Neural Process Lett
DOI 10.1007/s11063-012-9255-8

Using Laplacian Eigenmap as Heuristic Information to Solve Nonlinear Constraints Defined on a Graph and Its Application in Distributed Range-Free Localization of Wireless Sensor Networks

Shuai Li · Zheng Wang · Yangming Li

Abstract In this paper, we are concerned with the problem of nonlinear inequalities defined on a graph. The feasible solution set to this problem is often infinity and Laplacian eigenmap is used as heuristic information to gain better performance in the solution. A continuous-time projected neural network, and the corresponding discrete-time projected neural network are both given to tackle this problem iteratively. The convergence of the neural networks are proven in theory. The effectiveness of the proposed neural networks are tested and compared with others via its applications in the range-free localization of wireless sensor networks. Simulations demonstrate the effectiveness of the proposed methods.

Keywords Projected dynamic neural network · Constrained optimization · Laplacian eigenmap · Wireless sensor networks

1 Introduction

With the development of studies on networked systems, e.g., metabolic networks [1], power networks [2], wireless sensor networks (WSNs) [3], protein networks [4], robot networks [5,6], many new problems, which are different from conventional problems due to the presence of topology constraints, are emerging and have posed appeals for efficient solutions.

S. Li (✉)
Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ 07030, USA
e-mail: lishuai8@gmail.com

Z. Wang
Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S3G4, Canada

Y. Li
Robot Sensor and Human-Machine Interaction Lab, Institute of Intelligent Machines, Chinese Academy of Sciences, Hefei 230031, Anhui, China
e-mail: ymli@iim.ac.cn

Published online: 17 November 2012

Among the problems defined on a graph, one class of problems can be described by a set of nonlinear inequalities relevant to the interactions between graph nodes. Typical examples include the range-free localization of WSNs, which is concerned with finding an estimation of sensor node positions based on topological constraints [7]. Other application examples include connectivity maintenance in networked robotic systems [8], tower positioning for cellular radio networks [9], the coverage problem in sensor networks [10], etc. In this paper, we formulate this class of problems as a constrained optimization problem with the nonlinear constraints defined on a graph. Due to the existence of the nonlinear constraints, the analytical solution of this problem is generally difficult to obtain and thus we aim to seek an iterative approach to solve the problem.

Resulting from the potential of parallel implementation and high efficiency in real-time computation, recurrent neural networks have received considerable studies in many application fields, such as signal processing [11,12], pattern classification [13,14], robotics [15,16], grammatical structure learning [17], optimization [18]. In this paper, we design recurrent neural networks to tackle the formulated nonlinear inequality constrained optimization problem defined on a graph in real time and apply the neural network model to range-free localization of WSNs.

Without the loss of generality, in this paper, the range-free localization of WSNs is used to test and compare the effectiveness of the proposed method with its peers. Range-free localization, which localizes the blind sensor nodes in a network by purely exploiting proximity information, is a costless by-product of wireless communication and thus has attracted intensive studies. In [19], a linear matrix inequality (LMI) based method is proposed to solve the problem. In this framework, the computation is assumed by a single base station. Therefore this method is fragile to the failure of the base station and has a strict requirement on the computational power of the base station. In contrast, for distributed range-free localization methods, such as the DV-HOP method [20] and APIT method [21], computation burdens are distributed to every nodes in the network, resulting in higher scalability to large-scale networks and robustness against node failures. We refer the reader to paper [21] and citations therein for a complete knowledge on range-free localization in WSNs.

In spite of the great successes of recurrent neural networks in both theory and applications [12,22–26], difficulties still exist when the recurrent neural network is used to solve nonlinear inequalities defined on a large scale graph in aims of a distributed solution. In our previous work, [27] and its extension [28] consider the problem of inter-phone localization with a similar mathematical formulation. In [27,28], the nonlinear inequality constraints are modeled as a penalty function and gradient neural networks are employed to solve the problem. In [29], the problem is viewed as a constrained optimization problem with no explicit objective function and a recurrent neural network developed in this paper finds a feasible solution to the problem. Actually, there often exists infinite number of feasible solutions for the problem of nonlinear inequalities defined on a graph and some heuristic information is applicable to reach a plausible better solution. On this purpose, a two stage approach is proposed in [30] to solve the problem. In this method, the neural network approach proposed in [27] is first applied to obtain a feasible solution and a solution improvement neural network, which is initialized with the obtained feasible solution, is then used to refine the solution based on the heuristic information. It is noteworthy that the dynamics of the solution improvement neural network is confined inside the constraint sets and remains feasible along the dynamic evolution due to the introduction of a barrier term in the neural dynamics. However, as argued in [31,32], the final solution often deviates from the theoretical optimal one, since the barrier term is equivalents to an extra penalty term in the objective function. To remedy this problem, in this paper, we propose a novel combined neural network, with both its continuous-time

version and the discrete-time version, to solve the problem by recursively projecting the solution to the feasible set. The proposed neural network, which is in contrast to the two neural networks, namely the feasible solution neural network and the solution improvement neural network, proposed in [30] for the same problem, completes the tasks with theoretically identical solution to the formulated optimization problem but with much simpler structure compared with the networks proposed in our previous work [30].

The remainder of this paper is organized as follows. In Sect. 2, the problem of nonlinear inequalities defined on a graph is modeled as an optimization problem with the heuristic information as the objective function and the nonlinear inequalities as constraints. For the convenience of analysis, in Sect. 3, the problem is considered in its dual space. Then, A continuous-time recurrent neural network and a discrete-time recurrent neural network are respectively proposed in Sects. 4 and 5 to solve the problem iteratively. Section 6 applies the proposed approaches to the problem of distributed range-free localization of WSNs. In Sect. 7, simulation examples are given to demonstrate the effectiveness of the methods. Section 8 concludes this paper.

