Journal of Global Optimization (2005) 31:271–286 © Springer 2005

# A Gradient-based Continuous Method for Large-scale Optimization Problems*

LI-ZHI LIAO[1], LIQUN QI[2] and HON WAH TAM[3]
[1]*Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong, PR China (e-mail: liliao@hkbu.edu.hk)*
[2]*Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong (e-mail: maqilq@polyu.edu.hk)*
[3]*Department of Computer Science, Hong Kong Baptist University, Kowloon Tong, Hong Kong, PR China (e-mail: tam@comp.hkbu.edu.hk)*

(Received: 30 March 2004; revised and accepted: 1 April 2004)

**Abstract.** In this paper, we study a gradient-based continuous method for large-scale optimization problems. By converting the optimization problem into an ODE, we are able to show that the solution trajectory of this ODE tends to the set of stationary points of the original optimization problem. We test our continuous method on large-scale problems available in the literature. The simulation results are very attractive.

**Key words:** Continuous method, Large-scale optimization, Ordinary differential equation

## 1. Introduction

We are interested in the following unconstrained optimization problem:

$$\min_{x \in R^n} f(x), \tag{1}$$

where $f(x)$ has continuous first-order derivatives in $R^n$ and $n$ is large, say $n \geqslant 10^6$.

The conventional optimization methods for (1) normally generate a sequence of points, say $\{x_k\}$, starting at the initial point and ending at a stationary point of (1). Two common features for this sequence of points are that: (a) the sequence of points $\{x_k\}$ forms a discrete path from the initial point to the stationary point; (b) the convergence of this sequence of points is normally not considered. Instead, the convergence of $\nabla f(x_k)$ is addressed.

Since 1950s, continuous-path methods have been investigated for the optimization problem (1). The research in this direction was very active in the last two decades because of the seminal work of Hopfield's artificial neural network [7, 8]. Generally speaking, the key idea in continuous-path

*This research was supported in part by Grants FRG/99-00/II-23 and FRG/00-01/II-63 of Hong Kong Baptist University and the Research Grant Council of Hong Kong.

methods is first to convert the optimization problem (1) into the following dynamical system or ordinary differential equation

$$\frac{dx(t)}{dt} = -\nabla f(x(t)), \tag{2}$$

then to study the convergence of the solution $x(t)$ of the ODE (2). Different from the discrete path $\{x_k\}$ generated in conventional optimization methods, a continuous path $x(t)$, which is the solution of (2), can be formed. Therefore, these methods are called continuous-path methods, or simply continuous methods.

The study of continuous methods for optimization involves two steps. First, how should the ODE or dynamical system be established? In general, there are many possible ODE systems for an optimization problem. The ODE (2), which is established based on the steepest descent direction, is just one of them for problem (1). Second, how should we guarantee that the solution of the underlying ODE converges to the desired point? This is a very important and difficult issue. For ODE (2), an equilibrium point is defined as a point satisfying $\nabla f(x) = 0$. Obviously, any equilibrium point of (2) corresponds to a stationary point of (1), and vice verse. Thus, the problem of finding if $x(t)$ converges to an equilibrium point of (2) as $t$ increases becomes a central issue in the continuous method with ODE (2) for problem (1). In the neural network approach for optimization (see [14], a merit function is normally formulated as a companion to ODE (2). One important requirement for this merit function is that the function must be monotonically nonincreasing along the trajectory $x(t)$ of (2) as $t$ increases. With the introduction of the merit function, the convergence of the solution $x(t)$ of (2) can be fully investigated in continuous methods. This result is very attractive theoretically.

The study of gradient-based methods in the form of (2) has been investigated in the literature. In [12], Xia has adopted the same framework for linear programming problems in which $f(x)$ is a convex and quadratic function. In [13], a general methodology is provided for establishing a continuous model which is guaranteed to be globally convergent. Since we are considering a general function $f(x)$ in (1), some of their conditions (say Step 2, 2) cannot be met globally. Therefore, their model is not applicable to the general problem (1). Recently, Han et al. [4] have studied the same approach of (2) for problem (1). Some strong and globally convergent results have been obtained. In this paper, we focus on the numerical aspect of (2) for large-scale optimization problems.

The rest of the paper is organized as follows. In Section 2, a detailed study on the convergence of the solution $x(t)$ of (2) is provided. Some related properties will also be revealed. Section 3 is devoted to the extensive numerical experiment of solving ODE (2) on large-scale optimization problems available in the literature. Finally, some concluding remarks will be drawn in Section 4.

## 2. Convergence of the ODE Solution

The gradient-based method represented by (2) has been investigated in [4] for (1). In the convergence proof of $x(t)$, Barbalat's lemma [11] was used. But the proof is quite long and complicated in [4]. In this paper, a simpler proof is provided. Our proof is based on the LaSalle invariant set theorem under mild assumptions. First, let us make the following assumptions.

ASSUMPTIONS.

(Al) The function $f(x)$ has continuous first-order partial derivatives in $R^n$. In addition, $\nabla f(x)$ is locally Lipschitz continuous.

(A2) For any $x_0$, the level set

$$L(x_0) = \{x \in R^n | f(x) \leqslant f(x_0)\}$$

is bounded.

THEOREM 1. *Under Assumptions* (A1) and (A2), *for arbitrary to* $t_0 \geqslant 0$ *and* $x_0 \in R^n$ *there exists a unique solution* $x(t)$ *of ODE* (2) *satisfying* $x(t_0) = x_0$, $t \in [t_o, +\infty)$.

