Neural, Parallel, and Scientific Computations 18 (2010) 89-108

# OPTIMAL CONTROL FOR STOCHASTIC LINEAR QUADRATIC SINGULAR TAKAGI-SUGENO FUZZY SYSTEM USING ANT COLONY PROGRAMMING

N. KUMARESAN

Institute of Mathematical Sciences, Faculty of Science, University of Malaya,
Kuala Lumpur 50603, Malaysia
drnk2008@gmail.com, drnk2008@um.edu.my

**ABSTRACT.** In this paper, optimal control for stochastic linear singular Takagi-Sugeno (T-S) fuzzy system with quadratic performance is obtained using ant colony programming (ACP). To obtain the optimal control, the solution of matrix Riccati differential equation (MRDE) is computed by solving differential algebraic equation (DAE) using ACP approach. The solution of this novel method is compared with the traditional Runge Kutta (RK) method. An illustrative numerical example is presented for the proposed method.

**Key words:** Ant colony programming, Differential algebraic equation, Matrix Riccati differential equation, Optimal control, Runge Kutta method and Stochastic linear quadratic singular Takagi-Sugeno fuzzy system

**AMS (MOS) Subject Classifications.** 49 K 45, 68 N 19, 93 E 20

## 1. INTRODUCTION

A fuzzy system consists of linguistic IF-THEN rules that have fuzzy antecedent and consequent parts. It is a static nonlinear mapping from the input space to the output space. The inputs and outputs are crisp real numbers and not fuzzy sets. The fuzzification block converts the crisp inputs to fuzzy sets and then the inference mechanism uses the fuzzy rules in the rule-base to produce fuzzy conclusions or fuzzy aggregations and finally the defuzzification block converts these fuzzy conclusions into the crisp outputs. The fuzzy system with singleton fuzzifier, product inference engine, center average defuzzifier and Gaussian membership functions is called as standard fuzzy system (Wang, 1998).

Two main advantages of fuzzy systems for the control and modeling applications are (i) fuzzy systems are useful for uncertain or approximate reasoning, especially for the system with a mathematical model that is difficult to derive and (ii) fuzzy logic allows decision making with the estimated values under incomplete or uncertain information (Zadeh, 1975). Fuzzy controllers are rule-based nonlinear controllers,

Received January 19, 2010 

therefore their main application should be the control of nonlinear systems. However, since linear systems are good approximations of nonlinear systems around the operating points, it is of interest to study fuzzy control of linear systems. Additionally, fuzzy controllers due to their nonlinear nature may be more robust than linear controllers even if the plant is linear. Furthermore, fuzzy controllers designed for linear systems may be used as initial controllers for nonlinear adaptive fuzzy control systems where on-line turning is employed to improve the controller performance. Therefore, a systematic fuzzy controllers for linear systems is of theoretical and practical interest. Stability and optimality are the most important requirements in any control system. Stable fuzzy control of linear systems has been studied by a number of researchers. It is well-known that nowadays that fuzzy controllers are universal nonlinear controllers. All these studies are preliminary in nature and deeper studies can be done. For optimality, it seems that the field of optimal fuzzy control is totally open.

Ant colony programming is a metaheuristic approach that is inspired by the behaviour of real ant colonies, to find a good solution to the given problems in a reasonable amount of computation time. It allows the programmer to avoid the tedious task of creating a program to solve a well-defined problem (Boryczka & Wiezorek, 2003). ACP is a stochastic search technique that is carried out on a space graph where the nodes represent functions, variables and constants. Functions are usually defined mathematically in terms of arithmetic operators, operands and boolean functions. The set of functions defining a given problem is called a function set $\mathbb{F}$ and the collection of variables and constants to be used are known as the terminal set $\mathbb{T}$.

Ants are able to find their way efficiently from their nest to food sources. While searching for food, ants initially explore the area surrounding their nest in a random manner. As soon as an ant finds a food source, it evaluates the quantity and the quality of the food and carries some of it back to the nest. During the return trip, the ant deposits a chemical pheromone trail on the ground. The quantity of pheromone deposited may depend on the quantity and quality of the food, will guide other ants to the food source. If an ant has a choice of trails to follow, the preferred route is the trail with the highest deposit of pheromone (Wilson & Hölldobler, 1990). This behaviour helps the ants to find the optimal route without any need for direct communication or central control. Therefore, the artificial ants used in the ACP have some features taken from the behaviour of real ants. The features are

(a) Artificial ants move in a random fashion.
(b) Choice of a route of an artificial ant depends on the amount of pheromone.
(c) Artificial ants co-operate in order to achieve the best result.

The ant colony algorithm can be described at a very simplified level as given in Figure 1.

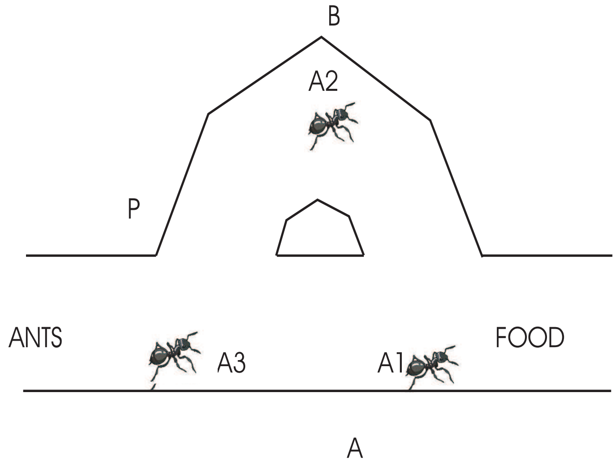


FIGURE 1. Shortest path of ants

Two ants A1 and A2 are travelling along route P and come to a junction. A1 takes path A and A2 takes path B. As they are travelling along the route, the ants are depositing a pheromone trail. Both ants continue along their chosen paths, collect the food and return to the nest. A1 will reach the nest first because it has travelled the shortest route. The third ant A3 now leaves the nest, travels along path P and reaches the junction. At this point, A2 has not yet returned through the junction and is still travelling along path B so there is twice the amount of pheromone deposited along path A at the junction as along path B. Therefore, A3 will opt for path A. Thus, the thickness of the pheromone level is increasing on path A. In fact, experiments by biologists have shown that ants probabilistically prefer paths with high pheromone concentration. Dorigo et al. (1991 & 1996) used the ant colony algorithm for solving the Travelling sales man problem. Roux and Fonlupt (2000) made the first attempt to apply ant colony algorithm for solving symbolic regression and multiplexer problem. Recently, researchers have been dealing with the relation of ant colony algorithm to other methods for learning, approximations and optimization. They have applied ACP in the field of optimal control and reinforcement learning (Birattari et al., 2002). In this paper, the ant colony algorithm is used in ACP to compute optimal control for stochastic linear quadratic singular fuzzy system.

