Recurrent Neural Networks with Fixed Time Convergence for Linear and Quadratic Programming

Juan Diego Sánchez-Torres, Edgar N. Sanchez and Alexander G. Loukianov

***Abstract*— In this paper, a new class of recurrent neural networks which solve linear and quadratic programs are presented. Their design is considered as a sliding mode control problem, where the network structure is based on the Karush-Kuhn-Tucker (KKT) optimality conditions with the KKT multipliers considered as control inputs to be implemented with fixed time stabilizing terms, instead of common used activation functions. Thus, the main feature of the proposed network is its fixed convergence time to the solution. That means, there is time independent to the initial conditions in which the network converges to the optimization solution. Simulations show the feasibility of the current approach.**

I. Introduction

THE use of dynamical systems which can solve other problems as been a high interest research topic due to its advantages for real time implementation. This class of systems was introduced with the works of Chua [1], Tank and Hopfield [2] and Brockett [3] for linear programming and, Kennedy and Chua [4] for nonlinear programming. Some of these systems were presented as circuits [5], [6] or in the form of the so-called *recurrent neural networks*, as it is shown by Wang and Xia [7], [8].

Usually, discontinuous activation functions are used for recurrent neural networks design. The presence of these discontinuous terms can induce *sliding modes*. This behavior occurs when those terms drive the dynamics of a system to a sliding manifold, that is an integral manifold with finite reaching time [9]; exhibiting features such as finite time convergence, robustness to uncertainties and insensitivity to external bounded disturbances [10]. For this case, the sliding modes and finite time convergence to sets defined by the optimization constraints are desirable characteristics of the neural network [11].

Taking advantage of the features presented by some discontinuous systems, several recurrent neural networks have been proposed using different activation functions as hard-limiting [12]–[14], Heaviside [15] and dead-zone [16], [17]. For such cases, the network is proposed based on the Karush-Kuhn-Tucker (KKT) optimality conditions [18], [19]. This set of conditions can be used to propose the network structure, usually by using the KKT multipliers as activation functions. Further results on networks with these dynamical properties were presented in [20]–[22], where the analysis is based on the theory of differential inclusions and differential equations with discontinuous right-hand [23]–[25].

Therefore, the design of some recurrent neural networks can be proposed as a sliding mode control problem. That is, the KKT multipliers are considered as control inputs which makes attractive a set defined for the constraints of the programming problem. Usually, the sliding mode design consists of two steps [26]: (i) the design of a sliding manifold to ensure the desired dynamics of the system and, (ii) the enforcing of the sliding motion by means of a finite time stabilizing term in the control input. For the step (i), the sets are defined by the problem. On the other hand, for the step (ii), relay controllers (hard functions) are used as the stabilizing terms.

However, additionally to commonly used discontinuous functions, the sliding manifold can be implemented by different methods including use of continuous functions with discontinuous derivatives (so called *higher order sliding modes*) [27] and, with terms which produce fixed time stability as generalizations of the *super-twisting* algorithm [28] and relay controllers plus polynomial terms [29]. All these alternatives open possibilities to design new activation functions, improving the networks performance.

The aim of this paper is to present a class of recurrent neural network to solve linear and quadratic programming problems. Its design is considered as a sliding mode control problem, where the network structure is based on the Karush-Kuhn-Tucker (KKT) optimality conditions (step (i)), and the KKT multipliers are regarded as control inputs to be implemented with fixed time stabilizing terms (step (ii)) instead of common used activation functions. Thus, the main feature of the proposed network is its fixed convergence time to the solution. That means, there is time independent to the initial conditions in which the network converges to the optimization solution.

In the following, Section II presents the mathematical preliminaries and some useful definitions. Sections III and IV describe the proposed recurrent neural networks for the solution of linear an quadratic programs, including an conceptual approach to stability analysis. Section V shows the activation function design, here the stability proof is performed by means of the Lyapunov approach. The simulations are presented in section VI. Finally, in Section VII the conclusions are given.

