Computers and Mathematics with Applications 63 (2012) 816–826

Contents lists available at SciVerse ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa

# Heuristic regularization methods for numerical differentiation

Dinh Nho Hào [a,b,*], La Huu Chuong [a], D. Lesnic [b]

[a] Hanoi Institute of Mathematics, 18 Hoang Quoc Viet Road, Hanoi, Viet Nam

[b] Department of Applied Mathematics, University of Leeds, Leeds LS2 9JT, UK

A R T I C L E I N F O

*Article history:*
Received 19 September 2011
Received in revised form 21 November 2011
Accepted 22 November 2011

*Keywords:*
Numerical differentiation
Ill-posed problem
Spline
*L*-curve method
de Boor method
Heuristic regularization method

A B S T R A C T

In this paper, we use smoothing splines to deal with numerical differentiation. Some heuristic methods for choosing regularization parameters are proposed, including the *L*-curve method and the de Boor method. Numerical experiments are performed to illustrate the efficiency of these methods in comparison with other procedures.

## 1. Introduction

Numerical differentiation is the problem of approximating the derivatives of a certain function. It frequently arises in practice as well as in theory. For example, the inverse problem of calculating engineering loads from strain data requires the second-order differentiation of bending moment of a beam, [1]. Other examples can be found e.g. in [2–6] and the references therein.

Suppose that $f(x)$ is a function defined on $[0, 1]$. Often we obtain the values of $f(x)$ at particular points by measurement which encounters unavoidable errors. Hence let us denote the measured function by $f^\delta(x)$, where the so-called noise level $\delta > 0$ is such that $\|f - f^\delta\| < \delta$, ($\|\cdot\|$ is a certain norm, e.g., the $L^2$-norm). From $f^\delta$, we aim to approximate the derivatives of $f$ of some orders. This problem, which we call numerical differentiation, is well-known to be ill-posed in the sense of Hadamard. This means that a small noise in the measured data, i.e., small $\delta$, may cause intolerable errors in the computed derivatives.

Various techniques have been proposed for stable numerical differentiation (see [2–11] for further references and surveys). We can classify them into two groups:

(1) Methods which are based on knowledge or good estimate of the noise level $\delta$.

(2) Methods which do not require knowledge of $\delta$, but seek to extract this information from the nature of the problem and the data.

Methods in the second group are often called *heuristic methods*. Since the noise level $\delta$ or prior information about the exact solution is not always available, it is pragmatic to develop heuristic methods for choosing regularization parameters. In the literature, there are a number of such methods such as the *L*-curve method [12,13] and the generalized cross validation method (GCV for short) [3] which have been demonstrated to be fairly satisfactory. However, for numerical differentiation,

* Corresponding author at: Hanoi Institute of Mathematics, 18 Hoang Quoc Viet Road, Hanoi, Viet Nam.
*E-mail addresses:* hao@math.ac.vn, H.DinhNho@leeds.ac.uk (D.N. Hào), lahuuchuong@yahoo.com (L.H. Chuong), amt5ld@maths.leeds.ac.uk (D. Lesnic).

0898-1221/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2011.11.047

the use of such methods is rather sparse. To the best of our knowledge, the most comprehensive analysis is the use of GCV in [3].

For numerical differentiation, smoothing splines are recognized to be useful objects. A breakthrough in using smoothing splines was made with the work of Schoenberg and Reinsch in 1960s (see [14] and references therein). They proved that the minimizer of problem (1) (see Section 2) is a spline of order $2k$. After that, there were some papers dealing with the issue of choosing the regularization parameter. One of the most prominent methods is the generalized cross validation method (GCV) which was investigated thoroughly by Craven and Wahba (see [3]). Another popularly cited paper was due to Hanke and Scherzer [4]. In their paper, with the choice $\alpha = \delta^2$, they proved the convergence result for using cubic smoothing spline to approximate derivatives. Following this paper, several authors dealt with higher smoothing splines with the same parameter choice, e.g., [6]. Parallel to the development of using splines is the implementation in Matlab with the Spline Toolbox by de Boor [15]. In this toolbox, the author implemented the procedure for choosing the regularization parameter by the discrepancy principle and remarkably, the heuristic method based on trace balance which we call later the ad hoc procedure, or *the de Boor method*. The users can also find various applications of splines in this toolbox.

In this paper, we are particularly interested in heuristic methods for choosing regularization parameters in smoothing splines. By working with the Matlab program, especially the Spline Toolbox of de Boor [15] and the Regularization Toolbox [12] by Hansen, we see the possibility of using smoothing splines for numerical differentiation with heuristic strategies for choosing the regularization parameters, including the $L$-curve method and the so-called ad hoc procedure (or de Boor method). We implement the $L$-curve method (which makes use of the regularization package of Hansen) for smoothing splines and modify the idea of trace balance of de Boor for higher smoothing splines. We also combine these methods to choose more suitable regularization parameters.

This paper is organized as follows. In Section 2, we formulate the problem and recall some well-known results on smoothing splines. Section 3 illustrates these results by examining the cases when $k = 2$ and 3 and devises some heuristic methods for choosing the regularization parameters, including the $L$-curve method and the de Boor method. Numerical examples are presented in Section 4. In the course of experiments, we find these methods satisfactory and, in some cases, even better than methods using knowledge of the noise level.

