10 June 2002

PHYSICS LETTERS A

ELSEVIER

Physics Letters A 298 (2002) 271–278

www.elsevier.com/locate/pla

A dual neural network for convex quadratic programming subject to linear equality and inequality constraints ☆

Yunong Zhang [1], Jun Wang *

Department of Automation and Computer-Aided Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong

Received 31 October 2001; accepted 1 April 2002

Communicated by A.R. Bishop

Abstract

A recurrent neural network called the dual neural network is proposed in this Letter for solving the strictly convex quadratic programming problems. Compared to other recurrent neural networks, the proposed dual network with fewer neurons can solve quadratic programming problems subject to equality, inequality, and bound constraints. The dual neural network is shown to be globally exponentially convergent to optimal solutions of quadratic programming problems. In addition, compared to neural networks containing high-order nonlinear terms, the dynamic equation of the proposed dual neural network is piecewise linear, and the network architecture is thus much simpler. The global convergence behavior of the dual neural network is demonstrated by an illustrative numerical example. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Dual neural network; Quadratic programming; Linear constraint; Projection operator; Global convergence

1. Introduction

Linearly constrained quadratic programming problems, due to its fundamental role, arise in numerous areas of applications such as manufacturing, economic, social, and public planning. Optimization problems with nonlinear objective functions are usually approximated by a second-order quadratic system and then solved by a standard quadratic programming technique sequentially. The computational complexity of serial-processing algorithms performed on digital computers may limit their usage in large-scale or on-line optimization applications, such as in rigid body mechanics [13], fluid dynamics and elastic-plastic torsion [7].

The dynamical system approach is one of the important methods for solving optimization problems, which was first proposed by Pyne (see [1]) in the late 1950s. Recently, due to the in-depth research in neural networks, numerous dynamic solvers based on neural networks have been developed and investigated (see [2–13]). Specifically, Tank and Hopfield proposed their working neural network implemented on analogue circuits, which opened new avenue for neural networks to solve optimization problems (see [2]). The neural network approach is now thought to be a pow-

☆ This research was supported by the Hong Kong Research Grants Council under Grant CUHK4165/98E.

* Corresponding author.
E-mail address: jwang@acae.cuhk.edu.hk (J. Wang).

[1] The author currently is a Ph.D. student in the Department of Automation and Computer-Aided Engineering, the Chinese University of Hong Kong, Shatin, Hong Kong.

erful tool for real-time optimization, in view of the nature of parallel distributed computation and hardware implementability.

In the past two decades, various neural network models have been developed for solving the linearly constrained quadratic optimization problems, e.g., those based on the penalty-parameter method [3], the Lagrange method [5], the gradient and projected method [7,10], the primal–dual method [9,11], and the dual method [12,13]. It is well-known that the neural network model [3] contains finite penalty parameters and generates approximate solutions only. When solving inequality-constrained quadratic programs, the Lagrange neural network may exhibit the premature defect. In addition, the dimensionality of Lagrange network is much larger than that of original problems, due to the introduction of slack and surplus variables. The gradient and projected method [7,10] were proposed for solving special quadratic program with simple bound constraints only, which cannot be generalized to solve general constrained quadratic programming problems. As a much flexible tool for exactly solving quadratic programs, the primal–dual neural networks [9,11] were developed with the feature that they handle the primal quadratic program and its dual problem simultaneously by minimizing the duality gap with Karush–Kuhn–Tacker condition and gradient method. Unfortunately, the dynamic equations of the primal–dual neural network are usually complicated, and may contain high-order nonlinear terms. Moreover, the network size are usually larger than the dimensionality of the primal quadratic program plus its dual problem.

As stated in [4], to formulate an optimization problem solvable by a neural network, there exist two types of methods. One approach commonly used in developing an optimization neural network is to first convert the constrained optimization problem into an associated unconstrained optimization problem, and then design a neural network that solves the unconstrained problem with a gradient descent. The other approach is to construct a set of differential equations such that their equilibrium points correspond to the desired solutions and then find an appropriate Lyapunov function such that all trajectories of the systems converges to equilibrium points. As a special case of the primal–dual neural network, the dual neural network is proposed by using the dual decision variables only. But different form the primal–dual network, the dual neural network is developed with the latter approach of the above design methodologies to reduce network complexity and increase computational efficiency. But up to now, only a few dual neural network models have been developed for quadratic programming subject to inequality constraints [12] or simple bound constraints [13]. Linear constraints of the general form, which may include equality, inequality and bound constraints simultaneously, should be addressed to cover the need for engineering applications. In addition to the aim of developing a neural network with much simple architecture, the above consideration is our motivation of this study.