Juan Diego Sánchez-Torres, Edgar N. Sanchez and Alexander G. Loukianov are with Automatic Control Laboratory, CINVESTAV-IPN Guadalajara, Av. del Bosque 1145 Col. El Bajío CP 45019, México (email: {dsanchez, sanchez, louk}@gdl.cinvestav.mx).

This work was supported by the National Council of Science and Technology (CONACYT), Mexico, under Grant 129591

II. Mathematical Preliminaries

A. On Fixed Time Stability

Consider the system

$$\dot{\xi} = f(t,\xi) \tag{1}$$

where $\xi \in \mathbb{R}^n$ and $f : \mathbb{R}_+ \times \mathbb{R}^n \to \mathbb{R}^n$. If f is a discontinuous (or non-smooth) function, (1) is understood in Filippov sense [23].

Definition 1 (Globally fixed-time attraction [29]): Let a non-empty set $M \subset \mathbb{R}^n$. It is said to be globally fixed-time attractive for the system (1) if any solution $\xi(t,\xi_0)$ of (1) reaches M in some finite time moment $t = T(\xi_0)$ and the settling-time function $T(\xi_0) : \mathbb{R}^n \to \mathbb{R}_+ \cup \{0\}$ is bounded by some positive number $T_{\max}$, i.e. $T(\xi_0) \le T_{\max}$ for $\xi_0 \in \mathbb{R}^n$.

With the definition of a globally fixed-time attractive set, the following lemma provides a Lyapunov characterization of these sets on the state space

Lemma 1 (Lyapunov function [29]): If there exists a continuous radially unbounded function

$$V : \mathbb{R}^n \to \mathbb{R}_+ \cup \{0\}$$

such that $V(\xi) = 0$ for $\xi \in M$ and any solution $\xi(t)$ satisfies

$$\dot{V} \le -\left(\alpha V^p(\xi(t)) + \beta V^q(\xi(t))\right)^k$$

for $\alpha, \beta, p, q, k > 0$ that $pk < 1$ and $qk > 1$, then the set M is globally fixed-time attractive for the system (1) and $T_{\max} = \frac{1}{\alpha^k(1-pk)} + \frac{1}{\beta^k(qk-1)}$.

Consider now the system

$$\dot{\xi} = f(t,\xi) + g(\xi)u + \Delta \tag{2}$$

where $\xi \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $f : \mathbb{R}_+ \times \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$, with $m \le n$ and $\Delta \in \mathbb{R}^n$ a bounded disturbance. If u is a discontinuous (or non-smooth) function, (2) is understood in Filippov sense [23].

For the system (2), a special choice for u is considered in the following definition

Definition 2 (Globally fixed time stabilizer to a set): Let a non-empty set $M \subset \mathbb{R}^m$ and $\Delta = 0$. The function $\mathcal{FS}(\xi, M)$ is called a globally fixed time stabilizer to the set M if taking $u_s = \mathcal{FS}(\xi, M)$, M is a globally fixed-time attractive set of

$$\dot{\xi} = f(t,\xi) + g(\xi)u_s.$$

If this condition on M stills fulfilling despite of $\Delta \neq 0$, $\mathcal{FS}(\cdot)$ is called a robust stabilizer.

A particular case of fixed time stabilizer is given by

Definition 3 (Globally fixed time sliding mode operator): Let $\sigma : \mathbb{R}^n \to \mathbb{R}^m$ a continuous function. If in *Definition 2* the set M is such that $\sigma = 0$, the globally fixed time stabilizer is called *globally fixed time sliding mode operator* denoted by $u_s = \mathcal{SL}(\sigma)$, and $\sigma = 0$ is called a sliding manifold.

Notice from *Lemma 1*, that $\mathcal{FS}(\xi, M) = 0$ for $\xi \in M$.