## 2. Preliminaries

Let us formulate the problem of smoothing splines and state some previous results (see [6]). Suppose that $y = y(x)$ is a function from $[0, 1]$ into $\mathbb{R}$. Let $n$ be a natural number,

$$\Delta = \{0 = x_0 < x_1 < x_2 < \cdots < x_n = 1\}$$

a grid on $[0, 1]$, and $\delta$ a given constant which indicates the level of noise in the data. Denote step sizes by

$$h_i = x_{i+1} - x_i, \quad i = 0, 1, 2, \dots, n-1.$$

Next, by measurement, we have a number of noisy samples $y_i$ of the values $y(x_i)$ satisfying

$$|y_i - y(x_i)| \le \delta, \quad i = 0, 1, 2, \dots, n.$$

From these samples, we aim to construct approximations of the function, as well as its derivatives of some orders. To this end, we seek for a solution of the variational problem

$$\Phi(f) = \frac{1}{n-1}\sum_{j=1}^{n-1}(y_j - f(x_j))^2 + \alpha\int_0^1 f^{(k)}(x)^2 dx = \min! \tag{1}$$

over the admissible set

$$\Lambda = \left\{f \in H^k(0, 1), f(0) = y_0, f(1) = y_n\right\},$$

where the Sobolev space $H^k(0, 1)$ is defined by (see, [2,3,5,6])

$$H^k(0, 1) = \left\{f \in C^{k-1}[0, 1], f^{(k-1)}(x) = a + \int_0^x \psi(s)ds, a \in \mathbb{R}, \psi \in L^2(0, 1)\right\}.$$

Here $\alpha > 0$, called a regularization parameter, controls the infidelity of the computed function to the data, presented by the term $\frac{1}{n-1}\sum_{j=1}^{n-1}(y_j - f(x_j))^2$ and its roughness, represented by the term $\int_0^1 f^{(k)}(x)^2 dx$. The minimizer, denoted by $f_\alpha$, turns out to be a spline of order $2k$. Its derivatives with respect to some suitable $\alpha$ are then used to approximate the derivatives (up to order $k-1$) of the original function. Observe that, for $\alpha = 0$, the minimizer $f_0$ is the spline that interpolates the data while, as $\alpha \to \infty$, the minimizers approach the straight line that best fits the data.

Sometimes the noise level $\delta$ can be estimated accurately. However, in general, we can neither estimate this quantity nor understand its behavior. This motivates computational methods which do not depend on the knowledge the noise level.

Problem (1) was investigated thoroughly in [6] (see also [4]). It is stated that the minimizer $f_*$ can be given by

$$f_*(x) = c_{j,1} + c_{j,2}x + \cdots + c_{j,2k}x^{2k}, \quad x \in [x_j, x_{j+1}), \tag{2}$$

for $j = 0, 1, \dots, n-1$, where the $2kn$ parameters $c_{j,l}$ are determined by solving the following system of linear equations

$$\begin{cases} f_*^{(i)}(x_j+) - f_*^{(i)}(x_j-) = 0, & \text{for } i = 0, 1, 2, \dots, 2k-2, j = 1, \dots, n-1, \\ f_*^{(2k-1)}(x_j+) - f_*^{(2k-1)}(x_j-) = \dfrac{y_j - f_*(x_j)}{\alpha(-1)^k(n-1)} & \text{for } j = 1, \dots, n-1, \\ f_*^{(k+j)}(0) = f_*^{(k+j)}(1) = 0 & \text{for } j = 0, 1, \dots, k-2, \\ f_*(0) = y_0, f_*(1) = y_n. \end{cases} \tag{3}$$

**Theorem 2.1** (*Wang et al. [6]*). *Suppose that $f_*$ satisfies (2) and (3). Then $f_*$ is a minimizer of $\Phi(f)$ over $\Lambda$. Moreover, $\Phi(f)$ has a unique minimizer.*

**Theorem 2.2** (*Wang et al. [6]*). *Problem (2)–(3) is uniquely solvable.*

## 3. Some heuristic methods for choosing the regularization parameter

To illustrate the theoretical results on smoothing splines, as well as to make the use of some heuristic methods more intuitive, we investigate the special cases when $k = 2$ and 3.

### 3.1. Cubic smoothing splines for approximating the first-order derivative ($k = 2$)

For $k = 2$ the minimizer of problem (1) turns out to be a cubic spline. We adapt the notations used by de Boor [16, pp. 235–239] in describing this spline. The minimizer takes the form

$$f(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \quad x \in [x_i, x_{i+1}), i = 0, 1, \dots, n-1.$$

The smoothness at grid points yields

(i) $c_{i+1} - c_i = 3h_i d_i$;
(ii) $b_{i+1} - b_i = 2h_i c_i + 3h_i^2 d_i$;
(iii) $a_{i+1} - a_i = h_i b_i + h_i^2 c_i + h_i^3 d_i$;

for $i = 0, 1, \dots, n-2$.

By simple calculations, we obtain a linear system for $a_i$ and $c_i$, namely

$$\begin{cases} h_{i-1}c_{i-1} + 2(h_{i-1} + h_i)c_i + c_{i+1}h_{i+1} = 3\left(\dfrac{a_{i-1}}{h_{i-1}} - \left(\dfrac{1}{h_{i-1}} + \dfrac{1}{h_i}\right)a_i + \dfrac{a_{i+1}}{h_i}\right); \\ c_0 = c_n = 0; \quad i = 1, 2, \dots, n-2. \end{cases}$$

Denote the vectors

$$\mathbf{a} = (a_0, a_1, \dots, a_n)^t; \qquad \mathbf{c} = (c_1, c_2, \dots, c_{n-1})^t;$$

and the matrices

$$R = \begin{pmatrix} 2(h_0 + h_1) & h_1 & 0 & \cdots & 0 \\ h_1 & 2(h_1 + h_2) & h_2 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & h_{n-3} & 2(h_{n-3} + h_{n-2}) & h_{n-2} \\ 0 & \cdots & 0 & h_{n-2} & 2(h_{n-2} + h_{n-1}) \end{pmatrix},$$

$$Q = \begin{pmatrix} \left(-\dfrac{1}{h_0} - \dfrac{1}{h_1}\right) & \dfrac{1}{h_1} & 0 & \cdots & 0 \\ \dfrac{1}{h_1} & \left(-\dfrac{1}{h_1} - \dfrac{1}{h_2}\right) & \dfrac{1}{h_2} & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \dfrac{1}{h_{n-3}} & \left(-\dfrac{1}{h_{n-3}} - \dfrac{1}{h_{n-2}}\right) & \dfrac{1}{h_{n-2}} \\ 0 & 0 & \cdots & \dfrac{1}{h_{n-2}} & \left(-\dfrac{1}{h_{n-2}} - \dfrac{1}{h_{n-1}}\right) \\ 0 & 0 & \cdots & 0 & \dfrac{1}{h_{n-1}} \end{pmatrix}.$$

Then

$$R\mathbf{c} = 3Q^t\mathbf{a}.$$

It is easy to see that $R$ is a symmetric positive matrix of order $n-1$ and $Q$ is a tridiagonal matrix of the size $(n+1)\times(n-1)$. In addition, the characteristic of the minimizer gives

$$\begin{cases} c_0 = c_n = 0, \\ d_{i+1} - d_i = \dfrac{1}{6\alpha(n-1)}(y_i - a_i), \quad i = 0, 1, \ldots, n-2. \end{cases}$$

In combination with (i), this yields

$$\frac{c_i}{h_i} - c_{i+1}\left(\frac{1}{h_i} + \frac{1}{h_{i+1}}\right) + \frac{c_{i+2}}{h_{i+1}} = \frac{1}{2\alpha(n-1)}(y_i - a_i)$$

for $i = 0, 1, \ldots, n-2$.

