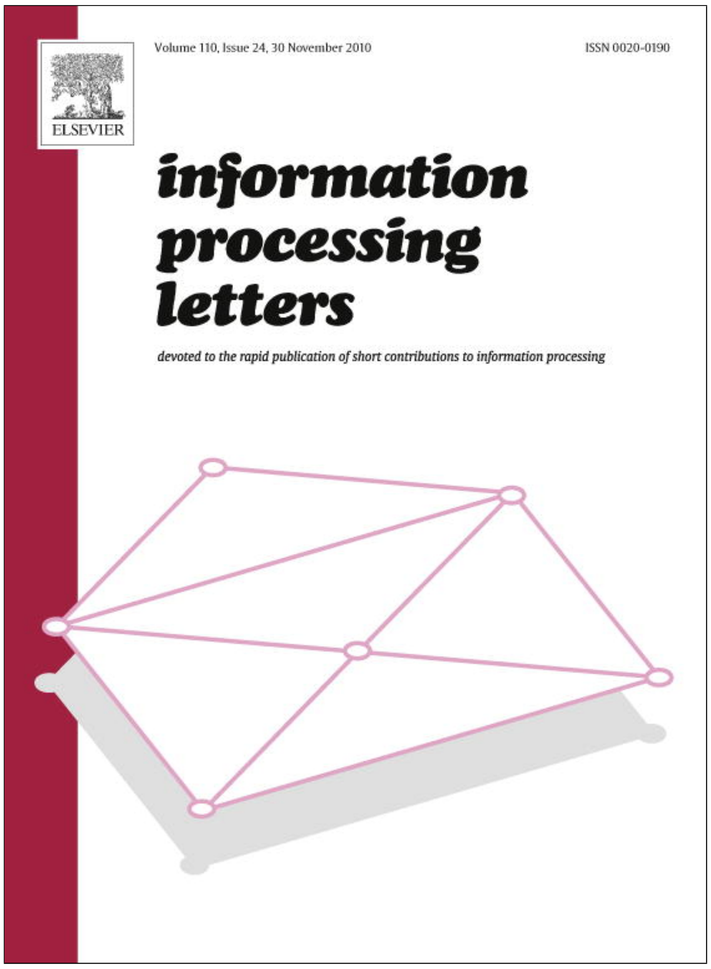
Volume 110, Issue 24, 30 November 2010
ISSN 0020-0190
ELSEVIER
information processing letters
devoted to the rapid publication of short contributions to information processing

Information Processing Letters 110 (2010) 1103–1109

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Time-varying square roots finding via Zhang dynamics versus gradient dynamics and the former's link and new explanation to Newton–Raphson iteration [☆]

Yunong Zhang [*], Zhende Ke, Peng Xu, Chenfu Yi

School of Information Science and Technology, Sun Yat-Sen University, Guangzhou 510006, China

A R T I C L E   I N F O

Article history:
Received 31 March 2010
Received in revised form 3 September 2010
Accepted 29 September 2010
Available online 9 October 2010
Communicated by A.A. Bertossi

Keywords:
Parallel algorithms
Real-time systems
Zhang dynamics
Gradient dynamics
Indefinite error-function
Square roots finding
Newton–Raphson iteration

A B S T R A C T

Different from conventional gradient-based neural dynamics, a special class of neural dynamics have been proposed by Zhang et al. since 12 March 2001 for online solution of time-varying and static (or termed, time-invariant) problems (e.g., nonlinear equations). The design of Zhang dynamics (ZD) is based on the elimination of an indefinite error-function, instead of the elimination of a square-based positive or at least lower-bounded energy-function usually associated with gradient dynamics (GD) and/or Hopfield-type neural networks. In this paper, we generalize, develop, investigate and compare the continuous-time ZD (CTZD) and GD models for online solution of time-varying and static square roots. In addition, a simplified continuous-time ZD (S-CTZD) and discrete-time ZD (DTZD) models are generated for static scalar-valued square roots finding. In terms of such scalar square roots finding problem, the Newton iteration (also termed, Newton–Raphson iteration) is found to be a special case of the DTZD models (by focusing on the static-problem solving, utilizing the linear activation function and fixing the step-size to be 1). Computer-simulation results via a power-sigmoid activation function further demonstrate the efficacy of the ZD solvers for online scalar (time-varying and static) square roots finding, in addition to the DTZD's link and new explanation to Newton–Raphson iteration.

1. Introduction

The problem of online solution of time-varying square roots in the form of $x^2(t) - a(t) = 0$ and/or static (or termed, constant, time-invariant) square roots in the form of $x^2 - a = 0$ is considered to be a basic problem arising in science and engineering fields. Thus, many numerical algorithms are presented for such a problem solving [1–9]. However, it may not be efficient enough for most numerical algorithms due to their serial-processing nature performed on digital computers [10]. Suitable for analog VLSI implementation [11,12] and in view of potential high-speed parallel processing, the neural-dynamic approach is now regarded as a powerful alternative to online problem solving [13–26]. Besides, it is worth mentioning that most reported computational-schemes were theoretically/intrinsically designed for time-invariant (or termed, static, constant) problems solving and/or related to gradient methods [13–15,20,24,25].

