Available online at www.sciencedirect.com
SCIENCE DIRECT
Neural Networks 17 (2004) 1003–1015
Neural Networks
www.elsevier.com/locate/neunet

A recurrent neural network with exponential convergence for solving convex quadratic program and related linear piecewise equations

Youshen Xia[a,*], Gang Feng[a], Jun Wang[b]

[a]Department of Manufacturing Engineering and Engineering Management, The City University of Hong Kong, Hong Kong, China
[b]Department of Automation and Computer-Aided Engineering, The Chinese University of Hong Kong, Hong Kong, China

Received 13 March 2003; revised 20 May 2004; accepted 20 May 2004

Abstract

This paper presents a recurrent neural network for solving strict convex quadratic programming problems and related linear piecewise equations. Compared with the existing neural networks for quadratic program, the proposed neural network has a one-layer structure with a low model complexity. Moreover, the proposed neural network is shown to have a finite-time convergence and exponential convergence. Illustrative examples further show the good performance of the proposed neural network in real-time applications.

Keywords: Neural network; Finite-time convergence; Exponential convergence; Piecewise equation; Convex quadratic program

1. Introduction

We are concerned with the following quadratic programming problem

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}x^{\mathrm{T}}Qx + c^{T}x \\ &\text{subject to} \quad b^1 \le Ax \le b^2,\ d^0 \le x \le h^0 \end{aligned} \tag{1}$$

where $Q \in R^{n\times n}$ is a symmetric and positive definite matrix, $A \in R^{m\times n}$, h^0, $d^0 \in R^n$, b^1, $b^2 \in R^m$, and $c \in R^n$. Since the objective function is strictly convex, the problem (1) has a unique optimal solution. The problem (1) can include many other quadratic programming problems. For example, let us consider the following quadratic programming problem

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}x^{\mathrm{T}}Qx + c^{\mathrm{T}}x \\ &\text{subject to} \quad Bx = b,\ Dx \le d,\ d^0 \le x \le h,^0 \end{aligned} \tag{2}$$

where $B \in R^{l\times n}$, $D \in R^{r\times n}$, $d \in R^r$, $b \in R^l$, and Q, d^0, h^0, c is defined in Eq. (1), respectively. Let

$$A = \begin{pmatrix} B \\ D \end{pmatrix}, \qquad b^1 = \begin{pmatrix} b \\ p \end{pmatrix}, \qquad b^2 = \begin{pmatrix} b \\ d \end{pmatrix},$$

and $m = l + r$, where $p = [-\infty, \ldots, -\infty] \in R^r$. Then the problem (2) can be transformed into the problem with the form (1). The quadratic programming problem arises in a wide variety of scientific and engineering applications (Bazaraa, Sherali, & Shetty, 1993) including regression analysis, image and signal processing, parameter estimation, filter design, robot control, etc. In many real-time applications these optimization problems have a time-varying nature, they have to be solved in real time. One typical application of real-time optimization in robotics is for robot motion control (Yoshikawa, 1990). Another typical application of real-time optimization in signal processing is for adaptive beamforming (Kalouptisidis, 1997). Because of the nature of digital computers, conventional numerical optimization algorithms may not be competent. As parallel computational models, neural networks possess many desirable properties such as real-time information processing. In particular, recurrent neural networks for optimization, control, and signal processing received tremendous interest (Cichocki & Unbehauen, 1993; Golden, 1996; Sevrani & Abe, 2000; Xia & Wang, 2000, 2001). At present, there are several recurrent neural network approaches to solving quadratic programming problems. Kennedy and Chua (1988) presented a primal neural network for solving Eq. (1). Because the network contains a finite penalty parameter, it converges an approximate solution only. To overcome the penalty parameter, Xia (1996) proposed a primal-dual neural

* Corresponding author. Department of Applied Mathematics, Nanjing University of Posts and Telecommunications, China.
E-mail address: ysxia2001@yahoo.com (Y. Xia).

0893-6080/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2004.05.006

network for solving Eq. (1) and its dual, and Xia and Wang (2001) developed a dual neural network for kinematically redundant manipulators. The primal-dual neural network has a two-layer structure and the dual neural network is used to solve some special convex quadratic programming problems. Moreover, they cannot be shown to have a finite-time convergence and exponential convergence to the optimal solution of Eq. (1). It is well known that designing a neural network with a low complexity and a fast convergence rate (Cichocki & Unbehauen; Sudharsanan & Sundareshan, 1991; Zhang, Heng, & Fu, 1999) is of importance.

This paper first transforms Eq. (1) into a piecewise equation and then proposes a recurrent neural network for solving the piecewise equation. It is shown that the output trajectory of the proposed neural network has a finite-time convergence and an exponential convergence to the optimal solution of Eq. (1). In addition, unlike the existing primal-dual neural network, the proposed neural network has a one-layer structure with a low model complexity.

2. A piecewise formulation

In this section, using standard optimization techniques (Bazaraa et al., 1993; Kinderlehrer & Stampcchia, 1980) we transform Eq. (1) into a piecewise formulation.

First, let

$$E = \begin{pmatrix} A \\ I \end{pmatrix}, \qquad d = \begin{pmatrix} b^1 \\ d^0 \end{pmatrix}, \qquad h = \begin{pmatrix} b^2 \\ h^0 \end{pmatrix},$$

where $I \in R^{n\times n}$ is an unit matrix. Then the problem (1) can be rewritten as

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}x^{\mathrm{T}}Qx + c^{\mathrm{T}}x \\ &\text{subject to} \quad d \le Ex \le h, \end{aligned} \tag{3}$$

where $d = [d_1, \ldots, d_{n+m}]^{\mathrm{T}}$ and $h = [h_1, \ldots, h_{n+m}]^{\mathrm{T}}$. Consider the Lagrangian function of Eq. (3)

$$L(x, y, \eta) = \frac{1}{2}x^{\mathrm{T}}Qx + c^{\mathrm{T}}x - u^{\mathrm{T}}(Ex - \eta),$$

where $u \in R^{n+m}$ is referred to as the Lagrange multiplier and $\eta \in X = \{u \in R^{n+m} | d \le u \le h\}$. According to the well-known Saddle point theorem (Bazaraa et al., 1993), we see that x^* is an optimal solution of Eq. (3) if and only if there exist u^* and η^* satisfying

$$L(x^*, u, \eta^*) \le L(x^*, u^*, \eta^*) \le L(x, u^*, \eta).$$

That is

$$\begin{aligned} &\frac{1}{2}(x^*)^{\mathrm{T}}Qx^* + c^{\mathrm{T}}x^* - u^{\mathrm{T}}(Ex^* - \eta^*) \\ &\le \frac{1}{2}(x^*)^{\mathrm{T}}Qx^* + c^{\mathrm{T}}x^* - (u^*)^{\mathrm{T}}(Ex^* - \eta^*) \\ &\le \frac{1}{2}x^{\mathrm{T}}Qx + c^{\mathrm{T}}x - (u^*)^{\mathrm{T}}(Ex - \eta), \end{aligned} \tag{4}$$

$\forall x \in R^n,\ u \in R^{n+m},\ \eta \in X.$

From the first inequality in Eq. (4) we obtain

$$(u - u^*)^{\mathrm{T}}(Ex^* - \eta^*) \ge 0, \qquad \forall u \in R^{n+m}.$$

Then $Ex^* = \eta^*$. From the second inequality in Eq. (4) we get

$$f(x^*) - f(x) \le (u^*)^{\mathrm{T}}(\eta^* - \eta) \qquad \forall x \in R^n,\ \eta \in X,$$

where $f(x) = \frac{1}{2}x^{\mathrm{T}}Qx + c^{\mathrm{T}}x - (u^*)^{\mathrm{T}}Ex$. If there exists $\hat{x} \in R^n$ such that $f(x^*) - f(\hat{x}) > 0$, then

$$0 < (u^*)^{\mathrm{T}}(\eta^* - \eta) \qquad \forall \eta \in X,$$

which is contradictive when $\eta = \eta^*$. Thus for any $x \in R^n$ we have $f(x^*) - f(x) \le 0$ and

$$(u^*)^{\mathrm{T}}(\eta - \eta^*) \ge 0, \qquad \forall \eta \in X.$$

From the projection formulation (Kinderlehrer & Stampcchia, 1980) it can be seen that the above inequality can be equivalently represented as

$$\eta^* = P_X(\eta^* - u^*),$$

where $P_X(u) = [P_X(u_1), \ldots, P_X(u_{n+m})]^{\mathrm{T}}$ and for $i = 1, \ldots, n+m$

$$P_X(u_i) = \begin{cases} d_i & u_i < d_i \\ u_i & d_i \le u_i \le h_i \\ h_i & u_i > h_i. \end{cases}$$

