Neurodynamic Optimization: New Models and kWTA Applications

Jun Wang

jwang@mae.cuhk.edu.hk

Department of Mechanical & Automation Engineering
The Chinese University of Hong Kong
Shatin, New Territories, Hong Kong
http://www.mae.cuhk.edu.hk/ ~jwang

Introduction

Optimization is ubiquitous in nature and society.

Introduction

Optimization is ubiquitous in nature and society.

Optimization arises in a wide variety of scientific problems.

Introduction

Optimization is ubiquitous in nature and society.

Optimization arises in a wide variety of scientific problems.

Optimization is an important tool for design, planning, control, operation, and management of engineering systems.

Problem Formulation

Consider a general optimization problem:

$$\begin{aligned} \mathrm{OP}_1: \ \text{Minimize} \quad & f(x) \\ \text{subject to} \quad & c(x) \leq 0, \\ & d(x) = 0, \end{aligned}$$

where $x \in \Re^n$ is the vector of decision variables, $f(x)$ is an objective function, $c(x) = [c_1(x), \ldots, c_m(x)]^T$ is a vector-valued function, and
$d(x) = [d_1(x), \ldots, d_p(x)]^T$ a vector-valued function.

Problem Formulation

Consider a general optimization problem:

$$\begin{aligned} \mathrm{OP}_1: \ \ \text{Minimize} \quad & f(x) \\ \text{subject to} \quad & c(x) \le 0, \\ & d(x) = 0, \end{aligned}$$

where $x \in \Re^n$ is the vector of decision variables, $f(x)$ is an objective function, $c(x) = [c_1(x), \ldots, c_m(x)]^T$ is a vector-valued function, and
$d(x) = [d_1(x), \ldots, d_p(x)]^T$ a vector-valued function.

If $f(x)$ and $c(x)$ are convex and $d(x)$ is affine, then OP is a convex programming problem CP. Otherwise, it is a nonconvex program.

Quadratic and Linear Programs

$$
\begin{aligned}
\mathrm{QP}_1: \ \text{minimize} \quad & \frac{1}{2}x^T Q x + q^T x \\
\text{subject to} \quad & Ax = b, \\
& l \le Cx \le h,
\end{aligned}
$$

where $Q \in \Re^{n\times n}$, $q \in \Re^n, A \in \Re^{m\times n},$
$b \in \Re^m, C \in \Re^{n\times n}, l \in \Re^n, h \in \Re^n.$

Quadratic and Linear Programs

$$
\begin{aligned}
\mathrm{QP}_1: \ \text{minimize} \quad & \frac{1}{2}x^T Q x + q^T x \\
\text{subject to} \quad & Ax = b, \\
& l \le Cx \le h,
\end{aligned}
$$

where $Q \in \Re^{n\times n}$, $q \in \Re^n$, $A \in \Re^{m\times n}$, $b \in \Re^m, C \in \Re^{n\times n}, l \in \Re^n, h \in \Re^n$.
When $Q = 0$, and $C = I$, QP_1 becomes a linear program with equality and bound constraints:

$$
\begin{aligned}
\mathrm{LP}_1: \ \text{minimize} \quad & q^T x \\
\text{subject to} \quad & Ax = b, \\
& l \le x \le h
\end{aligned}
$$

Dynamic Optimization

In many applications (e.g., online pattern recognition and onboard signal processing), real-time solutions to optimization problems are necessary or desirable.

Dynamic Optimization

In many applications (e.g., online pattern recognition and onboard signal processing), real-time solutions to optimization problems are necessary or desirable.

For such applications, classical optimization techniques may not be competent due to the problem dimensionality and stringent requirement on computational time.

Dynamic Optimization

In many applications (e.g., online pattern recognition and onboard signal processing), real-time solutions to optimization problems are necessary or desirable.

For such applications, classical optimization techniques may not be competent due to the problem dimensionality and stringent requirement on computational time.

It is computationally challenging when optimization procedures have to be performed in real time to optimize the performance of dynamical systems.

Dynamic Optimization

In many applications (e.g., online pattern recognition and onboard signal processing), real-time solutions to optimization problems are necessary or desirable.

For such applications, classical optimization techniques may not be competent due to the problem dimensionality and stringent requirement on computational time.

It is computationally challenging when optimization procedures have to be performed in real time to optimize the performance of dynamical systems.

One very promising approach to dynamic optimization is to apply artificial neural networks.

Neurodynamic Optimization

Because of the inherent nature of parallel and distributed information processing in neural networks, the convergence rate of the solution process is not decreasing as the size of the problem increases.

Neurodynamic Optimization

Because of the inherent nature of parallel and distributed information processing in neural networks, the convergence rate of the solution process is not decreasing as the size of the problem increases.

Neural networks can be implemented physically in designated hardware such as ASICs where optimization is carried out in a truly parallel and distributed manner.

Neurodynamic Optimization

Because of the inherent nature of parallel and distributed information processing in neural networks, the convergence rate of the solution process is not decreasing as the size of the problem increases.

Neural networks can be implemented physically in designated hardware such as ASICs where optimization is carried out in a truly parallel and distributed manner.

This feature is particularly desirable for dynamic optimization in decentralized decision-making situations.

Existing Approaches

In their seminal work, Tank and Hopfield (1985, 1986) applied the Hopfield networks for solving a linear program and the traveling salesman problem.

Existing Approaches

In their seminal work, Tank and Hopfield (1985, 1986) applied the Hopfield networks for solving a linear program and the traveling salesman problem.

Kennedy and Chua (1988) developed a neural network for nonlinear programming, which contains finite penalty parameters and thus its equilibrium points correspond to approximate optimal solutions only.

Existing Approaches

In their seminal work, Tank and Hopfield (1985, 1986) applied the Hopfield networks for solving a linear program and the traveling salesman problem.

Kennedy and Chua (1988) developed a neural network for nonlinear programming, which contains finite penalty parameters and thus its equilibrium points correspond to approximate optimal solutions only.

The two-phase optimization networks by Maa and Shanblatt (1992).

Existing Approaches

In their seminal work, Tank and Hopfield (1985, 1986) applied the Hopfield networks for solving a linear program and the traveling salesman problem.

Kennedy and Chua (1988) developed a neural network for nonlinear programming, which contains finite penalty parameters and thus its equilibrium points correspond to approximate optimal solutions only.

The two-phase optimization networks by Maa and Shanblatt (1992).

The Lagrangian networks for quadratic programming by Zhang and Constantinides (1992) and Zhang, et al. (1992).

Existing Approaches (cont'd)

A recurrent neural network for quadratic optimization with bounded variables only by Bouzerdoum and Pattison (1993).

Existing Approaches (cont'd)

A recurrent neural network for quadratic optimization with bounded variables only by Bouzerdoum and Pattison (1993).

The deterministic annealing network for linear and convex programming by Wang (1993, 1994).

Existing Approaches (cont’d)

A recurrent neural network for quadratic optimization with bounded variables only by Bouzerdoum and Pattison (1993).

The deterministic annealing network for linear and convex programming by Wang (1993, 1994).

The primal-dual networks for linear and quadratic programming by Xia (1996, 1997).

Existing Approaches (cont’d)

A recurrent neural network for quadratic optimization with bounded variables only by Bouzerdoum and Pattison (1993).

The deterministic annealing network for linear and convex programming by Wang (1993, 1994).

The primal-dual networks for linear and quadratic programming by Xia (1996, 1997).

The projection networks for solving projection equations, constrained optimization, etc by Xia and Wang (1998, 2002, 2004) and Liang and Wang (2000).

Existing Approaches (cont'd)

The dual networks for quadratic programming by Xia and Wang (2001), Zhang and Wang (2002).

Existing Approaches (cont'd)

The dual networks for quadratic programming by Xia and Wang (2001), Zhang and Wang (2002).

A two-layer network for convex programming subject to nonlinear inequality constraints by Xia and Wang (2004).

Existing Approaches (cont'd)

The dual networks for quadratic programming by Xia and Wang (2001), Zhang and Wang (2002).

A two-layer network for convex programming subject to nonlinear inequality constraints by Xia and Wang (2004).

A simplified dual network for quadratic programming by Liu and Wang (2006)

Existing Approaches (cont'd)

The dual networks for quadratic programming by Xia and Wang (2001), Zhang and Wang (2002).

A two-layer network for convex programming subject to nonlinear inequality constraints by Xia and Wang (2004).

A simplified dual network for quadratic programming by Liu and Wang (2006)

Two one-layer networks with discontinuous activation functions for linear and quadratic programming by Liu and Wang (2008).

Primal-Dual Network

The primal-dual network for solving LP_2[a]:

$$\epsilon\frac{dx}{dt} = -(q^T x - b^T y)q - A^T(Ax - b) + x^+,$$
$$\epsilon\frac{dy}{dt} = (q^T x - b^T y)b,$$

where $\epsilon > 0$ is a scaling parameter, $x \in \Re^n$ is the primal state vector, $y \in \Re^m$ is the dual (hidden) state vector, $x^+ = (x_1^+), ..., x_n^+)^T$, and $x_i^+ = \max\{0, x_i\}$.

[a] Y. Xia, "A new neural network for solving linear and quadratic programming problems," *IEEE Transactions on Neural Networks*, vol. 7, no. 6, 1544-1548, 1996.

Primal-Dual Network

The primal-dual network for solving LP_2[a]:

$$\epsilon\frac{dx}{dt} = -(q^T x - b^T y)q - A^T(Ax - b) + x^+,$$
$$\epsilon\frac{dy}{dt} = (q^T x - b^T y)b,$$

where $\epsilon > 0$ is a scaling parameter, $x \in \Re^n$ is the primal state vector, $y \in \Re^m$ is the dual (hidden) state vector, $x^+ = (x_1^+), ..., x_n^+)^T$, and $x_i^+ = \max\{0, x_i\}$. The network is globally convergent to an optimal solution to LP_1.

[a] Y. Xia, "A new neural network for solving linear and quadratic programming problems," *IEEE Transactions on Neural Networks*, vol. 7, no. 6, 1544-1548, 1996.

Lagrangian Network for QP

If $C = 0$ in QP_1:

$$\epsilon \frac{d}{dt}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -Qx(t) - A^T y(t) - q, \\ Ax - b \end{pmatrix}.$$

where $\epsilon > 0$, $x \in \Re^n, y \in \Re^m$.

It is globally exponentially convergent to the optimal solution[a].

---

[a] J. Wang, Q. Hu, and D. Jiang, “A Lagrangian network for kinematic control of redundant robot manipulators,” *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1123-1132, 1999.

Projection Network

A recurrent neural network called the projection network was developed for optimization with bound constraints only[a]

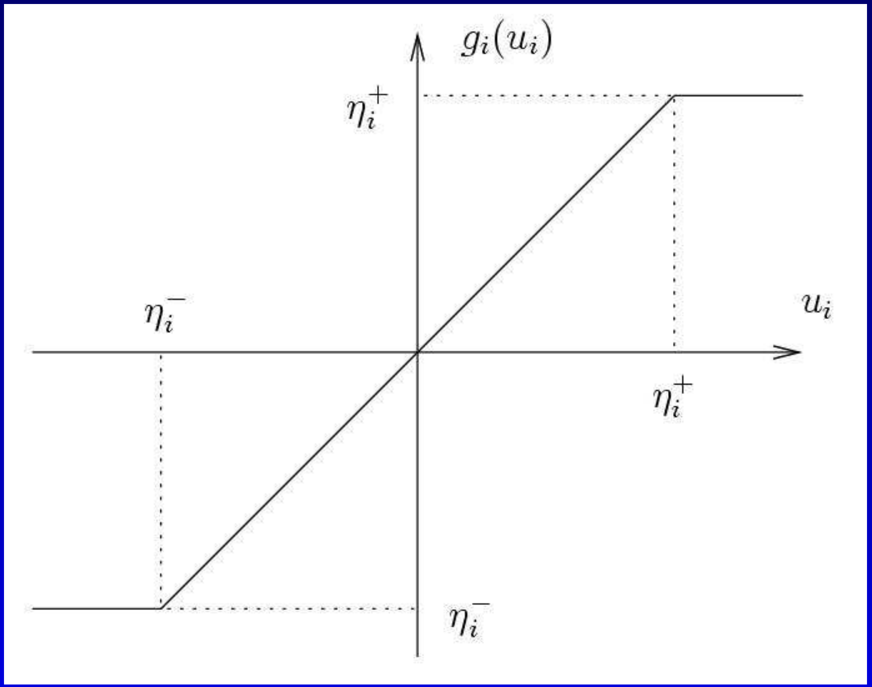
$$\epsilon \frac{dx}{dt} = -x + g(x - \nabla f(x)),$$

where $g(\cdot)$ is a vector-valued piecewise-linear activation function.

[a] Y.S. Xia and J. Wang, "On the stability of globally projected dynamic systems," *J. of Optimization Theory and Applications*, vol. 106, no. 1, pp. 129-150, 2000.

Piecewise-Linear Activation Function

$$g(x_i) = \begin{cases} l_i & x_i < l_i \\ x_i & l_i \le x_i \le h_i \\ h_i & x_i > h_i. \end{cases}$$



Two-layer Projection Network for QP$_1$

If $C = I$ in QP$_1$, let $\alpha = 1$ in the two-layer neural network for CP:

$$\epsilon \frac{d}{dt} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -x + g((I - Q)x + A^T y - q) \\ -Ax + b \end{pmatrix}.$$

where $\epsilon > 0$, $x \in \Re^n$, $y \in \Re^m$,
$g(x) = [g(x_1), ..., g(x_n)]^T$ is the piecewise-linear activation function defined before.

It is globally asymptotically convergent to the optimal solution[a].

[a] Y.S. Xia, H. Leung, and J. Wang, "A projection neural network and its application to constrained optimization problems," *IEEE Trans. Circuits and Systems I*, vol. 49, no. 4, pp. 447-458, 2002.

General projection Network for QP$_1$

The dynamic equation of the general projection neural network (GPNN):

$$\epsilon \frac{dz}{dt} = (M+N)^T\{-Nz + g((N-M)z)\},$$

where $\epsilon > 0$ and $z = (x^T, y^T)^T$ is the state vector,

$$M = \begin{pmatrix} Q & -A^T \\ 0 & I \end{pmatrix}, N = \begin{pmatrix} I & 0 \\ A & 0 \end{pmatrix}.$$

The GPNN is globally convergent to an exact solution of the problem[a].

---

[a] Y. Xia and J. Wang, "A general projection neural network for solving optimization and related problems," *IEEE Trans. Neural Networks*, vol. 15, pp. 318-328, 2004.

Dual Network for QP$_2$

For strictly convex QP$_2$, Q is invertible. The dynamic equation of the dual network:

$$
\begin{aligned}
\epsilon \frac{dy(t)}{dt} &= -CQ^{-1}C^T y + g\left(CQ^{-1}C^T y - y - Cq\right) \\
&\quad + Cq + b, \\
x(t) &= Q^{-1}C^T y - q,
\end{aligned}
$$

where $\epsilon > 0$. It is also globally exponentially convergent to the optimal solution[a] [b].

---

[a] Y. Xia and J. Wang, "A dual neural network for kinematic control of redundant robot manipulators," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 31, no. 1, pp. 147-154, 2001.

[b] Y. Zhang and J. Wang, "A dual neural network for convex quadratic programming subject to linear equality and inequality constraints," *Physics Letters A*, pp. 271-278, 2002.