III. Linear Programming

Let the following linear programming problem:

$$\begin{cases} \min_x & \mathbf{c}^T x \\ \text{s.t} & \mathbf{A}x = \mathbf{b} \\ & l \le x \le h \end{cases} \tag{3}$$

where $x = \begin{bmatrix} x_1 & \dots & x_n \end{bmatrix}^T \in \mathbb{R}^n$ are the decision variables, $\mathbf{c} \in \mathbb{R}^n$ is a cost vector, $\mathbf{A}$ is an $m \times n$ matrix such that rank$(\mathbf{A}) = m$ and $m \le n$; $\mathbf{b}$ is a vector in $\mathbb{R}^m$ and, $l = \begin{bmatrix} l_1 & \dots & l_n \end{bmatrix}$, $h = \begin{bmatrix} h_1 & \dots & h_n \end{bmatrix} \in \mathbb{R}^n$.

Let $y = \begin{bmatrix} y_1 & \dots & y_m \end{bmatrix}^T \in \mathbb{R}^m$ and $z = \begin{bmatrix} z_1 & \dots & z_n \end{bmatrix}^T \in \mathbb{R}^n$. Hence, the Lagrangian of (3) is

$$L(x,y,z) = \mathbf{c}^T x + z^T x + y^T(\mathbf{A}x - \mathbf{b}). \tag{4}$$

The KKT conditions claim that x^* is a solution for (3) if and only if x^*, y and z in (3)-(4) are such that

$$\nabla_x L(x^*, y, z) = \mathbf{c} + z + \mathbf{A}^T y = 0 \tag{5}$$
$$\mathbf{A}x^* - \mathbf{b} = 0 \tag{6}$$
$$z_i x_i^* = 0 \text{ if } l_i < x_i^* < h_i,\ \forall i = 1,\dots,n. \tag{7}$$

Following the KKT approach, here a recurrent neural network which solves the problem (3) in fixed time is proposed. For this purpose, let

$$\Omega_e = \{x \in \mathbb{R}^n : \mathbf{A}x - \mathbf{b} = 0\}$$
$$\Omega_d = \{x \in \mathbb{R}^n : l \le x \le h\}.$$

According to (3), $x^* \in \Omega$ where $\Omega = \text{int}(\Omega_d \cap \Omega_e)$.

From (5), let

$$\dot{x} = -\mathbf{c} + z + \mathbf{A}^T y. \tag{8}$$

Then, y and z will be designed such that Ω is a fixed time attractive set, fulfilling conditions (5)-(7).

In addition to condition (7), z is considered such that

$$\begin{cases} z_i \ge 0 & \text{if } x_i \ge h_i \\ z_i \le 0 & \text{if } x_i \le l_i \end{cases}. \tag{9}$$

Thus, defining

$$\sigma = \mathbf{A}x - \mathbf{b} \tag{10}$$

a suitable choice for y and z is proposed as

$$y = \mathcal{SL}(\sigma) \tag{11}$$
$$z = \mathcal{FS}(x, \Omega_d). \tag{12}$$

With the stabilizing terms (11)-(12), the dynamics of the system (8)-(10) become

$$\dot{\sigma} = -\mathbf{A}\mathbf{c} + \mathbf{A}\mathcal{FS}(x,\Omega_d) + \mathbf{A}\mathbf{A}^T\mathcal{SL}(\sigma) \tag{13}$$
$$\dot{x} = -\mathbf{c} + \mathcal{FS}(x,\Omega_d) + \mathbf{A}^T\mathcal{SL}(\sigma) \tag{14}$$

As a conceptual approach to the stabilization of (13)-(14), it can be considered from the definition of $\mathcal{FS}(x,\Omega_d)$ in (14), that x reaches the set Ω_d in a fixed time t_d. For $t > t_d$ the operator $\mathcal{FS}(x,\Omega_d) = 0$, then from (13) it follows

$$\dot{\sigma} = -\mathbf{A}\mathbf{c} + \mathbf{A}\mathbf{A}^T\mathcal{SL}(\sigma). \tag{15}$$

And, from the definition of $\mathcal{SL}(\sigma)$, the solutions of (15) reach $\sigma = 0$ in a fixed time $t_e > t_d$.

At this point, it is clear that the conditions (6) and (7) are satisfied. Now, by using the *equivalent control method* [26] as solution of $\dot{x} = 0$ in (8) for $t > t_e$, it follows that $z = \mathcal{FS}(x, \Omega_d)_{eq} = 0$ and $\mathbf{c} + \mathbf{A}^T \mathcal{SL}(\sigma)_{eq} = 0$, satisfying the condition (5).

