Optimal control for stochastic linear quadratic singular neuro Takagi–Sugeno fuzzy system with singular cost using genetic programming

N. Kumaresan *, Kuru Ratnavelu
Institute of Mathematical Sciences, Faculty of Science, University of Malaya, Kuala Lumpur 50603, Malaysia

A R T I C L E   I N F O
Article history:
Received 17 January 2013
Received in revised form 1 April 2014
Accepted 3 August 2014
Available online xxx

A M S ( M O S ) subject classifications:
49 K 45
68 N 19
92 D 10
93 B 52
94 D 05
03 E 72
93 E 20

Keywords:
Differential algebraic equation
Genetic programming
Matrix Riccati differential equation
Runge–Kutta method
Optimal control and stochastic linear
quadratic singular neuro Takagi-Sugeno fuzzy system

A B S T R A C T
In this paper, optimal control for stochastic linear quadratic singular neuro Takagi–Sugeno (T-S) fuzzy system with singular cost is obtained using genetic programming (GP). To obtain the optimal control, the solution of matrix Riccati differential equation (MRDE) is computed by solving differential algebraic equation (DAE) using a novel and nontraditional GP approach. The obtained solution in this method is equivalent or very close to the exact solution of the problem. Accuracy of the solution computed by GP approach to the problem is qualitatively better. The solution of this novel method is compared with the traditional Runge–Kutta (RK) method. A numerical example is presented to illustrate the proposed method.

© 2014 Elsevier B.V. All rights reserved.

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 to produce fuzzy conclusions or fuzzy aggregations and finally the defuzzification block converts these fuzzy conclusions into crisp outputs. The fuzzy system with singleton fuzzifier, product inference engine, center average defuzzifier and Gaussian membership functions is called a standard fuzzy system [1].

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 [2].

Neural networks or simply neural nets are computing systems, which can be trained to learn a complex relationship between two or many variables or data sets. Having the structures similar to their biological counterparts, neural networks (NNs) are representational and computational models. NNs are processing information in a parallel distributed fashion composed of interconnecting simple processing nodes [3]. Neural net techniques have been successfully applied in various fields such as function approximation, signal processing and adaptive (or) learning control for nonlinear systems. Using neural networks, a variety of offline learning control algorithms have been developed for nonlinear systems [4,5].

* Tel.: +60 37967 4126; fax: +60 37967 4143.
E-mail address: drnk2008@um.edu.my (N. Kumaresan).

http://dx.doi.org/10.1016/j.asoc.2014.08.006
1568-4946/© 2014 Elsevier B.V. All rights reserved.
Neuro fuzzy systems are a combination of two popular soft computing techniques: neural networks and fuzzy systems. Also NNs have the capability to learn from examples, yet the learned knowledge can not be represented explicitly. On the other hand, knowledge in fuzzy systems is represented via explicit fuzzy if then rules, yet fuzzy systems have no learning capability. Neuro fuzzy system is a hybrid approach in which a fuzzy system is trained using techniques similar to those applied to neural networks. One of the first neuro fuzzy systems was the adaptive network-based fuzzy inference system (ANFIS) [6]. ANFIS represents a Takagi–Sugeno fuzzy system as a multilayer feedforward network which can be trained via backpropagation algorithm. Neural networks and fuzzy systems can be combined to join its advantages and to cure its individual illness. Neural networks introduce its computational characteristics of learning in the fuzzy systems and receive from them the interpretation and clarity of systems representation. Thus, the disadvantages of the fuzzy systems are compensated by the capacities of the neural networks. Hayashi et al. [7] showed that a feedforward neural network could approximate any fuzzy rule based system and any feedforward neural network may be approximated by a rule based fuzzy inference system [8]. Fusion of artificial neural networks and fuzzy inference systems have attracted the growing interest of researchers in various scientific and engineering areas due to the growing need of adaptive intelligent systems to solve the real world problems [9–11].