Simplified Dual Net for QP$_1$

For strictly convex QP$_1$, Q is invertible. The dynamic equation of the simplified dual network [a]:

$$\epsilon\frac{du}{dt} = -Cx + g(Cx - u),$$
$$x = Q^{-1}(A^T y + C^T u - q),$$
$$y = (AQ^{-1}A^T)^{-1}\left[-AQ^{-1}C^T u + AQ^{-1}q + b\right],$$

where $u \in \mathbb{R}^n$ is the state vector, $\epsilon > 0$.
It is proven to be globally asymptotically convergent to the optimal solution.

[a] S. Liu and J. Wang, "A simplified dual neural network for quadratic programming with its KWTA application," *IEEE Trans. Neural Networks*, vol. 17, no. 6, pp. 1500-1510, 2006.

Illustrative Example

$$\begin{aligned}
\text{minimize} \quad & 3x_1^2 + 3x_2^2 + 4x_3^2 + 5x_4^2 + 3x_1x_2 + 5x_1x_3 + \\
& x_2x_4 - 11x_1 - 5x_4 \\
\text{subject to} \quad & 3x_1 - 3x_2 - 2x_3 + x_4 = 0, \\
& 4x_1 + x_2 - x_3 - 2x_4 = 0, \\
& -x_1 + x_2 \leq -1, \\
& -2 \leq 3x_1 + x_3 \leq 4.
\end{aligned}$$

$$Q = \begin{bmatrix} 6 & 3 & 5 & 0 \\ 3 & 6 & 0 & 1 \\ 5 & 0 & 8 & 0 \\ 0 & 1 & 0 & 10 \end{bmatrix}, \; q = \begin{bmatrix} -11 \\ 0 \\ 0 \\ -5 \end{bmatrix},$$

$$A = \begin{bmatrix} 3 & -3 & -2 & 1 \\ 4 & 1 & -1 & -2 \end{bmatrix}, \; b = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

$$C = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 3 & 0 & 1 & 0 \end{bmatrix}, \; l = \begin{bmatrix} -\infty \\ -2 \end{bmatrix}, \; h = \begin{bmatrix} -1 \\ 4 \end{bmatrix}.$$

Illustrative Example (cont'd)

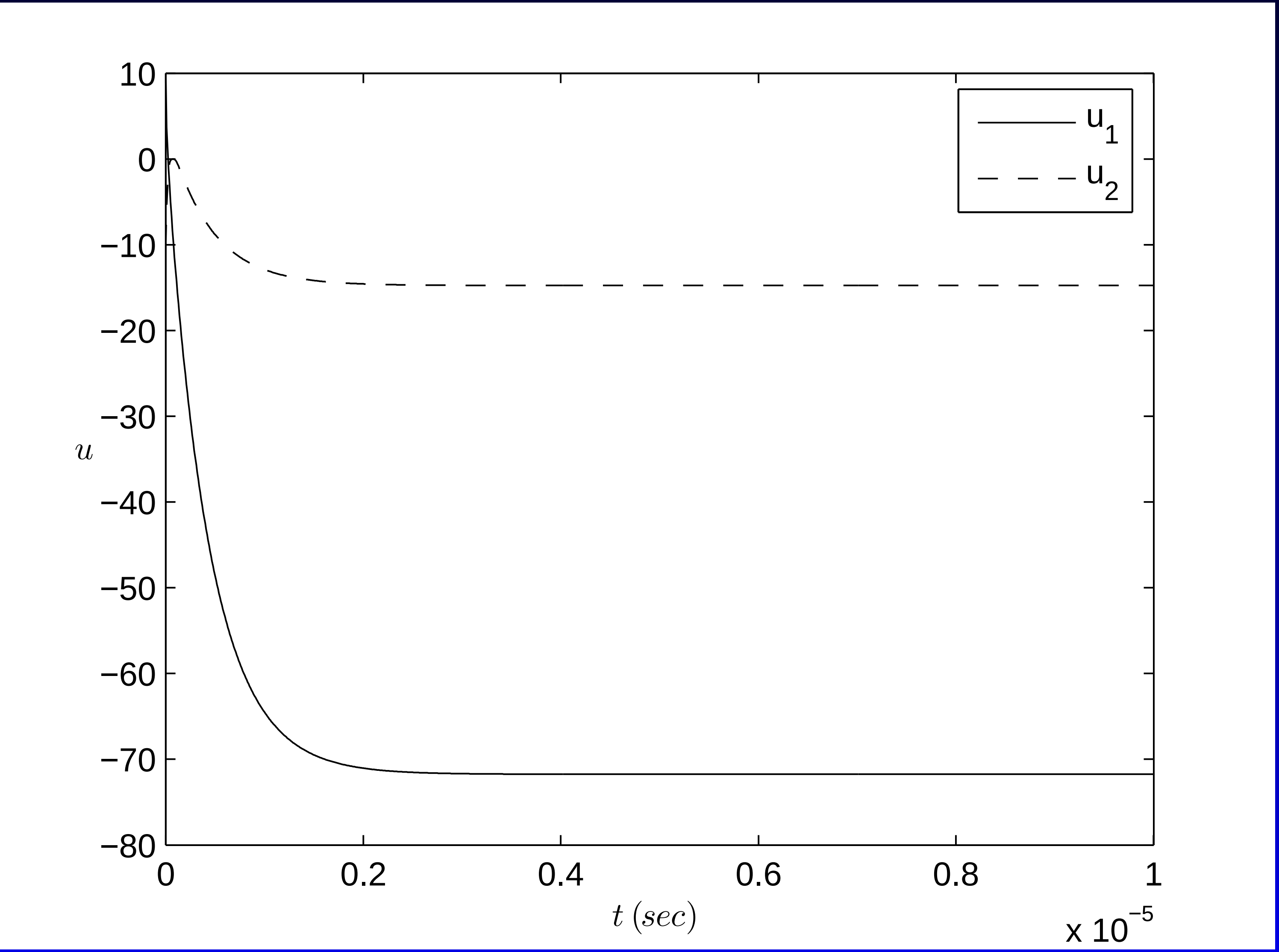
The simplified dual neural network for solving this quadratic program needs only two neurons only.

In contrast, the Lagrange neural network needs twelve neurons.

The primal-dual neural network needs nine neurons.

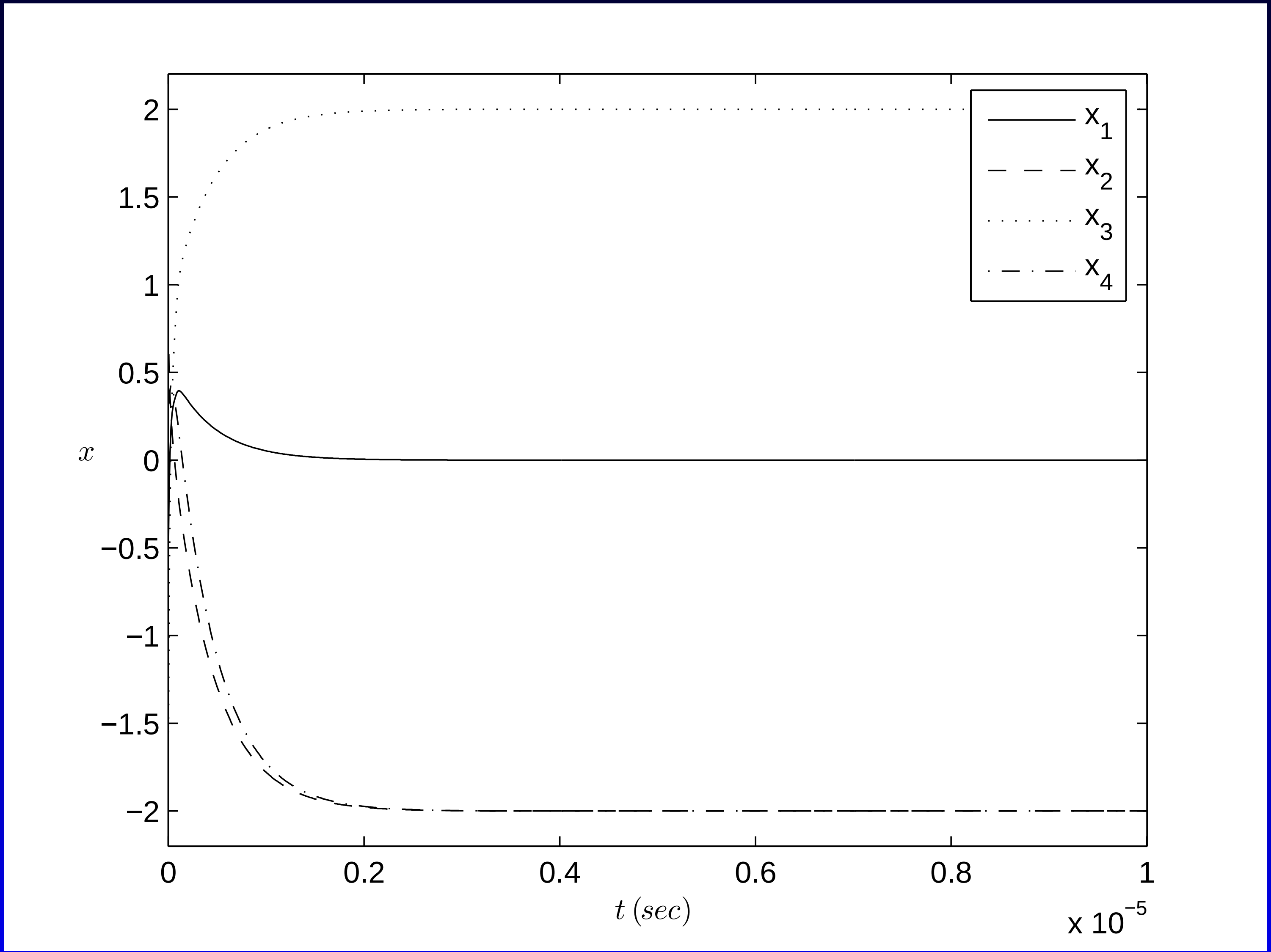
The dual neural network needs four neurons.

Illustrative Example (cont'd)



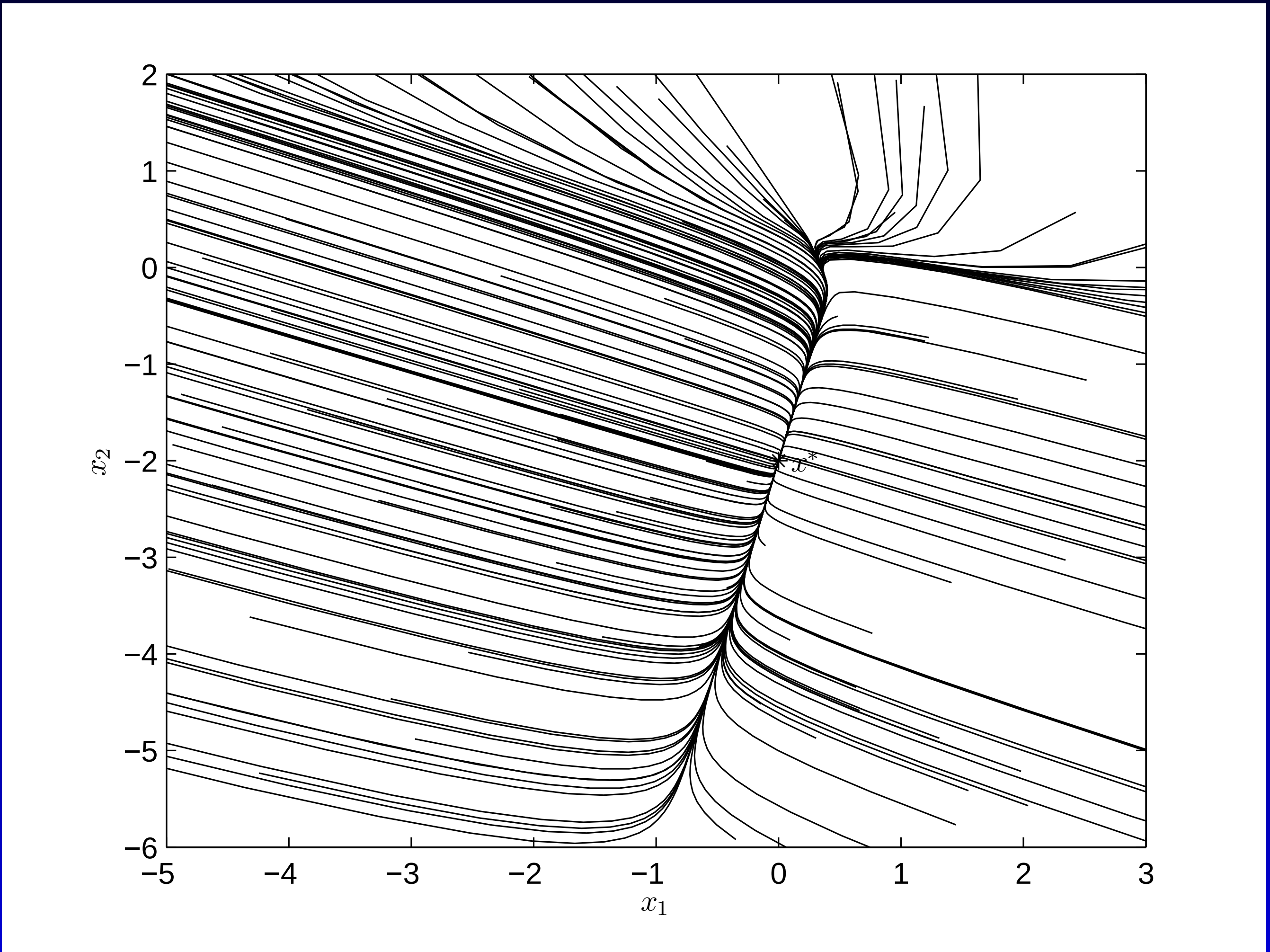
Transient behaviors of the state vector u.

Illustrative Example (cont'd)



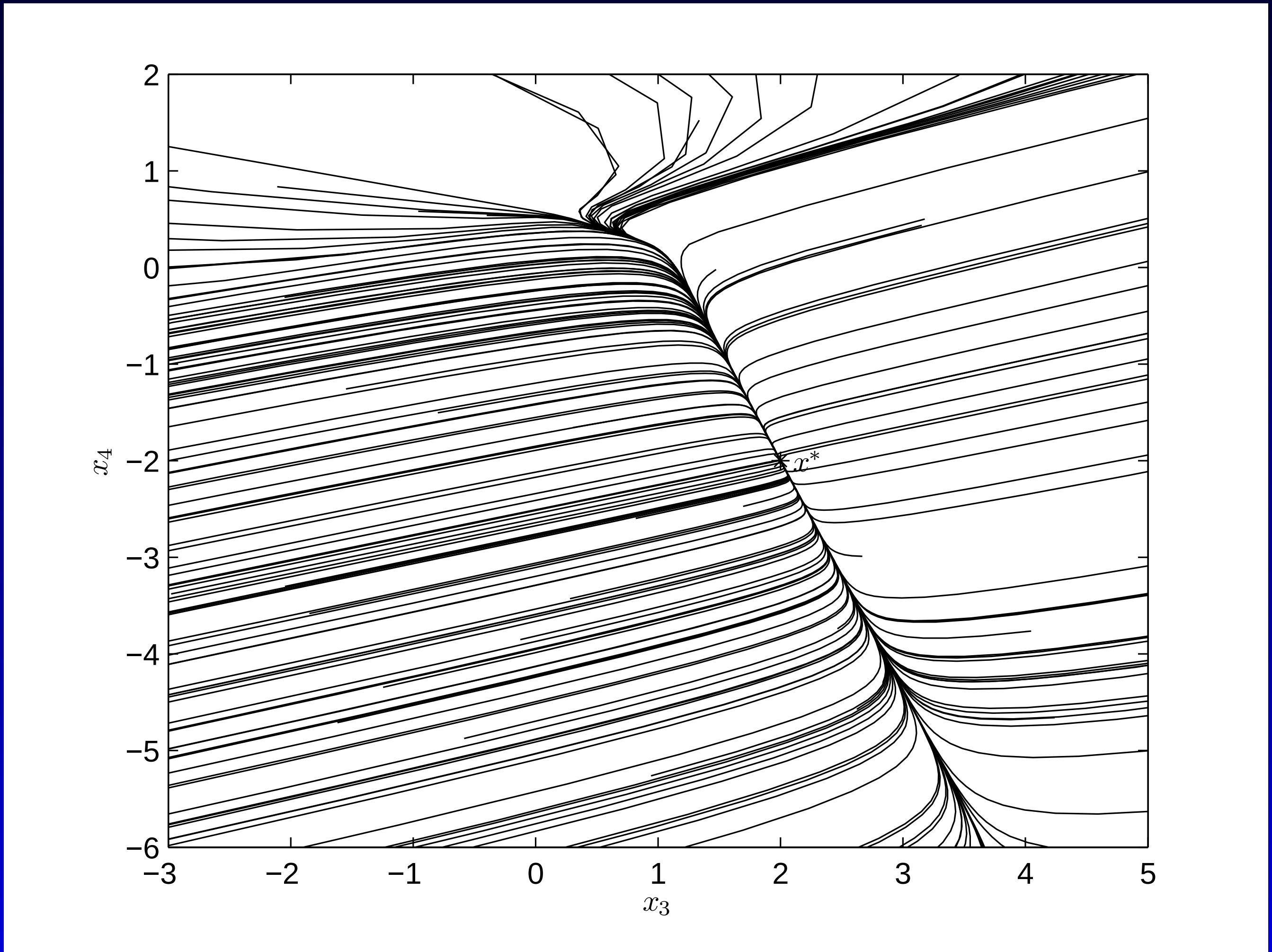
Transient behaviors of the output vector x.

