Numerical solution of Sylvester matrix equations: Application to dynamical systems

Shukooh Sadat Asari1*, Majid Amirfakhrian1

(1) Department of Mathematics, Central Tehran Branch, Islamic Azad University, PO. Code 14168-94351, Tehran, Iran.

Copyright 2016 © Shukooh Sadat Asari and Majid Amirfakhrian. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract

Many problems of control theory specially dynamical system lead to Sylvester equations. In this paper, we employ an iterative method of optimization based on partial swarm theory to solve the Sylvester system. To this purpose we consider dynamical system with different construction of state observer which lead to Sylvester observer equation. Using Pso to optimize the solution, obtain the solution with high accuracy comparison with other numerical methods, since the stability analysis of particle dynamics of PSO associated with the best particle is based on nonlinear feedback systems. Finally, some examples demonstrate the efficiency of the proposed method.

Keywords: Control theory, Dynamical system, Particle Swarm Theory, Sylvester observer equation.

1 Introduction

Control theories consist classical control theory that is known as conventional control theory, and also modern and robust control theories. Today, automatic control plays very essential role in various field of engineering and science. In other words, the importance of automatic control is so much so that can’t be ignored in different control systems such as: space-vehicle systems, robotic systems, modern manufacturing systems, and any industrial operations involving control of temperature, pressure, humidity, flow, etc. The idea of designing control system is to perform specific function under some specifications which must be given before the design process begins. Specifications can be defined differently, for example, in terms of steady-state requirements or frequency-response terms. Generally, to design problems, some routine terms such as accuracy, stability and speed of response, may be given as precise numerical values which need to be modified during the process, since the given specifications may never be satisfied (because of conflicting requirements) or may lead to a very expensive system. If a control system be insensitive to differences between the real system and the mathematical model of the system it would be robust. A mathematical model of a dynamic system would be in the form of differential equations [1]. These equations have been derived form physical laws which relate to a particular system such as Newton’s laws. If the state variables are not
measured, they should estimate by some estimators. To solve this kind of equations may some indirect methods like observer in which system leads to Sylvester matrix. The Sylvester matrix equation is generally of the form of

$$AX + XB = C$$  \hspace{1cm} (1.1)

where $A = (aij) \in \mathbb{R}^{n \times m}$ and $B = (bij) \in \mathbb{R}^{m \times n}$, $C = (cij) \in \mathbb{R}^{p \times q}$ and $X \in \mathbb{R}^{m \times n}$ is unknown. The application of this matrix equation is in control theory, filtering, model reduction, image restoration and etc. [12, 13, 14, 15] and the references therein. The Hessenberg-Schur method and the Bartels-Stewart method which are based on transforming the coefficient matrices, are two standard solution methods for Sylvester equations of the form (1.1) [16, 17]. Several iterative schemes to solve Sylvester equations have also been proposed, for methods focusing on large sparse systems [18, 19, 20]. In this paper, we consider an iterative method of optimization to obtain the best approximate of state vector $x(t)$. The mentioned method is based on Particle Swarm Optimization (PSO), which is Section 3. Particle Swarm optimization (PSO), has recently attracted by many researches, since it’s applicable in various problems such as classification [22], fuzzy modeling [21], fuzzy control [23], power system optimization [24] and etc. PSO algorithm which is constructed from bird flocking and fish schooling, is based on creating several candidate solutions (particles) for the multidimensional optimization function. Then the velocity and position of each particle will be updated in every iteration [25]. In next section, we introduce simulation of dynamical system and some preliminaries related to observer estimator is brought. In Section 3, Sylvester matrix which is obtain from observer and application of PSO to Sylvester equation are discussed. Finally, the robustness and efficiency of the method is illustrated by some examples.