Genetic programming is an evolutionary algorithm that attempts to evolve solution to the given problem by using concepts taken from naturally occurring evolving process. The technique is based on the evolution of a large number of candidate solutions through genetic operations such as reproduction, crossover and mutation. It is based upon the genetic algorithm (GA) [12], which exploits the process of natural selection based on a fitness measure to breed a population of trial solutions that improve over time. While GA usually operates on (coded) strings of numbers, GP uses the principles and ideas from biological evolution to guide the computer to acquire desired solution. The search space is too large to attempt a brute force search, the method must be utilized to reduce the number of examined solutions. In this search, initially the population looks a bit like a cloud of randomly selected points, but that generation after generation it moves in the search space following a well defined trajectory. The generation is achieved with the help of grammatical evolution, because grammatical evolution can produce grammars in an arbitrary language. The genetic operations are faster and also because it is more convenient to symbolically differentiate mathematical expression. The code production is performed using a mapping process governed by grammar expressed in Backus Naur Form (BNF) [13]. In analogy to nature, the potential solution is an individual in some collection or population of potential solutions. The individuals who are stronger, meaning higher ranked according to fitness function, will be used to determine the next collection of potential solution. A new generation will be arisen by employing analogs of reproduction, crossover and mutation.

This means that GP has advantages over other algorithms as it can perform optimization at a structural level. This enabled Koza [14] to demonstrate the application of GP algorithm to a number of problem domains, including regression, control and classification. Research in this area has grown rapidly and encompassed a wide range of problems. GP techniques have been successfully applied in various engineering fields like signal processing [15], electrical circuit design [16], scheduling [17], process controller evolution [18] and modelling of both steady-state and dynamic processes [19].

In this paper, optimal control of a stochastic linear quadratic singular neuro T–S fuzzy system is obtained using genetic programming. The linear T–S fuzzy system is the most popular fuzzy model due to its 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 [20]; the optimal fuzzy control technique is used to minimize the performance index from local-concept or global-concept approaches [21,22].

Stochastic linear quadratic regulator (LQR) problems have been studied by many researchers [23,24]. Chen et al. [25] 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 [26] 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. Recently the conventional Riccati equation has been analyzed using neural network approach and genetic programming approach see [27–34] to minimize the error. A variety of numerical algorithms [35] have been developed for solving the algebraic Riccati equation.

Singualr 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 [36,37]. As the theory of optimal control of linear systems with quadratic performance criteria is well developed, the results are most complete and close to use in many practical designing problems. The theory of the quadratic cost control problem has been treated as a more interesting problem. The optimal feedback with minimum cost control has been characterized by the solution of a Riccati equation. Da Prato and Ichikawa [38] showed that the optimal feedback control and the minimum cost are characterized by the solution of a Riccati equation. Solving the Matrix Riccati differential equation (MRDE) is the central issue in optimal control theory.

Although parallel algorithms can compute the solutions faster than sequential algorithms, there have been no report on genetic programming solutions for MRDE [34]. This paper focuses upon the implementation of genetic programming approach for solving MRDE in order to get the optimal solution.

This paper is organized as follows. In Section 2, the statement of the problem is given. In Section 3 and 4, solution of the MRDE is presented. In Section 5, numerical example is discussed to illustrate the proposed method. The final conclusion section demonstrates the efficiency of the method.

2. Statement of the problem

In this section, a class of adaptive networks is proposed. The proposed architecture is referred to as ANFIS. For simplicity, the fuzzy inference system has two inputs $x_1$ and $x_2$ and one output $z = f(x_1, x_2)$. Suppose that the rule base contains two fuzzy if-then rules of Takagi and Sugeno’s type [39]:

Rule 1: If $x_1$ is $A_1$ and $x_2$ is $B_1$, then $f_1 = p_1x_1 + q_1x_2 + r_1$.

Rule 2: If $x_1$ is $A_2$ and $x_2$ is $B_2$, then $f_2 = p_2x_1 + q_2x_2 + r_2$. {Rule 1 and Rule 2 combined over the whole domain of operation}

Please cite this article in press as: N. Kumaresan, K. Ratnavelu, Optimal control for stochastic linear quadratic singular neuro Takagi–Sugeno fuzzy system with singular cost using genetic programming, Appl. Soft Comput. J. (2014), http://dx.doi.org/10.1016/j.asoc.2014.08.006
Layer 2. Every node in this layer is a circle node labeled $\Pi$ which multiplies the incoming signals and sends the product out. For instance,

$$w_i = \mu_A(x_1) \times \mu_B(x_2), \quad i = 1, 2.$$  

Each node output represents the firing strength of a rule.

Layer 3. Every node in this layer is a circle node labeled $N$. The $i$th node calculates the ratio of the $i$th rule’s firing strength to the sum of all rules’ firing strengths:

$$\overline{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2.$$  

For convenience, outputs of this layer will be called normalized firing strengths.