Illustrative Example (cont'd)



Trajectories of x_1 and x_2 from different initial states.

Illustrative Example (cont'd)



Trajectories of x_3 and x_4 from different initial states.

Improved Dual Net for special QP$_1$

For special convex QP$_1$ when $Q = I$, the dynamic equation of the improved dual network[a]:

$$\begin{aligned}
\epsilon \frac{dy}{dt} &= -y + (y + Ax - b)^+, \\
\epsilon \frac{dz}{dt} &= -Cx + d, \\
x &= g_\Omega(-A^T y + C^T z - p),
\end{aligned}$$

where $g(\cdot)$ and $(\cdot)^+$ are two activation functions. It is proven to be globally convergent to the optimal solution.

[a] X. Hu and J. Wang, "An improved dual neural network for solving a class of quadratic programming problems and its k winners-take-all application," *IEEE Trans. Neural Networks*, vol. 19, no. 12, pp. in press, 2008.

A One-layer Net for LP

A new recurrent neural network model with a discontinuous activation function was recently developed for linear programming LP_1[a]:

$$\epsilon\frac{dx}{dt} = -Px - \sigma(I-P)g(x) + s,$$

where $g(x) = (g_1(x_1), g_2(x_2), \ldots, g_n(x_n))^T$ is the vector-valued activation function, ϵ is a positive scaling constant, σ is a nonnegative gain parameter, $P = A^T(AA^T)^{-1}A$, and $s = -(I-P)q + A^T(AA^T)^{-1}b$.

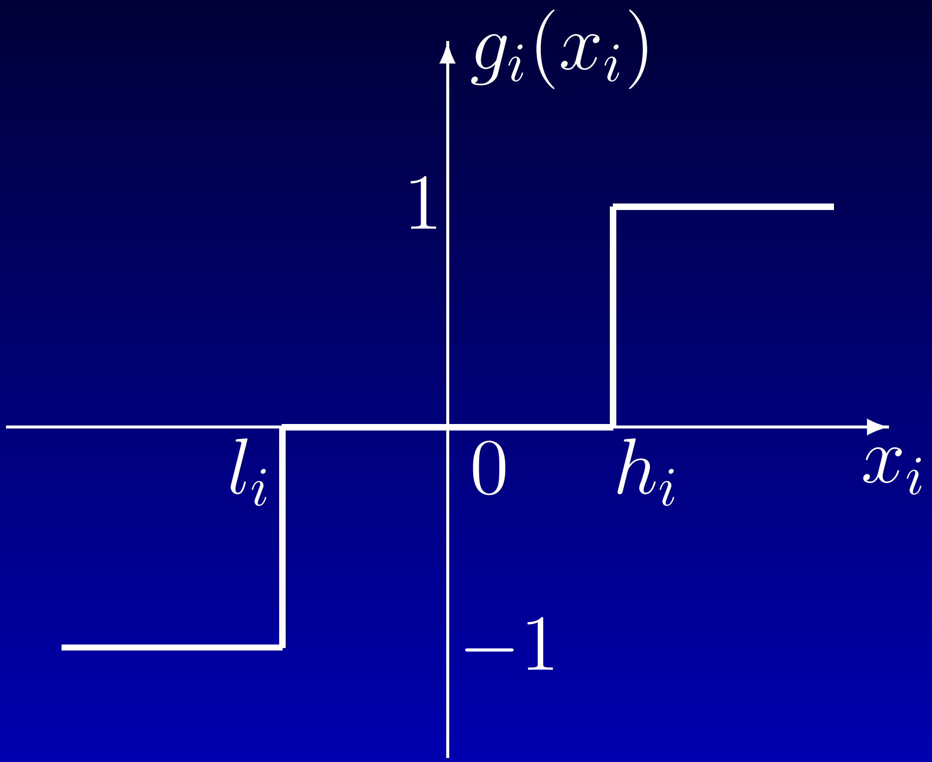
[a] Q. Liu, and J. Wang, "A one-layer recurrent neural network with a discontinuous activation function for linear programming," *Neural Computation*, vol. 20, no. 5, pp. 1366-1383, 2008.

Activation Function

A discontinuous activation function is defined as follows: For $i = 1, 2, \ldots, n$;

$$g_i(x_i) = \begin{cases} 1, & \text{if } x_i > h_i, \\ [0, 1], & \text{if } x_i = h_i, \\ 0, & \text{if } x_i \in (l_i, h_i), \\ [-1, 0], & \text{if } x_i = l_i, \\ -1, & \text{if } x_i < l_i. \end{cases}$$

Activation Function (cont'd)



Convergence Results

The neural network is globally convergent to an optimal solution of LP_1 with $C = I$, if $\bar{\Omega} \subset \Omega$, where $\bar{\Omega}$ is the equilibrium point set and $\Omega = \{x | l \leq x \leq h\}$.

Convergence Results

- The neural network is globally convergent to an optimal solution of LP_1 with $C = I$, if $\bar{\Omega} \subset \Omega$, where $\bar{\Omega}$ is the equilibrium point set and $\Omega = \{x | l \le x \le h\}$.

- The neural network is globally convergent to an optimal solution of LP_1 with $C = I$, if it has a unique equilibrium point and $\sigma \ge 0$ when $(I - P)c = 0$ or one of the following conditions holds when $(I - P)c \ne 0$:

  (i) $\sigma \ge \|(I - P)c\|_p / \min^{+}_{\gamma \in X} \|(I - P)\gamma\|_p$ for $p = 1, 2, \infty$, or

  (ii) $\sigma \ge c^T(I - P)c / \min^{+}_{\gamma \in X} \{|c^T(I - P)\gamma|\}$,

  where $X = \{-1, 0, 1\}^n$

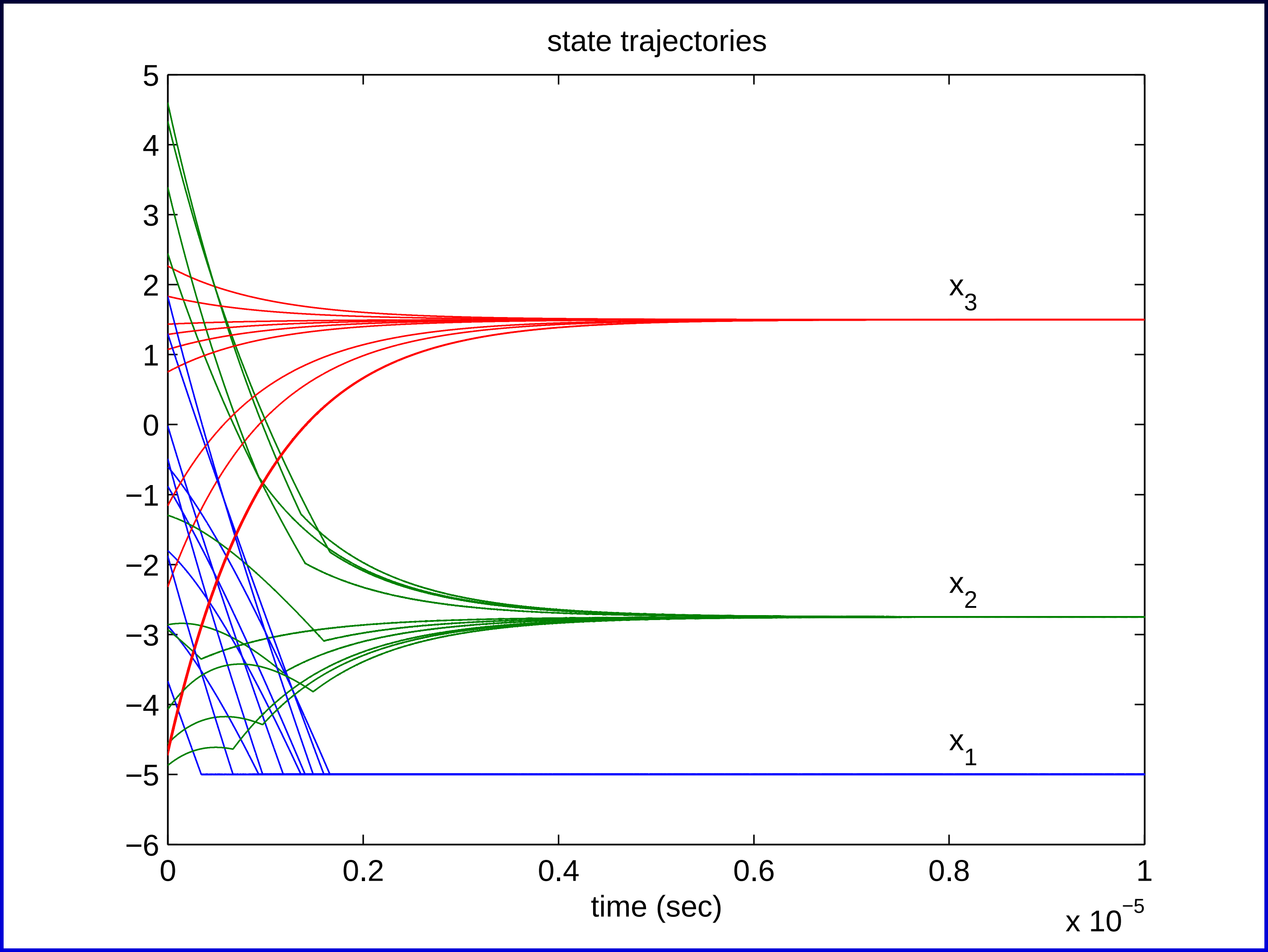
Simulation Results

Consider the following LP problem:

$$\begin{array}{ll} \text{minimize} & 4x_1 + x_2 + 2x_3, \\ \text{subject to} & x_1 - 2x_2 + x_3 = 2, \\ & -x_1 + 2x_2 + x_3 = 1, \\ & -5 \leq x_1, x_2, x_3 \leq 5. \end{array}$$

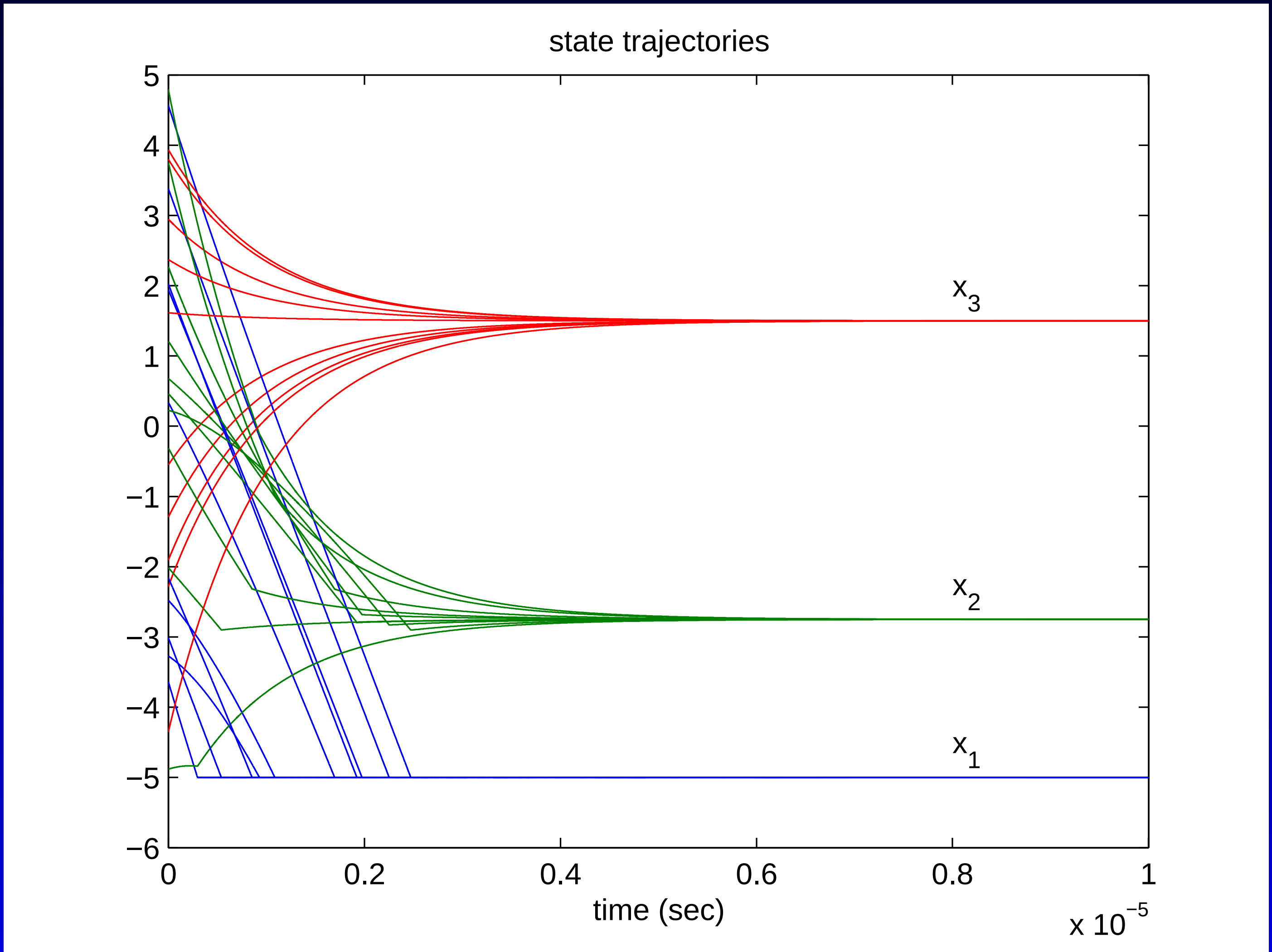
According to the above condition, the lower bound of σ is 9

Simulation Results (cont’d)



