J. Appl. Math. Comput. (2018) 56:131–149
https://doi.org/10.1007/s12190-016-1065-0

ORIGINAL RESEARCH

Numerical comparisons based on four smoothing functions for absolute value equation

B. Saheya[1] · **Cheng-He Yu**[2] · **Jein-Shan Chen**[2]

Received: 24 March 2016 / Published online: 26 October 2016

Abstract The system of absolute value equation, denoted by AVE, is a non-differentiable NP-hard problem. Many approaches have been proposed during the past decade and most of them focus on reformulating it as complementarity problem and then solve it accordingly. Another approach is to recast the AVE as a system of nonsmooth equations and then tackle with the nonsmooth equations. In this paper, we follow this path. In particular, we rewrite it as a system of smooth equations and propose four new smoothing functions along with a smoothing-type algorithm to solve the system of equations. The main contribution of this paper focuses on numerical comparisons which suggest a better choice of smoothing function along with the smoothing-type algorithm.

Keywords Smoothing function · Smoothing algorithm · Singular value · Convergence

Mathematics Subject Classification 26B05 · 26B35 · 65K05 · 90C33

✉ Jein-Shan Chen
jschen@math.ntnu.edu.tw

B. Saheya
saheya@imnu.edu.cn

Cheng-He Yu
60240031S@ntnu.edu.tw

1 College of Mathematical Science, Inner Mongolia Normal University, Hohhot 010022, Inner Mongolia, People's Republic of China

2 Department of Mathematics, National Taiwan Normal University, Taipei 11677, Taiwan

1 Introduction

The absolute value equation (AVE) is in the form of

$$Ax + B|x| = b, \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times n}$, $B \neq 0$, and $b \in \mathbb{R}^n$. Here $|x|$ means the componentwise absolute value of vector $x \in \mathbb{R}^n$. When $B = -I$, where I is the identity matrix, the AVE (1) reduces to the special form:

$$Ax - |x| = b. \tag{2}$$

It is known that the AVE (1) was first introduced by Rohn in [20] and recently has been investigated by many researchers, for example, Hu and Huang [8], Jiang and Zhang [9], Ketabchi and Moosaei [10], Mangasarian [11–15], Mangasarian and Meyer [16], Prokopyev [17], and Rohn [22].

In particular, Mangasarian and Meyer [16] show that the AVE (1) is equivalent to the bilinear program, the generalized LCP (linear complementarity problem), and the standard LCP provided 1 is not an eigenvalue of A. With these equivalent reformulations, they also show that the AVE (1) is NP-hard in its general form and provide existence results. Prokopyev [17] further improves the above equivalence which indicates that the AVE (1) can be equivalently recast as LCP without any assumption on A and B, and also provides a relationship with mixed integer programming. In general, if solvable, the AVE (1) can have either unique solution or multiple (e.g., exponentially many) solutions. Indeed, various sufficiency conditions on solvability and non-solvability of the AVE (1) with unique and multiple solutions are discussed in [16,17,21]. Some variants of the AVE, like the AVE associated with second-order cone and the absolute value programs (AVP), are investigated in [5] and [23], respectively.

As for its numerical solvers, many numerical methods for solving the AVEs (1)–(2) have been proposed. A parametric successive linearization algorithm for the AVE (1) that terminates at a point satisfying necessary optimality conditions is studied in [12]. The generalized Newton algorithm for the AVE (2) is investigated in [13], in which it was proved that this algorithm converges linearly from any starting point to the unique solution of the AVE (2) under the condition that $\|A^{-1}\| < \frac{1}{4}$. The generalized Newton algorithm with semismooth and smoothing Newton steps combined into the algorithm is considered in [24]. The smoothing-type algorithms for solving the AVEs (1)–(2) are studied in [1,8,9]. A branch and bound method for the AVP, which is an extension of the AVE, is studied in [23].

Among the aforementioned approaches, many of them focus on reformulating it as complementarity problem and then solve it accordingly. An alternative approach is to recast the AVE as a system of nonsmooth equations and then tackle with the nonsmooth equations by applying nonsmooth Newton algorithm [18] or smoothing Newton algorithm [19]. In this paper, we follow the latter pathway. More specifically, we rewrite it as a system of smooth equations and propose four new smoothing functions along with a smoothing-type algorithm to solve the system of equations. To see this, motivated by the approach in [1,9], we define $H_i : \mathbb{R}^{n+1} \to \mathbb{R}^{n+1}$ as

$$H_i(\mu, x) = \begin{bmatrix} \mu \\ Ax + B\Phi_i(\mu, x) - b \end{bmatrix} \quad \text{for } \mu \in \mathbb{R} \text{ and } x \in \mathbb{R}^n \tag{3}$$

where $\Phi_i : \mathbb{R}^{n+1} \to \mathbb{R}^n$ is given by

$$\Phi_i(\mu, x) := \begin{bmatrix} \phi_i(\mu, x_1) \\ \phi_i(\mu, x_2) \\ \vdots \\ \phi_i(\mu, x_n) \end{bmatrix} \quad \text{for } \mu \in \mathbb{R} \text{ and } x \in \mathbb{R}^n \tag{4}$$

with four various smoothing functions $\phi_i : \mathbb{R}^2 \to \mathbb{R}$ that will be introduced later. The role of ϕ_i looks similar to the function ϕ_p used in [9]. However, they are substantially different. More specifically, the function ϕ_p employed in [9] is strongly semismooth on $\mathbb{R}^2$, whereas each ϕ_i proposed in this paper is continuously differentiable on $\mathbb{R}^2$. Now, we present the exact form for each function ϕ_i, which is defined as below:

$$\phi_1(\mu, t) = \mu \left[\ln\left(1 + e^{-\frac{t}{\mu}}\right) + \ln\left(1 + e^{\frac{t}{\mu}}\right) \right] \tag{5}$$

$$\phi_2(\mu, t) = \begin{cases} t & \text{if } t \geq \frac{\mu}{2}, \\ \dfrac{t^2}{\mu} + \dfrac{\mu}{4} & \text{if } -\frac{\mu}{2} < t < \frac{\mu}{2}, \\ -t & \text{if } t \leq -\frac{\mu}{2}. \end{cases} \tag{6}$$

$$\phi_3(\mu, t) = \sqrt{4\mu^2 + t^2} \tag{7}$$

$$\phi_4(\mu, t) = \begin{cases} \dfrac{t^2}{2\mu} & \text{if } |t| \leq \mu, \\ |t| - \frac{\mu}{2} & \text{if } |t| > \mu. \end{cases} \tag{8}$$

Some of the smoothing functions have appeared in other contexts for other optimization problems, but they are all novel ones for dealing with the AVE (1). The main idea in this paper is showing that the AVE (1) has a solution if and only if $H_i(\mu, x) = 0$, ϕ_i is continuously differentiable at any $(\mu, t) \in \mathbb{R}_{++} \times \mathbb{R}$, and $\lim_{\mu \downarrow 0} \phi_i(\mu, x) = |x|$. Then, with these four new smoothing functions, we consider the smoothing-type algorithm studied in [7,25] to solve the AVE (1). In other words, we reformulate the AVE (1) as parameterized smooth equations and then employ a smoothing-type algorithm to solve it. In addition, we show that the algorithm is well-defined under the assumption that the minimal singular value of the matrix A is strictly greater than the maximal singular value of the matrix B. We also show that the proposed algorithm is globally and locally quadratically convergent no matter which smoothing function ϕ_i is used. Numerical implementations and comparisons based on these four different ϕ_i are reported as well. From the numerical results, we conclude that ϕ_2 is the best choice of smoothing function when we apply the proposed smoothing-type algorithm. More detailed reports will be seen in Sect. 4.