The remainder of this Letter is organized in six sections. Section 2 provides the background information and problem formulation of a quadratic program under equality, inequality and bound constraints. Section 3 proposes a dual neural network model for solving quadratic programming problems subject to general linear constraints. The convergence results on globally exponential convergence are given in Section 4. Section 5 presents an illustrative example of quadratic program under linear equality, inequality and bound constraints solved by using the proposed dual neural network. Section 6 concludes the Letter with final remarks.

2. Preliminaries

Consider the following quadratic programming problem subject to various linear constraints:

$$\text{Minimize} \quad \frac{1}{2}x^T Qx + c^T x, \tag{1}$$

$$\text{Subject to} \quad Ax = b, \tag{2}$$

$$Cx \leqslant d, \tag{3}$$

$$x^- \leqslant x \leqslant x^+, \tag{4}$$

where x is the n-dimensional decision vector, $Q \in R^{n\times n}$ is a positive-definite symmetric matrix, the other coefficient matrices and vectors are defined, respectively, as $c \in R^n$, $A \in R^{p\times n}$, $b \in R^p$, $C \in R^{q\times n}$, $d \in R^q$, the n-dimensional vectors x^- and x^+ are, respectively, the lower and upper bounds of x.

The Lagrange neural network [5] was developed originally for solving equality-constrained quadratic programming problems. If extending Lagrange neural network to solve the above general constrained quadratic program, the network architecture is eight-layered with the total number of neurons equal to $5n + p + 2q$. Additionally, the premature defect of such network appears that at some situations the equilibria of the Lagrange neural network may ensure feasibility only rather than optimality. The reason is that the design procedure does not take into account the complimentary slack condition.

A single-layered recurrent neural networks [7,10], based on the gradient and projected method, is an effective tool to solve the bound-constrained quadratic programming problem, i.e., (1) subject to (4) only. The most generalized form of the simple bound constraint (4) that can be handled is $x^- \leqslant Px \leqslant x^+$, where P is a nonsingular square matrix of dimension n. If P is singular or not a square matrix, then a transform involved in the design procedure becomes inapplicable and the network cannot be built up. That is why such neural networks cannot be extended to handle the general quadratic program (1)–(4).

In the literature of recurrent neural networks on quadratic program, few studies have been conducted explicitly to include simultaneously all types of linear constraints such as (2)–(4). The usual approach is to design a recurrent neural network (e.g., the primal–dual neural networks [9,11]) by considering only one or two extendable basic kinds of constraints, then to include the other constraints by converting them into the basic ones. However, the dimensionality of the resultant neural network for solving hybrid-constrained quadratic programs may be much larger than expected. For instance, a single-layered dual neural network [12] was developed for solving quadratic program under inequality constraint (3) only. The dynamic equation and output equation are

$$\begin{aligned}\dot{u} &= \left(I_q + CQ^{-1}C^T\right) \\ &\quad \times \left\{g\left(\left(I_q - CQ^{-1}C^T\right)u - CQ^{-1}c - d\right) - u\right\}, \\ x &= -Q^{-1}\left(C^T u + c\right), \qquad (5)\end{aligned}$$

where the dual decision variable vector $u \in R^q$ represents the states of neurons, I_q denotes the $q \times q$ identity matrix, and the vector-valued function $g(u) = [g_1(u_1), \ldots, g_q(u_q)]^T$ is defined as $g_i(u_i) = \max(0, u_i)$, $i = 1, \ldots, q$. Now let us convert the constraints (2)–(4) as

$$\begin{aligned}&Ax \leqslant b, \qquad -Ax \leqslant -b, \\ &Cx \leqslant d, \\ &I_n x \leqslant x^+, \qquad -I_n x \leqslant -x^-.\end{aligned}$$

The coefficient matrix and vector of the resulting inequality constraint $\tilde{C}x \leqslant \tilde{d}$ are thus defined as

$$\tilde{C} = \begin{pmatrix} A \\ -A \\ C \\ I_n \\ -I_n \end{pmatrix}, \qquad \tilde{d} = \begin{pmatrix} b \\ -b \\ d \\ x^+ \\ -x^- \end{pmatrix}.$$

Clearly, to solve (1)–(4), the dimensionality of a dual neural network generalized from (5) is $2p + q + 2n$, which is smaller than that of the aforementioned Lagrange neural network. But such generalized neural network is still not sufficiently economical in terms of the neuron number and network architecture. Besides, for network implementation, the hardware complexity of analogue circuits increases substantially as the total number of neurons increases.