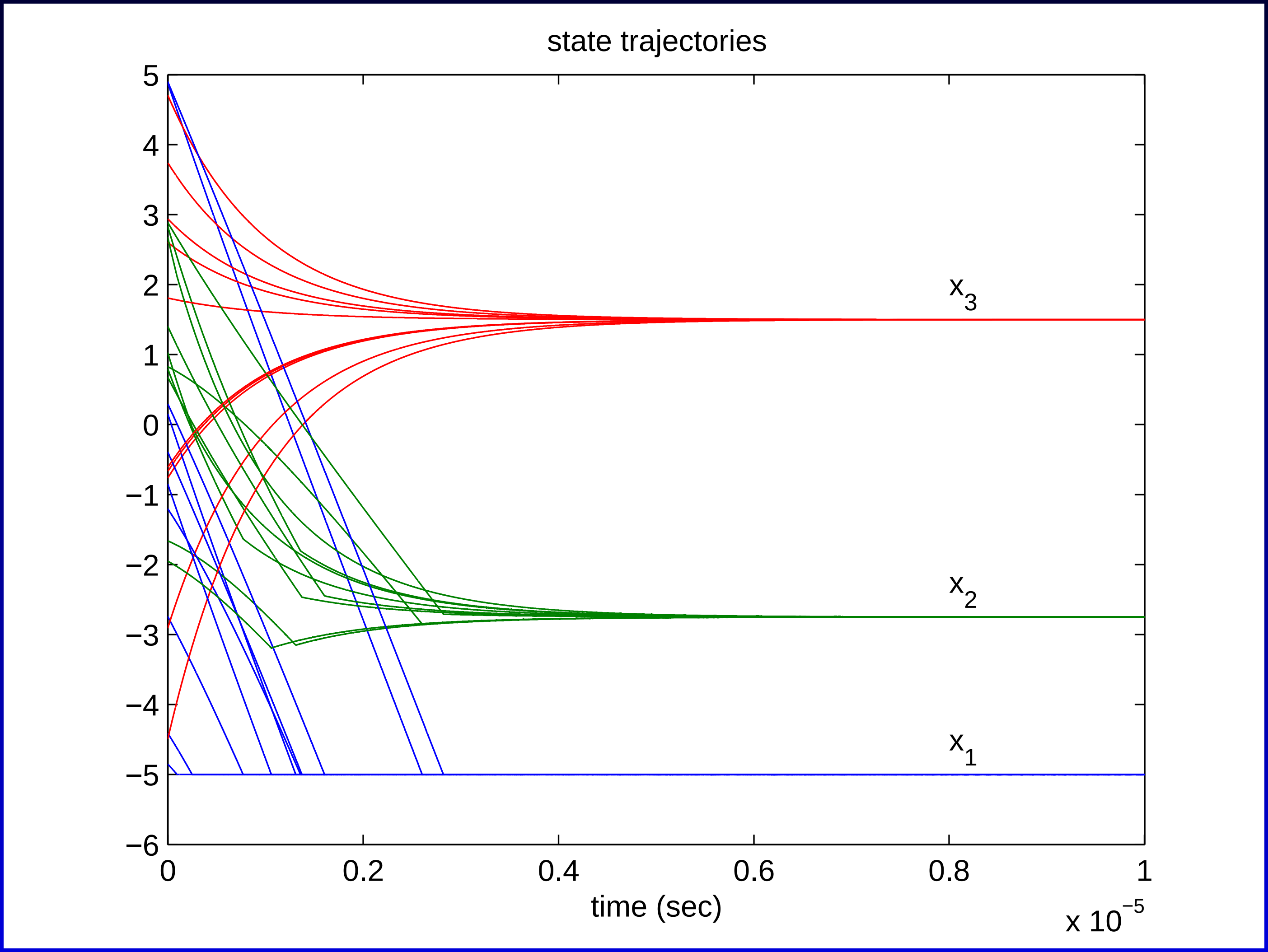
Transient behaviors of the states with $\sigma = 15$.

Simulation Results (cont’d)



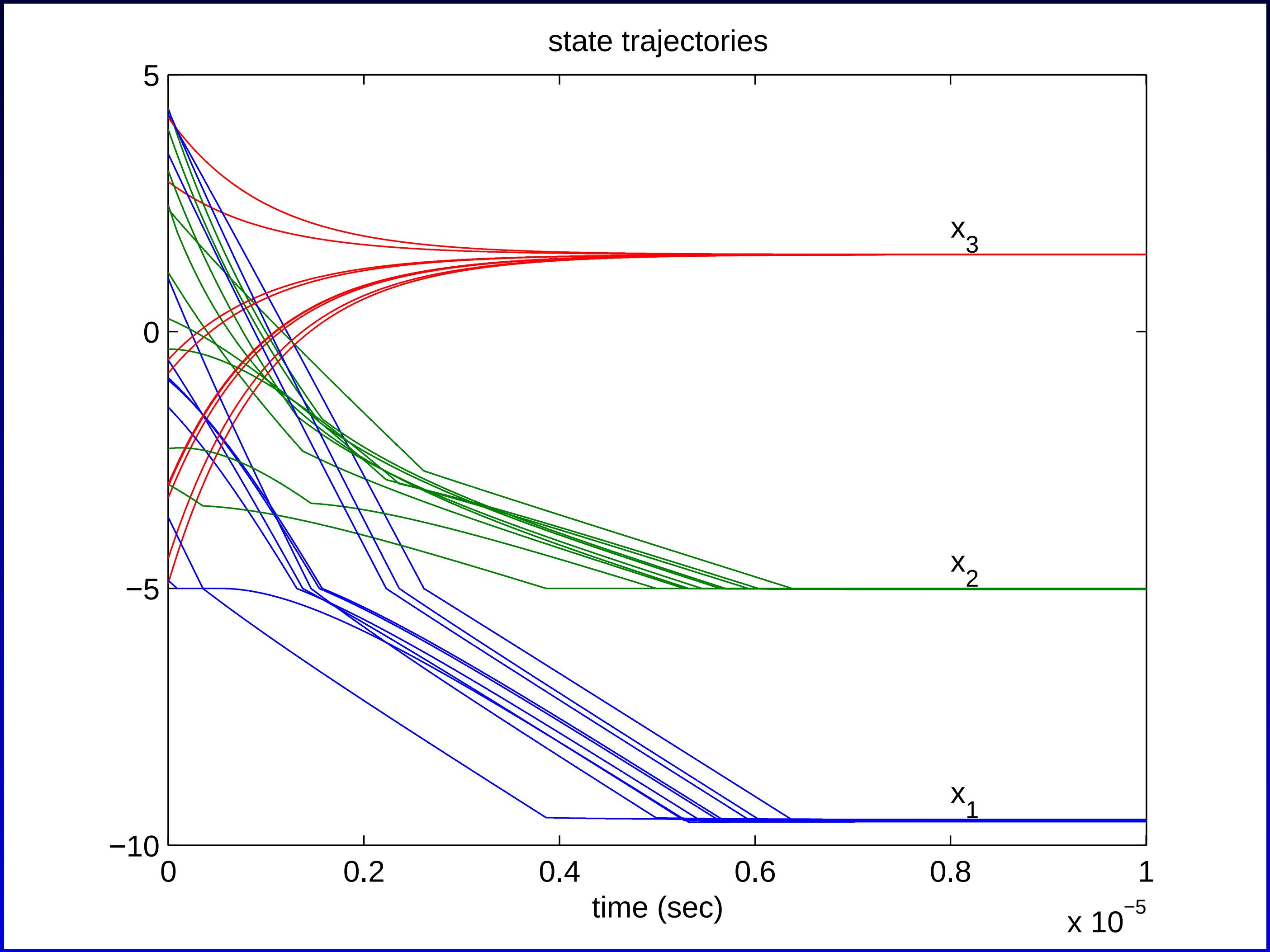
Transient behaviors of the states with $\sigma = 9$.

Simulation Results (cont'd)



Transient behaviors of the states with $\sigma = 5$.

Simulation Results (cont’d)



Transient behaviors of the states with $\sigma = 3$.

A New One-layer Net for QP

A new one-layer recurrent neural net was recently developed[a]:

$$
\begin{aligned}
\epsilon \frac{dz}{dt} &= -(I-P)z - [(I-P)Q + \alpha P]g(z) + q, \\
x &= ((I-P)Q + \alpha P)^{-1}(-(I-P)z + s),
\end{aligned}
$$

where ϵ is a positive scaling constant, $\alpha > 0$ is a parameter, $s = -q + Pq + \alpha A^T(AA^T)^{-1}b$, and $g(\cdot)$ is a vector-valued activation function.

[a] 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.

Activation Function

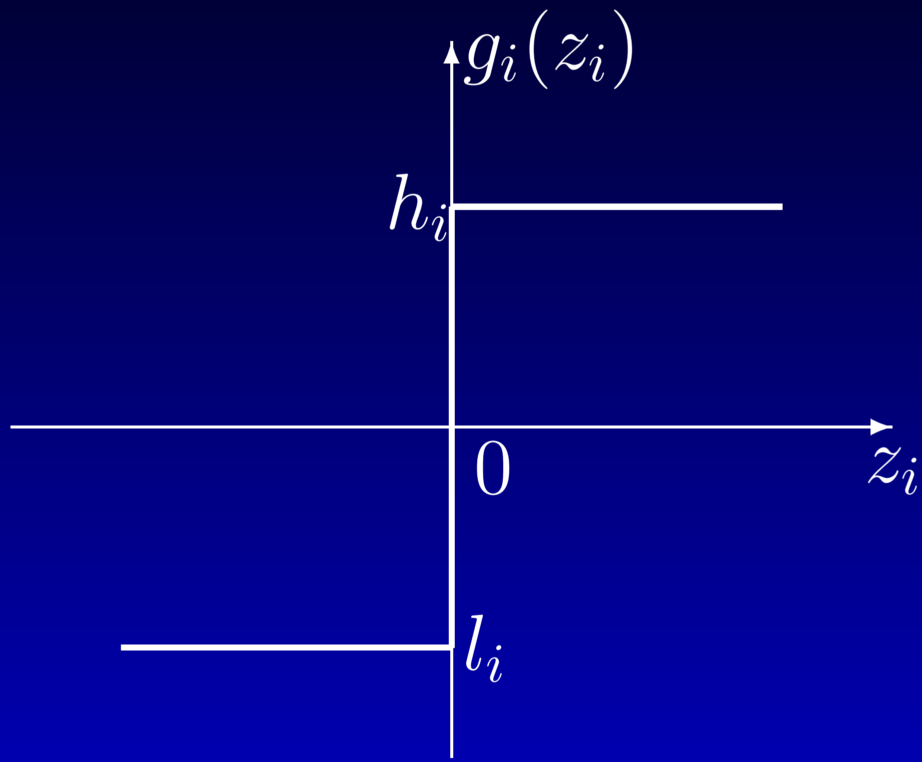
The following hard-limiting activation function is defined:

$$g_i(z_i) \begin{cases} = h_i, & \text{if } z_i > 0, \\ \in [l_i, h_i], & \text{if } z_i = 0, \\ = l_i, & \text{if } z_i < 0. \end{cases}$$

If $l_i \neq h_i$, then g_i is discontinuous.

When $z_i = 0$, $g_i(z_i)$ can take any values between l_i and h_i.

Activation Function (cont'd)



Convergence results

Assume that Q is positive definite. If $\alpha \geq \lambda_{\max}(Q)/2$ or $\alpha \geq \mathrm{trace}(Q)/2$, then the state vector $z(t)$ of the neural network is globally convergent to an equilibrium point and the output vector $x(t)$ is globally convergent to an optimal solution of QP.

Assume that the objective function $f(x)$ is strictly convex on the set $\mathcal{S} = \{x \in \mathbb{R}^n : Ax = b\}$. If

$$\alpha > \lambda_{\max}(Q^2)\lambda_{\max}(Q^{-1})/4,$$

then the state vector $z(t)$ of the neural network is globally convergent to an equilibrium point and the output vector $x(t)$ is globally convergent to an optimal solution of QP.

Illustrative Example

Consider the following QP problem:

$$
\begin{array}{ll}
\text{minimize} & f(x) = -0.5x_1^2 + x_2^2 + 2x_1x_2 + 6x_1 - \\
\text{subject to} & 3x_1 - 2x_2 = 1, \\
 & 0 \le x_1, x_2 \le 10.
\end{array}
$$

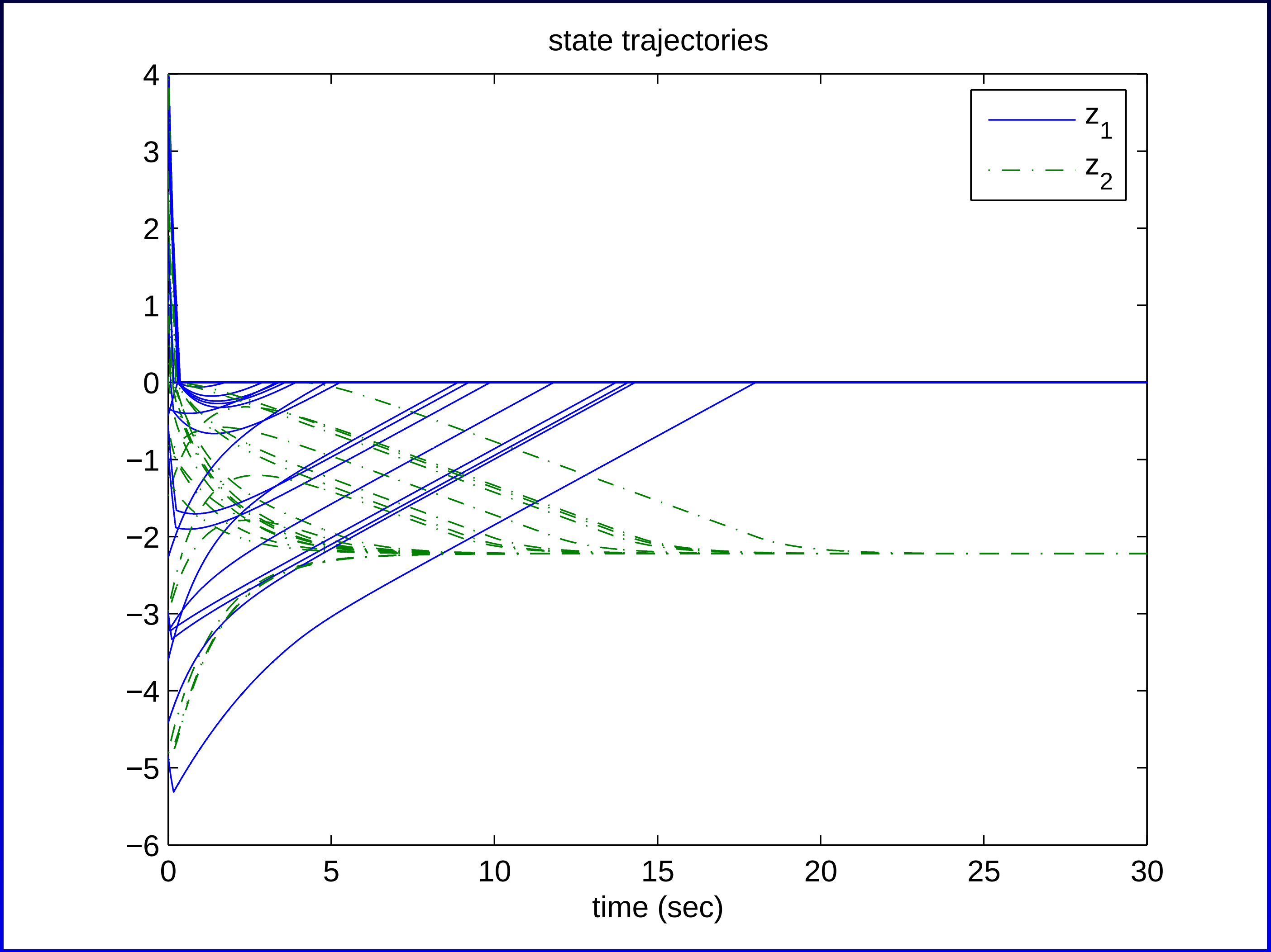
As

$$
Q = \begin{pmatrix} -0.5 & 1 \\ 1 & 1 \end{pmatrix}
$$

is not positive definite, the objective function is not convex everywhere. However, if we substitute $x_1 = 2x_2/3 + 1/3$ into the objective function, then $\tilde{f}(x_2) = 19x_2^2/9 + 22x_2/9 - 35/18$ is convex.