Different from the gradient-based neural-dynamic approach [13–15,18,20,24,25], a special class of neural dynamics has been formally proposed by Zhang et al. [13–26] (since March 2001 [21]) for time-varying and/or static problems solving, such as time-varying convex quadratic program [15], matrix inversion [16,18,22,26], and Sylvester equation solving [21]. In addition, the proposed Zhang dynamics (ZD) is designed based on an indefinite error-function instead of a square-based positive (or at least lower-bounded) energy-function usually associated with

[☆] This work is supported by the National Natural Science Foundation of China under Grants 61075121, 60935001 and 60775050, and also by the Fundamental Research Funds for the Central Universities of China.

[*] Corresponding author.
E-mail address: zhynong@mail.sysu.edu.cn (Y. Zhang).

0020-0190/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.ipl.2010.09.013

gradient-based models or Lyapunov design-analysis methods [13–15,18,20,24].

In this paper, a novel continuous-time ZD (CTZD) model is introduced, generalized, developed and investigated for online time-varying scalar square roots finding. For comparative purposes, the conventional gradient dynamics (GD) model is also generated and investigated for this time-varying problem solving. As we know, time-invariant (or, static) problems solving could be viewed as a special case of the time-varying problems solving. Correspondingly, a simplified CTZD model [14] is generated. In addition, we also develop and study the discrete-time ZD (DTZD) model for static square roots finding. Comparing with the Newton–Raphson iteration for finding square roots, we discover that the DTZD model contains Newton–Raphson iteration as its special case. In detail, Newton–Raphson iteration can also be obtained (different from the traditional explanation [1–9]) by utilizing the linear activation function and fixing the step-size to be 1 in a general DTZD model. Computer-simulation results using a power-sigmoid activation function further substantiate the efficacy of the CTZD model for online time-varying square roots finding, as well as the efficacy of the DTZD model and Newton–Raphson iteration for static square roots finding.

To the best of the authors' knowledge, most reported computational-schemes were theoretically/intrinsically designed for time-invariant (or termed, static, constant) problems solving and/or related to gradient methods. There is almost no other literature dealing with such a specific problem of online solution of time-varying square roots at present stage, and the main contributions of the Letter may lie in the following facts.

- In this Letter, a novel CTZD model is generalized, developed and investigated for online solution of time-varying square roots in real time t. To show the novelty and difference, we compare the CTZD model with the conventional GD model by solving the same time-varying square root problem.
- The S-CTZD model is presented in this Letter for the time-invariant square root problem solving (which could be viewed as a special case of the time-varying problem solving). In addition, the DTZD model is generalized for the static case for the purposes of possible hardware implementation.
- This Letter additionally shows the link of the new neural-dynamic models to Newton–Raphson iteration, which gives a new effective explanation to Newton–Raphson iteration, differing very much from the traditional (or to say, standard) explanation appearing in almost all literature and textbooks (i.e., via Taylor series expansion).
- Computer-simulation results using the power-sigmoid activation function further demonstrate the efficacy of the proposed ZD models (i.e., CTZD, S-CTZD and DTZD models) for online time-varying and static square roots finding, and comparatively show the less favorable property of the GD model when applied to solving the time-varying square root problem, in addition to the former's link to Newton–Raphson iteration by focusing on the static problem solving.

2. Problem formulation and continuous-time neural-dynamic solvers

In the ensuing subsections, the CTZD and GD models are presented for time-varying square-roots finding. In addition, a simplified CTZD is developed to find static scalar square roots, which can be viewed as a special case of the time-varying scalar square root problem solving.

2.1. Problem formulation

Let us consider the following time-varying scalar square root problem (which is written in the form of a time-varying nonlinear equation [13,24] for convenience):

$$x^2(t) - a(t) = 0, \quad t \in [0, +\infty), \tag{1}$$

where $a(t) \in R$ in this work denotes a smoothly time-varying positive scalar, which, together with time derivative $\dot{a}(t) \in R$, is assumed to be known numerically or could be measured accurately. Our objective in this work is to find $x(t) \in R$ in real time t such that the above smoothly time-varying nonlinear equation (1) holds true. For presentation convenience, let $x^*(t) \in R$ denote a theoretical time-varying square root of $a(t)$.

2.2. Zhang dynamics

By following Zhang et al.'s neural-dynamic design method [13–16,21,22,24–26], to monitor and control the time-varying square roots finding process, we could firstly define an indefinite error function (of which the word "indefinite" means that such an error function can be positive, zero, negative or even lower-unbounded):

$$e(t) := x^2(t) - a(t) \in R.$$

Secondly, the time-derivative $\dot{e}(t)$ of error-function $e(t)$ could be chosen and forced such that $e(t)$ mathematically converges to zero. Specifically, we can choose $e(t)$ converging to zero via the following ZD design formula:

$$\frac{de(t)}{dt} = -\gamma \phi\big(e(t)\big),$$

where design-parameter $\gamma > 0$ is used to scale the convergence rate of ZD, and $\phi(\cdot) : R \to R$ denotes a general activation function mapping. Expanding the above design formula, we could thus obtain

$$2x(t)\dot{x}(t) - \dot{a}(t) = -\gamma \phi\big(x^2(t) - a(t)\big),$$

which could lead to the following CTZD model for time-varying square root finding:

$$\dot{x}(t) = \frac{\dot{a}(t) - \gamma \phi(x^2(t) - a(t))}{2x(t)}, \tag{2}$$

where $x(t)$, starting from randomly-generated initial condition $x(0) = x_0 \in R$, is the neural-activation state corresponding to theoretical solution $x^*(t)$ of (1). In view of ZD