2 Problem Formulation

In this paper, we consider the following inequalities:

$$f_{ij}(x_i, x_j) \leq 0 \quad \text{for } j \in \mathbb{N}(i) \bigcup \{i\}, \forall i \in \mathbb{V} \tag{1}$$

which is defined on a graph $\mathcal{G}(\mathbb{V}, \mathbb{E})$ with $\mathbb{V}$, $\mathbb{E}$ denoting the vertex set and the edge set, $\mathbb{N}(i) = \{j \in \mathbb{V} | \{i, j\} \in \mathbb{E}\}$ denoting the neighbor set of vertex i, $x_i \in \mathbb{R}^m$ representing the state variable associated with vertex i, $f_{ij}(x_i, x_j) \in \mathbb{R}$ denoting a nonlinear convex function with respect to x_i and x_j. Note that we assume the accurate value of x_k for $k \in \mathbb{B}$ are pre-known, where $\mathbb{B}$ represents the beacon vertex sets.

Mostly, the solution to the problem (1) is not unique. Each solution is associated with a particular distribution meeting the topological constraints. Incorporating heuristic information into the problem helps find the most likely solution among all the feasible ones. Actually, the one with an uniform distribution of vertices often outperforms other distributions as it corresponds to the maximum entropy estimation in the feasible solution set [33]. Therefore, like in [30], heuristic information is encoded in the objective function to enforce the tendency to uniform distribution of vertices,

$$\text{minimize} \qquad c_0 \sum_{i=1}^{n} \sum\nolimits_{j \in \mathbb{N}(i)} (x_i - x_j)^T (x_i - x_j) \tag{2a}$$

$$\text{subject to} \quad f_{ij}(x_i, x_j) \leq 0 \quad \text{for } j \in \mathbb{N}(i) \bigcup \{i\}, \forall i \in \mathbb{V} \tag{2b}$$

where $c_0 > 0$ is a constant. Clearly, for all $c_0 > 0$ the optimization problem (2) shares the same optimal solutions and we can choose $c_0 = 1$ without alternating the optimality. However, introducing c_0 in the objective function results in different evolving manifold for the recurrent neural networks as will be discussed in Remark 1.

In this paper, the variable x_i is defined in m-dimensional space and thus the objective function can be expressed in a compact form as,

$$c_0 \sum_{i=1}^{n} \sum_{j \in \mathbb{N}(i)} (x_i - x_j)^T (x_i - x_j)$$

$$= 2c_0 x^T (L \otimes I_m) x \tag{3}$$

where x is the vector formed by stacking all x_i from $i = 1$ to $i = n$, i.e., $x = [x_1^T, x_2^T, ..., x_n^T]^T$, I_m is a $m \times m$ identify matrix, '$\otimes$' is the Kronecker product operator which is defined as follows for matrix $A = [a_{ij}]$ and matrix B,

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix} \tag{4}$$

and L is the Laplacian matrix with the $i-j$th element defined as follows,

$$L_{ij} = \begin{cases} -1 & \text{for } i \neq j \text{ and } j \in \mathbb{N}(i) \\ 0 & \text{for } i \neq j \text{ and } j \notin \mathbb{N}(i) \\ -\sum_{k \in \mathbb{N}(i)} L_{ik} & \text{for } i = j \end{cases} \tag{5}$$

The quadratic expression shown in (3) is widely used as the objective function in Laplacian eigenmap based manifold learning [34] to obtain the relative coordinates based on relative distance measurements. The objective function in (2) thus tends to regulate x_i to its true position according to the topological constraints and on the other hand the inequalities in (2) imposes additional constraints and further improves the estimation accuracy on x_i for all possible i.

Due to the presence of the nonlinear inequality constraints in (2), it is generally difficult to obtain the close-form solution to the problem. Instead, we seek an iterative procedure to solve the problem asymptomatically. It is noteworthy that the so-called projected gradient method [35] directly solves the constrained optimization problem by projecting the solution without considering constraints into the convex constraint set. However, for the problem (2), it is difficult to analytically derive the projection of the variables onto the constraint set. This difficulty motivates us to consider the problem in dual space and use another iterative procedure to approach the projection by recursively adjusting the dual variables.

3 Dual Space Transformation

In this section, we study the problem (2) equivalent in the dual space.

According to Karash–Kuhn–Tucker (KKT) conditions [35], the solution to problem (2) satisfies,

$$\begin{aligned} &4c_0 \textstyle\sum_{j \in \mathbb{N}(i)} (x_i - x_j) + \lambda_{ii} \frac{\partial f_{ii}(x_i, x_i)}{\partial x_i} \\ &+ \textstyle\sum_{j \in \mathbb{N}(i)} \left(\lambda_{ij} \frac{\partial f_{ij}(x_i, x_j)}{\partial x_i} + \lambda_{ji} \frac{\partial f_{ji}(x_j, x_i)}{\partial x_i} \right) = 0 \ \forall i \in \mathbb{V} \end{aligned} \tag{6a}$$

$$f_{ij}(x_i, x_j) \begin{cases} = 0 & (\lambda_{ij} > 0) \\ < 0 & (\lambda_{ij} = 0) \end{cases} \quad \text{for } j \in \mathbb{N}(i) \bigcup \{i\}, \forall i \in \mathbb{V} \tag{6b}$$

where $\lambda_{ij} \in \mathbb{R}$, $\lambda_{ji} \in \mathbb{R}$ and $\lambda_{ii} \in \mathbb{R}$ are all dual variables. Note that (6b) can be simplified into the following form,

$$g\left(f_{ij}(x_i, x_j) + \lambda_{ij}\right) = \lambda_{ij} \quad \text{for } j \in \mathbb{N}(i) \bigcup \{i\}, \forall i \in \mathbb{V} \tag{7}$$

with the function $g(u) = [g_1(u_1), g_2(u_2), ..., g_k(u_k)]^T$ for $u = [u_1, u_2, ..., u_k]^T$ defined as follows for $i = 1, 2, ..., k$,

$$g_i(u_i) = \begin{cases} u_i & \text{if } u_i > 0 \\ 0 & \text{if } u_i \leq 0 \end{cases} \tag{8}$$