On the other side, $f(x^*) \le f(x)$ implies that $\nabla f(x^*) = Qx^* + c - E^{\mathrm{T}}u^* = 0$. Thus x^* is an optimal solution of Eq. (3) if and only if there exist u^* and η^* such that (x^*, u^*, η^*) satisfies

$$\begin{cases} Ex = \eta, \\ Qx + c - E^{\mathrm{T}}u = 0, \\ \eta = P_X(\eta - u). \end{cases}$$

Substituting $\eta = Ex$ and $x = -Q^{-1}(c - E^{\mathrm{T}}u)$ into equation $\eta = P_X(\eta - u)$, we have

$$EQ^{-1}(E^{\mathrm{T}}u - c) = P_X(EQ^{-1}(E^{\mathrm{T}}u - c) - u).$$

Then x^* is an optimal solution of Eq. (3) if and only if there exists u^* such that (x^*, u^*) satisfies

$$\begin{cases} EQ^{-1}E^{\mathrm{T}}u + q = P_X(EQ^{-1}E^{\mathrm{T}}u - EQ^{-1}c - u) \\ x = Ru + a, \end{cases}$$

where $R = Q^{-1}E^{\mathrm{T}}$ and $a = -Q^{-1}c$. Therefore, we have the following proposition.

Proposition 1. *Let u^* be a solution of the following piecewise equation*

$$Wu + q = P_X(Wu + q - u), \tag{5}$$

where $W \in R^{(n+m)\times(n+m)}$ is a matrix and $q \in R^{(n+m)}$ is a vector. If $W = EQ^{-1}E^{\mathrm{T}}$ and $q = -EQ^{-1}c$, then $x^* = Ru^* + a$ is the optimal solution of Eq. (1).

According to Proposition 1, we can see that the optimal solution of Eq. (1) can be obtained by solving the piecewise equation (5).

3. One-layer neural network

We propose a recurrent neural network for solving both Eqs. (1) and (5), whose dynamical equation is defined as follows.

State equation

$$\frac{\mathrm{d}u}{\mathrm{d}t} = \lambda\{P_X(Wu + q - u) - Wu - q\}. \tag{6}$$

Output equation

$$x(t) = Ru(t) + a,$$

where $\lambda > 0$ is a scaling constant, $u(t) \in R^{(n+m)}$ is the state variable, $x(t) \in R^n$ is the output variable, and R, W, q, a are defined in Eq. (5). It can be seen that the state equation described by Eq. (6) can be implemented by an analog circuit with a single-layer structure shown in Fig. 1, where $p = n + m$. The circuit consists of $(n+m)^2 + n + m + 1$ summers, $n + m$ integrators, and $(n+m)^2$ weighted connections. The projection operator $P_X(z_i)$ may be implemented by using a linear piecewise function. As for the convergence of the proposed neural network, we will obtain the following results:

(I) The state trajectory of the proposed neural network in Eq. (6) is globally convergent to a solution of piecewise equation (5) within a finite time if W is symmetric and semidefinite, and is globally exponentially convergent if W is symmetric and definite.

(II) If $W = EQ^{-1}E^{\mathrm{T}}$ and $q = -EQ^{-1}c$, the output trajectory of the proposed neural network converges globally to a unique optimal solution of Eq. (1) within a finite time. Moreover, it has the following convergence rate

$$\|x(t) - x^*\|^2 \le \frac{\gamma}{\lambda(t - t_0)}, \qquad \forall t > t_0,$$

where $\|\cdot\|$ is l_2 norm, γ is a positive constant, and $\lambda > 0$ is defined in Eq. (6).

We now give a comparison between the proposed neural network and existing neural networks for solving Eq. (1). First, Kennedy–Chua neural network contains a penalty parameter, while the proposed neural network overcomes the disadvantage of the penalty parameters. Next, consider

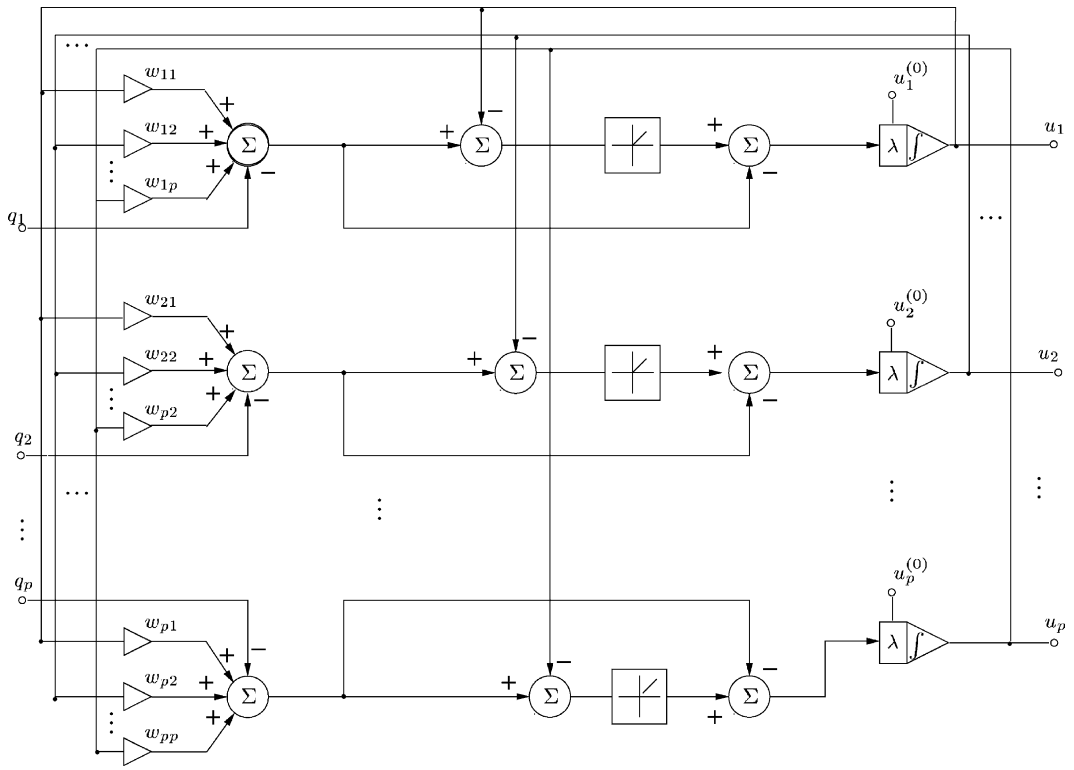

Fig. 1. Architecture of the state equation defined in Eq. (6).

the existing primal-dual neural network for solving Eq. (1) (Xia, 1996). Its dynamical equation is defined as

State equation

$$\frac{\mathrm{d}w}{\mathrm{d}t} = (I + M^{\mathrm{T}})\{P_{\Omega}(w - (Mw + q)) - w\}. \tag{7}$$

Output equation

$$x(t) = [I_n, O]w(t),$$

where $w(t)$ is the state vector and $I_n \in R^{n\times n}$ is an unite matrix, $O \in R^{n\times 2m}$ is a zero matrix, $\Omega = \{w = (x, y, z) \in R^{n+2m} | d \le x \le h, y \ge 0, z \ge 0\}$, and

$$M = \begin{bmatrix} Q & E_1^{\mathrm{T}} \\ -E_1 & 0 \end{bmatrix}, \qquad E_1 = \begin{bmatrix} A \\ -A \end{bmatrix}, \qquad q = \begin{bmatrix} c \\ b^2 \\ -b^1 \end{bmatrix}.$$

It can be seen that the primal-dual neural network has a two-layer structure and the circuit realizing the state equation defined in Eq. (7) consists of $2(n+2m)^2+n+2m$ summers, $n+2m$ integrators, $2(n+2m)^2$ weighted connections. Therefore, the proposed neural network has a lower complexity than the primal-dual neural network. Finally, unlike all existing neural networks for solving Eq. (1), the proposed neural network is shown to have a fast convergence rate including the finite-time convergence and the exponential convergence. Illustrative examples in Section 5 further show that the proposed neural network for solving Eq. (1) has a faster convergence rate than existing neural networks and related optimization methods.

