Research and Simulation of Compartmental Network Based on Matlab

Caixia Wang, Jipeng Liu, Minghua Liu

College of Electrical Engineering, Northwest University for Nationalities, Lanzhou, 730124, China

Keywords: Compartmental Network; Multi-Agent System; DAEs

Abstract. Compartmental network is a specific type of dynamical multi-agent system, which is comprised of special vertices called compartments, interconnected through a network, and each vertex contains some substance or information. Compartmental network has been extensively studied in various fields, especially engineering systems. The stability of multi-agent systems is very important, and has been noticed by more scholars. The technique of solving descriptor compartmental networks can be considered as a class of complex DAEs. According to the difficulties in simulation of singular dynamic multi-agent systems, this paper describes a specific type of dynamical multi-agent system, establishes the model of the system, analysis the DAEs, and proposes a method to solve the problems, and then a numerical instance has been exhibited in the paper, the results of the simulation with MATLAB.

Introduction

Compartmental network [1] can be regarded as a specific type of dynamical multi-agent system, which is comprised of special vertices called compartments, inter-connected through a network, and each vertex contains some substance or information. The neighbouring compartments in the network can dynamically exchange the substance or information with each other. Many systems that have been extensively studied in various fields such as biology, chemistry, economics, and engineering can be treated as compartmental systems. For instance, in engineering, some of the artificial neuron networks [2] and sensor networks belong to compartmental networks [3].

In recent years, scholars start to notice the stability problems for descriptor multi-agent systems. Descriptor systems are also called singular systems. A descriptor model is more general and precise than a normal model to depict a dynamical physical system, especially as certain algebraic constraints exist among the state variables [4] or as the system dynamics include components with different temporal scales [5]. Xi et al. [6-7] early paid attention to the consensus problems of descriptor multi-agent systems, mostly with LMI methods. Yang et al. [8] analysed the consensus conditions for singular multi-agent systems with output feedback protocols. Zhou et al. [9] concerned the stability of a class of switching descriptor systems.

Based on the scholars' study, there are still three problems when singular dynamic multi-agent systems are simulated: 1) Numerical simulation of DAEs is very difficult. The initial-values need to be matched, because not all initial-values are suitable for DAEs. When initial values are not suitable for DAEs, the solver will guess initial-values automatically. 2) There is a technical limitation for Matlab to dispose a DAE, witch be solved by index 1. Index is a notion used in the theory of DAEs. 3) The numerical simulation of complex network is dif-ficult, because the dimensions of complex networks are usually very large. So when the simulation of singular complex networks is to be dealt by scholars, they often transform a singular complex network into a series of equivalent ordinary systems and simulate the system with conventional techniques [10]. Usually, it is troublesome to perform such a transformation.

In order to break the dilemmas, this paper offered a simple method to solve the above-mentioned problems and break the limitation of index 1. Consequently, this paper should have referential significance for solving DAEs numerically.

Ode15s is a specific command of Matlab which can resolve initial value problems for ordinary differential equations. It solves the equations form y'=f(t,y) or problems that involve a mass matrix, M(t,y) y'=f(t,y). Ode15s can settle problems with a mass matrix that is singular, i.e., differential-algebraic equations (DAEs).

This paper focuses on the application of ode15s in computing the numerical solution of descriptor compartmental networks. In this paper the technique of solving descriptor compartmental networks as a class of complex DAEs is detailed described, and a numerical simulation is offered to verify the effectiveness.

The organization of the remaining part of this paper is as follows. Section 2 will describe the descriptor compartmental network model. Section 3 will analyse DAEs. Section 4 will show numerical examples. Finally, Section 5 will be the conclusion.

