*(t_i)) = \boldsymbol{\epsilon}_i = \int_{t_{i-1}}^{t_i} \mathrm{d}\boldsymbol{w}$, where $\boldsymbol{w}$ is a Wiener process, we will have a stochastic differential equation for $\boldsymbol{\lambda}_*(t)$.

Based on the numerical experiments, we would like to use the SQP method for solving problem (2.6)–(2.7). We now state this formal SQP Lagrangian local method.

At the current iterate z, given an approximate multiplier $\boldsymbol{\lambda} \in \Re^{(N-1)n}$ and a $(Nn + p)$ by $(Nn + p)$ symmetric matrix $B(z)$ to approximate the Hessian of the Lagrangian function $\nabla_z^2 l(z, \boldsymbol{\lambda})$, the SQP Lagrangian local method for solving (2.6)–(2.7) is characterized by the iterative procedure

$$z_+ = z + \bar{\boldsymbol{s}}, \tag{2.38}$$

where $\bar{\boldsymbol{s}} \in \Re^{Nn+p}$ is the solution of the quadratic problem

$$\min_{\boldsymbol{s} \in R^{Nn}} \boldsymbol{g}(z)^{\mathrm{T}} \boldsymbol{s} + \frac{1}{2} \boldsymbol{s}^{\mathrm{T}} B(z) \boldsymbol{s} \tag{2.39}$$

$$\text{s.t.} \quad \boldsymbol{c}_d(z) + A_d(z)\boldsymbol{s} = 0, \tag{2.40}$$

$$\boldsymbol{c}_p(z) + \nabla \boldsymbol{c}_p(z)\boldsymbol{s} = 0. \tag{2.41}$$

The straightforward SQP Newton method is derived by setting $B(z)$ to be

$$\begin{aligned} B^{\mathrm{Ne}}(z) &\triangleq \nabla_z^2 l(z, \boldsymbol{\lambda}) \\ &= \frac{1}{N}[H_1(z) + H_2(z) + H_3(z, \boldsymbol{\lambda})]. \end{aligned} \tag{2.42}$$

This is called the SQP Newton approximation. To enlarge its convergence region, a line search or trust region global convergence strategy is needed.

To compensate for possible lack of positive definiteness of $\nabla_z^2 l(z, \boldsymbol{\lambda})$ and to reduce the computational complexity of the SQP Newton approximation, Bock (1983) proposed to use the SQP Gauss–Newton approximation. This takes $B(z)$ to be

$$B^{\mathrm{GN}}(z) \triangleq H_1(z). \tag{2.43}$$

Subsequently, forward successive block Gauss elimination was used to reduce the number of variables. We should mention that, rather than using the trust region global convergence strategy, the Bock method uses a line search global convergence strategy. Moreover, the Bock method still needs to embed suitable initial conditions for integrating sensitivity equations.

The use of the SQP Gauss–Newton approximation is recommended only if the hypothesized model is correct and the sample data set is large enough. If these assumptions hold, it has the advantages of efficiency and stability. Otherwise, this kind of approximation could result in poor performance. In order to improve the performance of the SQP Gauss–Newton approximation, Biegler and his co-workers (Tjoa & Biegler, 1991 and Tanartkit & Biegler, 1995) proposed a simple hybrid SQP method based on a test on the relative merit function value. If the relative merit function value is small, the SQP Gauss–Newton approximation is used; otherwise, the choice is the SQP BFGS approximation. However, this switching rule may refuse the use of the SQP Gauss–Newton approximation even if its performance is satisfactory. Thus, it reduces the efficiency of the algorithm. If the data set is large, then the law of large numbers can be used to show that the size of the residuals is not the point. If the sample size is large enough, the effect of these large residuals can be pinned down due to cancellation.

The size of problem (2.39)–(2.40) is large but it has few effective degrees of freedom, so the variable reduction method has a natural application, see Fletcher (1987, pp. 230–236). To take advantage of the almost block-bidiagonal structure of the linearized constraints arising from the ODE, forward successive block Gauss elimination was used in Bock (1983). This reduction technique is similar to the compactification numerical method for the boundary value problems of ODEs for it can suffer from instability, much like the single shooting method, see Ascher *et al.* (1995). In the earlier version of the Tjoa and Biegler method in Tjoa & Biegler (1991) and Tanartkit & Biegler (1995), a complete pivoting Gauss elimination method was used without taking advantage of the almost block-bidiagonal structure. Later, Biegler (1998) proposed the use of the block Gauss elimination method to take advantage of the almost block-bidiagonal structure. However, in these examples, the aim of the elimination is to simplify overhead calculation rather than to explicitly reduce the number of variables in the optimization problem.

For these reasons, a new SQP algorithm is proposed that incorporates a number of novel features:

1. A different approach to distinguish between two competing models (the SQP Gauss–Newton approximation and the SQP Newton approximation) is taken: our local quadratic model contains the use of either the SQP Gauss–Newton approximation or the SQP Newton approximation. The choice is determined by numerical performance, not by the size of residuals. Our algorithm permits the use of the SQP Gauss–Newton approximation even for large merit function value problems if its performance is better. This will allow us to choose the SQP Gauss–Newton approximation as often as possible;
2. The orthogonal cyclic reduction process is used for variable reduction. This procedure avoids the hard task of explicitly adjoining extra initial or boundary conditions to take up the intrinsic degrees of freedom in the solution set of the differential equations in order to guarantee that the estimation problem is stably posed;
3. The trust region global convergence strategy is used. This makes our algorithm more robust.

An outline of the algorithm is given below (for more details see Section 4).

ALGORITHM 2.1 *General description of the new algorithm* (PESOL)

Step 0. Set $z_0 \in \Re^{Nn+p}$, $B_0 = B^{\mathrm{GN}}(z_0) \in R^{(Nn+p)\times(Nn+p)}$, $\boldsymbol{\lambda}_0^v = 0 \in \Re^{(N-1)n}$, $\rho_0 \geqslant 0$, $\delta_0 > 0$, $0 < \alpha_1 \leqslant \alpha_2 \leqslant 1$, $\alpha_3 > 1$, $0 < \eta_1 \leqslant \eta_2 \leqslant 1$, $\epsilon > 0$, $k = 0$;

Step 1. If $\| \nabla l(z_k, \boldsymbol{\lambda}_k^v) \|_2 = \| [\nabla l_z(z_k, \boldsymbol{\lambda}_k^v)^{\mathrm{T}}, \boldsymbol{c}(z_k)^{\mathrm{T}}]^{\mathrm{T}} \|_2 \leqslant \epsilon$, then stop; otherwise,

