KYBERNETIKA — VOLUME 32 (1996), NUMBER 5, PAGES 465 - 482

OPTIMIZATION OF DYNAMICAL SYSTEMS[1]

LADISLAV LUKŠAN AND JAN VLČEK

Consider an optimization problem where the objective function is an integral containing the solution of a system of ordinary differential equations. Suppose we have efficient optimization methods available as well as efficient methods for initial value problems for ordinary differential equations. The main purpose of this paper is to show how these methods can be efficiently applied to a considered problem. First, the general procedures for the evaluation of gradients and Hessian matrices are described. Furthermore, the new efficient Gauss–Newton–like approximation of the Hessian matrix is derived for the special case when the objective function is an integral of squares. This approximation is used for deriving the Gauss–Newton–like trust region method, with which global and superlinear convergence properties are proved. Finally several optimization methods are proposed and computational experiments illustrating their efficiency are shown.

1. INTRODUCTION

Consider the problem of minimizing the objective function

$$F(x) = \int_{t_0}^{t_1} f_A(y(x,t),t)\,\mathrm{d}t + f_T(y(x,t_1)) \tag{1a}$$

where

$$\frac{\mathrm{d}y(x,t)}{\mathrm{d}t} = f_S(x,y(x,t),t), \qquad y(x,t_0) = f_I(x)\,. \tag{1b}$$

Here $x \in \mathbb{R}^n$ is a parameter vector, $y : \mathbb{R}^n \times [t_0,t_1] \to \mathbb{R}^{n_S}$ is the solution vector, $F : \mathbb{R}^n \to \mathbb{R}$ is the objective function, $f_A : \mathbb{R}^{n_S} \times [t_0,t_1] \to \mathbb{R}$ is an approximation function, $f_T : \mathbb{R}^{n_S} \to \mathbb{R}$ is a terminal function, $f_S : \mathbb{R}^n \times \mathbb{R}^{n_S} \times [t_0,t_1] \to \mathbb{R}^{n_S}$ is a state function, $f_I : \mathbb{R}^n \to \mathbb{R}^{n_S}$ is an initial function. Suppose that all the above functions have continuous second order derivatives on $X \times \mathbb{R}^{n_S} \times [t_0,t_1]$ where $X \subset \mathbb{R}^n$ is a compact set that contains all parameter vectors used in the optimization process, and that smooth solution of the system of ordinary differential equations (1b) exists on $[t_0,t_1]$ whenever $x \in X$. In this case we can compute derivatives of both the solution vector $y(x,t)$ and the integral in (1a) with respect to the parameter vector $x \in \mathbb{R}^n$, by changing the order of differentiation, as will be shown in Section 2.

[1] This work was supported by the Grant Agency of the Czech Republic under grant 201/93/0429.

From a numerical point of view we can replace the problem (1) by

$$F(x) = F_A(x, t_1) + f_T(y(x, t_1)) \tag{2a}$$

where

$$\frac{\mathrm{d}y(x,t)}{\mathrm{d}t} = f_S(x, y(x,t), t), \qquad y(x, t_0) = f_I(x) \tag{2b}$$

and

$$\frac{\mathrm{d}F_A(x,t)}{\mathrm{d}t} = f_A(y(x,t), t), \qquad F_A(x, t_0) = 0 \tag{2c}$$

so that the integral in (1a) is replaced by an additional differential equation (2c). The main advantage of this replacement consist in the elimination of all interior points of the interval $[t_0, t_1]$. The objective function depends only on the terminal values $y(x, t_1)$ and $F_A(x, t_1)$. Therefore both (2b) and (2c) can be solved simultaneously using efficient numerical methods utilizing large integration steps obtained by suitable stepsize control.

Suppose we have available efficient optimization methods and efficient methods for initial value problems of ordinary differential equations. The main purpose of this paper is to show how these methods can be efficiently applied to dynamical systems described by (1) or (2). Even if the description (1) or (2) is not the most general, it contains a broad class of real problems and it can be easily generalized using the approach proposed in Section 2. Note also that (1) or (2) define only the objective function. If we have available efficient constrained optimization methods we can append arbitrary constraints on parameters to (1) or (2).

We confine most of our attention to the special case when the objective function is an integral of squares (8). This objective function is commonly used as a scalar criterion for continuous approximation, and it is applicable, for instance, as a tool for the design of electrical networks or mechanical systems in the time domain.

The paper is organized as follows. In Section 2 we describe procedures for both the gradient and the Hessian matrix evaluations. These procedures consist in solving differential equations as augmented and as adjoint (the augmented system is solved in a forward direction while the adjoint system is solved in a backward direction). Furthermore, we derive a Gauss–Newton–like method that is suitable for small residual integral of squares problems. Convergence properties of this method are studied in Section 3. Section 4 contains practical considerations concerning optimization methods and methods for initial value problems. Numerical experiments are reported in Section 5. In these sections we use the notation $\mathrm{d}/\mathrm{d}t$ and $\mathrm{d}/\mathrm{d}x$ for differentiation with respect to t and total differentiation with respect to x, respectively, and the notation $\partial/\partial x$ and $\partial/\partial y$ for partial differentiation with respect to x and differentiation with respect to y, respectively.

2. COMPUTATION OF DERIVATIVES

In this section we describe several procedures for computing the gradient and the Hessian matrix (or its approximation) of the objective function (1a). We suppose that all conditions stated in Section 1 are satisfied so that smooth solutions of both

(2b) and (2c) exist and their derivatives can be computed by a changing order of differentiation.

(a) Gradient evaluation using forward integration:

Let $u(x,t) = \mathrm{d}y(x,t)/\mathrm{d}x \in \mathbb{R}^{n_S \times n}$ be a matrix with n_S rows and n columns. Differentiating (2) we obtain

$$g^{\mathrm{T}}(x) = g_A^{\mathrm{T}}(x,t_1) + \frac{\partial f_T(y(x,t_1))}{\partial y}\, u(x,t_1) \tag{3a}$$

where

$$\frac{\mathrm{d}u(x,t)}{\mathrm{d}t} = \frac{\partial f_S(x,y(x,t),t)}{\partial y}\, u(x,t) + \frac{\partial f_S(x,y(x,t),t)}{\partial x}, \qquad u(x,t_0) = \frac{\mathrm{d}f_I(x)}{\mathrm{d}x} \tag{3b}$$

and

$$\frac{\mathrm{d}g_A^{\mathrm{T}}(x,t)}{\mathrm{d}t} = \frac{\partial f_A(y(x,t),t)}{\partial y}\, u(x,t), \qquad g_A^{\mathrm{T}}(x,t_0) = 0 \tag{3c}$$

and where $g^{\mathrm{T}}(x) = \mathrm{d}F(x)/\mathrm{d}x$ and $g_A^{\mathrm{T}}(x,t) = \mathrm{d}F_A(x,t)/\mathrm{d}x$. Thus we have to solve the system of $(n_S+1)(n+1)$ differential equations (2b), (2c) and (3b), (3c) in the forward direction for simultaneous computation of both the value (2a) and the gradient (3a) of the objective function.