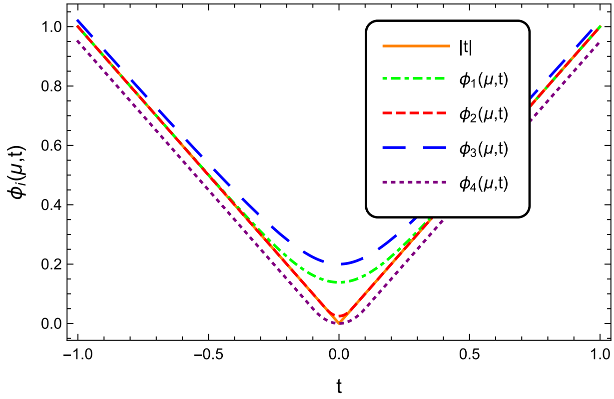

Fig. 1 Graphs of $|t|$ and all four $\phi_i(\mu, t)$ with $\mu = 0.1$

2 Smoothing Reformulation

In this section, we depict the graphs of ϕ_i for $i = 1, 2, 3, 4$ and investigate their properties. Then, we show the equivalent reformulation that $H_i(\mu, x) = 0$ if and only if x solves the AVE (1), and talk about the condition to guarantee the unique solvability of the AVE (1). We begin with showing the pictures of ϕ_i for $i = 1, 2, 3, 4$, see Fig. 1.

From Fig. 1, we see that ϕ_2 is the one which best approximates the function $|t|$ under the sense that it is closest to $|t|$ among all ϕ_i for $i = 1, 2, 3, 4$. To see this, we adopt the max norm to measure the distance of two real-valued functions. In other words, for given two real-valued functions f and g, the distance between them is defined as

$$\|f - g\|_\infty = \max_{t\in\mathbb{R}} \{f(t) - g(t)\}.$$

Now, for any fixed $\mu > 0$, we know that

$$\lim_{|t|\to\infty} \left|\phi_i(\mu, t) - |t|\right| = 0, \quad \text{for } i = 1, 2, 3.$$

This implies that

$$\max_{t\in\mathbb{R}} \left|\phi_i(\mu, t) - |t|\right| = |\phi_i(\mu, 0)|, \quad \text{for } i = 1, 2, 3.$$

Since, $\phi_1(\mu, 0) = (2\ln 2)\mu \approx 1.4\mu$, $\phi_2(\mu, 0) = \frac{\mu}{4}$, and $\phi_3(\mu, 0) = 2\mu$, we obtain

$$\begin{aligned}
\left\|\phi_1(\mu, t) - |t|\right\|_\infty &= (2\ln 2)\mu \approx 1.4\mu \\
\left\|\phi_2(\mu, t) - |t|\right\|_\infty &= \frac{\mu}{4} \\
\left\|\phi_3(\mu, t) - |t|\right\|_\infty &= 2\mu
\end{aligned}$$

On the other hand, we see that

$$\lim_{t\to\infty} \left|\phi_4(\mu, t) - |t|\right| = \frac{\mu}{2} \quad \text{and} \quad \phi_4(\mu, 0) = 0,$$

which says

$$\max_{t\in\mathbb{R}} \left|\phi_4(\mu, t) - |t|\right| = \frac{\mu}{2}.$$

Hence, we obtain

$$\left\|\phi_4(\mu, t) - |t|\right\|_\infty = \frac{\mu}{2}.$$

From all the above, we conclude that

$$\left\|\phi_3(\mu, t)-|t|\right\|_\infty > \left\|\phi_1(\mu, t)-|t|\right\|_\infty > \left\|\phi_4(\mu, t) - |t|\right\|_\infty > \left\|\phi_2(\mu, t)-|t|\right\|_\infty. \tag{9}$$

This shows that ϕ_2 is the function among ϕ_i, $i = 1, 2, 3, 4$ which best approximates the function $|t|$. In fact, for fixed $\mu > 0$, there has the local behavior that

$$\phi_3(\mu, t) > \phi_1(\mu, t) > \phi_2(\mu, t) > |t| > \phi_4(\mu, t). \tag{10}$$

A natural question arises here, does the smoothing algorithm based on ϕ_2 perform best among all $\phi_1, \phi_2, \phi_3, \phi_4$? This will be answered in Sect. 4.

Proposition 2.1 *Let* $\phi_i : \mathbb{R}^2 \to \mathbb{R}$ *for* $i = 1, 2, 3, 4$ *be defined as in* (5), (6), (7) and (8)*, respectively. Then, we have*

(a) ϕ_i *is continuously differentiable at* $(\mu, t) \in \mathbb{R}_{++} \times \mathbb{R}$*;*
(b) $\lim_{\mu\downarrow 0} \phi_i(\mu, t) = |t|$ *for any* $t \in \mathbb{R}$.

Proof (a) In order to prove the continuous differentiability of ϕ_i, we need to write out the expressions of $\frac{\partial\phi_i(\mu,t)}{\partial t}$ and $\frac{\partial\phi_i(\mu,t)}{\partial\mu}$; and then show the continuity of $\frac{\partial\phi_i(\mu,t)}{\partial t}$ and $\frac{\partial\phi_i(\mu,t)}{\partial\mu}$.

(i) For $i = 1$, we compute that

$$\begin{aligned}
\frac{\partial\phi_1(\mu, t)}{\partial t} &= \frac{1}{1+e^{-\frac{t}{\mu}}} - \frac{1}{1+e^{\frac{t}{\mu}}},\\
\frac{\partial\phi_1(\mu, t)}{\partial\mu} &= \left[\ln\left(1+e^{-\frac{t}{\mu}}\right) + \ln\left(1+e^{\frac{t}{\mu}}\right)\right] + \frac{t}{\mu}\left[\frac{-1}{1+e^{-\frac{t}{\mu}}} + \frac{1}{1+e^{\frac{t}{\mu}}}\right].
\end{aligned}$$

Then, it is clear to see that $\frac{\partial\phi_1(\mu,t)}{\partial t}$ and $\frac{\partial\phi_1(\mu,t)}{\partial\mu}$ are continuous. Hence, ϕ_1 is continuously differentiable.