It is evident that (2) is a convex constrained convex programming problem by noting that the constraints are assumed to be convex and the objective function is convex since it can be regarded as the summation of compositions of the squared norm operation $\|\cdot\|^2$ with a linear function $(x_i - x_j)$. Accordingly, the application of KKT conditions to the problem (2) results in an equivalent solution, i.e., the solution to (6a) and (7) is equivalent to the solution of (2). To summarize, the solution of (2) is the solution of the following equations,

$$\begin{aligned} &4c_0 \textstyle\sum_{j\in\mathbb{N}(i)}(x_i - x_j) + \lambda_{ii}\frac{\partial f_{ii}(x_i,x_i)}{\partial x_i} \\ &+ \textstyle\sum_{j\in\mathbb{N}(i)}\left(\lambda_{ij}\frac{\partial f_{ij}(x_i,x_j)}{\partial x_i} + \lambda_{ji}\frac{\partial f_{ji}(x_j,x_i)}{\partial x_i}\right) = 0 \ \forall i \in \mathbb{V} \end{aligned} \tag{9a}$$

$$g\left(f_{ij}(x_i, x_j) + \lambda_{ij}\right) = \lambda_{ij} \quad \text{for } j \in \mathbb{N}(i)\bigcup\{i\}, \forall i \in \mathbb{V} \tag{9b}$$

The close-form solution to the Eq. (9) is difficult to obtain due to the presence of nonlinear terms in it. Instead, in the following sections, we will design recurrent neural networks to solve the problem iteratively. Concretely, a continuous-time neural network, modeled by a ordinary differential equation and a discrete-time recurrent neural network, modeled by a difference equation, will be presented in the following sections to solve (9).

4 Continuous-Time Neural Network Solution

In this section, the continuous-time neural network model for solving (9), or equivalently (2), will be given and its convergence will be proved in theory.

4.1 The Continuous-Time Model

We use a continuous-time recurrent neural network to solve the variables in (9) as follows:

$$\begin{aligned} \dot{x}_i = -\epsilon\Bigg(&4c_0 \sum_{j\in\mathbb{N}(i)}(x_i - x_j) + \lambda_{ii}\frac{\partial f_{ii}(x_i, x_i)}{\partial x_i} \\ &+ \sum_{j\in\mathbb{N}(i)}\left(\lambda_{ij}\frac{\partial f_{ij}(x_i, x_j)}{\partial x_i} + \lambda_{ji}\frac{\partial f_{ji}(x_j, x_i)}{\partial x_i}\right)\Bigg) \ \forall i \in \mathbb{V} \\ \dot{\lambda}_{ij} = \epsilon g\left(f_{ij}(x_i, x_j) + \lambda_{ij}\right) - \epsilon\lambda_{ij} \quad &\text{for } j \in \mathbb{N}(i)\bigcup\{i\}, \forall i \in \mathbb{V} \end{aligned} \tag{10}$$

where $\epsilon > 0$ is a scaling factor, $c_0 > 0$ is a constant, the function $g(\cdot)$ is defined in (8), $\lambda_{ij} \in \mathbb{R}$, $\lambda_{ii} \in \mathbb{R}$ and $\lambda_{ji} \in \mathbb{R}$ are dual variables associated with the edge $\{i, j\} \in \mathbb{E}$, the vertex i, and the edge $\{j, i\} \in \mathbb{E}$, respectively.

The proposed neural model solves the problem (2) in a distributed fashion as can be observed that all information required to update x_i comes from the vertex x_j with j neighboring i and the edges $i - j$, which bridges vertex i to its neighbor j and all the information required to update λ_{ij}, which corresponds to the connection between an existing edge i–j

on the graph, comes from either itself or the two connected vertices i and j. Therefore, the proposed neural approach is scalable to a large network with a huge number of vertices involved.

on the parameter c_0, we have the following remark,

Remark 1 In the neural dynamics (10), the term $-4c_0\epsilon\sum_{j\in\mathbb{N}(i)}(x_i-x_j)$ represents the effect of the objective function while the other terms are all introduced by the constraints in (2). For the dynamic evolution of x_i in the continuous-time neural network (10), changing the value of c_0 varies the dynamic transient of the system.

4.2 Convergence of the Continuous-Time Model

For the convenience of analysis, the neural network dynamic (10) can be equivalently written into a compact form,

$$\begin{aligned}\dot{x} &= -4c_0\epsilon(L\otimes I_m)x-\epsilon\left(\nabla^T\bar{F}(x)\right)\bar{\Lambda}\\ \dot{\bar{\Lambda}} &= \epsilon\left(g\left(\bar{F}(x)+\bar{\Lambda}\right)-\bar{\Lambda}\right)\end{aligned}\tag{11}$$

where x is a mn dimensional vector with $n=|\mathbb{V}|$ denoting the number of vertices on the graph, m denoting the dimension of x_i for $i\in\mathbb{V}$ and $x=[x_1^T,x_2^T,...,x_n^T]^T$, L is the Laplacian matrix with its $i-j$th element as defined in (5), $F(x)\in\mathbb{R}^{n\times n}$ is a matrix function of x with the ijth entry defined as follows:

$$F_{ij}(x)=\begin{cases}f_{ij}(x_i,x_j) & \text{for } i\in\mathbb{V},\ j\in\mathbb{V}\text{ and } j\in\mathbb{N}(i)\\ 0 & \text{for } i\in\mathbb{V},\ j\in\mathbb{V}\text{ and } j\notin\mathbb{N}(i)\end{cases}\tag{12}$$

Similarly, $\Lambda\in\mathbb{R}^{n\times n}$ is defined as:

$$\Lambda_{ij}=\begin{cases}\lambda_{ij} & \text{for } i\in\mathbb{V},\ j\in\mathbb{V}\text{ and } j\in\mathbb{N}(i)\\ \mu_{ij} & \text{for } i\in\mathbb{V},\ j\in\mathbb{V}\text{ and } j\notin\mathbb{N}(i)\end{cases}\tag{13}$$

where $\mu_{ij}\in\mathbb{R}$ and is always initialized to be zero, i.e., $\mu_{ij}(0)=0$. In (11), $\bar{F}(x)$ is defined to be a n^2 dimensional vector by stacking the columns of $F(x)$ into a single column vector and $\bar{\Lambda}$ is so defined by stacking the columns of Λ into a single column vector.