(b) Gradient evaluation using backward integration:

Let $p(t) \in \mathbb{R}^{n_S}$ be an arbitrary function and $y(x,t)$ be a solution to the differential system (2b) so that $f_S(x,y(x,t),t) - \mathrm{d}y(x,t)/\mathrm{d}t = 0$ for all $t \in [t_0,t_1]$. Then using (1a) we can write

$$F(x) = \int_{t_0}^{t_1} \left\{ f_A(y(x,t),t) + p^{\mathrm{T}}(t) \left(f_S(x,y(x,t),t) - \frac{\mathrm{d}y(x,t)}{\mathrm{d}t} \right) \right\} \mathrm{d}t + f_T(y(x,t_1))$$

and utilizing integration per partes we obtain

$$\begin{aligned} F(x) \;=\; & \int_{t_0}^{t_1} \left\{ f_A(y(x,t),t) + p^{\mathrm{T}}(t)\, f_S(x,y(x,t),t) + \frac{\mathrm{d}p^{\mathrm{T}}(t)}{\mathrm{d}t} y(x,t) \right\} \mathrm{d}t \\ & + p^{\mathrm{T}}(t_0)\, y(x,t_0) - p^{\mathrm{T}}(t_1)\, y(x,t_1) + f_T(y(x,t_1)) \,. \end{aligned}$$

The last formula can be differentiated with respect to the parameter vector $x \in \mathbb{R}^n$ so that we get

$$\begin{aligned} g^{\mathrm{T}}(x) \;=\; & \int_{t_0}^{t_1} \left\{ \left[\frac{\partial f_A(y(x,t),t)}{\partial y} + p^{\mathrm{T}}(t)\, \frac{\partial f_S(x,y(x,t),t)}{\partial y} + \frac{\mathrm{d}p^{\mathrm{T}}(t)}{\mathrm{d}t} \right] \frac{\mathrm{d}y(x,t)}{\mathrm{d}x} \right. \\ & \left. + p^{\mathrm{T}}(t) \frac{\partial f_S(x,y(x,t),t)}{\partial x} \right\} \mathrm{d}t \\ & + p^{\mathrm{T}}(t_0) \frac{\mathrm{d}f_I(x)}{\mathrm{d}x} + \left[\frac{\partial f_T(y(x,t_1))}{\partial y} - p^{\mathrm{T}}(t_1) \right] \frac{\mathrm{d}y(x,t_1)}{\mathrm{d}x} \,. \end{aligned}$$

Now we can chose the function $p(t)$ in such a way to eliminate terms with $\mathrm{d}y(x,t)/\mathrm{d}t$. If we choose

$$-\frac{\mathrm{d}p(x,t)}{\mathrm{d}t} = \left(\frac{\partial f_S(x,y(x,t),t)}{\partial y}\right)^{\mathrm{T}} p(x,t) + \left(\frac{\partial f_A(y(x,t),t)}{\partial y}\right)^{\mathrm{T}},$$
$$p(x,t_1) = \left(\frac{\partial f_T(y(x,t_1))}{\partial y}\right)^{\mathrm{T}}$$

then

$$g^{\mathrm{T}}(x) = \int_{t_0}^{t_1} p^{\mathrm{T}}(x,t)\,\frac{\partial f_S(x,y(x,t),t)}{\partial x}\,\mathrm{d}t + p^{\mathrm{T}}(x,t_0)\,\frac{\mathrm{d}f_I(x)}{\mathrm{d}x}\,. \tag{4a}$$

This result can be summarized in the form

$$g(x) = \tilde{g}_A(x,t_0) + \left(\frac{\mathrm{d}f_I(x)}{\mathrm{d}x}\right)^{\mathrm{T}} p(x,t_0) \tag{5a}$$

where

$$-\frac{\mathrm{d}p(x,t)}{\mathrm{d}t} = \left(\frac{\partial f_S(x,y(x,t),t)}{\partial y}\right)^{\mathrm{T}} p(x,t) + \left(\frac{\partial f_A(y(x,t),t)}{\partial y}\right)^{\mathrm{T}},$$
$$p(x,t_1) = \left(\frac{\partial f_T(y(x,t_1))}{\partial y}\right)^{\mathrm{T}} \tag{5b}$$

and

$$-\frac{\mathrm{d}\tilde{g}_A(x,t)}{\mathrm{d}t} = \left(\frac{\partial f_S(x,y(x,t),t)}{\partial x}\right)^{\mathrm{T}} p(t), \qquad \tilde{g}_A(x,t_1) = 0 \tag{5c}$$

(here $\tilde{g}_A(x,t)$ is different from $g_A(x,t)$ in (3)). Thus we have to solve the system of (n_S+1) differential equations (2b), (2c) in the forward direction for computation of the value (2a) and the system of (n_S+n) differential equations (5b), (5c) in the backward direction for computation of the gradient (5a).

(c) Hessian matrix evaluation using forward integration:

Denote $v(x,t) = \mathrm{d}u(x,t)/\mathrm{d}x = \mathrm{d}^2y(x,t)/\mathrm{d}x^2 \in \mathbb{R}^{n_S\times n\times n}$. Differentiating (3) we obtain

$$G(x) = G_A(x,t_1) + u^{\mathrm{T}}(x,t_1)\,\frac{\partial^2 f_T(y(x,t_1))}{\partial y^2}\,u(x,t_1) + \frac{\partial f_T(y(x,t_1))}{\partial y}\,v(x,t_1) \tag{6a}$$

where

$$\begin{aligned}\frac{\mathrm{d}v(x,t)}{\mathrm{d}t} &= \frac{\partial f_S(x,y(x,t),t)}{\partial y}\,v(x,t)\\ &+\left[\frac{\partial^2 f_S(x,y(x,t),t)}{\partial y^2}\,u(x,t) + \frac{\partial^2 f_S(x,y(x,t),t)}{\partial y\partial x}\right]\circ u(x,t)\\ &+\frac{\partial^2 f_S(x,y(x,t),t)}{\partial x\partial y}\,u(x,t) + \frac{\partial^2 f_S(x,y(x,t),t)}{\partial x^2},\end{aligned}$$

$$v(x,t_0) = \frac{\mathrm{d}^2 f_I(x)}{\mathrm{d}x^2} \tag{6b}$$

and

$$\begin{aligned} &\frac{\mathrm{d}G_A(x,t)}{\mathrm{d}t} = u^{\mathrm{T}}(x,t)\,\frac{\partial^2 f_A(y(x,t),t)}{\partial y^2}\,u(x,t) + \frac{\partial f_A(y(x,t),t)}{\partial y}\,v(x,t), \\ &G_A(x,t_0) = 0 \end{aligned} \tag{6c}$$

and where $G(x) = \mathrm{d}^2F(x)/\mathrm{d}x^2$ and $G_A(x,t) = \mathrm{d}^2 f_A(x,t)/\mathrm{d}x^2$. Thus we have to solve the system of $(n_S+1)\,(n^2+n+1)$ differential equations (2b), (2c) and (3b), (3c) and (6b), (6c) in the forward direction for simultaneous computation of all the value (2a) and the gradient (3a) and the Hessian matrix (6a) of the objective function. Note that we have used a nonstandard matrix notation for tensor quantities: the symbol "$\circ$" means a summation over the last but one (middle) index of a cube matrix.