Model formulation

A compartmental network being of nth order implies that each vertex may contain n different types of substance or information, where any type can be transformed into other types. The network's being undirected implicates that if some substance flows from vertex i to j, then both the quantities of substance in the two vertices will simultaneously alter. When there exist certain internal algebraic constrains among various quantities, the compartmental network should be depicted by a descriptor model:

A compartmental network can be regarded as a specific type of dynamical multi-agent system. It is supposed to be composed of m vertices indexed from 1 to m, each of mth order. The state of vertex i is denoted by $x_i = [x_{i1}, x_{i2}, \cdots, x_{in}]^T \in R^n$ which represents the quantity of substance or information on the vertex. The communication network among vertices is represented by a graph topology G of order m. The arc weight of G between vertex i and j is denoted by $w_{ij} \geq 0$, which can be regarded as the strength of communication link. If $w_{ij} \geq 0$, then vertex j is vertex i's neighbor. The graph can be denoted by its adjacency matrix W:

$$G:W=\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ w_{m1} & w_{m2} & \cdots & w_{mm} \end{bmatrix}$$

The dynamics of the LTI descriptor compartmental network that will be concerned is described by:

$$\begin{cases} E\dot{x}_1 = F\sum_{j=1}^{m} w_{1j}(x_j - x_1) \\ E\dot{x}_2 = F\sum_{j=1}^{m} w_{2j}(x_j - x_2) \\ \vdots \\ E\dot{x}_m = F\sum_{j=1}^{m} w_{mj}(x_j - x_m) \end{cases} \tag{1}$$

The dynamics of the overall compartmental network (1) can be described by the matrix state equation below:

$$E\dot{X} = -FXL^T \tag{2}$$

where $L = L(G)$ is the Laplacian matrix of the graph topology G.

Let $x = [x_1^T \quad x_2^T \quad \cdots \quad x_m^T]^T$ be the stack state vector of the system, and the dynamics is depicted by

$$(I_N \otimes E)\dot{x} = -(L \otimes F)x \tag{3}$$

which is the vector form counterpart of (2).

Lemma 1

The Laplacian matrix L of a directed graph G has exactly a single zero eigenvalue $\lambda_1 = 0$ $\lambda_1 = 0$ if G has a spanning tree, with the corresponding eigenvector $\phi = [1 \quad 1 \quad \cdots \quad 1]^T$. Meanwhile, all the other eigenvalues $\lambda_2, \cdots, \lambda_N$ locate in the open right half plane.

Lemma 2

For two given matrices A and B, let $\lambda(A)$ be an eigenvalue of A with corresponding eigenvector

$\gamma(A)$ and $\lambda(B)$ be an eigenvalue of B with eigenvector $\gamma(B)$, then $\lambda(A)\lambda(B)$ is an eigenvalue of $A \otimes B$ with corresponding eigenvector $\gamma(A) \otimes \gamma(B)$.

Simulating a dynamic compartmental network means solving a high-dimensional DAE. When processing a DAE, scholars may encounter some problems, such as initial-value matching and limitation of index 1. For initial-value matching, DAEs have solutions only when the initial value of state variable y_0 and the initial value of first order derivative of state variable y_{p0} are consistent, and y_{p0} need satisfy the condition of $M(t_0, y_0)y_{p0} = f(t_0, y_0)$. If and are not consistent, the solver treats them as guesses, and continues to handle the problem. Another problem is the limitation of index 1. When disposing a DAE, scholars may get the error message as follows: "This DAE appears to be of index greater than 1". Index can be used for measuring the distance from a DAE to its related ODE. The index is a nonnegative integer that provides useful information about the mathematical structure and potential complications in the analysis and the numerical solution of the DAE. In general, the higher the index of a DAE, the more difficulties one can expect for its numerical solution. So if a DAE is to be dealt, usually it must be of index 1. Therefore, those problems need to be handled.

Analysis of differential algebraic equations index 1 problem

Differential-algebraic Equations and Matlab Commands

Differential-Algebraic Equations is a system which is composed of differential equations and algebraic equations without derivative. It is hard to find exact solutions for DAEs. This situation is similar to partial differential equation. But approximate solution can be obtained by using Matlab solvers, such as ode15s and ode23t. ode15s and ode23t can directly resolve ODEs with a mass matrix that is singular, i.e., differential-algebraic equations (DAEs).