Step 2. Use the orthogonal cyclic reduction procedure to reduce the $(Nn+p)$-dimensional optimization

problem with $\boldsymbol{s} = [\mathbf{d}\boldsymbol{x}_1^{\mathrm{T}}, \ldots, \mathbf{d}\boldsymbol{x}_N^{\mathrm{T}}, \mathbf{d}\boldsymbol{\theta}^{\mathrm{T}}]^{\mathrm{T}}$

$$\min_{\boldsymbol{s}\in\Re^{Nn+p}} \boldsymbol{g}(\boldsymbol{z}_k)^{\mathrm{T}}\boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^{\mathrm{T}} B_k \boldsymbol{s} \tag{2.44}$$

$$\text{s.t.} \quad \boldsymbol{c}_d(\boldsymbol{z}_k) + A_d(\boldsymbol{z}_k)\boldsymbol{s} = 0, \tag{2.45}$$

$$\boldsymbol{c}_p(\boldsymbol{z}_k) + \nabla \boldsymbol{c}_p(\boldsymbol{z}_k)\boldsymbol{s} = 0, \tag{2.46}$$

$$\| \boldsymbol{s} \|_2 \leqslant \delta_k \tag{2.47}$$

to the following $(2n + p)$-dimensional optimization problem $\tilde{\boldsymbol{s}} = [\mathbf{d}\boldsymbol{x}_1^{\mathrm{T}}, \mathbf{d}\boldsymbol{x}_N^{\mathrm{T}}, \mathbf{d}\boldsymbol{\theta}^{\mathrm{T}}]^{\mathrm{T}}$

$$\min_{\tilde{\boldsymbol{s}}\in\Re^{2n+p}} \tilde{\boldsymbol{g}}_k^{\mathrm{T}}\tilde{\boldsymbol{s}} + \frac{1}{2}\tilde{\boldsymbol{s}}^{\mathrm{T}}\tilde{B}_k\tilde{\boldsymbol{s}} \tag{2.48}$$

$$\text{s.t.} \quad \tilde{\boldsymbol{c}}_k + \tilde{A}_k\tilde{\boldsymbol{s}} = 0, \quad \tilde{\boldsymbol{c}}_k \in \Re^{n+q}, \tag{2.49}$$

$$\| \tilde{\boldsymbol{s}} \|_2 \leqslant \tilde{\delta}_k \tag{2.50}$$

so that the number of unknown variables and the number of constraints are independent of N, see Section 3.

Step 3. Use the Byrd and Omojokun algorithm (Lalee *et al.*, 1998) to compute an approximation solution $\tilde{\boldsymbol{s}}_k$ of problem (2.48)–(2.50), see Section 4.1.

Step 4. Use the interpolation formulae from the cycle orthogonal reduction procedure to recover the total approximate solution $\boldsymbol{s}_k$ through $\tilde{\boldsymbol{s}}_k$, see Section 4.4.

Step 5. Update the penalty parameter ρ_k, see Section 4.2.

Step 6. Test the step and the choice of the Hessian approximation B_k, see Section 4.3.

Step 7. Update $\boldsymbol{\lambda}_k^v$ to give $\boldsymbol{\lambda}_{k+1}^v$, see Section 4.4.

Step 8. Update B_{k+1} by computing $B^{\mathrm{GN}}(\boldsymbol{z}_{k+1})$, and $B^{\mathrm{Ne}}(\boldsymbol{z}_{k+1})$ if necessary according to the model switching strategy chosen.

Step 9. Modify δ_k, see Section 4.2.

Step 10. Set $k := k + 1$ and go to Step 1.

We will show in the rest of this paper that this algorithm can be converted into efficient software for parameter estimation of the ODE. Since the key to good performance lies in the orthogonal cyclic reduction procedure applied to the subproblem (2.44)–(2.47), we begin by studying the orthogonal cyclic reduction procedure for an almost block-bidiagonal system.

3. Orthogonal cyclic reduction

We wish to develop efficient and robust methods for reducing the following almost block-bidiagonal system to a system with unknown parameters of $\mathbf{d}\boldsymbol{x}_1$, $\mathbf{d}\boldsymbol{x}_N$ and $\mathbf{d}\boldsymbol{\theta}$

$$\begin{bmatrix} D_1 & C_1 & & & & & E_1 \\ & D_2 & C_2 & & & & E_2 \\ & & \ddots & \ddots & & & \vdots \\ & & & D_{N-2} & C_{N-2} & & E_{N-2} \\ & & & & D_{N-1} & C_{N-1} & E_{N-1} \end{bmatrix} \begin{bmatrix} \mathbf{d}\boldsymbol{x}_1 \\ \vdots \\ \mathbf{d}\boldsymbol{x}_N \\ \mathbf{d}\boldsymbol{\theta} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{c}_1 \\ \vdots \\ -\boldsymbol{c}_{N-1} \end{bmatrix}, \tag{3.1}$$

where $D_i \in \Re^{n\times n}$, $C_i \in \Re^{n\times n}$ and $E_i \in \Re^{n\times p}$, $\mathbf{d}\boldsymbol{x}_i$ and $\boldsymbol{c}_i$ are n-dimensional vectors.

Note that (3.1) can be rewritten as

$$D_i\mathbf{d}\boldsymbol{x}_i + C_i\mathbf{d}\boldsymbol{x}_{i+1} + E_i\mathbf{d}\boldsymbol{\theta} = -\boldsymbol{c}_i \quad i = 1, 2, \ldots, N-1. \tag{3.2}$$

Properties of cyclic reduction are discussed in Osborne (1997) and Wright (1992).

It simplifies addressing in developing the cyclic reduction procedure to assume that $N = 2^l + 1$, but it should be noted that this is not a necessary assumption, orthogonal wrap-around partitioning (Hegland & Osborne, 1998) does not depend on exact factorization of the order of the current submatrix. It has the advantage that stable factorization methods which preserve the underlying structure can be used and provides a profound generalization of cyclic reduction. Here, we also assume there are boundary point constraints only. If intermediate point constraints are present, we partition the total interval into sub-intervals so that the intermediate points become boundary points and each partition is processed independently to eliminate the intermediate state variables. Consider the frontal matrix obtained by displaying the data corresponding to consecutive rows in the matrix representation of (3.2) in the case i even. This gives the matrix

$$\begin{bmatrix} D_{i-1}^{(0)} & C_{i-1}^{(0)} & 0 & E_{i-1}^{(0)} & -\boldsymbol{c}_{i-1}^{(0)} \\ 0 & D_i^{(0)} & C_i^{(0)} & E_i^{(0)} & -\boldsymbol{c}_i^{(0)} \end{bmatrix}, \tag{3.3}$$

where $D_i^{(0)} := D_i$, $C_i^{(0)} := C_i$, $E_i^{(0)} := E_i$ and $\boldsymbol{c}_i^{(0)} := \boldsymbol{c}_i$.

Now we are ready to describe the orthogonal cyclic reduction process. By applying orthogonal transformations to the system (3.3), we have

$$(Q_i^{(0)})^{\mathrm{T}} \begin{bmatrix} D_{i-1}^{(0)} & C_{i-1}^{(0)} & 0 & E_{i-1}^{(0)} & -\boldsymbol{c}_{i-1}^{(0)} \\ 0 & D_i^{(0)} & C_i^{(0)} & E_i^{(0)} & -\boldsymbol{c}_i^{(0)} \end{bmatrix}$$

$$\Downarrow$$

$$\begin{bmatrix} V_i^{(1)} & -I & U_i^{(1)} & W_i^{(1)} & \boldsymbol{u}_i^{(1)} \\ D_{i/2}^{(1)} & 0 & C_{i/2}^{(1)} & E_i^{(1)} & -\boldsymbol{c}_{i/2}^{(1)} \end{bmatrix}, \tag{3.4}$$

where $Q_i^{(0)}$ is formed as a product of a series of Householder transformations, and the information needed to reconstruct $Q_i^{(0)}$ can be stored in the space formerly occupied by the zeroed elements of $\begin{bmatrix} C_{i-1}^{(0)} \\ D_i^{(0)} \end{bmatrix}$.