IV. Quadratic Programming

Similarly to (3), the quadratic programming problem is defined as:

$$\begin{cases} \min_x & \frac{1}{2} x^T \mathbf{Q} x + \mathbf{c}^T x \\ \text{s.t} & \mathbf{A}x = \mathbf{b} \\ & l \leq x \leq h \end{cases} \tag{16}$$

where $x = \begin{bmatrix} x_1 & \dots & x_n \end{bmatrix}^T \in \mathbb{R}^n$ are the decision variables, $\mathbf{c} \in \mathbb{R}^n$ is a cost vector, $\mathbf{Q}$ is a $n \times n$ symmetric matrix, $\mathbf{A}$ is a $m \times n$ matrix such that $\text{rank}(\mathbf{A}) = m$ and $m \leq n$; $\mathbf{b}$ is a vector in $\mathbb{R}^m$ and, $l = \begin{bmatrix} l_1 & \dots & l_n \end{bmatrix}$, $h = \begin{bmatrix} h_1 & \dots & h_n \end{bmatrix} \in \mathbb{R}^n$.

Let $y = \begin{bmatrix} y_1 & \dots & y_m \end{bmatrix} \in \mathbb{R}^m$ and $z = \begin{bmatrix} z_1 & \dots & z_n \end{bmatrix} \in \mathbb{R}^n$. Hence, the Lagrangian of (16) is

$$L(x, y, z) = \frac{1}{2} x^T \mathbf{Q} x + \mathbf{c}^T x + z^T x + y^T (\mathbf{A}x - \mathbf{b}) \tag{17}$$

The Karush-Kuhn-Tucker (KKT) conditions claim that x^* is a solution for (16) if and only if x^*, y and z in (16)-(17) are such that

$$\nabla_x L(x^*, y, z) = \mathbf{Q}x^* + \mathbf{c} + z + \mathbf{A}^T y = 0 \tag{18}$$

$$\mathbf{A}x^* - \mathbf{b} = 0 \tag{19}$$

$$z_i x_i^* = 0 \text{ if } l_i < x_i^* < h_i, \ \forall i = 1, \dots, n. \tag{20}$$

Then, let $\Omega_e = \{x \in \mathbb{R}^n : Ax - b = 0\}$ and $\Omega_d = \{x \in \mathbb{R}^n : l \leq x \leq h\}$. According to (16), $x^* \in \Omega$ where $\Omega = \text{int}(\Omega_d \cap \Omega_e)$.

From (5), the structure for the network is

$$\dot{x} = -\mathbf{Q}x - \mathbf{c} + z + \mathbf{A}^T y \tag{21}$$

then, as for the linear programming case, y and z are designed such that Ω is a fixed time attractive set, fulfilling conditions (18)-(20).

Using the conditions (9)-(20), defining $\sigma = \mathbf{A}x - \mathbf{b}$ and selecting y and z as in (11)-(12), the system (21) reduces to

$$\dot{\sigma} = -\mathbf{AQ}x - \mathbf{Ac} + \mathbf{A}\mathcal{FS}(x, \Omega_d) + \mathbf{AA}^T \mathcal{SL}(\sigma) \tag{22}$$

$$\dot{x} = -\mathbf{Q}x - \mathbf{c} + \mathcal{FS}(x, \Omega_d) + \mathbf{A}^T \mathcal{SL}(\sigma) \tag{23}$$

The same analysis used for the linear programming is applied as conceptual approach to (22)-(23), it can be considered from the definition of $\mathcal{FS}(x, \Omega_d)$, in (23) that $x \in \Omega_d$ in a fixed time t_d. For $t > t_d$ the operator $\mathcal{FS}(x, \Omega_d) = 0$, then from (22)

$$\dot{\sigma} = -\mathbf{AQ}x - \mathbf{Ac} + \mathbf{AA}^T \mathcal{SL}(\sigma). \tag{24}$$

And, from the definition of $\mathcal{SL}(\sigma)$, in (15) $\sigma = 0$ in a fixed time $t_e > t_d$.