4. Convergence results

In this section, we prove that the proposed neural network has a fast convergence rate, including the finite-time convergence and exponential convergence. A definition and two lemmas are first introduced.

Definition 1. A neural network is said to have a finite-time convergence to the optimal solution x^* of Eq. (1) if there exists a time τ_0 such that the output trajectory $x(t)$ of this network reaches x^* for $t \ge \tau_0$. A neural network is said to be exponentially convergent to the optimal solution x^* of Eq. (1) if the output trajectory $x(t)$ of this network satisfies

$$\|x(t) - x^*\| \le c_0\, \mathrm{e}^{-\eta(t-t_0)}, \qquad \forall t \ge t_0,$$

where η is a positive constant independent of the initial point and c_0 is a positive constant dependent on the initial point.

Lemma 1. *Let $\Omega \subset R^{\mathrm{N}}$ be a closed convex set. Then*

$$(v - P_{\Omega}(v))^{\mathrm{T}}(P_{\Omega}(v) - x) \ge 0, \qquad v \in R^{\mathrm{N}},\ x \in \Omega,$$

and

$$\|P_{\Omega}(u) - P_{\Omega}(v)\| \le \|u - v\|, \qquad v,\ u \in R^{\mathrm{N}},$$

where $P_{\Omega}(u)$ is a projection function on Ω, given by

$$P_{\Omega}(u) = \arg \min_{v\in\Omega} \|u - v\|.$$

Proof. See Kinderlehrer and Stampcchia (1980). □

Lemma 2. (i) *For any initial point there exists a unique continuous solution $u(t)$ for Eq. (6).* (ii) *Assume that W is positive semidefinite. Then the set of equilibrium points of Eq. (6) is nonempty.*

Proof. (i) By Lemma 1 we see that the right-hand term of Eq. (6) is Lipschitz continuous. Thus, for any initial point there exists a unique continuous solution $u(t)$ for Eq. (6). (ii) According to the projection theorem (Kinderlehrer & Stampcchia, 1980), we know that there exists a solution to Eq. (5) and such a solution to an equilibrium point of Eq. (6), and thus the set of equilibrium points of Eq. (6) is nonempty. □

We now establish the main results on the proposed neural network.

Theorem 1. *If W is symmetric and positive definite, then the state trajectory of the proposed neural network is globally convergent to a solution of Eq.* (5) *within a finite time when the designing parameter λ is large enough. Moreover, if W is positive definite, then the state trajectory of the proposed neural network is globally exponentially convergent to a unique solution of Eq.* (5).

Proof. Let $u(t)$ be the trajectory of the state equation defined in Eq. (6) with any given initial point $u(t_0) = u_0$. Consider the following Lyapunov function

$$V(u) = \tfrac{1}{2}\|G(u - u^*)\|^2,$$

where u^* is an equilibrium point of Eq. (6) and $G \in R^{(n+m)\times(n+m)}$ is a symmetric and positive definite matrix satisfying $G^{\mathrm{T}}G = (I + W)$. Then

$$\frac{\mathrm{d}}{\mathrm{d}t}V(u) = \left(\frac{\mathrm{d}V}{\mathrm{d}u}\right)^{\mathrm{T}}\frac{\mathrm{d}u}{\mathrm{d}t}$$

$$= \lambda(u - u^*)^{\mathrm{T}}G^2\{P_X(Wu + q - u) - Wu - q\}.$$

Because u^* satisfies $P_{\Omega}(Wu + q - u) - Wu - q = 0$, u^* is a solution to the following problem (Pang & Yao, 1995): finding u^* such that $Wu^* + q \in X$ and

$$(v - Wu^* - q)^{\mathrm{T}}u^* \ge 0, \qquad v \in X.$$

Let $v = P_X(Wu + q - u)$. Then

$$\{P_X(Wu + q - u) - Wu^* - q\}^{\mathrm{T}}u^* \ge 0, \qquad \forall u \in R^{n+m}.$$

On the other hand, by Lemma 1 we have

$$(v - P_X(v))^{\mathrm{T}}(P_X(v) - u) \geq 0, \qquad v \in R^{n+m},\ u \in X.$$

Let $v = Wu + q - u$, and let $u = Wu^* + q$, then

$$[P_X(Wu + q - u) - Wu^* - q]^{\mathrm{T}}[Wu + q - u - P_X(Wu + q - u)] \geq 0, \qquad \forall u \in R^{n+m}.$$

Adding the two resulting inequalities we get

$$\{P_X(Wu + q - u) - Wu^* - q\}^{\mathrm{T}}\{(u^* - u) - P_X(Wu + q - u) + Wu + q\} \geq 0,$$

then

$$\{(u - u^*) + W(u - u^*)\}^{\mathrm{T}}\{P_X(Wu + q - u) - Wu - q\} \leq -(W(u - u^*))^{\mathrm{T}}(u - u^*) - \|Wu - q - P_X(Wu + q - u)\|^2.$$

It follows

$$(u - u^*)^{\mathrm{T}}(I + W)\{P_X(Wu + q - u) - Wu - q\} \leq -(u - u^*)^{\mathrm{T}}W(u - u^*) - \|P_X(Wu + q - u) - Wu - q\|^2.$$

Thus

$$\frac{\mathrm{d}}{\mathrm{d}t}V(u) = \lambda(u - u^*)^{\mathrm{T}}(I + W)\{P_X(Wu + q - u) - Wu - q\} \leq -\lambda(u - u^*)^{\mathrm{T}}W(u - u^*) - \lambda\|P_X(Wu + q - u) - Wu - q\|^2.$$

Since $\mathrm{d}V/\mathrm{d}t = 0$ if and only if $\mathrm{d}u/\mathrm{d}t = 0$ and $V(u) \geq \alpha\|u - u^*\|^2/2$, where $\alpha > 0$ is the minimal eigenvalue of $I + W$, the proposed neural network is globally convergent to its equilibrium point. We now show that the convergence time is finite. Suppose that the initial point u_0 is not an equilibrium point of Eq. (6). Then

$$\|P_X(Wu_0 + q - u_0) - Wu_0 - q\|^2 > 0.$$

Define

$$f(u(t)) = \|P_X(Wu(t) + q - u(t)) - Wu(t) - q\|^2, \quad \forall t \geq t_0.$$

Then $f(u(t)) \geq 0$ and $f(u(t_0)) > 0$. Since $f(u(t))$ is continuous, there exist $\tau > 0$ and $\delta > 0$ such that for any $t \in [t_0, t_0 + \tau)$, $f(u(t)) \geq \delta$. On the other side, from

$$\frac{\mathrm{d}V(u(t))}{\mathrm{d}t} \leq -\lambda\|P_X(Wu + q - u) - Wu - q\|^2,$$

it follows that for any $t \geq t_0 + \tau$

$$\begin{aligned} V(u(t)) &\leq V(u(t_0)) - \lambda\int_{t_0}^{t} \|P_X(Wu(s) + q - u(s)) - Wu(s) - q\|^2 \mathrm{d}s \\ &\leq V(u(t_0)) - \lambda\int_{t_0}^{\tau + t_0} \|P_X(Wu(s) + q - u(s)) - Wu(s) - q\|^2 \mathrm{d}s \\ &= V(u(t_0)) - \lambda\int_{t_0}^{\tau + t_0} f(u(s))\mathrm{d}s \leq V(u(t_0)) - \lambda\delta\tau. \end{aligned}$$

By taking

$$\lambda = \frac{V(u(t_0))}{\delta\tau},$$

we see that when $t \geq t_0 + \tau$,

$$V(u(t)) \leq 2V(u(t_0)) - 2\lambda\delta\tau = 0.$$

Thus $V(u(t)) = 0$ for any $t \geq t_0 + \tau$ and hence $u(t) = u^*$ for any $t \geq t_0 + \tau$. So, the state trajectory of the proposed neural network is globally convergent to a solution of Eq. (5) within a finite time.