*Proof.* From Assumption (A2), we know that $L(x_0)$ is a compact set. From Assumption (A1) and the Heine–Borel covering theorem, $\nabla f(x)$ is Lipschitz continuous in $L(x_0)$. Therefore, Theorem 3.1 in [3] ensures that there exists a unique solution $x(t)$ of (2) for any $x_0 \in R^n$. In addition, it is easy to see that $df(x)/dt = -\|\nabla f(x)\|^2 \leqslant 0$. Therefore, any solution $x(t)$ of ODE (2) will stay in $L(x_0)$. Then $x(t)$ can be extended to $[t_0, +\infty)$. □

THEOREM 2. *Let* $x(t)$ *be the solution of* ODE (2) *with* $x(t_0) = x_0$ *and Assumptions* (A1) and (A2) *hold. Then we have the following*:

(i) $\dfrac{df(x(t))}{dt} \leqslant 0, \quad \forall t \geqslant t_0$ *and* $\dfrac{df(x(t))}{dt} = 0 \Leftrightarrow \nabla f(x(t)) = 0.$

(ii) $\lim_{t\to+\infty} \nabla f(x(t)) = 0.$

(iii) If $\nabla f(x_0) \neq 0$, *then* $\nabla f(x(t)) \neq 0$, $\forall t \geqslant t_0$.

*Proof.* (i) From (2), we have

$$\frac{df(x(t))}{dt} = -\|\nabla f(x(t))\|^2 \leqslant 0.$$

Then (i) holds.

(ii) In Theorem 2, if we let $V(x) = f(x)$ and $d(x) = -\nabla f(x)$, then we have $\lim_{t\to+\infty} \nabla f(x(t)) = 0$.

(iii) In Theorem 3, if we let $V(x) = f(x)$ and $d(x) = -\nabla f(x)$, then we have that (iii) is true. □

THEOREM 3. *Let $x(t)$ be the solution of* ODE (2) *with $x(t_0) = x_0$ an Assumptions* (A1) *and* (A2) *hold. Then for any initial point $x_0$, the trajectory $x(t)$ of* (2), *satisfying $x(t_0) = x_0$, will tend to the set of stationary points of* (1) *as $t \to +\infty$.*

*Proof.* First, from Theorem 1 the solution of ODE (2) with $x(t_0) = x_0$ exists and is unique. From Assumption (A2), the level set $L(x_0)$ is bounded. In addition, Theorem 2 (i) ensures that $df(x)/dt \leqslant 0$. Therefore from Theorem 3.4 (local invariant set theorem) [11] with $V(x) = f(x)$ and ODE (2), the trajectory $x(t)$ of (2), satisfying $x(t_0) = x_0$, will tend to the set of stationary points of (1) as $t \to +\infty$. This completes the proof. □

A similar result has been obtained in Theorem 4 [4] under a different condition (no Assumption (A2) but requiring $\nabla f(x)$ being Lipschitz continuous in $R^n$). But our proof here is much shorter than the one in [4].

Noticing that there is no matrix involved in (2), if the solution $x(t)$ of (2) can be solved without using any matrix operation, the proposed continuous method would provide an ideal methodology for large-scale problems.

## 3. Numerical Results

In order to solve the ODE arisen from Section 2, we utilize the ODE solver LSODAR [5, 6] from the ODEPACK package of the netlib libraries. LSODAR solves the initial value problem for stiff or nonstiff systems[1] of first order ODEs of the form

$$\frac{dx(t)}{dt} = F(x(t), t),$$
$$x(0) = \eta_0,$$

where $x(t) = (x^1(t), \ldots, x^n(t))^T$, and at the same time locates the roots of any of a set of functions

$$g_j = g(j, t, x^1, \ldots, x^n), \quad j = 1, \ldots, m_g.$$

This ODE solver is peculiar in that it automatically switches between stiff and nonstiff methods [10], thus relieving the user from determining whether the given problem is stiff or not. The code starts from a nonstiff method. It marches along the time domain $t$ step by step, obtaining a sequence of vector approximations $x_k = (x_k^1, \ldots, x_k^n)^T$ to the exact solution of $x(t)$ at

[1]See [2] for a description of stiff and nonstiff ODEs.

time points $t_1, \ldots, t_k, \ldots$ At the $k$-th step, LSODAR controls the local error $e_k^i$ of $x_k^i$ by ensuring that

$$e_k^i \leqslant rtol|x_k^i| + atol, \quad i = 1, \ldots, n$$

for the user given relative tolerance parameter *rtol* and absolute tolerance parameter *atol*. LSODAR also estimates the step size $h_{k+1} = t_{k+1} - t_k$ for the next step using an estimated local error $e_{k+1}^i$, $i = 1, \ldots, n$. Periodically, if LSODAR determines that switching from a nonstiff method to a stiff method, or vice versa, will give a much bigger estimated $h_{k+1}$ for the next step than the original method, LSODAR will perform the method switch.

A stiff method is more versatile than a nonstiff method in the sense that a stiff method is able to solve stiff problems. However, a stiff method pays a price by requiring the solution of a linear system of the form

$$\left(I - h\alpha_0 \frac{\partial F}{\partial x}\right) y = z \tag{3}$$

in each step, where $\alpha_0$ is a constant. If $\partial F/\partial x$ is dense, $O(n^3)$ operations are required per step to solve the linear system. In addition, the Jacobian matrix $\partial F/\partial x$ takes up $n^2$ storage locations. For some of our test problems where $\partial F/\partial x$ is narrowly banded, improvements can be made and only $O(n)$ operations are required per step for the linear system. The storage needed for $\partial F/\partial x$ in the banded case is also $O(n)$. For comparison, a non-stiff method needs $O(n)$ operations per step, and its storage requirement is also $O(n)$.