T[T,Y] = solver (odefun, tspan, y_0, options): handles as above with default integration parameters replaced by property values specified in options, an argument created with the odeset function. Commonly used properties include a scalar relative error tolerance RelTol(1e-3 by default) and a vector of absolute error tolerances AbsTol (all components are 1e-6 by default). If certain components of the solution must be nonnegative, use the odeset function to set the Non-Negative property to the indices of these components. Where the solver is one of ode45, ode23, ode113, ode15s, ode23s, ode23t, or ode23tb.

The solvers of the ODE suite can solve problems of the form $M(t, y)y' = f(t, y)$ with time and state-dependent mass matrix M.

If a problem has a mass matrix, create a function $M = MASS(t, y)$, that returns the value of the mass matrix, and use odeset to set the Mass property to @MASS. If the mass matrix is constant, the matrix should be used as the value of the Mass property.

If the mass matrix M is singular, then $M(t, y)y' = f(t, y)$ is a system of differential algebraic equations. ode15s and ode23t solvers can solve DAEs of index 1 provided that y_0 is sufficiently close to being consistent. If there is a mass matrix, odeset can be used to set the MassSingular property to 'yes', 'no', or 'maybe'. The default value of 'maybe' causes the solver to test whether the problem is a DAE. The initial value of first order derivative of state variable y_{p0} can be provided as the value of the InitialSlope property. The default is the zero vector. If a problem is a DAE, and y_0 and y_{p0} are not consistent, the solver treats them as guesses, attempts to compute consistent values that are close to the guesses, and continues to solve the problem. When solving DAEs, it is very advantageous to formulate the problem so that M is a diagonal matrix.

Analysis of index 1 problem

It is hard for some DAEs to satisfy the condition of index 1, especially for higher order DAEs. When settling DAEs, ode15s and ode23t solvers can solve DAEs of the form $M(t, y)y' = f(t, y)$, where $M(t, y)$ is singular. A singular $M(t, y)$ implies that a DAE is dealt. If so, the DAE is required to be of index 1.The index 1 condition is satisfied when the matrix

$$(M + \lambda \partial f / \partial y) \tag{4}$$

is nonsingular, for all non-zero λ [13]. In this paper, the model is (3). According to matrix (4) and model (3), the matrix can be showed as follow:

$$(I_N \otimes E) - \lambda (L \otimes F) \tag{5}$$

Proposition 1

When the zero eigenvector of is the same E as the zero eigenvector of F, matrix (5) is singular.

Proof: Suppose $(I_N \otimes E)\alpha_1 = 0$ and $(L \otimes F)\alpha_2 = 0$, with as the zero eigenvector of $(I_N \otimes E)$ and α_2 as the zero eigenvector of . α is one of the eigenvector of I_N. β is the zero eigenvector of E. γ is the zero eigenvector of L. δ is the zero eigenvector of F.

According to *Lemma 2*, $\alpha_1 = \alpha \otimes \beta$, $\alpha_2 = \gamma \otimes \delta$ .Since matrix E and matrix L are singular, and are singular, $\therefore \exists \alpha_1, \alpha_2$ are nonzero. Because α can be arbitrary, let $\alpha = \gamma$. When $\beta = \delta, \alpha_1 = \alpha_2$, let $\alpha_1 = \alpha_2 = \alpha \neq 0$,

$$\therefore [(I_N \otimes E) - \lambda (L \otimes F)]a = 0, a \neq 0 \Rightarrow (I_N \otimes E) - \lambda (L \otimes F) \tag{6}$$

is singular. The proposition is proved.

Arbitrary tests and trials show that in most cases matrix (5) is singular. If matrix (5) is mounted a surplus numerically small nonsingular matrix, it will take the following form:

$$(I_N \otimes E) - \lambda (L \otimes F + \varepsilon \cdot I_N) \tag{7}$$

where ε is a small value.