(ii) For $i = 2$, we compute that

$$\frac{\partial \phi_2(\mu,t)}{\partial t} = \begin{cases} 1 & \text{if } t \geq \frac{\mu}{2}, \\ \frac{2t}{\mu} & \text{if } -\frac{\mu}{2} < t < \frac{\mu}{2}, \\ -1 & \text{if } t \leq -\frac{\mu}{2}. \end{cases}$$

$$\frac{\partial \phi_2(\mu,t)}{\partial \mu} = \begin{cases} 0 & \text{if } t \geq \frac{\mu}{2}, \\ -\left(\frac{t}{\mu}\right)^2 + \frac{1}{4} & \text{if } -\frac{\mu}{2} < t < \frac{\mu}{2}, \\ 0 & \text{if } t \leq -\frac{\mu}{2}. \end{cases}$$

Then, it can be verified that $\frac{\partial \phi_2(\mu,t)}{\partial t}$ and $\frac{\partial \phi_2(\mu,t)}{\partial \mu}$ are continuous because

$$\lim_{t \to \frac{\mu}{2}} \frac{\partial \phi_2(\mu,t)}{\partial t} = \lim_{t \to \frac{\mu}{2}} \frac{2t}{\mu} = 1,$$
$$\lim_{t \to -\frac{\mu}{2}} \frac{\partial \phi_2(\mu,t)}{\partial t} = \lim_{t \to -\frac{\mu}{2}} \frac{2t}{\mu} = -1.$$

and

$$\lim_{t \to \frac{\mu}{2}} \frac{\partial \phi_2(\mu,t)}{\partial \mu} = \lim_{t \to \frac{\mu}{2}} \left[-\left(\frac{t}{\mu}\right)^2 + \frac{1}{4} \right] = 0,$$
$$\lim_{t \to -\frac{\mu}{2}} \frac{\partial \phi_2(\mu,t)}{\partial \mu} = \lim_{t \to -\frac{\mu}{2}} \left[-\left(\frac{t}{\mu}\right)^2 + \frac{1}{4} \right] = 0.$$

Hence, ϕ_2 is continuously differentiable.

(iii) For $i = 3$, we compute that

$$\frac{\partial \phi_3(\mu,t)}{\partial t} = \frac{t}{\sqrt{4\mu^2 + t^2}},$$
$$\frac{\partial \phi_3(\mu,t)}{\partial \mu} = \frac{4\mu}{\sqrt{4\mu^2 + t^2}}.$$

Again it is clear to see that $\frac{\partial \phi_3(\mu,t)}{\partial t}$ and $\frac{\partial \phi_3(\mu,t)}{\partial \mu}$ are continuous. Hence, ϕ_3 is continuously differentiable.

(iv) For $i = 4$, we compute that

$$\frac{\partial \phi_4(\mu,t)}{\partial t} = \begin{cases} 1 & \text{if} \quad t > \mu, \\ \frac{t}{\mu} & \text{if} \quad -\mu \leq t \leq \mu, \\ -1 & \text{if} \quad t < -\mu. \end{cases}$$

$$\frac{\partial \phi_4(\mu,t)}{\partial \mu} = \begin{cases} -\frac{1}{2} & \text{if} \quad t > \mu, \\ -\frac{1}{2}\left(\frac{t}{\mu}\right)^2 & \text{if} \quad -\mu \leq t \leq \mu, \\ -\frac{1}{2} & \text{if} \quad t < -\mu. \end{cases}$$

Then, we conclude that $\frac{\partial\phi_4(\mu,t)}{\partial t}$ and $\frac{\partial\phi_4(\mu,t)}{\partial\mu}$ are continuous by checking

$$\lim_{t\to\mu}\frac{\partial\phi_4(\mu,t)}{\partial t}=\lim_{t\to\mu}\frac{t}{\mu}=1,$$
$$\lim_{t\to-\mu}\frac{\partial\phi_4(\mu,t)}{\partial t}=\lim_{t\to-\mu}\frac{t}{\mu}=-1.$$

and

$$\lim_{t\to\mu}\frac{\partial\phi_4(\mu,t)}{\partial\mu}=\lim_{t\to\mu}\left[-\frac{1}{2}\times\left(\frac{t}{\mu}\right)^2\right]=-\frac{1}{2},$$
$$\lim_{t\to-\mu}\frac{\partial\phi_4(\mu,t)}{\partial\mu}=\lim_{t\to-\mu}\left[-\frac{1}{2}\times\left(\frac{t}{\mu}\right)^2\right]=-\frac{1}{2}.$$

Hence, ϕ_4 is continuously differentiable.

From all the above, we prove that ϕ_i is continuously differentiable at $(\mu,t)\in\mathbb{R}_{++}\times\mathbb{R}$.
(b) For $i=1,2,3,4$, we always have the following:

$$\lim_{\mu\to0}\frac{\partial\phi_i(\mu,t)}{\partial t}=\begin{cases}1 & \text{if } t>0,\\ -1 & \text{if } t<0,\end{cases}$$

which verifies part (b). □

For subsequent needs in convergence analysis and numerical implementations, we summarize the gradient of each ϕ_i as below.

$$\nabla\phi_1(\mu,t)=\begin{bmatrix}\left[\ln(1+e^{-\frac{t}{\mu}})+\ln(1+e^{\frac{t}{\mu}})\right]+\frac{t}{\mu}\left[\frac{-1}{1+e^{-\frac{t}{\mu}}}+\frac{1}{1+e^{\frac{t}{\mu}}}\right]\\ \frac{1}{1+e^{-\frac{t}{\mu}}}-\frac{1}{1+e^{\frac{t}{\mu}}}\end{bmatrix}.$$

$$\nabla\phi_2(\mu,t)=\begin{bmatrix}\xi_1\\ \xi_2\end{bmatrix},\ \text{where } \xi_1=\begin{cases}0 & \text{if } t\ge\frac{\mu}{2},\\ -\left(\frac{t}{\mu}\right)^2+\frac{1}{4} & \text{if } -\frac{\mu}{2}<t<\frac{\mu}{2},\\ 0 & \text{if } t\le-\frac{\mu}{2}.\end{cases}$$

$$\xi_2=\begin{cases}1 & \text{if } t\ge\frac{\mu}{2},\\ \frac{2t}{\mu} & \text{if } -\frac{\mu}{2}<t<\frac{\mu}{2},\\ -1 & \text{if } t\le-\frac{\mu}{2}.\end{cases}$$

$$\nabla\phi_3(\mu,t)=\begin{bmatrix}\frac{4\mu}{\sqrt{4\mu^2+t^2}}\\ \frac{t}{\sqrt{4\mu^2+t^2}}\end{bmatrix}.$$

$$\nabla\phi_4(\mu,t)=\begin{bmatrix} v_1 \\ v_2 \end{bmatrix},\ \text{where } v_1=\begin{cases} -\frac{1}{2} & \text{if } t>\mu, \\ -\frac{1}{2}\left(\frac{t}{\mu}\right)^2 & \text{if } -\mu\le t\le\mu, \\ -\frac{1}{2} & \text{if } t<-\mu. \end{cases}$$

$$v_2=\begin{cases} 1 & \text{if } t>\mu, \\ \frac{t}{\mu} & \text{if } -\mu\le t\le\mu, \\ -1 & \text{if } t<-\mu. \end{cases}$$

In fact, Proposition 2.1 can be also depicted by geometric views. In particular, from Figs. 2, 3, 4 and 5, we see that when $\mu \downarrow 0$, ϕ_i is getting closer to $|t|$, which verifies Proposition 2.1(b).