(d) Hessian matrix evaluation using backward integration:

Let $u(t) \in \mathbb{R}^{n_S \times n}$ be an arbitrary function and $p(x,t)$ be a solution to the differential system (5b) so that $\mathrm{d}p(x,t)/\mathrm{d}t + (\partial f_S(x,y(x,t),t)/\partial y)^{\mathrm{T}} p + (\partial f_A(y(x,t),t)/\partial y)^{\mathrm{T}} = 0$. Then using (4) we can write

$$\begin{aligned} g^{\mathrm{T}}(x) \;=\; &\int_{t_0}^{t_1} \left\{ p^{\mathrm{T}}(x,t)\,\frac{\partial f_S(x,y(x,t),t)}{\partial x} \right. \\ &\left. + \left[\frac{\mathrm{d}p^{\mathrm{T}}(x,t)}{\mathrm{d}t} + p^{\mathrm{T}}(x,t)\,\frac{\partial f_S(x,y(x,t),t)}{\partial y} + \frac{\partial f_A(y(x,t),t)}{\partial y} \right] u(t) \right\} \mathrm{d}t \\ &+ p^{\mathrm{T}}(x,t_0)\,\frac{\mathrm{d}f_I(x)}{\mathrm{d}x} \end{aligned}$$

and utilizing integration per partes we obtain

$$\begin{aligned} g^{\mathrm{T}}(x) \;=\; &\int_{t_0}^{t_1} \left\{ p^{\mathrm{T}}(x,t)\,\frac{\partial f_S(x,y(x,t),t)}{\partial x} \right. \\ &\left. + \left[p^{\mathrm{T}}(x,t)\,\frac{\partial f_S(x,y(x,t),t)}{\partial y} + \frac{\partial f_A(y(x,t),t)}{\partial y} \right] u(t) - p^{\mathrm{T}}(x,t)\,\frac{\mathrm{d}u(t)}{\mathrm{d}t} \right\} \mathrm{d}t \\ &+ p^{\mathrm{T}}(x,t_0)\,\frac{\mathrm{d}f_I(x)}{\mathrm{d}x} + \frac{\partial f_T(y(x,t_1))}{\partial y}\,u(t_1) - p^{\mathrm{T}}(x,t_0)\,u(t_0)\,. \end{aligned}$$

The last formula can be differentiated with respect to the parameter vector $x \in \mathbb{R}^n$ so that we get

$$\begin{aligned} G(x) \;=\; &\int_{t_0}^{t_1} \left\{ \left(\frac{\partial f_S(x,y(x,t),t)}{\partial x} \right)^{\mathrm{T}} \frac{\mathrm{d}p(x,t)}{\mathrm{d}x} \right. \\ &\left. + p^{\mathrm{T}}(x,t) \left(\frac{\partial^2 f_S(x,y(x,t),t)}{\partial x^2} + \frac{\partial^2 f_S(x,y(x,t),t)}{\partial x \partial y}\,\frac{\mathrm{d}y(x,t)}{\mathrm{d}x} \right) \right. \end{aligned}$$

$$+ u^{\mathrm{T}}(t)\left[p^{\mathrm{T}}(x,t)\left(\frac{\partial^2 f_S(x,y(x,t),t)}{\partial y \partial x} + \frac{\partial^2 f_S(x,y(x,t),t)}{\partial y^2}\frac{\mathrm{d}y(x,t)}{\mathrm{d}x}\right)\right]$$
$$+u^{\mathrm{T}}(t)\,\frac{\partial^2 f_A(y(x,t),t)}{\partial y^2}\,\frac{\mathrm{d}y(x,t)}{\mathrm{d}x} + \left(\frac{\partial f_S(x,y(x,t),t)}{\partial y}\,u(t)\right)^{\mathrm{T}}\frac{\mathrm{d}p(x,t)}{\mathrm{d}x}$$
$$-\left(\frac{\mathrm{d}u(t)}{\mathrm{d}t}\right)^{\mathrm{T}}\frac{\mathrm{d}p(x,t)}{\mathrm{d}x}\Bigg\}\,\mathrm{d}t + \left(\frac{\mathrm{d}f_I(x)}{\mathrm{d}x}\right)^{\mathrm{T}}\frac{\mathrm{d}p(x,t_0)}{\mathrm{d}x} + p^{\mathrm{T}}(x,t_0)\,\frac{\mathrm{d}^2 f_I(x)}{\mathrm{d}x^2}$$
$$+u^{\mathrm{T}}(t_1)\,\frac{\partial^2 f_T(y(x,t_1))}{\partial y^2}\,\frac{\mathrm{d}y(x,t_1)}{\mathrm{d}x} - u^{\mathrm{T}}(t_0)\,\frac{\mathrm{d}p(x,t_0)}{\mathrm{d}x}\,.$$

Now we can chose the function $u(t)$ in such a way to eliminate terms with $\mathrm{d}p(x,t)/\mathrm{d}x$. Therefore we choose

$$\frac{\mathrm{d}u(x,t)}{\mathrm{d}t} = \frac{\partial f_S(x,y(x,t),t)}{\partial y}\,u(x,t) + \frac{\partial f_S(x,y(x,t),t)}{\partial x}, \qquad u(x,t_0) = \frac{\mathrm{d}f_I(x)}{\mathrm{d}x}$$

so that $u(x,t) = \mathrm{d}y(x,t)/\mathrm{d}x$ by (3b). Then

$$\begin{aligned} G(x) \;=\; & \int_{t_0}^{t_1}\Bigg\{p^{\mathrm{T}}(x,t)\left(\frac{\partial^2 f_S(x,y(x,t),t)}{\partial x^2} + \frac{\partial^2 f_S(x,y(x,t),t)}{\partial x \partial y}\,u(x,t)\right) \\ & + u^{\mathrm{T}}(x,t)\left[p^{\mathrm{T}}(x,t)\left(\frac{\partial^2 f_S(x,y(x,t),t)}{\partial y \partial x} + \frac{\partial^2 f_S(x,y(x,t),t)}{\partial y^2}\,u(x,t)\right)\right] \\ & + u^{\mathrm{T}}(x,t)\frac{\partial^2 f_A(y(x,t),t)}{\partial y^2}\,u(x,t)\Bigg\}\,\mathrm{d}t \\ & + p^{\mathrm{T}}(x,t_0)\,\frac{\mathrm{d}^2 f_I(x)}{\mathrm{d}x^2} + u^{\mathrm{T}}(x,t_1)\frac{\partial^2 f_T(y(x,t_1))}{\partial y^2}\,u(x,t_1)\,. \end{aligned}$$

This result can be summarized in the form

$$G(x) = \widetilde{G}_A(x,t_0) + p^{\mathrm{T}}(x,t_0)\,\frac{\mathrm{d}^2 f_I(x)}{\mathrm{d}x^2} \tag{7a}$$

where

$$\begin{aligned} -\frac{\mathrm{d}\widetilde{G}_A(x,t)}{\mathrm{d}t} \;=\; & p^{\mathrm{T}}(x,t)\left(\frac{\partial^2 f_S(x,y(x,t),t)}{\partial x^2} + \frac{\partial^2 f_S(x,y(x,t),t)}{\partial x \partial y}\,u(x,t)\right) \\ & + u^{\mathrm{T}}(x,t)\left[p^{\mathrm{T}}(x,t)\left(\frac{\partial^2 f_S(x,y(x,t),t)}{\partial y \partial x} + \frac{\partial^2 f_S(x,y(x,t),t)}{\partial y^2}\,u(x,t)\right)\right] \\ & + u^{\mathrm{T}}(x,t)\frac{\partial^2 f_A(y(x,t),t)}{\partial y^2}\,u(x,t) \end{aligned}$$

$$\widetilde{G}_A(x,t_1) = u^{\mathrm{T}}(x,t_1)\frac{\partial^2 f_T(y(x,t_1))}{\partial y^2}\,u(x,t_1) \tag{7c}$$

(here $\widetilde{G}_A(x,t)$ is different from $G_A(x,t)$ in (6)). Thus we have to solve the system of $(n_S+1)\,(n+1)$ differential equations (2b), (2c) and (3b), (3c) in the forward direction

for simultaneous computation of both the value (2a) and the gradient (3a) and the system of $(n_S + n^2)$ differential equations (5b), (7c) in the backward direction for computation of the Hessian matrix (7a).

Equations (7) were derived from (5) using an arbitrary matrix $u(t) \in \mathbb{R}^{n_S \times n}$. Note that the same result can be obtained from (3) using an arbitrary vector $p(t) \in \mathbb{R}^{n_S}$.