(2) [and hereafter GD (3)], different choices of activation function $\phi(\cdot)$ and design parameter γ may lead to different performance of the neural dynamics. Generally speaking, any monotonically-increasing odd activation function $\phi(\cdot)$ can be used for the construction of the neural dynamics. Since March 2001 [21], we have introduced and used five types of activation functions (i.e., linear activation function, power activation function, power-sum activation function, sigmoid activation function and power-sigmoid activation function) for the proposed ZD models (for more details, see [13–18,22,24–26]). Moreover, similar to usual neural-dynamic approaches, design parameter γ in ZD (2) [and hereafter in GD (3)], being the reciprocal of a capacitance parameter in the hardware implementation, should be set as large as hardware permits (e.g., in analog circuits or VLSI [11,12]) or selected appropriately (e.g., between 10^3 and 10^8) for experimental and/or simulative purposes.

As for the convergence of CTZD (2), we have the following proposition [13,19,24].

Proposition 1. *Consider a smoothly time-varying positive scalar $a(t) \in R$ in Eq. (1). If a monotonically-increasing odd activation-function $\phi(\cdot)$ is used, then*

- *neural-state $x(t) \in R$ of CTZD (2), starting from randomly-generated positive initial state $x_0 \in R$, converges to theoretical positive time-varying square root $x^*(t)$ [or denoted by $a^{1/2}(t)$] of $a(t)$; and,*
- *neural-state $x(t) \in R$ of CTZD (2), starting from randomly-generated negative initial state $x_0 \in R$, converges to theoretical negative time-varying square root $x^*(t)$ [or denoted by $-a^{1/2}(t)$] of $a(t)$.*

2.3. Gradient dynamics

For comparative purposes, the gradient dynamics is also developed and exploited for online solution of the above-presented time-varying square root problem (1). To solve Eq. (1), by following the conventional gradient-based design method, we could firstly define a scalar-valued square-based energy function such as $\mathcal{E} = (x^2 - a)^2/2$. Then, a typical continuous-time learning rule based on the negative-gradient information leads to the following differential equation {which is the linear form of our presented gradient-dynamics (GD) [13–15,18,20,24,25]}:

$$\frac{dx(t)}{dt} := -\gamma \frac{\partial \mathcal{E}}{\partial x} = -2\gamma x(t)\big(x^2(t) - a(t)\big).$$

Under the inspiration of [22], from the above we could obtain the nonlinear form of a general GD model by adding a nonlinear activation function $\phi(\cdot)$ as follows:

$$\dot{x}(t) = -2\gamma x(t)\phi\big(x^2(t) - a(t)\big), \tag{3}$$

where $x(t)$, starting from randomly-generated initial condition $x(0) = x_0 \in R$, is the activation neural state.

Remarks. It is worth comparing the two design methods of CTZD (2) and GD (3), both of which are exploited for finding the time-varying square roots of nonlinear equation (1). The differences lie in the following facts.

(1) The design of CTZD (2) is based on the elimination of an indefinite error-function $e(t) := x^2(t) - a(t)$, which could be positive, negative, bounded or unbounded. In contrast, the design of GD (3) is based on the elimination of the square-based positive energy function $\mathcal{E} = (x^2 - a)^2/2$.
(2) The design of CTZD (2) is based on a new method intrinsically for time-varying problems solving. Thus, it could converge to an exact/theoretical time-varying square root of (1). In contrast, the design of GD (3) is based on a method intrinsically for the scalar square-root equation but with a static/constant coefficient. It is thus only able to approximately approach the theoretical time-varying square root of (1).
(3) CTZD (2) methodically and systematically exploits the time-derivative information of $a(t)$ [i.e., $\dot{a}(t)$] during its real-time finding process. This appears to be the reason why CTZD (2) could converge to an exact/theoretical time-varying square root of (1). In contrast, GD (3) has not exploited the time-derivative information of $a(t)$, and thus is not effective enough on solving the time-varying square root problem.

Note that, as a special case of time-varying nonlinear equation (1), let us consider the following time-invariant nonlinear equation (which corresponds to the static square root problem formulated in most textbooks, e.g., in [1–3]):

$$x^2(t) - a = 0, \tag{4}$$

where $a \in R$ is a positive constant. By considering the constant a in (4), $\dot{a}(t) \equiv 0$ and CTZD (2) becomes

$$\dot{x}(t) = -\gamma \frac{\phi(x^2(t) - a)}{2x(t)}, \tag{5}$$

which is termed as a simplified-CTZD (S-CTZD) model used to find the time-invariant (i.e., static) square roots. By Proposition 1, we have the following proposition on the convergence of S-CTZD (4).