Furthermore, when W is positive definite, there exists a $\mu > 0$ such that $(u(t) - u^*)^{\mathrm{T}}W(u(t) - u^*) \geq \mu\|u(t) - u^*\|^2$. Since

$$\frac{\mathrm{d}}{\mathrm{d}t}V(u) \leq -\lambda(u - u^*)^{\mathrm{T}}W(u - u^*),$$

$$\frac{\mathrm{d}}{\mathrm{d}t}V(u) \leq -\lambda\mu\|u(t) - u^*\|^2 \leq -\frac{2\lambda\mu}{\alpha}V(u).$$

It follows that

$$\|x(t) - x^*\|^2 \leq 2V(u(t) \leq 2V(u(t_0))\mathrm{e}^{-\hat{\mu}(t - t_0)}, \qquad \forall t \geq t_0,$$

where $\hat{\mu} = 2\lambda\mu/\alpha$. This proof is completed. □

Theorem 2. *Assume that $W = EQ^{-1}E^{\mathrm{T}}$ and $q = -EQ^{-1}c$. The output trajectory of the proposed neural network converges globally to a unique optimal solution of Eq.* (1) *within a finite time. Moreover, it has the following convergence rate*

$$\|x(t) - x^*\|^2 \leq \frac{\gamma}{\lambda(t - t_0)}, \qquad \forall t > t_0, \tag{8}$$

where γ is a positive constant.

Proof. Let $u(t)$ be the trajectory of the state equation defined in Eq. (6) with any given initial point $u(t_0) = u_0$. Since $W = EQ^{-1}E^{\mathrm{T}}$ is symmetric and semidefinite, by Theorem 1 we see that $u(t)$ converges to an equilibrium point of Eq. (6) within a finite time. From Proposition 1 it follows that the output trajectory $x(t)$ of the proposed neural network converges globally to a unique optimal solution of Eq. (1) within a finite time. In order to estimate the convergence

rate of $x(t)$, we let $u(t) - u^* = Pz(t)$, where

$$P = \begin{pmatrix} I_1 & O \\ -A^{\mathrm{T}} & I_2 \end{pmatrix},$$

$I_1 \in R^{m\times m}$ is a unit matrix, $I_2 \in R^{n\times n}$ is a unit matrix, and $O \in R^{m\times n}$ is a zero matrix. Then

$$(u(t) - u^*)^{\mathrm{T}} W(u(t) - u^*) = z(t)^{\mathrm{T}} P^{\mathrm{T}} W P z(t).$$

Since

$$W = \begin{pmatrix} A \\ I \end{pmatrix} Q^{-1}(A^{\mathrm{T}}, I) = \begin{pmatrix} AQ^{-1}A^{\mathrm{T}} & AQ^{-1} \\ Q^{-1}A^{\mathrm{T}} & Q^{-1} \end{pmatrix},$$

$$P^{\mathrm{T}} W P = \begin{pmatrix} I_1 & -A \\ O & I_2 \end{pmatrix} \begin{pmatrix} AQ^{-1}A^{\mathrm{T}} & AQ^{-1} \\ Q^{-1}A^{\mathrm{T}} & Q^{-1} \end{pmatrix} \begin{pmatrix} I_1 & O \\ -A^{\mathrm{T}} & I_2 \end{pmatrix}$$

$$= \begin{pmatrix} O_1 & O \\ O^{\mathrm{T}} & Q^{-1} \end{pmatrix},$$

where $O_1 \in R^{m\times m}$ is a zero matrix. It follows

$$(u(t) - u^*)^{\mathrm{T}} W(u(t) - u^*) = z_{II}(t)^{\mathrm{T}} Q^{-1} z_{II}(t),$$

where $z(t) = [z_I(t), z_{II}(t)]^{\mathrm{T}}$, $z_I(t) \in R^m$, and $z_{II}(t) \in R^n$. According to the analysis of Theorem 1 we see that

$$V(u(t)) - V(u(t_0)) \le -\lambda \int_{t_0}^{t} (u(t) - u^*)^{\mathrm{T}} W(u(t) - u^*) \mathrm{d}s,$$

and

$$\alpha \|u(t) - u^*\|^2 \le 2V(u(t_0)) - 2\lambda \int_{t_0}^{t} (u(t) - u^*)^{\mathrm{T}} W(u(t) - u^*) \mathrm{d}s.$$

Then

$$\alpha \|u(t) - u^*\|^2 \le 2V(u(t_0)) - 2\lambda \int_{t_0}^{t} z_{II}(t)^{\mathrm{T}} Q^{-1} z_{II}(t) \mathrm{d}s.$$

Let $\delta_1 > 0$ be the minimal eigenvalue of Q^{-1}. Then

$$\alpha \|u(t) - u^*\|^2 \le 2V(u(t_0)) - 2\lambda\delta_1 \int_{t_0}^{t} \|z_{II}(t)\|^2 \mathrm{d}s.$$

Let $\delta_2 > 0$ be the minimal eigenvalue of P^2 and note that $u(t) - u^* = Pz(t)$. Then

$$\delta_2 \alpha \|z_{II}(t)\|^2 \le \alpha\delta_2 \|z(t)\|^2 \le 2V(u(t_0)) - 2\lambda\delta_1 \int_{t_0}^{t} \|z_{II}(t)\|^2 \mathrm{d}s.$$

Thus

$$\|z_{II}(t)\|^2 \le c_1 - \beta \int_{t_0}^{t} \|z_{II}(t)\|^2 \mathrm{d}s,$$

where

$$c_1 = \frac{2V(u(t_0))}{\delta_2\alpha}, \quad \beta = \frac{2\lambda\delta_1}{\delta_2\alpha}.$$

Then

$$\|z_{II}(t)\|^2 + \beta \int_{t_0}^{t} \|z_{II}(t)\|^2 \mathrm{d}s \le c_1,$$

and thus

$$\int_{t_0}^{t} \|z_{II}(s)\|^2 \mathrm{d}s < c_1/\beta.$$

It follows that for any $t > t_0$

$$\|z_{II}(t)\|^2 < \|z_{II}(\hat{t})\|^2 = \frac{\int_{t_0}^{t} \|z_{II}(s)\|^2 \mathrm{d}s}{t - t_0} \le \frac{c_1}{\beta(t - t_0)},$$

where $t_0 < \hat{t} < t$. Note that $x^* = Q^{-1}(E^{\mathrm{T}}u^* - c)$, then

$$x(t) - x^* = Q^{-1}E^{\mathrm{T}}(u(t) - u^*) = Q^{-1}[A^{\mathrm{T}}, I]Pz(t)$$

$$= Q^{-1}[A^{\mathrm{T}}, I] \begin{pmatrix} I_1 & O \\ -A^{\mathrm{T}} & I_2 \end{pmatrix} \begin{pmatrix} z_I(t) \\ z_{II}(t) \end{pmatrix}$$

$$= Q^{-1}[A^{\mathrm{T}}, I] \begin{pmatrix} z_I(t) \\ -A^{\mathrm{T}}z_I(t) + z_{II}(t) \end{pmatrix} = Q^{-1} z_{II}(t).$$

Thus

$$\|x(t) - x^*\|^2 \le \|Q^{-1}\|^2 \|z_{II}(t)\|^2 \le \|Q^{-1}\|^2 \frac{c_1}{\beta(t - t_0)}, \quad \forall t > t_0.$$

Let $\gamma = \delta_2\alpha c_1/2\delta_1$. Then

$$\|x(t) - x^*\|^2 \le \frac{\gamma}{\lambda(t - t_0)}, \quad \forall t > t_0.$$

This proof is completed. □

5. Illustrative examples

In order to demonstrate the effectiveness and efficiency of the proposed neural network approach, we have implemented it in MATLAB to solve Eq. (1). In addition to the exponential convergence and real-time application of the proposed approach, we compared the performance of this implementation with the Kennedy–Chua neural network (KCNN), the primal-dual neural network (PDNN), and the numerical optimization method, respectively.