(e) Hessian matrix approximation using forward integration:

Suppose that the functions $f_A : \mathbb{R}^{n_S} \times [t_0, t_1] \to \mathbb{R}$ and $f_T : \mathbb{R}^{n_S} \to \mathbb{R}$ have the special form

$$f_A(y(x,t),t) = \frac{1}{2}(y(x,t) - z(t))^{\mathrm{T}}\, W(t)(y(x,t) - z(t)) \tag{8a}$$

so that

$$\frac{\partial f_A(y(x,t),t)}{\partial y} = W(t)(y(x,t) - z(t)), \qquad \frac{\partial^2 f_A(y(x,t),t)}{\partial y^2} = W(t)$$

and

$$f_T(y(x,t_1)) = \frac{1}{2}(y(x,t_1) - z(t_1))^{\mathrm{T}}\, W_1(y(x,t_1) - z(t_1)) \tag{8b}$$

so that

$$\frac{\partial f_T(y(x,t_1))}{\partial y} = W_1(y(x,t_1) - z(t_1)), \qquad \frac{\partial^2 f_T(y(x,t_1))}{\partial y^2} = W_1$$

where $W(t) \in \mathbb{R}^{n_S \times n_S}$ is a symmetric positive semidefinite matrix and $W_1 \neq W(t_1)$ in general. If $F(x) \to 0$ then necessarily $y(x,t) \to z(t)$ so that $\partial f_A(y(x,t),t)/\partial y \to 0$ and $\partial f_T(y(x,t_1))/\partial y \to 0$. After substituting the last assertions into (6) we obtain

$$G(x) \approx B(x) = B_A(x,t_1) + u^{\mathrm{T}}(x,t_1)\, W_1\, u(x,t_1) \tag{9a}$$

and

$$\frac{\mathrm{d}B_A(x,t)}{\mathrm{d}t} = u^{\mathrm{T}}(x,t)\, W(t)\, u(x,t), \qquad B_A(x,t_0) = 0. \tag{9c}$$

Thus we have to solve the system of $(n_S + 1)(n+1) + n^2$ differential equations (2b), (2c) and (3b), (3c) and (9c) in the forward direction for simultaneous computation of all the value (2a) and the gradient (3a) and the approximation of the Hessian matrix (9a).

(f) Hessian matrix approximation using backward integration:

Assume that $\partial f_A(y(x,t),t)/\partial y = 0$ and $\partial f_T(y(x,t_1))/\partial y = 0$, as in the previous case. Then (5b) implies that $p(x,t) = 0$ for all $t \in [t_0, t_1]$. Substituting this solution into (7), we obtain

$$G(x) \approx B(x) = \widetilde{B}_A(x,t_0)$$

where

$$-\frac{\mathrm{d}\widetilde{B}_A(x,t)}{\mathrm{d}t} = u^{\mathrm{T}}(x,t)\,W(t)\,u(x,t), \qquad \widetilde{B}_A(x,t_1) = u^{\mathrm{T}}(x,t_1)\,W_1\,u(x,t_1).$$

Therefore, $\widetilde{B}_A(x,t_0) = B_A(x,t_1) + u^{\mathrm{T}}(x,t_1)\,W_1\,u(x,t_1)$ and backward integration gives the same result as the forward one. Since forward integration can be implemented by a more easier and more efficient way than the backward one, backward integration is not suitable for Hessian matrix approximation.

We have derived several procedures for both the gradient and the Hessian matrix evaluations. In fact Hessian matrices will not be used in practical implementations since their evaluations require a great amount of computations. Still, the formulas for Hessian matrices allowed us to derive an efficient procedure (9) for their approximations which leads to efficient Gauss–Newton–like methods.

3. CONVERGENCE PROPERTIES

In this section, we will study convergence properties of Gauss–Newton–like methods for integral of squares problems that use the matrix $B(x)$ given by (9) instead of the Hessian matrix $G(x)$. The Gauss–Newton–like methods are usually realized in a trust region framework which leads to good global convergence properties. Assume a class of methods which can be described by the following algorithmic scheme.

Algorithm 3.1.

Data: $0 < \beta_1 < \beta_2 < 1 < \gamma_1$, $0 < \rho_1 < \rho_2 < 1$, $0 < \varepsilon_1 < \varepsilon_2 < 1$.

Step 1: Choose an initial point $x_0 \in \mathbb{R}^n$ and an initial trust region bound $\Delta_0 > 0$. Set $i := 0$.

Step 2: Compute the value $F_i = F(x_i)$, the gradient $g_i = g(x_i)$ and the approximation of the Hessian matrix $B_i = B(x_i)$ by (2), (3) and (9) respectively. If either $F_i \leq \varepsilon_1$ or $\|g_i\| \leq \varepsilon_2$ then stop.

Step 3: Determine the vector $d_i \in \mathbb{R}^n$ so that:

$$d_i = \operatorname*{arg\,min}_{\|d\| \leq \Delta_i} Q_i(d)$$

where

$$Q_i(d) = \frac{1}{2} d^{\mathrm{T}} B_i d + d^{\mathrm{T}} g_i$$

is a local quadratic approximation of the objective function $F : \mathbb{R}^n \to \mathbb{R}$ in a neighborhood of the point x_i.

Step 4: Compute the value $F(x_i + d_i)$ by (2) and the ratio $\rho_i = (F(x_i + d_i) - F_i)/Q_i(d_i)$. If $\rho_i < \rho_1$ then compute the value β_i, $\beta_1 \leq \beta_i \leq \beta_2$, by a quadratic interpolation and set $\Delta_{i+1} = \beta_i \|d_i\|$. If $\rho_1 \leq \rho_i \leq \rho_2$ then set $\Delta_{i+1} = \Delta_i$. If $\rho_2 < \rho_i$ then set $\Delta_{i+1} = \max(\Delta_i, \gamma_1 \|d_i\|)$.

Step 5: If $\rho_i \leq 0$ then set $x_{i+1} = x_i$, $i := i + 1$ and go to Step 3, otherwise set $x_{i+1} = x_i + d_i$, $i := i + 1$ and go to Step 2.

Typical data are $\beta_1 = 0.05$, $\beta_2 = 0.75$, $\gamma_1 = 2.0$, $\rho_1 = 0.1$, $\rho_2 = 0.9$, $\varepsilon_1 = 10^{-12}$, $\varepsilon_2 = 10^{-6}$.

Convergence properties of trust region methods were studied in [7]–[8]. We use these results together with classical theory of differential equations to prove the global and superlinear convergence of the Gauss–Newton–like method represented by Algorithm 3.1. Denote

$$X = \{x \in \mathbb{R}^n : F(x) \leq F(x_0)\}$$

and assume the following conditions hold

(A1) System (1b) has a unique continuous solution $y(x,t)$ on $[t_0, t_1]$ for all $x \in X$ and

$$\max_{t \in [t_0,t_1]} \|y(x,t)\| \leq \overline{K}$$

holds for all $x \in X$.

(A2) Functions $W(t)$ and $z(t)$ are bounded on $[t_0, t_1]$, i.e. $\|W(t)\| \leq \overline{K}$ and $\|y(x,t) - z(t)\| \leq \overline{K}$, say, on $[t_0, t_1]$ for all $x \in X$. Also $\|W_1\| \leq \overline{K}$.

(A3) Function $f_I(x)$ is Lipschitz continuously differentiable with respect to x on X. It means that $\mathrm{d}f_I(x)/\mathrm{d}x$ exists on X and

$$\left\| \frac{\mathrm{d}f_I(x_2)}{\mathrm{d}x} - \frac{\mathrm{d}f_I(x_1)}{\mathrm{d}x} \right\| \leq \overline{L} \|x_2 - x_1\|$$

for all $x_1 \in X$ and $x_2 \in X$.