We write these equations in the matrix form

$$2\alpha Q\mathbf{c} = D^{-2}(\mathbf{y} - \mathbf{a}),$$

where $D = \operatorname{diag}(\sqrt{n-1}, \ldots, \sqrt{n-1})$, i.e., the diagonal matrix with $(\sqrt{n-1}, \ldots, \sqrt{n-1})$ as its diagonal, in agreement with notations used by de Boor [16, pp. 235–239]. Also we denote $p = 1/(\alpha+1)$ or,

$$\alpha = \frac{1-p}{p} \quad \text{for } p \in (0, 1]. \tag{4}$$

For convenience, we rewrite the system defining $\mathbf{a}$ and $\mathbf{c}$ as

$$\begin{cases} R\mathbf{c} = 3Q^t\mathbf{a}, \\ 2(1-p)Q\mathbf{c} = pD^{-2}(\mathbf{y} - \mathbf{a}). \end{cases}$$

Substituting $\mathbf{c} = 3R^{-1}Q^t\mathbf{a}$ into the second equation, we get

$$\left(6QR^{-1}Q^t + \frac{p}{1-p}D^{-2}\right)(\mathbf{y} - \mathbf{a}) = 6QR^{-1}Q^t\mathbf{y}. \tag{5}$$

In terms of $\mathbf{c}$, this takes the form

$$\left(6(1-p)Q^tD^2Q + pR\right)\mathbf{c} = 3pQ^t\mathbf{y}. \tag{6}$$

Recall that the variational problem (1) for $k = 2$, and using (4), now becomes

$$p\frac{1}{n-1}\sum_{i=1}^{n-1}(y_i - f(x_i))^2 + (1-p)\int_0^1 f''(x)^2dx = \min!,$$

or,

$$pS(f) + (1-p)T(f) = \min!,$$

where

$$S(f) = \frac{1}{n-1}\sum_{i=1}^{n-1}(y_i - f(x_i))^2, \qquad T(f) = \int_0^1 f''(x)^2dx. \tag{7}$$

Denote the unique solution of this problem corresponding to $p$ by $f_p$. We expect the monotone behavior of $S(f_p)$ and $T(f_p)$, as $p$ varies from $0^+$ to $1^-$, as the standard Tikhonov regularization does. In fact, the following lemma indicates that this is the case.

**Lemma 3.1.** *As p goes from $0^+ \to 1^-$, $S(f_p)$ is monotonically decreasing, while $T(f_p)$ is monotonically increasing.*

**Proof.** The decisive trick is based on the positiveness of $R$. Let $0 < p < q < 1$ and denote by $\mathbf{a}_p$, $\mathbf{a}_q$ the column vectors of $(f_p(x_i))$ and $(f_q(x_i))$ for $i = 0, 1, \ldots, n$. So are $\mathbf{c}_p$, $\mathbf{c}_q$ defined by the equations defining $f_p, f_q$. Then

$$\left(6QR^{-1}Q^t + \frac{p}{1-p}D^{-2}\right)(\mathbf{y} - \mathbf{a}_p) = 6QR^{-1}Q^t\mathbf{y},$$

$$\left(6QR^{-1}Q^t + \frac{q}{1-q}D^{-2}\right)(\mathbf{y} - \mathbf{a}_q) = 6QR^{-1}Q^t\mathbf{y}.$$

Setting $\mathbf{z}_p = \mathbf{y} - \mathbf{a}_p$, $\mathbf{z}_q = \mathbf{y} - \mathbf{a}_q$ and subtracting side by side the equations above yield

$$\left(6QR^{-1}Q^t + \frac{p}{1-p}D^{-2}\right)(\mathbf{z}_p - \mathbf{z}_q) + \frac{p-q}{(1-p)(1-q)}D^{-2}\mathbf{z}_q = \mathbf{0},$$
$$\left(6QR^{-1}Q^t + \frac{p}{1-p}D^{-2}\right)(\mathbf{z}_p - \mathbf{z}_q) = \frac{q-p}{(1-p)(1-q)}D^{-2}\mathbf{z}_q.$$

It follows that

$$(\mathbf{z}_p - \mathbf{z}_q)^t\left(6QR^{-1}Q^t + \frac{p}{1-p}D^{-2}\right)(\mathbf{z}_p - \mathbf{z}_q) = \frac{q-p}{(1-p)(1-q)}(\mathbf{z}_p - \mathbf{z}_q)^t D^{-2}\mathbf{z}_q.$$

Note that $6QR^{-1}Q^t + \frac{p}{1-p}D^{-2}$ is positive definite. Since $q - p > 0$, we have

$$(\mathbf{z}_p - \mathbf{z}_q)^t D^{-2}\mathbf{z}_q \geq 0,$$

or,

$$\mathbf{z}_p^t D^{-2}\mathbf{z}_q \geq \mathbf{z}_q^t D^{-2}\mathbf{z}_q.$$

Applying the Schwarz inequality, we conclude that

$$\mathbf{z}_p^t D^{-2}\mathbf{z}_p \geq \mathbf{z}_q^t D^{-2}\mathbf{z}_q.$$

This is exactly the inequality $S(f_p) \geq S(f_q)$.