Since matrix (5) is mounted to be a nonsingular matrix, the matrix (7) would be nonsingular. For final results, the influence of this change can be ignored, because the numerical solution of DAE is relative to the stability of equation parameters. Meanwhile, ε is arbitrarily small. The change of equation parameters is actually small when ε is small enough. In addition, for numerical solution is approximate solution itself, the impact of this change is negligible. So the limitation of index 1 can be solved in this way.

Computational examples and numerical solutions

In order to illustrate the feasibility and effectiveness of the algorithm, a numerical instance will be exhibited to illustrate the application of ode15s in this section.

This system contains five subsystems; the order of each subsystem is three. In order to obtain the odefun, this system will be extended to fifteen subsystems. The network topologies are shown in Fig.1 with,

$$W = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 2 & 0 & 1 & 2 & 0 \\ 0 & 1 & 0 & 2 & 2 \\ 2 & 2 & 0 & 0 & 2 \\ 1 & 2 & 2 & 2 & 0 \end{bmatrix}$$

It has spanning trees.

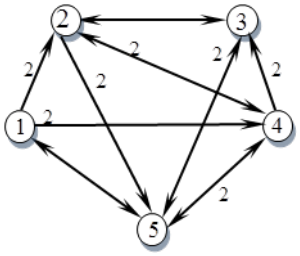

Fig.1: Network topologies. 1 is default weight of arc.

And then the Laplacian matrix is $L = diag(W * [1 \quad 1 \quad 1 \quad 1 \quad 1]') - W$.

$$L=\begin{bmatrix} 1 & 0 & 0 & 0 & -1 \\ -2 & 5 & -1 & -2 & 0 \\ 0 & -1 & 5 & -2 & -2 \\ -2 & -2 & 0 & 6 & -2 \\ -1 & -2 & -2 & -2 & 7 \end{bmatrix}$$

Suppose that

$$E=\begin{bmatrix} 2 & 2 & 0 \\ -1 & -2 & 1 \\ 1 & 3 & -2 \end{bmatrix} \quad \text{and} \quad F=\begin{bmatrix} 2 & 2 & 2 \\ 0 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

Next, work out mass matrix M, $M = kron((eye5), E)$. Since matrix E is singular, matrix M is singular.

$$M=\begin{bmatrix}
2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 3 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 3 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 3 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 3 & -2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 3 & -2
\end{bmatrix}$$

The coefficient matrix $A = kron(-L, F)$. Matrix A is singular.

$$A=\begin{bmatrix}
-2 & -2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 \\
0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & -1 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\
4 & 4 & 4 & -10 & -10 & -10 & 2 & 2 & 2 & 4 & 4 & 4 & 0 & 0 & 0 \\
0 & 0 & 2 & 0 & 0 & -5 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 \\
0 & 2 & 4 & 0 & -5 & -10 & 0 & 1 & 2 & 0 & 2 & 4 & 0 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 & 2 & -10 & -10 & -10 & 4 & 4 & 4 & 4 & 4 & 4 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -5 & 0 & 0 & 2 & 0 & 0 & 2 \\
0 & 0 & 0 & 0 & 1 & 2 & 0 & -5 & -10 & 0 & 2 & 4 & 0 & 2 & 4 \\
4 & 4 & 4 & 4 & 4 & 4 & 0 & 0 & 0 & -12 & -12 & -12 & 4 & 4 & 4 \\
0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & -6 & 0 & 0 & 2 \\
0 & 2 & 4 & 0 & 2 & 4 & 0 & 0 & 0 & 0 & -6 & -12 & 0 & 2 & 4 \\
2 & 2 & 2 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & -14 & -14 & -14 \\
0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & -7 \\
0 & 1 & 2 & 0 & 2 & 4 & 0 & 2 & 4 & 0 & 2 & 4 & 0 & -7 & -14
\end{bmatrix}$$

Setting $option = odeset('Mass','M','MassSingular','yes')$, then ode15s can be used to solve this system. The simulation result is shown in Fig. 2 with thick dots as starting points and

$$X(0)=\begin{bmatrix} 3.6869 & -2.4013 & 4.1065 & -3.5446 & 0.7970 \\ -4.1556 & 3.0007 & -3.1815 & -3.6393 & 0.4986 \\ -1.0022 & -0.6859 & -2.3620 & 3.6929 & -3.5505 \end{bmatrix}.$$