Layer 4. Every node in this layer is a square node with a node function

$$O_i^3 = \overline{w}_i f_i = \overline{w}_i (p_i x_1 + q_i x_2 + r_i).$$

where $\overline{w}_i$ is the output of layer 3 and $(p_i, q_i, r_i)$ is the parameter set. Parameters in this layer will be referred to as consequent parameters.

Layer 5. The single node in this layer is a circle node labeled $\Sigma$ that computes the overall output as the summation of all incoming signals i.e.,

$$O_i^5 = \text{overall output} = \sum_i \overline{w}_i f_i = \sum_i \frac{w_i f_i}{w_1 + w_2}.$$  

Thus an adaptive network is constructed equivalent to a type-3 fuzzy inference system.

Hence the input variables $x_i$ are fuzzified as fuzzy variables whose corresponding term sets $T_{xi}$ have Gaussian membership function with mean $m_{ji}$ and the standard deviation $\sigma_{ji}$, the corresponding output for the neural network [40] is

$$Sx(t) = \sum (A_i x(t) + B_i u(t)) dt + D_i u(t) dW(t), \quad i = 1, \ldots, r.$$  

In other words, the proposed neuro fuzzy network is in fact a neural-based linear T-S fuzzy modelling structure. Using neural learning technique, this structure will proceed the structure and parameter learning concurrently and generate the following linear T-S fuzzy system:

$$R^L: \text{if } x_j \in T_{ji}(m_{ji}, \sigma_{ji}), i = 1, \ldots, r \text{ and } j = 1, \ldots, n, \text{then}$$

$$Sx(t) = \sum (A_i x(t) + B_i u(t)) dt + D_i u(t) dW(t), \quad x(0) = x_0,$$

where $R^L$ denotes the ith rule of the fuzzy model, $m_{ji}$ and $\sigma_{ji}$ are the mean and standard deviation of the Gaussian membership function, $x(t) \in R^n$ 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 $A_j \in R^{nxn}$, $B_j \in R^{nxm}$ and $D_j \in R^{nxm}$ are known as coefficient matrices, $x_0$ is given initial state vector, $t_f$ is the final time and $m \leq n$ and $Sx(t)$ denotes $F_i x(t)$ for the continuous case.

Neuro fuzzy optimal control system. Consider the stochastic linear dynamical singular T-S fuzzy system [41] that can be expressed in the form:

$$R^L: \text{if } x_j \in T_{ji}(m_{ji}, \sigma_{ji}), i = 1, \ldots, r \text{ and } j = 1, \ldots, n, \text{then}$$

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

where

$$R^L: \text{if } x_j \in T_{ji}(m_{ji}, \sigma_{ji}), i = 1, \ldots, r \text{ and } j = 1, \ldots, n, \text{then}$$

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

for any continuous controller $u(t)$ subject to the constraint $u(t) \in U_{ad}$. Then the optimal control problem is to find a controller $u^*(t)$ that minimizes

$$J(u) = \int_0^{t_f} c(x(t), u(t)) dt$$

subject to

$$x(t) = [A x(t) + B u(t)] dt + D u(t) dW(t), \quad x(0) = x_0,$$

where $c$ is a cost function.
In order to minimize both the state and control signals of the feedback control system, a quadratic performance index is usually minimized:

\[ J = \frac{1}{2} x^T(t_f)P_0x(t_f) + \frac{1}{2} \int_0^{t_f} \left[ x^T(t)Qx(t) + u^T(t)Ru(t) \right] dt, \]

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) weighing matrices for \( x(t) \), \( R \in \mathbb{R}^{n \times n} \) is a singular weighing matrix for \( u(t) \). It will be assumed that \( |s_i| - A_i | \neq 0 \) 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 Eq. (1), where

\[ \lambda(t) = K_i(t)F_i x(t), \]

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

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

\[
\begin{align*}
F_i^T K_i(t) F_i + F_i^T K_i(t) A_i + A_i^T K_i(t) F_i + Q \\
- F_i^T K_i(t) B_i (R + D_i^T K_i(t) D_i)^{-1} B_i^T K_i(t) F_i = 0
\end{align*}
\]

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

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. Runge–Kutta approach to MRDE

Consider the DAE of (2) for each rule of the fuzzy model

\[
\begin{align*}
\dot{k}_{ij}(t) &= \phi_{ij}(k_{ij}(t)), \quad k_{ij}(t) = A_{ij} \quad (i, j = 1, 2, \ldots, n - 1) \quad (3) \\
k_{1n}(t) &= \psi(k_{1n}(t)), \quad k_{1n}(t) = A_{1n},
\end{align*}
\]

where \( k_{ij} \) and \( k_{1n} \) are the elements of the matrix \( K_i(t) \).