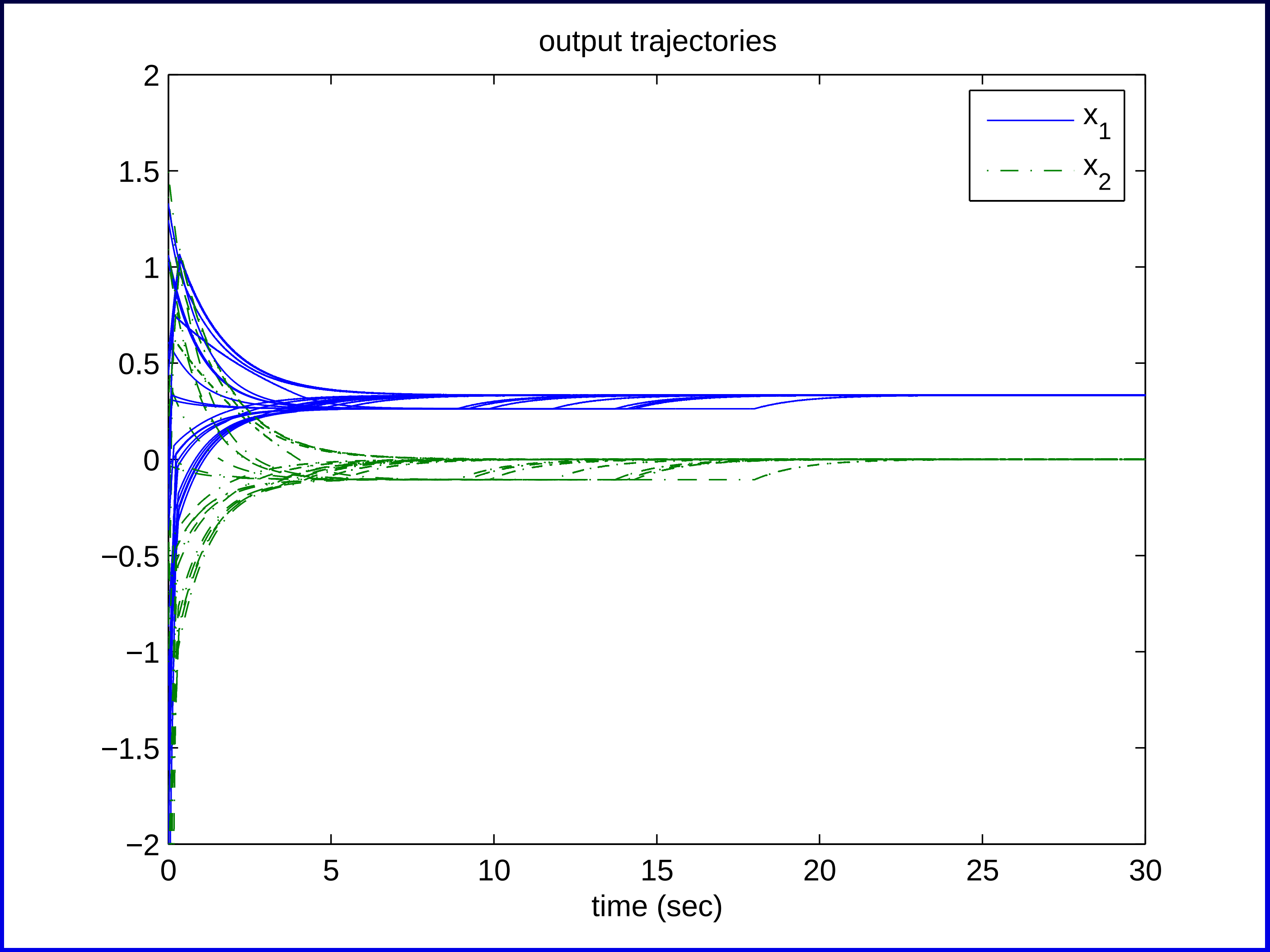
Illustrative Example (cont’d)

The state variables of the new network.



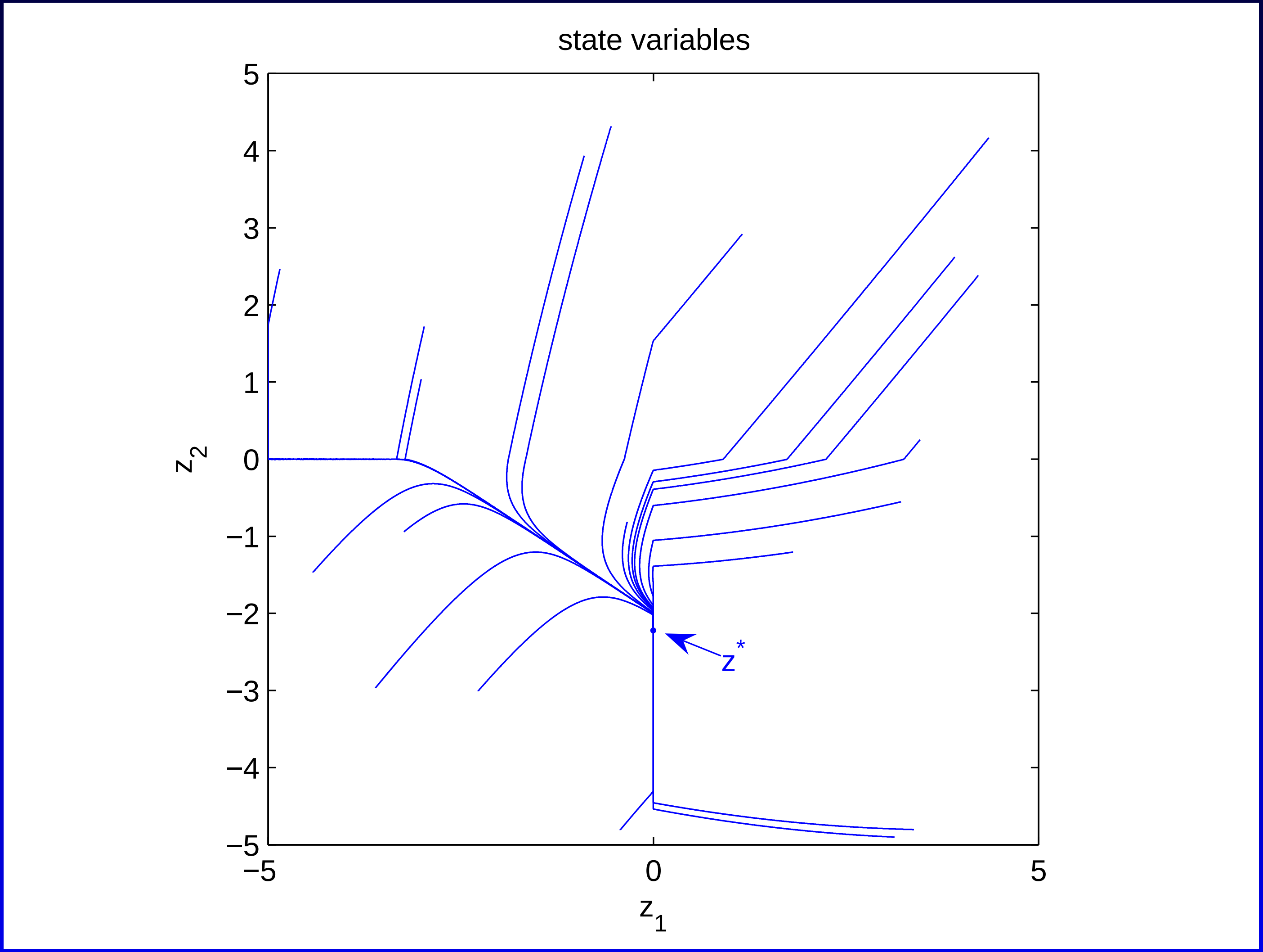
Illustrative Example (cont'd)

The output variables of the new network.



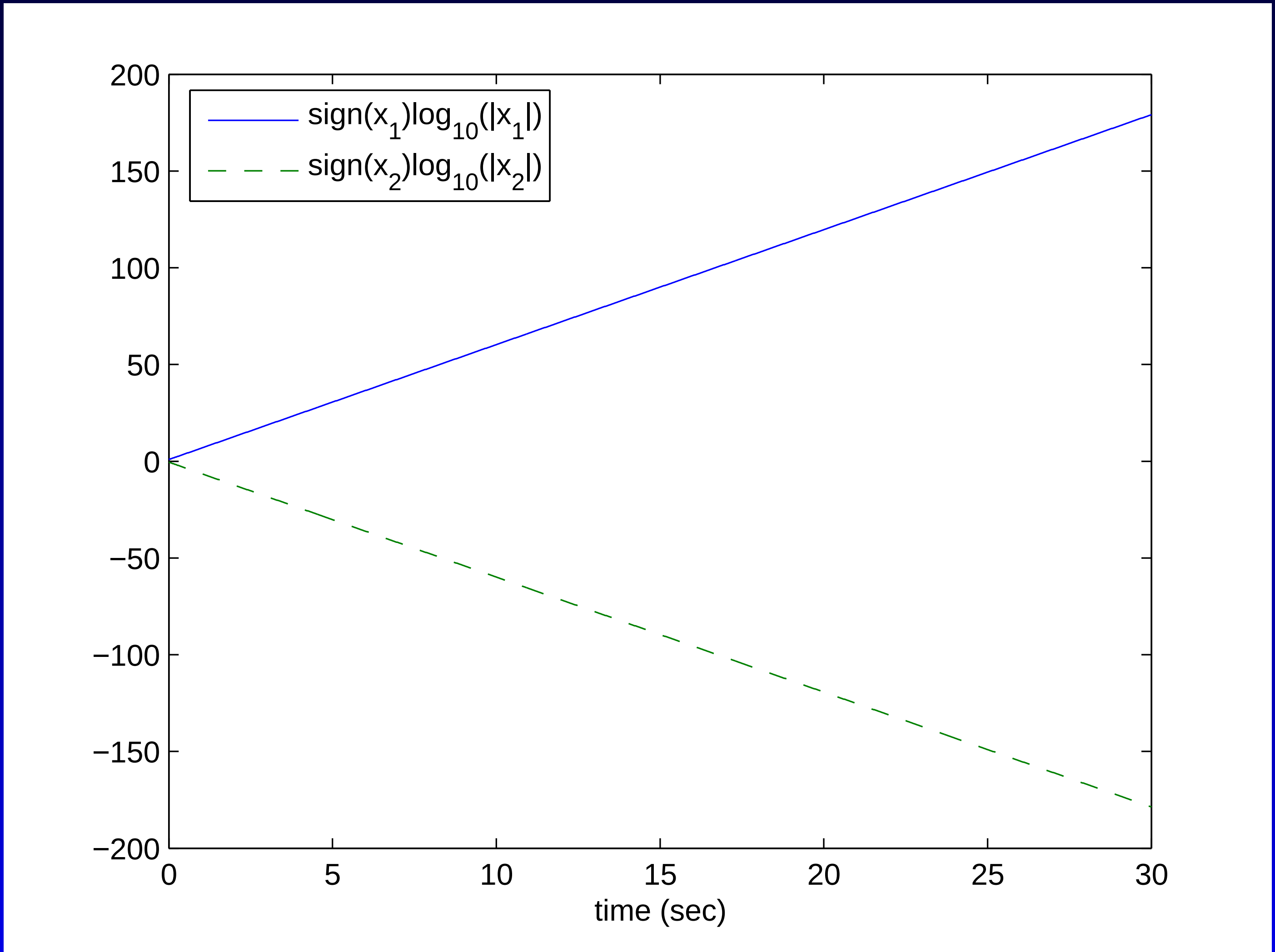
Illustrative Example (cont'd)

Phase plot of the output variabnles.



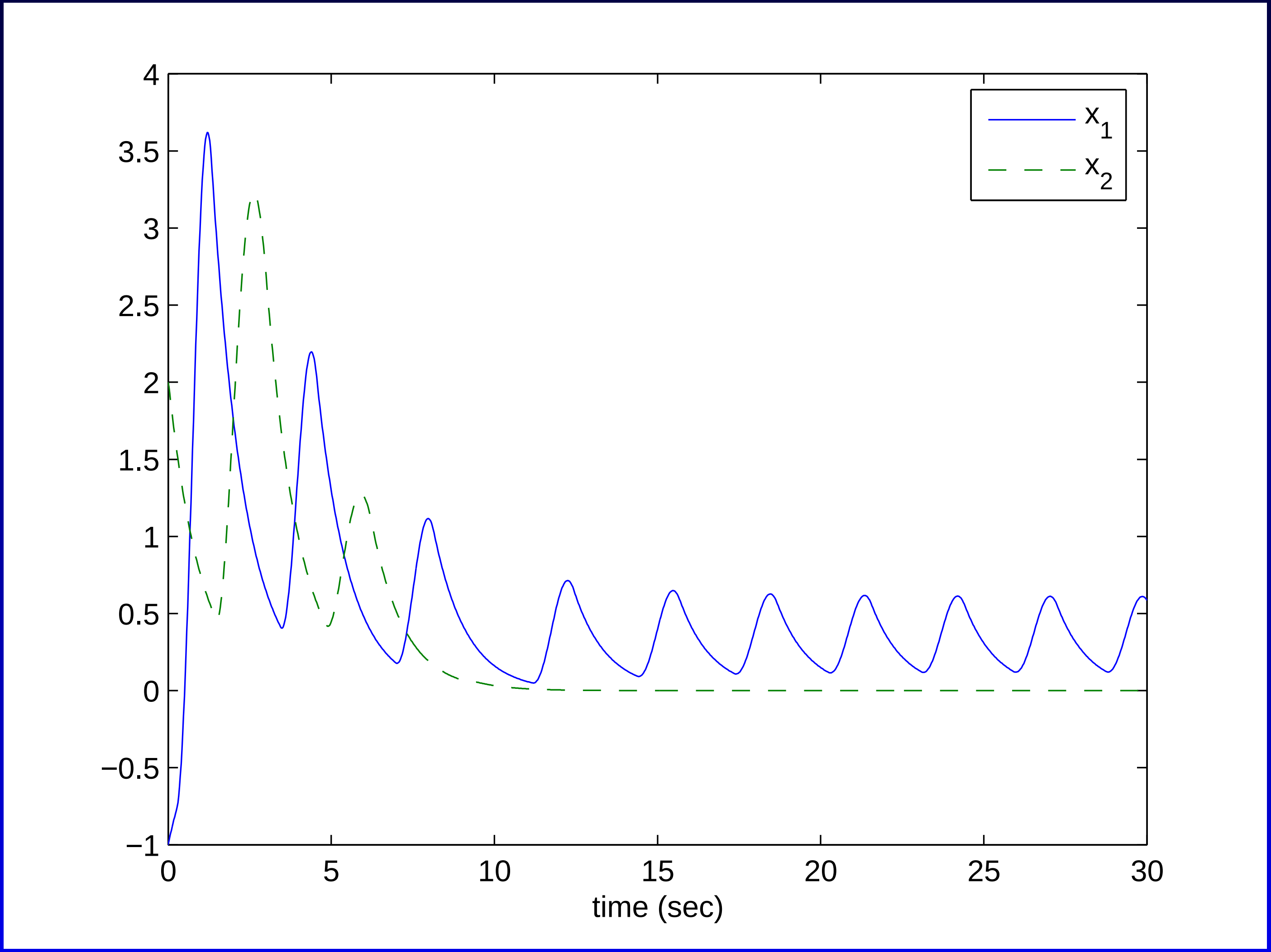
Illustrative Example (cont'd)

The simulation result of the dual network.