Examples 1 and 2 illustrate the exponential convergence of the proposed neural network.

Example 1. Consider the following quadratic programming problem

$$\begin{aligned} \text{minimize } & f(x) = x_1^2 + x_2^2 + x_1x_2 - 30x_1 - 30x_2 \\ & \tfrac{5}{12}x_1 - x_2 \le \tfrac{35}{12}, \\ \text{subject to } & \tfrac{5}{2}x_1 + x_2 \le \tfrac{35}{2}, \ . \\ & -5 \le x_1, x_2 \le 5 \end{aligned}$$

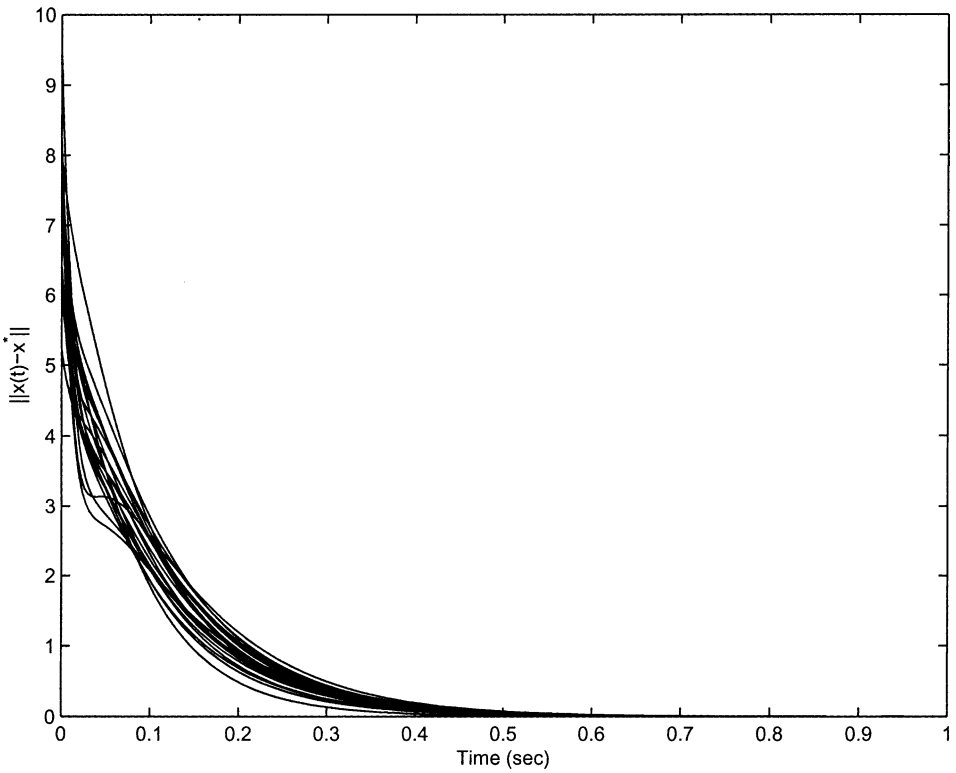

Fig. 2. Convergence behavior of the error $\|x(t) - x^*\|$ in Example 1.

This problem has a unique optimal solution $x^* = [5, 5]^{\mathrm{T}}$ (Tank & Hopfield, 1986). Let

$$Q = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \qquad A = \begin{bmatrix} 5/12 & -1 \\ 5/2 & 1 \end{bmatrix},$$

and let

$$d^0 = \begin{bmatrix} -5 \\ -\infty \end{bmatrix}, \qquad h^0 = \begin{bmatrix} +\infty \\ 5 \end{bmatrix}, \qquad b^1 = \begin{bmatrix} -\infty \\ -\infty \end{bmatrix},$$

$$b^2 = \begin{bmatrix} 35/12 \\ 35/2 \end{bmatrix}.$$

Then the above problem can be converted into the problem with the form (1). We use the proposed neural network to solve the above problem. All simulation results show that the output trajectory $x(t)$ of the proposed neural network is always exponentially convergent to x^*. For example, let $\lambda = 10$. Fig. 2 displays the convergence behavior of the l_2 norm error $\|x(t) - x^*\|$ based on the proposed neural network with 20 random initial points.

Example 2. Consider the following quadratic programming problem

$$\text{minimize } f(x) = x_1^2 + x_2^2 + 0.5x_3^2 + x_1x_2 + x_1x_3 - 4x_1 - 3x_2 - 2x_3, \tag{9}$$

$$\text{subject to } \begin{cases} x_1 + x_2 + 2x_3 \le 3 \\ 3x_1 - 9x_2 + 9x_3 = 1 \\ 0 \le x_i \le 4/3 \ \ (i = 1, 2, 3) \end{cases}.$$

The problem (9) has a unique optimal solution $x^* = [4/3, 7/9, 4/9]^{\mathrm{T}}$ (Cichocki & Unbehauen, 1993). Let

$$Q = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 1 \end{bmatrix}, \qquad A = \begin{bmatrix} 3 & -9 & 9 \\ 1 & 1 & 2 \end{bmatrix},$$

and let

$$c = \begin{bmatrix} -4 \\ -3 \\ -2 \end{bmatrix}, \qquad d^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \qquad h^0 = \begin{bmatrix} 4/3 \\ 4/3 \\ 4/3 \end{bmatrix},$$

$$b^1 = \begin{bmatrix} -\infty \\ 1 \end{bmatrix}, \qquad b^2 = \begin{bmatrix} 3 \\ 1 \end{bmatrix}.$$

Then the problem (9) can be converted into the problem with the form (1). We use the proposed neural network to solve Eq. (9). All simulation results show that the output trajectory $x(t)$ of the proposed neural network is always exponentially convergent to x^*. For example, let $\lambda = 5$. Fig. 3 displays the convergence behavior of the l_2

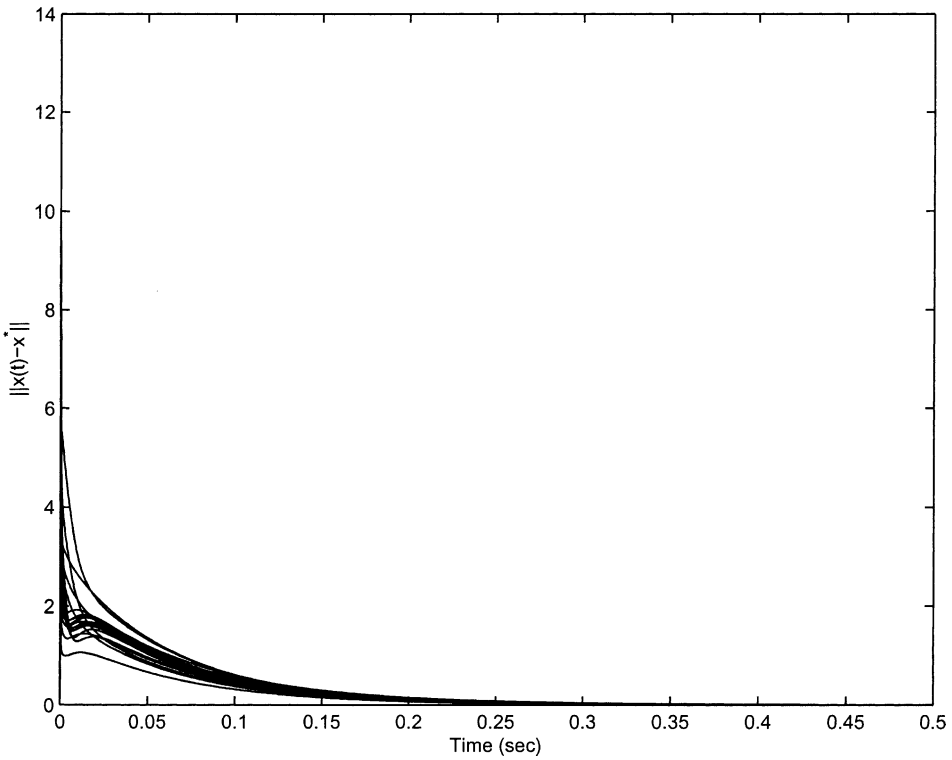

Fig. 3. Convergence behavior of the error $\|x(t) - x^*\|$ in Example 2.

norm error $\|x(t) - x^*\|$ based on the proposed neural network with 20 random initial points.