In this paper, optimal control of stochastic linear quadratic singular T-S fuzzy system is obtained using ant colony programming. The linear T-S fuzzy system is the most popular fuzzy model due to its further intrinsic analysis: the linear matrix inequality (LMI)-based fuzzy controller is to minimize the upper bound of the performance index; structure oriented and switching fuzzy controllers are developed for more complicated systems (Tanaka & Iwazaki, 2001); the optimal fuzzy control technique is used to minimize the performance index from local-concept or global-concept approaches (Wu & Lin, 2000 & 2000).

Stochastic linear quadratic regulator (LQR) problems have been studied by many researchers (Athens, 1971; Bensoussan, 1983; Bucci & Pandolfi, 2000; Davis, 1997; Wonham, 1968). Chen et al. (1998) have shown that the stochastic LQR problem is well posed if there are solutions to the Riccati equation and then an optimal feedback control can be obtained. For LQR problems, it is natural to study an associated Riccati equation. However, the existence and uniqueness of the solution of the Riccati equation in general, seem to be very difficult problems due to the presence of the complicated nonlinear term. Zhu and Li (2003) used the iterative method for solving stochastic Riccati equations for stochastic LQR problems. There are several numerical methods to solve conventional Riccati equation as a result of the nonlinear process, essential error accumulations may occur. In order to minimize the error, recently, the conventional Riccati equation has been analyzed using neural network approach and genetic programming approach see (Balasubramaniam et al., 2006, 2007 & 2007; Vincent Antony Kumar & Balasubramaniam, 2007). A variety of numerical algorithms (Choi, 1990) have been developed for solving the algebraic Riccati equation.

Singular systems contain a mixture of algebraic and differential equations. In that sense, the algebraic equations represent the constraints to the solution of the differential part. These systems are also known as degenerate, differential algebraic, descriptor or semi state and generalized state space systems. The complex nature of singular system causes many difficulties in the analytical and numerical treatment of such systems, particularly when there is a need for their control. The system arises naturally as a linear approximation of system models or linear system models in many applications such as electrical networks, aircraft dynamics, neutral delay systems, chemical, thermal and diffusion processes, large scale systems, robotics, biology, etc., see (Campbell, 1980 & 1982; Lewis, 1986).

Although parallel algorithms can compute the solutions faster than sequential algorithms, there have been no report on ant colony programming solutions for MRDE. This paper focuses upon the implementation of ant colony programming approach for solving MRDE in order to get the optimal solution. An example is given to illustrate the advantage and accuracy of ACP solution by comparing RK solution.

This paper is organized as follows. In section 2, the statement of the problem is given. In section 3, solution of the MRDE is presented. In section 4, numerical example is discussed. The final conclusion section demonstrates the efficiency of the method.

## 2. STATEMENT OF THE PROBLEM

Consider the stochastic linear dynamical singular T-S fuzzy system (Wu et al., 2005) that can be expressed in the form: $R^i$: If $x_j$ is $T_{ji}(m_{ji}, \sigma_{ji})$, $i = 1, \ldots, r$ and

$j = 1, \ldots, n$, then

$$F_i dx(t) = [A_i x(t) + B_i u(t)]dt + D_i u(t)dW(t), \quad x(0) = x_0, \quad t \in [0, t_f], \tag{1}$$

where $R^i$ denotes the $i$th rule of the fuzzy model, $m_{ji}$ and $\sigma_{ji}$ are the mean and standard deviation of the Gaussian membership function, the matrix $F_i$ is singular, $x(t) \in R^n$ is a generalized state space vector, $u(t) \in R^m$ is a control variable and it takes value in some Euclidean space, $W(t)$ is a Brownian motion and $A_i \in \mathbb{R}^{n\times n}$, $B_i \in \mathbb{R}^{n\times m}$ and $D_i \in \mathbb{R}^{n\times m}$ are known as coefficient matrices associated with $x(t)$ and $u(t)$ respectively, $x_0$ is given initial state vector, $t_f$ is the final time and $m \le n$.

In order to minimize both state and control signals of the feedback control system, a quadratic performance index is usually minimized:

$$J = E\Big\{\frac{1}{2}x^T(t_f)F_i^T S F_i x(t_f) + \frac{1}{2}\int_0^{t_f}[x^T(t)Qx(t) + u^T(t)Ru(t)]dt\Big\},$$

where the superscript $T$ denotes the transpose operator, $S \in \mathbb{R}^{n\times n}$ and $Q \in \mathbb{R}^{n\times n}$ are symmetric and positive definite (or semidefinite) weighting matrices for $x(t)$, $R \in \mathbb{R}^{m\times m}$ is a singular weighting matrix for $u(t)$. It will be assumed that $|sF_i - A_i| \neq 0$ for some $s$. This assumption guarantees that any input $u(t)$ will generate one and only one state trajectory $x(t)$.

If all state variables are measurable, then a linear state feedback control law

$$u(t) = -(R + D_i^T K_i(t) D_i)^{-1} B_i^T \lambda(t)$$

can be obtained to the system described by equation (1), where

$$\lambda(t) = K_i(t) F_i x(t), \tag{2}$$

$K_i(t) \in \mathbb{R}^{n\times n}$ is a symmetric matrix and the solution of MRDE.

The relative MRDE for the stochastic linear singular system (1) is

$$\begin{aligned} F_i^T \dot{K}_i(t) F_i + F_i^T K_i(t) A_i + A_i^T K_i(t) F_i + Q \\ - F_i^T K_i(t) B_i (R + D_i^T K_i(t) D_i)^{-1} B_i^T K_i(t) F_i = 0 \end{aligned} \tag{3}$$

with terminal condition(TC) $K_i(t_f) = F_i^T S F_i$ and $(R + D_i^T K_i(t) D_i) > 0$.

## Derivation of MRDE

It is well known that minimizing $J$ is equivalent to minimize the Hamiltonian equation

$$\left.\begin{aligned} H(x(t), u(t), \lambda_1(t), \lambda_2(t), t) &= \tfrac{1}{2}x^T(t)Qx(t) + \tfrac{1}{2}u^T Ru(t) \\ &\quad + \lambda_1^T(t)[A_i x(t) + B_i u(t)] + \lambda_2^T(t)[D_i u(t)], \end{aligned}\right\}$$

where $\lambda_2(t) = K_i(t) D_i u(t)$, along the optimal trajectory.