Illustrative Example (cont'd)

The simulation result of the projection network.



Model Comparisons for QP_1

model	layers	neurons	connections	convergence condition
Lagrangian network	2	$3n+m$	n^2+2mn	$f(x)$ is strictly convex
Primal-dual network	2	$n+m$	$3n^2+3mn$	$f(x)$ is convex
General projection net	2	$n+m$	n^2+2mn	$f(x)$ is convex
Dual network	1	$n+m$	$(n+m)^2$	$f(x)$ is strictly convex
Simplified dual network	1	n	n^2	$f(x)$ is strictly convex
New neural network	1	n	$2n^2$	$f(x)$ is strictly convex on $\mathcal{S}$

where $\mathcal{S} = \{x \in \mathbb{R}^n : Ax = b\}$.

k Winners Take All Operation

The k-winners-take-all (kWTA) operation is to select the k largest inputs out of n inputs ($1 \leq k \leq n$).

k Winners Take All Operation

The k-winners-take-all (kWTA) operation is to select the k largest inputs out of n inputs ($1 \leq k \leq n$).

The kWTA operation has important applications in machine learning, such as k-neighborhood classification, k-means clustering, etc.

k Winners Take All Operation

The k-winners-take-all (kWTA) operation is to select the k largest inputs out of n inputs ($1 \leq k \leq n$).

The kWTA operation has important applications in machine learning, such as k-neighborhood classification, k-means clustering, etc.

As the number of inputs increases and/or the selection process should be operated in real time, parallel algorithms and hardware implementation are desirable.

kWTA Problem Formulations

The kWTA function can be defined as:

$$x_i = f(u_i) = \begin{cases} 1, & \text{if } u_i \in \{k \text{ largest elements of } u\}, \\ 0, & \text{otherwise,} \end{cases}$$

where $u \in \mathbb{R}^n$ and $x \in \mathbb{R}^n$ is the input vector and output vector, respectively.

kWTA Problem Formulations

The kWTA function can be defined as:

$$x_i = f(u_i) = \begin{cases} 1, & \text{if } u_i \in \{k \text{ largest elements of } u\}, \\ 0, & \text{otherwise,} \end{cases}$$

where $u \in \mathbb{R}^n$ and $x \in \mathbb{R}^n$ is the input vector and output vector, respectively.
The kWTA solution can be determined by solving the following linear integer program:

$$\begin{aligned} \text{minimize} \quad & -\sum_{i=1}^{n} u_i x_i, \\ \text{subject to} \quad & \sum_{i=1}^{n} x_i = k, \\ & x_i \in \{0, 1\}, \quad i = 1, 2, \ldots, n. \end{aligned}$$

kWTA Problem Formulations

If the kth and $(k+1)$th largest elements of u are different (denoted as $\bar{u}_k$ and $\bar{u}_{k+1}$ respectively), the kWTA problem is equivalent to the following LP or QP problems:

$$\begin{array}{ll} \text{minimize} & -u^T x \ \text{ or } \ \frac{a}{2}x^T x - u^T x, \\ \text{subject to} & \sum_{i=1}^{n} x_i = k, \\ & 0 \leq x_i \leq 1, \quad i = 1, 2, \ldots, n, \end{array}$$

where $a \leq \bar{u}_k - \bar{u}_{k+1}$ is a positive constant.

QP-based Primal-Dual Network

The primal-dual network based on the QP formulation needs $3n + 1$ neurons and $6n + 2$ connections, and its dynamic equations can be written as:

$$\left\{ \begin{array}{rcl} \epsilon \frac{dx}{dt} & = & -(1+a)(x - (x + ve + w - ax + u)^+) \\ & & -(e^T x - k)e - x - y + e \\ \epsilon \frac{dy}{dt} & = & -y + (y + w)^+ - x - y + e \\ \epsilon \frac{dv}{dt} & = & -e^T (x - (x + ve + w - ax + u)^+) \\ & & +e^T x - k \\ \epsilon \frac{dw}{dt} & = & -x + (x + ve + w - ax + u)^+ \\ & & -y + (y + w)^+ + x + y - e \end{array} \right.$$

where $x, y, w \in \mathbb{R}^n$, $v \in \mathbb{R}$, $e = (1, 1, \ldots, 1)^T \in \mathbb{R}^n$, $\epsilon > 0$, $x^+ = (x_1^+, \ldots, x_n^+)^T$, and $x_i^+ = \max\{0, x_i\}$

QP-based Projection Network

The projection neural network for kWTA operation based on the QP formulation needs $n+1$ neurons and $2n+2$ connections, which dynamic equations can be written as:

$$\left\{\begin{array}{lcl} \epsilon\frac{dx}{dt} & = & -x+g(x-\eta(ax-ye-u)), \\ \epsilon\frac{dy}{dt} & = & -e^T x+k. \end{array}\right.$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}$, ϵ and η are positive constants, $g(x) = (g(x_1), \ldots, g(x_n))^T$ and

$$g(x_i) = \begin{cases} 0, & \text{if } x_i < 0, \\ x_i, & \text{if } 0 \le x_i \le 1, \\ 1, & \text{if } x_i > 1. \end{cases}$$

LP-based Projection Network

Based on the equivalent LP formulation, we propose a recurrent neural network for KWTA operation with its dynamical equations as follows:

$$\begin{cases} \epsilon \frac{dx}{dt} = -x + g(x + \alpha e y + \alpha u), \\ \epsilon \frac{dy}{dt} = e^T x - k, \end{cases}$$

where $\epsilon > 0$, $\alpha > 0$, $x \in R^n$, $y \in R$.

QP-based Simplified Dual Net

The simplified dual neural network for kWTA operation based on the QP formulation [a] needs n neurons and $3n$ connections, and its dynamic equation can be written as:

$$\begin{cases} \epsilon\frac{dy}{dt} & = & -My + g((M-I)y - s) - s \\ x & = & My + s, \end{cases}$$

where $x, y \in \mathbb{R}^n$, $M = 2(I - ee^T/n)/a$, $s = Mu + ke/n$, I is an identity matrix, ϵ and g are defined as before.

[a] S. Liu and J. Wang, "A simplified dual neural network for quadratic programming with its KWTA application," *IEEE Trans. Neural Networks*, vol. 17, no. 6, pp. 1500-1510, 2006.

LP-based One-layer kWTA Net

The dynamic equation of a new LP-based kWTA network model is described as follows:

$$\epsilon \frac{dx}{dt} = -Px - \sigma(I - P)g(x) + s,$$

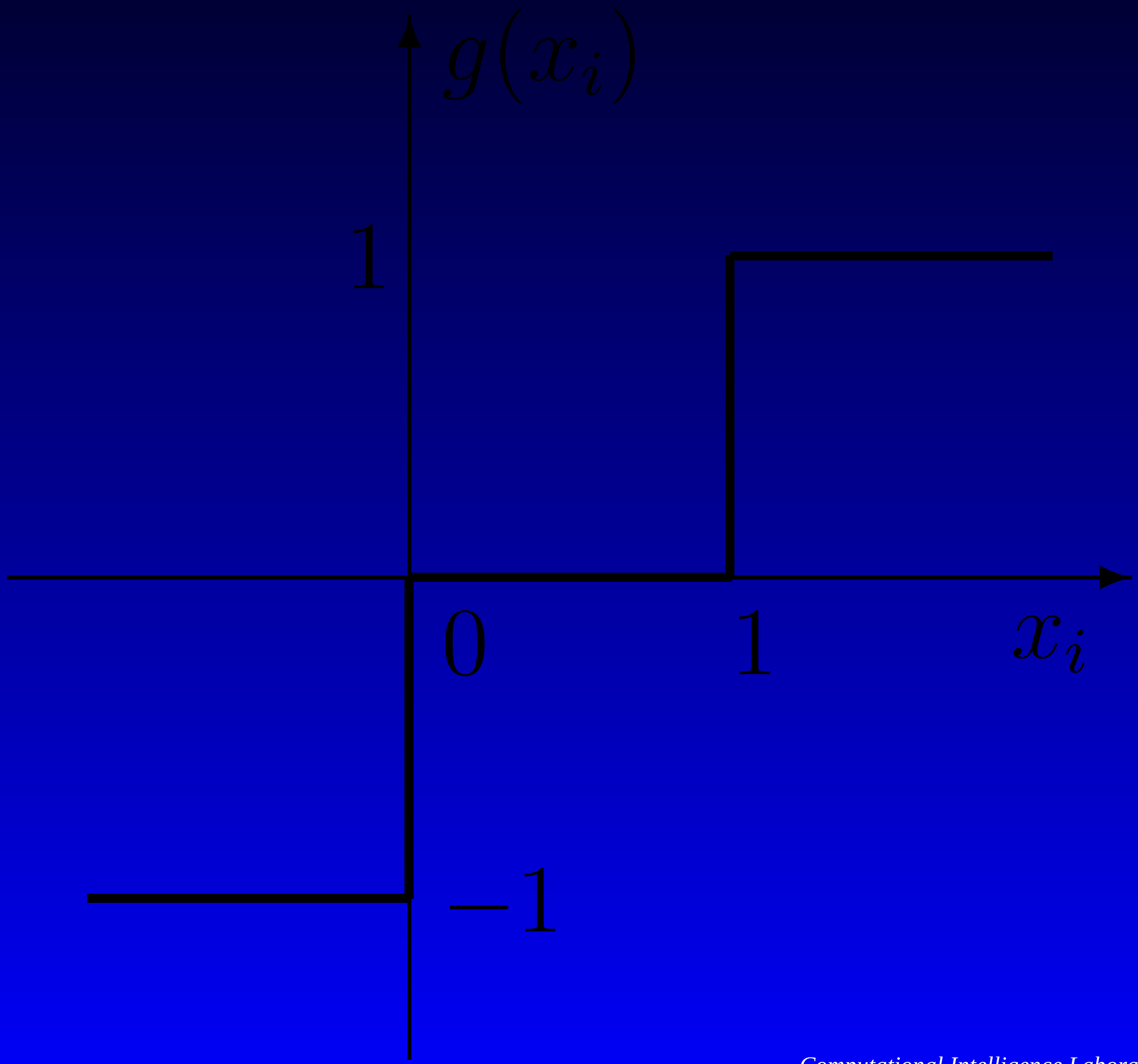
where $P = ee^T/n, s = u - Pu + ke/n$, ϵ is a positive scaling constant, σ is a nonnegative gain parameter, and $g(x) = (g(x_1), g(x_2), \ldots, g(x_n))^T$ is a discontinuous vector-valued activation function.

Activation Function

A discontinuous activation function is defined as follows:

$$g(x_i) = \begin{cases} 1, & \text{if } x_i > 1, \\ [0, 1], & \text{if } x_i = 1, \\ 0, & \text{if } 0 < x_i < 1, \\ [-1, 0], & \text{if } x_i = 0, \\ -1, & \text{if } x_i < 0. \end{cases}$$

Activation Function (cont'd)



Convergence Results

The network can perform the kWTA operation if $\bar{\Omega} \subset \{x \in \mathbb{R}^n : 0 \le x \le 1\}$, where $\bar{\Omega}$ is the set of equilibrium point(s).

The network can perform the kWTA operation if it has a unique equilibrium point and $\sigma \ge 0$ when $(I - ee^T/n)u = 0$ or one of the following conditions holds when $(I - ee^T/n)u \neq 0$:

(i) $\sigma \ge \frac{\sum_{i=1}^n |u_i - \sum_{j=1}^n u_j/n|}{2n-2}$, or

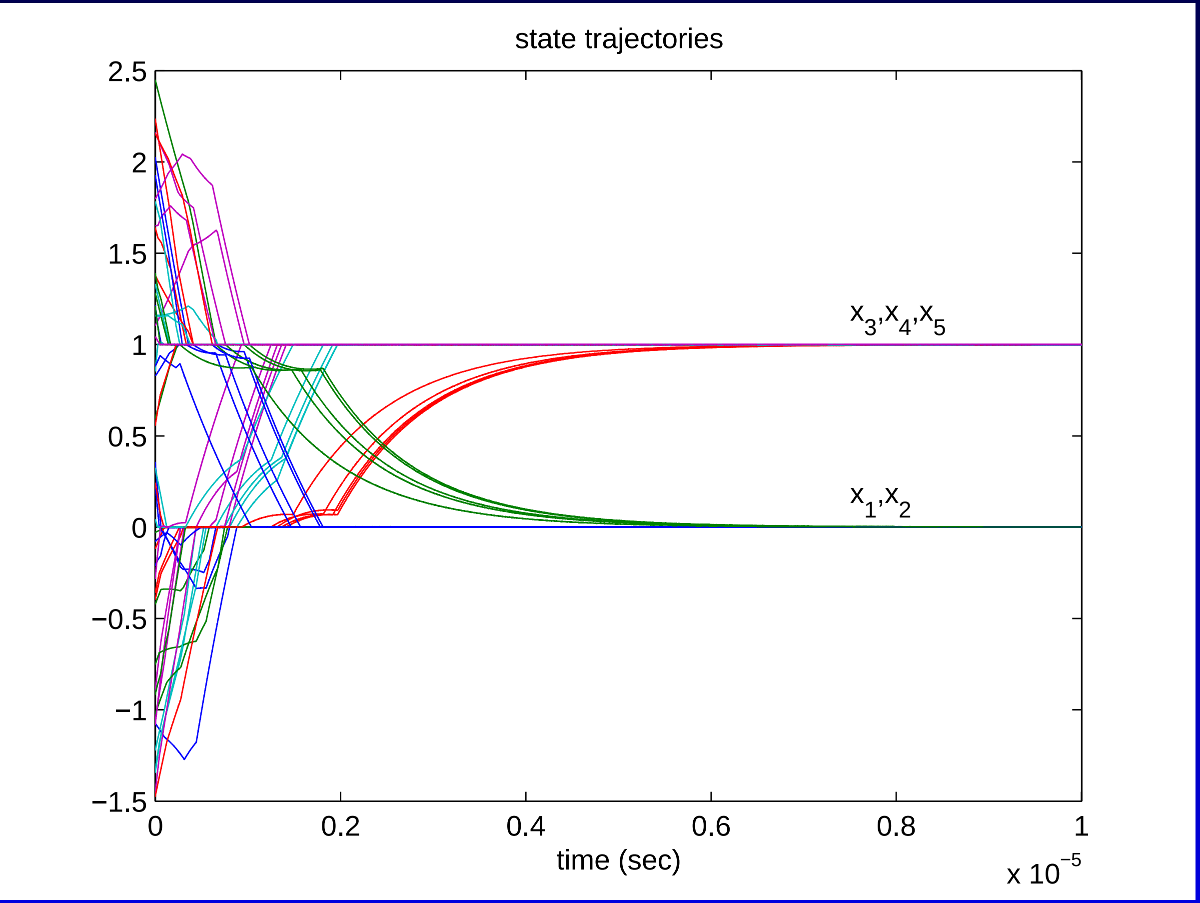
(ii) $\sigma \ge n\sqrt{\frac{\sum_{i=1}^n (u_i - \sum_{j=1}^n u_j/n)^2}{n(n-1)}}$, or

(iii) $\sigma \ge 2\max_i |u_i - \sum_{j=1}^n u_j/n|$, or,

(iv) $\sigma \ge \frac{\sqrt{\sum_{i=1}^n (u_i - \sum_{j=1}^n u_j/n)^2}}{\min^{+}_{\gamma_i \in \{-1,0,1\}} \left\{ |\sum_{i=1}^n (u_i - \sum_{j=1}^n u_j/n)\gamma_i| \right\}}$.