Example 3 illustrates that the proposed neural network approach has a faster convergence rate than other related methods.

Example 3. Consider the quadratic programming problem (1), where 10×10 matrix

$$Q = \begin{bmatrix} 2 & 1 & 0 & \cdots & 0 \\ 1 & 2 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & 2 & 1 \\ 0 & \cdots & 0 & 1 & 2 \end{bmatrix},$$

$c = [-1, 4, -1, 1, 0, 0, 1, 0, 1, 0]^{\mathrm{T}}$, $b^1 = [1, -1, 0]^{\mathrm{T}}$; $h = [7, 5, 1]^{\mathrm{T}}$, and

$$A = \begin{bmatrix} 1 & -1 & -1 & 1 & -1 & 1 & 0 & 1 & 1 & -1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ -1 & 1 & -1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}.$$

This problem has a unique optimal solution

$x^* = [0.5, -1.5, 0, 1, -1.5, 2, -1, 0.5, 0, 0]^{\mathrm{T}}$.

For a comparison, we compute this example by using the Kennedy–Chua neural network with the penalty parameter being 1000, the primal-dual neural network, the proposed neural network, and the numerical optimization method in MATLAB toolbox, respectively. The computational results are listed in Table 1, where the accuracy is defined by l_2-norm error $\|x(t) - x^*\|$. From Table 1 we see that the proposed neural network not only gives a better solution, but also has a faster convergence rate than other methods.

Example 4 illustrates the real-time application of the proposed neural network in robot motion control.

Example 4. Consider the resolution problem of the following forward kinematics equation

$$r(t) = f(\theta(t)), \tag{10}$$

where $t \in [0, T]$, $\theta(t)$ is an m-vector of joint variables, $r(t)$ is an n-vector of positions and orientations of the end-effector, and $f(\cdot)$ is a continuous nonlinear function with structure and parameters for a given manipulator. Since Eq. (10) is a time-varying nonlinear equations, it is more difficult for one to solve this problem directly. Differentiating Eq. (10) with respect to time yields the following linear relation between joint velocity $\dot{\theta}(t) \in R^m$ and Cartesian velocity $\dot{r}(t)$ of the end-effector

$$\dot{r}(t) = J(\theta(t))\dot{\theta}(t), \tag{11}$$

where $J(\theta(t)) \in R^{n \times m}$ is Jacobian matrix of f and can be rank-deficient of singularity configuration. One resolution problem of kinematically redundant manipulators can be

Table 1
Results for several methods in Example 3, where e is a vector with the element being 1

	Method			
	CUNN method	PDNN method	Optimization method	Proposed method
Initial point	$-10\times e \in R^{10}$	$-10\times e \in R^{16}$	$-10\times e \in R^{10}$	$-10\times e \in R^{3}$
Iterative no.	12564	269	172	80
CPU time (s)	19.8	0.72	0.63	0.4
l_2-norm error	0.37	8.3×10^{-4}	9.9×10^{-4}	3.2×10^{-4}

formulated as

$$\begin{aligned} \text{minimize} \quad & f(\dot{\theta}) = \frac{1}{2}\dot{\theta}^{\mathrm{T}} M\dot{\theta} + z^{\mathrm{T}} M\dot{\theta}, \\ \text{subject to} \quad & J(\theta)\dot{\theta} = \dot{r}(t), \\ & \dot{\theta}^{-} \le \dot{\theta} \le \dot{\theta}^{+}, \end{aligned} \tag{12}$$

where M is an $m\times m$ scaling matrix, the superscripts $+$ and $-$, respectively, denote the upper and lower limits, $z = \alpha(\theta - \theta(0))$, and α is a positive parameter to scale the magnitude of the manipulator response to joint displacement (Cheng, Chen, & Sun, 1994). Because of the time-varying feature in Eq. (12), the resolution problem has to be solved by real-time solution methods. It is easy to see that the above resolution problem can be written as the form of Eq. (1). The proposed neural network in Eq. (6) can be applied to solve the resolution problem.

We simulate a 7-DOF redundant manipulator (PA10) by using the proposed neural network. The PA10 manipulator (Portable General Purpose Intelligent Arm) has seven degrees-of-freedom (three rotation axes and four pivot axes). Joint angle limits $\theta^{\pm}$ and joint rate bounds $\dot{\theta}^{\pm}$ are shown in paper (Zhang, Wang, & Xia, 2003). Simulation has been performed for the path-following task that the end-effector of PA10 manipulator tracks a given circle in the three-dimensional workspace. For simplicity, let the weighting matrix $M = I$ and $\alpha = 4$ in the simulation. We simulate the redundancy resolution problem of the PA10 manipulator for a circular motion with the radius of 0.2 m. The task time of the motion is 10 s, and the initial angular vector is $\theta(0) = [0, -\pi/4, 0, \pi/2, 0, -\pi/4, 0]^{\mathrm{T}}$. Fig. 4 illustrates the transient behavior of the joint variables θ. Fig. 5 illustrates the transient behavior of the joint velocity variables $\dot{\theta}$. Fig. 6 illustrates the transient behavior of the Cartesian position error, where e_x, e_y and e_z denote the components of tracking position error, respectively, along the x, y and z axes of the base frame. Fig. 7 illustrates the transient behavior of the velocity error, where $\dot{e}_x$, $\dot{e}_y$ and $\dot{e}_z$ denote, respectively, the x, y and z-axis components of

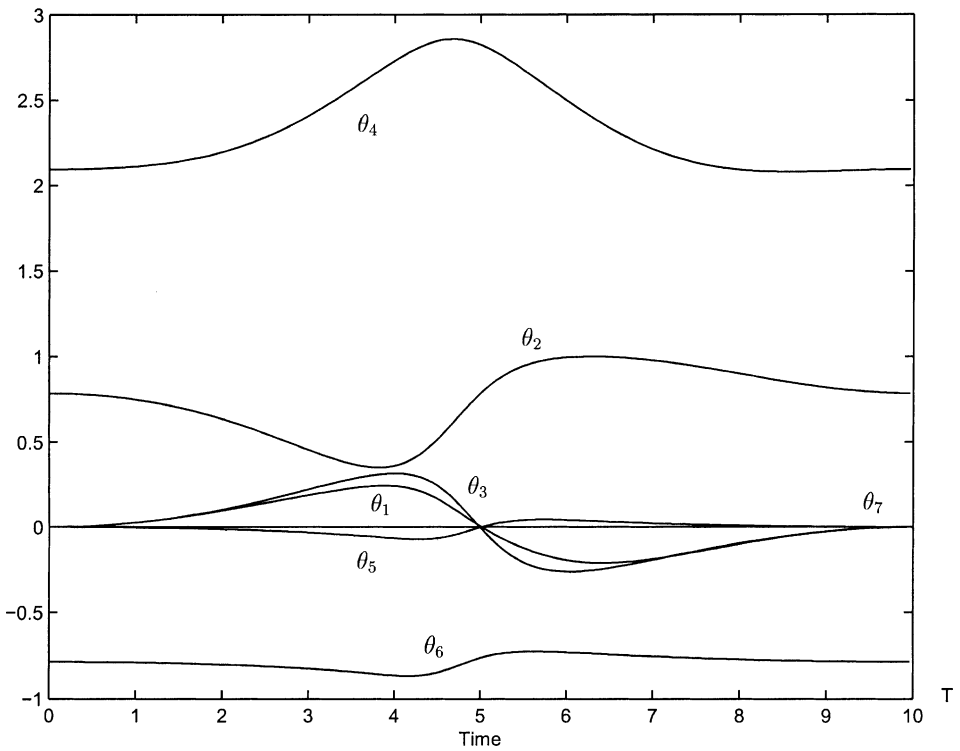

Fig. 4. Joint variables of the PA10 manipulator along a circle in Example 4.