LSODAR has an option for the user to specify whether the $\partial F/\partial x$ involved is full or banded, and whether it is user supplied or numerically computed internally. Except for problem B15 in our test pool, all the $\partial F/\partial x$ used in our test problems are internally generated by LSODAR.

Depending on the structure of $\partial F/\partial x$, the solution of the linear system (3) can some times be computed using iterative methods, sparse techniques, matrix-free techniques and etc. Because this paper intends to demonstrate the feasibility of converting an optimization problem to an ODE problem, we do not explore these channels for our test problems.

The root finding capability of LSODAR is suitable for our optimization method in that it allows for an optimization tolerance. For our application, we set $F = -\nabla f(x(t))$, $m_g = 1$, and

$$g_1 = \begin{cases} 1.0 & \text{if } \|\nabla f(x(t))\|_\infty > \delta, \\ 0.0 & \text{if } \|\nabla f(x(t))\|_\infty \leqslant \delta, \end{cases}$$

where $\delta$ is our tolerance for optimization control. We believe that there is some relationship between $\delta$ and the tolerances *rtol* and *atol* of the ODE solver. For a given $\delta$, a well chosen set of *rtol* and *atol* will reduce the computation time of the ODE solver. Such a relationship can be studied in a future paper. For the time being, after trial and error, we have chosen $rtol = atol = \delta$ and our numerical results appear reasonable.

To examine our method in practice, we select all 27 large-scale problems from [1] and [9] (see the Appendix). These test problems are divided into two groups. The first group (problems G1–G12) consists of problems whose Jacobian matrices $\partial F/\partial x$ or Hessian matrices $\nabla^2_{xx}f(x)$ are dense, whereas the second group (problems B1–B15) consists of problems with a banded $\partial F/\partial x$ or $\Delta^2_{xx}f(x)$. Problem G9 of the first group is special in that its Jacobian is actually banded with a bandwidth $n/2$. Because of the relatively wide bandwidth, the banded solver of LSODAR has no benefit in this case and the dense matrix linear solver is used instead. Thus problem G9 is included in the first group. (Admittedly, if LSODAR had an iterative linear solver, this problem would be solved much more efficiently.)

The numerical results are presented in Tables 1 and 2. A Sun Microsystem E25O work-station with two UltraSPARC 2 CPUs running at 400 MHz and 1 G main memory is used. The compiler employed is f77. Each test problem utilizes only a single CPU. For problems where the Jacobian matrix $\partial F/\partial x$ is dense, we limit $n$ to be 8000 so that the resulting process image takes up 490 M of main memory. This process size is about as big as can be supported by our machine without inducing much disk swapping when two such problems are run at the same time. For problems where $\partial F/\partial x$ is narrowly banded, $n$ is taken to be $10^6$.

Our test runs also include nonstiff experiments for each test problem. These experiments are done by modifying LSODAR to disable the method switch mechanism, so that LSODAR always retains the nonstiff method. The nonstiff experiments have their merit in that some test problems are intrinsically nonstiff or mildly stiff within the range of $\delta$ we choose. For problems that are nonstiff or mildly stiff, the nonstiff methods may run faster than the stiff methods because no linear equation is solved in each step. In 7Tables 1 and 2, the label **nonstiff** means running LSODAR with only nonstiff methods, whereas the label **stiff** means letting LSODAR switch between nonstiff and stiff methods automatically. Another reason to perform the nonstiff experiments is that a bigger problem size can sometimes be tested. For some problems where the stiff experiment limits the problem size $n$ to be 8000, we run the nonstiff experiments for both $n = 8000$ and $n = 10^6$. The $n = 10^6$ runs are possible because the nonstiff methods do not require storage for the Jacobian matrix $\partial F/\partial x$.

We have tested each problem with $\delta = 10^{-2}, 10^{-4}, \ldots, 10^{-8}$. If a test does not return within three days, or if LSODAR has completed 60,000 steps in the $t$-domain without satisfying $\|\nabla f(x(t))\|_\infty \leqslant \delta$, the test is terminated. We mark these by an $*$ in Tables 1 and 2.

*Table 1.* Computation times (in seconds) for test problems with dense $\partial F/\partial x$.