For a general projection neural network, the following lemma holds,

Lemma 1 ([36]) Assume that $\nabla h_1(x)$ is positive semi-definite on Ω and $h_2(x)$ is convex on Ω. Then the following dynamic system (14) with any initial point $(x_1(0),x_2(0),x_3(0))\in\Omega\times{\mathbb{R}_+}^m\times\mathbb{R}^r$ is stable in the sense of Lyapunov and converges exponentially to its equilibrium point.

$$\begin{aligned}\dot{x}_1 &= -\epsilon\left(x_1-P_\Omega\left(x_1-h_1(x_1)-(\nabla^T h_2(x_1))x_2-D^T x_3\right)\right)\\ \dot{x}_2 &= -\epsilon\left(x_2-g\left(x_2+h_2(x_1)\right)\right)\\ \dot{x}_3 &= -\epsilon(Dx_1-d)\end{aligned}\tag{14}$$

where x_1, x_2 and x_3 are state variables of the dynamic system, $P_\Omega(x)$ is the Euclidean projection of x onto the set Ω defined as $P_\Omega(x)=argmin_{y\in\Omega}\|x-y\|$ with $\|\cdot\|$ denoting the Euclidean norm, $\epsilon>0$ is a scaling factor, x_1, x_2 and x_3 are vector variables of appropriate sizes, d and D are vector and matrix of appropriate sizes, respectively, $g(\cdot)$ is as defined in (8).

By choosing $D = 0, d = 0, h_1(x) = 4c_0(L \otimes I_m)x$ (note that $\nabla h_1(x) = 4c_0(L \otimes I_m)$, which is indeed positive semi-definite), $h_2(x) = \bar{F}(x)$, $\Omega = \mathbb{R}^k$ in Lemma 1, the system (14) reduces to the recurrent neural network (11) or (10), which is considered in this paper. Therefore, we can state the convergence result on the neural network (10) as follows,

Theorem 1 *The recurrent neural network* (10), with $f_{ij}(x_i, x_j)$ convex for $j \in \mathbb{N}(i) \bigcup \{i\}$, $\forall i \in \mathbb{V}, \epsilon > 0$ and the initial value of $\lambda_{ij}(0) \geq 0$ for all $i \in \mathbb{V}$ and $j \in \mathbb{E} \bigcup \{i\}$, exponentially converges to a solution of problem (1).

Proof There are two parts in the proof:

First, By choosing $D = 0, d = 0, h_1(x) = 4c_0(L \otimes I_m)x, h_2(x) = \bar{F}(x), \Omega = \mathbb{R}^k$, with noting that $P_\Omega(x) = x$ in this case, the dynamic system (14) reduces to the following,

$$\begin{aligned} \dot{x}_1 &= -4c_0\epsilon(L \otimes I_m)x - \epsilon \left(\nabla^T \bar{F}(x_1)\right) x_2 \\ \dot{x}_2 &= \epsilon \left(g\left(x_2 + \bar{F}(x_1)\right) - x_2\right) \\ \dot{x}_3 &= 0 \end{aligned} \tag{15}$$

The functions in the above dynamic equations follows the requirement in Lemma 1 and thus the convergence conclusion drawn in Lemma 1 applies to system (15), or equivalently (10), by noting that x_3 has no influence to the dynamics of x_1 and x_2 in (15).

Second, note that problem (1) is a convex constrained convex programming problem and the KKT condition results in an equivalent solution to the problem. Thus, the equilibrium point (10), which is identical to the solution of (9) is indeed the solution to problem (1).

Together, we conclude that the recurrent neural network (10), with the initial value of $\lambda_{ij}(0) \geq 0$ for all $i \in \mathbb{V}$ and $j \in \mathbb{E} \bigcup \{i\}$, exponentially converges to its equilibrium point, which is also the solution of problem (1). This completes the proof. □

5 Extension: Discrete-Time Neural Network Solution

In this section, the discrete-time neural network model for solving (9), or equivalently (2), will be given and its convergence will be proved in theory.

5.1 The Discrete-time Model

In this part, we use a discrete-time recurrent neural network to solve the variables in (9) as follows:

$$\begin{aligned} x_i^{t+1} &= x_i^t - \epsilon^t \left(4c_0 \sum_{j \in \mathbb{N}(i)} (x_i^t - x_j^t) + \lambda_{ii}^t \frac{\partial f_{ii}(x_i^t, x_i^t)}{\partial x_i^t} \right. \\ &\quad \left. + \sum_{j \in \mathbb{N}(i)} \left(\lambda_{ij}^t \frac{\partial f_{ij}(x_i^t, x_j^t)}{\partial x_i^t} + \lambda_{ji}^t \frac{\partial f_{ji}(x_j^t, x_i^t)}{\partial x_i^t} \right) \right) \quad \forall i \in \mathbb{V} \\ \lambda_{ij}^{t+1} &= g\left(\epsilon^t f_{ij}(x_i^t, x_j^t) + \lambda_{ij}^t\right) \quad \text{for } j \in \mathbb{N}(i) \bigcup \{i\}, \forall i \in \mathbb{V} \end{aligned} \tag{16}$$

where the superscript t and $t + 1$ represent the time step, $\epsilon^t > 0$ is the step length at time t. the function $g(\cdot)$ is defined in (8), $\lambda_{ij} \in \mathbb{R}$, $\lambda_{ii} \in \mathbb{R}$ and $\lambda_{ji} \in \mathbb{R}$ are dual variables associated with the edge $\{i, j\} \in \mathbb{E}$, the vertex i, and the edge $\{j, i\} \in \mathbb{E}$, respectively.

The proposed discrete-time neural model also solves the problem (2) in a distributed fashion as all information required to update x_i and λ_{ij} both come from their neighborhood.

On the difference between the proposed discrete-time neural network (16) and the projected gradient descending method for problem (2), we have the following remark,

Remark 2 It is noteworthy that the so-called projected gradient method [35] directly solves the constrained optimization problem by projecting the solution into the convex constraint set and the convergence can be guaranteed under some mild conditions. However, for problem (2), it is difficult to obtain an analytical expression of the projection onto the nonlinear constraints. Differently, by introducing the dual variables and considering the problem in dual space, the problem is converted to a projected optimization problem onto a bounded region and can be solved recursively by the discrete-time neural network (16) (the nonlinear function $g(\cdot)$ corresponds to the projection onto a bounded region).