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, 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 a system of nonlinear differential equation by differentiating the algebraic equation one time 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 \( K_i(t) \) is symmetric and the system is singular, \( k_{12} = k_{21} \) and \( k_{22} = 0 \). 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.

\[
\begin{align*}
k_{11}(i + 1) &= k_{11}(i) + \frac{1}{6} (k_{11} + 2k_{21} + 2k_3 + k_4) \\
k_{12}(i + 1) &= k_{12}(i) + \frac{1}{6} (k_{12} + 2k_{22} + 2k_3 + k_4),
\end{align*}
\]

where

\[
\begin{align*}
k_1 &= h \cdot \phi_{11} (k_{11}, k_{12}) \\
l_1 &= h \cdot \phi_{12} (k_{11}, k_{12}) \\
k_2 &= h \cdot \phi_{11} \left( k_{11} + \frac{k_1}{2}, k_{12} + \frac{l_1}{2} \right) \\
l_2 &= h \cdot \phi_{12} \left( k_{11} + \frac{k_1}{2}, k_{12} + \frac{l_1}{2} \right) \\
k_3 &= h \cdot \phi_{11} \left( k_{11} + \frac{k_2}{2}, k_{12} + \frac{l_2}{2} \right) \\
l_3 &= h \cdot \phi_{12} \left( k_{11} + \frac{k_2}{2}, k_{12} + \frac{l_2}{2} \right) \\
k_4 &= h \cdot \phi_{11} \left( k_{11} + k_{12} + k_{13} + l_3 \right) \\
l_4 &= h \cdot \phi_{12} \left( k_{11} + k_{12} + k_{13} + l_3 \right).
\end{align*}
\]

In the similar way, the original system (3) can be solved for \( n^2 \) first order ODEs.

4. Genetic programming method

In this approach, GP is used to obtain a set of expressions. If the required number expressions satisfy the fitness function, it will be the optimal solution of MRDE. The scheme of computing optimal solution is illustrated in Fig. 3.
The grammar of the proposed method.

**Table 1**

| S ::= <exp> | (0) |
| <exp>::= <exp> <op> <exp> | (0) |
| <op>::= + | (0) |
| - | (1) |
| * | (2) |
| / | (3) |
| <fun>::= sin | (0) |
| cos | (1) |
| exp | (2) |
| log | (3) |
| <digit>::= 0 | (0) |
| 1 | (1) |
| 2 | (2) |
| 3 | (3) |
| 4 | (4) |
| 5 | (5) |
| 6 | (6) |
| 7 | (7) |
| 8 | (8) |
| 9 | (9) |

4.1. Initialization of the population

The first step is to initialize the population. An initial population of the desired size is generated randomly. The length of each chromosome is to be set according to the nature of the problem. Each program or individual in the population is generally represented as a parse tree composed of function and data/terminals appropriate to the problem domain.

4.2. Grammatical evolution

Grammatical evolution is an evolutionary algorithm that can produce code in any programming language. The algorithm begins from the start symbol of the grammar and gradually creates the program string, by replacing non-terminal symbol with right-hand of the selected production rule, first read an element from the chromosome (with value V) and compute Rule = V mod NR, where NR is the number of rules for the specific non-terminal symbol which is shown in Table 1. The symbol S in the grammar denotes the start symbol of the grammar. For example, suppose we have the chromosome x = 7, 2, 10, 4, 4, 2, 11, 16, 30, 5. The method of producing a valid expression is shown in Table 2. The rule = V mod NR is applied in each row of Table 2. In first line V = 7, NR = 7 (Number of elements in <exp> group in Table 1). The next line <exp><op><exp> is obtained from Table 1 for the result zero of the first line. Similarly the remaining rows of the Table 2 can be found. The function which corresponds to the chromosome is exp(x) + log(exp(y)), for details see (42, 43).