(A4) Function $f_S(x,y,t)$ is Lipschitz continuously differentiable with respect to x and y on $X \times Y \times [t_0, t_1]$ where $Y = \{y \in \mathbb{R}^{n_S} : \|y\| \leq \overline{K}\}$. It means that $\partial f_S(x,y,t)/\partial x$ exists on X and

$$\left\| \frac{\partial f_S(x_2,y,t)}{\partial x} - \frac{\partial f_S(x_1,y,t)}{\partial x} \right\| \leq \overline{L} \|x_2 - x_1\|$$

$$\left\| \frac{\partial f_S(x,y_2,t)}{\partial x} - \frac{\partial f_S(x,y_1,t)}{\partial x} \right\| \leq \overline{L} \|y_2 - y_1\|$$

for all $x \in X$, $x_1 \in X$, $x_2 \in X$, $y \in Y$, $y_1 \in Y$, $y_2 \in Y$ and $t \in [t_0, t_1]$, and the same holds for $\partial f_S(x,y,t)/\partial y$.

For the sake of simplicity, we use the same constant $\overline{K}$ in both (A1) and (A2) and the same constant $\overline{L}$ in both (A3) and (A4).

Assuming (A1) is very natural since we require, for optimization process, that a bounded unique solution $y(x,t)$ exists for all $x \in X$. This assumption together with assumptions (A2)–(A4) imply an existence and continuity of the function $u(x,t) = \mathrm{d}y(x,t)/\mathrm{d}t$ which have to satisfy the equation (3b) (see [4]). For subsequent considerations we will need the following lemma (see [4]).

Lemma 3.1. Consider the linear system

$$\mathrm{d}y(t)/\mathrm{d}t = A(t)y(t) + b(t), \qquad y(t_0) = y_0$$

with $A(t)$ and $b(t)$ continuous on $[t_0, t_1]$. Then

$$\|y(t)\| \leq (\|y(t_0)\| + \int_{t_0}^{t} \|b(\tau)\| \, \mathrm{d}\tau) \exp\left(\int_{t_0}^{t} \|A(\tau)\| \, \mathrm{d}\tau\right)$$

for all $t \in [t_0, t_1]$.

Now we can prove the main results.

Theorem 3.1. Let the assumptions (A1)–(A4) hold. Then Algorithm 3.1 is globally convergent in the sense that

$$\liminf_{i \to \infty} \|g(x_i)\| = 0 \, .$$

P r o o f. We have to prove that the matrix $B(x)$ given by (9) is bounded on X and that the gradient $g(x)$ given by (3) is Lipschitz continuous on X. These conditions already imply global convergence of a trust region method as it is proved in [7]–[8].

First we prove boundedness of the matrix $B(x)$. Since Lipschitz continuity on a compact set imply boundedness, we can assume that $\|\mathrm{d}f_I(x)/\mathrm{d}x\| \leq \overline{K}$ on X and $\|\partial f_S(x,y,t)/\partial x\| \leq \overline{K}$, $\|\partial f_S(x,y,t)/\partial y\| \leq \overline{K}$ on $X \times Y \times [t_0, t_1]$ respectively (for the sake of simplicity we use the same constant $\overline{K}$ as in (A1) and (A2)). If we apply Lemma 3.1 on the system (3b) we obtain

$$\|u(x,t)\| \leq (\overline{K} + \overline{K}(t_1 - t_0)) \exp(\overline{K}(t_1 - t_0)) \triangleq \overline{M}$$

so that by (9) we can write

$$\begin{aligned} \|B(x)\| &\leq \int_{t_0}^{t_1} \|u^{\mathrm{T}}(x,t) \, W(t) \, u(x,t)\| \, \mathrm{d}t + \|u^{\mathrm{T}}(x,t_1) \, W_1 \, u(x,t_1)\| \\ &\leq \overline{KM}^2 (t_1 - t_0) + \overline{KM}^2 \, . \end{aligned}$$

Second, we prove Lipschitz continuity of the gradient $g(x)$. From boundedness of $u(x,t) = \mathrm{d}y(x,t)/\mathrm{d}t$ on X it follows that $\|y(x_2,t) - y(x_1,t)\| \leq \overline{M}\|x_2 - x_1\|$ which together with (A4) gives as

$$\begin{aligned} \left\| \frac{\partial f_S(x_2,y_2,t)}{\partial x} - \frac{\partial f_S(x_1,y_1,t)}{\partial x} \right\| &\leq \left\| \frac{\partial f_S(x_2,y_2,t)}{\partial x} - \frac{\partial f_S(x_2,y_1,t)}{\partial x} \right\| \\ &+ \left\| \frac{\partial f_S(x_2,y_1,t)}{\partial x} - \frac{\partial f_S(x_1,y_1,t)}{\partial x} \right\| \\ &\leq \overline{L}(\|y_2 - y_1\| + \|x_2 - x_1\|) \\ &\leq \overline{L}(\overline{M} + 1) \, \|x_2 - x_1\| \end{aligned}$$

as similar inequality

$$\left\| \frac{\partial f_S(x_2, y_2, t)}{\partial y} - \frac{\partial f_S(x_1, y_1, t)}{\partial y} \right\| \leq \overline{L}(\overline{M} + 1) \, \|x_2 - x_1\| \, .$$

Using (3) we get

$$\begin{aligned} \frac{d(u(x_2,t) - u(x_1,t))}{\mathrm{d}t} &= \frac{\partial f_S(x_2, y_2, t)}{\partial y} (u(x_2,t) - u(x_1,t)) \\ &+ \left(\frac{\partial f_S(x_2, y_2, t)}{\partial y} - \frac{\partial f_S(x_1, y_1, t)}{\partial y} \right) u(x_1,t) \\ &+ \left(\frac{\partial f_S(x_2, y_2, t)}{\partial x} - \frac{\partial f_S(x_1, y_1, t)}{\partial x} \right) \end{aligned}$$

and

$$u(x_2, t_0) - u(x_1, t_0) = \frac{\mathrm{d}f_I(x_2)}{\mathrm{d}x} - \frac{\mathrm{d}f_I(x_1)}{\mathrm{d}x} \, .$$

Applying Lemma 3.1 on the last system and using the above inequalities together with boundedness of $u(x,t)$ on X we obtain

$$\|u(x_2,t) - u(x_1,t)\| \leq \overline{L}(1 + (\overline{M}+1)^2 (t_1 - t_0)) \exp(\overline{K}(t_1 - t_0)) \|x_2 - x_1\| \triangleq \overline{N} \|x_2 - x_1\| \, .$$