Now, in light of Proposition 2.1, we obtain the equivalent reformulation $H_i(\mu, x) = 0$ for the AVE (1).

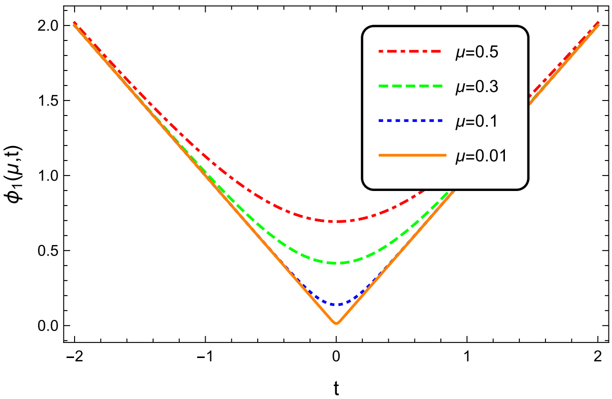

Fig. 2 Graphs of $\phi_1(\mu, t)$ with $\mu = 0.01, 0.1, 0.3, 0.5$

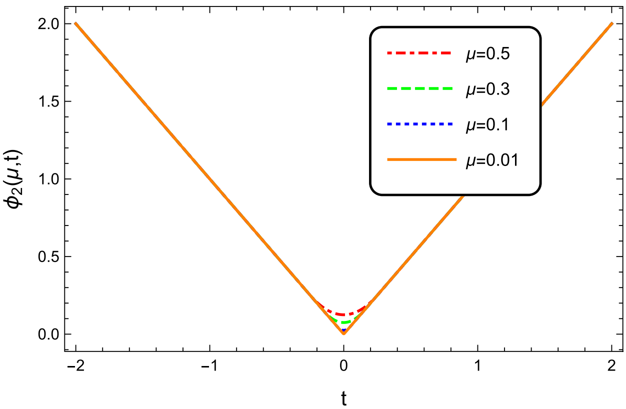

Fig. 3 Graphs of $\phi_2(\mu, t)$ with $\mu = 0.01, 0.1, 0.3, 0.5$

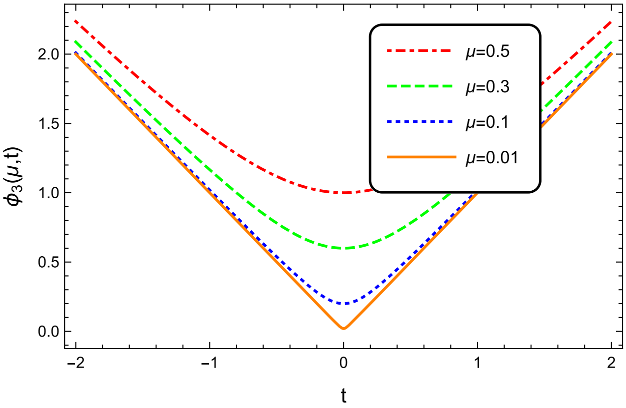

Fig. 4 Graphs of $\phi_3(\mu, t)$ with $\mu = 0.01, 0.1, 0.3, 0.5$

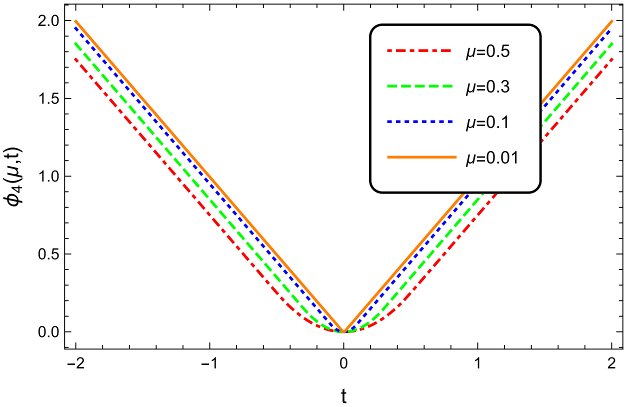

Fig. 5 Graphs of $\phi_4(\mu, t)$ with $\mu = 0.01, 0.1, 0.3, 0.5$

Proposition 2.2 *Let $\Phi_i(\mu, x)$ for $i = 1, 2, 3, 4$ be defined as in (4). Then, we have*

(a) *$H_i(\mu, x) = 0$ if and only if x solves the AVE (1);*
(b) *H_i is continuously differentiable on $\mathbb{R}^{n+1} \setminus \{\mathbf{0}\}$ with the Jacobian matrix given by*

$$\nabla H_i(\mu, x) := \begin{bmatrix} 1 & 0 \\ B\,\nabla_1 \Phi_i(\mu, x) & A + B\,\nabla_2 \Phi_i(\mu, x) \end{bmatrix} \tag{11}$$

where

$$\nabla_1 \Phi_i(\mu, x) := \begin{bmatrix} \frac{\partial \phi_i(\mu, x_1)}{\partial \mu} \\ \frac{\partial \phi_i(\mu, x_2)}{\partial \mu} \\ \vdots \\ \frac{\partial \phi_i(\mu, x_n)}{\partial \mu} \end{bmatrix},$$

$$\nabla_2 \Phi_i(\mu, x) := \begin{bmatrix} \frac{\partial \phi_i(\mu, x_1)}{\partial x_1} & 0 & \cdots & 0 \\ 0 & \frac{\partial \phi_i(\mu, x_2)}{\partial x_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \frac{\partial \phi_i(\mu, x_n)}{\partial x_n} \end{bmatrix}.$$

Proof This result follows from Proposition 2.1 immediately and the computation of the Jacobian matrix is straightforward. □

For completeness, we also talk about the unique solvability of the AVE (1), which is presumed in our numerical implementations. The following assumption and proposition are both employed from [9]. Assumption 2.3 will be also used to guarantee that $\nabla H_i(\mu, x)$ is invertible at any $(\mu, x) \in \mathbb{R}_{++} \times \mathbb{R}^n$, see Proposition 3.2 in Sect. 3.