Before ending this section, it is worth discussing the reason why the aforementioned neural network approaches are not economical. To solve (1)–(4), the usual neural network design methods convert equality constraint and two-sided inequality constraint into two one-sided inequality constraints, which has unnecessarily increased the dimension of the dual space, and thus introduces some excessive neurons. If we can treat the equality constraint (2) as a special two-sided bound constraint (with lower bounds equal to upper ones) and treat the one-sided inequality constraint (3) as a two-sided bound constraint (with lower bounds equal to "the negative infinity"). This may provide a unified treatment of both equality, inequality and simple bound constraints, and the resultant neural network for solving general-form quadratic programming (1)–(4) is of size $n + p + q$ only.

3. Model description

As a generalization of the design procedure of the dual neural network [13] for quadratic programming with simple bound constraint, we can reformulate a

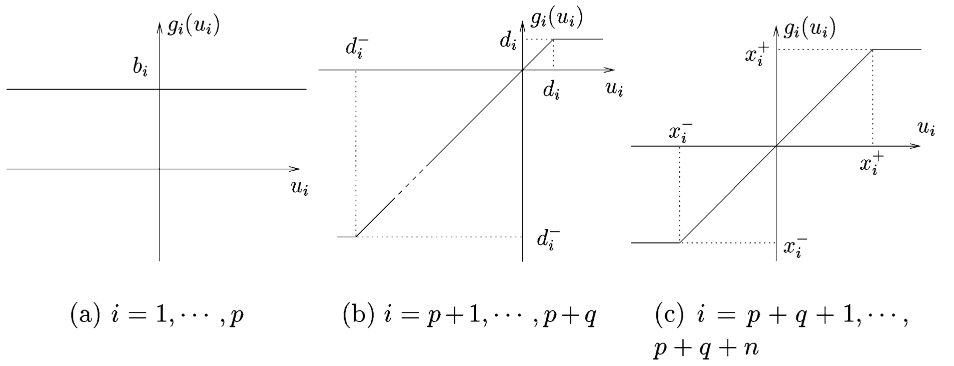

Fig. 1. Projection operator $g_i(u_i)$ on $[r_i^-, r_i^+]$.

generally-constrained quadratic program (1)–(4) into a unified form. That is, to treat equality and one-sided inequality constraints as special cases of two-sided bound constraints, we define

$$r^- := \begin{pmatrix} b \\ d^- \\ x^- \end{pmatrix}, \qquad r^+ := \begin{pmatrix} b \\ d \\ x^+ \end{pmatrix}, \qquad J := \begin{pmatrix} A \\ C \\ I_n \end{pmatrix},$$

where $d^- \in R^p$, and $\forall j \in \{1, \ldots, p\}$, $d_j^- \ll 0$ sufficiently large to represent $-\infty$. Then (1)–(4) are rewritten in the following form

$$\begin{aligned} &\text{Minimize} && \frac{1}{2}x^T Q x + c^T x, \\ &\text{Subject to} && r^- \leqslant Jx \leqslant r^+. \end{aligned} \tag{6}$$

In the above formulation, the generalized feasibility region $[r^-, r^+]$ is constructed as a closed convex set to facilitate the design and analysis of the dual neural network via the Karush–Kuhn–Tucker condition and the projection operator.

It follows from the Karush–Kuhn–Tucker condition that x is a solution to (6) if and only if there exists $u \in R^{p+q+n}$ such that $Qx - J^T u + c = 0$ and

$$\begin{cases} [Jx]_i = r_i^-, & \text{if } u_i > 0, \\ [Jx]_i = r_i^+, & \text{if } u_i < 0, \\ r_i^- \leqslant [Jx]_i \leqslant r_i^+, & \text{if } u_i = 0. \end{cases} \tag{7}$$

The complementarity condition (7) is equivalent to the system of piecewise linear equation $Jx = g(Jx - u)$ [14–16], where the vector-valued function $g(u) = [g_1(u_1), \ldots, g_{p+q+n}(u_{p+q+n})]^T$ is defined as

$$g_i(u_i) = \begin{cases} r_i^-, & \text{if } u_i < r_i^-, \\ u_i, & \text{if } r_i^- \leqslant u_i \leqslant r_i^+, \\ r_i^+, & \text{if } u_i > r_i^+, \end{cases}$$
$$i = 1, \ldots, p+q+n, \tag{8}$$

which may include three situations as depicted in Fig. 1 by the definitions of r^+ and r^-. Therefore, x is a solution to (6) if and only if there exists the dual decision vector $u \in R^{p+q+n}$ such that $Qx - J^T u + c = 0$ and $Jx = g(Jx - u)$; that is,

$$\begin{cases} x = Q^{-1}J^T u - Q^{-1}c, \\ g\big(JQ^{-1}J^T u - JQ^{-1}c - u\big) \\ \quad = JQ^{-1}J^T u - JQ^{-1}c. \end{cases} \tag{9}$$