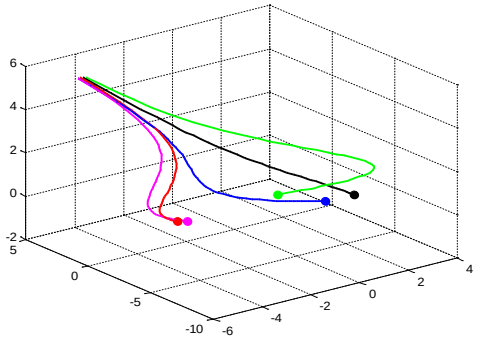

Fig.2: Trajectories of a Compartmental Network

Conclusions

This paper describes a specific type of dynamical multiagent system, and proposes a method to solve the problems of this system with Matlab. Since the model is described by DAEs, the ode15s command should be used to solve it. Many problems might be encountered when handling a DAE.

A method is provided to solve the problem of dealing a DAE that may not be of index 1. The limitation of index 1 can be removed by mounting a surplus numerically small nonsingular matrix. The influence of such a surplus small nonsingular matrix can be ignored in final results. The numerical example reflects the effectiveness of our method when dealing DAEs..

Acknowledgements

The research work is supported by National Natural Science Foundation (NNSF) of China (grants 61203189, 61174067, 61263002, & 61374054).

References

[1] Zazworsky R.M. & Knudsen H.K., "Controllability and observability of linear time-invariant compartmental models", IEEE Trans. Autom. Control, vol. 23, no. 5, pp. 872-877, 1978.
[2] Hopfield J.J., "Neural networks and physical systems with emergent collective computational abilities", Proc. Natl. Acad. Sci. USA, vol. 79, no. 8, pp. 2554-2558, 1982.
[3] Fasolo E., Rossi M., and Widmer J. et al., "In-network aggregation techniques for wireless sensor networks: a survey", Wirel. Commun. IEEE, vol. 14, no. 2, pp. 70-87, 2007.
[4] Duan G., Analysis and Design of Descriptor Linear Systems, New York: Springer-Verlag, 2010.
[5] Khalil H.K., Nonlinear Systems (3rd Edition), Upper Saddle River: Prentice Hall, 2002.
[6] Xi J., Shi Z., & Zhong Y., "Admissible consensus and consensualization of high order linear time-invariant singular swarm systems", Physica A, vol. 391, pp. 5839-5849, 2012.
[7] Xi J., Meng F., & Shi Z. et al., "Delay-dependent admissible consensualization for singular time-delayed swarm systems", Syst. Control Lett., vol. 61, no.11, pp. 1089-1096, 2012.
[8] Yang X. & Liu G., "Necessary and sufficient consensus conditions of descriptor multi-agent systems", IEEE Trans. Circuit Syst.-I, vol. 59, no. 11, pp. 2669-2677, 2012.
[9] Zhou L., Ho D.W.C., & Zhai G., "Stability analysis of switched linear singular systems", Automatica, vol. 49, no. 5, pp. 1481-1487, 2013.
[10] Karim Alloula, Fabien Monfreda, & Raphaële Thery Hétreux et al., "Converting DAE Models to ODE Models: Application to Reactive Rayleigh Distillation", Chemical engineering transactions, vol. 32, pp. 1315-1320, 2013.
[11] Li W., "Stability analysis of swarms with general topology", IEEE Trans. Syst. Man Cybernet.-B: Cybernet., vol. 38, no. 4, pp. 1084-1097, 2008.
[12] Horn R.A. & Johnson C.R., Matrix Analysis, Cambridge: Cambridge University Press, 1985.
[13] Shampine L. F., Lawrence F., & Reichelt et al., "Solving Index-1 DAEs in MATLAB and Simulink", SIAM Review, vol. 41, No. 3, pp. 538-552, 1997.