Using the stochastic optimality conditions and stochastic maximum principle (Bismut, 2000), we obtain

$$\frac{\partial H}{\partial u(t)}(x(t), u(t), \lambda_1(t), \lambda_2(t), t) = 0$$

implies that

$$(R + D_i^T K_i(t) D_i) u(t) + B_i^T \lambda_1(t) = 0$$

$$\Rightarrow u(t) = -(R + D_i^T K_i(t) D_i)^{-1} B_i^T \lambda_1(t) \tag{4}$$

and

$$\frac{\partial H}{\partial x(t)} = F_i^T d\lambda_1(t)$$

$$\Rightarrow F_i^T d\lambda_1(t) = [-Qx(t) - A_i^T \lambda_1(t)]dt + \lambda_2 dW(t) \tag{5}$$

$$\frac{\partial H}{\partial \lambda_1(t)} = F_i dx(t),$$

$$\Rightarrow F_i dx(t) = [A_i x(t) + B_i u(t)]dt + D_i u(t) dW(t)$$

and from (4), we have

$$F_i dx(t) = [A_i x(t) - B_i (R + D_i^T K_i(t) D_i)^{-1} B_i^T \lambda_1(t)]dt + D_i u(t) dW(t). \tag{6}$$

From (2), we have

$$d\lambda_1(t) = \dot{K}_i(t) F_i x(t) + K_i(t) F_i dx(t)$$

and also we have

$$F_i^T d\lambda_1(t) = F_i^T \dot{K}_i(t) F_i x(t) + F_i^T K_i(t) F_i dx(t). \tag{7}$$

Using the equations (5) and (6) in (7), we obtain

$$\left.\begin{aligned}[F_i^T \dot{K}_i(t) F_i \quad & + F_i^T K_i(t) A_i + A_i^T K_i(t) F_i + Q \\ & - F_i^T K_i(t) B_i (R + D_i^T K_i(t) D_i)^{-1} B_i^T K_i(t) F_i] x(t) dt + dM = 0,\end{aligned}\right\} \tag{8}$$

where $dM = D_i u(t) - \lambda_2(t) dW(t)$ and $M$ is the integrable Martingale.

Since equation (8) holds for all non-zero $x(t)$ and $M = 0$, then the term pre-multiplying $x(t)$ must be zero. Therefore, we obtain the following (MRDE) for the stochastic bilinear singular system (1).

$$\left.\begin{aligned}F_i^T \dot{K}_i(t) F_i + F_i^T K_i(t) A_i \quad & + A_i^T K_i(t) F_i + Q \\ & - F_i^T K_i(t) B_i (R + D_i^T K_i(t) D_i)^{-1} B_i^T K_i(t) F_i = 0.\end{aligned}\right\}.$$

This equation is going to be solved for $K_i(t)$ in the next section for the optimal solution.

After substituting the appropriate matrices in the above equation, it becomes a DAE of index one. Therefore, solving MRDE is equivalent to solving the DAE of index one.

## 3. SOLUTION OF MRDE

Consider the DAE for (3)for each rule of the fuzzy model

$$\begin{aligned} &\dot{k}_{ij}(t) = \phi_{ij}(k_{ij}(t)), \quad k_{ij}(t_f) = A_{ij} \quad (i, j = 1, 2, \ldots, n-1) \\ &k_{1n}(t) = \psi(k_{ij}(t)), \quad k_{1n}(t_f) = A_{1n}. \end{aligned} \tag{9}$$

3.1. **RUNGE KUTTA METHOD.** Numerical integration is one of the oldest and most fascinating topics in numerical analysis. It is the process of producing a numerical value for the integration of a function over a set. Numerical integration is usually utilized when analytic techniques fail. Even if the indefinite integral of the function is available in a closed form, it may involve some special functions, which cannot be computed easily. In such cases too, we can use numerical integration. RK algorithms have always been considered as the best tool for the numerical integration of ordinary differential equations (ODEs). The DAE can be changed into system of nonlinear differential equation by differentiating the algebraic equation once since the DAE is of index one type. The system (3) contains $n^2$ first order ODEs with $n^2$ variables. In particular $n = 2$, the system will contain four equations. Since the matrix $K(t)$ is symmetric and the system is singular, $k_{12} = k_{21}$ and $k_{22}$ is free (let $k_{22} = 0$). Finally the system will have two equation in two variables. Hence RK method is explained for a system of two first order ODEs with two variables.

$$\begin{aligned} k_{11}(i+1) &= k_{11}(i) + \frac{1}{6}\Big(k1 + 2k2 + 2k3 + k4\Big) \\ k_{12}(i+1) &= k_{12}(i) + \frac{1}{6}\Big(l1 + 2l2 + 2l3 + l4\Big) \end{aligned}$$

where

$$\begin{aligned} k1 &= h * \phi_{11}\Big(k_{11}, k_{12}\Big) \\ l1 &= h * \phi_{12}\Big(k_{11}, k_{12}\Big) \\ k2 &= h * \phi_{11}\Big(k_{11} + \frac{k1}{2}, k_{12} + \frac{l1}{2}\Big) \\ l2 &= h * \phi_{12}\Big(k_{11} + \frac{k1}{2}, k_{12} + \frac{l1}{2}\Big) \\ k3 &= h * \phi_{11}\Big(k_{11} + \frac{k2}{2}, k_{12} + \frac{l2}{2}\Big) \\ l3 &= h * \phi_{12}\Big(k_{11} + \frac{k2}{2}, k_{12} + \frac{l2}{2}\Big) \\ k4 &= h * \phi_{11}\Big(k_{11} + k3, k_{12} + l3\Big) \\ l4 &= h * \phi_{12}\Big(k_{11} + k3, k_{12} + l3\Big). \end{aligned}$$

In a similar way, the original system (3) can be solved for $n^2$ first order ODE's.

3.2. **ANT COLONY PROGRAMMING METHOD.** In this approach, ACP is used to obtain a set of expressions. If the required number expressions satisfy the fitness function, it will be the optimal solution of (3). The scheme of computing optimal solution is given in Figure 2.