The dual neural network model for solving (6) is thus developed with the following dynamical equation and output equation

$$\begin{aligned} \Lambda \dot{u} &= -JQ^{-1}J^T u \\ &\quad + g\big(JQ^{-1}J^T u - u - JQ^{-1}c\big) + JQ^{-1}c, \\ x &= Q^{-1}J^T u - Q^{-1}c, \end{aligned} \tag{10}$$

where $\Lambda \in R^{(p+q+n)\times(p+q+n)}$ is a positive diagonal matrix to scale the convergence rate of the proposed dual network.

Clearly, to solve the quadratic program with general linear constraints, the dynamic equation of the dual neural network is piecewise linear and does not contain any high-order nonlinear term or penalty parameter. Compared to other recurrent neural networks, the dual neural network model is of single-layer with no more than $p + q + n$ neurons and much simpler in terms of network architecture. The block diagram of the dual neural network system is depicted in Fig. 2. A circuit realizing the dual neural network consists of summers, integrators and weighted connections, and the piecewise linear activation function $g_i(u_i)$ may be implemented by using an operational amplifier.

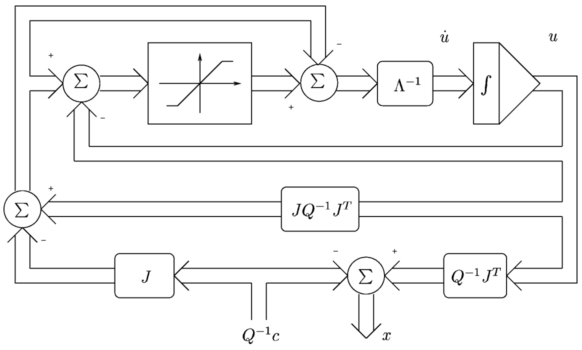

Fig. 2. Block diagram of the dual neural network for solving quadratic programs.

4. Convergence results

In the section, we show the global exponential convergence of the proposed dual neural network for solving quadratic programs.

Related definitions and a lemma are presented first. A neural network is said to be globally convergent if starting form any initial point taken in the associated Euclidean space, every state trajectory of the neural network converges to an equilibrium point that depends on the initial state of the trajectory. Furthermore, the neural network is said to be globally exponentially convergent if every trajectory starting from any initial point $x(t_0)$ satisfies

$$\|x(t) - x^*\| \leqslant \alpha \|x(t_0) - x^*\| \exp(-\beta(t - t_0)),$$
$$\forall t \geqslant t_0 \geqslant 0,$$

where α and β are positive constants, x^* is an equilibrium point. The exponential convergence is the most desirable convergence property.

Lemma 1 [15,17–19]. *Assume that the set $\Omega \subset R^q$ is a closed convex set, then the following two inequalities hold*

$$(v - P_\Omega(v))^T (P_\Omega(v) - u) \geqslant 0, \quad \forall v \in R^q, u \in \Omega,$$
$$\|P_\Omega(u) - P_\Omega(v)\| \leqslant \|u - v\|, \quad \forall u, v \in R^q,$$

where $P_\Omega(x) = \arg\min_{v\in\Omega} \|x - v\|$ is a projection operator from R^q to Ω.

It is clear that the set $\Omega := \{u \in R^{p+q+n} \mid r^- \leqslant u \leqslant r^+\}$ is a closed convex set, and $g(\cdot)$ in (8) possesses the above projection property. The convergence results of the dual neural network for constrained quadratic programming are discussed as follows.

Theorem 1. *The state of the dual neural network* (10) *is globally exponentially convergent to an equilibrium point u^*.*

Proof. To show the convergence property, the following inequalities are derived.

At u^*, we have the following inequality property [16,19]

$$(v - JQ^{-1}J^Tu^* + JQ^{-1}c)^T u^* \geqslant 0, \quad \forall v \in \Omega, \tag{11}$$

which can be obtained by discussing the following three cases:

Case 1. If for some $i \in \{1, \ldots, p+q+n\}$, $u_i^* = 0$, $r_i^- \leqslant [Jx^*]_i \leqslant r_i^+$, then $(v_i - [Jx^*]_i)u_i^* = 0$;
Case 2. If for some $j \in \{1, \ldots, p+q+n\}$, $u_j^* > 0$, $[Jx^*]_j = r^-$ and $r_j^- \leqslant v_j \leqslant r_j^+$, then $v_j - [Jx^*]_j \geqslant 0$ and thus $(v_j - [Jx^*]_j)u_j^* \geqslant 0$;
Case 3. If for some $k \in \{1, \ldots, p+q+n\}$, $u_k^* < 0$, $[Jx^*]_k = r_k^+$ and $r_k^- \leqslant v_k \leqslant r_k^+$, then $v_k - [Jx^*]_k \leqslant 0$ and thus $(v_k - [Jx^*]_k)u_k^* \geqslant 0$.