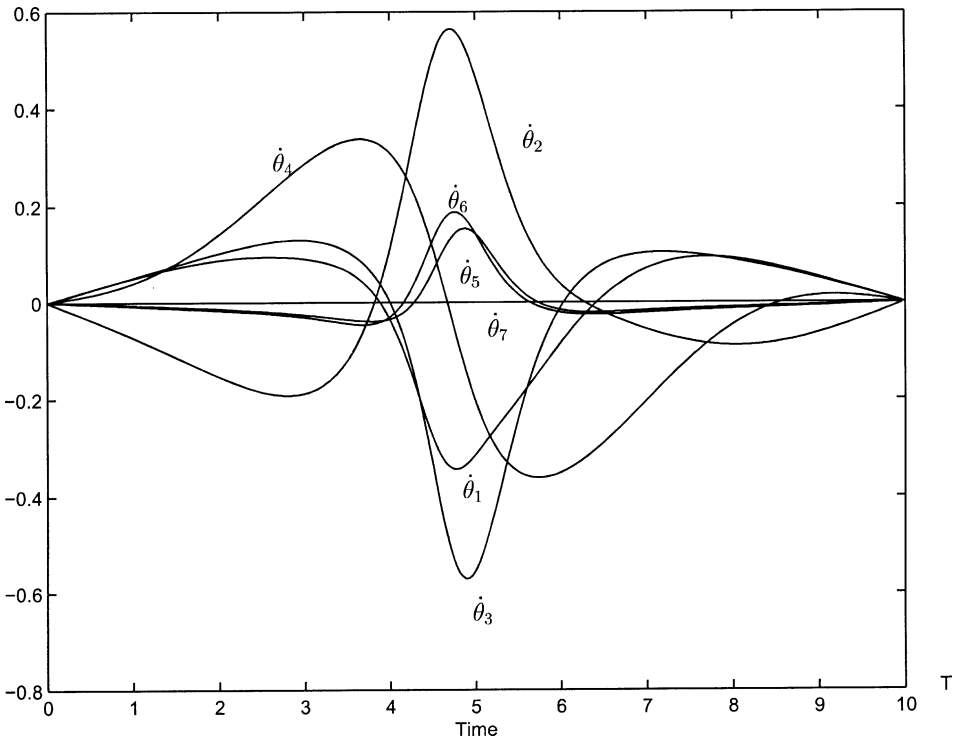

Fig. 5. Joint velocity variables of the PA10 manipulator along a circle in Example 4.

tracking velocity error at the end-effector of PA10 robot arm. For a comparison, the PA10 manipulator is also simulated by the numerical optimization method in MATLAB toolbox. Figs. 8 and 9 show the transient behavior of the velocity and position error based on the optimization method. It can be seen that the proposed neural network obtains the smaller error than the optimization method.

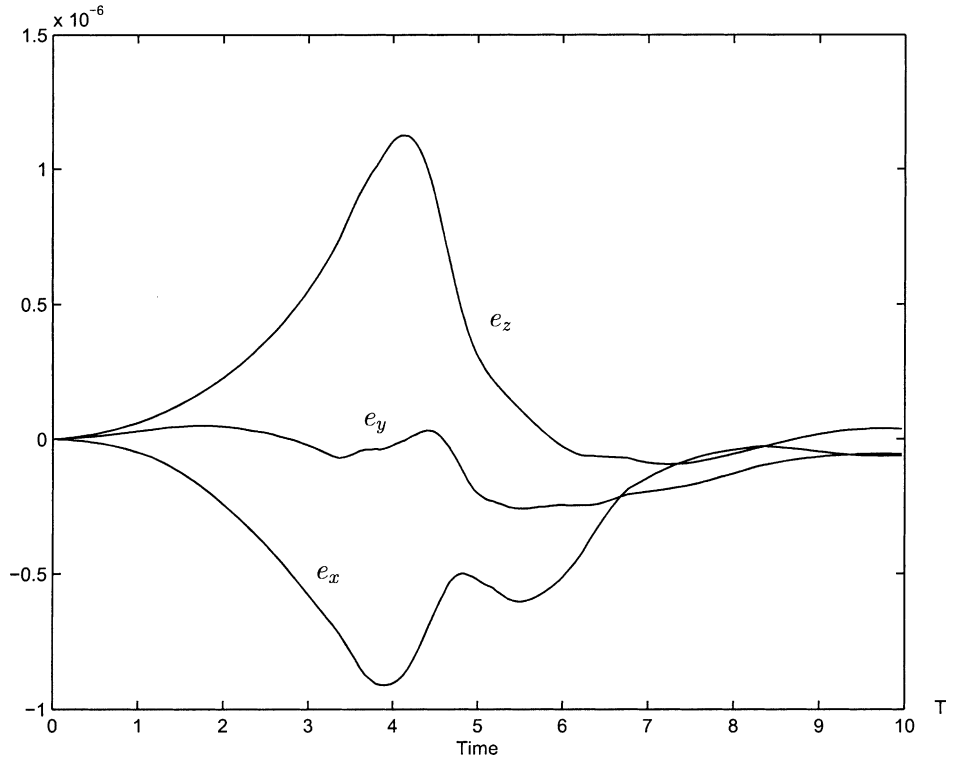

Fig. 6. Position errors of the PA10 manipulator based on the proposed neural network in Example 4.

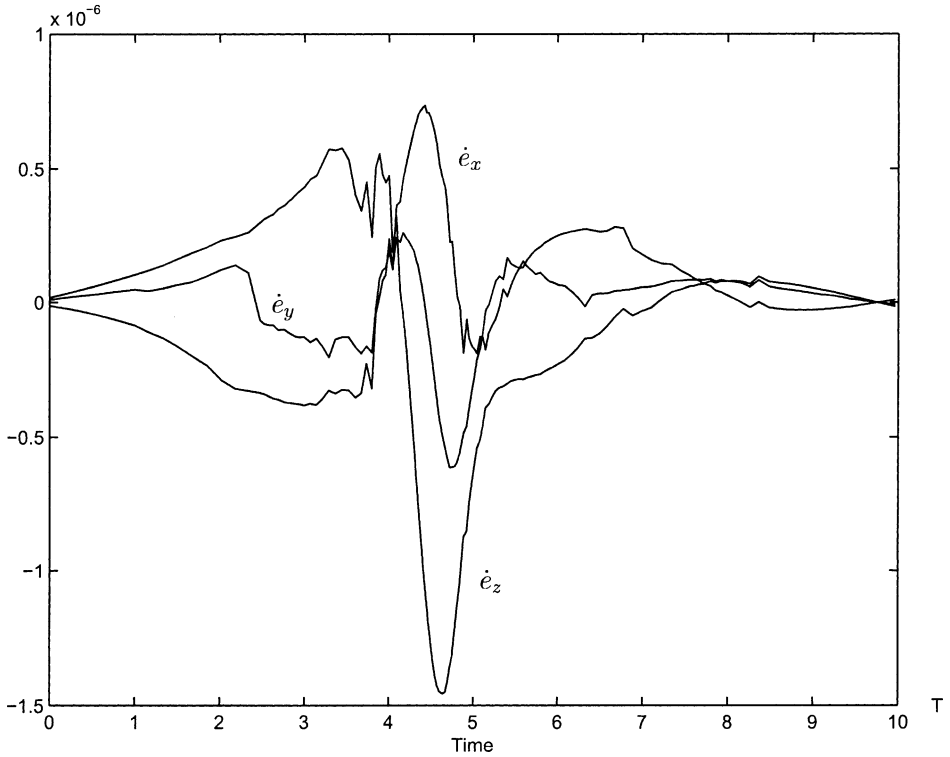

Fig. 7. Velocity errors of the PA10 manipulator based on the proposed neural network in Example 4.