**Table 2**

<table>
<thead>
<tr>
<th>String</th>
<th>Chromosome</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;exp&gt;</td>
<td>7, 2, 10, 4, 4, 2, 11, 16, 30, 5</td>
<td>7 mod 7 = 0</td>
</tr>
<tr>
<td>&lt;exp&gt;&lt;op&gt;&lt;exp&gt;</td>
<td>2, 10, 14, 4, 2, 11, 16, 30, 5</td>
<td>2 mod 7 = 2</td>
</tr>
<tr>
<td>&lt;fun&gt;&lt;exp&gt;&lt;op&gt;&lt;exp&gt;</td>
<td>10, 4, 4, 2, 11, 16, 30, 5</td>
<td>10 mod 4 = 2</td>
</tr>
<tr>
<td>Exp &lt;exp&gt;&lt;op&gt;&lt;exp&gt;</td>
<td>4, 2, 11, 16, 30, 5</td>
<td>4 mod 7 = 4</td>
</tr>
<tr>
<td>Exp &lt;op&gt;&lt;exp&gt;</td>
<td>4, 2, 11, 16, 30, 5</td>
<td>4 mod 4 = 0</td>
</tr>
<tr>
<td>Exp &lt;op&gt;&lt;exp&gt;</td>
<td>2, 11, 16, 30, 5</td>
<td>2 mod 7 = 2</td>
</tr>
<tr>
<td>Exp&lt;op&gt;fun&lt;exp&gt;</td>
<td>11, 16, 30, 5</td>
<td>11 mod 4 = 3</td>
</tr>
<tr>
<td>Exp&lt;op&gt;[log]&lt;exp&gt;</td>
<td>16, 30, 5</td>
<td>16 mod 7 = 2</td>
</tr>
<tr>
<td>Exp&lt;op&gt;log&lt;fun&gt;&lt;exp&gt;</td>
<td>30, 5</td>
<td>30 mod 4 = 2</td>
</tr>
<tr>
<td>Exp&lt;op&gt;log&lt;Exp&gt;&lt;exp&gt;</td>
<td>5</td>
<td>5 mod 7 = 5</td>
</tr>
</tbody>
</table>

**Table 3**

<table>
<thead>
<tr>
<th>Population</th>
<th>Fitness</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>40</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
</tr>
</tbody>
</table>
number is generated for each chromosome. If the random number is less than the crossover probability, then the chromosome is chosen for crossover operation. After crossover operations, the new child chromosomes replaces the old parent chromosomes.

**Mutation.** Mutation adds new information in a random way to the genetic search process and ultimately helps to avoid getting trapped at local optima. It is an operator that introduces diversity in the population whenever the population tends to become homogeneous due to repeated use of reproduction and crossover operators. Mutation may cause the chromosomes of individuals to be different from those of their parent individuals. In the mutation process, for every element in a chromosome a random number in the range [0, 1] is chosen. If the number is less than or equal to the mutation rate the corresponding element is changed randomly, otherwise it remains intact.

4.4. Termination control

In each generation, a set of expressions are generated by the chromosomes. If an expression minimizes the fitness function $E_i$ (4) to zero or very close to zero and satisfies the initial condition, the process may be stopped; otherwise, the GP approach must be continued.

4.5. Genetic programming algorithm

The algorithm has the following steps:

- step 1. Initialize random population.
- step 2. Create valid function using grammar.
- step 3. Evaluate fitness value of the chromosome.
- step 4. If Fitness tends to zero, stop the procedure. Otherwise proceed to next step.
- step 5. Generate new population using Genetic operations, Go to step 2.

5. Numerical examples

**Example 1.** Consider the optimal control problem:

$$J = E \left\{ \frac{1}{2} x^T(t_f)K_1x(t_f) + \frac{1}{2} \int_0^{t_f} [x^T(t)Qx(t) + u^T(t)Ru(t)]dt \right\}$$

subject to the stochastic linear quadratic singular neuro T-S fuzzy system $R^*$: If $x_i$ is $T_i(u(m_1, \sigma_i), i = 1, 2, \ldots, N)$, then

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

where

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

The numerical implementation could be adapted by taking $k_f = 2$ for solving the related MRDE of the above stochastic linear singular fuzzy system with the matrix $A_i$. The appropriate matrices are substituted in Eq. (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 $K_1(t)$ is free and let $k_{22} = 0$. Then the optimal control of the system can be found out by the solution of MRDE.

5.1. Solution obtained using genetic programming

To solve the MRDE, each chromosome is split uniformly in 36 parts and Population size is 100. In the computation process, the parameters are taken as 0.01 for replication rate, 0.9 for the crossover probability and 0.05 for the mutation rate.