5.2 Convergence of the Discrete-Time Model

For the convenience of analysis, the neural network dynamic (16) can be equivalently written into a compact form,

$$\begin{aligned} x^{t+1} &= x^t - \epsilon^t \left(4c_0(L \otimes I_m)x^t + \left(\nabla^T \bar{F}(x^t)\right) \bar{\Lambda}^t\right) \\ \bar{\Lambda}^{t+1} &= g\left(\epsilon^t \bar{F}(x^t) + \bar{\Lambda}^t\right) \end{aligned} \tag{17}$$

where the superscript t and $t+1$ denote the time step, ϵ^t is the step length at time t, x^t is a mn dimensional vector with $n = |\mathbb{V}|$ denoting the number of vertices on the graph, m denoting the dimension of x_i^t for $i \in \mathbb{V}$ and $x^t = [x_1^{tT}, x_2^{tT}, ..., x_n^{tT}]^T$, L is the Laplacian matrix with its $i-j$th element as defined in (5), $F(x^t) \in \mathbb{R}^{n\times n}$ is a matrix function of x as defined in (12), $\Lambda \in \mathbb{R}^{n\times n}$ is as defined in (13).

Noticing that the function $g(\cdot)$ actually is a projection operator onto non-negative coordinates, the updating law in the iterative Eq. (17) is consistent with the the subgradient iteration of the following saddle-point problem proposed in [37],

$$\min_{x\in\mathbb{R}^{mn}} \max_{\bar{\Lambda}\in\mathbb{R}^{mn+}} \quad 2c_0 x^T (L \otimes I_m)x + \bar{\Lambda}^T \bar{F}(x) \tag{18}$$

where $\mathbb{R}^{mn+} = \{x = [x_1, x_2, ..., x_{mn}] \in \mathbb{R}^{mn}, x_i \geq 0 \,\forall i = 1, 2, ..., mn\}$. As addressed in [37], (16) converges to a bounded region centered at the saddle point of (18), or the equilibrium point of (16) by choosing a constant step size $\epsilon^t = \epsilon$ and thus can be used to approximate the saddle point. As argued in [37], an adaptive diminishing step size ϵ^t that varies with t can be used to get better convergence accuracy.

Until now, we have shown that the convergence property of the proposed discrete-time neural network based on the work [37]. We next show that the equilibrium point of (16), i.e., the saddle point of (18), is identical to the solution of problem (2), or equivalently the solution of (9). By comparing the equations for the equilibrium points of (16) with (9), it is evident that we only need to show that the equation $\lambda_{ij} = g\left(f_{ij}(x_i, x_j) + \lambda_{ij}\right)$ is equivalent to $\lambda_{ij} = g\left(\epsilon^t f_{ij}(x_i, x_j) + \lambda_{ij}\right)$ for $\epsilon^t > 0$ to conclude the result. Noticing that this first equation is equivalent to (6b) and the second one is equivalent to the following one,

$$f_{ij}(x_i, x_j) \begin{cases} = 0 & (\epsilon^t \lambda_{ij} > 0) \\ < 0 & (\epsilon^t \lambda_{ij} = 0) \end{cases} \quad \text{for } j \in \mathbb{N}(i) \bigcup \{i\}, \forall i \in \mathbb{V} \tag{19}$$

Together with the fact that $\epsilon^t > 0$, we find (19) is identical to (6b) and thus suffices the conclusion that the the equilibrium point of (16) is the solution of the problem (2).

6 Range-Free Localization of WSNs with the Proposed Methods

To demonstrate the effectiveness of the proposed neural network, we apply this model to solve the range-free localization problem in WSNs. This problem can be expressed mathematically as follows [19]:

$$\|x_i - x_j\| \leq R \quad \text{for } j \in \mathbb{N}(i) \tag{20}$$

where R is the maximum communication range, x_i, x_j are the position of node i and node j, respectively, I_k is a $k \times k$ identity matrix with k representing the dimension of variable x.

This model is widely investigated in the past years to localize wireless sensors in a network without distance measurement [21,38–41]. It is evident that (20) falls into the framework of problem (1) by defining $f_{ij}(x_i, x_j) = (x_i - x_j)^T(x_i - x_j) - R^2$ for $j \in \mathbb{N}(i), \forall i \in \mathbb{V}$ and $f_{ii}(x_i, x_i) = 0$ for $\forall i \in \mathbb{V}$ and thus can be solved by using the proposed iterative approaches. Specifically, we have the following continuous-time recurrent neural network model to solve the distributed range-free localization problem (20) by substituting the particular expression of f_{ij} for this problem inside (10),

$$\begin{aligned}
\dot{x}_i &= -\epsilon \sum_{j \in \mathbb{N}(i)} (4c_0 + 2\lambda_{ij} + 2\lambda_{ji})(x_i - x_j) \ \forall i \in \mathbb{V} \\
\dot{\lambda}_{ij} &= \epsilon g\left((x_i - x_j)^T(x_i - x_j) - R^2 + \lambda_{ij}\right) - \epsilon\lambda_{ij} \quad \text{for } j \in \mathbb{N}(i) \bigcup \{i\}, \forall i \in \mathbb{V}
\end{aligned} \tag{21}$$

The discrete-time recurrent neural network model for solving the distributed range-free localization problem (20) writes as follows,

$$\begin{aligned}
x_i^{t+1} &= x_i^t - \epsilon^t \sum_{j \in \mathbb{N}(i)} (4c_0 + 2\lambda_{ij}^t + 2\lambda_{ji}^t)(x_i^t - x_j^t) \ \forall i \in \mathbb{V} \\
\lambda_{ij}^{t+1} &= g\left(\epsilon^t\left((x_i^t - x_j^t)^T(x_i^t - x_j^t) - R^2\right) + \lambda_{ij}^t\right) \quad \text{for } j \in \mathbb{N}(i) \bigcup \{i\}, \forall i \in \mathbb{V}
\end{aligned} \tag{22}$$

Apparently, the function $f_{ij}(x_i, x_j) = (x_i - x_j)^T(x_i - x_j) - R^2$ is convex for $j \in \mathbb{N}(i), \forall i \in \mathbb{V}$ and the convergence conclusions drawn in the previous sections apply to both the continuous-time neural network (21) and the discrete-time neural network (22).