Again, by using the *equivalent control method* [26] as solution of $\dot{x} = 0$ in (21) for $t > t_e$, it follows that $z = \mathcal{FS}(x, \Omega_d)_{eq} = 0$ and $\mathbf{Q}x^* + \mathbf{c} + \mathbf{A}^T \mathcal{SL}(\sigma)_{eq} = 0$, satisfying the condition (18).

V. Activation Function Design

Usual recurrent neural networks approaches to solve (3)-(16) are based on the idea of define a dynamical system with some structural features from the KKT. Then, an important step is the design of activation function which fulfills (7)-(20). This section presents a class of activation functions which satisfy the KKT conditions, providing fixed time stabilization.

As fixed time stabilizer to the set Ω_d, $z = \mathcal{FS}(x, \Omega_d)$ the following function is proposed

$$\mathcal{FS}(x, \Omega_d) = \begin{bmatrix} \mathcal{FS}_1(x_1, [h_1, l_1]) & \dots & \mathcal{FS}_n(x_n, [h_n, l_n]) \end{bmatrix}^T$$

where $\mathcal{FS}_i(\cdot)$ is proposed as

$$\mathcal{FS}_i(x_i, [l_i, h_i]) = \begin{cases} f_s(x_i - l_i) & \text{if } x_i \leq l_i \\ 0 & \text{if } l_i < x_i < h_i \\ f_s(x_i - h_i) & \text{if } x_i \geq h_i \end{cases} \tag{25}$$

with $f_s(\cdot) = -k_{i1}\text{sign}(\cdot) - k_{i2}(\cdot) - k_{i3}(\cdot)^3$.

For this case, $y = \mathcal{SL}(\sigma)$ is selected as

$$\mathcal{SL}(\cdot) = -k_{i4}\text{sign}(\cdot) - k_{i5}(\cdot) - k_{i6}(\cdot)^3. \tag{26}$$

Without lost of generality, in order to analyze the stability of the proposed network, only the case of quadratic programming will be considered.

The stability analysis es performed in two stages

1) Reaching phase to the set Ω_d and,
2) reaching phase stability to the sliding set $\sigma = 0$.

1) For the first stage, consider

$$\dot{x} = -\mathbf{Q}x - \mathbf{c} + \mathcal{FS}(x, \Omega_d) + \mathbf{A}^T \mathcal{SL}(\sigma) \tag{27}$$

now, let $[-\mathbf{Q}x + \mathbf{c} + \mathbf{A}^T \mathcal{SL}(\sigma)]_i$ the i-th row of the vector $-\mathbf{Q}x - \mathbf{c} + \mathbf{A}^T \mathcal{SL}(\sigma)$ with $i = 1, \dots, n$. Assuming $\Delta_i = [-\mathbf{Q}x - \mathbf{c} + \mathbf{A}^T \mathcal{SL}(\sigma)]_i$ as a bounded term $\Delta_i < \delta_i$ with $\delta_i > 0$, the dynamics for x_i is given by

$$\dot{x}_i = \mathcal{FS}_i(x_i, [l_i, h_i]) + \Delta_i. \tag{28}$$

For (28), the following Lyapunov candidate is proposed

$$V_{d_i} = \begin{cases} \frac{1}{2}(x_i - l_i)^2 & \text{if } x_i \leq l_i \\ 0 & \text{if } l_i < x_i < h_i \\ \frac{1}{2}(x_i - h_i)^2 & \text{if } x_i \geq h_i \end{cases} \tag{29}$$

and its derivative is given by

$$\dot{V}_{d_i} = \begin{cases} g_s(x_i - l_i) + \Delta_i & \text{if } x_i \leq l_i \\ 0 & \text{if } l_i < x_i < h_i \\ g_s(x_i - h_i) + \Delta_i & \text{if } x_i \geq h_i \end{cases} \tag{30}$$

with $f_s(\cdot) = -k_{i1}|(\cdot)| - k_{i2}(\cdot)^2 - k_{i3}(\cdot)^4$.