| Problem | Method | $n$ | $\delta$ | | | |
|---|---|---|---|---|---|---|
| | | | $10^{-2}$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ |
| G1 | Stiff | 8000 | 2.01 | 6.60 | 129080.00 | 372915.00 |
| | Nonstiff | 8000 | 2.04 | 6.71 | 45.77 | 120.86 |
| | Nonstiff | $10^6$ | 764.61 | 2539.30 | 7128.94 | 15676.33 |
| G2 | Stiff | 8000 | * | * | * | * |
| | Nonstiff | 8000 | 125.45 | * | * | * |
| G3 | Stiff | 8000 | * | * | * | * |
| | Nonstiff | 8000 | 3.05 | 6.06 | 63.61 | * |
| | Nonstiff | $10^6$ | * | * | * | * |
| G4 | Stiff | 8000 | 0.00 | 0.00 | 197937.05 | 657734.44 |
| | Nonstiff | 8000 | 0.00 | 0.00 | 3.48 | 24.40 |
| | Nonstiff | $10^6$ | 0.00 | 0.00 | 196.08 | 708.52 |
| G5 | Stiff | 8000 | 98008.46 | 293524.34 | 637870.31 | * |
| | Nonstiff | 8000 | * | * | * | * |
| | Nonstiff | $10^6$ | * | * | * | * |
| G6 | Stiff | 8000 | 0.30 | 0.93 | 2.27 | 4.76 |
| | Nonstiff | 8000 | 0.30 | 0.99 | 2.51 | 5.27 |
| | Nonstiff | $10^6$ | 78.04 | 267.32 | 670.68 | 1403.01 |
| G7 | Stiff | 8000 | 43569.23 | 87193.72 | 130710.40 | 171871.23 |
| | Nonstiff | 8000 | 0.79 | 2.11 | 4.56 | 8.36 |
| | Nonstiff | $10^6$ | 314.02 | 830.17 | * | * |
| G8 | Stiff | 8000 | 43497.54 | 87011.55 | 130521.93 | 173990.92 |
| | Nonstiff | 8000 | 0.77 | 2.12 | 4.61 | 8.34 |
| | Nonstiff | $10^6$ | 317.35 | 826.80 | * | * |
| G9 | Stiff | 8000 | 8028.10 | 15660.60 | 27264.96 | 34497.54 |
| | Nonstiff | 8000 | 3.95 | 9.82 | 21.80 | 44.23 |
| | Nonstiff | $10^6$ | 578.44 | 1442.80 | 3236.49 | 6648.64 |
| G10 | Stiff | 8000 | 0.00 | 0.00 | * | * |
| | Nostiff | 8000 | 0.00 | 0.00 | 3.53 | * |
| | Nonstiff | $10^6$ | 0.00 | 0.00 | 197.98 | 851.88 |
| G11 | Stiff | 8000 | 70006.52 | 196008.11 | 307896.47 | 433518.66 |
| | Nonstiff | 8000 | 7.54 | 24.93 | 57.89 | 117.55 |
| | Nonstiff | $10^6$ | 12626.28 | 57853.62 | 140071.50 | 249783.34 |
| G12 | Stiff | 8000 | 84603.47 | 211087.11 | 306775.31 | 428840.34 |
| | Nonstiff | 8000 | 31.00 | 556.89 | * | * |
| | Nonstiff | $10^6$ | * | * | * | * |

For problems G4 and Gl0, the initial values of $\|\nabla f(x(t))\|_\infty$ are already smaller than $10^{-4}$. Thus the computation time reported is 0 for both $\delta = 10^{-2}$ and $\delta = 10^{-4}$.

Problems G7 and G8 require special treatment in that $\partial F/\partial x$ is very stiff so that the initial stepsize $h_1$ chosen by LSODAR is of the order $10^{-24}$. The computation is so lengthy that LSODAR does not finish after three days. Our remedy is to use the transformation $z_j = mjx_j$, so that

$$f(z) = \frac{(m+1)(2m+1)}{6m}\left(\sum_{j=1}^{n} z_j\right)^2 - (m+1)\left(\sum_{j=1}^{n} z_j\right) + m$$

*Table 2.* Computation times (in seconds) for test problems with dense $\partial F/\partial x$.

| Problem | Method | $n$ | $\delta$ | | | |
|---|---|---|---|---|---|---|
| | | | $10^{-2}$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ |
| B1 | Stiff | $10^6$ | 344.96 | 978.99 | 2862.04 | 9491.93 |
| | Nonstiff | $10^6$ | 221420.40 | 78893.29 | 183793.59 | * |
| B1 | Stiff | $10^6$ | 304.82 | 1149.88 | 3149.38 | 8130.51 |
| | Nonstiff | $10^6$ | 3661.40 | 133313.00 | * | * |
| B3 | Stiff | $10^6$ | 182.14 | 884.93 | 2882.07 | 8175.06 |
| | Nonstiff | $10^6$ | 1283.51 | * | * | * |
| B4 | Stiff | $10^6$ | 132.59 | 412.71 | 1010.71 | 2098.95 |
| | Nonstiff | $10^6$ | 219.86 | 623.20 | 1367.57 | 2562.01 |
| B5 | Stiff | $10^6$ | 234.20 | 623.45 | 1408.28 | 2782.53 |
| | Nonstiff | $10^6$ | 225.40 | 645.28 | 1443.35 | 2830.59 |
| B6 | Stiff | $10^6$ | 63.63 | 207.58 | 513.54 | 1073.00 |
| | Nonstiff | $10^6$ | 63.94 | 206.63 | 507.97 | 1054.57 |
| B7 | Stiff | $10^6$ | 592.26 | 1891.26 | 5244.41 | 11781.14 |
| | Nonstiff | $10^6$ | 14018.70 | * | * | * |
| B8 | Stiff | $10^6$ | 163.07 | 2451.60 | 6499.99 | 14432.80 |
| | Nonstiff | $10^6$ | 601.35 | * | * | * |
| B9 | Stiff | $10^6$ | 287.88 | 116523.41 | 334475.50 | * |
| | Nonstiff | $10^6$ | 6511.07 | * | * | * |
| B10 | Stiff | $10^6$ | 213.78 | 681.43 | 1645.38 | 3423.48 |
| | Nonstiff | $10^6$ | 246.25 | 681.41 | 1569.26 | 3187.38 |
| B11 | Stiff | $10^6$ | 406.14 | * | * | * |
| | Nonstiff | $10^6$ | 42.51 | * | * | * |
| B12 | Stiff | $10^6$ | 282.62 | 1117.27 | 3157.84 | 8247.62 |
| | Nonstiff | $10^6$ | 387.00 | 3089.74 | * | * |
| B13 | Stiff | $10^6$ | 1008.55 | 2615.83 | 5578.16 | 11292.10 |
| | Nonstiff | $10^6$ | * | * | * | * |
| B14 | Stiff | $10^6$ | 5968.29 | 18298.46 | 40949.78 | 104688.27 |
| | Nonstiff | $10^6$ | * | * | * | * |
| B15 | Stiff | $10^6$ | 2323.79 | 7445.71 | 18823.09 | 43998.01 |
| | Nonstiff | $10^6$ | * | * | * | * |