The last row of (3.4) combines entries two time steps apart, i.e.

$$D_{i/2}^{(1)}\mathbf{d}\boldsymbol{x}_{i-1} + C_{i/2}^{(1)}\mathbf{d}\boldsymbol{x}_{i+1} + E_{i/2}^{(1)}\mathbf{d}\boldsymbol{\theta} = -\boldsymbol{c}_{i/2}^{(1)} \tag{3.5}$$

and the first yields

$$\mathbf{d}\boldsymbol{x}_i = V_i^{(1)}\mathbf{d}\boldsymbol{x}_{i-1} + U_i^{(1)}\mathbf{d}\boldsymbol{x}_{i+1} + W_i^{(1)}\mathbf{d}\boldsymbol{\theta} + \boldsymbol{u}_i^{(1)} \tag{3.6}$$

which expresses how the extreme solution values $\mathbf{d}\boldsymbol{x}_{i-1}$, $\mathbf{d}\boldsymbol{x}_{i+1}$ and $\mathbf{d}\boldsymbol{\theta}$ are to be interpolated to give the intermediate value $\mathbf{d}\boldsymbol{x}_i$. This transformation can be applied recursively l times as a consequence of the assumption of $N = 2^l + 1$. The result gives the constraint equation

$$D_1^{(l)}\mathbf{d}\boldsymbol{x}_0 + C_1^{(l)}\mathbf{d}\boldsymbol{x}_N + E_1^{(l)}\mathbf{d}\boldsymbol{\theta} = -\boldsymbol{c}_1^{(l)}, \tag{3.7}$$

i.e.

$$[D_1^{(l)}, C_1^{(l)}, E_1^{(l)}] \begin{bmatrix} \mathbf{d}\boldsymbol{x}_1 \\ \mathbf{d}\boldsymbol{x}_N \\ \mathbf{d}\boldsymbol{\theta} \end{bmatrix} = -\boldsymbol{c}_1^{(l)},$$

which yields (2.49) if we add the linearization of (2.4) with respect to parameters $(\boldsymbol{x}_1, \boldsymbol{x}_N, \boldsymbol{\theta})$

$$\boldsymbol{c}_p(z_k) + \nabla\tilde{\boldsymbol{c}}_p(z_k)\tilde{\boldsymbol{s}} = 0 \tag{3.8}$$

and set

$$\tilde{\boldsymbol{c}}_k = \begin{bmatrix} \boldsymbol{c}^{(l)} \\ \boldsymbol{c}_p \end{bmatrix}, \quad \tilde{A}(z_k) = \begin{bmatrix} D_1^{(l)}, C_1^{(l)}, E_1^{(l)} \\ \nabla\tilde{\boldsymbol{c}}_p \end{bmatrix} \quad \text{and} \quad \tilde{\boldsymbol{s}} = \begin{bmatrix} \mathbf{d}\boldsymbol{x}_1 \\ \mathbf{d}\boldsymbol{x}_N \\ \mathbf{d}\boldsymbol{\theta} \end{bmatrix}.$$

If equations (3.6) determining the eliminated values are updated simultaneously then the result is the interpolation equations

$$\mathbf{d}\boldsymbol{x}_i = V_i\mathbf{d}\boldsymbol{x}_1 + U_i\mathbf{d}\boldsymbol{x}_N + W_i\mathbf{d}\boldsymbol{\theta} + \mathbf{u}_i, \quad i = 2, \ldots, N-1. \tag{3.9}$$

4. The implementational details of Algorithm PESOL

Part of this section appeared in Li *et al.* (2002). For easy reference, we restate it here.

4.1 *Solution of the SQP problem (2.48)–(2.50)*

We are now in a position to discuss how to get an approximate solution of the problem (2.48)–(2.50).

This section is based on Lalee *et al.* (1998) which gives a careful discussion of the implementation of the ideas due to Byrd and Omojokun. The key to this approach is to decouple it into two independent smaller subproblems with constraints in the normal and tangent space of the constraints, respectively. Note that restricting the size of the step by $\| \tilde{\boldsymbol{s}} \|_2 \leqslant \tilde{\delta}_k$ may preclude us from satisfying the linear constraints $\tilde{\boldsymbol{c}}_k + \tilde{A}(z_k)\tilde{\boldsymbol{s}}_k = 0$. To compromise, let $\zeta \in (0, 1)$ be a relaxation factor, and consider a vertical (normal) step toward constraint satisfaction defined by

$$\min_{\boldsymbol{v}\in\Re^{2n+p}} \| \tilde{A}_k\boldsymbol{v} + \tilde{\boldsymbol{c}}_k \|_2 \tag{4.1}$$

$$\text{s.t.} \quad \| \boldsymbol{v} \|_2 \leqslant \zeta\tilde{\delta}_k. \tag{4.2}$$

The uncoupling idea is employed by setting $\boldsymbol{v} = \tilde{A}(z_k)^{\mathrm{T}}\boldsymbol{w}$ which leads to a simple constrained least-squares problem for $\boldsymbol{w}$. Now, to reduce the function value, consider the variant given by

$$\min_{\tilde{\boldsymbol{s}}\in\Re^{2n+p}} \tilde{\boldsymbol{s}}^{\mathrm{T}}\tilde{\boldsymbol{g}}_k + \frac{1}{2}\tilde{\boldsymbol{s}}^{\mathrm{T}}\tilde{B}_k\tilde{\boldsymbol{s}} \tag{4.3}$$

$$\text{s.t.} \quad \tilde{A}_k\tilde{\boldsymbol{s}} = \tilde{A}_k\boldsymbol{v}_k, \tag{4.4}$$

$$\| \tilde{\boldsymbol{s}} \|_2 \leqslant \tilde{\delta}_k. \tag{4.5}$$

This problem has a non-empty feasible region because it always contains $\boldsymbol{v}$. The Byrd and Omojokun approach is modified so that the full step $\tilde{\boldsymbol{s}}_k$ need not move any closer to the feasible manifold than $\boldsymbol{v}_k$ does. We solve for $\tilde{\boldsymbol{s}}_k$ by seeking a step complementary to $\boldsymbol{v}_k$. To this end, we compute a matrix Z_k whose columns form the orthogonal basis for the null space of $\tilde{A}_k$ such that $\tilde{A}_k Z_k = 0$ and $Z_k^{\mathrm{T}} Z_k = I$, and we define the total step of the algorithm as

$$\tilde{\boldsymbol{s}}_k = \boldsymbol{v}_k + Z_k\boldsymbol{u}_k, \tag{4.6}$$

where $\boldsymbol{u}_k$ is yet to be determined. With this choice, the linear constraints (4.4) are automatically satisfied so for the new problem, after dropping constant terms, we know that $\boldsymbol{u}_k$ solves the following problem:

$$\min_{\boldsymbol{u}\in\Re^{n+p-q}} h(\boldsymbol{u}) = \boldsymbol{u}^{\mathrm{T}}[Z^{\mathrm{T}}(\tilde{\boldsymbol{g}}_k + \tilde{B}_k\boldsymbol{v}_k)] + \frac{1}{2}\boldsymbol{u}^{\mathrm{T}} Z_k^{\mathrm{T}}\tilde{B}_k Z_k\boldsymbol{u} \tag{4.7}$$

$$\text{s.t.} \quad \| \boldsymbol{u} \|_2 \leqslant \sqrt{\tilde{\delta}_k^2 - \| \boldsymbol{v}_k \|_2^2}. \tag{4.8}$$