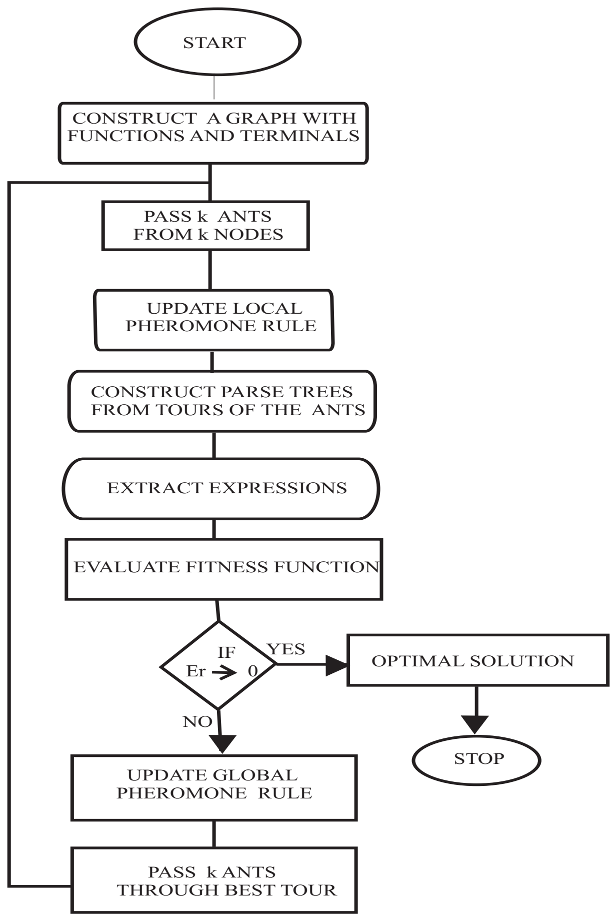


FIGURE 2. Flow Chart

According to Boryczka and Wiezorek (2003), the following four preparatory steps are essential for a searching process.

- Choice of terminals and functions
- Construction of graph
- Defining fitness function
- Defining terminal criteria.

**Choice of Terminals and Functions**

A terminal symbol $t_i \in \mathbb{T}$ can be a constant or a variable. Every function $f_i \in \mathbb{F}$ can be an arithmetic operator $\{+, -, *, /\}$, an arithmetic function (sin, cos, exp, log) and an arbitrarily defined function appropriate to the problem under consideration. The terminal symbols and functions have chosen such that they provide sufficient expressive power to express the solution to a problem. This means that the problem must be solved by a composition of functions and terminals specified. For solving the DAE (3), terminal set and function set are taken as $\mathbb{T} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, t\}$ and $\mathbb{F} = \{+, -, *, /, \sin, \cos, \exp, \log\}$.

## Construction of Graph

In ant colony programming technique, the search space consists of a graph with $\ell$ nodes where the nodes are the functions or terminals and edges are weighted by pheromone. The examples of such a graph are given in Figures 3 and 4. Each node in the graph holds either a function or a terminal. This graph is generated by a randomized process.

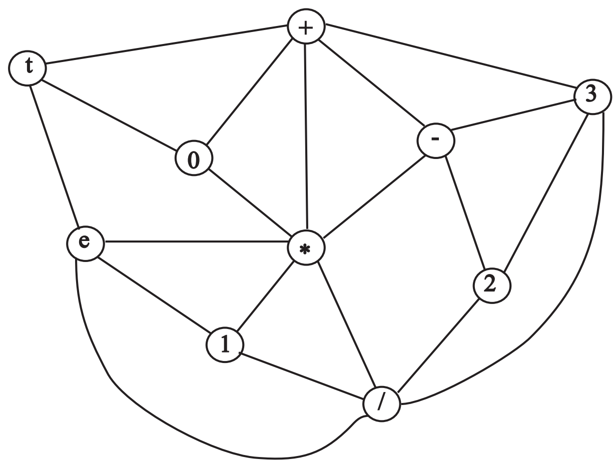


FIGURE 3. Graph with functions and terminals

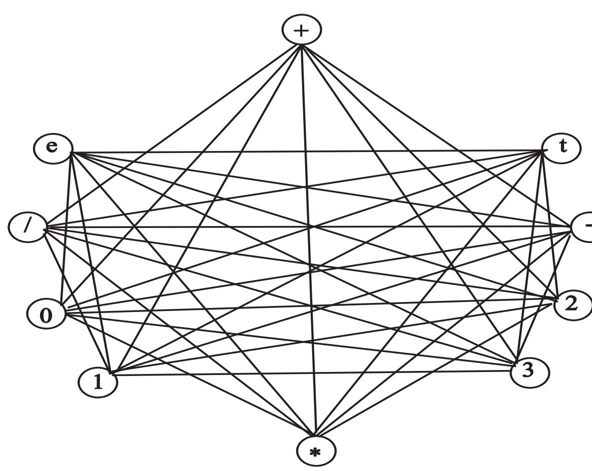


FIGURE 4. Graph with functions and terminals

## Fitness Function

The aim of the fitness function is to provide a basis for competition among available solutions and to obtain the optimal solution. Hence the fitness function for (3) is defined as

(10)

$$E_r = \Big(k_{1n}(t_m) - \varphi(k_{ij}(t_m))\Big)^2 + \sum_{i,j=1}^{n-1} \Big(\dot{k}_{ij}(t_m) - \phi_{ij}(k_{ij}(t_m))\Big)^2, \quad (m = 0, 1, 2, \ldots, t_f),$$

where $m$ represents the equidistance points in the relevant range $[0, t_f]$.

## Terminal Criteria

The group of ants and their collective tours form a generation. In each generation, a set of expressions are generated by the artificial ants. If the required number of expressions minimize the fitness function $E_r$ to zero or very close to zero and they satisfy the terminal conditions, the process may be stopped; otherwise continue the ACP approach.

## ACP Methodology

Artificial ants build solutions by performing randomized tours on the completely connected graph $G(\mathbb{V}, \mathbb{E})$. In the graph, vertices ($\mathbb{V}$) are represented by Functions and Terminals and the set ($\mathbb{E}$) of edges connect the vertices. The ants move on the graph by applying a stochastic local decision policy that makes use of pheromone trails and heuristic information. In this way, ants incrementally build solutions to the given problem.