This together with (A2) and boundedness of $u(x,t)$ on X gives

$$\begin{aligned} & \|u^{\mathrm{T}}(x_2,t) \, W(t) \, (y(x_2,t) - z(t)) - u^{\mathrm{T}}(x_1,t) \, W(t) \, (y(x_1,t) - z(t))\| \\ \leq \; & \|u^{\mathrm{T}}(x_2,t) \, W(t) \, (y(x_2,t) - y(x_1,t))\| \\ + \; & \|(u(x_2,t) - u(x_1,t))^{\mathrm{T}} \, W(t) \, (y(x_1,t) - z(t))\| \\ \leq \; & \overline{KM} \|y(x_2,t) - y(x_1,t)\| + \overline{K}^2 \|u(x_2,t) - u(x_1,t)\| \\ \leq \; & \overline{K}(\overline{M}^2 + \overline{KN}) \|x_2 - x_1\| \end{aligned}$$

so that (3) and (8) imply

$$\begin{aligned} \|g(x_2) - g(x_1)\| \leq & \int_{t_0}^{t_1} \|u^{\mathrm{T}}(x_2,t) \, W(t) \, (y(x_2,t) - z(t)) \\ & - u^{\mathrm{T}}(x_1,t) \, W(t) \, (y(x_1,t) - z(t))\| \mathrm{d}t \\ & + \|u^{\mathrm{T}}(x_2,t_1) \, W_1 \, (y(x_2,t_1) - z(t)) \\ & - u^{\mathrm{T}}(x_1,t) \, W_1 \, (y(x_1,t_1) - z(t_1))\| \\ \leq & \; \overline{K}(\overline{M}^2 + \overline{KN}) \, ((t_1 - t_0) + 1) \, \|x_2 - x_1\| \end{aligned}$$

and Lipschitz continuity of the gradient $g(x)$ is proved. □

Theorem 3.2. Let $\{x_i\}_{i=0}^{\infty}$, be a sequence of points generated by Algorithm 3.1 such that $x_i \to x^*$ as $i \to \infty$ where $x^* \in \mathbb{R}^n$ is a point that satisfies a second order sufficient condition for local minimum of the function $F(x)$. Suppose that (A1)

and (A2) hold and continuous and bounded second oder derivatives $\partial^2 f_I(x)/\partial x^2$, $\partial^2 f_S(x,y,t)/\partial x^2$, $\partial^2 f_S(x,y,t)/\partial x \partial y$, $\partial^2 f_S(x,y,t)/\partial y^2$ exist for all x from some neighborhood $X^* \subset X$ of $x^* \in \mathbb{R}^n$ and for all $y \in Y$ and $t \in [t_0, t_1]$. Then, if $F(x_i) \to 0$ as $i \to \infty$, the sequence $\{x_i\}_{i=0}^{\infty}$ converges superlinearly to $x^* \in \mathbb{R}^n$ in the sense that

$$\lim_{i \to \infty} \frac{\|x_{i+1} - x^*\|}{\|x_i - x^*\|} = 0.$$

Proof. We have to prove that $B(x_i) \to G(x_i)$ when $x_i \to \infty$. This condition together with the positive definiteness of $B(x^*)$ already imply superlinear convergence of a trust region method as it is proved in [7].

Continuity and boundedness of second order derivatives imply continuity and boundedness of the function $v(x,t) = \mathrm{d}u(x,t)/\mathrm{d}x$ on $X^* \times [t_0, t_1]$ (it follows from (6b) using Lemma 3.1). Therefore we can write $\|v(x_i,t)\| \leq \overline{C}$ for all $t \in [t_0, t_1]$ whenever $x_i \in X^*$. Using this fact together with (6) and (9) we get

$$\begin{aligned} \|G(x_i) - B(x_i)\| &\leq \int_{t_0}^{t_1} \|(y(x_i,t) - z(t))^{\mathrm{T}} W(t)\, v(x_i,t)\| \,\mathrm{d}t \\ &\quad + \|(y(x_i,t_1) - z(t_1))^{\mathrm{T}} W_1\, v(x_i,t_1)\| \\ &\leq \overline{CK}^{1/2} \int_{t_0}^{t_1} \|W^{1/2}(t)\,(y(x_i,t) - z(t))\| \,\mathrm{d}t \\ &\quad + \overline{CK}^{1/2} \|W_1^{1/2}\,(y(x_i,t_1) - z(t_1))\| \,. \end{aligned}$$

But from (1a) and (8) we obtain

$$\begin{aligned} 2F(x_i) &= \int_{t_0}^{t_1} (y(x_i,t) - z(t))^{\mathrm{T}} W(t)\,(y(x_i,t) - z(t)) \,\mathrm{d}t \\ &\quad + (y(x_i,t_1) - z(t_1))^{\mathrm{T}} W_1\,(y(x_i,t_1) - z(t_1)) \\ &= \int_{t_0}^{t_1} \|W^{1/2}(t)\,(y(x_i,t) - z(t))\|^2 \,\mathrm{d}t + \|W_1^{1/2}\,(y(x_i,t_1) - z(t_1))\|^2 \end{aligned}$$

so that $F(x_i) \to 0$ only if $W^{1/2}(t)\,(y(x_i,t) - z(t)) \to 0$ in the L_2 norm and $W_1^{1/2}(y(x_i,t_1) - z(t_1)) \to 0$ in the Euclidean norm. This together with estimation of $\|G(x_i) - B(x_i)\|$ proves that $B(x_i) \to G(x_i)$ as $i \to \infty$ (since L_1 norm of bounded function on bounded interval is equivalent with the L_2 norm). □

We have proved that Gauss–Newton–like method represented by Algorithm 3.1 converges superlinearly if it is used for zero residual problem. In the case of a large residuum the superlinear convergence is usually lost. Therefore it is advantageous to combine Gauss–Newton–like method with the BFGS quasi–Newton method. A very effective possibility is proposed in [2]. It consists in replacing Step 2 of Algorithm 3.1 by a sequence of the following two steps.

Step 2a: If $i = 0$ or $F_{i-1} - F_i > \eta_1 F_{i-1}$ then compute the value $F_i = F(x_i)$, the gradient $g_i = g(x_i)$ and the symmetric positive semidefinite matrix $B_i = B(x_i)$ by (2), (3) and (9) respectively. If either $F_i \leq \varepsilon_1$ or $||g_i|| \leq \varepsilon_2$ then stop.

Step 2b: If $i > 0$ and $F_{i-1} - F_i \leq \eta_1 F_{i-1}$ then compute the value $F_i = F(x_i)$, and the gradient $g_i = g(x_i)$ by (2) and (3) respectively. If either $F_i \leq \varepsilon_1$ or $||g_i|| \leq \varepsilon_2$ then stop. Otherwise set $d_{i-1} = x_i - x_{i-1}$, $y_{i-1} = g_i - g_{i-1}$ and compute

$$B_i = B_{i-1} + \frac{y_{i-1} y_{i-1}^{\mathrm{T}}}{d_{i-1}^{\mathrm{T}} y_{i-1}} - \frac{B_{i-1} d_{i-1} (B_{i-1} d_{i-1})^{\mathrm{T}}}{d_{i-1}^{\mathrm{T}} B_{i-1} d_{i-1}} .$$

A typical value is $\eta_1 = 10^{-4}$. We denote such a combination as GN+QN method. A more detailed description of GN+QN-like methods is given in [2] and [6].

4. PRACTICAL CONSIDERATIONS

First we would note that forward integration leads to larger systems of differential equations then a backward one. On the other hand backward integration has this unpleasant feature: the adjoint system requires the solution of the basic system (2b) which is usually obtained by forward integration. There are two possibilities for proceeding. The first possibility, we denote as B1, consists in additional solution of the basic system in the backward direction

$$\frac{\mathrm{d}y(x,t)}{\mathrm{d}t} = f_S(x, y(x,t), t), \qquad y(x,t_1) - \text{given by forward integration.}$$

This system is added to the system (5b) and (5c) so that the resulting system contains $2n_S + n$ differential equations. When the basic system (2b) is sensitive to initial values and, at the same time, the value $y(x,t_1)$ computed by forward integration is affected by unneglected global truncation error then we can lose some precision and also stability. However this situation never appeared in our numerical experiments.

The second possibility, we denote as B2, consists in storing the solution to the basic system in all mesh points during forward integration. Backward integration then uses the same mesh points as a forward one so that the solution of the basic system is always available. If we denote by n_A the number of mesh points used in forward integration, we have to store $n_A n_S$ additional values. Since mesh points are given automatically by a stepsize control (based on local truncation error estimation) their number could be too large. Moreover, since the adjoint system (5b) used in backward integration is different from the basic system (2b), the mesh points obtained during forward integration can be unsuitable for backward integration. Also utilizing uniformly distributed mesh points may not be suitable since solution of the basic system can vary quickly in some parts of the integration interval so that uniformly distributed mesh points can be insufficient in this case.