If we define $\hat{\boldsymbol{g}}_k = Z_k^{\mathrm{T}}(\tilde{\boldsymbol{g}}_k + \tilde{B}_k\boldsymbol{v}_k)$, $\hat{B}_k = Z_k^{\mathrm{T}}\tilde{B}_k Z_k$ and $\hat{\delta}_k = \sqrt{\tilde{\delta}_k^2 - \| \boldsymbol{v}_k \|_2^2}$, then we get the following equivalent system:

$$\min_{\boldsymbol{u}\in\Re^{n+p-q}} \boldsymbol{u}^{\mathrm{T}}\hat{\boldsymbol{g}}_k + \frac{1}{2}\boldsymbol{u}^{\mathrm{T}}\hat{B}_k\boldsymbol{u} \tag{4.9}$$

$$\text{s.t.} \quad \| \boldsymbol{u} \|_2 \leqslant \hat{\delta}_k. \tag{4.10}$$

Note that this has the same form as a trust region step in an unconstrained algorithm.

Powell's dogleg method (Powell, 1970) is used for approximately solving the trust region problem (4.7)–(4.8). Because this method requires the matrix $\hat{B}_k$ to be positive definite, we use the modified Cholesky factorization of Gill and Murray in Dennis & Schnabel (1986). The idea is to change $\hat{B}_k$ to $\hat{B}_k + \mu_k I$, where $\mu_k > 0$ is not much larger, ideally, than the smallest μ that will make $\hat{B}_k + \mu_k I$ positive definite and reasonably well conditioned. For simplicity of notation, we retain $\hat{B}_k$ to denote the resulting matrix from this procedure.

Dogleg method First calculate the Cauchy step

$$\boldsymbol{u}_k^{cp} = -\alpha_k^u\hat{\boldsymbol{g}}_k, \tag{4.11}$$

which is the minimizer of $h(\boldsymbol{u})$ in the direction of steepest descent at $\boldsymbol{u} = 0$, subject to the trust constraint, where

$$\alpha_k^u = \begin{cases} \hat{\boldsymbol{g}}_k^{\mathrm{T}}\hat{\boldsymbol{g}}_k / \hat{\boldsymbol{g}}_k^{\mathrm{T}}\hat{B}_k\hat{\boldsymbol{g}}_k & \text{if } (\hat{\boldsymbol{g}}^{\mathrm{T}}\hat{\boldsymbol{g}}_k)^{3/2} / \hat{\boldsymbol{g}}_k^{\mathrm{T}}\hat{B}_k\hat{\boldsymbol{g}}_k \leqslant \hat{\delta}_k; \\ \hat{\delta}_k / \| \hat{\boldsymbol{g}}_k \|_2 & \text{otherwise} \end{cases} \tag{4.12}$$

and the Newton step

$$\boldsymbol{u}_k^n = -\hat{B}_k^{-1}\hat{\boldsymbol{g}}_k. \tag{4.13}$$

The dogleg path consists of the two segments from $\boldsymbol{u} = 0$ to $\boldsymbol{u} = \boldsymbol{u}_k^{cp}$ and from $\boldsymbol{u} = \boldsymbol{u}_k^{cp}$ to $\boldsymbol{u} = \boldsymbol{u}_k^n$. The dogleg method finds the minimizer along this path subject to $\| \boldsymbol{u} \| \leqslant \hat{\delta}_k$. 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.

4.2 *Trust region method*

In order to test the viability of our algorithm in a trust region globalization framework, we need to specify a merit function to decide whether a step $\boldsymbol{s}_k$ makes sufficient progress toward the solution of the problem (2.6)–(2.7). We choose a merit function of the form

$$\phi(z, \rho) = m(z) + \rho \left\| \begin{bmatrix} \boldsymbol{c}_d(z) \\ \boldsymbol{c}_p(z) \end{bmatrix} \right\|_2. \tag{4.14}$$

This type of merit function is used in Lalee *et al.* (1998) and Li *et al.* (2002). Therefore, the actual reduction $\text{ared}(z_k, \rho_k)$ in the merit function in the step from z_k to $z_k + \boldsymbol{s}_k$ is given by

$$\begin{aligned} \text{ared}(z_k, \rho_k) &= \phi(z_k, \rho_k) - \phi(z_k + \boldsymbol{s}_k, \rho_k) \\ &= m(z_k) - m(z_k + \boldsymbol{s}_k) \\ &\quad + \rho_k \left(\left\| \begin{bmatrix} \boldsymbol{c}_d(z_k) \\ \boldsymbol{c}_p(z_k) \end{bmatrix} \right\|_2 - \left\| \begin{bmatrix} \boldsymbol{c}_d(z_k + \boldsymbol{s}_k) \\ \boldsymbol{c}_p(z_k + \boldsymbol{s}_k) \end{bmatrix} \right\|_2 \right). \end{aligned} \tag{4.15}$$

Also we need a prediction function to predict the progress of the merit function. We choose this as

$$\psi(z_k, \rho_k) = m(z_k) + \boldsymbol{s}_k^{\mathrm{T}} \nabla m(z_k) + \frac{1}{2} \boldsymbol{s}_k^{\mathrm{T}} B_k \boldsymbol{s} + \rho_k \left\| \begin{bmatrix} \boldsymbol{c}_d(z_k) + A_d(z_k)\boldsymbol{s}_k \\ \boldsymbol{c}_p(z_k) + \nabla \boldsymbol{c}_p(z_k)\boldsymbol{s}_k \end{bmatrix} \right\|_2 \tag{4.16}$$

and define the predicted reduction $\text{pred}(z_k, \rho_k)$ by

$$\begin{aligned} \text{pred}(z_k, \rho_k) = -\boldsymbol{s}_k^{\mathrm{T}} \nabla m(z_k) - \frac{1}{2} \boldsymbol{s}_k^{\mathrm{T}} B_k \boldsymbol{s}_k + \rho_k \left(\left\| \begin{bmatrix} \boldsymbol{c}_d(z_k) \\ \boldsymbol{c}_p(z_k) \end{bmatrix} \right\|_2 \right. \\ \left. - \left\| \begin{bmatrix} \boldsymbol{c}_d(z_k) + A_d(z_k)\boldsymbol{s}_k \\ \boldsymbol{c}_p(z_k) + \nabla \boldsymbol{c}_p(z_k)\boldsymbol{s}_k \end{bmatrix} \right\|_2 \right). \end{aligned} \tag{4.17}$$