In the first generation, all edges are initialized by equal pheromone weight. Send $k$ $(< \ell)$ ants through the graph from $k$ starting points in a random fashion. Each ant is initially put on a randomly chosen start node. Each ant is moving from the node $r$ to node $s$ in the graph at time $t$ according to the following probability law (Boryczka, 2005)

$$p_{rs}(t) = \frac{\tau_{rs}(t) \cdot [\gamma_s]^\beta}{\sum_{i \in J_r^k} [\tau_{ri}(t)] \cdot [\gamma_i]^\beta},$$

where $\gamma_s = (1/(2 + \pi_s))^d$, $\pi_s$ is the power of symbol $s$ which can be either a terminal symbol or a function, $d$ is the current length of the arithmetic expression, $\beta$ is a parameter which controls the relative weight of the pheromone trail and visibility and $J_r^k$ is the set of unvisited nodes. The power of the symbols can be calculated from Table 1. When an ant reaches a node, it determines whether the node is a terminal or a function node. If the ant is on a terminal node, the end of the tour has been reached by that ant.

| Terminal symbol or function | Power |
|---|---|
| Constant, variable | -1 |
| Functions | 1 |

TABLE 1. Power of terminal symbols and functions

After completing the tour, the ant deposits pheromone information on the edges through which it travelled. It constitutes a local update of the pheromone trail, which also comprises partial evaporation of the trail. The local update process is carried out according to the formula:

$$\tau_{ij}(t+1) = (1-\rho).\tau_{ij}(t) + \rho \cdot \tau_0$$

where $(1-\rho)$, $\rho \in (0,1]$, is the pheromone decay coefficient, $\tau_{ij}$ is the amount of pheromone on edge $(i,j)$ and $\tau_0$ is the initial amount of pheromone on edge $(i,j)$.

Each ant has a working memory that stores data about its tour. The ant's memory is represented programmatically by a parse tree structure. In this tree, the root and branches are functions and leaves are terminals. The depth of the memory tree is limited according to the nature of the problem.

The tour $e{*}t{+}1{*}{+}5$ of an ant is represented as parse trees in Figure 5. The tour $e{*}t/7{*}{+}4/5$ of another ant is represented as parse trees in Figure 6. The tours of the

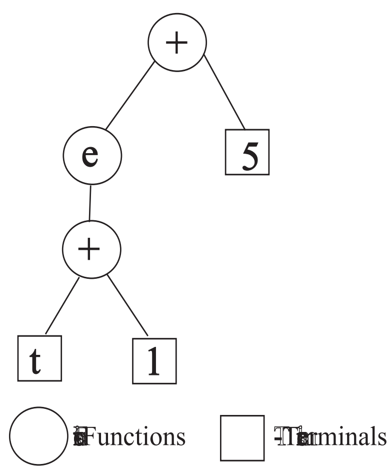


FIGURE 5. Tour of an ant and its parse tree

ants and their corresponding expressions extracted from the parse trees are given in Table 2. Some tours of the ants can not be represented as the parse trees. Such type of tours are given in Table 3. They are discarded when the parse tree construction process is carried out for the tours of the ants. This parse tree construction is helpful to converge the solution quickly and also reduces the computation time by discarding the unnecessary tours. After each generation, a global update of pheromone trail

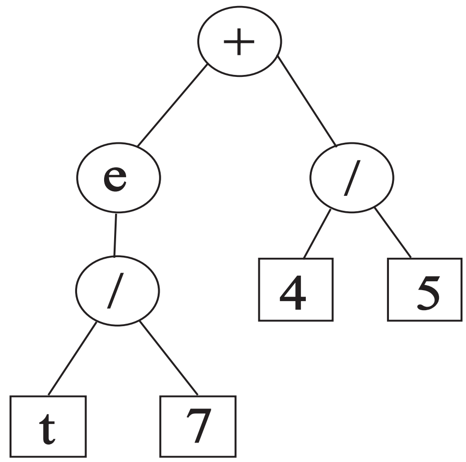


FIGURE 6. Tour of an ant and its parse tree

| Tours of ants | Expressions |
|---|---|
| $e * t *$ | $e^t$ |
| $e * t + 1 * +5$ | $e^{t+1} + 5$ |
| $e * t/5 * +0$ | $e^{t/5} + 0$ |
| $e * t/7 * +4/5$ | $e^{t/7} + 4/5$ |
| $e * 3 * t - 2 * * + 5/2$ | $e^{3(t-2)} + 5/2$ |
| $e * 3 * t * +5/2 - e * t * /7$ | $e^{3t} + 5/2 - e^t/7$ |

TABLE 2. Tours and Expressions

| Tours of ants |
|---|
| $+e * t * +1$ |
| $e * t + 1 * + - 5$ |
| $e * t/ - *2$ |
| $- * e * t/7 * + - / + 0$ |

TABLE 3. Discarded tours

takes place. The level of pheromone is then changed as follows:

$$\tau_{ij}(t + g) = (1 - \rho) \cdot \tau_{ij}(t) + \rho \cdot \frac{1}{L},$$

where $g$ is the number of generations, edges $(i, j)$ belong to the optimal tour found so far and $L$ is the length of this tour. The aim of the pheromone value update rule is to increase the pheromone values on the solution path. The update rule reduces the size of the searching region in order to find high quality solution with reasonable computation time. On the updated graph, the consecutive cycles of the ant colony algorithm are carried out by sending the ants through the best tour of the previous generation. The procedure is repeated until the fitness function (10) becomes zero or very close to zero. The optimal tour of the ACP and its corresponding tree are given in Figures 7 and 8 respectively.