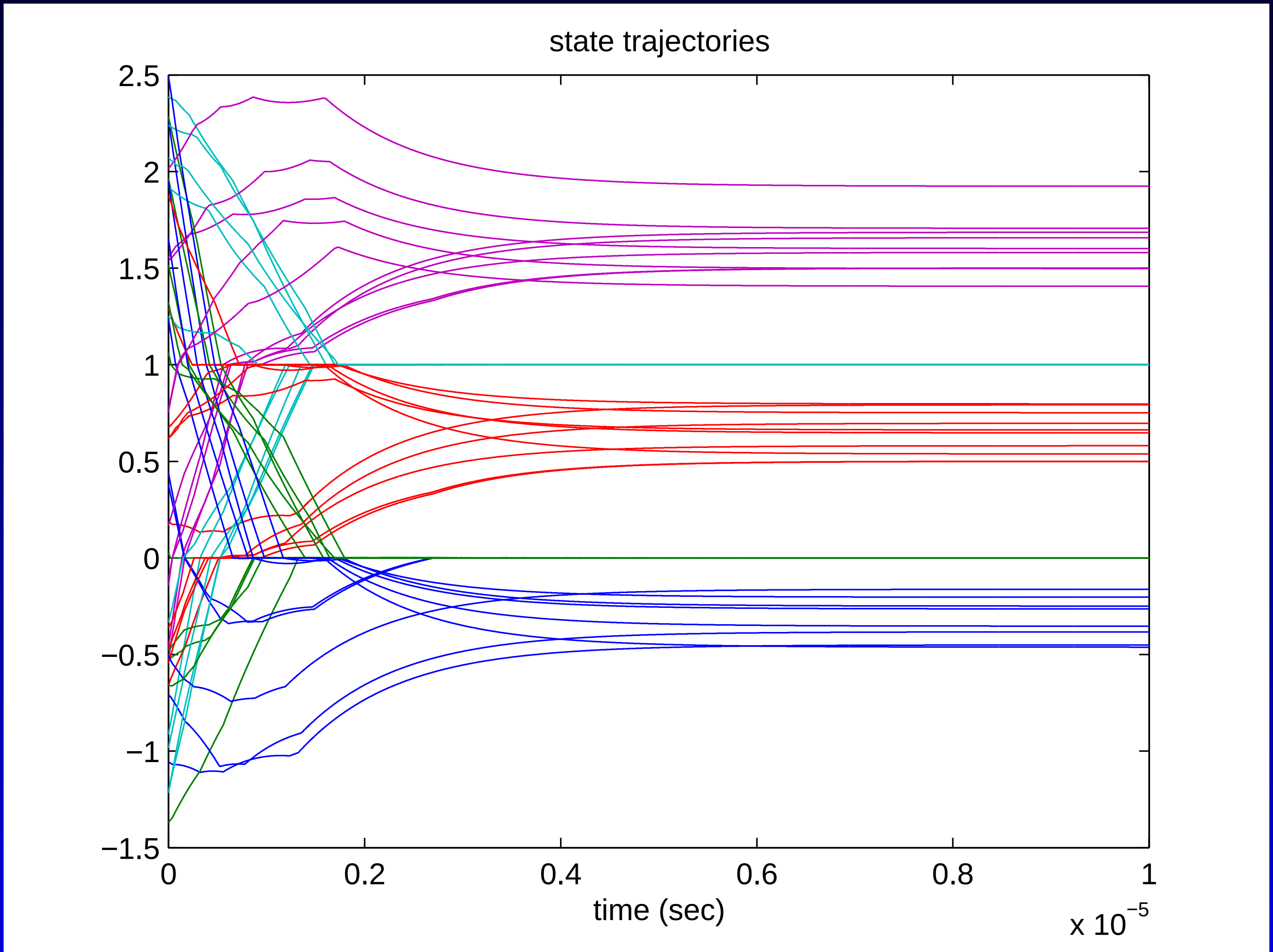
Simulation Results

Consider a kWTA problem with input vector $u_i = i \; (i = 1, 2, \ldots, n)$, $n = 5, k = 3$.



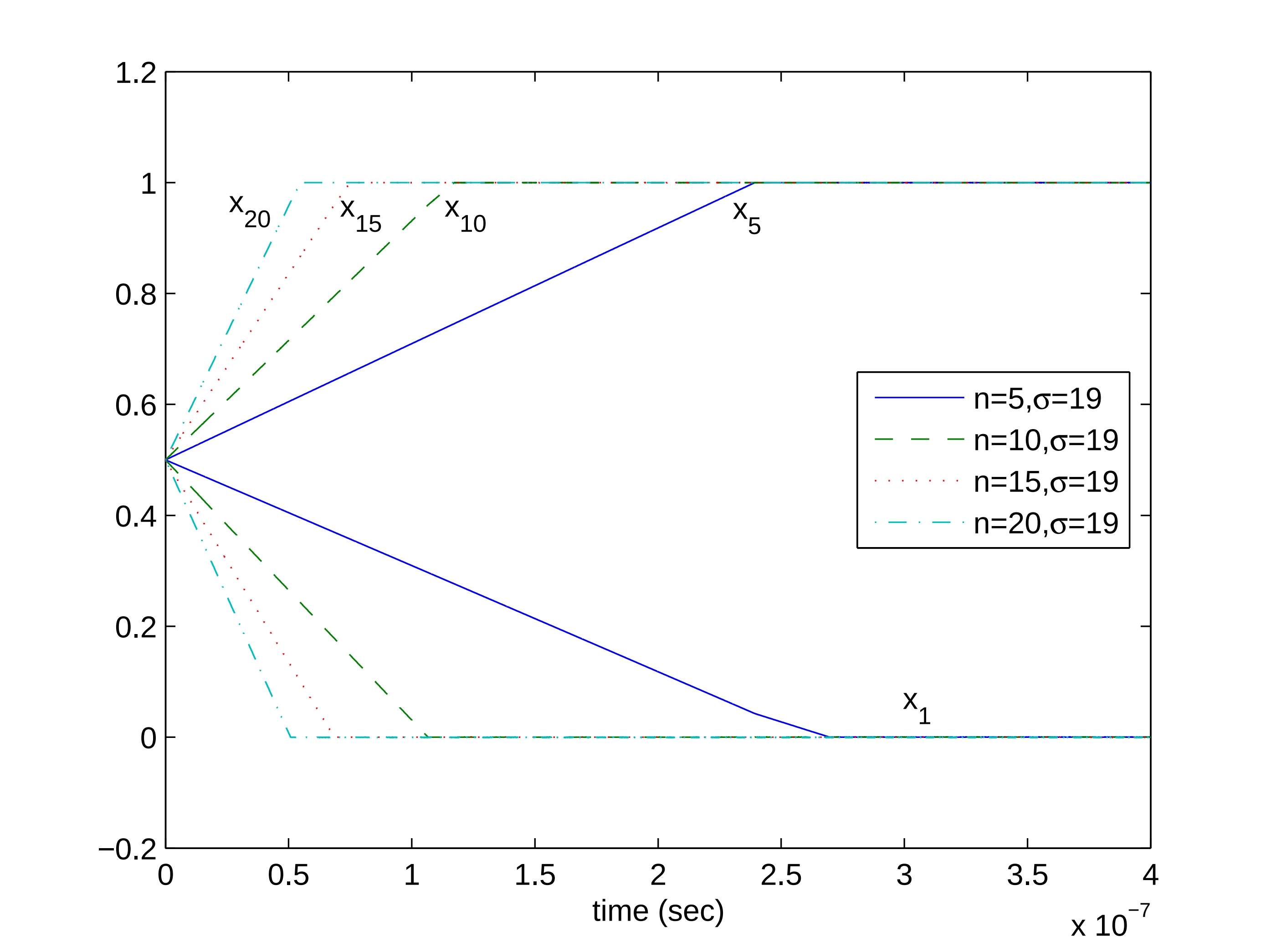
Transient behaviors of the kWTA network $\sigma = 6$.

Simulation Results (cont'd)



Transient behaviors of the kWTA network with $\sigma = 2$.

Simulation Results (cont’d)



Convergence behavior of the kWTA network with respect to different values of n.

QP-based One-layer kWTA Net

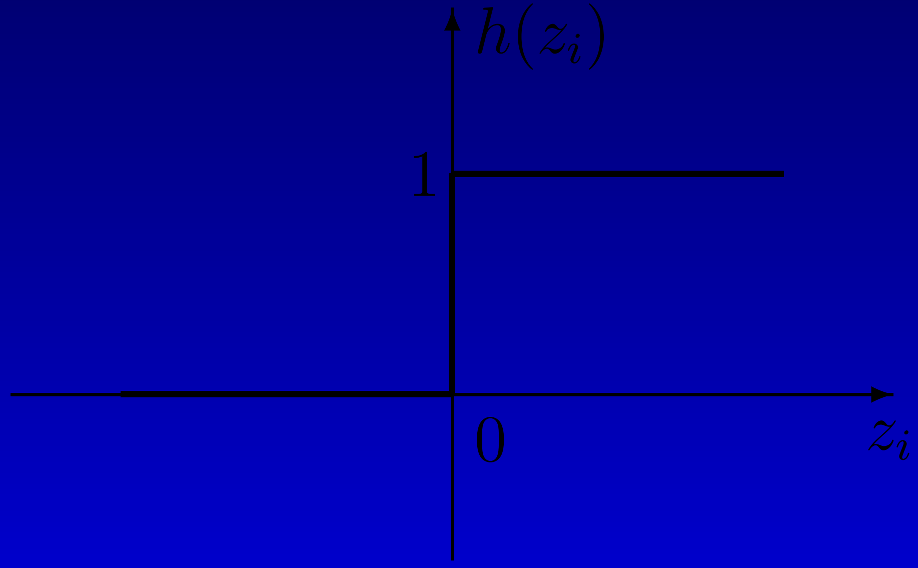
A QP-based kWTA network model with a discontinuous activation function is described as follows:

$$
\begin{aligned}
\epsilon \frac{dz}{dt} &= -(I-P)z - [aI + (1-a)P]g(z) + s, \\
x &= -\frac{1}{a}(I-P)z + \frac{s}{a} + \frac{k(a-1)}{na}e,
\end{aligned}
$$

where $g(z) = (g(z_1), g(z_2), \ldots, g(z_n))^T$ is a discontinous activation function and ϵ is a positive scaling constant.

Activation Function

$$g(z_i) = \begin{cases} 1, & \text{if } z_i > 0, \\ [0, 1], & \text{if } z_i = 0, \\ 0, & \text{if } z_i < 0. \end{cases}$$