Numerical experiments have suggested that efficient performance of the algorithm is linked to keeping the penalty parameter as small as possible. However, global convergence theory (El-Alem, 1995) requires that the sequence ρ_k be non-decreasing, and that the predicted reduction in the merit function at each iteration be at least a fraction of the Cauchy decrease of the residual norm of the linearized constraints. The idea now is to keep the penalty parameter as small as possible, while satisfying the two conditions needed for convergence. Hence, our strategy will be to start with $\rho_0 = 1$ and increase it only when necessary in order to satisfy these two conditions. This aim can be achieved by the following scheme:

(a) If $\mathrm{pred}(z_k, \rho_k) \geqslant \xi \rho_k \left(\left\| \begin{bmatrix} c_d(z_k) \\ c_p(z_k) \end{bmatrix} \right\|_2 - \left\| \begin{bmatrix} c_d(z_k)+A_d(z_k)s_k \\ c_p(z_k)+\nabla c_p(z_k)s_k \end{bmatrix} \right\|_2 \right)$, where $\xi \in (0, 1)$, then we set

$$\rho_{k+1} := \rho_k; \tag{4.18}$$

(b) else, we set

$$\rho_{k+1} := \frac{1}{1-\xi} \cdot \frac{s_k^{\mathrm{T}} \nabla m(z_k) + \frac{1}{2} s_k^{\mathrm{T}} B_k s_k}{\left\| \begin{bmatrix} c_d(z_k) \\ c_p(z_k) \end{bmatrix} \right\|_2 - \left\| \begin{bmatrix} c_d(z_k) + A_d(z_k)s_k \\ c_p(z_k) + \nabla c_p(z_k)s_k \end{bmatrix} \right\|_2} + 0{\cdot}0001, \tag{4.19}$$

where adding 0·0001 is to make sure that ρ_{k+1} is not too small; choice of other small values does not appear critical, the value of 0·0001 is typically used.

Once z_{k+1} has been found, we decide which trust region radius to use first when seeking z_{k+2}. The radius chosen is as follows:

(a) if

$$\frac{\mathrm{ared}(z_{k+1}^p, \rho_{k+1})}{\mathrm{pred}(z_{k+1}^p, \rho_{k+1})} \geqslant 0{\cdot}75, \tag{4.20}$$

we set $\delta_{k+1} = \min(2\delta_k, \delta_*)$, where δ_* is the maximum step length allowed in the algorithm;

(b) else if

$$\frac{\mathrm{ared}(z_{k+1}^p, \rho_{k+1})}{\mathrm{pred}(z_{k+1}^p, \rho_{k+1})} < 0{\cdot}1, \tag{4.21}$$

we set δ_{k+1} to be a fraction of the failed step length such that

$$\delta_{k+1} \in [0{\cdot}1 \parallel s_k \parallel_2, 0{\cdot}5 \parallel s_k \parallel_2]. \tag{4.22}$$

The precise value is computed by assuming that the ratio of actual to predicted reduction is a linear function $w(\parallel s \parallel_2)$ of the step length $\parallel s \parallel_2$, satisfying $w(0) = 1$ and $w(\parallel s_k \parallel_2) = \mathrm{ared}(z_{k+1}, \rho_{k+1}) / \mathrm{pred}(z_{k+1}, \rho_{k+1})$, and then finding the value of $\parallel s \parallel_2$ where $w(\parallel s \parallel)$ equals η_1 by the following formula:

$$\delta_{k+1} = \frac{1-\eta_1}{1 - (\mathrm{ared}(z_{k+1}, \rho_{k+1}) / \mathrm{pred}(z_{k+1}, \rho_{k+1}))} \parallel s_k \parallel_2; \tag{4.23}$$

this value is truncated such that $\delta_{k+1} \in [0{\cdot}1 \parallel s_k \parallel_2, 0{\cdot}5 \parallel s_k \parallel_2]$;

(c) else, we keep the same radius, i.e. set

$$\delta_{k+1} = \min(2\delta_k, \delta_*). \tag{4.24}$$

4.3 *Model switching strategy*

Our experiments show that for some steps the SQP Gauss–Newton approximation works better than the SQP Newton approximation, so it seems useful to have some way to decide which approximation to use

at each iteration. Following the ideas in Dennis *et al.* (1986), a model switching strategy is developed for this purpose. Our implementation includes these two approximations. Because the SQP Gauss–Newton approximation tends to do well initially, we start with this. The first trial step is calculated using the currently preferred algorithm whose predicted reduction is better in the last iteration.

Note that a valuable feature in our code is the internal doubling step. For a given z_k and δ_k, suppose s_{δ_k} is generated such that δ_k restricts s_{δ_k} and the reduction in the merit function $\phi(z_k, \rho_k)$ predicted by the current algorithm agrees with the actual reduction in the merit function $\phi(z_k, \rho_k)$ to a high precision. Normally, one would accept s_{δ_k}, set $z_{k+1} = z_k + s_{\delta_k}$ and double δ_k for the next step. The internal doubling procedure is to remember $z^p_{k+1} = z_k + s_{\delta_k}$, double δ_k and generate a new s_{δ_k} from z_k. Note that this procedure only costs evaluations of $m(z_k)$ and $c(z_k)$. If successful, it may save several evaluations of $\nabla m(z_k)$ and $A(z_k)$. In practice, it has been successful often enough to warrant leaving it in. For convenient presentation of our switching strategy, we use $q(z_k, B_k)$ to denote the currently preferred quadratic function (2.44) and $q^a(z_k, B^a_k)$ for the alternate quadratic function with replacement of B_k by B^a_k, and we use $\psi(z_k, \rho_k)$ and $\psi^a(z_k, \rho_k)$ for their corresponding predicting functions, respectively.

We begin our current iteration by computing a prospective z_{k+1}, say z^p_{k+1}, based on $q(z_k, B_k)$ and the current radius. We compute $m(z^p_{k+1})$ and $c(z^p_{k+1})$, but we do not yet compute $\nabla m(z^p_{k+1})$, $A(z^p_{k+1})$ and λ_{k+1}; our only gradient calculation in this iteration is $\nabla m(z_{k+1})$ and $A(z_{k+1})$. We compute the approximate multiplier λ_{k+1} only if we have found z_{k+1}. If

$$\frac{\text{ared}(z^p_{k+1}, \rho_{k+1})}{\text{pred}(z^p_{k+1}, \rho_{k+1})} \geqslant \eta_1, \quad \eta_1 \in (0, 1), \tag{4.25}$$

then the step is a good one. Furthermore, if

$$\frac{\text{ared}(z^p_{k+1}, \rho_{k+1})}{\text{pred}(z^p_{k+1}, \rho_{k+1})} \geqslant \eta_2, \quad \eta_2 \in (\eta_1, 1), \tag{4.26}$$

this means that the direction appears worth pursuing. Then we save z^p_{k+1} and $m(z^p_{k+1})$ and double the trust region radius $\delta_k := \min(2\delta_k, \delta_*)$, where δ_* is the maximum trust region radius. This strategy has the advantage that it avoids recomputation of $\nabla m(z_k)$, $A(z_k)$ and λ_{k+1} when this is expensive. We compute $z^{p'}_{k+1}$ on the basis of $q(z_k, B_k)$ and the increased trust radius. If $\phi(z^{p'}_{k+1}, \rho_k) \geqslant \phi(z^p_{k+1}, \rho)$, then we accept z^p_{k+1} as z_{k+1} and prepare for the next iteration. If $\phi(z^{p'}_{k+1}, \rho_k) < \phi(z^p_{k+1}, \rho_k)$, then we replace z^p_{k+1} by $z^{p'}_{k+1}$ and return to test (4.25). If ever (4.25) is true but (4.26) is false, then z^p_{k+1} is accepted as z_{k+1} and we prepare for the next iteration.