6. Conclusion

In this paper, we have presented a recurrent neural network for solving strict convex quadratic programming problems and related linear piecewise equations. The proposed neural network can easily be implemented by a parallel circuit without analog multipliers for variables or penalty parameters. Compared with existing

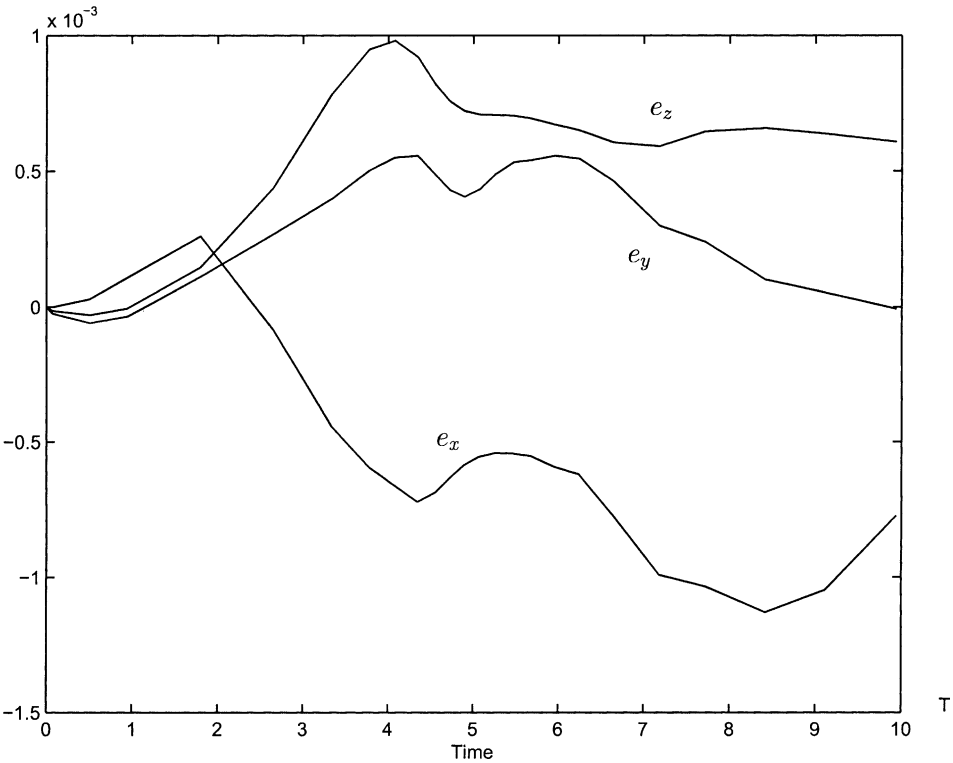

Fig. 8. Position errors of the PA10 manipulator based on the numerical optimization method in Example 4.

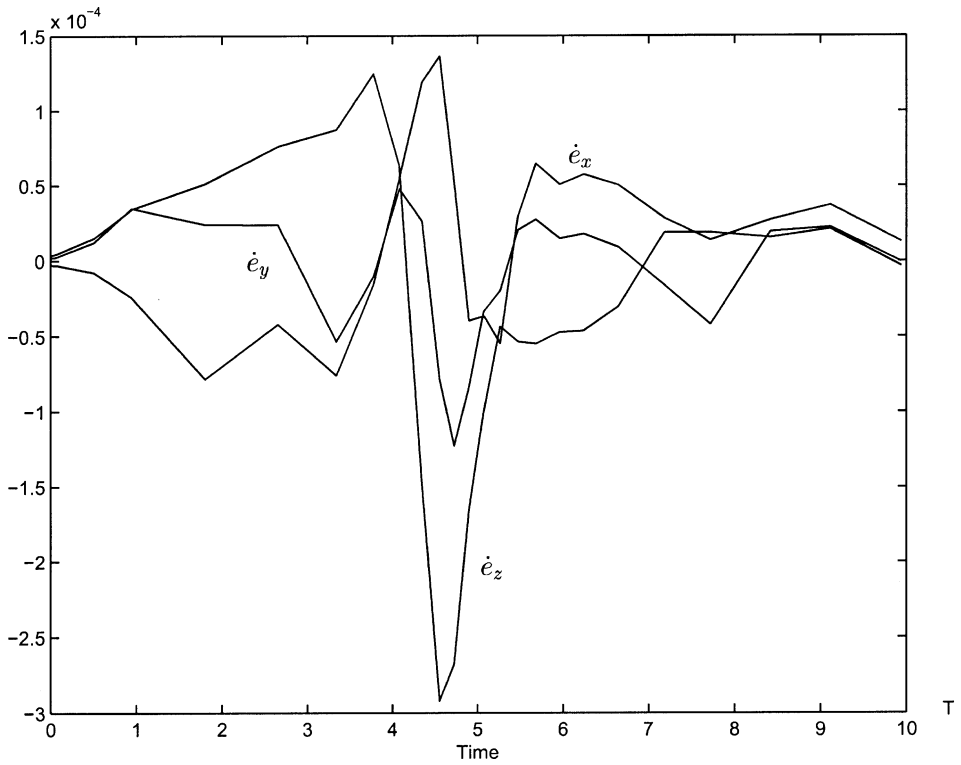

Fig. 9. Velocity errors of the PA10 manipulator based on the numerical optimization method in Example 4.

neural networks for solving such problems, the proposed neural network has a low model complexity. In the theoretical aspect, it is shown that the state trajectory of the proposed neural network is globally convergent to a solution of piecewise equation (5) within a finite time and has an exponential convergence rate, and the output trajectory of the proposed neural network converges globally to a unique optimal solution of Eq. (1) and has a convergence rate in the form of Eq. (8). In the real-time application aspect, the proposed neural network is used directly to solve the resolution problem of kinematically redundant manipulator. Illustrative examples further show that the proposed neural network has a faster convergence rate than existing neural networks and related optimization methods.

Acknowledgements

The authors thank the action editor and reviewers for their valued comments, which helped in improving the quality of the paper. This work was supported by Young Scholar funding from Faculty of Science and Engineering in City University of Hong Kong.

References

Bazaraa, M. S., Sherali, H. D., & Shetty, C. M. (1993). *Nonlinear programming—Theory and algorithms* (2nd ed.). New York: Wiley.

Cheng, F.-T., Chen, T.-H., & Sun, Y.-Y. (1994). Resolving manipulator redundancy under inequality constraints. *IEEE Journal of Robotics and Automation*, *10*(1), 65–71.

Cichocki, A., & Unbehauen, R. (1993). *Neural networks for optimization and signal processing*. England: Wiley.

Golden, R. M. (1996). *Mathematical methods for neural network analysis and design*. Cambridge, MA: MIT Press.

Kalouptisidis, N. (1997). *Signal processing systems, theory and design*. New York: Wiley.

Kennedy, M. P., & Chua, L. O. (1988). Neural networks for nonlinear programming. *IEEE Transactions on Circuits and Systems*, *35*(5), 554–562.

Kinderlehrer, D., & Stampcchia, G. (1980). *An introduction to variational inequalities and their applications*. New York: Academic Press.

Pang, J. S., & Yao, J. C. (1995). On a generalization of a normal map and equation. *SIAM Journal of Control Optimization*, *33*, 168–184.

Sevrani, F., & Abe, K. (2000). On the synthesis of brain-state-in-a-box neural models with application to associative memory. *Neural Computation*, *12*, 451–472.

Sudharsanan, S., & Sundareshan, M. (1991). Exponential stability and a systematics synthesis of a neural network for quadratic minimization. *Neural Networks*, *4*(5), 599–613.

Tank, D. W., & Hopfield, J. J. (1986). Simple neural optimization networks: An A/D converter, signal decision circuit, and a linear

programming circuit. *IEEE Transactions on Circuits and Systems*, *33*(5), 533–541.

Xia, Y. (1996). A new neural network for solving linear and quadratic programming problems. *IEEE Transactions on Neural Networks*, *7*, 1544–1547.

Xia, Y., & Wang, J. (2000). A recurrent neural network for solving linear projection equations. *Neural Networks*, *13*, 337–350.

Xia, Y., & Wang, J. (2001). A dual neural network for kinematic control of redundant robot manipulators. *IEEE Transactions on Systems, Man and Cybernetics—Part B*, *31*(1), 147–154.

Yoshikawa, T. (1990). *Foundations of robotics: Analysis and control*. Cambridge, MA: MIT Press.

Zhang, Y., Heng, P. A., & Fu, A. W. C. (1999). Estimate of convergence rate and exponential stability for neural networks. *IEEE Transactions on Neural Networks*, *10*, 1487–1493.

Zhang, Y., Wang, J., & Xia, Y. S. (2003). A dual neural network for redundancy resolution of kinematically redundant manipulators subject to joint limits and joint velocity limits. *IEEE Transactions on Neural Networks*, *14*, 658–667.