2 Preliminaries and notations

Sylvester equation, known as the Sylvester-observer equation, arises in the construction of observers and in solutions of the eigenvalue assignment (or pole placement) problems [2]. To this purpose, we first describe some properties of observer for a dynamical system and in the next section, observer method and Sylvester matrix would be analyzed. Simulation model of a dynamical system rely on a truth model of the dynamics which includes relevant characteristics to be controlled in process. A general form of a dynamical system is the finite set of ordinary differential equations of the following form:

$$\begin{align*}
\dot{x}(t) &= f(t, x(t), u(t)), \\
y(t) &= h(t, x(t), u(t))
\end{align*}$$  \hspace{1cm} (2.2)

where the state $x \in \mathbb{R}^n$, the input $u \in \mathbb{R}^m$, the output $y \in \mathbb{R}^p$ and $f, h$ are vector-valued functions with $f: \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ and $h: \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^p$.

Nonlinear modeling usually lead to complex problems. To this end, the linear time invariant (LTI) dynamical modeling is often adopted for control and observation purposes.

**Definition 2.1.** (Linear Time-Invariant (LTI) systems): Continuous time state-space model of a LTI system is as follows:

$$\begin{align*}
\dot{x}(t) &= Ax(t) + Bu(t), \\
y(t) &= Cx(t) + Du(t)
\end{align*}$$  \hspace{1cm} (2.3)

where $x(t_0)$ is the initial condition, $x(t) \in \mathbb{R}^n$ is system state vector, $u(t) \in \mathbb{R}^m$ is system input vector, $x(t) \in \mathbb{R}^n$ is system state vector, $m \leq n$ and $y(t) \in \mathbb{R}^p$ is system output. And also the matrices $(A)_{n \times n}, (B)_{n \times m}, (C)_{p \times n}$, and $(D)_{p \times m}$ are time-invariant matrices.
Definition 2.2 (Controllability [2]):
The system (2.3) is said to be controllable, if starting from any initial state \( x(0) \); the system can be driven to any final state \( x_1 = x(t_1) \) in some finite time \( t_1 \), choosing the input variable \( u(t) \) for \( t \in [0,1] \) appropriately.

Theorem 2.1. Let \( A \in \mathbb{R}^{n \times n} \) and \( B \in \mathbb{R}^{m \times n} \). The following are equivalent:

(a) The system (2.3) is controllable.
(b) The eigenvalues of \( A - BK \) can be arbitrarily assigned (assuming that the complex eigenvalues occur in conjugate pairs) by a suitable choice of \( K \).

Proof. The proof is in [2].

Definition 2.3 (Observability [2]):
The Eq. (2.3) is said to be observable if there exists \( t_1 > 0 \) such that the initial state \( x(0) \) can be uniquely determined from the knowledge of \( u(t) \) and \( y(t) \) all for \( t \in [0,1] \).

3 Main section

3.1. Observer design
Consider the following LTI:
\[
\dot{x}(t) = Ax(t) + Bu(t), \quad x(t_0) = x_0
\]
\[
y(t) = Cx(t)
\]
We desire to approximate a state vector \( x(t) \) by \( \hat{x}(t) \) such that \( e(t) = x(t) - \hat{x}(t) \) approaches zero. In other words, we want to minimize \( e(t) \).

Theorem 3.1. ([2]) Let \( y(t), u(t) \) and the matrices \( A, B, C \) satisfy in (2.3). If \( (A,B) \) be observable, then the states \( x(t) \) can be estimated by
\[
\dot{\hat{x}} = A\hat{x} + Bu + K[y - \hat{y}]
\]
where \((K)_{n \times p}\) and \( p \) is the number of outputs. So \( \dot{\hat{x}} = A\hat{x} + Bu + K[Cx - \hat{C}\hat{x}] \). Then we would have
\[
\dot{e} = \dot{x} - \dot{\hat{x}} = Ax - Bu - A\hat{x} - Bu - K[Cx - \hat{C}\hat{x}]
\]
\[
= (A - KC)x - (A - KC)\hat{x} = (A - KC)e.
\]
Analytical solution of differential equation (3.5), is \( e(t) = e^{(A - KC)t}e(0) \). It shows the eigenvalues of the matrix \((A - KC)\) can be controlled vector \( e(t) \) such that it leads to zero. For example, if \( Re\lambda(A - KC) < -\alpha \), then \( e(t) \) will lead to zero faster than \( e^{-\alpha t}e(0) \). Luenberger [3], proposed different construction of state observer which obtained another dynamical system as follows;
\[
\dot{\hat{z}}(t) = H\hat{z}(t) + Fy(t) + Gu(t)
\]
If \((A,C)\) is observable and \((H,F)\) is controllable, then a unique full-rank solution of the Sylvester observer equation
\[
HX - XA + FC = 0
\]
exists and with \( G := XB \), the solution \( z(t) \) of (3.7) is a state observer for any initial values \( x_0, z(0) \), and any input function \( u(t) \).