From (29)-(30) and taking $k_{i1} > \delta_i$, it follows that

$$\dot{V}_{d_i} \begin{cases} = 0 & \text{if } l_i < x_i < h_i \\ < -k_{i1}\sqrt{2}V_{d_i}^{\frac{1}{2}} - 4k_{i3}V_{d_i}^{2} & \text{otherwise} \end{cases} \tag{31}$$

therefore, $x \in \Omega_d$ in a fixed time t_d.

2) To analyze the reaching phase stability to the sliding set $\sigma = 0$, consider

$$\dot{\sigma} = -\mathbf{A}\mathbf{Q}x - \mathbf{A}\mathbf{c} + \mathbf{A}\mathbf{A}^T\mathcal{S}\mathcal{L}(\sigma). \tag{32}$$

With a similar analysis to the first stage, let the Lyapunov candidate

$$V_e = \frac{1}{2}\sigma^T[\mathbf{A}\mathbf{A}^T]^{-1}\sigma \tag{33}$$

assuming $\left\|\sigma^T[\mathbf{A}\mathbf{A}^T]^{-1}[\mathbf{A}\mathbf{Q}x + \mathbf{A}\mathbf{c}]\right\|_1 < \gamma$ with $\gamma > 0$, the derivative of (33) is given by

$$\dot{V}_e < \sigma^T\mathcal{S}\mathcal{L}(\sigma) + \gamma. \tag{34}$$

Defining $k_4 = \min\{k_{i4}, \ldots, k_{m4}\}$ and selecting $k_4 > \gamma$, the trajectories of (32) converge to zero in a fixed time $t_e > t_d$.

VI. Numerical Simulation Results

A. Linear Programming

Let the following linear programming problem [16]:

$$\begin{cases} \min_x & 4x_1 + x_2 + 2x_3 \\ \text{s.t} & x_1 - 2x_2 + x_3 = 2 \\ & -x_1 + 2x_2 + x_3 = 1 \\ & -5 \le x_1, x_2, x_3 \le 5 \end{cases}. \tag{35}$$

The proposed neural network (8), with al parameters equal to 10, gives the results shown in Fig. 1.

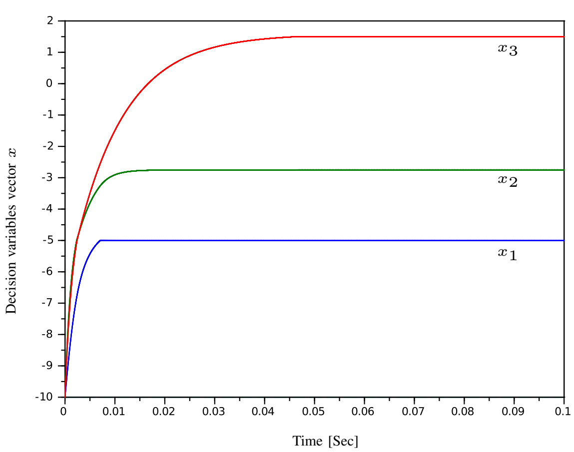

Fig. 1. Transient behavior of the x variables.

Here, it can be observed that the network converges to the optimal solution $x^* = [-5, -2.75, 1.5]$.

B. Quadratic Programming

Now, consider the following quadratic programming problem [12]:

$$\begin{cases} \min_x & -0.5x_1^2 + x_2^2 + 2x_1x_2 + 6x_1 - 2x_2 \\ \text{s.t} & 3x_1 - 2x_2 = 1 \\ & 0 \le x_1, x_2 \le 10 \end{cases}. \tag{36}$$

The proposed neural network (21), with al parameters equal to 10, gives the results shown in Fig. 2.