and

$$f(z) = 2 + \frac{1}{m^2}\left[\frac{(m-1)m(2m-1)}{6} - 1\right]\left(\sum_{j=2}^{n-1} z_j\right)^2 + \frac{(m-2)}{m^2}\left[\left(\sum_{j=2}^{n-1} z_j\right) + m\right]^2$$
$$- \frac{2}{m^2}\left[\frac{(m-1)m}{2} - 1\right]\left[\left(\sum_{j=2}^{n-1} z_j\right)^2 + m\left(\sum_{j=2}^{n-1} z_j\right)\right],$$

respectively for problems G7 and G8.

The solution of problem B15 must also be computed differently because of the term $(\exp(x_{i+1} - \exp(x_i))/(x_{i+1} - x_i)$. This term causes a loss of accuracy of the order $1/(x_{i+1} - x_i)$. Consequently, the Jacobian $\partial F/\partial x$ numerically generated by LSODAR suffers a loss of accuracy of order $(1/(x_{i+1} - x_i))^3$. In fact, we have tried to ask LSODAR to compute problem

B15 like the other problems by internally generating the Jacobian, and we encounter $x_i = NaN$ for $\delta = 10^{-4}$. Our remedy is to supply an accurate $\partial F/\partial x$ (in series form) to LSODAR for this problem. For example, the $(i, i)$-th element of $\partial F/\partial x$ is given by

$$
\begin{aligned}
-\frac{\partial f^2}{\partial x_i^2} &= -\frac{4}{h} - 2h\lambda\left[\frac{2(e^{x_i} - e^{x_{i-1}})}{(x_i - x_{i-1})^3} - \frac{2e^{x_i}}{(x_i - x_{i-1})^2} + \frac{e^{x_i}}{(x_i - x_{i-1})}\right.\\
&\quad \left. + \frac{2(e^{x_{i+1}} - e^{x_i})}{(x_{i+1} - x_i)^3} - \frac{2e^{x_i}}{(x_{i+1} - x_i)^2} - \frac{e^{x_i}}{(x_{i+1} - x_i)}\right]\\
&= -\frac{4}{h} - 2h\lambda\left[e^{x_{i-1}}\left(\frac{1}{3} + \frac{1}{4}(x_i - x_{i-1}) + \frac{1}{10}(x_i - x_{i-1})^2 + \frac{1}{36}(x_i - x_{i-1})^3\right.\right.\\
&\quad \left. + \frac{1}{168}(x_i - x_{i-1})^4 + \frac{1}{960}(x_i - x_{i-1})^5\right)\\
&\quad + e^{x_i}\left(\frac{1}{3} + \frac{1}{12}(x_{i+1} - x_i) + \frac{1}{60}(x_{i+1} - x_i)^2 + \frac{1}{360}(x_{i+1} - x_i)^3\right.\\
&\quad \left.\left. + \frac{1}{2520}(x_{i+1} - x_i)^4 + \frac{1}{20160}(x_{i+1} - x_i)^5\right)\right].
\end{aligned}
$$

The results reported in Table 2 are computed using the above series form of $\partial F/\partial x$.

We consider a problem to be successfully solved if either the nonstiff or stiff method is able to solve it for the four given values of $\delta$.

For the problems with a banded Jacobian, Table 2 shows that except for problems B9 and B11, all test problems are solved for the different values of $\delta$. The nonstiff runs of problems B6 and B10 are slightly faster than the stiff methods, whereas the stiff runs are better in all the other cases.

## 4. Concluding Remarks

In this paper, we have reported the numerical results of a gradient-based continuous methods for 27 large-scale problems. Our results indicate that

(i) For problems with banded Hessian matrices, stiff ODE solvers can be used and they are effective. For the 15 problems B1–B15, stiff ODE solvers can solve 14 of them (except B11), and they either outperform or are competitive to nonstiff ODE solvers.
(ii) For problems with dense Hessian matrices, nonstiff ODE solvers are effective for nonstiff problems. For the 12 problems G1–G12, nonstiff ODE solvers outperform stiff ODE solvers on 11 problems (except G5).

In addition, we have proved that the solution of the ODE tends to the set of stationary points of the corresponding optimization problem globally.

## Appendix: Test Problems

Here, we list all 27 large-scale problems from [1] and [9].