The matrix equation \( AX - XB = C \) is known as the Sylvester equation where \( A \in \mathbb{R}^{m \times n} \), \( B \in \mathbb{R}^{m \times n} \) and \( X, C \in \mathbb{R}^{m \times n} \).
Theorem 3.1. The system (3.7) is a state-observer of the system (3.4); that is, $z(t)$ is an estimate of $Xx(t)$ in the sense that the error $e(t) = z(t) - Xx(t)$ → 0 as $t → ∞$ 1 for any initial conditions $x(0)$, $z(0)$ and $u(t)$ if

1. $XA - FX = GC$,
2. $P = XB$
3. $F$ is stable.

Sylvester observer equation is a linear equation for $X$, and can be written as a large linear system of the standard form for linear equations:

$$Gx = c$$

for an $mn \times mn$ matrix

$$G = I_n \otimes A - B^T \otimes I_m$$

where $\otimes$ denotes the Kronecker product, and $x$ and $c$ are vectors in $\mathbb{R}^{mn}$ whose components are the entries of successive rows of the matrices $X$ and $C$ respectively [11]. Equation $AX - XB = C$ has a unique solution if and only if the matrices $A$ and $B$ have no common eigenvalues.

3.2. Particle swarm optimization (PSO)

PSO is considered by Kennedy and Eberhart [4], based on the simulation of a simplified model. PSO has a population with random positions, each of these particles has a velocity, and the particles "fly" around the search space [5]. Fundation of developing of the PSO algorithm is considered in [6]. In PSO, positions of companions (particles) are considered as solutions of the d-dimensional problem and move of the particles in the way of finding food is the process of obtaining the best solution [7, 8, 9]. The position and velocity of each particle is defined by $x = (x_1, \ldots, x_d)$ and $v = (v_1, \ldots, v_d)$, respectively. Acceleration of particles store their best position in their memory (pbest) which in personal particle it is shown by, $P_i = (p_{i1}, \ldots, p_{id})$ and the best global position (gbest), which for each particle is shown by, $P_g = (p_{g1}, \ldots, p_{gd})$, weighted by an acceleration factor in each time step. Mathematical model of a particles is as follows:

$$v_{id}(new) = w_{id}v_{id}(old) + c_1\eta_i(p_{id} - x_{id}) + c_2\eta_d(p_{gd} - x_{id})$$

$$x_{id}(new) = x_{id}(old) + \mu v_{id}(new)$$

where $w$ is an weight which is applied to balance global search in problem and would be updated by

$$w = w_{initial} - m \times \text{iter.}$$

where $w_{initial}$ is the initial inertia weight and $m$ the slope of inertia weight variation. , $c_1$ and $c_2$ in Eq. (3.10), are learning factors which are positive constants and usually are defined experimentally. $\eta_1, \eta_2 \in [0,1]$ are selected randomly. A time parameter $\mu$ in Eq. (3.11) determines the different flying time for each particle. In this study, random numbers uniformly distributed in $[0,1]$ are used as the time parameter by experience. Eq. (3.10) is used to update the velocity according to its previous velocity and the distances of its current position from $p_i$ and $p_g$. Then, the particle flies toward a new position according to Eq.(3.11).

Such an adjustment of the particle’s movement through the space causes it to search around the two best positions [10].