Assumption 2.3 *The minimal singular value of the matrix A is strictly greater than the maximal singular value of the matrix B.*

Proposition 2.4 ([9, Proposition 2.3]) *The AVE (1) is uniquely solvable for any $b \in \mathbb{R}^n$ if Assumption 2.3 is satisfied.*

3 A smoothing-type algorithm

From Proposition 2.2, we know that the AVE (1) is equivalent to $H_i(\mu, x) = 0$. Accordingly, in this section, we consider the smoothing-type algorithm as in [1,9] to solve $H_i(\mu, x) = 0$. In fact, this type of algorithm has been also proposed for solving other kinds of problems, see [2,7,25] and references therein.

Algorithm 3.1 (A smoothing-type algorithm)

Step 0 *Choose* $\delta, \sigma \in (0, 1), \mu_0 > 0, x^0 \in \mathbb{R}^n$. *Set* $z^0 := (\mu, x^0)$. *Denote* $e^0 := (1, 0) \in \mathbb{R} \times \mathbb{R}^n$. *Choose* $\beta > 1$ *such that* $\left(\min\left\{1, \|H_i(z^0)\|\right\}\right)^2 \leq \beta\mu_0$. *Set* $k := 0$.

Step 1 *If* $\|H_i(z^k)\| = 0$, *stop.*

Step 2 *Set* $\tau_k := \min\left\{1, \|H_i(z^k)\|\right\}$, *and compute* $\triangle z^k := (\triangle\mu_k, \triangle x^k) \in \mathbb{R} \times \mathbb{R}^n$ *by using*

$$\nabla H_i(z^k) \triangle z^k = -H_i(z^k) + (1/\beta)\tau_k^2 e^0, \tag{12}$$

where $\nabla H_i(\cdot)$ *is defined as in* (11).

Step 3 *Let* α_k *be the maximum of the values* $1, \delta, \delta^2, \cdots$ *such that*

$$\|H_i(z^k + \alpha_k \triangle z^k)\| \leq [1 - \sigma(1 - 1/\beta)\alpha_k] \|H_i(z^k)\| \tag{13}$$

Step 4 *Set* $z^{k+1} := z^k + \alpha_k \triangle z^k$ *and* $k := k + 1$. *Back to Step 1.*

Following the same arguments as in [6,7], the line search (13) in the above scheme is well-defined. In other words, the Algorithm 3.1 is well-defined and possesses some nice properties.

Proposition 3.2 (a) *Suppose that Assumption 2.3 holds. Then, the Algorithm 3.1 is well-defined.*
(b) *Let the sequence* $\{z^k\}$ *be generated by Algorithm 3.1. Then,*
 (i) *both* $\{\|H_i(z^k)\|\}$ *and* $\{\tau_k\}$ *are monotonically decreasing;*
 (ii) $\tau_k^2 \leq \beta\mu_k$ *holds for all k;*
 (iii) *the sequence* $\{\mu_k\}$ *is monotonically decreasing, and* $\mu_k > 0$ *for all k.*

Proof Please refer to [7, Remark 2.1] or [9, Proposition 3.1]. □

The key point in the above scheme is the solvability of Newton equations (12) in Step 2. The following result is regarding this issue. Since the Φ_i function plays almost the same role as the function Φ_p used in [9], the below Proposition 3.3 can be obtained by mimicking the same arguments as in [9, Theorem 3.2]. We omit its proof and only state it.

Proposition 3.3 *Let* H_i *and* ∇H_i *be given as in* (3) *and* (11), *respectively. Suppose that Assumption 2.3 holds. Then,* $\nabla H_i(\mu, x)$ *is invertible at any* $(\mu, x) \in \mathbb{R}_{++} \times \mathbb{R}^n$.

Next, we discuss the global and local convergence. Again, although the function Φ_i here is continuously differentiable and the function Φ_p used in [9] is only semi-smooth, their roles in the proof are almost the same. Consequently, the arguments for convergence analysis are almost the same. Hence, we also omit the detailed proof and only present the convergence result.

Proposition 3.4 *Suppose that Assumption 2.3 holds and that the sequence* $\{z^k\}$ *is generated by Algorithm 3.1. Then,*

(a) $\{z^k\}$ *is bounded;*
(b) *any accumulation point of* $\{z^k\}$ *is a solution of the AVE* (1).
(c) *The whole sequence* $\{z^k\}$ *convergence to* z^* *with* $\|z^{k+1} - z^k\| = o\left(\|z^k - z^*\|\right)$ *and* $\mu_{k+1} = \mu_k^2$.

4 Numerical implementations

In this section, we report the numerical results of Algorithm 3.1 for solving the AVE (1) and (2). All numerical experiments are carried out in Mathematica 10.0 running on a PC with Intel i5 of 3.00 GHz CPU processor, 4.00 GB Memory and 32-bit Windows 7 operating system.

In our numerical experiments, the stoping criteria for Algorithm 3.1 is $\|H_i(z^k)\| \leq 1.0e{-}6$. We also stop programs when the total iteration is more than 100. Throughout the computational experiments, the following parameters are used:

$$\delta = 0.5, \quad \sigma = 0.0001, \quad \mu_0 = 0.1, \quad \beta = \max\left\{1, 1.01 * \tau_0^2/\mu\right\}.$$

4.1 Experiments on the AVE $Ax - |x| = b$

In this subsection we consider the simplified form of AVE (2). Consider the ordinary differential equation [4, Example 4.2]:

$$\frac{\mathrm{d}^2 x}{\mathrm{d}t^2} - |x| = (1 - t^2), \quad x(0) = -1, \quad x(1) = 0, \quad t \in [0, 1]. \tag{14}$$

As explained in [4, Example 4.2], after descretization (by using finite difference method), the above ODE can be recast an AVE in form of

$$Ax - |x| = b, \tag{15}$$

where the matrix A is given by

$$a_{i,j} = \begin{cases} -242, & i = j, \\ 121, & |i - j| = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

We implement the above problems by using ϕ_i, $i = 1, 2, 3, 4$ and $n = 2, 5, 10, 20, \ldots,$ 100, respectively. Every starting point x is randomly generated 10 times from a uniform distribution on $x \in [-2, 2]$. The results are put together in Table 1, where Dim denotes the size of problem, N_ϕ_i denotes the average number of iterations, T_ϕ_i denotes the average value of the CPU time in seconds, Ar_ϕ_i denotes the average value of $\|H(z^k)\|$ when Algorithm 3.1 stop.

From Table 1, in terms of the average number of iterations, the efficiency of $\phi_2(\mu, t)$ is best, followed by $\phi_4(\mu, t)$, $\phi_3(\mu, t)$ and $\phi_1(\mu, t)$. This is especially true for the problem of high dimension ordinary differential equation (14). In terms of time efficiency, $\phi_1(\mu, t)$ is still better than other functions too. In other words, for the AVE (2) arising from the ODE (15), we have

$$\phi_2(\mu, t) > \phi_4(\mu, t) > \phi_3(\mu, t) > \phi_1(\mu, t)$$

where ">" means "better performance".