Now let us trace the branch that originates when (4.25) is false. In this case, we do not regard z^p_{k+1} very highly as a candidate for z_{k+1}, but its fate will be decided by further tests. We first test whether it might be useful to try changing models, but only if this is the first time through (4.25) in the current iteration. If

$$\frac{|\ \psi(z^p_{k+1}, \rho_{k+1}) - \phi(z^p_{k+1}, \rho_{k+1})\ |}{|\ \psi^a(z^p_{k+1}, \rho_{k+1}) - \phi(z^p_{k+1}, \rho_{k+1})\ |} > 1{\cdot}5, \tag{4.27}$$

then we change our algorithm preference in the sense that we compute x^a_{k+1} with the same trust radius and penalty parameter ρ_k. If $\phi(z^a_{k+1}, \rho_{k+1}) < \phi(z^p_{k+1}, \rho_{k+1})$, then we change our algorithm preference, so z^a_{k+1} becomes z^p_{k+1} and we return to test (4.25); otherwise we retain our current algorithm preference and decrease the radius of the trust region.

If we reach this point without having decided on z_{k+1}, then we have a poorly proposed new iterate x_{k+1}^p and we have rejected the notion of switching models. If

$$\frac{\text{ared}(z_{k+1}^p, \rho_{k+1})}{\text{pred}(z_{k+1}^p, \rho_{k+1})} < 10^{-4}, \tag{4.28}$$

then we reject z_{k+1}^p and shrink the trust region radius by picking

$$\delta_k \in [\alpha_1 \parallel s_k \parallel_2, \alpha_2 \parallel s_k \parallel_2], \tag{4.29}$$

where $0 < \alpha_1 < \alpha_2 < 1$. We then recompute z_{k+1}^p and return to test (4.25). If (4.28) is false, then we accept z_{k+1}^p as z_{k+1}.

Our numerical testing suggests that the following values give good performances: $\eta_1 = 0{\cdot}1$, $\eta_2 = 0{\cdot}75$, $\alpha_1 = \alpha_2 = 0{\cdot}5$, $\alpha_3 = 2{\cdot}0$, $\delta_0 = 1$ and $\delta_* = 100$.

After we have found an acceptable z_{k+1}, we decide whether to change algorithm preferences for computing z_{k+2}. We have found that it is best to retain the currently preferred algorithm if (4.27) holds with $z_{k+1}^p = z_{k+1}$ and $\phi(z_{k+1}^a, \rho_{k+1}) > \phi(z_{k+1}, \rho_{k+1})$ unless the other algorithm does a significantly better job of predicting the new function value.

4.4 *Summary of the algorithm implementation*

Now we are ready to describe the complete form of our adaptive algorithm PESOL

Algorithm PESOL

Step 0. Set $z_0 \in \Re^{Nn+p}$, $B_0 = B^{\text{GN}}(z_0) \in R^{(Nn+p)\times(Nn+p)}$, $\lambda_0^v = 0 \in \Re^{(N-1)n}$, $\rho_0 \geqslant 0$, $\delta_0 > 0$, $0 < \alpha_1 \leqslant \alpha_2 \leqslant 1$, $0 < \eta_1 \leqslant \eta_2 \leqslant 1$, $\epsilon > 0$, $k = 0$.

Step 1. If $\parallel \nabla l(z_k, \lambda_k^v) \parallel_2 = \parallel [\nabla l_z(z_k, \lambda_k^v)^{\text{T}}, c(z_k)^{\text{T}}]^{\text{T}} \parallel_2 \leqslant \epsilon$, then stop; otherwise,

Step 2. Use the cyclic orthogonal reduction procedure to reduce the $(Nn+p)$-dimensional optimization problem

$$\min_{s\in\Re^{Nn+p}} g(z_k)^{\text{T}} s + \frac{1}{2} s^{\text{T}} B_k s \tag{4.30}$$

$$\text{s.t.} \quad c_d(z_k) + A_d(z_k) s = 0, \tag{4.31}$$

$$c_p(z) + \nabla c_p(z) s = 0, \tag{4.32}$$

$$\parallel s \parallel_2 \leqslant \delta_k \tag{4.33}$$

to the following $(2n + p)$-dimensional optimization problem:

$$\min_{\tilde{s}\in\Re^{2n+p}} \tilde{g}_k^{\text{T}} \tilde{s} + \frac{1}{2} \tilde{s}^{\text{T}} \tilde{B}_k \tilde{s} \tag{4.34}$$

$$\text{s.t.} \quad \tilde{c}_k + \tilde{A}_k \tilde{s} = 0, \quad \tilde{c}_k \in \Re^{n+q}, \tag{4.35}$$

$$\parallel \tilde{s} \parallel_2 \leqslant \tilde{\delta}_k \tag{4.36}$$

so that the number of unknown variables and the number of constraints are independent of N.

Step 3. Use the Byrd and Omojokun algorithm (Lalee *et al.*, 1998) to compute an approximation solution $\tilde{s}_k$ of problem (4.34)–(4.36).

Step 4. Use the interpolation formulae from the cyclic orthogonal reduction procedure to recover the total approximate solution s_k through $\tilde{s}_k$ by

$$\mathbf{d}x_i = V_i \mathbf{d}x_1 + U_i \mathbf{d}x_N + W_i \mathbf{d}\theta + \mathbf{u}_i, \quad i = 2, \ldots, N-1. \tag{4.37}$$

Step 5. Update the penalty parameter ρ_k through the formulae (4.18) and (4.19).

Step 6. Test the step and the choice of the Hessian approximation B_k, see Section 4.3.

Step 7. Update λ_k^v to give λ_{k+1}^v by

$$\lambda_{k+1}^v = -[A(z_k)A(z_k)^{\mathrm{T}}]^{-1} A(z_k)[\nabla m(z_k) + B_k s_k]. \tag{4.38}$$

Step 8. Update B_{k+1} by computing $B^{\mathrm{GN}}(z_{k+1})$, and $B^{\mathrm{Ne}}(z_{k+1})$ if necessary according to the model switching strategy chosen.

Step 9. Modify δ_k through the formulae (4.20)–(4.24).

Step 10. Set $k := k + 1$ and go to Step 1.

For the purpose of comparison we also implemented the SQP Gauss–Newton method and the SQP Newton method, and ran it side by side with PESOL. Our implementation of the SQP Gauss–Newton and the SQP Newton method are the following. For simplicity we will refer to them as the $\mathrm{PE}_{\mathrm{GN}}$ and $\mathrm{PE}_{\mathrm{Ne}}$ algorithms respectively.

Algorithm $\mathbf{PE}_{\mathrm{GN}}$: All steps are identical to PESOL except for Step 2, where the SQP Gauss–Newton approximation to the Hessian of the Lagrange is used.