Proposition 2. *Consider a time-invariant positive scalar $a \in R$ in nonlinear equation (4). If a monotonically-increasing odd activation-function $\phi(\cdot)$ is used, then*

- *neural-state $x(t) \in R$ of S-CTZD (5), starting from randomly-generated positive initial state $x_0 \in R$, converges to theoretical positive time-invariant square root x^* [or denoted by $a^{1/2}$] of a; and,*
- *neural-state $x(t) \in R$ of S-CTZD (5), starting from randomly-generated negative initial state $x_0 \in R$, converges to theoretical negative time-invariant square root x^* [or denoted by $-a^{1/2}$] of a.*

3. Discrete-time ZD with its link and new explanation to Newton–Raphson iteration

For possible hardware implementation (e.g., on digital circuits) of Zhang neural dynamics used for static square roots finding, we could discretize the S-CTZD model (5) by using Euler forward-difference rule [1–9,16]:

$$\dot{x}(t = k\tau) \approx \big(x\big((k+1)\tau\big) - x(k\tau)\big)/\tau,$$

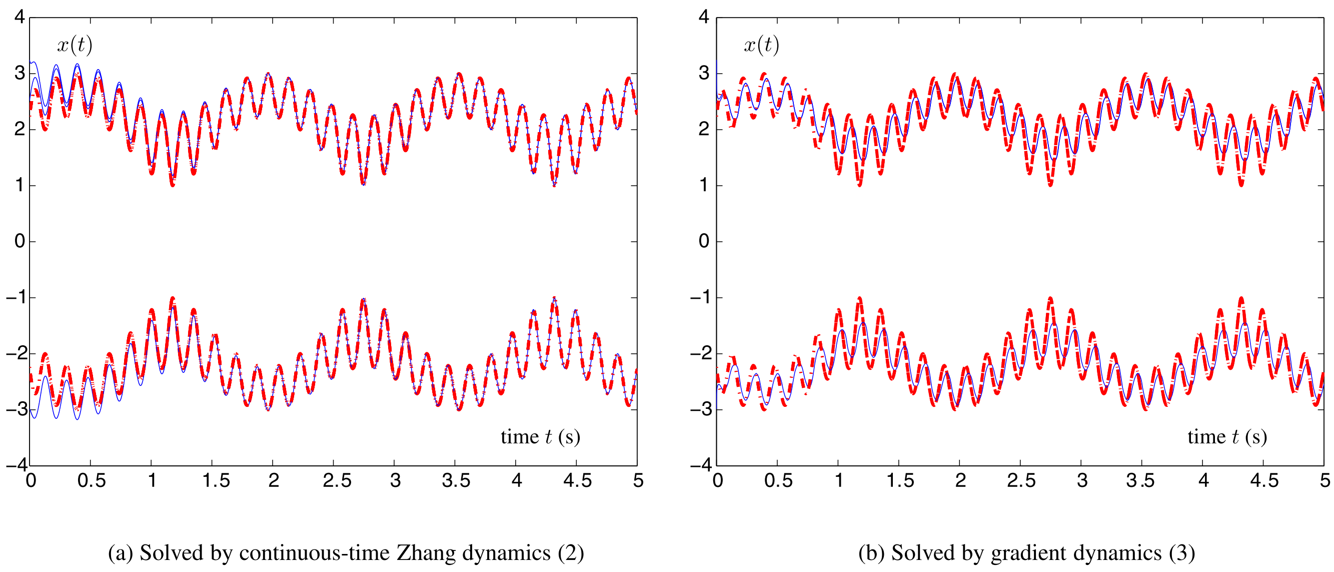

Fig. 1. Time-varying square roots finding of (9) by CTZD (2) and GD (3) using a power-sigmoid activation function with $\gamma = 1$ and starting from randomly-generated initial states within $[-4, 4]$, where the neural-dynamic solutions are denoted by solid blue curves, and the theoretical time-varying square roots are denoted by dash-dotted red curves. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

where $\tau > 0$ denotes the sampling gap, and $k = 0, 1, 2, \ldots$. In general, we denote $x_k = x(t = k\tau)$ for presentation convenience [27]. Then, the discrete-time ZD (DTZD) model for the static square roots finding can be derived from S-CTZD (5) as follows:

$$(x_{k+1} - x_k)/\tau = -\gamma \frac{\phi(x_k^2 - a)}{2x_k},$$

which could be further formulated as

$$x_{k+1} = x_k - h\frac{\phi(x_k^2 - a)}{2x_k}, \tag{6}$$

where parameter $h = \tau\gamma$ denotes the step-size (or termed, step-length). Generally speaking, different choices of step-size h and activation function $\phi(\cdot)$ may lead to different convergence performance of DTZD (6). From an important theorem in [19], the following proposition on the convergence of DTZD (6) could be achieved finally.

Proposition 3. *Consider the static square root problem* (4). *When the step-size h of DTZD solver* (6) *satisfies* $0 < h < 2/\theta$ [*with* $\theta := \phi'(0)$], *there generally exists a positive number* δ *such that the sequence* $\{x_k\}_{k=0}^{+\infty}$ *could converge to a theoretical root* x^* *of* (4) *for any initial state* $x_0 \in [x^* - \delta, x^* + \delta]$.