Therefore it follows from (11) that

$$\big(g(JQ^{-1}J^Tu - JQ^{-1}c - u) - JQ^{-1}J^Tu^* + JQ^{-1}c\big)^T u^* \geqslant 0. \tag{12}$$

Defining $\tilde{u} := JQ^{-1}J^Tu - JQ^{-1}c$, it follows from Lemma 1 that $\forall u \in R^{p+q+n}$,

$$\big(g(\tilde{u} - u) - JQ^{-1}J^Tu^* + JQ^{-1}c\big)^T \times \big(\tilde{u} - u - g(\tilde{u} - u)\big) \geqslant 0. \tag{13}$$

Then, adding (12) and (13) yields

$$\big(g(\tilde{u} - u) - JQ^{-1}J^Tu^* + JQ^{-1}c\big)^T \times \big(u^* + \tilde{u} - u - g(\tilde{u} - u)\big) \geqslant 0. \tag{14}$$

Defining

$$\tilde{g} := g(JQ^{-1}J^Tu - JQ^{-1}c - u) - JQ^{-1}J^Tu + JQ^{-1}c,$$

i.e., $\tilde{g} = g(\tilde{u} - u) - \tilde{u}$, (14) is reformulated as

$$\big(\tilde{g} + JQ^{-1}J^T(u - u^*)\big)^T \big((u - u^*) + \tilde{g}\big) \leqslant 0,$$

and subsequently as

$$\big(u-u^*\big)^T\tilde g+\tilde g^T JQ^{-1}J^T\big(u-u^*\big)$$
$$\leqslant -\|\tilde g\|^2-\big(u-u^*\big)^T JQ^{-1}J^T\big(u-u^*\big). \tag{15}$$

Now we choose a Lyapunov function candidate as

$$V\big(u(t)\big)=\frac{1}{2}\big\|M\big(u(t)-u^*\big)\big\|^2, \tag{16}$$

where matrix M is symmetric positive definite and $M^2=(I+JQ^{-1}J^T)\Lambda>0$. Clearly, $V(u)$ is positive definite (i.e., $V>0$ if $u\neq u^*$ and $V=0$ iff $u=u^*$) for any u taken in the domain $\Gamma(u^*)\subset R^{p+q+n}$ (i.e., the attraction region of u^*). dV/dt is negative definite, since, in view of (15),

$$\begin{aligned}\frac{dV}{dt}&=\big(u-u^*\big)^T M^2\dot u\\&=\big(u-u^*\big)^T\big(I+JQ^{-1}J^T\big)\tilde g\\&\leqslant -\|\tilde g\|^2-\big(u-u^*\big)^T JQ^{-1}J^T\big(u-u^*\big)\\&\leqslant 0,\end{aligned} \tag{17}$$

and $dV/dt=0$ iff $u=u^*$ in $\Gamma(u^*)$. Thus it follows that the dual neural network (10) is globally convergent to an equilibrium point u^* which depends on the initial state of the trajectory.

Furthermore, to show the global exponential convergence [20,21], review $V(u)$ and dV/dt. It follows from (16) that $c_1\|u-u^*\|^2\leqslant V(u)\leqslant c_2\|u-u^*\|^2$, where $c_2\geqslant c_1>0$ are, respectively, the maximal and minimal eigenvalues of $(I+JQ^{-1}J^T)\Lambda$. Clearly, c_1 and c_2 are proportional to the design parameter Λ. Moreover, it is reasonably assumed that $\exists\rho>0$, $\|\tilde g\|^2\geqslant\rho\|u-u^*\|^2$, since $\|\tilde g\|>0$ for any $u(t)\neq u^*$ in $\Gamma(u^*)$, and $\|\tilde g\|=0$ amounts to $u=u^*$. In addition, by analyzing the linear/saturation cases of $g_j([JQ^{-1}J^Tu-JQ^{-1}c-u]_j)$, the range of ρ is $(0,1]$. Therefore from (15), we have

$$\begin{aligned}\frac{dV(u)}{dt}&\leqslant -\rho\|u-u^*\|^2\\&\quad-\big(u-u^*\big)^T JQ^{-1}J^T\big(u-u^*\big)\\&=-\big(u-u^*\big)^T\big(\rho I+JQ^{-1}J^T\big)\big(u-u^*\big)\\&\leqslant -\beta V\big(u(t)\big),\end{aligned}$$

where $\beta=\rho/c_2>0$. Thus we have $V(u(t))=O(\exp(-\beta(t-t_0)))$, $\forall t\geqslant t_0$, and hence $\|u(t)-u^*\|=O(\exp(-\beta(t-t_0)/2))$, $\forall t\geqslant t_0$. □