Algorithm $\mathbf{PE}_{\mathrm{Ne}}$: All steps are identical to PESOL except for Step 2, where the full Hessian of the Lagrangian is used.

5. Numerical results

In this section, we present two well-known parameter estimation problems which have been devised to illustrate difficulties in the initial-value approach, see Bock (1983) and Ascher *et al.* (1995). The objective here is to show the stability and efficiency properties of our method. For the purpose of comparison, the results of the SQP Gauss–Newton method $\mathrm{PE}_{\mathrm{GN}}$ and the SQP Newton method $\mathrm{PE}_{\mathrm{Ne}}$ are also presented. The SQP Gauss–Newton approximation is the choice of the methods in Bock (1983), Bock & Schlöder (1987), Bock *et al.* (1988) and Childs & Osborne (1996). This choice is preferred to the Newton approximation because it has equally good scaling properties, better global convergence characteristics, and is cheaper to compute. Although the SQP Newton method has the best performance in terms of the number of iterations in our simulations, the idea here is that the SQP Newton method, which had its own problems, is used only when the good features of the SQP Gauss–Newton approximation do not apply.

The algorithms described above have been tested on a Sun Ultra Sparc 5 Workstation in double-precision C with compiler Sun C version 4.2 under Solaris 2.6 operating system. In our experiments, the stopping tolerance ϵ is set to 10^{-6}.

TABLE 1 *Number of iterations on Problem* 5.1

N	$\sigma = 1{\cdot}0$			$\sigma = 0{\cdot}1$			$\sigma = 0{\cdot}01$		
	PE_{Ne}	PE_{GN}	PESOL	PE_{Ne}	PE_{GN}	PESOL	PE_{Ne}	PE_{GN}	PESOL
2^5+1	9	14	10	9	12	8	8	12	9
2^7+1	6	8	8	6	8	4	5	8	6
$2^{10}+1$	5	8	6	4	5	4	4	4	4

PROBLEM 5.1 (A parameter estimation problem with one parameter) This is a parameter estimation problem with two states and one parameter. It is adapted from Bock (1983). We formulate this problem as an initial-value problem.

$$\min_{\theta} \psi(\theta) = \frac{1}{2N} \sum_{i=1}^{N} [\hat{y}_i - x_2(t_i)]^2 \tag{5.1}$$

$$\text{s.t.} \quad \frac{\mathrm{d}x_1}{\mathrm{d}t} = x_2, \tag{5.2}$$

$$\frac{\mathrm{d}x_2}{\mathrm{d}t} = \tau^2 x_1 - (\tau^2 + \theta^2)\sin(\theta t), \tag{5.3}$$

$$t \in [0, 1], \quad [x_1(0), x_2(0)]^{\mathrm{T}} = [0, \pi]^{\mathrm{T}}. \tag{5.4}$$

The fundamental matrix for the system (5.2)–(5.3) is

$$\Phi(t) = \begin{bmatrix} \cosh(\tau t) & \tau^{-1}\sinh(\tau t) \\ \tau \sinh(\tau t) & \cosh(\tau t) \end{bmatrix}. \tag{5.5}$$

It is characterized by rapidly varying fast and slow solutions determined by the terms $\mathrm{e}^{\tau t}$ and $\mathrm{e}^{-\tau t}$. When $\mathrm{e}^{\tau} \times macheps > 1$, then serious numerical problems are expected in the initial-value approach. Here *macheps* is the commonly used concept of *machine epsilon* which is defined as the smallest positive number ϵ such that $1 + \epsilon > 1$ on the computer in question.

The solution for the parameter $\theta = \pi$ is $x_1(t) = \sin(\pi t)$ and $x_2(t) = \pi \cos(\pi t)$. Measurements $\{\hat{y}_i\}_{i=1}^{N}$ are generated by adding random noise $N(0, \sigma^2)$ with standard deviations σ of $1{\cdot}0$, $0{\cdot}1$ and $0{\cdot}01$ to $x_2(t)$ at data points equal to the mesh points. The unknown parameter θ was initialized to 2 and the value of the constant τ was set to 100. The computational results are presented in Table 1.

PROBLEM 5.2 (A parameter estimation problem with two parameters) We next consider the parameter estimation problem, modified from Ascher *et al.* (1995), and also studied by Osborne (1997):

$$\min_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta}) = \frac{1}{2N} \sum_{i=1}^{N} [\hat{\boldsymbol{y}}_i - \boldsymbol{x}(t_i)]^{\mathrm{T}} [\hat{\boldsymbol{y}}_i - \boldsymbol{x}(t_i)] \tag{5.6}$$

$$\text{s.t.} \quad \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = M(t, \boldsymbol{\theta})\boldsymbol{x} + \boldsymbol{f}(t), \quad t \in [0, 1], \tag{5.7}$$

$$\boldsymbol{x}(0) + \boldsymbol{x}(\pi) = (1 + \mathrm{e}^{\pi})[1, 1, 1]^{\mathrm{T}}, \tag{5.8}$$

TABLE 2 *Number of iterations on Problem* 5.2

N	$\sigma = 5{\cdot}0$			$\sigma = 1{\cdot}0$			$\sigma = 0{\cdot}01$		
	PE_{Ne}	PE_{GN}	PESOL	PE_{Ne}	PE_{GN}	PESOL	PE_{Ne}	PE_{GN}	PESOL
2^5+1	15	55	18	6	11	7	4	4	4
2^7+1	16	20	22	6	10	7	3	4	3
$2^{10}+1$	7	13	10	4	5	4	3	3	3

where

$$M(t,\boldsymbol{\theta}) = \begin{bmatrix} \theta_2 - \theta_1\cos(\theta_2 t) & 0 & \theta_2 + \theta_1\sin(\theta_2 t) \\ 0 & \theta_1 & 0 \\ -\theta_2 + \theta_1\sin(\theta_2 t) & 0 & \theta_2 + \theta_1\cos(\theta_2 t) \end{bmatrix} \tag{5.9}$$

and

$$\boldsymbol{f}(t) = \mathrm{e}^t \begin{bmatrix} -1 + 19[\cos(t) - \sin(t)] \\ -18 \\ 1 - 19[\cos(t) + \sin(t)] \end{bmatrix}. \tag{5.10}$$

The fundamental matrix for the system (5.7) is

$$\Phi(t) = \begin{bmatrix} \mathrm{e}^{-(\theta_1-\theta_2)t}\cos(2\theta_2 t) & 0 & \mathrm{e}^{(\theta_1+\theta_2)t}\sin(2\theta_2 t) \\ 0 & \mathrm{e}^{\theta_1 t} & 0 \\ -\mathrm{e}^{-(\theta_1-\theta_2)t}\sin(2\theta_2 t) & 0 & \mathrm{e}^{(\theta_1+\theta_2)t}\cos(2\theta_2 t) \end{bmatrix}. \tag{5.11}$$

It is characterized by rapidly varying fast and slow solutions determined by the terms $\mathrm{e}^{(\theta_1+\theta_2)t}$ and $\mathrm{e}^{-(\theta_1-\theta_2)t}$ if the difference between the two positive parameters θ_1 and θ_2 is large. These can cause numerical problems unless an appropriate solution method is used (Ascher *et al.*, 1995) where numerical data relating to the initial-value problem are given.