Eq. (3.10) is used to update each particle’s speed. The second factor is composed of a cognitive part, which is based on the difference between the actual position of the particle and the best position it has achieved in history (pbest). The last factor is composed of a social component, whose calculation is based on the particle’s actual position and the best position achieved by any particle in the algorithm’s execution (gbest). Eq. (3.11) represents the position update of a particle, according to its previous position and its actual speed, considering to $\mu = 1$, [6].
3.3. Optimization by Particle Swarm Theory

Sylvester-observer equation arises in construction of an observer and is a matrix equation is a variation of the classical Sylvester equation which has the following form:

\[ AX + TX = R \] (3.13)

where \( A, T \) and \( R \) are given and \( X \) is unknown matrix. The main objective of this study is minimizing the \( |AX + TX - R| \). A first stage is given by the random assignment of a swarm of user defined integers. This space is bounded by \( X_{d_{min}} \) and \( X_{d_{max}} \). Thus, the position of each particle is initialized as follows:

\[ X_i^d = \left( X_{d_{min}} + \text{Rand} \times (X_{d_{max}} - X_{d_{min}}) \right) \] (3.14)

where Rand is a random generated number uniformly distributed between 0 and 1 and where the process must be carried out for each dimension [26]. Nevertheless, discritization of the domain guarantees that the position of each particle relates to an integer. Therefore, Eq. (3.14), is rounded to the nearest integer, resulting in a set of particles whose initial position is composed of integer number in all dimensions

\[ X_i^d = \text{Round}\left( X_{d_{min}} + \text{Rand} \times (X_{d_{max}} - X_{d_{min}}) \right) \] (3.15)

In Eq.(3.10), the first iteration, initial speed can be chosen as \( v_{id} = \forall \{0,0\} \) and weight is set to be 0.5. Thereafter, algorithm evaluates the objective function, \( |AX + TX - R| \), by replacing the position of each particle in the objective function. The system does not have an equilibrium point if \( p_{id} \neq p_{gd} \) and the equilibrium point is exist just for the best particle or candidate solution. Then the velocity and position of each particle would be reevaluated till the best particle obtain. This procedure is iterative and is repeated until the convergence criteria are met, or until all solutions in the search domain are found.

4 Numerical examples

We apply our method for two Sylvester-observer equations and. For each example, the computed values of the residuals are plotted over a number of iterations.

Example 4.1. Consider the Sylvester-observer equation of the design of a reduced-order observer for the Helicopter problem as the following form:

\[ AX + X^T B = C \] (4.16)

where

\[
A = \begin{bmatrix}
0.9268 & 0.3739 & 0.5080 \\
0.3157 & 0.1542 & 0.4521 \\
0.3271 & 0.3044 & 0.3816
\end{bmatrix}, \quad B = \begin{bmatrix}
0.1834 & 0.5337 & 0.9326 \\
0.1499 & 0.8615 & 0.0326 \\
0.9278 & 0.1793 & 0.0036
\end{bmatrix},
\]

\[
C = \begin{bmatrix}
-0.8494 & 0.5538 & 1.9573 \\
0.6707 & 0.4651 & 1.5172 \\
0.6024 & 0.2558 & 1.3079
\end{bmatrix}
\]

The objection function is

\[ G = |AX + X^T B - C| \] (4.18)

The main aim is minimizing \( G \) and get the optimized solution \( X \) by using PSO when the initial inertia weight is, \( W = 0.0004 \), \( c_1 = 1.2 \) and \( c_2 = 0.012 \). Figure 1 shows the residuals and execution times which is obtained by employing PSO and also it is shown that after 3000 iteration in 27 second which is measured...
by CPU time we obtain the solution of system $X$ with accuracy of $F(x) = 7.851537e - 3$. Obtained solution of the problem by PSO and the exact solution is shown in Table 1.

Figure 1: (a) Performance measured by CPU time and (b) residuals for solving a Sylvester equation using PSO for Example 4.1.

Table 1: Optimized solution and the exact one for Example 4.1.

<table>
<thead>
<tr>
<th>$X$</th>
<th>$x_{11}$</th>
<th>$x_{12}$</th>
<th>$x_{13}$</th>
<th>$x_{21}$</th>
<th>$x_{22}$</th>
<th>$x_{23}$</th>
<th>$x_{31}$</th>
<th>$x_{32}$</th>
<th>$x_{33}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_{Exact}$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$X_{PSO}$</td>
<td>1.00</td>
<td>9.99 $e^{-1}$</td>
<td>9.98 $e^{-1}$</td>
<td>-9.87 $e^{-1}$</td>
<td>-1.00</td>
<td>9.74 $e^{-1}$</td>
<td>-9.99 $e^{-1}$</td>
<td>9.88 $e^{-1}$</td>
<td>9.93 $e^{-1}$</td>
</tr>
</tbody>
</table>