To compare the performance of smoothing function $\phi_i(\mu, t), i = 1, 2, 3, 4$, we adopt the performance profile which is introduced in [3] as a means. In other words, we regard Algorithm 3.1 corresponding to a smoothing function $\phi_i(\mu, t)$, $i = 1, 2, 3, 4$ as a solver, and assume that there are n_s solvers and n_p test problems from the test set $\mathcal{P}$ which is generated randomly. We are interested in using the iteration number as performance measure for Algorithm 3.1 with different $\phi_i(\mu, t)$. For each problem p and solver s, let

$$f_{p,s} = \text{iteration number required to solve problem } p \text{ by solver } s.$$

Table 1 The numerical results of ordinary differential equation (14)

Dim	N_ϕ_1	T_ϕ_1	Ar_ϕ_1	N_ϕ_2	T_ϕ_2	Ar_ϕ_2	N_ϕ_3	T_ϕ_3	Ar_ϕ_3	N_ϕ_4	T_ϕ_4	Ar_ϕ_4
2	5.1	0.0967	3.30E−07	3.9	0.0015	6.92E−08	5.1	0.0016	5.93E−08	4	0.0062	5.99E−08
5	5.9	0.3697	2.23E−07	4.1	0.0031	7.47E−08	5.6	0.0062	2.21E−08	4.2	0.0016	6.54E−08
10	6.4	0.4851	2.98E−07	4.3	0.0094	2.10E−07	5.9	0.0031	1.05E−07	4.5	0.0031	4.67E−08
20	5.2	0.4290	2.41E−07	4.9	0.0078	1.10E−08	6.3	0.0078	2.13E−09	5	0.0094	2.46E−09
40	8.8	4.4117	4.66E−07	6.1	0.5210	5.28E−08	7.3	0.0172	6.59E−08	6.3	0.0156	1.88E−07
60	9.1	2.4289	2.31E−07	6.8	0.0281	4.49E−08	9	0.0312	1.20E−08	7.7	0.0312	1.31E−07
80	9.8	2.0514	3.61E−07	7.4	0.0374	3.21E−10	9.3	0.0452	3.21E−08	9.2	0.0593	3.15E−08
100	9.8	8.2306	4.44E−07	7.8	0.0577	8.78E−08	10	0.0671	2.26E−07	9.5	0.0827	2.83E−08

We employ the performance ratio

$$r_{p,s} := \frac{f_{p,s}}{\min\left\{f_{p,s} : s \in \mathcal{S}\right\}},$$

where $\mathcal{S}$ is the four solvers set. We assume that a parameter $r_{p,s} \leq r_M$ for all p, s are chosen, and $r_{p,s} = r_M$ if and only if solver s does not solve problem p. In order to obtain an overall assessment for each solver, we define

$$\rho_s(\tau) := \frac{1}{n_p}\text{size}\left\{p \in \mathcal{P} : r_{p,s} \leq \tau\right\},$$

which is called the performance profile of the number of iteration for solver s. Then, $\rho_s(\tau)$ is the probability for solver $s \in \mathcal{S}$ that a performance ratio $f_{p,s}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio.

We then need to test the four functions for ODE (14) at random starting points. In particular, starting points for each dimension are randomly chosen 20 times from a uniform distribution on $x \in [-2, 2]$. In order to obtain an overall assessment for the four functions, we are interested in using the number of iterations as a performance measure for Algorithm 3.1 with $\phi_1(\mu, t)$, $\phi_2(\mu, t)$, $\phi_3(\mu, t)$, and $\phi_4(\mu, t)$, respectively. The performance plot based on iteration number is presented in Fig. 6. From this figure, we see that $\phi_2(\mu, t)$ working with Algorithm 3.1 has the best numerical performance, followed by $\phi_4(\mu, t)$. In other words, in view of "iteration numbers", there has

$$\phi_2(\mu, t) > \phi_4(\mu, t) > \phi_3(\mu, t) > \phi_1(\mu, t)$$

where ">" means "better performance".

We are also interested in using the computing time as performance measure for Algorithm 3.1 with different $\phi_i(\mu, t)$, $i = 1, 2, 3, 4$. The performance plot based on

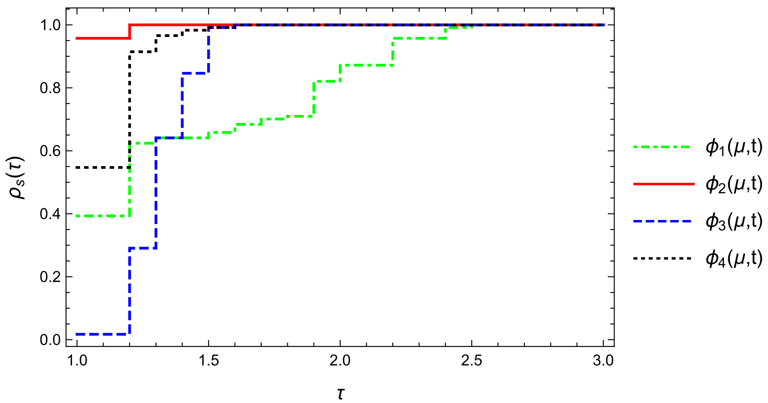

Fig. 6 Performance profile of iteration numbers of Algorithm 3.1 for the ODE (14)

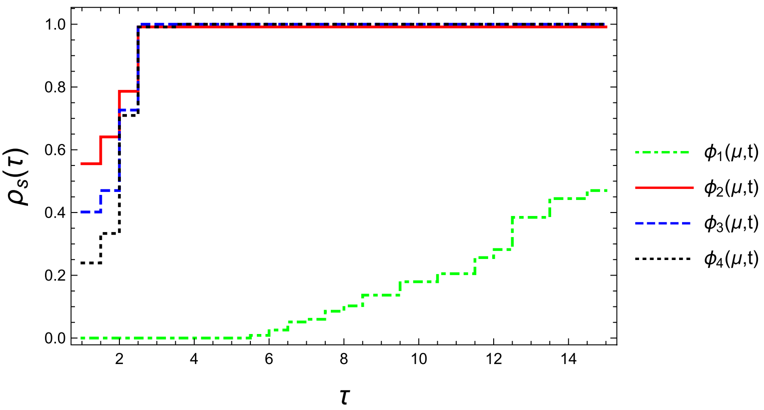

Fig. 7 Performance profile of computing time of Algorithm 3.1 for the ODE (14)

computing time is presented in Fig. 7. From this figure, we can also see the function $\phi_2(\mu, t)$ has best performance, then followed by $\phi_3(\mu, t)$. Note that the time efficiency of $\phi_1(\mu, t)$ is very bad. In other words, in view of "computing tim", there has

$$\phi_2(\mu, t) > \phi_3(\mu, t) > \phi_4(\mu, t) > \phi_1(\mu, t)$$

where ">" means "better performance".

In summary, for the special AVE (2) arising from the ODE (14), no matter the number of iterations or the computing time is taken into account, the function $\phi_2(\mu, t)$ is the best choice for the Algorithm 3.1.