The second part follows straightforward from the first one. In fact, since $f_p$ minimizes the functional $pS(f) + (1-p)T(f)$, we have

$$pS(f_p) + (1-p)T(f_p) \leq pS(f_q) + (1-p)T(f_q).$$

It follows that

$$p(S(f_p) - S(f_q)) \leq (1-p)(T(f_q) - T(f_p)).$$

Therefore, $T(f_p) - T(f_q) \leq 0$ since $S(f_p) - S(f_q) \geq 0$ by the first part. This completes the proof. □

**Lemma 3.2.** *In terms of* $\mathbf{c}$, *the quantities* $S(f_p)$ *and* $T(f_p)$ *have the representations*

$$S(f_p) = 4\left(\frac{1-p}{p}\right)^2 \mathbf{c}^t Q^t D^2 Q\mathbf{c}; \qquad T(f_p) = \frac{1}{6}\mathbf{c}^t R\mathbf{c}. \tag{8}$$

**Proof.** Recall that

$$2\alpha Q\mathbf{c} = D^{-2}(\mathbf{y} - \mathbf{a}) \quad \text{or,} \quad \mathbf{y} - \mathbf{a} = 2\alpha D^2 Q\mathbf{c}.$$

Hence

$$S(f_p) = (\mathbf{y} - \mathbf{a})^t D^{-2}(\mathbf{y} - \mathbf{a}) = 4\alpha^2 \mathbf{c}^t Q^t D^2 Q\mathbf{c}.$$

For any straight line $l$ defined on $[0, h]$, i.e., $l(\lambda) = l(0) + (l(h) - l(0))\lambda/h$, for $\lambda \in [0, h]$, we have

$$\int_0^h l(\lambda)^2 d\lambda = \frac{h}{3}\left(l(0)^2 + l(0)l(h) + l(h)^2\right).$$

Therefore,

$$T(f_p) = \sum_{j=1}^{n}\int_{x_{j-1}}^{x_j} f_p''(x)^2 dx = \frac{1}{3}\sum_{j=0}^{n-1} h_j(c_j^2 + c_j c_{j+1} + c_{j+1}^2).$$

It can be immediately verified that

$$T(f_p) = \frac{1}{6}\mathbf{c}^t R\mathbf{c}.$$

Substituting $\alpha = \frac{1-p}{p}$ yields the desired conclusions. □

At this moment, note that for $p = 0$, the minimizer $f_0$ is the straight line that best fits the data; for $p = 1$ the minimizer $f_1$ is the spline that interpolates the data. We should choose $p$ inside $(0, 1)$ in order to seek for a balance between the infidelity and the roughness of computed solutions. In other words, it is a compromise between $S(f_p)$ and $T(f_p)$. Lemma 3.1 gives a possibility for using the $L$-curve method for choosing $p$. (See [12,13] for a thorough analysis and perspective.)

Another idea was due to de Boor in *Spline Toolbox for Use with Matlab* [15], code *csaps*. In view of Eq. (6), he suggested to choose $p$ such that

$$pTrace(R) = 6(1-p)Trace(Q^tD^2Q),$$

where, for a matrix $M$, $Trace(M)$ is the sum of all entries on the diagonal of $M$. It, again, can be understood as an attempt to mediate between $T(f_p)$ and $S(f_p)$. Indeed, these quantities have close relationships, as observed in Lemma 3.2. An advantage of this choice is that it is cheap in the sense of computational cost. In contrast, the implementation of the $L$-curve method requires computing the curvature of the curve $(S(f_p), T(f_p))$ on a log–log scale. It is more complicated and computationally expensive as well.

## 3.2. Smoothing splines for approximating the second-order derivative ($k = 3$)

In order to use smoothing splines to approximate high-order derivatives, we naturally raise their order. In this section, we deal with the problem of approximation of the second-order derivative. We choose $k = 3$ and proceed as follows (see also [6]).

For simple calculation, we use the uniform grid on [0, 1], i.e.,

$$0 = x_0 < x_1 < x_2 < \cdots < x_n = 1, \qquad x_i = i/n, \quad i = 0, 1, \ldots, n; h = 1/n.$$

The minimizer of the functional can be constructed piecewisely by

$$f(x) = a_i + b_i(x-x_i) + c_i(x-x_i)^2 + d_i(x-x_i)^3 + e_i(x-x_i)^4 + f_i(x-x_i)^5, \quad x \in [x_i, x_{i+1}), i = 0, 1, \ldots, n-1.$$

Similar to Section 3.1, we obtain the linear system for **a**, **c**, **e** (see [6]):

$$\begin{cases} J\mathbf{c} = 2h^2K\mathbf{e}, \\ \mathbf{z} + G\mathbf{a} + \dfrac{h^4}{15}H\mathbf{e} = \dfrac{h^2}{3}K\mathbf{c}, \\ \dfrac{24\alpha}{h^2}G\mathbf{e} = \mathbf{a} - \mathbf{y}, \end{cases}$$

where

$$J = \begin{pmatrix} -1 & 1 & 0 & \cdots \\ 1 & -2 & 1 & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & 1 & -2 & 1 \\ \cdots & 0 & 1 & -1 \end{pmatrix}, \qquad K = \begin{pmatrix} 5 & 1 & 0 & \cdots \\ 1 & 4 & 1 & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & 1 & 4 & 1 \\ \cdots & 0 & 1 & 5 \end{pmatrix},$$

$$G = \begin{pmatrix} -2 & 1 & 0 & \cdots \\ 1 & -2 & 1 & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & 1 & -2 & 1 \\ \cdots & 0 & 1 & -2 \end{pmatrix}, \qquad H = \begin{pmatrix} 26 & 7 & 0 & \cdots \\ 7 & 16 & 7 & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & 7 & 16 & 7 \\ \cdots & 0 & 7 & 26 \end{pmatrix},$$

$$\mathbf{a} = (a_1, a_2, \ldots, a_{n-1})^t, \qquad \mathbf{c} = (c_1, c_2, \ldots, c_{n-1})^t, \qquad \mathbf{e} = (e_1, e_2, \ldots, e_{n-1})^t,$$
$$\mathbf{y} = (y_1, y_2, \ldots, y_{n-1})^t, \qquad \mathbf{z} = (y_0, 0, \ldots, 0, y_n)^t.$$