The data are generated randomly around the solution $[x_1(t), x_2(t), x_3(t)]^{\mathrm{T}} = [\mathrm{e}^t, \mathrm{e}^t, \mathrm{e}^t]^{\mathrm{T}}$ for the parameters $\theta_1 = 19$ and $\theta_2 = 1$ by the assumption that standard deviations $\sigma = 5{\cdot}0,\ 1{\cdot}0$ and $0{\cdot}01$. The unknown parameters θ_1 and θ_2 are initialized at 20% above their true values 19 and 1, respectively. The computational results are presented in Table 2.

From our numerical experiments, we found that our new method PESOL performs well, it converges for all cases of N. Its performance is close to that of the SQP Newton method in terms of the number of iterations. The orthogonal cyclic reduction technique is stable and efficient. As expected, when N is large enough, for example, $N = 2^{10}+1$, algorithm PESOL never switches to the Newton approximation because the SQP Gauss–Newton approximation works well; thus the steps are essentially the same as those of PE_{GN}. There is no great difference between algorithms PE_{Ne}, PESOL and PE_{GN}. However, when only a limited amount of data is available, for example, $N = 2^5 + 1$, the performance of PESOL is much better than that of the SQP Gauss–Newton algorithm PE_{GN}, and is close to that of PE_{Ne}. Our simulations shows that when the model is correct and the sample data is large enough, the SQP Gauss–Newton method works well. Otherwise, a more powerful method is needed.

REFERENCES

ASCHER, U. M., MATTHEIJ, R. M. M. & RUSSELL, R. D. (1995) *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*. Philadelphia, PA: SIAM.

BAER, M., HEGGER, R. & KANTZ, H. (1999) Fitting partial differential equations to space-time dynamics. *Phys. Rev.* E, **59**, 337–342.

BARD, Y. (1974) *Nonlinear Parameter Estimation*. London: Academic.

BIEGLER, L. T. (1998) Advances in nonlinear programming concepts for process control. *J. Process Control*, **8**, 301–311.

BOCK, H. G. (1983) Recent advances in parameter identification techniques for ODE. *Numerical Treatment of Inverse Problems in Differential and Integral Equations*. (P. Deuflhard & E. Hairer, eds). Basel: Birkhäuser, pp. 95–121.

BOCK, H. G. & SCHLÖDER, J. P. (1987) Recent progress in the development of algorithms and software for large scale parameter estimation problems in chemical reaction systems. *Technical Report* 441, Institute für Angewandte Mathematik, Universitet Heidelberg.

BOCK, H. G., EICH, E. & SCHLÖDER, J. P. (1988) Numerical solution of constrained least squares boundary value problems in differential algebraic equations. *Numerical Treatment of Differential Equations*. (K. Strehmel, ed.). Leipzig: Teubner, pp. 269–280.

CHILDS, S. B. & OSBORNE, M. R. (1995) Fitting solutions of ordinary differential equations to observed data. *Computational Techniques and Applications: CTAC95*. (R. L. May & A. K. Easton, eds). World Scientific, Singapore, pp. 193–198.

DENNIS, J. E. & SCHNABEL, R. B. (1983) *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Englewood Cliffs, NJ: Prentice-Hall.

DENNIS, J. E., GAY, D. M. & WELSCH, R. E. (1981) An adaptive nonlinear least-squares algorithm. *ACM Trans. Math. Software*, **7**, 348–368.

DEUFLHARD, P. & NOWAK, U. (1986) Efficient numerical simulation and identification of large chemical reaction systems. *Ber. Bunsenges. Phys. Chem.*, **90**, 940–946.

EL-ALEM, M. (1995) A robust trust-region algorithm with a nonmonotonic penalty parameter scheme for constrained optimization. *SIAM J. Optim.*, **5**, 348–378.

FLETCHER, R. (1987) *Practical Methods of Optimization* 2nd edn. Chichester: Wiley.

FOSS, S. D. (1971) Estimates of chemical kinetic rate constants by numerical integration. *Chem. Engng Sci.*, **26**, 485–486.

GALLITZENDOERFER, J. V. & BOCK, H. G. (1994) Parallel algorithms for optimization boundary value problems in DAE. *Praxisorientierte Parallelverarbeitung*. (H. Langendoerfer, ed.). Munich: Hanser, pp. 156–189.

HEGLAND, M. & OSBORNE, M. R. (1998) Wrap-around partitioning for block bidiagonal linear systems. *IMA J. Numer. Anal.*, **18**, 373–383.

HEMKER, P. W. (1972) Numerical methods for differential equations in system simulation and in parameter estimation. *Analysis and Simulation of Biochemical Systems*. (H. C. Hemker & B. Hess. eds). pp. 59–80.

IRVING, A. & DEWSON, T. (1997) Determing mixed linear-nonlinear coupled differential equations from multivariate time series sequences. *Physica* D, **102**, 15–36.

LALEE, M., NOCEDAL, J. & PLANTENGA, T. (1998) On the implementation of an algorithm for large-scale equality constrained optimization. *SIAM J. Optim.*, **8**, 682–706.

LI, Z. F. (2000) *Parameter Estimations of Ordinary Differential Equations*. Ph.D. Thesis, Australian National University, Canberra, Australia.

LI, Z. F., OSBORNE, M. R. & PRVAN, T. (2002) Adaptive algorithm for constrained least squares. *J. Optim. Theory Appl.*, **15**, 220–242.

MAHDAVI-AMIRI, N. & BARTES, R. H. (1989) Constrained nonlinear least squares: an exact penalty approach with projected structured quasi-Newton updates. *ACM Trans. Math. Software*, **15**, 220–242.

NOWAK, U. & DEUFLHARD, P. (1985) Numerical identification of selected rate constants in large chemical reaction systems. *Appl. Numer. Math.*, **1**, 59–75.

OSBORNE, M. R. (1997) Cyclic reduction, dichotomy, and the estimation of differential equations. *J. Comput. Appl. Math.*, **86**, 271–286.

OSBORNE, M. R. (2000) Scoring with constraints. *ANZIAM J.*, **42**, 9–25.

PARLITZ, U. & MERKWIRTH, C. (2000) Prediction of spatiotemporal time series based on reconstructed local states. *Phys. Rev. Lett.*, **84**, 1890–1893.

POWELL, M. J. D. (1970) A hybrid method for nonlinear equations. *Numerical Methods for Nonlinear Algebraic Equations*. (P. Rabinowitz, ed.). London: Gordan and Breach, pp. 87–114.

TANARTKIT, P. & BIEGLER, L. T. (1995) Stable decomposition for dynamic optimization. *Ind. Engng Chem. Res.*, **34**, 1253–1266.

TJOA, I. B. & BIEGLER, L. T. (1991) Simultaneous solution and optimization strategies for parameter estimation of differential-algebraic equation systems. *Ind. Engng Chem. Res.*, **30**, 376–385.

WEDIN, P. Ă. & LINDSTRÖM, P. (1987) Methods and Software for Nonlinear Least Squares Problems. *University of Umea Report* No. UMINF 133.87.

WRIGHT, S. (1992) Stable parallel elimination for two point BVPs. *SIAM J. Sci. Stat. Comput.*, **13**, 742–764.