7 Numerical Investigation

In this section, we consider a two 2-dimensional localization problem with blind nodes randomly deployed to show the effectiveness of the proposed approaches.

7.1 Simulation Setup

In this set of experiments, 202 blind nodes are randomly deployed to a normalized 1×1 square with 28 beacon nodes uniformly positioned around the perimeter. The maximum sensor range is set to be $R = 0.13$, the scaling factor $\epsilon = 10^5$ and the weighting coefficient $c_0 = 1$ for the neural network. The layout of the sensor network considered in this simulation is as shown in Fig. 1.

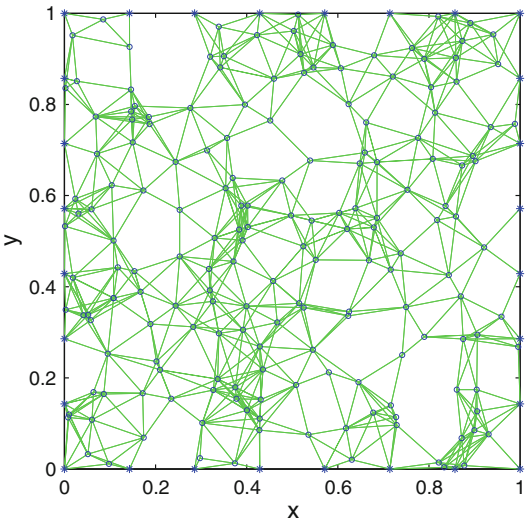

Fig. 1 True positions of nodes in the WSN and the communication topology in the simulation example. In this figure, the *green line*, the *blue circle* and the *blue star* represent the communication link, blind nodes and the beacon nodes, respectively. Color figure online

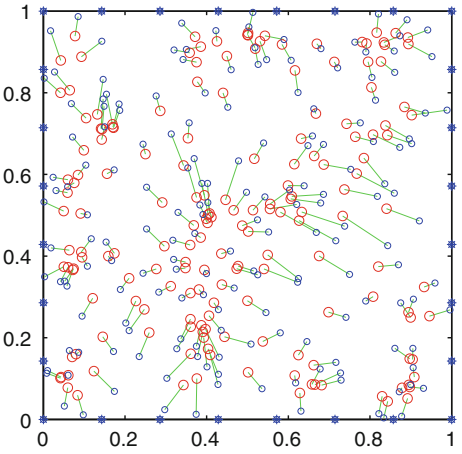

Fig. 2 Position estimation results by the proposed continuous-time neural network. In the figure, the *red circle*, the *blue circle* and the *blue star* represent the position estimation, the true position of blind nodes and the position of beacon nodes, respectively. Color figure online

7.2 Performance with the Proposed Continuous-Time Neural Network

Figure 2 shows the estimated positions of blind sensor after running the neural network for 10^{-4} s. The transient of estimated positions in x and y directions are plotted in Figs. 3 and 4, respectively. Note that the quantity $\frac{1}{N_{link}} \sum_{j\in\mathbb{N}_i, i\in\mathbb{V}} max(\|x_i - x_j\|^2, 0)$ (N_{link} represents the total number of communication links) measures to what extent the inequality constraints are violated since it equals zero if all the constraints are satisfied. Figure 5 shows the evolution of $\frac{1}{N_{link}} \sum_{j\in\mathbb{N}_i, i\in\mathbb{V}} max(\|x_i - x_j\|^2, 0)$ with time. which is an quantitive evaluation of the feasibility of the solution to the WSN localization problem (20). The value starts from 0.3649 and drops to 3.27×10^{-4} at the end of the simulation, which demonstrates that the inequality constraints are approximately satisfied by the ultimate output of the neural network.

7.3 Performance Comparison with Other Methods

Note that by choosing the weighting parameter $c_0 = 0$ in (18), the problem reduces to the feasible solution problem without imposing heuristic information [19,27–29] and thus the

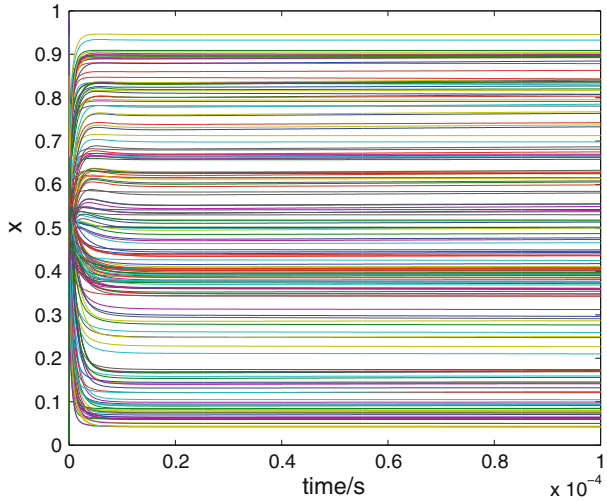

Fig. 3 Transient of the position in x direction estimated by the proposed continuous-time neural network