Review the DTZD model (6). If we utilize linear activation function $\phi(u) = u$, the DTZD solver (6) reduces to

$$x_{k+1} = x_k - h\frac{x_k^2 - a}{2x_k}. \tag{7}$$

In addition, if $h \equiv 1$, the DTZD (7) could further reduce to

$$x_{k+1} = x_k - \frac{x_k^2 - a}{2x_k}, \quad \text{i.e.,} \ x_{k+1} = \frac{x_k + a/x_k}{2}, \tag{8}$$

which is exactly the Newton–Raphson iteration for time-invariant square roots finding widely presented in textbooks and literature [1–9]. In other words, we have discovered once more [16] that a general form of Newton–Raphson iteration could be given by discretizing the S-CTZD model (5). Specifically, the Newton–Raphson iteration for time-invariant square roots finding can be derived by focusing on the static problem solving (4), discretizing (5) via Euler forward difference, utilizing a linear activation function and fixing the step-size to be 1. Evidently, this paper shows a new explanation to Newton iteration for static square roots finding, which is different from the traditional (or to say, standard) explanation appearing in almost all literature and textbooks, i.e., via Taylor series expansion [1–9].

4. Illustrative examples

The previous sections have proposed the continuous-time and discrete-time ZD models and their links to Newton iteration for online solution of scalar square roots. In this section, computer-simulation results and observations are provided for the substantiation of the proposed link (to Newton method) and efficacy of ZD models with the aid of suitable initial state x_0, a power-sigmoid activation function (with design parameters $\xi = 4$ and $p = 3$) [13–18,22, 24–26] and step-size h.

4.1. Time-varying square roots finding

For illustration and comparison, let us consider the following time-varying square root problem (written in the form of a nonlinear equation):

$$x^2(t) - 4\sin(20t)\cos(16t) - 5 = 0. \tag{9}$$

Both dynamic solvers [i.e., CTZD (2) and GD (3)] are exploited for solving online the time-varying square roots of (9) with design parameter $\gamma = 1$. The computer-simulation results are shown in Fig. 1. Note that, to check the two

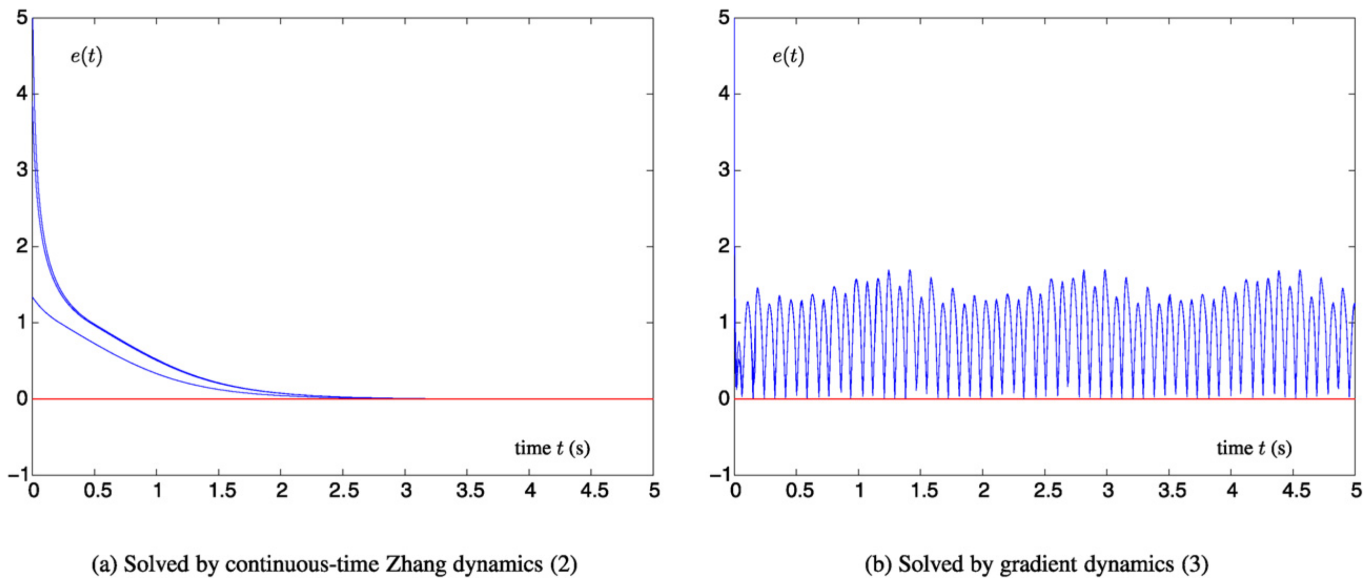

Fig. 2. Convergence performance of residual error $\|x^2(t) - 4\sin(20t)\cos(16t) - 5\|$ solved by CTZD (2) and GD (3) using a power-sigmoid activation function for finding online the time-varying square roots of (9) with design parameter $\gamma = 1$.

dynamic-solvers' effectiveness, the theoretical roots $x_1^*(t)$ and $x_2^*(t)$ [of which, $x_1^*(t) = -x_2^*(t)$] of (9) are shown and denoted via red dash-dotted curves in the figure. As seen from Fig. 1(a), starting from three initial states randomly generated in $[-4, 4]$, the neural state $x(t)$ of ZD model (2) could fit well with the theoretical square root $x_1^*(t)$ or $x_2^*(t)$ after a short time (e.g., 3s or so). In contrast, the neural state $x(t)$ of GD model (3) depicted in Fig. 1(b), also starting from three randomly-generated initial states within $[-4, 4]$, could not fit well with any one of the theoretical square roots [i.e., $x_1^*(t)$ and $x_2^*(t)$] even after a long period of time. Simply put, the steady-state $x(t)$ of GD model (3) lags behind the theoretical square root $x_1^*(t)$ or $x_2^*(t)$ of (9).