PROBLEM G1. Penalty function I (problem (23) in [9])

$$f(x) = \sum_{i=1}^{n} 10^{-5}(x_i - 1)^2 + \left[\left(\sum_{i=1}^{n} x_i^2\right) - \frac{1}{4}\right]^2,$$

$$[x_0]_i = i.$$

PROBLEM G2. Penalty function II (modification of problem (24) in [9])

$$f(x)=(x_l-0.2)^2+10^{-5}\sum_{i=2}^{n}\left[\exp\left(\frac{x_i}{m}\right)+\exp\left(\frac{x_{i-1}}{m}\right)-y_i\right]^2$$

$$+10^{-5}\sum_{i=n+1}^{2n-1}\left[\exp\left(\frac{x_{i-n+1}}{m}\right)-\exp\left(\frac{-1}{m}\right)\right]^2+\left(\left[\sum_{i=1}^{n}(n-i+1)x_i^2\right]-1\right)^2,$$

$$y_i=\exp\left(\frac{i}{m}\right)+\exp\left(\frac{i-1}{m}\right),\ [x_0]_i=0.5,\ m=\frac{n}{10}.$$

PROBLEM G3. Variable dimensioned function (problem (25) in [9])

$$f(x) = \sum_{i=1}^{n}(x_i - 1)^2 + \left[\sum_{i=1}^{n} i(x_i - 1)\right]^2 + \left[\sum_{i=1}^{n} i(x_i - 1)\right]^4,$$

$$[x_0]_i = 1 - i/n, \quad [x^*]_i = 1, \quad f(x^*) = 0.$$

PROBLEM G4. Trigonometric function (problem (26) in [9])

$$f(x) = \sum_{i=1}^{n}\left[n - \sum_{j=1}^{n} \cos x_j + i(1 - cos\, x_i) - \sin x_i\right]^2,$$

$$[x_0]_i = 1/n, \quad f(x^*) = 0.$$

PROBLEM G5. Brown almost linear function (problem (27) in [9])

$$f(x) = \sum_{i=1}^{n-1}\left[x_i + \sum_{j=1}^{n} x_j - (n+1)\right]^2 + \left[\left(\prod_{i=1}^{n} x_i\right) - 1\right]^2,$$

$$[x_0]_i = 0.5, \quad x^* = (\alpha, \ldots, \alpha, \alpha,^{1-n})^T, \quad f(x^*) = 0,$$

where $\alpha$ satifies

$$n\alpha^n - (n+1)\alpha^{n-1} + 1 = 0$$

in particular

$$\alpha = 1, \quad f(x^*) = 1 \quad \text{at } x^* = (0, \ldots, 0, n+1)^T.$$

PROBLEM G6. Discrete integral equation function (problem (29) in [9])

$$f(x)=\sum_{i=1}^{n}\left[x_i+0.5h[(1-t_i)\sum_{j=1}^{i}t_j(x_j+t_j+1)^3+t_i\sum_{j=i+1}^{n}(1-t_j)(x_j+t_j+1)^3\right]^2,$$

where

$$h=\frac{1}{n+1},\quad t_i=ih,\quad x_0=x_{n+1}=0,$$
$$[x_0]_i=t_i(t_i-1),\quad f(x^*)=0.$$

PROBLEM G7. Linear function-rank 1 (problem (33) in [9], with modified initial values)

$$f(x)=\sum_{i=1}^{m}\left[i\left(\sum_{j=1}^{n}jx_j\right)-1\right]^2(m\geqslant n),$$
$$[x_0]=\frac{1}{i},\qquad f(x^*)=\frac{m(m-1)}{2(2m+1)}$$

any point, where

$$\sum_{j=1}^{n}jx_j=\frac{3}{2m+1}.$$

PROBLEM G8. Linear function-rank 1 with zero columns and rows (problem (34) in [9], with modified initial values)

$$f(x)=2+\sum_{i=2}^{m-1}\left[(i-1)\left(\sum_{j=2}^{n-1}jx_j\right)-1\right]^2\quad(m\geqslant n),$$
$$[x_0]_i=\frac{1}{i},\qquad f(x^*)=\frac{m^2+3m-6}{2(2m-3)}$$

at any point, where

$$\sum_{j=2}^{n-1}jx_j=\frac{3}{2m-3}.$$

PROBLEM G9. Toint's 7-diagonal generalization of the Broyden Tridiagonal function (problem (14) in [1])

$$f(x)=1+\sum_{i=1}^{n}|(3-2x_i)x_i-x_{i-1}-x_{i+1}+1|^p+\sum_{i=1}^{n/2}|x_i+x_{i+n/2}|^p,$$

where $n$ is even

$$p = \frac{7}{3} \quad \text{and} \quad x_0 = x_{n+1} = 0,$$
$$[x_0]_i = -1.$$

PROBLEM G10. A trigonometric function (problem (15) in [1])

$$f(x) = \sum_{i=1}^{n} \left[ n + i - \sum_{j=1}^{n} (a_{ij} \sin x_j + b_{ij} \cos x_j) \right]^2,$$

where

$$a_{ij} = \delta_{ij}, \quad b_{ij} = 1 + i\delta_{ij},$$
$$[x_0]_i = 1/n.$$

PROBLEM G11. A penalty function (problem (18) in [1])

$$f(x) = 1 + \sum_{i=1}^{n} x_i + 10^3 \left( 1 - \sum_{i=1}^{n} 1/x_i \right)^2 + 10^3 \left( 1 - \sum_{i=1}^{n} i/x_i \right)^2,$$
$$[x_0]_i = 1.$$

PROBLEM G12. A generalization of a function due to A. Brown (problem (20) in [1])

$$f(x) = \left[ \sum_{i \in J} (x_i - 3) \right]^2 + \sum_{i \in J} [10^{-4}(x_i - 3)^2 - (x_i - x_{i+1}) + \exp(20(x_i - x_{i+1}))],$$

where $n$ is a multiple of 2 and $J = \{1, 3, 5, \ldots, n-1\}$,

$$x_0 = (0, -1, 0, -1, \ldots, 0, -1), \quad x^* = (3, 3.1498, 3, 3.1498, \ldots, 3, 3.1498)^T.$$