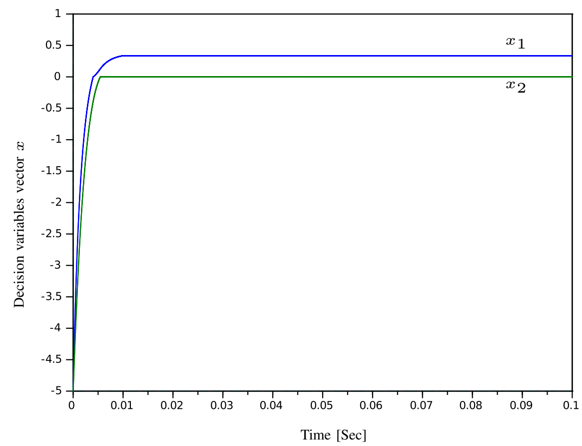

Fig. 2. Transient behavior of the x variables.

Here, it can be observed that the network converges to the optimal solution $x^* = [1/3, 0]^T$.

At this point, the proposed networks have been shown an acceptable performance. Similar results can be obtained with the approaches mentioned in the references. However, the Fig. 3 exposes the convergence features of the current approach against a network with hard-limiting activation functions

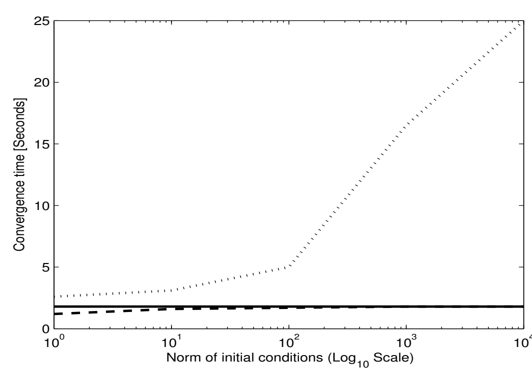

Fig. 3. Convergence time of both networks by growing initial condition norm

Here, the convergence time of the network with hard-limiting activation functions (dotted line) grows unboundedly with the norm of the initial condition, while the convergence time of the proposed network (dashed line) is asymptotically bounded by a constant for growing norm of the initial condition (solid line).

VII. Conclusions and Future Work

A new class of recurrent neural networks for linear and quadratic optimization problem has been proposed. The main feature of this proposal is the fixed time convergence time.

The design procedure and the stability proof was presented. As well, two simulations examples which exposes the performance of the proposed networks.

The presented approach opens the opportunity to apply additional classes of stabilizers and to solve a wider class of optimization problems. This consist of the future work.

References

[1] L. Chua and G.-N. Lin, "Nonlinear programming without computation," *IEEE Transactions on Circuits and Systems*, vol. 31, no. 2, pp. 182–188, 1984.

[2] D. Tank and J. Hopfield, "Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Transactions on Circuits and Systems*, vol. 33, no. 5, pp. 533–541, 1986.

[3] R. W. Brockett, "Dynamical systems that sort lists, diagonalize matrices and solve linear programming problems," in *Proc. 27th IEEE Conf. Decision and Control*, 1988, pp. 799–803.

[4] M. P. Kennedy and L. O. Chua, "Neural networks for nonlinear programming," *IEEE Transactions on Circuits and Systems*, vol. 35, no. 5, pp. 554–562, 1988.

[5] G. Wilson, "Quadratic programming analogs," vol. 33, no. 9, pp. 907–911, 1986.

[6] A. Rodriguez-Vazquez, R. Dominguez-Castro, A. Rueda, J. L. Huertas, and E. Sanchez-Sinencio, "Nonlinear switched capacitor 'neural' networks for optimization problems," *IEEE Transactions on Circuits and Systems*, vol. 37, no. 3, pp. 384–398, 1990.

[7] J. Wang, "Analysis and design of a recurrent neural network for linear programming," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 40, no. 9, pp. 613–618, 1993.

[8] Y. Xia and J. Wang, "A general methodology for designing globally convergent optimization neural networks," *IEEE Transactions on Neural Networks*, vol. 9, no. 6, pp. 1331–1343, 1998.

[9] S. V. Drakunov and V. Utkin, "Sliding mode control in dynamic systems," *International Journal of Control*, vol. 55, pp. 1029–1037, 1992.

[10] V. Utkin, J. Guldner, and J. Shi, *Sliding Mode Control in Electro-Mechanical Systems, Second Edition (Automation and Control Engineering)*, 2nd ed. CRC Press, 5 2009.