By simple arguments, we can show that $-J$, $-G$ are nonnegative and $K$, $H$ are positive. Set

$$A = 2h^4K - \frac{h^4}{5}JK^{-1}H - \frac{72\alpha}{h^2}JK^{-1}G^2, \qquad \mathbf{B} = 3(JK^{-1}G\mathbf{y} + JK^{-1}\mathbf{z}).$$

Then

$$\mathbf{e} = A^{-1}\mathbf{B}, \qquad \mathbf{c} = \frac{3}{h^2}K^{-1}\left(\mathbf{z} + G\mathbf{a} + \frac{h^4}{15}H\mathbf{e}\right), \qquad \mathbf{a} = \mathbf{y} + \frac{24\alpha}{h^2}G\mathbf{e}.$$

The variational problem (1) for $k = 3$, and using (4), now has the form

$$p\frac{1}{n-1}\sum_{i=1}^{n-1}(y_i - f(x_i))^2 + (1-p)\int_0^1 f'''(x)^2dx = \min!$$

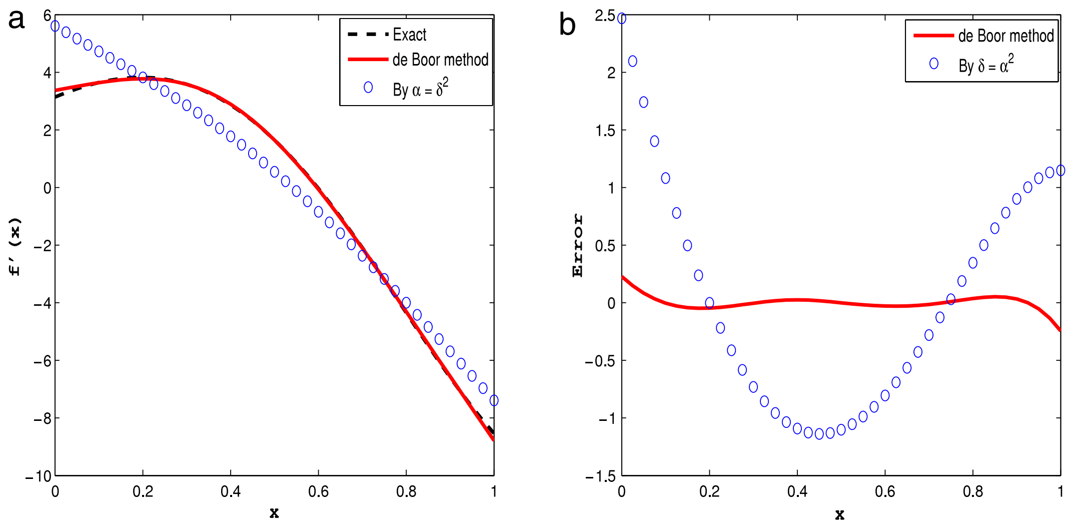

**Fig. 1.** (a) The exact and the computed first-order derivatives and (b) the error between the numerical and the exact first-order derivatives.

or,

$$pS(f_p) + (1-p)T_1(f_p) = \min!,$$

where $T_1(f) = \int_0^1 f'''(x)^2 dx$.

The following lemma can be proved in the same way as in the proof of Lemma 3.1.

**Lemma 3.3.** *As p goes from* $0^+ \to 1^-$, $S(f_p)$ *is monotonically decreasing, while* $T_1(f_p)$ *is monotonically increasing.*

The crucial part of regularization is to choose an appropriate regularization parameter $p$. Similar to the previous section, we can implement the $L$-curve method to obtain a suitable $p$. Experiments, however, show that it is likely to undersmooth, i.e., the selected $p$ is too close to 1.

Motivated by the idea of de Boor in [15], we choose $p$ as follows.

Rewrite the equation for $\mathbf{e}$

$$\left(ph^4 A_1 + \frac{1}{h^2}(1-p)A_2\right)\mathbf{e} = p\mathbf{B}$$

with

$$A_1 = 2K - \frac{1}{5}JK^{-1}H; \qquad A_2 = 72JK^{-1}G^2.$$

The regularization parameter is then obtained so that

$$ph^4 Trace(A_1) = \frac{(1-p)}{h^2} Trace(A_2),$$

or, explicitly,

$$p = 1/(1 + h^6 Trace(A_1)/Trace(A_2)).$$

This strategy, again, is likely to undersmooth the solution since the term $h^6$, which decays rapidly as $n$ increases, will give $p$ too close to 1. Indeed, in our experience, this $p$ is good for small noise levels (typically $\leq 10^{-3}$ or 0.1%), but it undersmooths the computed solutions if the magnitude of errors is larger. As a refinement, we suggest to choose $p$ by

$$p = 1/(1 + h^4 Trace(A_1)/Trace(A_2)).$$

This choice, which we call the de Boor method, keeps $p$ not too close to 1. Numerical results show that it can be a good candidate for the regularization parameter.

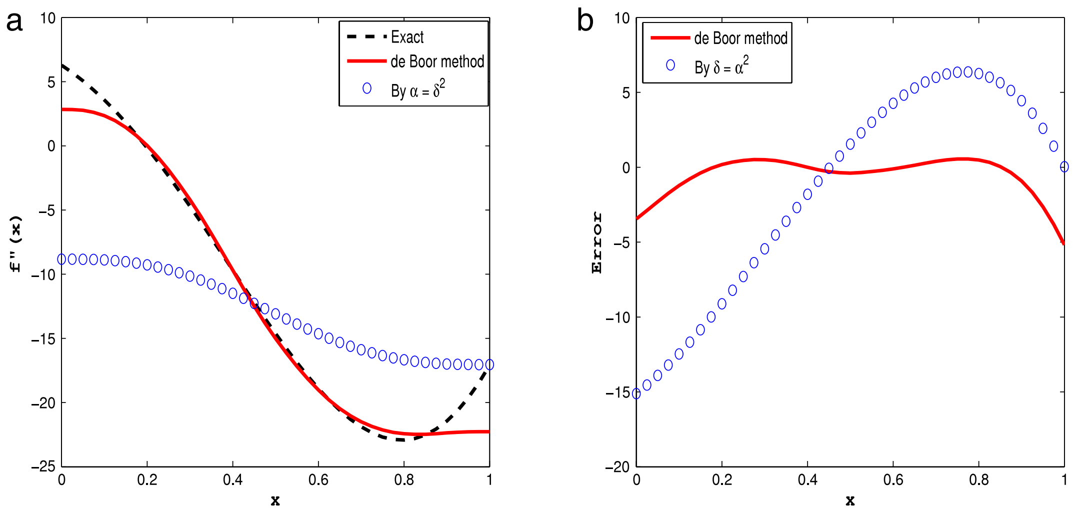

**Fig. 2.** (a) The exact and the computed second-order derivatives, and (b) the error between the numerical and the exact second-order derivatives.