PROBLEM B1. Extended Rosenbrock function (problem (21) in [9])

$$f(x) = \sum_{i=1}^{n} [100(x_{2i} - x_{2i-1})^2 + (1 - x_{2i-1})^2],$$
$$[x_0]_{2i-1} = -1.2, \quad [x_0]_{2i} = 1, \quad [x^*]_i = 1, \quad f(x^*) = 0.$$

PROBLEM B2. Extended Powell singular function (problem (22) in [9])

$$f(x) = \sum_{i=1}^{n} [(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4],$$
$$[x_0]_{4i-3} = 3, \quad [x_0]_{4i-2} = -1, [x_0]_{4i-1} = 0, \quad [x_0]_{4i} = 1,$$
$$[x^*]_i = 0, \quad f(x^*) = 0.$$

PROBLEM B3. Discrete boundary value function (problem (28) in [9])

$$f(x) = \sum_{i=1}^{n} [2x_i - x_{i-l} - x_{i+1} + h^2(x_i + t_i + 1)^3/2]^2,$$

where

$$h = \frac{1}{n+1}, \quad t_i = ih, \quad x_0 = x_{n+1} = 0,$$
$$[x_0]_i = t_i(t_i - 1), \quad f(x^*) = 0.$$

PROBLEM B4. Broyden tridiagonal function (problem (30) in [9])

$$f(x) = \sum_{i=1}^{n} [(3 - 2x_i)x_i - x_{i-1} - 2x_{i+1} + 1]^2,$$

where

$$x_0 = x_{n+1} = 0,$$
$$[x_0]_i = -1, \quad f(x^*) = 0.$$

PROBLEM B5. Broyden banded function (problem (31) in [9])

$$f(x) = \sum_{i=1}^{n} [x_i(2 + 5x_i^2) + 1 - \sum_{j \in J_i} x_j(1 + x_j)]^2,$$

where

$$J_i = \{j : j \neq i, \max(1, i - m_l,) \leqslant j \leqslant \min(n, i + m_u)\}, \quad m_l = 5,$$
$$m_u = 1, [x_0]_i = -1, \quad f(x^*) = 0.$$

PROBLEM B6. Linear function-full rank (problem (32) in [9])

$$f(x) = \sum_{i=1}^{n} \left[ x_i - \frac{2}{m} \left( \sum_{j=1}^{n} x_j \right) - 1 \right]^2 + \sum_{i=n+1}^{m} \left[ \frac{2}{m} \left( \sum_{j=1}^{n} x_j \right) + 1 \right]^2 \quad (m \geq n),$$

$$[x_0]_i = 1, \quad [x^*]_i = -1, \quad f(x^*) = m - n.$$

PROBLEM B7. The Chained Singular function (problem (5) in [1])

$$f(x) = \sum_{i \in J} [(x_i + 10x_{i+1})^2 + 5(x_{i+2} - x_{i+3})^2 + (x_{i+1} - 2x_{i+2})^4$$
$$+ 10(x_i - 10x_{i+3})^4],$$

where $n$ is a multiple of 4, and $J = \{1, 3, 5, \ldots, n - 3\}$,

$$[x_0]_{4i-3} = 3, \quad [x_0]_{4i-2} = -1, \quad [x_0]_{4i-1} = 0, \quad [x^*]_i = 0, \quad f(x^*) = 0.$$

PROBLEM B8. The Generalized Wood function (problem (7) in [1])

$$f(x) = 1 + \sum_{i \in J} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2 + 90(x_{i+3} - x_{i+2}^2)^2$$
$$+ (1 - x_{i+2})^2 + 10(x_{i+1} + x_{i+3} - 2)^2 + 0.1(x_{i+1} - x_{i+3})^2],$$

where $n$ is a multiple of 4, and $J = \{1, 5, 9, \ldots, n-3\}$,

$$x_0 = (-3, -1, -3, -1, -2, 0, -2, 0, \ldots, -2, 0)^T, \quad [x^*]_i = 1, \quad f(x^*) = 1.$$

PROBLEM B9. The Chained Wood function (problem (8) in [1])

$$f(x) = 1 + \sum_{i \in J} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2 + 90(x_{i+3} - x_{i+2}^2)^2$$
$$+ (1 - x_{i+2})^2 + 10(x_{i+1} + x_{i+3} - 2)^2 + 0.1(x_{i+1} - x_{i+3})^2],$$

where $n$ is a multiple of 4, and $J = \{1, 3, 5, \ldots, n-3\}$,

$$x_0 = (-3, -1, -3, -1, -2, 0, -2, 0, \ldots, -2, 0)^T, \quad [x^*]_i = 1, \quad f(x^*) = 1.$$

PROBLEM B10. A generalization of the Broyden Tridiagonal function (problem (10) in [1])

$$f(x) = 1 + \sum_{i=1}^{n} |(3 - 2x_i)x_i - x_{i-1} - x_{i+1} + 1|^p$$

where

$$p = \frac{7}{3}, \quad \text{and} \quad x_0 = x_{n+1} = 0,$$
$$[x_0]_i = -1.$$

PROBLEM B11. A generalization of the Broyden Banded function (problem (12) in [1])

$$f(x) = 1 + \sum_{i=1}^{n} |(2 + 5x_i^2)x_i + 1 + \sum_{j \in J_i} x_j(1 + x_j)|^p$$

where

$$p = \frac{7}{3}, \quad \text{and} \quad J_i = \{j : \max(1, i-5) \le j \le \min(n, i+1)\},$$
$$[x_0]_i = -1.$$