4.2 Experiments on the general AVE $Ax + B|x| = b$

In this subsection we consider the general AVE (1): $Ax + B|x| = b$. Here matrix A (or B) is equal to a normal distribution random matrix minus another one so that we can randomly generate the testing problems.

In order to ensure that Assumption 2.3 holds, we further modify the matrix A in light of the below conditions.

- If $\min\{w_{ii} : i = 1, \ldots, n\} = 0$ with $\{u, w, v\} =$ SingularValueDecomposition $[A]$, then we set $A = u(w + 0.01 \times \text{IdentityMatrix}[n])v$.
- Set $A = \frac{\lambda_{max}(B^{\mathrm{T}}B)+0.01}{\lambda_{min}(A^{\mathrm{T}}A)} A$.

Then, it is clear to verify that Assumption 2.3 is satisfied for such A. Moreover, we set p =2RandomVariate [NormalDistribution[ ],{n, 1}] and $b = Ap + B|p|$ so that the testing problems are solvable.

We implement the above problems for ϕ_i, $i = 1, 2, 3, 4$ and $n = 2, 5, 10, 20, \ldots, 100$, respectively. Every case is randomly generated 10 times for testing. The numerical results are listed in Table 2. From Table 2, in terms of the number of iterations and computation time, the efficiency of $\phi_2(\mu, t)$ is best, followed by $\phi_4(\mu, t)$. The iteration

Table 2 The numerical results of experiments

Dim	N_ϕ_1	T_ϕ_1	Ar_ϕ_1	N_ϕ_2	T_ϕ_2	Ar_ϕ_2	N_ϕ_3	T_ϕ_3	Ar_ϕ_3	N_ϕ_4	T_ϕ_4	Ar_ϕ_4
2	6.2	0.4596	5.00E−7	3.6	0.0031	8.56E−8	7.1	0.0016	1.79E−7	3.9	0	8.04E−8
5	7.4	0.2246	6.05E−7	4.1	0.0031	8.39E−8	9.6	0.0094	4.73E−7	4.3	0.0016	7.53E−8
10	10.2	1.0733	2.23E−7	4.3	0.0062	8.26E−8	17.2	0.0187	4.79E−7	4.7	0.0031	7.53E−8
20	19.8	3.7830	5.00E−7	4.8	0.0062	9.95E−8	26.3	0.0499	1.86E−7	5.9	0.0094	1.06E−7
30	28.7	5.0575	4.46E−7	5.6	0.0140	1.00E−7	43.2	0.1295	5.22E−8	9.3	0.0265	1.82E−7
40	38.6	3.0935	6.52E−7	7.1	0.0234	5.60E−8	54.1	0.2137	1.65E−7	11.9	0.0374	9.14E−8
50	42.7	1.9016	5.37E-7	5.3	0.0218	7.73E−8	61.5	0.3120	1.93E−8	10.4	0.0437	5.88E−8
60	52.1	2.5272	5.61E−7	6.6	0.0359	5.90E−8	78.7	0.4976	1.05E−8	13.9	0.0718	1.15E−7
70	60.2	3.7050	6.10E−7	9.9	0.0624	1.12E−7	94.4	0.7332	1.80E−7	18.7	0.1264	1.26E−7
80	58.0	4.1246	4.31E−7	8.9	0.0640	6.03E−8	98.5	0.8845	3.88E−8	17.5	0.1420	5.35E−8
90	78.2	11.170	6.28E−7	10.0	0.0905	2.23E−7	114.3	1.2745	1.46E−7	20.9	0.2028	1.46E−7
100	72.2	12.211	4.77E−7	7.5	0.0709	1.62E−7	110.8	1.6477	1.31E−7	16.9	0.1881	1.34E−7

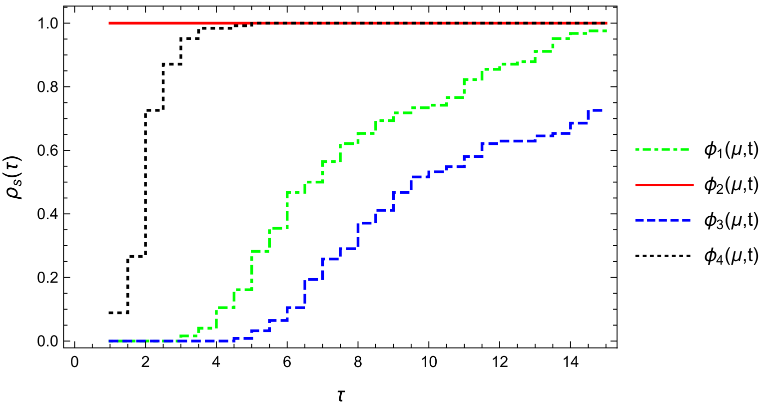

Fig. 8 Performance profile of iteration numbers of Algorithm 3.1 for general AVE

number of $\phi_1(\mu, t)$ is less than $\phi_3(\mu, t)$, but the computing time of $\phi_1(\mu, t)$ is more than $\phi_3(\mu, t)$.

Figure 8 shows the performance profile of iteration number in Algorithm 3.1 in the range of $\tau \in [1, 15]$ for four solvers on 100 test problem which are generated randomly. The four solvers correspond to Algorithm 3.1 with $\phi_1(\mu, t)$, $\phi_2(\mu, t)$, $\phi_3(\mu, t)$, and $\phi_4(\mu, t)$, respectively. From this figure, we see that $\phi_2(\mu, t)$ working with Algorithm 3.1 has the best numerical performance, followed by $\phi_4(\mu, t)$. In summary, from the viewpoint of "iteration numbers", we conclude that

$$\phi_2(\mu, t) > \phi_4(\mu, t) > \phi_1(\mu, t) > \phi_3(\mu, t)$$

where "$>$" means "better performance".

Finally, we are also interested in using the computing time as performance measure for Algorithm 3.1 with different $\phi_i(\mu, t)$, $i = 1, 2, 3, 4$. The performance plot based on computing time is presented in Fig. 9. From this figure, we can also see the function

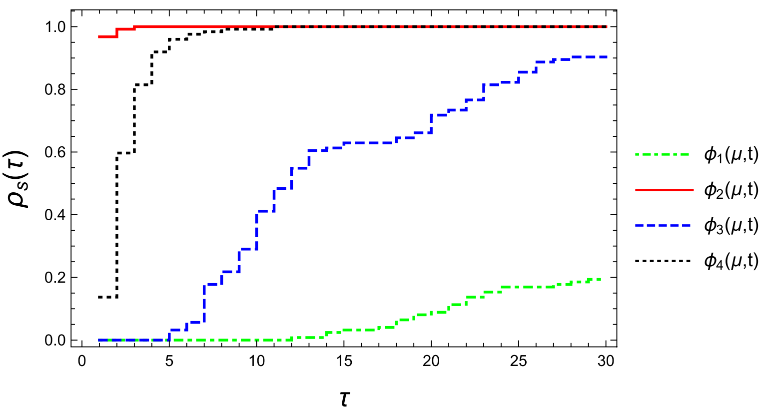

Fig. 9 Performance profile of computing time of Algorithm 3.1 for general AVE

$\phi_2(\mu, t)$ has best performance, then followed by $\phi_4(\mu, t)$. Note that the time efficiency of $\phi_1(\mu, t)$ is very bad. Again, from the viewpoint of "computing time", we conclude that

$$\phi_2(\mu, t) > \phi_4(\mu, t) > \phi_3(\mu, t) > \phi_1(\mu, t)$$

where ">" means "better performance".