Furthermore, in order to further investigate the convergence performance, we could also monitor the residual error $\|e(t)\| = \|x^2(t) - 4\sin(20t)\cos(16t) - 5\|$ during the problem solving process of both dynamic-solvers, ZD (2) and GD (3). Note that symbol $\|\cdot\|$ denotes the Euclidean norm for a vector (which, in our present situation, denotes the absolute value of a scalar argument). It is seen from Fig. 2(a) that, by applying ZD (2) to time-varying square roots finding of (9), the residual error $\|e(t)\|$ could converge to zero around 3 seconds. In contrast, it follows from Fig. 2(b) that, by applying GD (3) to online solution of (9) under the same simulation conditions (e.g., with design parameter $\gamma = 1$ for both dynamic-solvers), its residual error $\|x^2(t) - 4\sin(20t)\cos(16t) - 5\|$ is still quite large even after a long period of time. In our opinion, the reason why ZD does well in online time-varying square roots finding (compared with GD) may be that the time-derivative information of the time-varying coefficient in (9) has been fully utilized in Zhang et al.'s design method and models. In addition, it is worth pointing out that, as shown in Figs. 2(a) and 3, the convergence time of ZD (2) could be expedited from 3 seconds to 0.5 seconds and even 0.05 seconds, as design parameter γ is increased from 1 to 10 and 100, respectively. Moreover, if $\gamma = 10^4$, the convergence time is 0.5 milliseconds. This may show that ZD (2) has an exponential-convergence property, which can be expedited by increasing the value of γ.

Thus, we could summarize that the ZD model (2) is more efficient and effective for online time-varying square roots finding, as compared with the conventional GD model (3).

4.2. Constant square roots finding

Consider the following static square root problem

$$f(x) = x^2 - 9 = 0. \tag{10}$$

Evidently, the theoretical square roots of problem (10) are $x_1^* = 3$ and $x_2^* = -3$ for the verification of S-CTZD (5) solutions. As seen from Fig. 4(a), starting from randomly-generated initial states within $[-4, 4]$, the neural state $x(t)$ of S-CTZD model (5) could converge to a theoretical square root $x_1^* = 3$ or $x_2^* = -3$ rapidly and accurately. In addition, by employing the DTZD model (6) and Newton–Raphson iteration (8) to find the static square roots of (10), the computer-simulation results are shown in Fig. 4(b), where four initial states $x_0 = \pm 3.5, \pm 2.5$ are chosen for ZD (6) and Newton–Raphson iteration (8) to start. As shown in Fig. 4(b), setting step-size $h = 0.1$ [19] for DTZD model (6), we can find that the DTZD's neural states (denoted by red symbols "$*$") could converge to a theoretical square root (i.e., $x_1^* = 3$ or $x_2^* = -3$) after an appropriate number of iterations. In addition, the solutions of Newton–Raphson iteration (8) could also converge to the theoretical square roots of problem (10), which are denoted by black symbols "$\star$" in Fig. 4(b). This point confirms the new explanation to Newton iteration. In other words, the Newton–Raphson iteration could be viewed as a special case of DTZD (6) when the linear activation function and step-size $h = 1$ are used.