[11] E. K. P. Chong, S. Hui, and S. H. Zak, "An analysis of a class of neural networks for solving linear programming problems," *IEEE Transactions on Automatic Control*, vol. 44, no. 11, pp. 1995–2006, 1999.

[12] Q. Liu and J. Wang, "A one-layer recurrent neural network with a discontinuous hard-limiting activation function for quadratic programming," *IEEE Transactions on Neural Networks*, vol. 19, no. 4, pp. 558–570, 2008.

[13] ——, "Finite-time convergent recurrent neural network with a hard-limiting activation function for constrained optimization with piecewise-linear objective functions," vol. 22, no. 4, pp. 601–613, 2011.

[14] Z. Guo, Q. Liu, and J. Wang, "A one-layer recurrent neural network for pseudoconvex optimization subject to linear equality constraints," *IEEE Transactions on Neural Networks*, vol. 22, no. 12, pp. 1892–1900, 2011.

[15] Q. Liu and J. Wang, "A one-layer dual recurrent neural network with a heaviside step activation function for linear programming with its linear assignment application," in *Artificial Neural Networks and Machine Learning ICANN 2011*, ser. Lecture Notes in Computer Science, T. Honkela, W. Duch, M. Girolami, and S. Kaski, Eds. Springer Berlin Heidelberg, 2011, vol. 6792, pp. 253–260.

[16] ——, "A one-layer recurrent neural network with a discontinuous activation function for linear programming," *Neural Computation*, vol. 20, no. 5, pp. 1366–1383, Nov. 2007.

[17] Q. Liu, Z. Guo, and J. Wang, "A one-layer recurrent neural network for constrained pseudoconvex optimization and its application for dynamic portfolio optimization," *Neural Networks*, vol. 26, pp. 99 – 109, 2012.

[18] W. Karush, "Minima of functions of several variables with inequalities as side constraints," Master's thesis, Dept. of Mathematics, Univ. of Chicago, Chicago, Illinois., 1939.

[19] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Proc. Second Berkeley Symp. on Math. Statist. and Prob. (Univ. of Calif. Press)*, 1951.

[20] M. Forti, P. Nistri, and M. Quincampoix, "Generalized neural network for nonsmooth nonlinear programming problems," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 51, no. 9, pp. 1741–1754, 2004.

[21] L. Cheng, Z.-G. Hou, N. Homma, M. Tan, and M. M. Gupta, "Solving convex optimization problems using recurrent neural networks in finite time," in *Proc. Int. Joint Conf. Neural Networks IJCNN 2009*, 2009, pp. 538–543.

[22] Q. Liu, J. Cao, and G. Chen, "A novel recurrent neural network with finite-time convergence for linear programming," *Neural Computation*, vol. 22, no. 11, pp. 2962–2978, Aug. 2010.

[23] A. F. Filippov, *Dierential equations with discontinuous righthand sides*, Mathematics and . its Applications (Soviet Series), Eds. Kluwer Academic Publishers Group, Dordrecht, 1988.

[24] F. Clarke, *Optimization and Nonsmooth Analysis*, ser. Classics in applied mathematics. Society for Industrial and Applied Mathematics (SIAM), 1990.

[25] J. Aubin and A. Cellina, *Differential Inclusions: Set-Valued Maps and Viability Theory*, ser. Grundlehren Der Mathematischen Wissenschaften. Springer London, 2012.

[26] V. Utkin, *Sliding Modes in Control and Optimization*. Springer Verlag, 1992.

[27] A. Levant, "Sliding order and sliding accuracy in sliding mode control," *International Journal of Control*, vol. 58, no. 6, pp. 1247–1263, 1993.

[28] E. Cruz-Zavala, J. Moreno, and L. Fridman, "Uniform second-order sliding mode observer for mechanical systems," in *Variable Structure Systems (VSS), 2010 11th International Workshop on*, june 2010, pp. 14 –19.

[29] A. Polyakov, "Nonlinear feedback design for fixed-time stabilization of linear control systems," *IEEE Transactions on Automatic Control*, vol. 57, no. 8, pp. 2106–2110, 2012.