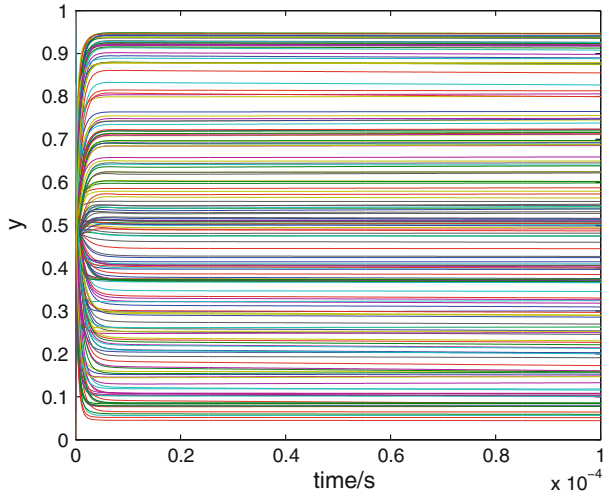

Fig. 4 Transient of the position in y direction estimated by the proposed continuous-time neural network

solution generated by the neural network with c_0 in (11) is representative to the solutions generated by the methods proposed in [19,27–29]. In this part, we compare the simulation results with $c_0 > 0$ in (11), which considers the heuristic information and the scenario with $c_0 = 0$ in (11). With $c_0 = 0$ and the other parameters set up in the same value, the position estimated by the neural network is shown in Fig. 6. Comparing the result shown in Fig. 6 and the result shown in Fig. 2, the latter one clearly stretches the position estimation to a more evenly distribution due to the introduction of the heuristic information. In quantity, we use the estimation error defined as $E = \sqrt{\sum_{i=1}^{n} (x_i - x_i^r)^T (x_i - x_i^r)/n}$ with x_i^r denoting the real position of the ith blind sensor node, to evaluate the localization accuracy. By running Monte Carlo simulation for the neural networks with both $c_0 = 0$ and $c_0 = 1$ for 50 times,

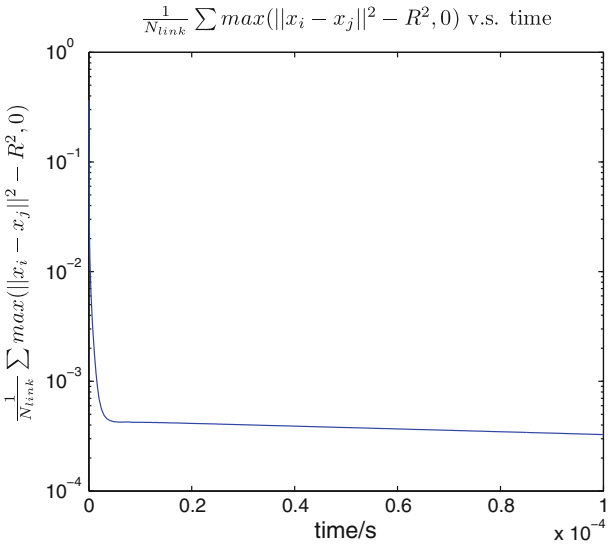

Fig. 5 Time evolution of $\frac{1}{N_{link}}\sum_{j\in\mathbb{N}_i, i\in\mathbb{V}} max(\|x_i - x_j\|^2 - R^2, 0)$ for estimation made by the proposed continuous-time neural network

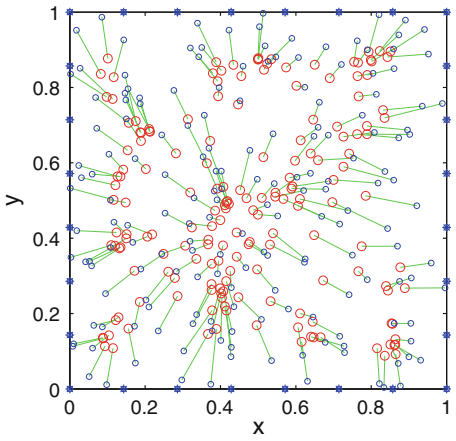

Fig. 6 Position estimation results by setting $c_0 = 0$ in the neural network, the solution of which is representative to the feasible solution problem investigated in [19,27–29]. In the figure, the *red circle*, the *blue circle* and the *blue star* represent the position estimation, the true position of blind nodes and the position of beacon nodes, respectively. Color figure online

the averaged estimation error E for the 50 simulations are 0.0856 and 0.0580 respectively, showing a clear improvement in performance the by introducing the heuristic information (more than 30 % reduction in error E). Table 1 briefly summarize the comparisons between the proposed continuous-time neural network and the methods investigated in [19,27–29].

8 Conclusions

In this paper, the problem of nonlinear constraints defined on a graph is considered and recurrent neural network solutions, incorporated with Laplacian eigenmap as heuristic information for a plausible better solutions, are supplied to solve the problem recursively in a distributed fashion. The convergence of the proposed neural networks are proven in theory. The neural

Table 1 Comparisons between methods to solve the problem of range-free localization of wireless sensor networks

	LMI based method [19]	Feasible solution neural network method [27,28]	Two neural network method [29]	The proposed method
Centralized vs. distributed	Centralized	Distributed	Distributed	Distributed
Analytical vs. iterative	Iterative (via semi-definite programming)	Iterative	Iterative	Iterative
Heuristic information?	No	No	Yes	Yes

network approaches are applied to WSN localization in a distributed, routing-free, range-free way. Finally, simulations demonstrate efficiency and accuracy of the method.

Acknowledgements This work was supported by NSFC Grant: 61105090. Shuai Li would like to thank the motivation and inspiration to him by his modified version of the lyric in the song 'Hello, tomorrow' by the pop-music group named Milk-Coffee singing that "I cannot cease from running with the fear of falling down in darkness. Never give up the belief of a promising tomorrow ahead glittering out hope and vigor, and the belief that I will never lose but can always gain support from those backing shoulders. Remain smile and positive, as positive as in the first trial, even after 1,000 times failure with tearing at difficulties. Resume the running after every crying. Life is the process of loss and gain at the same moment. Saying hello to tomorrow however weak the sound may be. Every trying tells and reminds me what braveness really is and a real man should be".