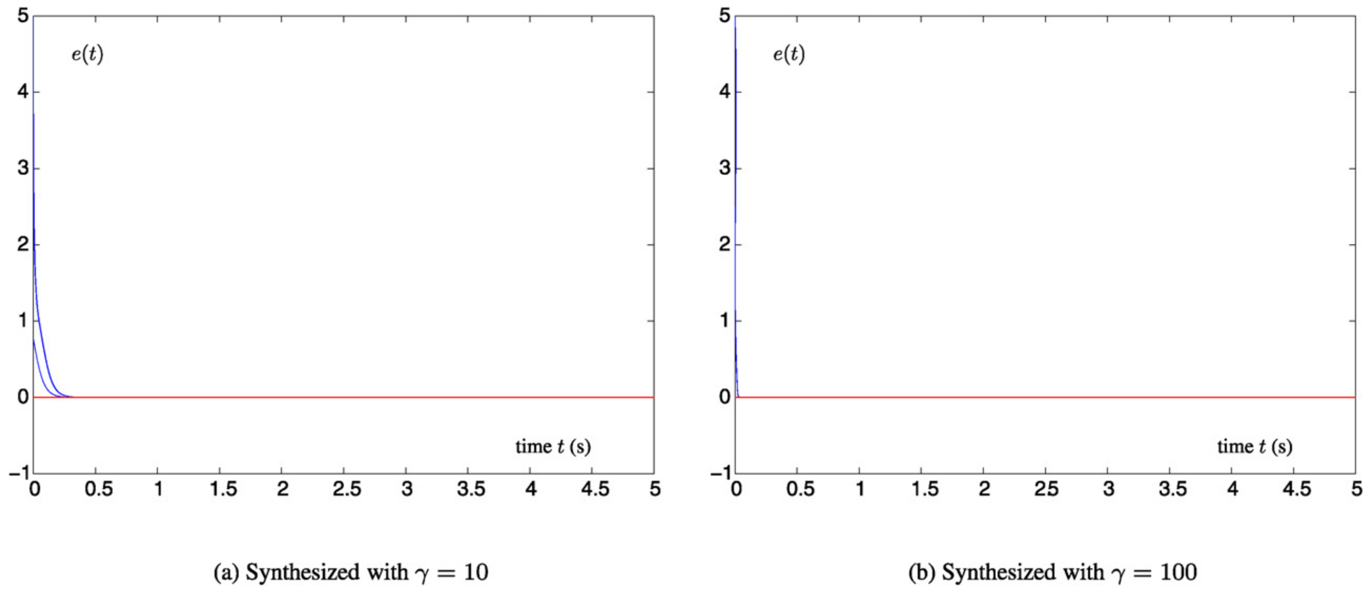

Fig. 3. Convergence performance of residual error $\|x^2(t) - 4\sin(20t)\cos(16t) - 5\|$ solved by CTZD (2) via a power-sigmoid activation function for finding online the time-varying square roots of (9) with design parameter $\gamma = 10$ and $\gamma = 100$.

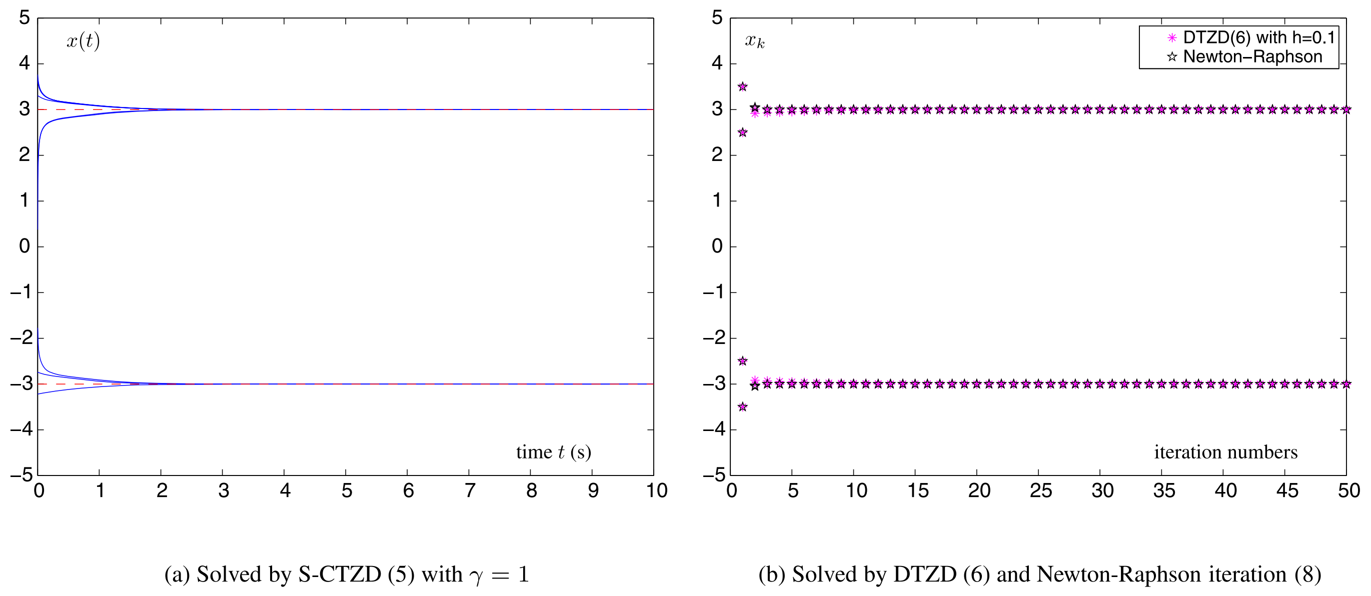

Fig. 4. Time-invariant square roots finding of (10) by S-CTZD (5), DTZD (6) and Newton–Raphson iteration (8).

5. Conclusions

In this paper, a special kind of neural dynamics (termed Zhang dynamics, ZD) has been developed and investigated for time-varying square roots finding in real time. Such a neural dynamics has been elegantly introduced by defining an indefinite error-monitoring function instead of the traditional positive (or at least low-bounded) energy function usually associated with gradient dynamics (GD). In addition, we have generated the discrete-time ZD (DTZD) model for time-invariant square roots finding, and discovered that such DTZD models contain Newton iteration as a special case in some sense, presenting a new reasonable interpretation about Newton iteration. Computer-simulation results via two examples further demonstrate the efficacy of ZD solvers on the scalar square roots finding, as well as the discrete-time model's link and new explanation to Newton–Raphson iteration.