5 Conclusion

In this paper, we recast the AVE (1) as a system of smooth equations. Accordingly, we have proposed four smoothing functions along with a smoothing-type algorithm studied in [1,9] to solve it. As mentioned in Sect. 2, there holds the local behavior shown as in (10):

$$\phi_3(\mu, t) > \phi_1(\mu, t) > \phi_2(\mu, t) > |t| > \phi_4(\mu, t).$$

and $\phi_2(\mu, t)$ is the one which best approximates the function $|t|$ shown as in (9), i.e.,

$$\left\|\phi_3(\mu, t) - |t|\right\|_\infty > \left\|\phi_1(\mu, t) - |t|\right\|_\infty > \left\|\phi_4(\mu, t) - |t|\right\|_\infty > \left\|\phi_2(\mu, t) - |t|\right\|_\infty.$$

Surprisingly, $\phi_2(\mu, t)$ is also the best choice of smoothing function no matter when the iteration number or the computing time is taken into account. For the "iteration" aspect, the order of numerical performance from good to bad is

$$\begin{cases} \phi_2(\mu, t) > \phi_4(\mu, t) > \phi_1(\mu, t) > \phi_3(\mu, t), & \text{for th AVE (1).} \\ \phi_2(\mu, t) > \phi_4(\mu, t) > \phi_3(\mu, t) > \phi_1(\mu, t), & \text{for th AVE (2).} \end{cases}$$

whereas for the "time" aspect, the order of numerical performance from good to bad is

$$\begin{cases} \phi_2(\mu, t) > \phi_4(\mu, t) > \phi_3(\mu, t) > \phi_1(\mu, t), & \text{for th AVE (1).} \\ \phi_2(\mu, t) > \phi_3(\mu, t) > \phi_4(\mu, t) > \phi_1(\mu, t), & \text{for th AVE (2).} \end{cases}$$

In other words, $\phi_2(\mu, t)$ is the best choice of smoothing function to work with the proposed smoothing-type algorithm, meanwhile it also best approximate the function $|t|$. This is a very interesting discovery which may be helpful in other contexts. One of future directions is to check whether such phenomenon occurs in other types of algorithms.

Acknowledgements The author B. Saheya's work is supported by Natural Science Foundation of Inner Mongolia (Award No. 2014MS0119). The author J.-S. Chen's work is supported by Ministry of Science and Technology, Taiwan.

References

1. Caccetta, L., Qu, B., Zhou, G.-L.: A globally and quadratically convergent method for absolute value equations. Comput. Optim. Appl. **48**, 45–58 (2011)
2. Chen, J.-S., Ko, C.-H., Liu, Y.-D., Wang, S.-P.: New smoothing functions for solving a system of equalities and inequalities. Pac. J. Optim. **12**, 185–206 (2016)
3. Dolan, E.D., More, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2002)
4. Haghani, F.K.: On generalized traubs method for absolute value equations. J. Optim. Theory Appl. **166**, 619–625 (2015)
5. Hu, S.-L., Huang, Z.-H., Zhang, Q.: A generalized Newton method for absolute value equations associated with second order cones. J. Comput. Appl. Math. **235**, 1490–1501 (2011)
6. Huang, Z.-H.: Locating a maximally complementary solution of the monotone NCP by using non-interior-point smoothing algorithms. Math. Methods of Oper. Res. **61**, 41–45 (2005)
7. Huang, Z.-H., Zhang, Y., Wu, W.: A smoothing-type algorithm for solving system of inequalities. J. Comput. Appl. Math. **220**, 355–363 (2008)
8. Hu, S.-L., Huang, Z.-H.: A note on absolute value equations. Optim. Lett. **4**, 417–424 (2010)
9. Jiang, X., Zhang, Y.: A smoothing-type algorithm for absolute value equations. J. Ind. Manag. Optim. **9**, 789–798 (2013)
10. Ketabchi, S., Moosaei, H.: Minimum norm solution to the absolute value equation in the convex case. J. Optim. Theory Appl. **154**, 1080–1087 (2012)
11. Mangasarian, O.L.: Absolute value programming. Comput. Optim. Appl. **36**, 43–53 (2007)
12. Mangasarian, O.L.: Absolute value equation solution via concave minimization. Optim. Lett. **1**, 3–5 (2007)
13. Mangasarian, O.L.: A generalized Newton method for absolute value equations. Optim. Let. **3**, 101–108 (2009)
14. Mangasarian, O.L.: Primal-dual bilinear programming solution of the absolute value equation. Optim. Lett. **6**, 1527–1533 (2012)
15. Mangasarian, O.L.: Absolute value equation solution via dual complementarity. Optim. Lett. **7**, 625–630 (2013)
16. Mangasarian, O.L., Meyer, R.R.: Absolute value equation. Linear Algebra Appl. **419**, 359–367 (2006)
17. Prokopyev, O.A.: On equivalent reformulations for absolute value equations. Comput. Optim. Appl. **44**, 363–372 (2009)
18. Qi, L.: Convergence analysis of some algorithms for solving nonsmooth equations. Math. Oper. Res. **18**, 227–244 (1993)
19. Qi, L., Sun, D., Zhou, G.-L.: A new look at smoothing Newton methods for nonlinear complementarity problems and box constrained variational inequality problems. Math. Program. **87**, 1–35 (2000)
20. Rohn, J.: A theorem of the alternatives for the equation $Ax + B|x| = b$. Linear Multilinear Algebra **52**, 421–426 (2004)
21. Rohn, J.: Solvability of systems of interval linear equations and inequalities. In: Fiedler, M., Nedoma, J., Ramik, J., Rohn, J., Zimmermann, K. (eds.) Linear Optimization Problems with Inexact Data, pp. 35–77. Springer, New York (2006)
22. Rohn, J.: An algorithm for solving the absolute value equation. Electron. J. Linear Algebra **18**, 589–599 (2009)
23. Yamanaka, S., Fukushima, M.: A branch and bound method for the absolute value programs. Optimization **63**, 305–319 (2014)
24. Zhang, C., Wei, Q.-J.: Global and finite convergence of a generalized Newton method for absolute value equations. J. Optim. Theory Appl. **143**, 391–403 (2009)
25. Zhang, Y., Huang, Z.-H.: A nonmonotone smoothing-type algorithm for solving a system of equalities and inequalities. J. Comput. Appl. Math. **233**, 2312–2321 (2010)