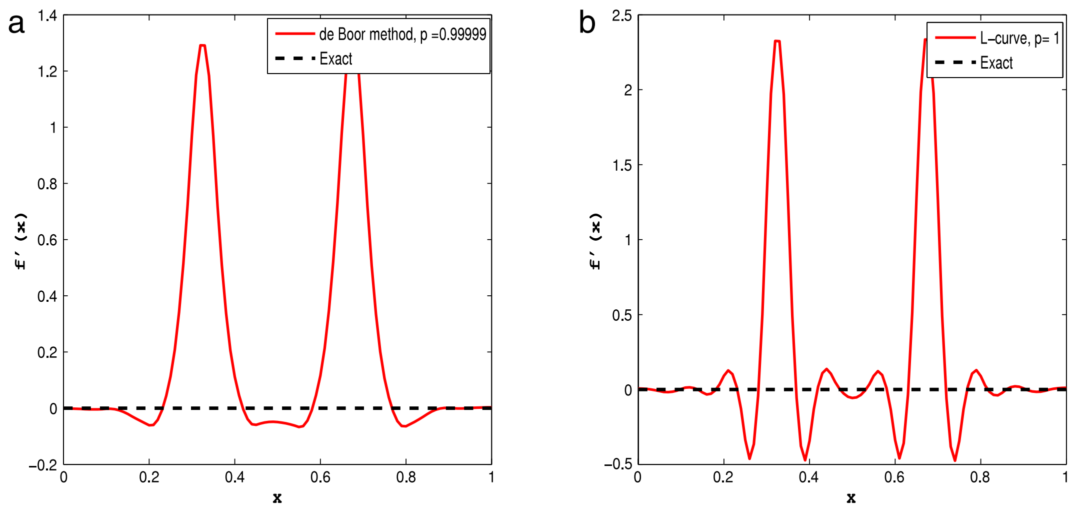

**Fig. 3.** The exact and the computed first-order derivatives using (a) the de Boor method, and (b) the *L*-curve method, in the case $a = 0.9$; $b = 1.0$; $c = 1.1$.

## 4. Numerical results

The following examples are executed in Matlab. The grid size is $n = 20$. Note that for implementation of the de Boor method, $n$ should not be large, otherwise $p$ is nearly identical to 1. To implement the *L*-curve method, we make use of the Regularization Toolbox by Hansen [12] which is available in the MathWorks.com. We use some strategies to choose the regularization parameter including the *L*-curve method [12,13]; the generalized cross validation method [3]; the method based on balance of traces suggested by de Boor which we called the ad hoc procedure, or the de Boor method [15, Sections 2.2–2.3]; the choice $\alpha = \delta^2$ [4,6]; the discrepancy principle (see code *csaps* in Spline Toolbox [15]). In our context, by the discrepancy principle, we understand the strategy which chooses $p$ such that $S(f_p) = \delta^2$. While the first three methods do not require knowledge of the noise level, the last two depend on a good estimate of $\delta$. Convergence rates, as $n \to \infty$ and $\delta \to 0$, of the last method were examined in [4,6].

**Example 4.1** (Figs. 1 *and* 2). We approximate the first and second derivatives of the function

$$f(x) = \exp(x)\sin(\pi x), \quad x \in [0, 1].$$

Here the step size is $h = 1/n = 1/20$. Noise is added to the exact data through the *rand* function in Matlab as $er = 10^{-2} * rand(size(\mathbf{x}))$ which yields the noise level $\delta = 10^{-2}$, where $\mathbf{x}$ is the vector of grid points. We use both the de Boor method and the one based on taking $\alpha = \delta^2$ as in [4,6]. Although the information about the noise level is not used, the former method appears to be more satisfactory than the latter. The results can be observed in Figs. 1 and 2.

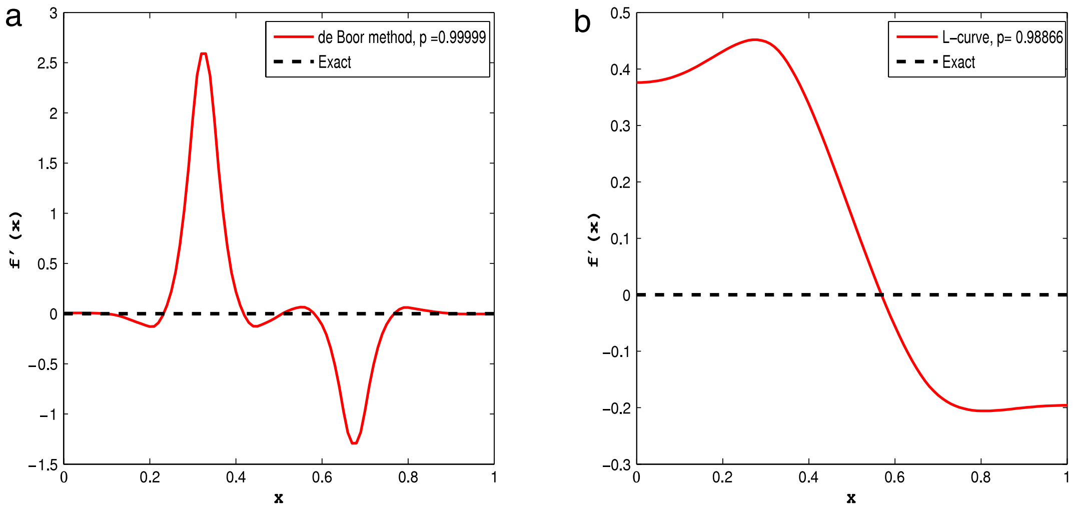

**Fig. 4.** The exact and the computed second-order derivatives using (a) the de Boor method, and (b) the *L*-curve method, in the case $a = 0.9$; $b = 1.1$; $c = 1.0$.