Theorem 2. *The output $x^*=Q^{-1}J^Tu^*-Q^{-1}c$ is the optimal solution to the constrained quadratic programming problem* (1)–(4), *where u^* denotes the equilibrium point of the dual neural network.*

Proof. The equilibrium of the dual neural network satisfies the following equations:

$$g\big(JQ^{-1}J^Tu^*-JQ^{-1}c-u^*\big)$$
$$-JQ^{-1}J^Tu^*+JQ^{-1}c=0,$$
$$x^*=Q^{-1}J^Tu^*-Q^{-1}c,$$

which is equivalent to

$$g\big(Jx^*-u^*\big)-Jx^*=0. \tag{18}$$

Let u_A^*, u_C^* and u_I^* denote the vectors of dual decision variables corresponding to the constraints (2), (3), (4), respectively, i.e., $u^*=[(u_A^*)^T,(u_C^*)^T,(u_I^*)^T]^T$. Eq. (18) becomes

$$g\left(\begin{bmatrix}Ax^*-u_A^*\\Cx^*-u_C^*\\x^*-u_I^*\end{bmatrix}\right)-\begin{bmatrix}Ax^*\\Cx^*\\x^*\end{bmatrix}=\mathbf{0}. \tag{19}$$

By the definition of $g(\cdot)$, it follows from (19) that $b-Ax^*=0$ with u_A^* unrestricted, and

$$\begin{cases}Cx^*-d\leqslant 0, \quad u_C^*\leqslant 0,\\ \big(u_C^*\big)^T\big(Cx^*-d\big)=0,\end{cases}$$
$$\begin{cases}x_i^*=x_i^-, & \text{if } \big[u_I^*\big]_i>0,\\ x_i^*=x_i^+, & \text{if } \big[u_I^*\big]_i<0,\\ x_i^-\leqslant x_i^*\leqslant x_i^+, & \text{if } \big[u_I^*\big]_i=0.\end{cases}$$

In addition to $Qx^*-A^Tu_A^*-C^Tu_C^*-u_I^*+c=0$ implied by $x^*=Q^{-1}J^Tu^*-Q^{-1}c$, the above equations constitute the Kurash–Kuhn–Tucker optimality conditions of (1)–(4), which completes the proof. □

Corollary. *The output $x(t)$ of the dual neural network is globally exponentially convergent to the unique equilibrium point x^*, which is the solution to constrained quadratic programming problem* (1)–(4).

Proof. Since the objective function in (1) is strictly convex due to positive definiteness of Q, and the constraint region constructed by (2)–(4) is a convex set, by the Theorem 3.4.2 of [22], we know that the constrained minimizer x^* to quadratic programming problem (1)–(4) is unique.

It follows from the output equation

$$x(t) = Q^{-1}J^T u(t) - Q^{-1}c,$$

and Theorem 2 that

$$\begin{aligned}\|x(t) - x^*\| &= \|Q^{-1}J^T(u(t) - u^*)\| \\ &\leqslant \|Q^{-1}J^T\|\,\|u(t) - u^*\|. \end{aligned} \qquad (20)$$

In view of the results of Theorem 1 that $\|u(t) - u^*\| = O(\exp(-\beta(t - t_0)/2))$, $\forall t \geqslant t_0 \geqslant 0$, the inequality (20) implies that the output $x(t)$ of the dual neural network (10) exponentially converges to x^*, which is the unique solution to the constrained quadratic program (1)–(4). □

5. Illustrative example

In the section, a numerical example in [12] is discussed to demonstrate the effectiveness and performance behavior of the proposed dual neural network for solving quadratic programs subject to general linear constraints:

$$\begin{aligned}\text{Minimize}\quad & 11x_1^2 + x_2^2 + x_3^2 - 2x_1x_2 + 6x_1x_3 - 4x_1,\\ \text{Subject to}\quad & 2x_1 + 2x_2 + x_3 = 0,\\ & -x_1 + x_2 \leqslant -1,\\ & 3x_1 + x_3 \leqslant 4,\\ & -6 \leqslant x_1, x_2, x_3 \leqslant 6.\end{aligned}$$