After 350 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 and it is given below:

$$Expressions: \quad k_{11} = \frac{18}{7} e^{(7/4(2-t))} - \frac{4}{7}, \quad k_{12} = \frac{20}{7} e^{(7/4(2-t))} - \frac{2}{7}.$$

Please cite this article in press as: N. Kumaresan, K. Ratnavelu, Optimal control for stochastic linear quadratic singular neuro Takagi–Sugeno fuzzy system with singular cost using genetic programming, Appl. Soft Comput. J. (2014), http://dx.doi.org/10.1016/j.asoc.2014.08.006
The numerical solutions of MRDE are calculated and displayed in Table 4 using the GP. Since this problem is having an explicit solution, the GP solution is equivalent to exact solution of the DAE. GP solution curves are shown in Figs. 5 and 6. In Fig. 5, the values of $t$ are noted in the X-axis and the values of $k_{11}$ are placed in the Y-axis. Similarly in Fig. 6, the values of $t$ and $k_{12}$ are in X-axis and Y-axis respectively. The parse trees for the solutions are given in Figs. 7 and 8.

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

**Example 2.** Consider the optimal control problem:

Minimize

$$J = E \left\{ \frac{1}{2} x^T(t_f)F x(t_f) + \frac{1}{2} \int_{0}^{t_f} [x^T(t)Q x(t) + u^T(t)R u(t)] dt \right\}$$

subject to the stochastic linear quadratic singular T-S fuzzy system

$R_1$: If $x_1$ is $T_1(m_{11}, \sigma_{11})$, $i = 1, 2$, then

$$F dx(t) = [A_1 x(t) + B_1 u(t)] dt + D_1 u(t) dW(t), \quad x(0) = x_0, \quad t \in [0, t_f],$$

where

$$S = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad F_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad A_1 = \begin{bmatrix} -1 & 1 & 1 \\ 0 & -2 & 1 \\ 0 & 0 & -3 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & -4 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad R = 0, \quad Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad D_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

The numerical implementation could be adapted by taking $t_f = 2$ for solving the related MRDE of the above stochastic linear singular fuzzy system with the matrix $A_1$. The appropriate matrices are substituted in Eq. (2), the MRDE is transformed into DAE in $k_{11}, k_{12}$ and $k_{13}$. In this problem, the value of $k_{22}, k_{32}, k_{23}$ and $k_{33}$ of the symmetric matrix $K(t)$ are free and let $k_{12} = 0, k_{23} = 0$ and $k_{33} = 0$. Then the optimal control of the system can be found out by the solution of MRDE.

### 5.2. Solution obtained using genetic programming

To solve the MRDE, each chromosome is split uniformly in 36 parts and Population size is 100. In the computation process, the parameters are taken as 0.01 for replication rate, 0.9 for the crossover probability and 0.05 for the mutation rate.

After 500 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 and it is given below:

**Expressions:** $k_{11} = \frac{14}{9} e^{(9/4)(-2)} - \frac{4}{5}$, $k_{12} = \frac{2}{9} e^{(9/4)(-2)} - \frac{2}{5}$, $k_{13} = \frac{2}{9} e^{(9/4)(-2)} - \frac{2}{5}$

The numerical solutions of MRDE are calculated and displayed in the Table 5 using the GP. Since this problem is having an explicit solution, the GP solution is equivalent to exact solution of the DAE. GP solution curves are shown in Figs. 9 and 10. In Fig. 9, the values of $t$ are noted in the X-axis and the values of $k_{11}$ are placed in the Y-axis. Similarly in Fig. 10, the values of $t$ and $k_{12}$ and $k_{13}$ are in X-axis and Y-axis respectively. The parse trees for the solutions are given in Figs. 11 and 12.
Similarly the solution of the above system with the matrix $A_2$ can be found out using genetic programming.

6. Conclusion

The optimal control for the stochastic linear singular neuro T-S fuzzy system with singular cost can be found by GP approach

The optimal control, the solution of MRDE is computed by solving differential algebraic equation (DAE) using a novel and nontraditional GP approach. The obtained solution in this method is equivalent to the exact solution of the problem. Accuracy of the solution computed by GP approach to the problem is qualitatively better when it is compared with RK solution. A numerical example is given to illustrate the derived results.

Acknowledgements

The author is very much thankful to the referees for their valuable comments and suggestions to improve this manuscript. The funding of this work by the HIR grant (account no: UM.C/625/1/HIR/MOHE /SC/13) and UMRG grant (account no: RG099/10AFR) is gratefully acknowledged.

References