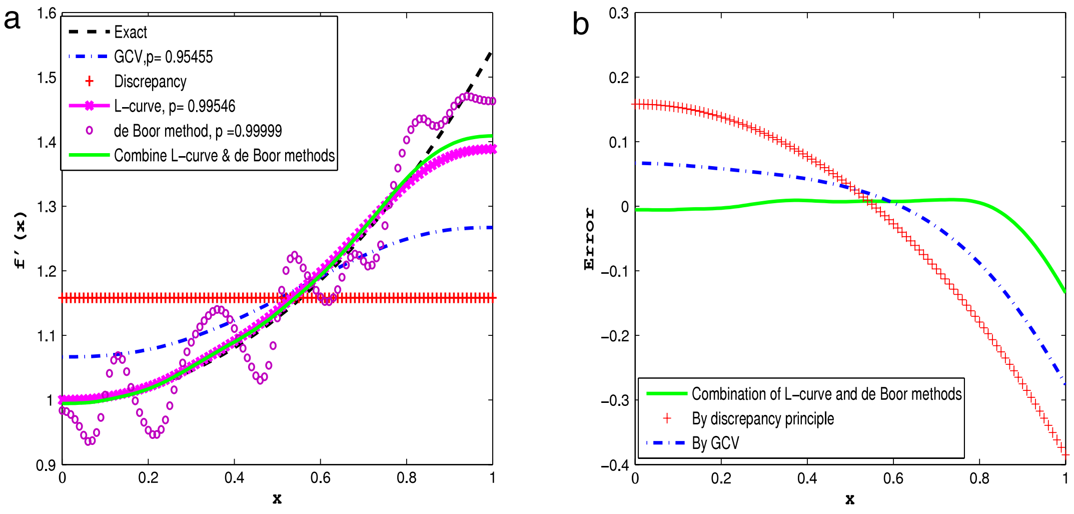

**Fig. 5.** (a) Computed first-order derivatives of the function $f(x) = \sinh(x)$ using various methods, and (b) the error between the numerical and the exact first-order derivatives. The noise level is $\delta = 10^{-2}$.

**Example 4.2** (*Detection of Discontinuity*, Figs. 3 *and* 4)**.** Consider the following function:

$$f(x) = \begin{cases} a & \text{if } 0 \le x < 1/3, \\ b & \text{if } 1/3 \le x < 2/3, \\ c & \text{if } 2/3 \le x \le 1. \end{cases} \tag{9}$$

We test for two cases, namely, $a = 0.9$; $b = 1.0$; $c = 1.1$, and $a = 0.9$; $b = 1.1$; $c = 1.0$. These are discontinuous functions whose jumps are different in both values and senses. As in [5], we aim to detect points of discontinuity of the function via approximations of its (first-order) derivative. If the graph of the derivative clearly reveals some peaks, this may indicate points of discontinuity. In our experiments, the noise level is $\delta = 10^{-3}$, i.e., 1% the jumps at discontinuous points.

The *L*-curve method, in our experience, seems to undersmooth the computed solutions as it chooses $p$ almost identical to 1. This is consistent with the general analysis presented in [11]. As a result, its use for detecting discontinuous points is less favorable; see Figs. 3(b) and 4(b). In contrast, the strategy suggested by de Boor, which we call the de Boor method or the ad hoc procedure, yields more satisfactory results; see Figs. 3(a) and 4(a).

In dealing with functions like the one in case $a = 0.9$; $b = 1.1$; $c = 1.0$, the *L*-curve method even fails; see Fig. 4(b).

The last two examples (see Figs. 5–7) are devoted to a comparison of some strategies for choosing the regularization parameter.