That is, $A = [2, 2, 1]$, $b = 0$, $x^+ = -x^- = [6, 6, 6]^T$ and

$$Q = \begin{bmatrix} 22 & -2 & 6 \\ -2 & 2 & 0 \\ 6 & 0 & 2 \end{bmatrix}, \qquad c = \begin{bmatrix} -4 \\ 0 \\ 0 \end{bmatrix},$$

$$C = \begin{bmatrix} -1 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix}, \qquad d = \begin{bmatrix} -1 \\ 4 \end{bmatrix}.$$

Using the Lagrange neural network [5] to solve the above quadratic program, the total number of neurons is 20, in addition to the premature defect. Using the primal–dual neural networks or other kind of dual network such as [9,11,12], the total number of neurons is usually more than 10. As a comparison, the size of the proposed dual neural network to solve the above quadratic programming problem is only 6. As opposed to primal–dual neural networks, the complexity reduction of the proposed dual network is more than $(10 - 6)/6 \approx 66\%$.

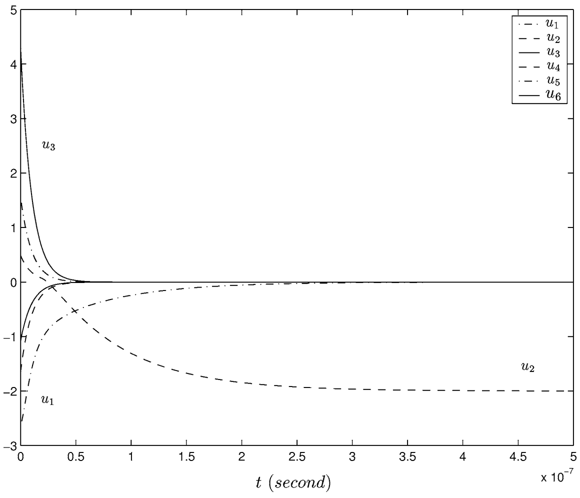

Fig. 3. The state transients of the dual neural network for solving the constrained quadratic program.

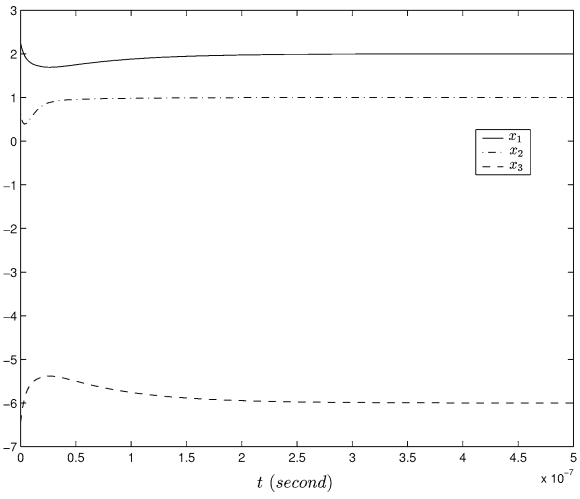

Fig. 4. The output transients of the dual neural network for solving the constrained quadratic program.

The dual neural network with $c = 10^{-8}$ is simulated, and the results are illustrated in Figs. 3–6. As shown in Figs. 3 and 4, started with an initial state randomly selected within $[-5, 5]^6$, the dual neural network converges to the optimal solution of the constrained quadratic program. That is, within 5×10^{-7} second (i.e., 0.5 μs), the network can approach $x^* = [2, 1, -6]^T$ and $u^* = [0, -2, 0, 0, 0, 0]^T$ without any

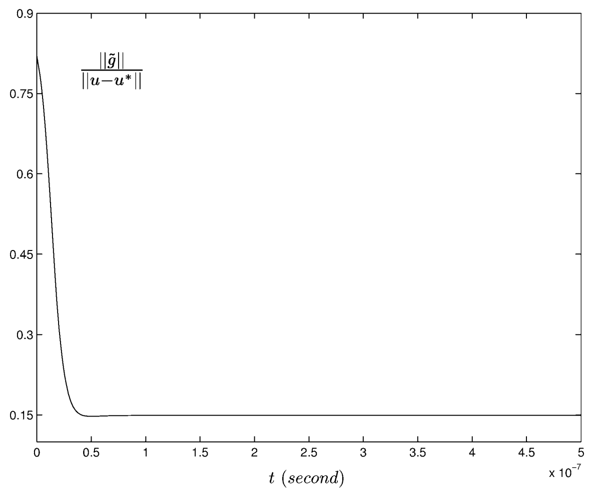

Fig. 5. The transients of $\|\tilde{g}\|/\|u-u^*\|$ in solving the constrained quadratic program.