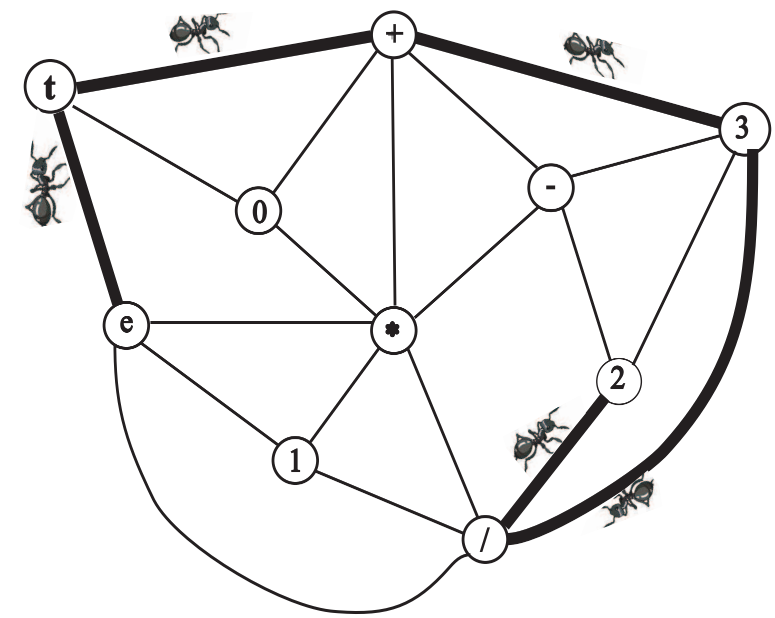


FIGURE 7. Optimal tour of the ACP

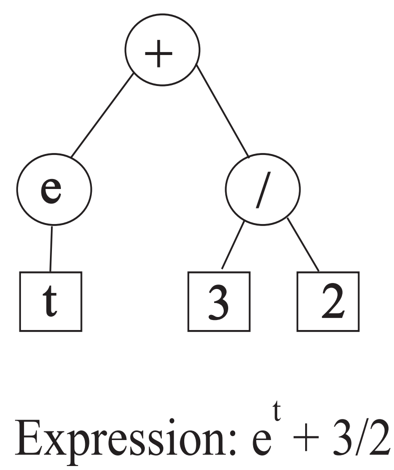


FIGURE 8. Parse tree and its expression

**ACP Algorithm.**

Step 1. Construct a graph with $\ell$ nodes.
Step 2. Initialize the equal weight of pheromone in each edge of the graph.
Step 3. Pass $k$ ants through the graph from $k$ starting points and They move to the next node according to the probability law.
Step 4. Apply local update rule after the tour of each ant.
Step 5. Construct parse trees from the tours of $k$ ants.

Step 6. Extract the expressions from the trees.

Step 7. Evaluate the fitness function.

Step 8. If $E_r \to 0$ and they satisfy the terminal conditions, then stop. Otherwise, apply global update rule.

Step 9. Identify the best tour of the previous generation.

Step 10. Pass the same $k$ ants through the best tour and go to Step 4.

## 4. NUMERICAL EXAMPLE

Consider the optimal control problem:

Minimize

$$J = E\Big\{\frac{1}{2}x^T(t_f)F_i^T S F_i x(t_f) + \frac{1}{2}\int_0^{t_f}[x^T(t)Qx(t) + u^T(t)Ru(t)]dt\Big\}$$

subject to the stochastic linear singular T-S fuzzy system $R^i$: If $x_j$ is $T_{ji}(m_{ji}, \sigma_{ji})$, $i = 1, 2$ and $j = 1, 2$, then

$$F_i dx(t) = [A_i x(t) + B_i u(t)]dt + D_i u(t)dW(t), \quad x(0) = x_0$$

where fuzzy term sets $T_{11}(0.4158, 0.6545)$, $T_{12}(0.597, 0.7889)$, $T_{21}(0.3982, 0.5249)$, $T_{22}(0.8596, 0.6376)$;

$$S = \begin{bmatrix} 2 & 3 \\ 3 & 5 \end{bmatrix}, \quad F_i = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad A_1 = \begin{bmatrix} -1 & -1 \\ 0 & 1 \end{bmatrix}, \quad A_2 = \begin{bmatrix} -2 & -2 \\ 0 & 2 \end{bmatrix},$$

$$B_i = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad R = 1, \quad Q = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \quad D_i = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

The numerical implementation could be adapted by taking $t_f = 2$ for solving the related MRDE of the above linear singular fuzzy system with the matrix $A_1$. The appropriate matrices are substituted in equation (2), the MRDE is transformed into DAE in $k_{11}$ and $k_{12}$. In this problem, the value of $k_{22}$ of the symmetric matrix $K(t)$ is free and let $k_{22} = 0$. Then the optimal control of the system can be found out by the solution of MRDE.

### 4.1. SOLUTION OBTAINED USING ANT COLONY PROGRAMMING.

The graph is generated randomly with 10 nodes. Let $\rho = 0.8$ and $\beta = 2$. Each edge is initialized by a pheromone weight of 1.0. Six ants are taken to be sent through the graph from 6 different points. The equidistance points of the interval $[0, 2]$ are taken as $m = 0, 1, 2$.

As the ant colony programming is carried out continuously, the solution of each generation will be improved by the pheromone updating rules. The construction of parse tree for the tour of the ants will converge the optimal solution quickly by discarding some unnecessary tours.

| | RK Solution | | ACP Solution | |
|---|---|---|---|---|
| $t$ | $k_{11}$ | $k_{12}$ | $k_{11}$ | $k_{12}$ |
| 0.0 | 0.004960 | 1.004960 | 0.004957 | 1.004957 |
| 0.2 | 0.009037 | 1.009037 | 0.009033 | 1.009033 |
| 0.4 | 0.016466 | 1.016466 | 0.016459 | 1.016459 |
| 0.6 | 0.030002 | 1.030002 | 0.029991 | 1.029991 |
| 0.8 | 0.054665 | 1.054664 | 0.054647 | 1.054647 |
| 1.0 | 0.099600 | 1.099600 | 0.099574 | 1.099574 |
| 1.2 | 0.181474 | 1.181474 | 0.181435 | 1.181435 |
| 1.4 | 0.330649 | 1.330649 | 0.330597 | 1.330597 |
| 1.6 | 0.602451 | 1.602451 | 0.602388 | 1.602388 |
| 1.8 | 1.097680 | 2.097680 | 1.097623 | 2.097623 |
| 2.0 | 2.000000 | 3.000000 | 2.000000 | 3.000000 |

TABLE 4. Solutions of MRDE

After 50 generations, the value of the fitness function is 301.7303 and the corresponding tours and expression are given below.

$$\text{Tours } : k_{11} = e * t - 2 * +3, \quad k_{12} = e * t - 2 * +4$$

$$\text{Expressions } : k_{11} = e^{(t-2)} + 3, \quad k_{12} = e^{(t-2)} + 4$$

After 100 generations, the value of the fitness function is 49.6532 and the corresponding tours and expression are given below.

$$\text{Tours } : k_{11} = e * t - 2 * +1, \quad k_{12} = e * t - 2 * +2$$

$$\text{Expressions } : k_{11} = e^{(t-2)} + 1, \quad k_{12} = e^{(t-2)} + 2$$

After 150 generations, the following expressions satisfy the fitness function and the value of the fitness function tends to 0. The expressions also satisfy the terminal conditions. Hence the solution of the DAE/MRDE is obtained.

$$\text{Tours } : k_{11} = e * 3 * t - 2 * * * 2, \quad k_{12} = e * 3 * t - 2 * * * 2 + 1$$

$$\text{Expressions } : k_{11} = 2e^{(3(t-2))}, \quad k_{12} = 2e^{(3(t-2))} + 1$$

The numerical solutions of MRDE are calculated and displayed in Table 4 using ACP and RK method. Since this problem provides explicit solution, the ACP solution is equivalent to exact solution of the DAE. ACP solution curves are shown in Figures 9 and 12.