The following table summarizes the requirements for individual procedures.

Table 1. Requirements of individual methods.

method	realization F	realization B1	realization B2
QN(0)	$N_f = n_S + 1$ $N_b = 0$ $N_e = n + 1$ $N_s = 0$	irrelevant	irrelevant
QN(1)	$N_f = (n_S + 1)(n + 1)$ $N_b = 0$ $N_e = 1$ $N_s = 0$	$N_f = n_S + 1$ $N_b = 2n_S + n^2$ $N_e = 1$ $N_s = 0$	$N_f = n_S + 1$ $N_b = n_S + n$ $N_e = 1$ $N_s = n_A n_S$
MN(1)	$N_f = (n_S + 1)(n + 1)$ $N_b = 0$ $N_e = n + 1$ $N_s = 0$	$N_f = n_S + 1$ $N_b = 2n_S + n^2$ $N_e = n + 1$ $N_s = 0$	$N_f = n_S + 1$ $N_b = n_S + n$ $N_e = n + 1$ $N_s = n_A n_S$
MN(2)	$N_f = (n_S + 1)(n^2 + n + 1)$ $N_b = 0$ $N_e = 1$ $N_s = 0$	$N_f = (n_S + 1)(n + 1)$ $N_b = n_S(n + 2) + n^2$ $N_e = 1$ $N_s = 0$	$N_f = (n_S + 1)(n + 1)$ $N_b = n_S + n^2$ $N_e = 1$ $N_s = n_A(n_S + 1)n$
GN(1)	$N_f = (n_S + 1)(n + 1) + n^2$ $N_b = 0$ $N_e = 1$ $N_s = 0$	unsuitable	unsuitable

Rows of Table 1 correspond to selected optimization methods:

QN(0) — quasi–Newton method with numerical differentiation,
QN(1) — quasi–Newton method with gradients computed by integration,
MN(1) — modified Newton method with numerical differentiation,
MN(2) — modified Newton method with Hessian matrices computed by integration,
GN(1) — Gauss–Newton method with gradients computed by integration.

The methods QN(0), QN(1), MN(1), MN(2) can be used in the general case while the method GN(1) is applicable only in the case of integral of squares. Columns of Table 1 correspond to various realizations of evaluation:

F — forward integration,
B1 — backward integration with recomputing the basic solution,
B2 — backward integration with storing the basic solution.

Table 1 contains four numbers:

N_f — number of equations in the forward system,
N_b — number of equations in the backward system,
N_e — number of repeated evaluations during numerical differentiation,
N_s — number of additional stored values.

From Table 1, we can deduce, for example, that the total number of solved differential equations, which is equal $(N_f + N_b)N_e$, is the same for both the QN(0)/F and

QN(1)/F methods while the total number of stored values, which is approximately equal $15\max(N_f, N_b) + N_s$ if we use the DOPRI8 integration procedure, is much less for the QN(0)/F method. This observation demonstrates certain advantages to methods that use numerical differentiation.

Now let us concentrate our attention on numerical solution to differential systems. There are two possibilities: the basic system (2b) can be either stiff or nonstiff. We confine our attention only to the nonstiff systems. In the nonstiff case we should use high order explicit methods which give solution with high precision and utilize sufficiently large steps. In [3] the Dormand–Prince methods DOPRI5 and DOPRI8 were recommended which are the Runge–Kutta methods of 5 and 8 order respectively, with automatic stepsize control. These methods require $9n_E$ and $15n_E$ of storage space respectively where n_E is a number of differential equations. All numerical experiments proposed in the next section were made using these methods.

Finally let us make several comments on optimization methods. Consider the special case of the sum of squares (8). If the problem has a small residuum, which means that the optimal value of $F(x)$ is small, then we can use the Gauss–Newton like method (9). In the opposite case the Gauss–Newton–like method can lose a convergence so that quasi–Newton methods can be more efficient. Another possibility is using the hybrid GN+QN method as described in Section 3. This method has usually good convergence properties for both small and large residual problems. Experience with all GN, QN and GN+QN methods is proposed in the next section.

At the end of this section, we would like to show a connection between our work and so-called automatic differentiation. Automatic differentiation is based on a set of highly structured explicit equations, generally nonlinear, which correspond to elementary functions and algebraic operations. The structure of these equations is determined by the computational graph, which defines the resulting function from elementary functions and algebraic operations. We want to compute derivatives of the resulting function, knowing explicitly the derivatives of elementary functions and algebraic operations; see [5]. This problem can be solved using an implicit function theorem. There are two possibilities how the implicit function theorem can be interpreted: direct elimination, which leads to so-called forward accumulation, and the Lagrange multiplier approach, which leads to so-called backward accumulation; see [1].

A similar approach can be applied to preliminarily discretized dynamical systems, which are also described by a set of explicit nonlinear equations. The computational graph is now given by a discretization method (the Euler method for instance), and the implicit function theorem gives two possible procedures: forward integration and backward integration. This approach, often used for optimization of dynamical systems, has one disadvantage: preliminary discretization usually does not allow us to obtain a minimizer of the original continuous problem with a required precision. Therefore, we used a slightly different approach; we consistently use a continuous formulation together with principles from calculus of variations. In such a way, we obtain differential equations, which allow us to compute resulting function values, together with gradients and Hessian matrices, with an arbitrary precision. This precision is influenced by the selected numerical method, i.e., by

the choice of an integration formula along with a reasonable strategy of stepsize control. Mesh points, characterizing discretization method or computational graph, are automatically generated during the integration process, according to the local truncation error. Therefore, computational graphs of forward and backward integration, respectively, may be different from each other. Moreover, higher order integration formulas, such as DOPRI5 and DOPRI8, use additional intermediate points, which are not distributed symetrically; they are different for both forward and backward traversal. Therefore, gradients and Hessian matrices, computed by backward integration, are not related to the function values, computed by forward integration, even if both the mesh points and the integration formula are the same. These considerations give explanation for the difference between the strategies B1 and B2 (see Table 1). Mesh points in the case B1 are selected from the systems (2) and (5) together, while mesh points in the case B2 are derived from the system (2) only. This fact can cause insufficient precision for the case B2, when a higher order integration formula is used: the stepsizes determined from the system (2) can be too large for the system (5). This phenomenon was proved by our computational experience; see Tables 2a and 2b for DOPRI8. The continuous approach has a further advantage: an arbitrary integration formula, not only explicit, can be used. This allows us to use implicit integration formulas for stiff dynamical systems.

5. NUMERICAL EXPERIMENTS

In this section we demonstrate properties of several optimization methods with various procedures for evaluation of gradients and approximation of Hessian matrices. We use the following test problems:

Problem A:

Consider the objective function

$$F(x) = \int_0^1 \sum_{i=1}^{3} (y_i(t) - z_i(t))^2 \mathrm{d}t$$

where

$$\begin{aligned} \mathrm{d}y_1(t)/\mathrm{d}t &= -x_1 y_1(t) + x_2 y_2(t), \quad y_1(0) = 2 \\ \mathrm{d}y_2(t)/\mathrm{d}t &= -x_1 y_2(t) + x_2 y_3(t), \quad y_2(0) = 1 \\ \mathrm{d}y_3(t)/\mathrm{d}t &= -x_1 y_3(t) + x_3 y_2(t), \quad y_3(0) = -1 \end{aligned}$$

and $z_1(t) = (2+t-t^2/2)\exp(-2t)$, $z_2(t) = (1-t)\exp(-2t)$ and $z_3(t) = -\exp(-2t)$. These functions are solutions to the given differential system so that we have a zero residual problem. The starting point is $x_0 = 0$.