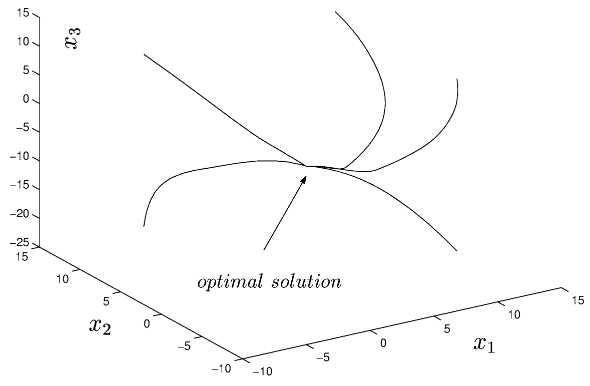

Fig. 6. Spatial trajectories of (x_1, x_2, x_3) of the dual neural network starting from five different initial states.

appreciable error. In addition, the difference between the neurally-computed solution and the theoretical solution x^* is less than 10^{-12}. Fig. 5 illustrates the transient of $\|\tilde{g}\|/\|u-u^*\|$ during solving the constrained quadratic program, and hence the constant ρ appearing in the proof of Theorem 1 is about 0.15 in the example. The nonzero ρ guarantees the exponential convergence property of the dual neural network. As illustrated in Fig. 6, the states of the dual neural network, starting from five different initial states, all converges to the optimal solution of the minimization problem. This substantiates the global convergence property of the dual neural network.

6. Concluding remarks

The Letter presents a single-layer dual neural network for solving constrained quadratic programming problems. Compared to other neural networks for quadratic programming, the proposed dual neural network is designed with fewer neurons and with global exponential convergence property. Moreover, the dynamic equation of the dual neural network is piecewise linear and does not contain any high-order nonlinear term, and thus the architecture is much simpler. The simulation results also substantiate the theoretical results and demonstrate superior performance of the dual neural network.

References

[1] I.B. Pyne, Trans. Am. Inst. Elec. Eng. 75 (1956) 139.
[2] D.W. Tank, J.J. Hopfield, IEEE Trans. Circuits Systems 33 (1986) 533.
[3] M.P. Kennedy, L.O. Chua, IEEE Trans. Circuits Systems 35 (5) (1998) 554.
[4] Y. Xia, J. Wang, Recurrent neural networks for optimization: the state of art, in: L.R. Medsker, L.C. Jain (Eds.), CRC Press, New York, 2000, pp. 13–47, chapter 2.
[5] S. Zhang, A.G. Constantinides, IEEE Trans. Circuits Systems 39 (7) (1992) 441.
[6] S. Sudharsanan, M. Sundareshan, Neural Networks 4 (5) (1991) 599.
[7] A. Bouzerdoum, T.R. Pattison, IEEE Trans. Neural Networks 4 (2) (1993) 293.
[8] J. Wang, Neural Networks 7 (4) (1994) 629.
[9] Y. Xia, IEEE Trans. Neural Networks 7 (6) (1996) 1544.
[10] X. Liang, J. Wang, IEEE Trans. Neural Networks 11 (6) (2000) 1251.
[11] Q. Tao, J. Cao, M. Xue, H. Qiao, Phys. Lett. A 288 (2) (2001) 88.
[12] J. Wang, Y. Xia, Intl. Joint Conf. Neural Net. 1 (1999) 588.
[13] Y. Xia, J. Wang, IEEE Trans. Systems Man Cybernet. 31 (1) (2001) 147.
[14] O.L. Mangasarian, J. Optim. Theory Appl. 22 (2) (1979) 465.
[15] D.P. Bertsekas, Parallel and Distributed Computation: Numerical Methods, Prentice-Hall, Englewood Cliffs, NJ, 1989.
[16] W. Li, J. Swetits, SIAM J. Optim. 7 (3) (1997) 595.
[17] J.S. Pang, Math. Oper. Res. 12 (1987) 474.
[18] J.S. Pang, J.C. Yao, SIAM J. Control Optim. 33 (1995) 168.
[19] Y. Xia, J. Wang, IEEE Trans. Neural Networks 9 (6) (1998) 1331.
[20] Y. Xia, J. Wang, IEEE Trans. Neural Networks 11 (4) (2000) 1017.
[21] Y. Xia, J. Wang, IEEE Trans. Automat. Control 46 (4) (2001) 635.
[22] M.S. Bazaraa, H.D. Sherali, C.M. Shetty, Nonlinear Programming—Theory and Algorithms, Wiley, New York, 1993.