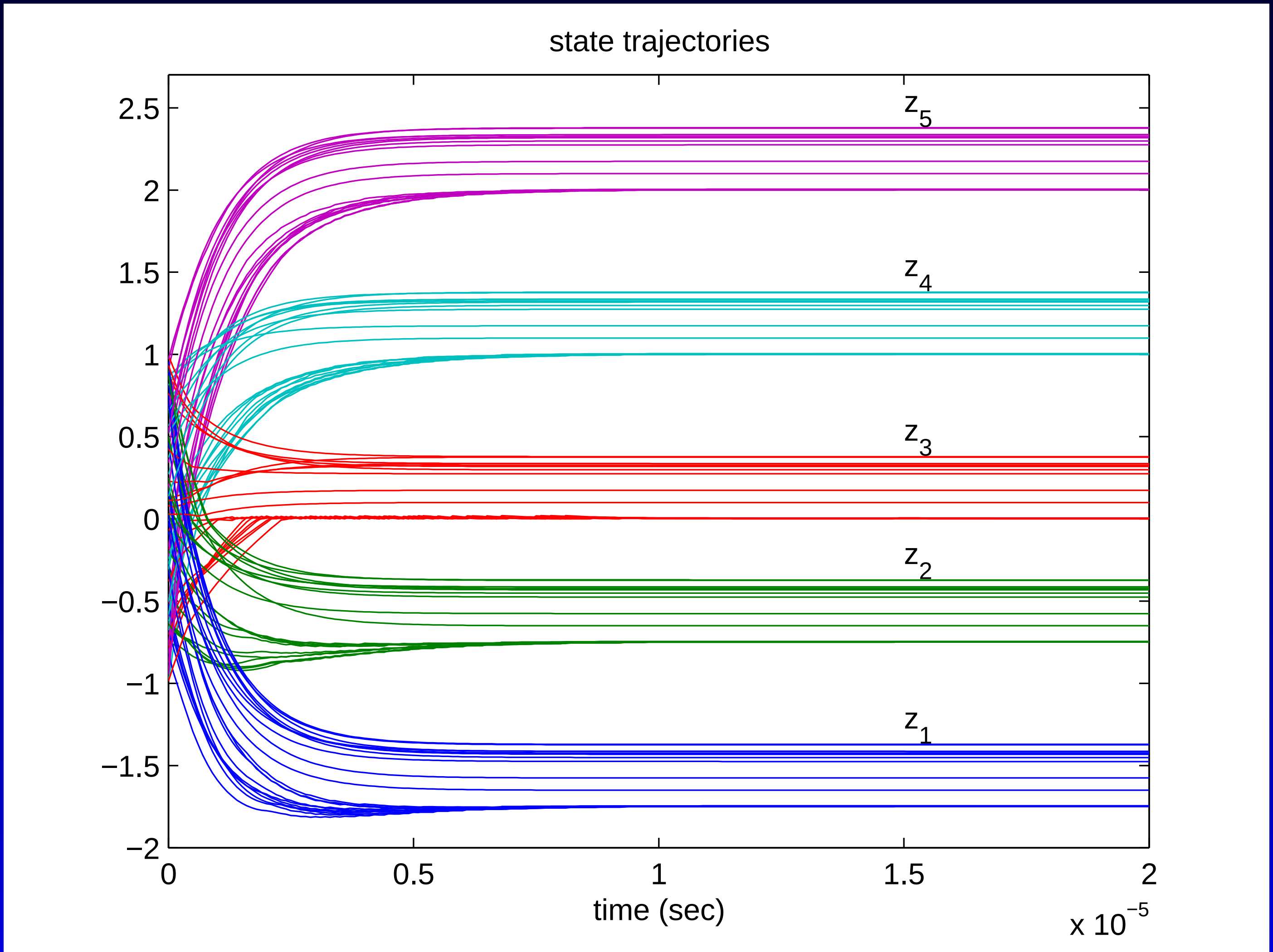


Convergence Results

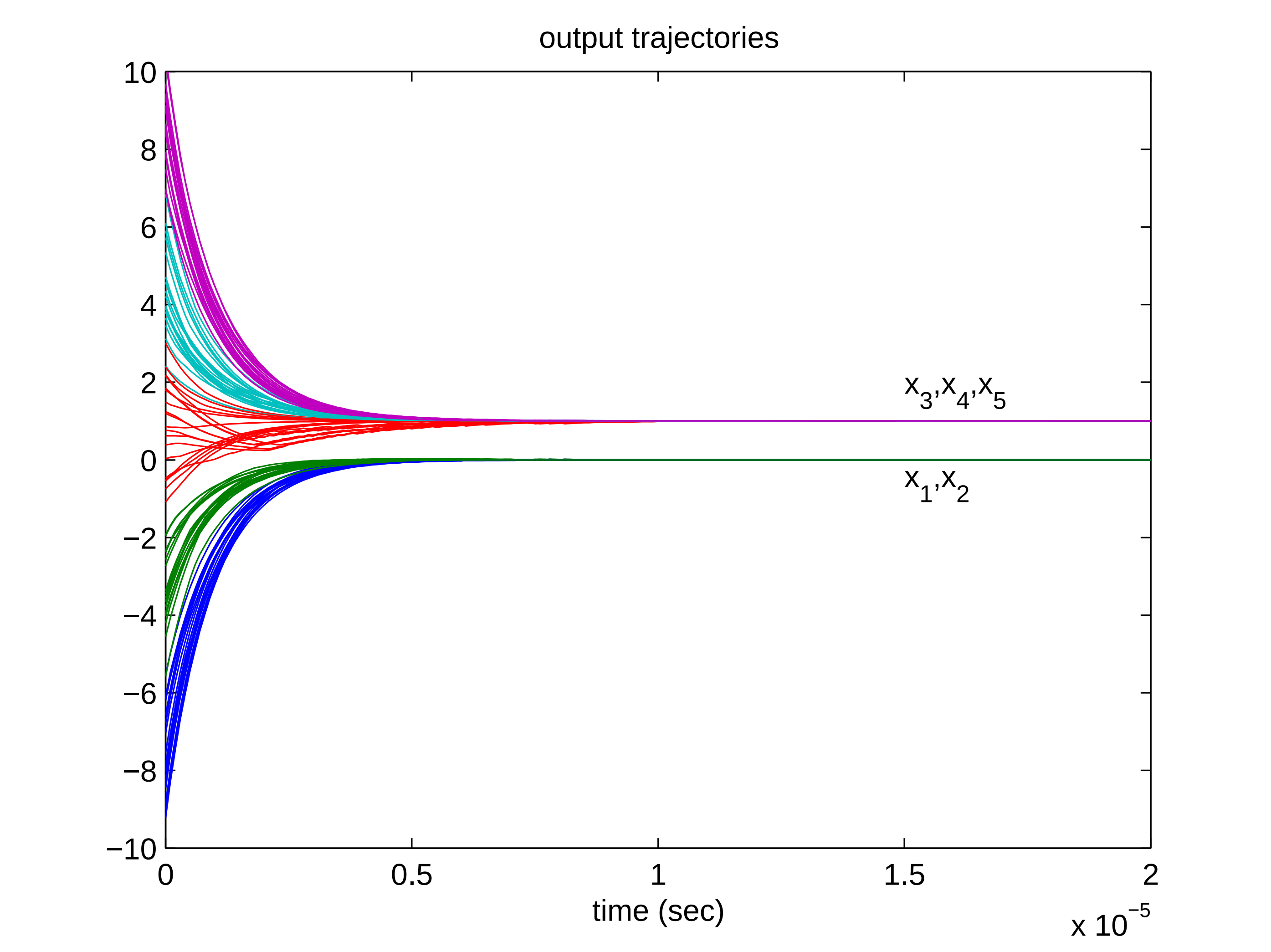
The neural network with any $a > 0$ is stable in the sense of Lyapunov and any trajectory is globally convergent to an equilibrium point.

$x^* = -(I - P)z^*/a + s/a + (a-1)ke/(na)$ is an optimal solution of kWTA problem, where z^* is an equilibrium point of the neural network.

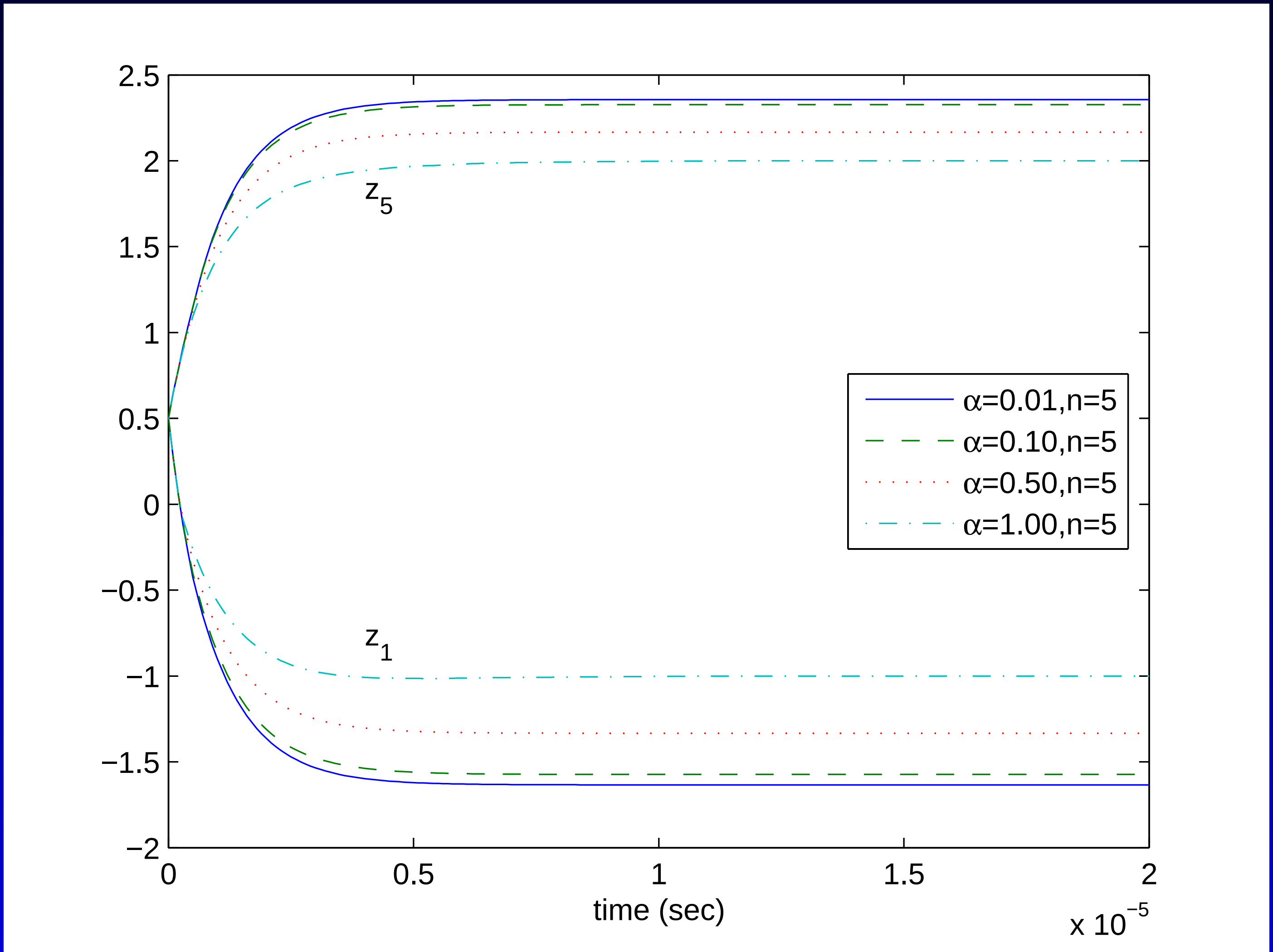
Simulation Results



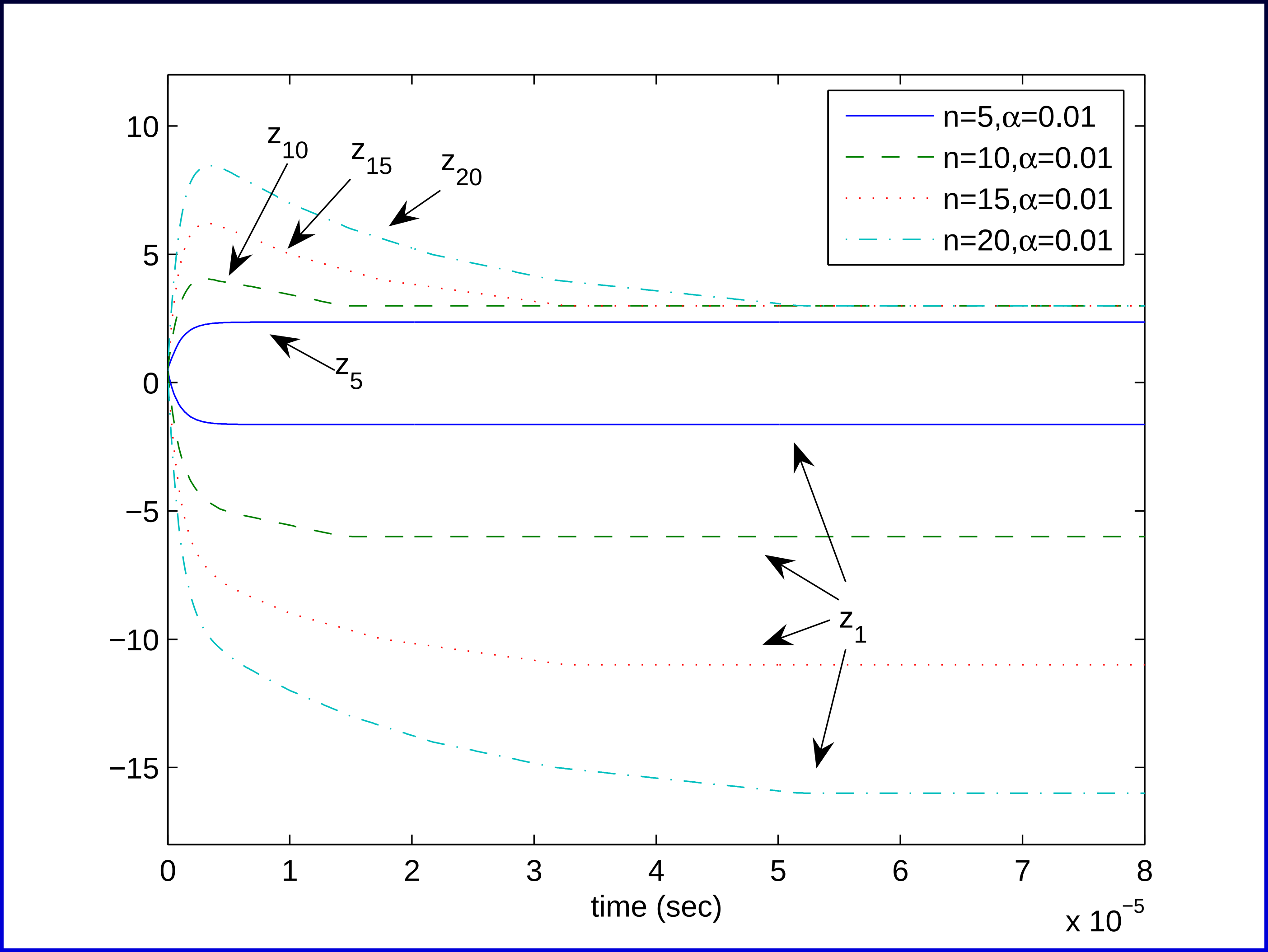
Simulation Results (cont'd)



Simulation Results (cont’d)

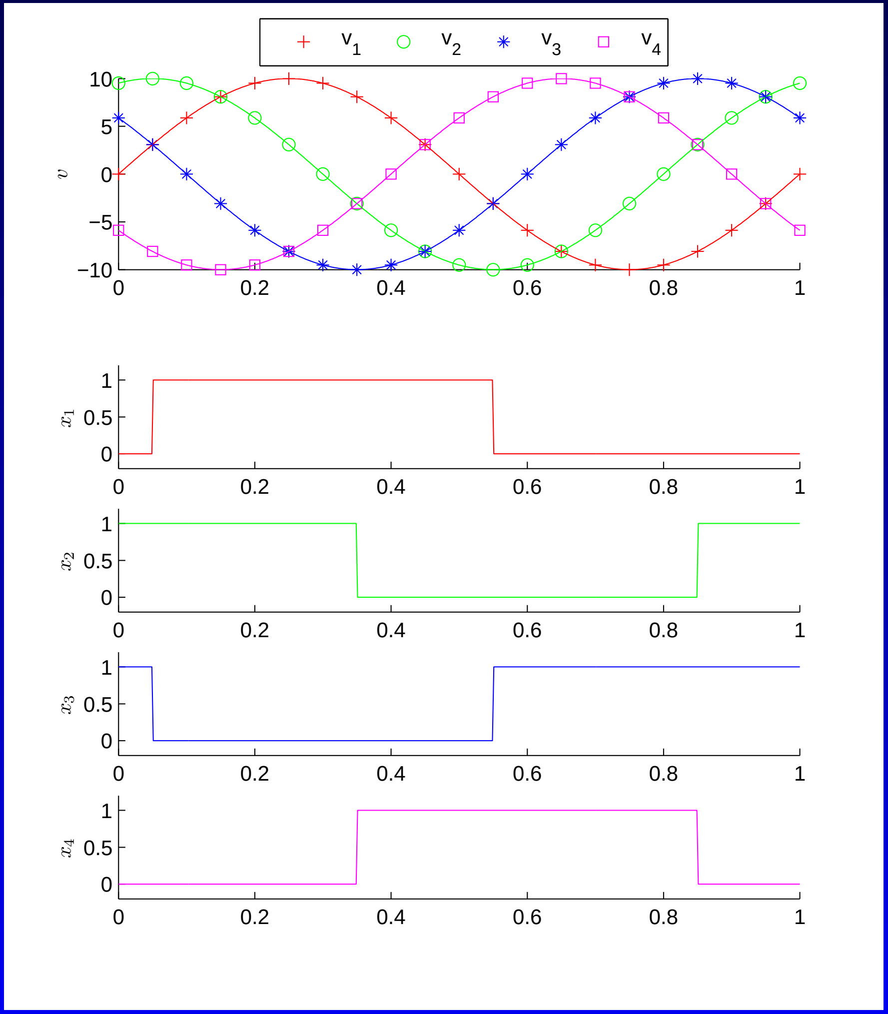


Simulation Results (cont’d)



A Dynamic Example

Let inputs be 4 sinusoidal input signals (i.e., $n = 4$) $u_i(t) = 10\sin[2\pi(1000t + 0.2(i-1)]$, and $k = 2$.



Model Comparisons

model	layer(s)	neurons	connections
LP-based primal-dual network	2	$n+1$	$2n+2$
QP-based primal-dual network	2	$3n+1$	$6n+2$
LP-based projection network	2	$n+1$	$2n+2$
QP-based projection network	2	$n+1$	$2n+2$
QP-based simplified dual network	1	n	$3n$
LP-based one-layer network	1	n	$2n$
QP-based one-layer network	1	n	$3n$

[a]

[a] Q. Liu, and J. Wang, "Two k-winners-take-all networks with discontinuous activation functions," *Neural Networks*, vol. 21, no. 2-3, pp. 406-413, 2008.

Concluding Remarks

Neurodynamic optimization has been demonstrated to be a powerful alternative approach to many optimization problems.

For convex optimization, recurrent neural networks are available with global convergence to the optimal solution.

Neurodynamic optimization approaches provide parallel distributed computational models more suitable for real-time applications.

Future Works

The existing neurodynamic optimization model can still be improved to reduce their model complexity or increase their convergence rate.

The available neurodynamic optimization model can be applied to more areas such as control, robotics, and signal processing.

Neurodynamic approaches to global optimization and discrete optimization are much more interesting and challenging.

It is more needed to develop neurodynamic models for nonconvex optimization and combinatorial optimization.