The parse trees for the solutions are given in Figures 10 and 13. The error region between ACP and RK solutions are given in Figures 11 and 14.

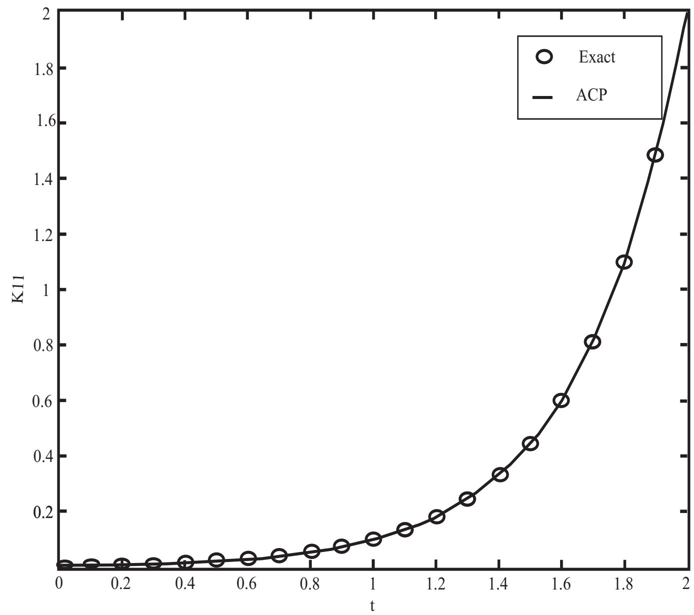


FIGURE 9. Solution curve for $k_{11}$

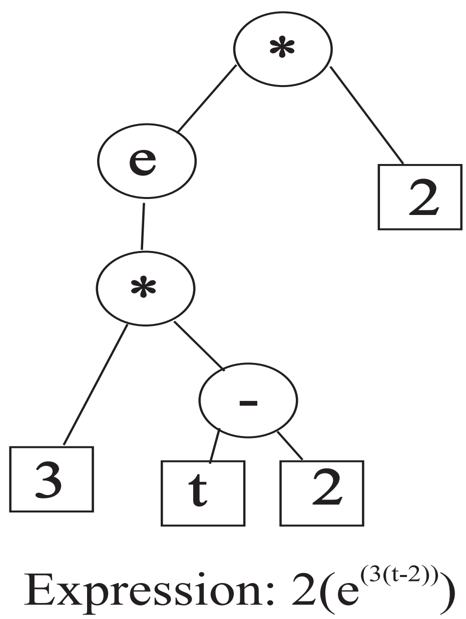


FIGURE 10. Tree and Expression for $k_{11}$

Similarly, the solution of the above system with the matrix $A_2$ can be found out using ant colony programming.