References

[1] J.H. Mathews, K.D. Fink, Numerical Methods Using MATLAB, Publishing House of Electronics Industry, Beijing, BJ, CHN, 2005.

[2] C. Lin, Numerical Computation Methods, Science Press, Beijing, BJ, CHN, 2005.

[3] J. Feng, G. Che, Y. Nie, Principles of Numerical Analysis, Science Press, Beijing, BJ, CHN, 2001.

[4] N.J. Higham, Stable iterations for the matrix square root, Numerical Algorithms 15 (2) (1997) 227–242.

[5] D. Takahashi, Implementation of multiple-precision parallel division and square root on distributed-memory parallel computers, in: Proceedings of the Parallel Processing, 2000, pp. 229–235.

[6] F. Kong, Z. Cai, J. Yu, D.X. Li, Improved generalized Atkin algorithm for computing square roots in finite fields, Information Processing Letters 98 (1) (2006) 1–5.

[7] S. Abbasbandy, Improving Newton–Raphson method for nonlinear equations by modified Adomian decomposition method, Applied Mathematics and Computation 145 (2–3) (2003) 887–893.

[8] J.B. Qian, C.A. Wang, How much precision is needed to compare two sums of square roots of integers, Information Processing Letters 100 (5) (2006) 194–198.

[9] M. Basto, V. Semiao, F.L. Calheiros, A new iterative method to compute nonlinear equations, Applied Mathematics and Computation 173 (1) (2006) 468–483.

[10] Y. Zhang, W.E. Leithead, D.J. Leith, Time-series Gaussian process regression based on Toeplitz computation of $O(N^2)$ operations and $O(N)$-level storage, in: Proceedings of the 44th IEEE Conference on Decision and Control, 2005, pp. 3711–3716.

[11] C. Mead, Analog VLSI and Neural Systems, Addison–Wesley, Reading, MA, USA, 1989.

[12] Y. Zhang, W. Ma, K. Li, C. Yi, Brief history and prospect of coprocessors, China Academic Journal Electronic Publishing House 13 (2008) 115–117.

[13] Y. Zhang, C. Yi, D. Guo, J. Zheng, Comparison on Zhang neural dynamics and gradient-based neural dynamics for online solution of nonlinear time-varying equation, Neural Computing and Applications, in press, doi:10.1007/s00521-010-0452-y.

[14] Y. Zhang, C. Yi, W. Ma, Comparison on gradient-based neural dynamics and Zhang neural dynamics for online solution of nonlinear equations, in: Lecture Notes in Computer Science, vol. 5370, 2008, pp. 269–279.

[15] Y. Zhang, Z. Li, Zhang neural network for online solution of time-varying convex quadratic program subject to time-varying linear-equality constraints, Physics Letters A 373 (18-19) (2009) 1639–1643.

[16] Y. Zhang, W. Ma, B. Cai, From Zhang neural network to Newton iteration for matrix inversion, IEEE Transactions on Circuits and Systems 56 (7) (2009) 1405–1415.

[17] The MathWorks Inc., Simulink 7 Getting Started Guide, Natick, MA, 2008.

[18] Y. Zhang, Revisit the analog computer and gradient-based neural system for matrix inversion, in: Proceedings of the IEEE International Symposium on Intelligent Control, 2005, pp. 1411–1416.

[19] Y. Zhang, P. Xu, N. Tan, Solution of nonlinear equations by continuous- and discrete-time Zhang dynamics and more importantly their links to Newton iteration, in: Proceedings of the IEEE International Conference Information, Communications and Signal Processing, 2009, pp. 1–5.

[20] Y. Zhang, Towards piecewise-linear primal neural networks for optimization and redundant robotics, in: Proceedings of the IEEE International Conference on Networking, Sensing and Control, 2006, pp. 374–379.

[21] Y. Zhang, D. Jiang, J. Wang, A recurrent neural network for solving Sylvester equation with time-varying coefficients, IEEE Transactions on Neural Networks 13 (5) (2002) 1053–1063.

[22] Y. Zhang, S.S. Ge, Design and analysis of a general recurrent neural network model for time-varying matrix inversion, IEEE Transactions on Neural Networks 16 (6) (2005) 1477–1490.

[23] Y. Zhang, A set of nonlinear equations and inequalities arising in robotics and its online solution via a primal neural network, Neurocomputing 70 (2006) 513–524.

[24] Y. Zhang, Y. Yang, Simulation and comparison of Zhang neural network and gradient neural network solving for time-varying matrix square roots, in: Proceedings of the 2nd International Symposium on Intelligent Information Technology Application, 2008, pp. 966–970.

[25] Y. Zhang, H. Peng, Zhang neural network for linear time-varying equation solving and its robotic application, in: Proceedings of the 6th International Conference on Machine Learning and Cybernetics, 2007, pp. 3543–3548.

[26] Y. Zhang, Z. Chen, K. Chen, B. Cai, Zhang neural network without using time-derivative information for constant and time-varying matrix inversion, in: Proceedings of the International Joint Conference on Neural Networks, 2008, pp. 142–146.

[27] S.K. Mitra, Digital Signal Processing-A Computer-Based Approach, 3rd ed., Tsinghua University Press, Beijing, BJ, CHN, 2006.