Example 4.2. As another example consider the following Sylvester-observer equation:

$$AX + X^TB = C$$

where

$$A = \begin{bmatrix} 0.1476 & 0.6364 & 0.2567 \\ 0.8492 & 0.5904 & 0.6943 \\ 0.9883 & 0.1258 & 0.9476 \end{bmatrix}, \quad B = \begin{bmatrix} 0.4434 & 0.2236 & 0.3336 \\ 0.4588 & 0.1729 & 0.0788 \\ 0.2129 & 0.8514 & 0.0130 \end{bmatrix}$$

$$C = \begin{bmatrix} -0.9795 & -1.0333 & 1.2819 \\ -0.2316 & 1.8553 & 2.4017 \\ 1.0427 & 3.0520 & 2.4870 \end{bmatrix}$$

The objection function is

$$G = \|AX + X^TB - C\|$$

The same as Example 4.1. we consider, $W = 0.0004$, $c_1 = 1.2$ and $c_2 = 0.012$. Residuals and execution times is shown in Figure 2. and after 3000 iteration in 27 second, the best solution obtain with accuracy of $F(x) = 4.899921e - 2$. Obtained solution of the problem by PSO and the exact solution is shown in Table 2.
Figure 2: (a) Performance measured by CPU time and (b) residuals for solving a Sylvester equation using PSO for Example 4.2.

Table 2: Optimized solution and the exact one for Example 4.2.

<table>
<thead>
<tr>
<th></th>
<th>$x_{11}$</th>
<th>$x_{12}$</th>
<th>$x_{13}$</th>
<th>$x_{21}$</th>
<th>$x_{22}$</th>
<th>$x_{23}$</th>
<th>$x_{31}$</th>
<th>$x_{32}$</th>
<th>$x_{33}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_{\text{Exact}}$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$X_{\text{PSO}}$</td>
<td>1.01</td>
<td>9.51 $e^{-1}$</td>
<td>1.05</td>
<td>$9.91 e^{-1}$</td>
<td>-1.05</td>
<td>-1.01</td>
<td>-1.01</td>
<td>1.07</td>
<td>9.72 $e^{-1}$</td>
</tr>
</tbody>
</table>

5 Conclusion

In this study, we proposed a successful implementation of an optimization based on particle swarm theory to tune the solution of Sylvester equation. Although this algorithm was unsupervised, the solution resulted from the method has sufficient accuracy in the space of problem. So far, different numerical methods applied to solve Sylvester equation [2], however they were cost computationally or in comparison with PSO they didn’t result a solution with sufficient accuracy. Our proposed method has the privilege of defining some parameters that are capability for updating and this characteristic, enable us to obtain the best solution. we illustrate the efficiency and robustness of the method by two examples.

References

http://dx.doi.org/10.1137/0301010


http://dx.doi.org/10.1109/TME.1964.4323124

http://dx.doi.org/10.1109/icnn.1995.488968

http://dx.doi.org/10.1016/j.chaos.2006.05.077

http://dx.doi.org/10.1109/TEVC.2010.2053935


http://dx.doi.org/10.1016/j.aca.2006.12.004


http://dx.doi.org/10.1145/355780.355784

http://dx.doi.org/10.1007/BF01932775

http://dx.doi.org/10.1137/0725061


http://dx.doi.org/10.1145/355780.355784

http://dx.doi.org/10.1145/361573.361582

http://dx.doi.org/10.1137/S0895479894273687
http://dx.doi.org/10.1016/0024-3795(92)90031-5

http://dx.doi.org/10.1016/0893-9659(88)90183-8

http://dx.doi.org/10.1016/j.measurement.2006.09.004

http://dx.doi.org/10.1016/j.eswa.2006.08.006

http://dx.doi.org/10.1109/TIE.2005.858737

http://dx.doi.org/10.1016/j.eswa.2008.06.027

http://dx.doi.org/10.1109/4235.985692

http://dx.doi.org/10.1016/j.amc.2013.10.007