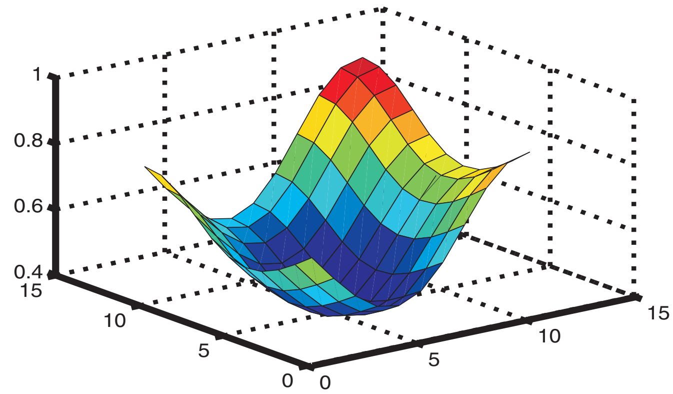


FIGURE 11. Error region for $k_{11}$

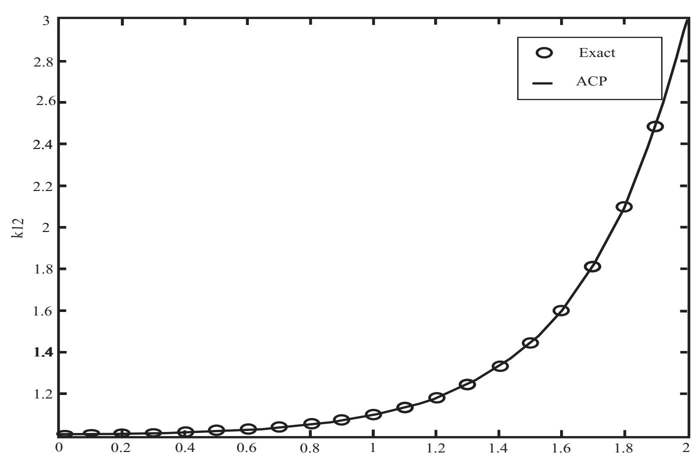


FIGURE 12. Solution curve for $k_{12}$

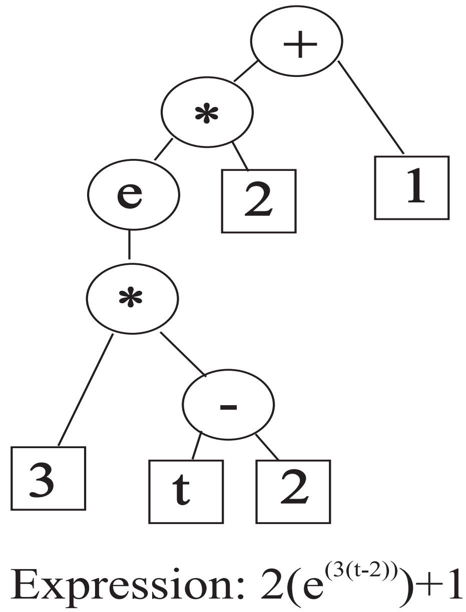


Figure 13. Tree and Expression for $k_{12}$

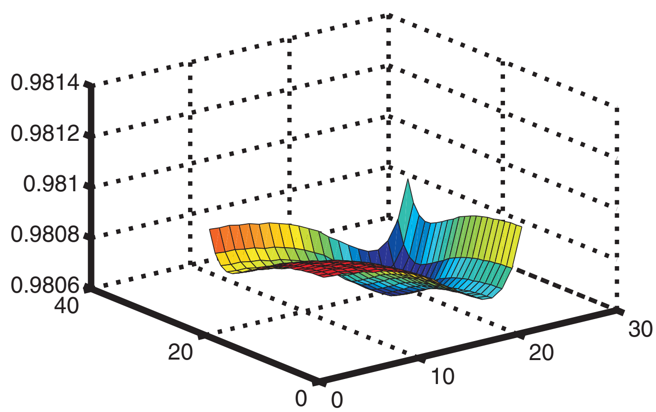


Figure 14. Error region for $k_{12}$

## 5. CONCLUSION

The optimal control for the stochastic linear quadratic singular T-S fuzzy system can be found by ACP approach. To obtain the optimal control, the solution of MRDE is computed by solving differential algebraic equation (DAE) using a novel and nontraditional ACP approach. The obtained solution in this method is equivalent to the exact solution of the problem. Accuracy of the solution computed by ACP approach to the problem is qualitatively better when it is compared with RK solution. A numerical example is given to illustrate the derived results.

## REFERENCES


[1] Athens, M., (1971). Special issues on linear quadratic Gaussian problem, *IEEE Trans. Automat. Control*, v.AC-16, Programming. 527–869.

[2] Balasubramaniam, P., Abdul Samath, J., Kumaresan, N., Vincent Antony Kumar, A., (2006). Solution of matrix Riccati differential equation for the linear quadratic singular system using neural networks, *Appl. Math. Comput.*, v.182, Programming. 1832–1839.

[3] Balasubramaniam, P., Abdul Samath, J., Kumaresan, N., (2007). Optimal control for nonlinear singular systems with quadratic performance using neural networks, *Appl. Math. Comput.*, v.187, Programming. 1535–1543.

[4] Balasubramaniam, P., Abdul Samath, J., Kumaresan, N., Vincent Antony Kumar, A., (2007). Neuro approach for solving matrix Riccati differential equation, *Neural, Parallel Sci. Comput.*, v.15, Programming. 125–135.

[5] Bensoussan, A. (1983). Lecture on stochastic control part I. in : *Nonlinear and Stochastic Control*, Lecture Notes in Math. 972, 1–39, Springer-Verlag, Berlin.

[6] Birattari, M., Di Caro, G., Dorigo, M., (2002). Toward the formal foundation of Ant programming, in:M. Dorigo, G. Di Caro, M. Sampels.(Eds.), Ant Algorithms, *Proceedings of ANTS*, , Programming. 188–201.

[7] Bismut, J. M., (2000) An introductory approach to duality in optimal stochastic control, *Systems Control Lett.*, v.39, Programming. 79–86.

[8] Boryczka, M., Wiezorek, W., (2003). Soving approximation problems using ant colony programming, *Proceedings of AI-METH*, Programming. 55–60.

[9] Boryczka, M., (2005). Eliminating introns in ant colony programming, *Fundamenta Informaticae*, v.68 , Programming. 1–19.

[10] Bucci, F., Pandolfi, L., (2000). The regulator problem with indefinite quadratic cost for boundary control systems: The finite horizon case. *Systems Control Lett.*, v.39, Programming. 79–86.

[11] Campbell, S. L. (1980). *Singular systems of differential equations*, Pitman, Marshfield MA.

[12] Campbell, S. L. (1982). *Singular systems of differential equations II*, Pitman, Marshfield MA.

[13] Choi, C. H., (1990). A survey of numerical methods for solving matrix Riccati differential equation, *Proceedings of Southeastcon*, Programming. 696–700.

[14] Chen, S. P., Li, X. J., Zho, X. Y., (1998). Stochastic linear quadratic regulators with indefinite control weight costs, *SIAM J. Control Optim.*, v.36, Programming. 1685–1702.

[15] Davis, M. H. A. (1997). *Linear estimation and stochastic control*, Chapman and Hall, London.

[16] Dorigo, M., Maniezzo, V., Colorni, A. (1991). *Positive feedback as a search strategy*, Technical report politechnico di Milano, Italy.

[17] Dorigo, M., Maniezzo, V., Colorni, A., (1996). The Ant system : Optimization by a colony of cooperating agents, *IEEE Trans. Systems, Man and Cybernetics - Part B*, v.26, Programming. 29–41.

[18] Lewis, F. L., (1986). A Survey of linear singular systems, *Circ. Syst. Sig. Proc.*, v.5 , Programming. 3–36.

[19] Roux, O., Fonlupt, C., (2000). Ant programming: or how to use ants for automatic programming, *Proceedings of ANTS*, Programming. 121–129.

[20] Tanaka, K., Iwazaki, M., (2001). Switching control of an R/C hovercraft: stabilization and smooth switching, *IEEE Trans. Fuzzy Systems*, v.31, Programming. 853–863.

[21] Vincent Antony Kumar, A., Balasubramaniam, P., (2007) Optimal control for linear singular system using genetic programming, *Appl. Math. Comput.*, v.192, Programming. 78–89.

[22] Wang, L. X., (1998). Stable and optimal fuzzy control of linear systems, *IEEE Trans. Fuzzy Systems.*, v.6, Programming. 137–143.

[23] Wilson, E., Hölldobler, B. (1990). The Ants, Springer-Verlag.

[24] Wonham, W. M., (1968). On a matrix Riccati equation of stochastic control, *SIAM J. Control Optim.*, v.6, Programming. 681–697.

[25] Wu, S. J., Lin, C. T., (2000). Optimal fuzzy controller design: local concept approach, *IEEE Trans. Fuzzy Systems*, v.8, Programming. 171–185.

[26] Wu, S. J., Lin, C. T., (2000). Optimal fuzzy controller design in continuous fuzzy system: global concept approach, *IEEE Trans. Fuzzy Systems*, v.8, Programming. 713–729.

[27] Wu, S. J., Chiang, H. H., Lin, H. T., Lee, T. T., (2005). Neural network based Optimal fuzzy controller design for nonlinear systems, *Fuzzy Set and Systems*, v.154, Programming. 182–207.

[28] Zadeh, L. A., (1975). The concept of a linguistic variable and its application to approximate reasoning, *Inform. Sciences*, v.Part I:(8), Programming. 199–249, v.Part II:(8), Programming. 301–357, v.Part III:(9), Programming. 43–80.

[29] Zhu, J., Li, K., (2003). An iterative method for solving stochastic Riccati differential equations for the stochastic LQR problem, *Optim. Methods Softw.*, v.18, Programming. 721–732.