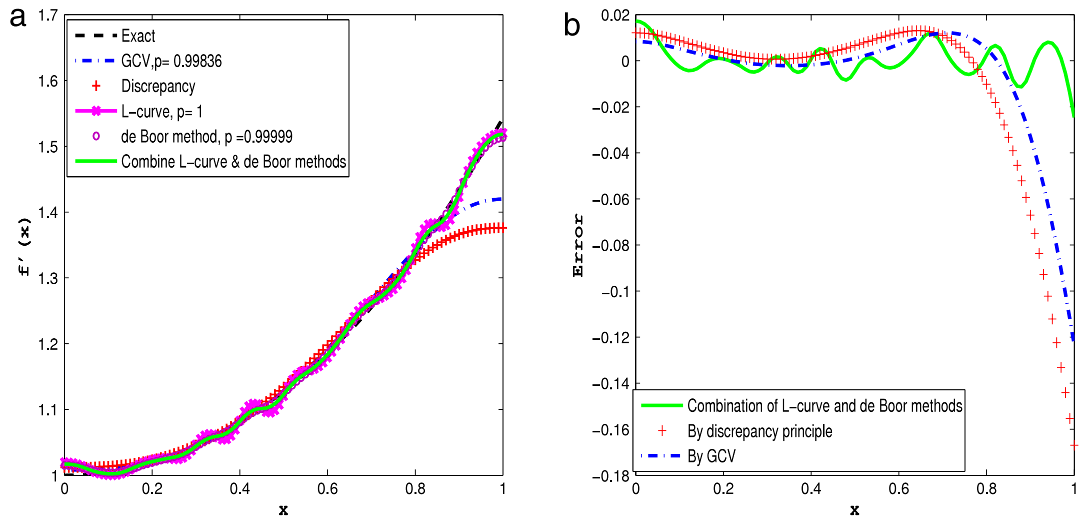

**Fig. 6.** The same as Fig. 5, but the noise level is $\delta = 10^{-3}$.

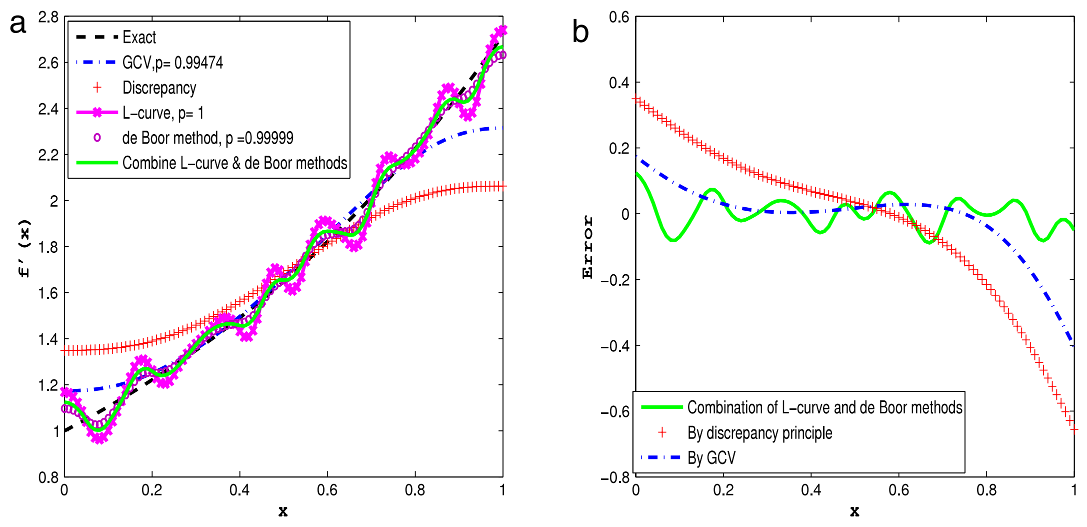

**Fig. 7.** The same as Fig. 5, but for the function $f(x) = \exp(x)$.

**Example 4.3** (Figs. 5 *and* 6)**.** We approximate the first-order derivative of the function $f(x) = \sinh(x)$ in the interval [0, 1]. The noise levels are $\delta = 10^{-2}$ in Fig. 5 and $\delta = 10^{-3}$ in Fig. 6.

**Example 4.4** (Fig. 7). Similar to Example 4.3, but for $f(x) = \exp(x)$ in the interval [0, 1]. The noise level is $\delta = 10^{-2}$.

The regularization parameters produced by the $L$-curve method $p_1$ and the de Boor method $p_2$ sometimes seem to be too close to 1 alternately. Therefore, the computed solutions, although close to the exact ones, are less smooth compared with other methods. So we may think about the possibility to compromise between the two strategies in order to get a more appropriate regularization parameter $p$. Here we simply choose $p = (p_1 + p_2)/2$. This can produce favorable solutions in most cases. In Figs. 5–7, the solid line represents the error of the solution with respect to this combination.

Finally, it is worth noting that the computed solutions in the presented examples display a mismatch at the end points in comparison with the exact solutions. This is due to the constraints on the end conditions in the formulation of problem (1).

## 5. Conclusion

In this paper, we use smoothing splines to approximate the derivatives of functions from their discrete data. We also go a further step in using computed derivatives to predict the discontinuity. The main goal was to investigate some methods for choosing regularization parameters which do not require knowledge of the noise level. Inspired by the idea of the $L$-curve by Hansen and the balance of traces by de Boor, we suggested some useful strategies named the de Boor method (or the ad hoc procedure) and the $L$-curve method. Numerical experiments are performed for the approximations of the first and second derivatives. In many cases, they yield very good results, even better than some methods that use knowledge of the noise level. Still there are examples incompatible with these choices. We believe that these methods can be improved and a more rigorous analysis can be carried out in the future.

## Acknowledgments

This research was supported by a Marie Curie International Incoming Fellowship within the 7th European Community Framework Programme and partially, by the Vietnam NAFOSTED Grant 101.01.22.09.

## References

[1] W.D. Collins, J. Kanval, S. Quegan, D. Smith, D.A.W. Taylor, A regularization method for calculating engineering loads from strain data, Inverse Probl. Eng. 4 (1997) 271–294.
[2] J. Cheng, X.Z. Jia, Y.B. Wang, Numerical differentiation and its applications, Inverse Probl. Sci. Eng. 15 (4) (2007) 339–357.
[3] P. Craven, G. Wahba, Smoothing noisy data with spline functions, Numer. Math. 31 (1979) 377–403.
[4] M. Hanke, O. Scherzer, Inverse problems light: numerical differentiation, Amer. Math. Monthly 108 (6) (2001) 512–521.
[5] Y.B. Wang, X.Z. Jia, J. Cheng, A numerical differentiation method and its application to reconstruction of discontinuity, Inverse Problems 18 (2002) 1461–1476.
[6] Y.B. Wang, Y.C. Hon, J. Cheng, Reconstruction of high order derivatives from input data, J. Inverse Ill-Posed Probl. 14 (2) (2009) 205–218.
[7] Dinh Nho Hào, A mollification method for ill-posed problems, Numer. Math. 68 (4) (1994) 469–506.
[8] Dinh Nho Hào, H.-J. Reinhardt, F. Seiffarth, Stable numerical fractional differentiation by mollification, Numer. Funct. Anal. Optim. 15 (5–6) (1994) 635–659.
[9] Dinh Nho Hào, H.-J. Reinhardt, A. Schneider, Stable approximation of fractional derivatives of rough functions, BIT 35 (4) (1995) 488–503.
[10] A. Kirsch, An Introduction to the Mathematical Theory of Inverse Problems, Springer-Verlag, New York, 1996.
[11] C.R. Vogel, Computational methods for inverse problems, in: Frontiers in Applied Mathematics, SIAM, Philadelphia, 2002.
[12] P.C. Hansen, Regularization tools, a matlab package for analysis and solution of discrete ill-posed problems, Version 4.1 for Matlab 7.3, 2008.
[13] P.C. Hansen, Analysis of discrete ill-posed problems by means of the $L$-curve, SIAM Rev. 34 (1992) 849–872.
[14] C.H. Reinsch, Smoothing by spline functions, Numer. Math. 10 (1967) 177–183.
[15] C. de Boor, Spline toolbox for use with matlab, The MathWorks, 1992.
[16] C. de Boor, A Practical Guide to Splines, Springer-Verlag, Berlin, 1978.