PROBLEM B12. A Generalization of the Cragg and Levy function (problem (17) in [1])

$$f(x) = \sum_{i \in J} [(\exp(x_i) - x_{i+1})^4 + 100(x_{i+1} - x_{i+2})^6$$
$$+ \tan^4(x_{i+2} - x_{i+3}) + x_i^8 + (x_{i+3} - 1)^2],$$

where $n$ is a multiple of 4, and $J = \{1, 5, 9, \ldots, n-3\}$,

$$x_0 = (1, 2, \ldots, 2)^T, \quad x^* = (0, 1, 1, 1, \ldots, 0, 1, 1, 1)^T.$$

PROBLEM B13. An Augmented Lagrangian function for a generalization of Hock and Schittkowski's 80th problem (problem (19) in [1])

$$f(x) = 1 + \sum_{i \in J} \left[ \left( \exp(x_i x_{i+1} x_{i+2} x_{i+3} x_{i+4}) + \frac{1}{2} \rho \left( \left( \sum_{j=0}^{4} x_{i+j}^2 - 10 - \lambda_1 \right)^2 \right. \right. \right.$$
$$\left. \left. \left. + (x_{i+1} x_{i+2} - 5 x_{i+3} x_{i+4} - \lambda_2)^2 + (x_i^3 + x_{i+1}^3 + 1 - \lambda_3)^2 \right) \right) \right],$$

where $n$ is a multiple of 5, and $J = \{1, 6, 11, \ldots, n-4\}$.

Here we choose

$$\rho = 20, \quad \lambda_1 = -0.002008, \quad \lambda_2 = -0.001900 \quad \text{and} \quad \lambda_3 = -0.000261,$$
$$x_0 = (-2, 2, 2, -1, -1, -1, -1, 2, -1, -1, \ldots, -1, -1, 2, -1, -1)^T.$$

PROBLEM B14. A generalization of another function due to A. Brown (problem (21) in [1])

$$f(x) = \sum_{i=1}^{n-1} \left[ (x_i^2)^{(x_{i+1}^2 + 1)} + (x_{i+1}^2)^{(x_i^2 + 1)} \right],$$
$$x_0 = (-1, 1, -1, 1, \ldots, -1, 1)^T, [x^*]_i = 0, \ f(x^*) = 0.$$

PROBLEM B15. The discretization of a variational problem (problem (23) in [1])

$$f(x) = 2 \sum_{i=1}^{n} [x_i (x_i - x_{i+1})]/h + 2\lambda h \sum_{i=0}^{n} [(\exp(x_{i+1} - \exp(x_i))/(x_{i+1} - x_i)],$$

where

$$h = \frac{1}{(n+1)} \quad \text{and} \quad x_0 = x_{n+1} = 0, \quad \lambda = -3.4,$$
$$[x_0]_i = 0.1 i h (1 - i).$$

## References

1. Conn, A.R., Gould, N.I.M. and Toint, P.L. (1998), Testing a class of methods for solving minimization problems with simple bounds on the variables, *Mathematical Computational*, 50, 399–430.

2. Gear, C.W. (1971), *Numerical Initial Value Problems in Ordinary Differential Equation*, Prentice Hall, New Jersey.
3. Hale, J.K. (1969), *Ordinary Differential Equations*, Wiley-Interscience, New York.
4. Han, Q.M., Liao, L.-Z., Qi, H.D. and Qi, L.Q. (2001), Stability analysis of gradient-based neural networks for optimization problems, *Journal of Global Optimization*, 19(4), 363–381.
5. Hiebert, K.L. and Shampine, L.F. (1980), Implicitly defined output points for solutions of ODEs, *Sandia Report* SAND80-0180.
6. Hindmarsh, A.C. (1983), ODEPACK, a systematized collection of ODE solvers, In: R.S. Stepleman et al. (eds), *Scientific Computing*, North-Holland, Amsterdam, 55–84.
7. Hopfield, J.J. (1982), Neural networks and physical systems with emergent collective computational ability, *Proceedings of the National Academic Science*, 79, 2554–2558.
8. Hopfield, J.J. and Tank, D.W. (1985), Neural computation of decisions in optimization problems, *Biological Cybernetics*, 52, 141–152.
9. Moré, J. Garbow, B. and Hillstrom, K. (1981), Testing unconstrained, optimization software, *ACM Transactions on Mathematical Software*, 7, 17–41.
10. Petzold, L.R. (1983), Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations, *SIAM J. Scientific and Statistical Computing*, 4, 136–148.
11. Slotine, J.-J.E. and Li, W. (1991), *Applied Nonlinear Control*, Prentice-Hall, Englewood Cliffs, New Jersey.
12. Xia, Y. (1996), A new neural networks for solving linear programming problems and its application, *IEEE Transactions of Neural Network*, 7, 525–529.
13. Xia, Y. and Wang, J. (1998), A general methodology for designing globally convergent optimization neural networks, *IEEE Transaction of Neural Network*, 9, 1331–1343.
14. Zhang, X.-S. (2000), *Neural Network in Optimization*, Kluwer Academic Publishers, Dordrecht.