References

1. Jeong H, Tombor B, Albert R, Oltvai ZN, Barabasi AL (2000) The large-scale organization of metabolic networks. Nature 407:651
2. Buldyrev SV, Parshani R, Paul G, Stanley HE, Havlin S (2009) Catastrophic cascade of failures in interdependent networks. Nature 464(7291):1025–1028
3. Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E (2002) Wireless sensor networks: a survey. Comput Netw 38:393–422
4. Mohammed JZ, Vinay N, Deb B, Chris B (2004) Predicting protein folding pathways. Bioinformatics 20(1):386–393
5. Cao YU, Fukunaga AS, Kahng AB (1997) Cooperative mobile robotics: antecedents and directions. Auton Robots 4:226–234
6. Bullo F, Cortés J, Martínez S (2009) Distributed control of robotic networks. Applied Mathematics Series. Princeton University Press. Electronically available at http://coordinationbook.info
7. Kulaib AR, Shubair RM, Al-Qutayri MA, Ng JWP (2011) An overview of localization techniques for wireless sensor networks. Innovations in Information Technology (IIT), 2011 International Conference on, pp 167–172, April 2011
8. Reich J, Misra V, Rubenstein D, Zussman G (2011) Connectivity maintenance in mobile wireless networks via constrained mobility. In INFOCOM, 2011 Proceedings IEEE, pp 927–935, April 2011
9. Mathar R, Niessen T (2000) Optimum positioning of base stations for cellular radio networks. Wirel. Netw 6(6):421–428
10. Ghrist R, Muhammad A (2005) Coverage and hole-detection in sensor networks via homology. In IPSN '05: Proceedings of the 4th international symposium on information processing in sensor networks. IEEE Press, Piscataway
11. Li S, Wang Y, Yu J, Liu B (2013) A nonlinear model to generate the winner-take-all competition. Commun Nonlinear Sci Numer Simul 18(3):435–442
12. Xia Y (2009) A compact cooperative recurrent neural network for computing general constrained l1 norm estimators. IEEE Trans Signal Process 57(9):3693–3697

13. Niranjan B, Burrows TL, Niranjan M (1994) The use of recurrent neural networks for classification. In IEEE workshop on neural networks for signal processing IV pp 117–125
14. Husken M, Stagge P (2003) Recurrent neural networks for time series classification. Neurocomputing 50:223–235
15. Li S, Cui H, Li Y, Liu B, Lou Y (2012) Decentralized control of collaborative redundant manipulators with partial command coverage via locally connected recurrent neural networks. Neural Comput Appl 1–10
16. Li S, Chen S, Liu B, Li Y, Liang Y (2012) Decentralized kinematic control of a class of collaborative redundant manipulators via recurrent neural networks. Neurocomputing 91(0):1–10
17. Tong M, Bickett A, Christiansen E, Cottrell G (2007) Learning grammatical structure with Echo State Networks.. Neural Netw 20(3):424–432
18. Li S, Chen S, Liu B (2012) Accelerating a recurrent neural network to finite-time convergence for solving time-varying sylvester equation by using a sign-bi-power activation function. Neural Process Lett 1–17. doi:10.1007/s11063-012-9241-1
19. Doherty L, pister KSJ, Ghaoui EL (2001) Convex position estimation in wireless sensor networks. In INFOCOM 2001. Twentieth annual joint conference of the IEEE computer and communications societies. Proceedings. IEEE,vol 3:1655–1663
20. Niculescu D, Nath B (2003) Dv based positioning in ad hoc networks. Telecommun Syst 22(1):267–280
21. He T, Huang C, Blum Brian M, Stankovic JA, Abdelzaher T (2003) Range-free localization schemes for large scale sensor networks. In Proceedings of the 9th annual international conference on Mobile computing and networking, MobiCom '03, ACM, New York, pp 81–95
22. Hopfield JJ (1984) Neurons with graded response have collective computational properties like those of two-state neurons. Proc Natl Acad Sci 81(10):3088–3092
23. Kennedy MP, Chua LO (1988) Neural networks for nonlinear programming. Circuits Syst IEEE Trans 35(5):554–562
24. Xia Y, Ye D (2008) On exponential convergence conditions of an extended projection neural network. Neural Comput 20(9):2227–2237
25. Jiang S, Han D, Yuan X (2012) Efficient neural networks for solving variational inequalities. Neurocomputing 86(0):97–106
26. Liu L, Ge R, Gao P (2011) A novel neural network for solving singular nonlinear convex optimization problems. In neural information processing, vol 7063 of Lecture Notes in Computer Science, pp 554–561
27. Li S, Chen S, Lou Y, Lu B, Liang Y (2012) A recurrent neural network for inter-localization of mobile phones. In neural networks (IJCNN), The 2012 International Joint Conference on June 2012, pp 1–5
28. Li S, Lou Y, Liu B Bluetooth aided mobile phone localization: a nonlinear neural circuit approach. ACM Transac Embed Comput Syst. Accepted
29. Li S, Qin F A dynamic neural network approach for solving nonlinear inequalities defined on a graph and its application to distributed, routing-free, range-free localization of WSNS. Neurocomputing. Accepted
30. Li S, Liu B, Chen B, Lou Y (2012) Neural network based mobile phone localization using bluetooth connectivity. Neural Comput Appl, 1–9
31. Zhang Y, Wang J (2002) A dual neural network for convex quadratic programming subject to linear equality and inequality constraints. Phys Lett A 298(4):271–278
32. Xia Y, Feng G, Wang J (2004) A recurrent neural network with exponential convergence for solving convex quadratic program and related linear piecewise equations. Neural Netw 17(7):1003–1015
33. Wu N (1997) The maximum entropy method (Springer series in information sciences). Springer, Berlin
34. Belkin M, Niyogi P (2001) Laplacian eigenmaps and spectral techniques for embedding and clustering. In advances in neural information processing systems 14, MIT Press, Cambridge, pp 585–591
35. Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge University Press, New York
36. Xia Y (2004) An extended projection neural network for constrained optimization. Neural Comput 16:863–883
37. Nedic A, Ozdaglar A (2009) Subgradient methods for saddle-point problems. J Optim Theory Appl 142:205–228
38. Mao G, Fidan B, Anderson BDO (2007) Wireless sensor network localization techniques. Computer Netw 51(10):2529–2553
39. Biswas P, Liang TC, Wang TC, Ye Y (2006) Semidefinite programming based algorithms for sensor network localization. ACM Trans Sens Netw 2:2006
40. Shang Y, Ruml W, Zhang Y, Fromherz MPJ (2003) Localization from mere connectivity. In Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing, MobiHoc '03. ACM, NY, pp 201–212
41. Stankovic JA, Stoleru R, He T (2007) Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks, vol 30, chapter range-free localization. Springer