Problem B:

Consider the same objective function and the same differential system as in Problem A, but now $z_1(t) = 2(1-t)$, $z_2(t) = (1-t)$ and $z_3(t) = (t-1)$. These functions

are not solutions to the given differential system so that we have a nonzero residual problem. The starting point is $x_0 = 0$.

Problem C:

Consider the objective function

$$F(x) = ((y_1(1) - 1)^2 + y_3^2(1))/2$$

where

$$\begin{aligned}
dy_1(t)/dt &= y_2(t), & y_1(0) &= x_1 \\
dy_2(t)/dt &= +0.64y_1(t)\exp(y_3(t)/(1+0.05y_3(t))), & y_2(t) &= 0 \\
dy_3(t)/dt &= y_4(t), & y_3(0) &= x_2 \\
dy_4(t)/dt &= -2.56y_1(t)\exp(y_3(t)/(1+0.05y_3(t))), & y_4(t) &= 0.
\end{aligned}$$

This problem is a reformulation of two point boundary value problem arising in chemical kinetics. It is of course a zero residual problem. The starting point is $x_0 = 0$.

We use these problems for demonstrating properties of the methods QN(0), QN(1)/F, QN(1)/B1, QN(1)/B2, GN(1) described in the previous section and also the hybrid method GN(1)+QN(1) based on ideas proposed in [2]. Result of numerical experiments are listed in three tables. Each table corresponds to one problem. Rows of tables correspond to individual methods and columns corresponds to different solvers (DOPRI8 and DOPRI5). Each table contains as numbers n_i, n_f, n_g (n_i is a number of iterations, n_f is a number of function evaluations, n_g is a number of gradient evaluations) as final values $|F|$, $||g||$ obtained by the iterative process as consumed computational time.

Table 2a. Results for problem A.

	DOPRI8 – precision 10^{-9}			DOPRI5 – precision 10^{-9}		
Method	$n_i - n_f - n_g$	$\|F\| - \|\|g\|\|$	time	$n_i - n_f - n_g$	$\|F\| - \|\|g\|\|$	time
QN(0)	18–76–0	$10^{-12} - 10^{-6}$	2.47	19–80–0	$10^{-9} - 10^{-6}$	5.16
QN(1)/F	14–15–15	$10^{-10} - 10^{-6}$	1.54	14–15–15	$10^{-10} - 10^{-6}$	2.80
QN(1)/B1	14–15–15	$10^{-10} - 10^{-6}$	1.38	14–15–15	$10^{-9} - 10^{-6}$	2.64
QN(1)/B2	insufficient precision			14–15–15	$10^{-9} - 10^{-6}$	2.81
GN(1)	5–11–6	$10^{-14} - 10^{-6}$	0.94	5–11–6	$10^{-9} - 10^{-8}$	1.75
GN(1)+QN(1)	5–11–6	$10^{-14} - 10^{-6}$	0.98	5–11–6	$10^{-9} - 10^{-8}$	1.76

Table 2b. Results for problem B.

	DOPRI8 – precision 10^{-9}			DOPRI5 – precision 10^{-9}		
Method	$n_i - n_f - n_g$	$\|F\| - \|\|g\|\|$	time	$n_i - n_f - n_g$	$\|F\| - \|\|g\|\|$	time
QN(0)	18–76–0	$10^{-1} - 10^{-6}$	1.97	18–76–0	$10^{-1} - 10^{-6}$	3.02
QN(1)/F	14–15–15	$10^{-1} - 10^{-6}$	1.38	14–15–15	$10^{-1} - 10^{-6}$	2.14
QN(1)/B1	14–15–15	$10^{-1} - 10^{-6}$	1.10	14–15–15	$10^{-1} - 10^{-6}$	1.81
QN(1)/B2	27–68–68	$10^{-1} - 10^{-5}$	4.34	14–15–15	$10^{-1} - 10^{-6}$	1.76
GN(1)	7–15–8	$10^{-1} - 10^{-6}$	1.05	7–15–8	$10^{-1} - 10^{-6}$	1.71
GN(1)+QN(1)	5–11–6	$10^{-1} - 10^{-6}$	0.83	5–11–6	$10^{-1} - 10^{-6}$	1.26

Table 2c. Results for problem C.

Method	DOPRI8 – precision 10^{-9}			DOPRI5 – precision 10^{-9}		
	$n_i - n_f - n_g$	$\|F\| - \|g\|$	time	$n_i - n_f - n_g$	$\|F\| - \|g\|$	time
QN(0)	35–130–0	$10^{-15} - 10^{-6}$	4.56	35–130–0	$10^{-15} - 10^{-6}$	8.07
QN(1)/F	16–22–22	$10^{-14} - 10^{-5}$	2.72	16–22–22	$10^{-14} - 10^{-5}$	5.44
QN(1)/B1	16–22–22	$10^{-14} - 10^{-5}$	3.02	16–22–22	$10^{-14} - 10^{-5}$	5.33
QN(1)/B2	18–23–23	$10^{-13} - 10^{-5}$	2.85	16–22–22	$10^{-13} - 10^{-5}$	4.56
GN(1)	9–20–10	$10^{-24} - 10^{-10}$	1.65	9–20–10	$10^{-24} - 10^{-10}$	2.96
GN(1)+QN(1)	9–20–10	$10^{-24} - 10^{-10}$	1.76	9–20–10	$10^{-24} - 10^{-10}$	2.96

The above tables show that integration method of higher order is more efficient, if expressed by consumed computational time, than lower order one, even if it requires a greater number of right hand side evaluations in each integration step. The further observation if that the Gauss–Newton–like method GN(1) is very efficient, especially if it is used for zero residual problems and that hybrid method GN(1)+QN(1) keeps this property also for nonzero residual problems. The most important implication of the above tables is that methods for optimization of dynamical systems based on higher order integration routines are able to find a solution with great precision (gradient can be computed with precision about 10^{-6}).

(Received July 27, 1995.)

REFERENCES

[1] Y. G. Evtushenko: Automatic differentiation viewed from optimal control theory. In: Automatic Differentiation of Algorithms: Theory, Implementation, and Application (A. Griewank and G. F. Corliss, eds.), SIAM, Philadelphia 1991, pp. 25–30.

[2] R. Fletcher, and C. Xu: Hybrid methods for nonlinear least squares. IMA J. Numer. Anal. *7* (1987), 371–389.

[3] E. Harier, S. P. Norsett and G. Wanner: Solving Ordinary Differential Equations I, Nonstiff Problems. Springer Verlag, Berlin 1987.

[4] P. Hartman: Ordinary Differential Equations. John Wiley & Sons, New York 1964.

[5] A. Griewank: On automatic differentiation. In: Mathematical Programming: Recent Development and Application (M. Iri and K. Tanabe, eds.), Kluwer Academic Publishers, London 1989, pp. 83–108.

[6] L. Lukšan: Hybrid methods for large sparse nonlinear least squares. J. Optim. Theory Appl. *89* (1996), 575–595.

[7] M. J. D. Powell: Convergence properties of a class of minimization algorithms. In: Nonlinear Programming 2 (O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds.), Academic Press, London 1975, pp. 1–27.

[8] M. J. D. Powell: On the global convergence of trust region algorithms for unconstrained minimization. Math. Programming *29* (1984), 297–303.

Ing. Ladislav Lukšan, DrSc., RNDr. Jan Vlček, CSc., Ústav informatiky a výpočetní techniky AV ČR (Institute of Computer Science – Academy of Sciences of the Czech Republic), Pod vodárenskou věží 2, 18207 Praha 8. Czech Republic.